COMBINING NEURAL AND SYMBOLIC REPRESENTATIONS IN NATURAL LANGUAGE PROCESSING

Matej Martinc

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Matej Martinc

COMBINING NEURAL AND SYMBOLIC REPRESENTATIONS IN NATURAL LANGUAGE PROCESSING

Doctoral Dissertation

KOMBINIRANJE NEVRONSKIH IN SIMBOLNIH REPREZENTACIJ ZA PROCESIRANJE NARAVNEGA JEZIKA

Doktorska disertacija

Supervisor: Asst. Prof. Dr. Senja Pollak

Ljubljana, Slovenia, March 2022

Acknowledgments

Without the help of many people who supported and helped me during my studies, this thesis would have never been finished. First, I would like to thank my supervisor Senja Pollak, whose never-ending flow of research ideas and enthusiasm for research in the field of natural language processing was extremely helpful and contagious, and consequently played a major part in my research. I am also very grateful to Nada Lavrač for encouraging me to pursue a PhD and a career in academia in the first place, and for showing me that science can be awesome. Big thanks goes also to the members of the thesis evaluation board, Marko Robnik-Šikonja, Matthew Purver and Antoine Doucet, for providing helpful comments.

The institutions that financially supported my research also deserve gratitude: the Department of Knowledge Technologies at the Jožef Stefan Institute and the Slovenian Research Agency ARRS that funded national projects TermFrame, CANDAS and Formica 2 in which I was involved. Finally, this research was also supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

I would also like to thank everyone, with whom I have collaborated and co-written a number of research papers, especially Blaž Škrlj for the work we did together on the TNT-KID and Tax2Vec methodologies; Marko Robnik-Šikonja for the idea of employing neural language models for text readability prediction; Petra Kralj Novak for a number of extremely insightful conversations about computer science and statistics fundamentals; Mili Bauer and Tina Anžič for taking care of all the administrative work that came with my research and for teaching me how to deal with bureaucracy.

My family and friends also deserve a big thank you for giving me a much needed moral boost when things looked dire and for constantly nagging me with the question of when will I finish my PhD. Finally, thank you COVID-19 for the role you played in shutting down travel and social interactions, which allowed me to focus entirely on work and the writing of my thesis. Without you, it would have taken much longer.

Abstract

The thesis addresses a novel representation learning framework, combining neural and symbolic text representations, and demonstrates its utility for tackling diverse natural language processing problems. The proposed approach, avoiding the deficiencies of purely symbolic and purely neural methods, can be applied for the generation of efficient text representations. Its usefulness is demonstrated on three use cases: author profiling, readability detection and keyword extraction.

First, we focus on the problem of author profiling and argue that semantic modelling of existing state-of-the-art approaches, which still in most cases rely on extensive feature engineering, could be improved by employing two strategies. The first one involves adding symbolic semantic features based on word taxonomies to bag-of-n-grams features. This approach shows good results when tested on a number of author profiling tasks, that is, predicting the gender, age and personality of the author of the text. The second strategy consists of combining the bag-of-n-grams features with neural features derived from the convolutional neural network and is tested on the task of language variety detection. While both approaches manage to outperform state-of-the-art methods, we argue that the second hybrid neuro-symbolic strategy is preferred since it does not require external resources and is therefore easier to employ on less resourced languages other than English.

We next shift our focus to the problem of readability prediction, where we propose a novel Ranked Sentence Readability Score, in which statistics derived from the neural language model are combined with shallow symbolic readability indicators that consider simple text statistics. The main novelty of the approach is the use of a neural language model in an unsupervised way, as a standalone unsupervised readability predictor. We argue this is possible since neural language models tend to capture much more information than traditional n-gram models and also model long-term dependencies. Through language model statistics, the proposed readability formula also considers background knowledge and discourse cohesion and therefore avoids the reductionism of traditional readability formulas. And since neural language models can be trained, the formula can also be adapted to a specific language and domain. We show that this results in a robust performance of the formula, which offers good correlation with gold standard readability scores across different genres and languages.

The final task we tackle is keyword extraction. We propose a transfer learning technique, in which a transformer-based keyword tagger is first pretrained as a language model on a large corpus and then fine-tuned on a small-sized corpus with manually labelled keywords in order to decrease the amount of required labelled data for successful training of the model. We propose several modifications to the transformer architecture in order to adapt it to task at hand and improve performance. We show that the proposed model offers performance comparable to the state-of-the-art neural models while requiring only a fraction of manually labelled data. Finally, we combine the neural model with a symbolic unsupervised TF-IDF-based keyword detector in order to improve the recall and make the system appropriate for usage as a recommendation system in the media house environment.

Povzetek

Disertacija predstavi novo strategijo kombiniranja nevronskih in simbolnih reprezentacij, s katero želimo preseči omejitve pristopov, ki temeljijo le na eni vrsti reprezentacij. S pomočjo predlaganega pristopa nam uspe razviti množico novih metod in tekstovnih reprezentacij za reševanje nalog s področja procesiranja naravnega jezika. Uporabnost strategije je prikazana na treh primerih, profiliranju avtorjev, detekciji berljivosti teksta in luščenju ključnih besed.

Najprej se posvetimo problemu profiliranja avtorjev besedil in postavimo tezo, da se da obstoječe pristope, ki v veliki meri še vedno temeljijo na ročni izdelavi značilk, izboljšati s pomočjo dveh metod. Prva metoda vključuje dodajanje simbolnih značilk, ki temeljijo na besednih taksonomijah, tradicionalnim značilkam, ki temeljijo na pristopu vreče n-gramov. Pristop je preizkušen na treh nalogah profiliranja avtorjev (določanje spola, starosti in osebnosti avtorjev besedila) in nudi dobre rezultate. Druga metoda temelji na kombiniranju značilk, ki temeljijo na pristopu vreče n-gramov, z nevronskimi značilkami, zgeneriranimi s pomočjo konvolucijske nevronske mreže, in je preizkušena na nalogi zaznavanja jezikovnih različic in dialektov. Medtem ko obe metodi izboljšata modeliranje semantike in nudita boljše rezultate kot ostale najsodobnejše metode, se v nadaljevanju disertacije osredotočimo le na drugo, saj za razliko od prve ne zahteva zunanjih jezikovnih virov in jo je zato lažje uporabiti v jezikih z manj jezikovnimi viri.

Nato se osredotočimo na problem določanja berljivosti teksta, pri čemer predlagamo novo mero z imenom Ranked Sentence Readability Score, v kateri so statistične značilke, pridobljene s pomočjo nevronskega jezikovnega modela, združene s plitkimi simbolnimi kazalniki berljivosti. Glavna novost pristopa je uporaba nevronskega jezikovnega modela na nenadzorovan način. Predlagana formula za berljivost s pomočjo statistik, ki jih pridobi iz jezikovnega modela, upošteva tudi semantiko in kohezivnost teksta ter se tako izogne redukcionizmu tradicionalnih formul za določanje berljivosti. Z eksperimenti pokažemo, da formula nudi dobre rezultate na množici korpusov, ki vsebujejo tekste iz različnih jezikov in žanrov. Dodatna prednost pristopa je, da je predlagano mero berljivosti mogoče prilagoditi posameznim jezikom in žanrom, saj je mogoče nevronske jezikovne modele natrenirati na jezikovno in žanrsko specifičnih besedilih.

Zadnja predstavljena naloga je luščenje ključnih besed. Ker želimo zmanjšati količino podatkov, potrebnih za treniranje, nevronski model, ki temelji na arhitekturi transformer, natreniramo s pomočjo tehnike transfernega učenja. Pri tej tehniki se sistem najprej trenira na velikem korpusu na nenadzorovan način, kot jezikovni model, nato pa šele kot klasifikator na majhnem korpusu z ročno označenimi ključnimi besedami. Predlagamo tudi več arhitekturnih sprememb za prilagoditev modela specifični nalogi luščenja ključnih besed, ki izboljšajo njegovo delovanje. S predlaganim pristopom dosežemo rezultate, ki so primerljivi z najsodobnejšimi nevronskimi metodami, a hkrati potrebujemo veliko manj ročno označenih podatkov. Na koncu nevronski model združimo s simbolnim modelom, ki ključne besede išče s pomočjo statistike TF-IDF. Na ta način izboljšamo priklic sistema in ga prilagodimo za uporabo kot priporočilni sistem v medijskem okolju.

Contents

List of Figures

List of Tables

Abbreviations

AP	Author Profiling
ARI	Automated Readability Index
ASL	Average Sentence Length
BERT	Bidirectional Encoder Representations from Transformers
BON	Bag-of-n-grams
CNN	Convolutional Neural Network
CRF	Conditional Random Field
DCRF	Dale-Chall Readability Formula
DSL	Discriminating between Similar Languages
ELMo	Embeddings from Language Models
ESL	English as a Second Language
FKGL	Flesch-Kincaid grade level
FRE	Flesch Reading Ease
GDI	German Dialect Identification
GFI	Gunning Fog Index
GloVe	Global Vectors for Word Representation
GLUE	General Language Understanding Evaluation
HeLI	Helsinki Language Identification
ILI	Indo-Aryan Language Identification
LSTM	Long Short-Term Memory
NER	Named Entity Recognition
NLL	Negative Log Loss
NLP	Natural Language Processing
POS	Part of Speech
PPL	Perplexity
RLM	Recurrent Language Model
RNN	Recurrent Neural Network
RSRS	Ranked Sentence Readability Score
SMOG	Simple Measure of Gobbledygook
SVM	Support Vector Machines
TCB	Temporal Convolutional Network
TF-IDF	Term Frequency-Inverse Document Frequency
TTR	Type-token Ratio
WNLL	Word Negative Log-likelihood

Chapter 1

Introduction

In the introductory chapter, we first describe the problem addressed in the thesis (Section 1.1) and propose the goals we aim to achieve (Section 1.2). We present the initial hypotheses on which the thesis is based in Section 1.3. Finally, we conclude by explaining the scientific contributions the thesis offers in Section 1.4 and by presenting the structural overview in Section 1.5.

1.1 Background and Problem Definition

Recently, deep neural networks (Goodfellow et al., 2016) have shown an impressive performance on many language-related tasks. In fact, they have achieved the state-of-the-art performance in most semantic tasks where sufficient amounts of data were available (Collobert et al., 2011a; X. Zhang et al., 2015). These tasks include diverse problems, such as text classification, sequence labeling, language modelling and text generation. For example, Mikolov et al. (2011) have shown that neural language models outperform n-gram language models by a high margin on large and also relatively small (less than 1 million tokens) data sets. When it comes to sequence labeling tasks of named entity recognition (NER) and part of speech tagging (POS), all state-of-the-art approaches leverage some type of neural architecture, ranging from convolutional neural networks (CNN) used in (Baevski et al., 2019) to recurrent neural networks (RNN) used in (Bohnet et al., 2018), and transformers used in (Heinzerling & Strube, 2019). Similar applies when it comes to a set of language understanding tasks known as the General Language Understanding Evaluation (GLUE) (Wang et al., 2019) benchmark, which includes problems like sentiment analysis, question answering and textual entailment: 30 best ranked systems are neural.[1]

This boost in performance can be to a large extent explained by the fact that neural architectures have become very successful at grasping the semantics of the text and can model semantic relations much more efficiently than algorithms based on simpler word frequency statistics, such as term frequency-inverse document frequency (TF-IDF) (Chowdhury, 2010). The main advancement enabling this superiority of neural nets is due to embeddings (Mikolov et al., 2013), which became a prevalent way to build representations for texts. Text embeddings use a large corpus of documents to extract vector representations for words, sentences, and documents. The first neural word embeddings like Word2vec (Mikolov et al., 2013) produced one vector for each word, irrespective of its polysemy. These so-called static embeddings have been further developed and the most popular static embeddings currently in use, besides Word2Vec, are GloVe (Global vectors for word representation) (Pennington et al., 2014) and FastText (Bojanowski et al., 2017).

[1]https://gluebenchmark.com/leaderboard, last accessed 24.8.2021

Recent developments, like ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), take a context of a sentence into account and produce different word vectors for different contexts of each word, i.e. the so-called contextual embeddings. Another novelty of these approaches is the employment of the unsupervised language model pretraining, which has recently become a well-established procedure in the field of natural language processing (NLP). The procedure relies on pretraining of the model as a masked or autoregressive language model on very large unlabeled textual resources and the transfer of knowledge obtained by the language model onto a specific downstream task by fine-tuning the model.

On the other hand, neural approaches also have some deficiencies. First, despite language model pretraining, they can be resource demanding and require vast amounts of labeled data for successful training, which means that their usage in domains and languages with scarce manually labeled resources is limited. The lack of data affects almost all NLP tasks, from text generation, language modeling to several text classification tasks. For example, the study by Zidarn (2020), in the scope of which the first abstractive summarizer for Slovenian was created, reports that there was not enough training data available for the system to offer performance comparable to systems trained on English. The study by Ulčar and Robnik-Šikonja (2020b) that covers the development of two trilingual transformer-based language models, one for Croatian, Slovenian, and English, and the other for Finnish, Estonian, and English, claims that one of the reasons for not opting to develop monolingual models instead was the lack of data. The possible downside of training the models on several languages to compensate for the lack of data is the so-called curse of multilinguality (Conneau et al., 2020), i.e. a trade-off between the number of languages the model supports and the overall decrease in performance on monolingual and cross-lingual benchmarks. Finally, the lack of data is not problematic just for some less resourced languages but also for some less resourced domains. An example of this would be some medical domains: to train a classifier to be able to distinguish between transcripts of patient with dementia and patients from the control group, very few datasets are available and these datasets are generally too small to train neural models that would outperform more traditional models such as random forest, SVM or logistic regression (Martinc, Haider, et al., 2021). To conclude, development of neural models that would require less resources would benefit a large set of diverse NLP tasks.

Another problem posed by neural networks is interpretability and making the neural models more interpretable would once again benefit a large set of diverse tasks. While the elimination of work required for manual feature engineering is generally a desirable quality of neural networks, on the downside this tends to decrease the interpretability and explainability, which are important for some applications, e.g. for medical and health care applications (Vellido, 2020), and for automatic determination of text readability employed in education (Sheehan et al., 2014), where the users of such technology (educators, teachers, researchers, etc.) need to understand what causes one text to be judged as more readable than the other and according to which dimensions. While several techniques that improve the overall interpretability of neural networks have been proposed (e.g., SHAP (Lundberg & Lee, 2017), the analysis and visualization of the attention mechanism (Vaswani et al., 2017), etc.), the interpretability of neural networks is still not comparable to the interpretability of approaches based on symbolic feature engineering.

The research described in this thesis went on for several years and in this time we witnessed a radical shift towards the usage of neural networks. At the beginning of our work, there were still some fields of NLP research, where neural networks were scarcely used due to their somewhat uncompetitive performance or other limitations. One of them is author profiling (AP), which deals with prediction of demographics for a person based on the text she or he produced. Tasks such as age, gender and language variety prediction (automatic distinction between similar dialects or languages) are becoming increasingly popular, in part because of the marketing potential of this research. AP research communities are centered around a series of scientific events and shared tasks on digital text forensics, the two most popular being the evaluation campaign VarDial (Varieties and Dialects)[2] (Zampieri et al., 2014a), which started in 2014 and is focused on tasks related to the study of linguistic variation, and an event called PAN (Uncovering Plagiarism, Authorship, and Social Software Misuse) (http://pan.webis.de/), which first took place in 2011, and was followed by a series of shared tasks organized since 2013 (Rangel et al., 2013a), with the focus on predicting user's age and gender.

The best approaches to AP still use traditional classifiers, such as Support Vector Machines (SVM), and are based on extensive feature engineering (Rangel et al., 2017a). This fact can be clearly seen if we look at the architectures used by the teams winning the AP shared tasks in recent years, more specifically the winning approaches to the VarDial DSL (discriminating between similar languages) shared tasks and PAN AP (gender, age, personality and language variety prediction) tasks between 2014 and 2017. In fact, six out of eight winning teams used one or an ensemble of SVM classifiers and bag-of-n-grams (BON) features for classification (two other winning teams used a LIBLINEAR classifier[3] and BON features (López-Monroy et al., 2014)), and when it comes to the task of discriminating between similar languages (all VarDial DSL tasks and the PAN 2017 AP task), SVM classifiers with BON features have been used by all the winning teams (Martinc & Pollak, 2019). The best ranked system that employed a deep learning architecture until 2017 was developed by Miura et al. (2017a) and ranked fourth in the PAN 2017 AP shared task. Nevertheless, this is changing and 2019 VarDial competition (Zampieri et al., 2019) was the first in which a neural model won. We should, however, mention that four out of five best ranked systems in the competition were based on SVMs and extensive feature engineering. When it comes to the 2019 and 2020 PAN events (Rangel et al., 2020; Rangel & Rosso, 2019), all the best ranked approaches still employed manually crafted feature sets and SVM, logistic regression or random forest classifiers.

Another NLP field where neural architectures are not yet widely adopted is automatic text readability prediction. Readability is concerned with the relation between a given text and the cognitive load of a reader to comprehend it. This complex relation is influenced by many factors, such as a degree of lexical and syntactic sophistication, discourse cohesion, and background knowledge (Crossley et al., 2017). Traditional readability formulas focused only on lexical and syntactic features expressed with statistical measurements, such as word length, sentence length, and word difficulty (Davison & Kantor, 1982). While these approaches have been criticized because of their reductionism and weak statistical bases (Crossley et al., 2017), they are still widely used since no alternative for unsupervised readability prediction has yet been proposed.

When it comes to supervised readability estimation, recently some neural approaches towards readability prediction have been proposed (Filighera et al., 2019; Nadeem & Ostendorf, 2018). These types of studies are relatively scarce and state-of-the-art approaches towards supervised readability prediction still employ a large set of manually engineered features (Xia et al., 2016). Furthermore, language model features designed to measure lexical and semantic properties of text, which can be found in many of the readability studies (Petersen & Ostendorf, 2009; Schwarm & Ostendorf, 2005; Vajjala & Meurers, 2012; Xia et al., 2016), are generated with traditional n-gram language models, even though language modelling has been drastically improved with the introduction of neural language models

[2]http://corporavm.uni-koeln.de/vardial/sharedtask.html

[3]It is unclear from the paper by López-Monroy et al. (2014) whether SVM or logistic regression from the LIBLINEAR library was used.

(Mikolov et al., 2011).

For the third prominent NLP task, sequence labeling, neural networks have been widely adopted due to large performance increases offered by these systems. The achieved differences in the performance are in most cases attributed to a richer contextual information available to neural networks, which are not limited to a usually much smaller contextual window (of up to five previous words) as is the case for previous systems modelling sequential information (Mikolov et al., 2011). The problem arises when a neural system for sequence labeling needs to be employed on the less resourced language or task for which large manually labeled datasets, which are generally needed for successful training of neural systems, are not available. While for the most popular sequence labeling tasks, such as NER and POS tagging, sufficient language resources generally exist even for the under-resourced languages, some less prominent tasks are not well covered. An example is keyword identification, which deals with automatic extraction of words that represent crucial semantic aspects of the text and summarize its content. The supervised neural algorithms (Meng et al., 2019; Yuan et al., 2020) achieve excellent performance on this task if training conditions are satisfactory, that is, if vast amounts of language and domain-specific data for training are available. In practice this means that most of these neural systems cannot be used in domains and languages that lack (manually) labeled resources of sufficient size.

For these languages and domains, novel unsupervised approaches, such as RaKUn (Škrlj et al., 2019) and YAKE (Campos et al., 2020) can be used. They generally work fairly well and have advantages over supervised approaches, as they are language and genre independent, do not require any training, and are computationally undemanding. On the other hand, they also have a couple of crucial deficiencies. First, TF-IDF and graph-based features, such as PageRank, used by these systems to detect the importance of each word in the document, are based only on simple statistics like word occurrence and co-occurrence, and are therefore unable to grasp the entire semantic information of the text. Second, since these systems cannot be trained, they cannot be adapted to the specifics of the syntax, semantics, content, genre, and keyword assignment regime of a specific text (e.g., a variance in a number of keywords). These deficiencies result in a much worse performance when compared to the state-of-the-art neural supervised approaches, which have direct access to the gold standard keyword set for each text during the training phase, enabling more efficient adaptation.

1.2 Purpose and Goals of the Dissertation

As mentioned above, traditional statistical approaches towards NLP do not model semantic relations well, which makes them uncompetitive on some NLP tasks. Neural approaches tend to perform better but only when enough training data is available. One solution to improve the performance of traditional approaches is to include symbolic semantic features based on word taxonomies, for which we have shown before (Škrlj et al., 2021) that they may improve the performance and robustness of the classifiers. Another solution, which is the main goal of this dissertation, is to demonstrate that it is possible to build hybrid systems leveraging both neural and symbolic text representations.

Before diving further into the proposed strategy of combining symbolic and neural representations, to avoid confusion, we should clearly define what we mean by symbolic representations. We should warn the reader that this term is used, in contrast to some related work, to encompass a large set of representations that a.) work on the level of symbols representing entities or concepts understandable to humans, and b.) are easily interpretable by providing full transparency into how they work; therefore avoiding the "black box" problem endemic to neural networks. In some cases, the representations are indeed what the related literature would anonymously define as symbolic (e.g., the taxonomy-based features used in the tax2vec methodology (see section 2.4). Nevertheless, referring to some representations we employ in this thesis as symbolic might be confusing to some readers. An example of these would be TF-IDF weighted word/ngram features, for which one could argue that they do not represent higher-level abstract concepts (e.g. logic-based representations and rule-based reasoning (Susskind et al., 2021)). We opted to consider these features as symbolic since they directly relate to human understandable concepts and entities (represented by words), and since the bag-of-words/ngrams matrix is interpretable (i.e. there is a clear formula that explains what each weight for a specific word is suppose to mean)[4]. On the other hand, we consider word embedding representations, even though they also directly relate to words, as nonsymbolic due to limited interpretability, i.e. it is not clear what each numerical value (dimension) inside the vector is suppose to represent. On the other hand, if these vector dimensions would be clearly defined, we would also consider word embeddings symbolic. While we understand that this somewhat non-conventional use of the concept "symbolic" might be a source of some confusion, we opted to use it anyway due to lack of a more appropriate term, and since there exist a fast growing corpus of work on neuro-symbolic approaches in NLP (Q. Chen et al., 2021; Ferrone & Zanzotto, 2020; Ma et al., 2019; Škvorc et al., 2022), in which we wish to position this thesis.

By achieving synergy between the neural and symbolic representations, we aim to alleviate the aforementioned deficiencies of neural and symbolic approaches and improve the performance of the methods by improving semantic modelling. Where feasible, we also aim at improving the overall transferability and adaptability of the approaches, by either proposing techniques that can be adapted to specific languages and domains through training, or by developing models that require less data for successful training.

To achieve the above general goals, we experiment with several types of combinations between neural and symbolic text representations, which can be roughly divided into distinct types according to two criteria:

- **Fusion time**: The related literature usually distinguishes between early and late fusion (Ebersbach et al., 2017; Li, 2018). The early fusion refers to the combination of representations on the feature level, that is, before applying a prediction model. In this case, distinct features are in most cases simply concatenated and the combined representation is fed to the model responsible for the final predictions. On the other hand, late fusion (or decision-level fusion) refers to the type of combination, where prediction scores of at least two prediction models, in our case relying on distinct presentations, are combined to derive a final prediction. The most popular procedures for combining presentations at the late stage are majority voting, averaging, or taking the prediction with the highest probability (Ebersbach et al., 2017).

- **Fusion type**: While we were unable to find an exhaustive typology of fusion types in the related work, we distinguish between a simple and complex fusion. With the simple fusion we refer to straight-forward ways of combining the symbolic and neural representations, such as concatenation and averaging. Complex fusion indicates a somewhat more sophisticated procedure employed to fuse the representations. This is usually necessary when representations to be fused appear in different formats, which renders the simple fusion unfeasible. When the complex fusion is used, it is

[4]Note that this is in line with some of the related work that considers probabilistic statistical methods as symbolic, e.g. the study by De Raedt et al. (2019).

harder to disentangle the influences of distinct symbolic and neural representations on the overall performance of the model.

We experiment with different combination strategies according to the applicability of a specific fusion type to a specific use case.

The dissertation focuses on three areas of NLP research:

- Author profiling, with the focus on language variety and gender classification,
- Readability prediction, with the focus on a novel algorithm for unsupervised readability prediction,
- Keyword detection, with the focus on settings with small amounts of training data.

For these three use cases, the dissertation aims to achieve the following goals:

- *G1*: In the field of AP, the goal is to improve semantic modeling by employing two techniques. The first strategy involves inclusion of word taxonomies as background knowledge and the second strategy relies on a hybrid neuro-symbolic architecture, which combines sophisticated feature engineering techniques used in traditional approaches to text classification with the newer neural automatic feature construction.
- *G2*: For the task of readability prediction, the goal is to propose a new readability formula that offers state-of-the-art performance and can be easily adapted to specific domains and languages. The strategy involves a novel approach to unsupervised readability measurement based both on shallow readability indicators and on deep neural network-based language models that takes into account background knowledge and discourse cohesion, two readability indicators missing from the traditional readability formulas.
- *G3*: For the task of keyword detection, the goal is to develop a novel keyword extractor capable of overcoming the deficiencies of current neural supervised and symbolic unsupervised approaches. More specifically, the goal is to build a keyword extractor that requires only a fraction of manually labeled data required by current state-of-the-art neural approaches, yet is capable of achieving performance comparable to state-of-the-art. Another goal is to improve the recall of the neural keyword extraction system by combining it with an unsupervised TF-IDF-based symbolic model.

1.3 Hypotheses

The general hypothesis (H_g) that we try to confirm or deny in this research is the following:

H_g. *Combining different representations, which carry information about distinct aspects of the text, and establishing synergy between these representations, can lead to a boost in the performance of a NLP system.*

This general hypothesis can nevertheless be deconstructed into several specific hypotheses according to each specific use case that we deal with.

First, word frequency statistics, such as TF-IDF, employed in traditional NLP approaches cannot model semantic relations as effectively as neural networks. This observation opens several options for the improvement of semantic modelling. The first option is the improvement of symbolic semantic modelling by proposing novel symbolic semantic features capable of modelling information not covered by the word frequency statistics. This leads to the first specific hypothesis of this thesis:

H1. Semantic modelling can be improved by combining traditional TF-IDF BON features with purely semantic features based on word taxonomies.

While we show in Section 2.4 that the above framework of improving semantic modelling is successful, it nevertheless requires external resources (i.e., taxonomies), which are not always available. This makes the approach less transferable to less resourced languages or specialized domains. To solve this deficiency of the taxonomy-based approach, we explore other options for improvements in semantic modelling that do not require external resources.

When it comes to text classification, neural network approaches tend to have an advantage over more traditional approaches, since they can effectively model the sequential information, which is to a large extent ignored in the BON presentations. On the other hand, the main disadvantage of the neural approaches could be the lack of an effective weighting scheme that would be capable of determining how specific each word is for every input document. The data is fed into a neural network in small batches, therefore it is impossible for it to obtain a global view on the data and its structure, which is encoded in the traditional TF-IDF weighted input matrix. A model that is an effective hybrid between a traditional feature engineering approach, which relies on different kinds of BON features, and a newer neural feature engineering approach would be capable of leveraging both the sequential word/character level information and the more global document/corpus-level information. By achieving synergy between these two data flows, the model could be able to outperform current state-of-the-art systems in the field of AP. These observations lead to the second specific hypothesis:

H2. Combining neural and symbolic text representations can advance the state-of-the-art on a variety of different NLP document classification tasks.

The shallow lexical sophistication indicators (e.g., length of a sentence, average word length etc.), which are still extensively used for unsupervised readability prediction correlate well with the readability of a text. On the other hand, two readability indicators missing from the traditional readability formulas are background knowledge (i.e., semantic information) and discourse cohesion, which can be measured with the employment of neural language models. This observation leads to the third specific hypothesis of this thesis:

H3. A novel readability measure, which would include background knowledge and discourse cohesion indicators obtained from neural language models besides the standard shallow symbolic lexical sophistication indicators, offers better readability estimations than traditional readability formulas.

Supervised neural approaches towards keyword detection tend to outperform other non-neural and unsupervised approaches by a large margin, when sufficient amounts of language- and domain-specific training resources are available. These systems cannot be used in less resourced domains and languages, where training data is scarce. Therefore, for these domains and languages, symbolic unsupervised models, which offer worse performance, are generally used. This observation leads to the fourth specific hypothesis:

H4. A system for sequence labeling, which combines neural and symbolic text representations, would achieve performance comparable to state-of-the-art, while requiring only a fraction of manually labeled data required by neural approaches.

1.4 Scientific Contributions

By exploring the deficiencies of semantics in feature engineering-based models and by exploring the hypothesis that combining neural and symbolic text representation can advance the state-of-the-art on a variety of different NLP tasks, we propose new efficient methods for text mining and text representation. We also gain relevant insights into whether the proposed novel methodology can contribute to the general advancement of the NLP field. Considering the three use cases, on which we test the above hypothesis, the scientific contributions of the thesis are as follows:

- A novel method for deriving new taxonomy-based symbolic features that can model semantic relations in the text and improve the performance of traditional feature engineering-based classifiers that are still widely used in the field of AP. The novel methodology is covered in detail in Chapter 2 and was published in the following publication:

  Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., & Pollak, S. (2021). Tax2vec: Constructing interpretable features from taxonomies for short text classification. *Computer Speech & Language*, 65, 101104.

- Besides presenting a state-of-the-art approach for AP tasks, such as language variety and gender classification, which relies on extensive feature engineering, we also propose a novel methodology that relies on combining neural and symbolic representations and leads to further advancement of the state-of-the-art in the field. We evaluate how different components of the system (i.e., symbolic and neural) contribute to the overall performance of the system with an ablation study, which offers insights into advantages and disadvantages of different types of presentations. The proposed approaches are covered in Chapter 2 and were published in the following publications:

  Martinc, M., & Pollak, S. (2019). Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, 25(5), 607–632.

  Martinc, M., Škrjanec, I., Zupan, K., & Pollak, S. (2017). PAN 2017: Author profiling - gender and language variety prediction. *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum*, Dublin, Ireland, 1866. http://ceur-ws.org/Vol-1866/paper_78.pdf

- A novel method for unsupervised readability prediction allows more reliable and more robust automatic readability determination, with possible application of the method in several educational fields, such as primary and secondary school teaching, and foreign language teaching. One of the deficiencies of the existing readability formulas is that most of them were designed for the specific use on English texts. On the other hand, the proposed novel method is based on neural language models and its trainable nature allows for customisation and personalisation for specific tasks, topics and languages. The proposed methodology is covered in Chapter 3 and was published in the following publication:

  Martinc, M., Pollak, S., & Robnik-Šikonja, M. (2021). Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, 47(1), 141–179.

- A novel supervised keyword detection system that can be employed for less resourced languages and domains, where labeled resources for model training are scarce. Unlike most existing neural supervised methods for keyword extraction, the system offers a good compromise between performance and required resources for training. It also offers much better performance than unsupervised algorithms that are currently used for languages and domains without sufficient training resources. We show that due to good precision and recall, the system could also be employed as a recommendation system in news media environment, when combined with a symbolic unsupervised

TF-IDF-based keyword extractor. The system and the methodology employed are covered in detail in Chapter 4 and were published in the following publications:

Martinc, M., Škrlj, B., & Pollak, S. (2020). TNT-KID: Transformer-based neural tagger for keyword identification. *Natural Language Engineering*, 1–40.

Koloski, B., Pollak, S., Škrlj, B., & Martinc, M. (2021). Extending neural keyword extraction with TF-IDF tagset matching. *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, online, 22–29. https://aclanthology.org/2021.hackashop-1.4

1.5 Organization of the Thesis

The thesis is structured as follows. In Chapter 2, we cover several approaches proposed for AP. After briefly describing the main characteristics and tasks in Section 2.1, we present related work on the topic in Section 2.2, by covering existing approaches towards AP tasks that either employ extensive feature engineering or neural architectures. While in Section 2.3 we propose our own approach for tackling AP tasks with classifiers based on extensive feature engineering, in Section 2.4 we explain our methodology for improving semantic modelling by leveraging taxonomies. Finally, in Section 2.5 we cover our methodology for combining neural and symbolic representations and present experiments and results of the approach. Final remarks on the topic of AP are presented in Section 2.6.

In Chapter 3, we cover our work on the topic of readability detection. After the quick introduction into the field of readability in Section 3.1, we cover the related work in Section 3.2. The methodology, experiments and results of the proposed approach towards readability detection that focuses on an unsupervised method combining neural language model statistics with shallow symbolic readability indicators are presented in Section 3.3. Final remarks on the topic of readability are presented in Section 3.4.

Chapter 4 covers our research on the topic of keyword detection. We briefly present the task of keyword extraction in Section 4.1 and related work on the topic in Section 4.2. After that, we present a novel neural approach towards keyword extraction named TNT-KID (Transformer-based Neural Tagger for Keyword Identification) in Section 4.3. Finally, in Section 4.4, we explain how the proposed neural architecture can be combined with an unsupervised symbolic TF-IDF-based approach towards keyword extraction, in order to further improve the recall of the system. Final remarks on the topic of keyword extraction are presented in Section 4.5.

We conclude the thesis in Chapter 5, where we first summarize the work and the contributions in Section 5.1. We discuss the strengths and weaknesses of the proposed methods in Section 5.2. We finish the thesis by presenting the planned future work in Section 5.3 and by giving instructions on how to reproduce the experiments in Section 5.4.

Chapter 2

Author Profiling

This chapter presents two novel methodologies for document classification in the field of author profiling (AP). First, in Section 2.1, we define the problem of AP and the most common tasks in the field. Next, Section 2.2 covers how these tasks have been solved in the past, by either employing classifiers based on Support Vector Machines (SVM) leveraging symbolic TF-IDF features or by employing neural methods. In Section 2.3, we present our successful approach for tackling the author profiling task by employing extensive feature engineering and traditional classifiers. This section also contains an enclosed conference paper titled *PAN 2017: Author profiling - gender and language variety prediction* (Martinc et al., 2017). In Section 2.4, we first focus on how the feature engineering-based approach can be improved upon by including novel features based on taxonomies capable of effective semantic modelling. This section contains condensed highlights of the proposed approach and an enclosed journal publication containing the details, *Tax2vec: Constructing interpretable features from taxonomies for short text classification* (Škrlj et al., 2021). Section 2.5 presents our state-of-the-art approach of combining symbolic features with neural features. This section contains an enclosed journal paper *Combining n-grams and deep convolutional features for language variety classification* (Martinc & Pollak, 2019).

2.1 Introduction

Recently, learning about the demographics, psychological characteristics and (mental) health of a person based on the text she or he produced has become an established subfield of natural language processing (NLP). This type of research is generally referred to as author profiling (AP) and has various applications in marketing, forensics, social psychology, and medical diagnosis. The most commonly addressed tasks in AP are the prediction of an author's gender, language variety, age, native language, personality, region of origin, or mental health.

The first AP task that has received a lot of attention from the NLP community was gender prediction, which became a mainstream research topic with the influential work by Koppel et al. (2002). Based on experiments on the subset of the British National Corpus, authors found that women have a more relational writing style (e.g., using more pronouns) and men have a more informational writing style (e.g., using more determiners). Later gender prediction research remained focused on English, yet the attention shifted to social media applications (Burger et al., 2011; Plank & Hovy, 2015; Schwartz et al., 2013).

In the last few years, the focus has been extended to other AP tasks and to languages other than English (Rangel et al., 2015; Rangel et al., 2016a) and the AP community has grown in size significantly. Another task that gained prominence is language variety identification, a task of classifying different varieties of the same language by determining lexical and semantic variations between them (Franco-Salvador et al., 2015). First studies performed classification on newspaper corpora, e.g., in Portuguese (Zampieri & Gebre, 2012) and Spanish (Zampieri et al., 2013). After that, social media became another popular resource for this task, e.g., Spanish Twitter messages (Maier & Gómez-Rodríguez, 2014) or online comments in Arabic (Tillmann et al., 2014; Zaidan & Callison-Burch, 2014).

In the last few years, a series of scientific events and shared tasks on the topic of digital text forensics have been organized within the AP community. The first prominent event is the evaluation campaign VarDial (Varieties and Dialects)[1] (Zampieri et al., 2014b), which focuses on the study of linguistic variation. The second is an event called PAN (Uncovering Plagiarism, Authorship, and Social Software Misuse)[2], which covers several AP tasks ranging from gender and age classification to prediction of language variety and personality of the authors. The first PAN event took place in 2011 and was followed by a series of shared tasks organized since 2013 (Rangel et al., 2013b).

2.2 Related Work

We can divide the approaches towards author profiling into two distinct groups. The first group consists of approaches that leverage extensive feature engineering and employ somewhat more traditional non-neural classifiers. These approaches are presented in subsection 2.2.1. The second group of approaches employ neural architectures for the task at hand and they are presented in subsection 2.2.2.

2.2.1 Author Profiling Approaches Leveraging Extensive Feature Engineering

As was already stated in Chapter 1, the majority of best performing approaches to AP still use more traditional classifiers and require extensive feature engineering (Rangel et al., 2017b), in spite of significant advances in semantic modelling offered by deep learning approaches. They usually rely on bag-of-n-grams (BON) features and SVM classifiers. For example, the winner of the VarDial 2017 DSL task, Bestgen (2017), used an SVM classifier with character n-grams, capitalized word character n-grams, n-grams of part-of-speech (POS) tags and global statistics (proportions of capitalized letters, punctuation marks, spaces, etc.) features. The novelty of this approach was the use of the BM25 weighting scheme (Robertson & Zaragoza, 2009) instead of the traditional term frequency-inverse document frequency (TF-IDF). The experiments conducted by Bestgen (2017) indicate that the choice of the appropriate weighting scheme affects the performance of the model. More generally, the fact that almost all best performing systems in past shared tasks (Zampieri et al., 2017) used some kind of feature weighting suggests that the use of weighting regimes might be positively correlated with gains in classification performance.

SVM-based systems with somewhat simpler features (just word and character n-grams) were used by the winners of the PAN 2017 and 2019 competitions (A. Basile et al., 2017; Pizarro, 2019), by the winners of the VarDial 2016 ADI task (Malmasi & Zampieri, 2016) and the winners of the VarDial 2016 DSL competition (Çöltekin & Rama, 2016). In the VarDial evaluation campaign 2020 (Gaman et al., 2020), the participants were required to build systems capable of discriminating between Moldavian and Romanian dialects in the scope of the Romanian Dialect Identification (RDI) task. The winning system consisted of multiple linear SVM classifiers based on character and word n-gram features (Çöltekin, 2020). An ensemble of linear classifiers with simple n-gram and statistical features was

[1]http://corporavm.uni-koeln.de/vardial/sharedtask.html
[2]http://pan.webis.de/

also employed by the winners of the PAN 2020 event (Katona et al., 2021), which dealt with profiling of fake news spreaders.

In the PAN AP 2016 task, the goal was to build an age and gender classifier for English and Spanish. The team achieving the best score for gender classification in English was Modaresi et al. (2016), who employed word and character n-grams, and the average spelling error as features. They used logistic regression learner for classification. The overall winners for English (Bougiatiotis & Krithara, 2016) (e.g., best overall approach for age and gender classification), trained an SVM model with na RBF kernel and a SVM model with a linear kernel for age and gender, respectively. Their feature set comprised of word n-grams and second-order attributes. Busger op Vollenbroek et al. (2016) were the overall winners of the competition (i.e., they achieved the best average result across both languages and both tasks) and employed a variety of features: word, character and POS n-grams, capitalization, punctuation, word and text length, vocabulary richness and hapax legomena, emoticons and topic-related words.

We should also mention two rare occasions when an SVM-based system did not win in a language variety classification shared task. They occurred at VarDial 2018 German Dialect Identification (GDI) and Indo-Aryan Language Identification (ILI) tasks, where a system called Helsinki language identification (HeLI) described in T. S. Jauhiainen et al. (2018a) and T. S. Jauhiainen et al. (2018b) won the competitions by a large margin. The non-neural system employs adaptive language modelling on character four-grams. The system was nevertheless outperformed by an SVM-based system described in Çöltekin et al. (2018) at the VarDial 2018 Discriminating between Dutch and Flemish in Subtitles task by a comfortable margin.

2.2.2 Neural Approaches for Author Profiling

While neural networks are still not the default approach for some author profiling tasks, there have been some quite successful attempts to tackle AP tasks with neural approaches in the past. In the PAN 2017 shared task, Miura et al. (2017b) ranked fourth on language variety prediction and gender classification tasks, by applying a network with a recurrent neural network layer, a CNN layer, and an attention mechanism. Another milestone was achieved in a set of VarDial 2018 evaluation campaign tasks, where a system based on character-level CNNs and recurrent networks ranked second on the task of distinguishing between four different Swiss German dialects (M. Ali, 2018b) and on the task of distinguishing between five Arabic dialects (M. Ali, 2018a), and fourth on the task of distinguishing five closely-related languages from the Indo-Aryan language family (M. Ali, 2018c).

While in VarDial DSL 2017 (Zampieri et al., 2017) neural networks were scarce and achieved uncompetitive results, in the scope of VarDial DSL 2016 shared task (Malmasi et al., 2016) three systems based on CNNs were proposed for language variety prediction. Belinkov and Glass (2016) ranked sixth out of seven teams by employing a vanilla character-level CNN. Bjerva (2016), on the other hand, combined CNN with recurrent units and added residual layers. The network was fed sentences represented at a byte level as an input and ranked fifth in the competition.

In the VarDial evaluation campaign 2020 Romanian Dialect Identification (RDI) task (Gaman et al., 2020), the second-ranked team employed fine-tuned Romanian BERT models pre-trained on Romanian corpora (Dumitrescu et al., 2020). They tested three strategies, employing cased BERT with document classification head, uncased BERT with document classification head, and an SVM ensemble, which was fed embeddings produced by five different BERT models, some multilingual and some trained on Romanian corpora. The last strategy proved the best in terms of performance.

Even though neural networks showed considerable capability to model semantics of the text, they are still outperformed by SVM-based BON approaches on most author profiling tasks. A hypothesis why this might be the case was proposed in Martinc and Pollak (2019), where we claimed that the main disadvantage of neural networks is its lack of an effective weighting scheme capable of determining how specific are character or word sequences for each document. Since the data is fed into neural networks in small batches, we claim that it is harder for the neural system to obtain a global view on the data and its structure, which is encoded in weighting schemes such as TF-IDF. We discuss one option of how this deficiency of neural networks can be alleviated in Section 2.5.

2.3 BON-based Classifier for Author Profiling

For the PAN 2017 competition, in which language variety and gender of authors of tweets in four languages (English, Spanish, Arabic and Portuguese) needed to be determined, our approach (Martinc et al., 2017) relied on extensive feature engineering and a logistic regression classifier. While most of the features used in the model were different types of n-grams, we also used POS-tag sequences and features that depend on the use of external resources (an emoji list and several word lists). More specifically, the following n-gram features were used in our final model:

- *word unigrams* calculated on lowercased tweets with removed stopwords;
- *word bigrams* calculated on lowercased tweets with removed punctuation;
- *word bound character tetragrams* calculated on lowercased tweets;
- *punctuation trigrams*, the so-called beg-puncts (Sapkota et al., 2015), in which the first character is punctuation but other characters are not, were calculated on lowercased tweets;
- *suffix character tetragrams*, the last four letters of every word that is at least four characters long (Sapkota et al., 2015), were calculated on lowercased tweets;

Other features used in the study are presented below:

- *POS trigrams*, i.e. sequences of three POS tags;
- *emoji counts*, i.e. the number of emojis in the document, counted by using the list of emojis created by Novak et al. (2015)[3];
- *document sentiment*: the above mentioned emoji list also contains the sentiment of a specific emoji, which allowed us to calculate the sentiment of the entire document by simply adding the sentiment of all the emojis in the document;
- *character flood counts*: we counted the number of times that three or more identical character sequences appear in the document;
- *language variety specific word lists*: the lists contain words specific for three distinct English language varieties (United States, Canada and Australia), which enabled us to count how many words from a specific language variety appear in a document and use these counts as features for the English variety classification task.

[3]http://kt.ijs.si/data/Emoji_sentiment_ranking/

With the proposed approach, we managed to achieve accuracies for gender and language variety tasks on four languages presented in Table 2.1. In terms of PAN 2017 competition ranking, we were placed second in terms of joint accuracy achieved on both tasks, second in gender classification and third in language variety classification out of 22 participating teams. Our model won on the task of gender classification in Arabic. The study is enclosed below.

Table 2.1: Accuracy results on the official PAN 2017 AP test set.

	Gender	Language Variety	Both
Arabic	0.8031	0.8288	0.6825
Portuguese	**0.8600**	**0.9838**	**0.8463**
Spanish	0.8193	0.9525	0.7850
English	0.8071	0.8688	0.7042

PAN 2017: Author Profiling - Gender and Language Variety Prediction

Notebook for PAN at CLEF 2017

Matej Martinc[1,2], Iza Škrjanec[2], Katja Zupan[1,2], and Senja Pollak[1]

[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
[2] Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia
matej.martinc@ijs.si,skrjanec.iza@gmail.com,
katja.zupan@ijs.si,senja.pollak@ijs.si

Abstract We present the results of gender and language variety identification performed on the tweet corpus prepared for the PAN 2017 Author profiling shared task. Our approach consists of tweet preprocessing, feature construction, feature weighting and classification model construction. We propose a Logistic regression classifier, where the main features are different types of character and word n-grams. Additional features include POS n-grams, emoji and document sentiment information, character flooding and language variety word lists. Our model achieved the best results on the Portuguese test set in both—gender and language variety—prediction tasks with the obtained accuracy of 0.8600 and 0.9838, respectively. The worst accuracy was achieved on the Arabic test set.

Keywords: author profiling, gender, language variety, Twitter

1 Introduction

Recent trends in natural language processing (NLP) have shown a great interest in learning about the demographics, psychological characteristics and (mental) health of a person based on the text she or he produced. This field, generally known as author profiling (AP), has various applications in marketing, security (forensics), research in social psychology, and medical diagnosis. A thriving subfield of AP is computational stylometry, which is concerned with how the content and genre of a document contribute to its style [4].

One of the commonly addressed tasks in AP is the prediction of an author's gender, but other tasks include the prediction of language variety, age, native language, personality, region of origin or mental health of an author. Within this lively AP community, a series of scientific events and shared tasks on digital text forensic called PAN (Uncovering Plagiarism, Authorship, and Social Software Misuse)[3] have been organized. The first PAN event took place in 2011, while the first AP shared task was organized in 2013 [18].

[3] http://pan.webis.de/

In this paper, we describe our approach to the shared task of PAN AP for 2017 [20], which involves the construction of a model for gender and language variety identification of Twitter users. The rest of the paper is structured as follows: in Section 2 the findings from related work are presented. Section 3 describes the corpus and how it was preprocessed. In Section 4 we present the methodology, while Section 5 presents the results. In Section 6, we conclude the paper and present ideas for future work.

2 Related work

The earliest attempts in author profiling cover gender identification, starting with [8], who used parts of the British National Corpus. Other genres include literary texts [1], scientific papers [27], and emails [15].

The focus of AP has settled much on the social media, including languages other than English. Besides age and gender identification, the PAN shared task has addressed the prediction of personality type [16], setting the task into a cross-lingual [17,16,21,19] and cross-genre [17,21] environment. Since this year the corpus does not contain Dutch tweets, we only describe the findings of PAN AP 2016 task winners for English and Spanish. The goal was to built an age and gender classifier, whereby the model was trained on tweets and tested on blogs without the contestants knowing in advance the genre of the test set. The performance of contestants was evaluated by observing the classification accuracy for gender and age separately, and additionally taking into account the joint identification of both dimensions.

The team achieving the best score for gender classification (0.7564) in English was [11], who used the following features: word uni- and bigrams, character tetragrams, and the average spelling error, and logistic regression for learning. For gender classification in Spanish, the best result was obtained by Deneva [21], who achieved 0.7321 accuracy; a description of the system was not provided. For some contestants, the second order representation has proven useful. This was also the case with the overall winners for English [3], who trained a SVM model with RBF kernel and a SVM model with a linear kernel for age and gender, respectively. Their feature set comprised of unigrams and trigrams, employing also second order attributes and achieving a joint accuracy of 0.3974. The team [28] were the overall winners of the competition. Their linear SVM model performed with the overall accuracy of 0.5258 by employing a variety of features: word, character and POS n-grams, capitalization (of words and sentences), punctuation (final and per sentence), word and text length, vocabulary richness and hapax legomena, emoticons and topic-related words.

Language variety identification is a task of classifying different varieties of the same language by determining lexical and semantic variations between them [6]. Several studies performed classification on newspaper corpora, e.g. in Portuguese [30] and Spanish [32]. Data from social media is another popular resource for this task, e.g. in Spanish Twitter messages [10] or online comments in Arabic [25,29]. For the classification based on language variety several types of features have been considered. Lexical variation is explored with character and/or word n-grams [30,10,31,29,25], grammatical characteristics and syntax are represented in POS n-grams or their distribution [32,25,9]. Variation in orthography was used as a feature by employing a list of spelling variants [9]. Not only linguistic, but also historical and cultural differences were examined in [23] by observing the share of loan words in Brazilian and European Portuguese, while [24] used a so called 'black list' of terms unwanted in Serbian, but accepted and used in Croatian.

3 Data set description and preprocessing

PAN 2017 training set consists of tweets in four different languages grouped by tweet authors, who are labeled by gender and language variety (Table 1). The number of authors for both categories (gender and variety) is balanced in every language. This training set was used for feature engineering, parameter tuning and training of the classification model.

Table 1. PAN 2017 training set structure

Language	Varieties	Authors	Tweets
English	Canada, Ireland, United States, Australia, New Zealand, Great Britain	3,600	360,000
Spanish	Argentina, Colombia, Venezuela, Spain, Chile, Mexico, Peru	4,200	419,998
Portuguese	Brazil, Portugal	1,200	120,000
Arabic	Egypt, Maghrebi, Gulf, Levantine	2,400	240,000

The following preprocessing steps were performed:

– *nonsense tweet removal*: on the English data set we discarded all tweets in which more than 90% of all tokens contain mistakes detected by a spell checker [7];
– *text reversal*: we reversed tweets in the Arabic data set since they are written from right-to-left.

Other preprocessing steps depend on feature construction and three data set transformations can be considered:

– *Tweets-cleaned*: replacing all hashtags, mentions and URLs with specific placeholders #HASHTAG, @MENTION, HTTPURL, respectively. Tweets-cleaned is also POS tagged (we used Averaged perceptron tagger from NLTK library[2] trained on POS tagged corpora for different languages found in NLTK);
– *Tweets-no punctuation*: removing punctuation from Tweets-cleaned;
– *Tweets-no stopwords*: stopwords are removed from Tweets-no punctuation. This preprocessing step is not used on Arabic language (Tweets-no stopwords transformation in Arabic is therefore identical to Tweets-cleaned transformation).

Finally, all tweets belonging to the same author are concatenated and used as one document in further processing.

4 Feature construction and classification model

The usefulness of character n-grams in authorship profiling has been proven before [16,17,26], as they contain information on punctuation, morphology and the lexis [4]. The setting with word uni- and bigrams, and character tri- and tetragrams was applied for gender and personality identification in [26]. For this reason most of the features used in our model were different types of n-grams. We also used other features, such as POS-tag sequences and features that depend on the use of external resources (an emoji list and word lists). We performed several different parameter tuning experiments (either manually or using the Scikit-learn grid search[4] to find best values) to try to find the best feature combination and parameters. All features were normalized with MinMaxScaler from the Scikit-learn library [13].

4.1 Features

The following n-gram features were used in our final model:

- *word unigrams*: calculated on lower-cased Tweets-no stopwords, TF-IDF weighting (parameters: minimum document frequency = 10, maximum document frequency = 80%);
- *word bigrams*: calculated on lower-cased Tweets-no punctuation, TF-IDF weighting (parameters: minimum document frequency = 20, maximum document frequency = 50%);
- *word bound character tetragrams*: calculated on lower-cased Tweets-cleaned, TF-IDF weighting (parameters: minimum document frequency = 4, maximum document frequency = 80%);
- *punctuation trigrams* (the so-called beg-punct [22], in which the first character is punctuation but other characters are not): calculated on lower-cased Tweets-cleaned, TF-IDF weighting (parameters: minimum document frequency = 10%, maximum document frequency = 80%);
- *suffix character tetragrams* (the last four letters of every word that is at least four characters long [22]): calculated on lower-cased Tweets-cleaned, TF-IDF weighting (parameters: minimum document frequency = 10%, maximum document frequency = 80%).

Other features used in the experiments were calculated on Tweets-cleaned data set transformation:

- *POS trigrams*: sequences of three POS tags, TF-IDF weighting (parameters: minimum document frequency = 10%, maximum document frequency = 60%);
- *emoji counts*: the number of emojis in the document, counted by using the list of emojis created by [12][5];

[4] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
[5] http://kt.ijs.si/data/Emoji_sentiment_ranking/

- *document sentiment*: the above-mentioned emoji list also contains the sentiment of a specific emoji, which allowed us to calculate the sentiment of the entire document by simply adding the sentiment of all the emojis in the document (as it turns out, this feature works better without normalizing the resulting sentiment with the number of all emojis in the document);
- *character flood counts*: we counted the number of times that three or more identical character sequences appear in the document;
- *language variety specific word lists*: according to [14] there are words that are specific for different language varieties. We managed to find an English spell checker dictionary[6] containing three different word list for three different English language varieties (United States, Canada and Australia). We calculated the intersection of these three word lists and removed the resulting common words from all three lists. In this way we obtained three language variety specific word lists, which enabled us to count the number of words appearing in a specific language variety list in every document. These features were only used in the English variety classification task.

We also experimented with TruncatedSVD topic modelling and Word2Vec embeddings but these features failed to improve the performance of our model so they were not included in the final model. Many different word count features (e.g., how many times a specific type of word appears in the document), punctuation count features and statistical features such as document length and average word length were also tested. All of these features were evaluated with chi2 feature selection utility from Scikit-learn[7] and proved statistically insignificant in relation to gender and variety target values. Moreover, they did not improve the performance of the model in the 10-fold cross-validation experiments on the training set, which is why they are not included in the final model.

4.2 Classification model

We tested several classifiers and different parameter sets. The following classifiers from Scikit-learn were tested:

- Linear SVM
- Logistic regression
- Random forest
- XGBoost (Extreme gradient boosting)

We also tested some classifier combinations:

- Logistic regression bagging
- Voting classifier with majority vote between Logistic regression, linear SVM and Random forest

Best results were obtained with Logistic regression. Bagging and voting did not improve the results. Logistic regression gave best results with C=1e2 and fit_intercept= False

[6] http://wordlist.aspell.net/dicts/
[7] http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html

parameters. With the help of Scikit-learn FeatureUnion[8], we were also able to specify the weights for different types of features we used. The weights were adjusted with the help of the following procedure:

1. Initialize all feature weights to 1.0.
2. Iterate the list of features. For every feature repeat adding or subtracting 0.1 to the weight until the accuracy of a 10-fold cross-validation is improving. When the best weight is found, move to the next feature on the list.
3. Repeat step 2 until the accuracy cannot be improved anymore.

The weights in our final Logistic regression model were the following:

- word unigrams and word bound character tetragrams: 0.8
- suffix character tetragrams: 0.4
- emoji and character flood counts, document sentiment and language variety specific word lists: 0.3
- POS trigrams: 0.2
- word bigrams and punctuation trigrams: 0.1

We considered adjusting weights for every task and language separately but initial experiments showed that no significant gains in accuracy can be achieved by doing this. This weight configuration proved optimal for both classification tasks and all the languages, which gave us some indication that no significant overfitting was taking place. Our classification model was therefore almost identical for all the languages and both tasks, with the exception of using language variety specific word lists as features in the English language variety task and using no POS trigrams as features in Arabic language.

5 Results

We present the accuracy of our model on the 10-fold cross-validation test as well as the accuracy of the model on the PAN 2017 official test set. The results of a 10-fold cross-validation test are shown in Table 2. All classes are balanced, so for gender the majority classifier's accuracy is 0.50. For language variety, the majority classifier would achieve 0.25 for Arabic, 0.50 for Portuguese, 0.143 for Spanish and 0.167 for English. As can be seen, the model performs best on Portuguese, where it achieved 0.8441 accuracy for gender and 0.9883 for the language variety prediction. The model reaches the lowest gender classification accuracy on Spanish and the lowest language variety classification accuracy on Arabic.

Accuracy results from the PAN 2017 official test set are presented in Table 3. The official PAN 2017 evaluator also measures the accuracy of the model in terms of predicting gender and language variety together (i.e., how many out of all the documents were correctly classified by both the gender and language variety), which is a measurement that was not employed in the 10-fold cross-validation experiments. Again, the model reached the best results on Portuguese, where it achieved 0.8600 accuracy for

[8] http://scikit-learn.org/stable/modules/generated/sklearn.pipeline.FeatureUnion.html

Table 2. Accuracy results of 10-fold cross-validation

	Gender	Language Variety
Arabic	0.8137	0.8345
Portuguese	**0.8441**	**0.9883**
Spanish	0.8059	0.9461
English	0.8280	0.8663

gender, 0.9838 accuracy for language variety prediction and 0.8463 accuracy for both. The model had the worst results for joint gender and language variety prediction on Arabic.

Table 3. Accuracy results on the official PAN 2017 AP test set

	Gender	Language Variety	Both
Arabic	0.8031	0.8288	0.6825
Portuguese	**0.8600**	**0.9838**	**0.8463**
Spanish	0.8193	0.9525	0.7850
English	0.8071	0.8688	0.7042

If we compare results from the 10-fold cross validation experiment and results from the official PAN 2017 test set, we can see that there are some differences. Surprisingly, Arabic is the only language where the results of the 10-fold cross-validation are better than the results on the official PAN 2017 test set on both classification tasks. On the contrary, the model achieved higher accuracy in both of these tasks on the Spanish official PAN 2017 test set. When it comes to English, higher accuracy for gender classification was achieved on the 10-fold cross-validation test and higher accuracy for language variety classification was reached on the official PAN 2017 test set (although the difference in accuracy is very small in this case). For Portuguese, higher accuracy for gender classification was achieved on the official PAN 2017 test set, while higher accuracy for language variety classification was obtained in the 10-fold cross-validation setting.

In general, we can conclude that differences in the accuracy measured on 10-fold cross-validation and official PAN 2017 test sets are not that large, meaning that our model did not overfit in most of the tasks in all the languages. The biggest difference in accuracy measurements is in English gender classification, where 10-fold cross-validation accuracy is more than 2% higher than on the official PAN 2017 test set. This suggests that some overfitting might have occurred in this case.

6 Conclusion and future work

In this paper we have presented our approach to the PAN 2017 author profiling task. We presented findings from the related work that were taken into consideration during the planning phase of our approach. We have also described the preprocessing techniques used, the methodology of our approach and the conducted experiments. Finally, we have presented the results achieved in the 10-fold cross-validation setting and on the official PAN 2017 test set. Our best results for the gender and language variety classification tasks in terms of accuracy were achieved for the Portuguese language and stand at 0.8600 and 0.9838, respectively. If we compare our performance with the results of other participants of PAN 2017, we were placed second in terms of joint accuracy achieved on both tasks, second in gender classification and third in language variety classification. Our model won on the task of gender classification in Arabic.

In our experience, the most difficult part of the task was finding the right features and properly weighting their combination. Our approach confirms the results from related work [22] that determined character n-grams as the most successful features in the AP tasks. Other n-grams, such as word unigrams and bigrams, also work well. The remaining features we used, i.e. POS tag sequences, emoji counts, character flood counts, language variety specific word lists and document sentiments, do not substantially contribute to the classification model accuracy but do, however, offer some new information to the classifier, so they can be considered as useful when combined with other features.

In the future, we plan to evaluate the model on different data sets to test and try to improve the cross-genre performance of the model. We will also consider a deep learning approach to gender and language variety classification. We also plan to address the gender classification task for other languages, such as Slovenian (there is a data set of Slovenian tweets and blogs with labeled gender [5]), Croatian and Serbian. We will also test our language variety classification model on the task of distinguishing between very similar languages, such as Serbian and Croatian.

Acknowledgments

The authors acknowledge the financial support from the Slovenian Research Agency core research programme Knowledge Technologies (P2-0103) and the project “Resources, Tools and Methods for the Research of Nonstandard Internet Slovene" (J6-6842).

References

1. Argamon, S., Goulain, J.B., Horton, R., Olsen, M.: Vive la différence! text mining gender difference in french literature. Digital Humanities Quarterly 3(2) (2009)
2. Bird, S.: Nltk: the natural language toolkit. In: Proceedings of the COLING/ACL on Interactive presentation sessions. pp. 69–72. Association for Computational Linguistics (2006)
3. Bougiatiotis, K., Krithara, A.: Author profiling using complementary second order attributes and stylometric features. CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers (2016)
4. Daelemans, W.: Explanation in computational stylometry. In: International Conference on Intelligent Text Processing and Computational Linguistics. pp. 451–462. Springer (2013)
5. Fišer, D., Erjavec, T., Ljubešić, N.: Janes v0.4: Korpus slovenskih spletnih uporabniških vsebin. Slovenščina 2.0 4(2), 67–99 (2016)

6. Franco-Salvador, M., Rangel, F., Rosso, P., Taulé, M., Martít, M.A.: Language variety identification using distributed representations of words and documents. In: Experimental IR Meets Multilinguality, Multimodality, and Interaction. Lecture Notes in Computer Science. pp. 28–40. Springer International Publishing Switzerland (2015)
7. Kelly, R.: Pyenchant: A spellchecking library for python. http://pythonhosted.org/pyenchant/api/enchant.html, [Online; accessed 15-January-2017]
8. Koppel, M., Argamon, S., Shimoni, A.R.: Automatically categorizing written texts by author gender. Literary and Linguistic Computing 17(4), 401–412 (2002)
9. Lui, M., Cook, P.: Classifying English documents by national dialect. In: Proceedings of Australasian Language Technology Association Workshop. pp. 5–15. Association for Computational Linguistics (2013)
10. Maier, W., Gómez-Rodríguez, C.: Language variety identification in Spanish tweets. In: Language Technology for Closely Related Languages and Language Variants. pp. 25–35. Association for Computational Linguistics (2014)
11. Modaresi, P., Liebeck, M., Conrad, S.: Exploring the effects of cross-genre machine learning for author profiling in PAN 2016. CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers (2016)
12. Novak, P.K., Smailović, J., Sluban, B., Mozetič, I.: Sentiment of emojis. PloS one 10(12), e0144296 (2015)
13. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in Python. The Journal of Machine Learning Research 12, 2825–2830 (2011)
14. Porta, J., Sancho, J.L.: Using maximum entropy models to discriminate between similar languages and varieties. In: Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects. pp. 120–128 (2014)
15. Prabhakaran, V., Reid, E.E., Rambow, O.: Gender and power: How gender and gender environment affect manifestations of power. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1965–1976. ACL (2014)
16. Rangel, F., Celli, F., Rosso, P., Potthast, M., Stein, B., Daelemans, W.: Overview of the 3rd author profiling task at PAN 2015. In: CLEF 2015 Working Notes. CEUR (2015)
17. Rangel, F., Rosso, P., Chugur, I., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daelemans, W.: Overview of the author profiling task at PAN 2014. In: CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers. CEUR (2014)
18. Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., Inches, G.: Overview of the author profiling task at pan 2013. Notebook Papers of CLEF pp. 23–26 (2013)
19. Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., Inches, G.: Overview of the author profiling task at PAN 2013. In: CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers. CEUR (2013)
20. Rangel, F., Rosso, P., Potthast, M., Stein, B.: Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. In: Cappellato, L., Ferro, N., Goeuriot, L., Mandl, T. (eds.) Working Notes Papers of the CLEF 2017 Evaluation Labs. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2017)
21. Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., Stein, B.: Overview of the 4th author profiling task at PAN 2016: cross-genre evaluations. In: CLEF 2016 Working Notes. CEUR-WS.org (2016)
22. Sapkota, U., Bethard, S., Montes-y-Gómez, M., Solorio, T.: Not all character n-grams are created equal: A study in authorship attribution. In: NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015. pp. 93–102 (2015), http://aclweb.org/anthology/N/N15/N15-1010.pdf

23. Soares da Silva, A.: Measuring and parameterizing lexical convergence and divergence between European and Brazilian Portuguese: endo/exogeneousness and foreign and normative influence. In: Advances in Cognitive Sociolinguistics. p. 41–84. De Gruyter Mouton (2010)
24. Tiedemann, J., Ljubešić, N.: Efficient discrimination between closely related languages. In: Proceedings of COLING 2012. p. 2619–2634. COLING (2012)
25. Tillmann, C., Al-Onaizan, Y., Mansour, S.: Improved sentence-level arabic dialect classification. In: Proceedings of the VarDial Workshop. pp. 110–119. Association for Computational Linguistics (2014)
26. Verhoeven, B., Daelemans, W., Plank, B.: Twisty: a multilingual twitter stylometry corpus for gender and personality profiling. In: 10th International Conference on Language Resources and Evaluation (LREC 2016) (2016)
27. Vogel, A., Jurafsky, D.: He said, she said: Gender in the acl anthology. In: Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries. pp. 33–41. ACL (2012)
28. Busger op Vollenbroek, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H., Nissim, M.: Gronup: Groningen user profiling notebook for PAN at clef 2016. CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers (2016)
29. Zaidan, O.F., Callison-Burch, C.: Arabic dialect identification. Computational Linguistics 40(1), 171–202 (2014)
30. Zampieri, M., Gebre, B.G.: Automatic identification of language varieties: The case of Portuguese. In: Jancsary, J. (ed.) Proceedings of KONVENS 2012. pp. 233–237. ÖGAI (September 2012)
31. Zampieri, M., Gebre, B.G.: Varclass: An open source language identification tool for language varieties. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation. pp. 3305–3308. LREC (2014)
32. Zampieri, M., Gebre, B.G., Diwersy, S.: N-gram language models and POS distribution for the identification of Spanish varieties. In: Proceedings of TALN 2013. p. 580–587 (2013)

2.4 Leveraging Taxonomies For Modelling Semantics

The main deficiency of AP approaches based on extensive feature engineering is their inability to model sequential information, which is ignored in the BON presentation, and their limited grasp of semantic information, since the usual TF-IDF-based features used by these systems to detect the importance of each word and also relationships between words are based only on simple frequency-based statistics like word occurrence and co-occurrence. We discuss in detail how to build systems for AP that also consider sequential information by incorporating neural features in Section 2.5. In this section, we focus on another option for improving semantic modelling of a TF-IDF-based system, that is, by leveraging taxonomies. **More specifically, in this section we aim to achieve the stated goal G1 and confirm the hypothesis H1.**

In Škrlj et al. (2021), we propose to use word taxonomies as means for semantic enrichment that would improve the performance of the SVM-based BON classifiers. We claim that the use of additional semantic features derived from taxonomies is useful for classification of short documents, where the amount of semantic data is especially limited. Classification of short documents is common in AP, where the data is frequently gathered online and consists of short texts, such as tweets or social media posts (Chu et al., 2010, 2012). Our approach, named *tax2vec*, builds interpretable semantic features automatically and we show that feeding these additional features together with BON features improves the performance of the model.

More specifically, tax2vec requires a corpus of documents and a word taxonomy (for English, we use the WordNet taxonomy (Miller, 1995)) as an input, and returns a matrix, in which each row represents a semantic vector representation of an input document in the corpus (see Figure 2.1). In the next step, the semantic features are concatenated with the traditional TF-IDF-based features, which means that we employ an early simple fusion according to the taxonomy explained in Chapter 1. We opted for this type of fusion mainly due to its simplicity, since it does not require a development of an additional classification model based purely on taxonomy-based features and since this fusion allows further optimization of the combination framework by exploring different feature selection mechanisms (see below).

When it comes to the production of semantic features, in the first step, a document-based taxonomy is constructed by mapping each sense of a word in the document to the hypernyms in the WordNet taxonomy. Lesk algorithm (P. Basile et al., 2014) is used for word sense disambiguation and at the end of this procedure, each word in the document is associated with a hypernym path in the taxonomy, ranging from the direct hypernym to the root of the taxonomy. After obtaining a taxonomy for each document in the corpus, which consists of all hypernyms of all words in the document, these taxonomies are joined into a corpus-based taxonomy.

In the next step, we construct semantic features by counting the number of times the word or one of its hypernyms appeared in a document and use these values to obtain a TF-IDF weight for each word and hypernym. In this way, each document is represented by a vector consisting of TF-IDF weights for words and hypernyms appearing in the document or document's taxonomy.

After that, feature selection is conducted by employing several selection criteria:

- **Term count**: only a predefined number of rarest words and hypernyms are selected.
- **Betweenness centrality** (Brandes, 2001): measures graph-theoretic properties of individual terms within the corpus-based taxonomy. Again, only a predefined number of best ranked terms according to this measure are kept.

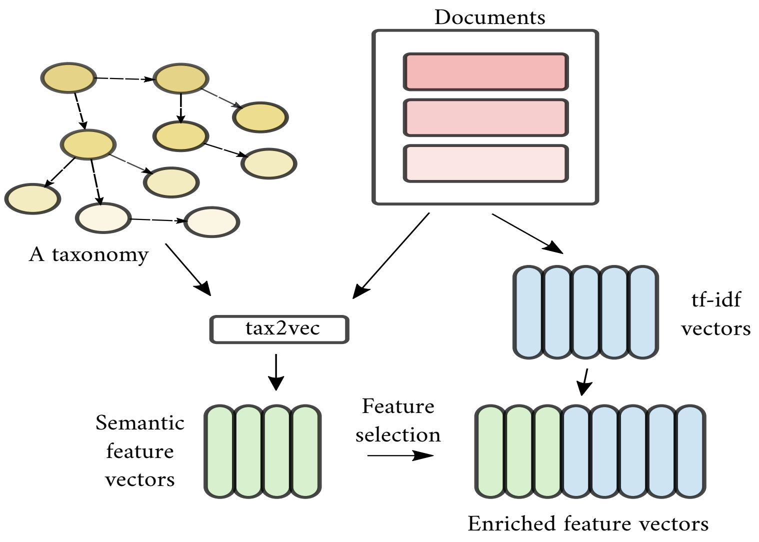


Figure 2.1: Schematic representation of tax2vec, combined with standard TF-IDF representation of documents. Note that darker nodes in the taxonomy represent more general terms.

- **Personalized PageRank** (Kralj et al., 2019): this measure considers graph-theoretic properties for node ranking. Only a predefined number of best ranked nodes are kept.
- **Mutual information** (Shannon, 1948): here, a mutual information metric between each term in the taxonomy and a specific class label is calculated, and only a predefined number of best ranked terms are kept. Note that in contrast to other three metrics, this metric is supervised, since it requires class labels for calculation.

The approach was tested on three author profiling datasets, namely:

- **PAN 2017 (Gender) data set**, where the task is to predict the user's gender from a set of user's tweets[4] (Rangel et al., 2017a);
- **PAN 2016 (Age) data set**, where the task is to predict tweet authors's age range[5] (Rangel et al., 2016b).
- **MBTI (Meyers-Briggs personality type) data set**, where the task is to predict user's personality from a set of user's tweets[6];

Additionally, the algorithm was also tested on a news article data set (more specifically, BBC news data set[7]) and two biomedical data sets (namely, Drug side effects [8] and Drug effectiveness (Grässer et al., 2018) datasets), to determine how well does the algorithm generalise across domains.

Results in Table 2.2 show what happens if a predefined number of semantic features (see column *#Semantic*, 0 means no features are added) is added to three classifiers, namely

[4]https://pan.webis.de/clef17/pan17-web
[5]https://pan.webis.de/clef18/pan18-web
[6]https://www.kaggle.com/datasnaek/mbti-type/kernels
[7]https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv
[8]http://archive.ics.uci.edu/ml/datasets

the SVM-based approach proposed in Martinc et al. (2017) and described in Section 2.2, a generic linear SVM classifier (Chang & Lin, 2011), trained on a predefined number of word and character level n-grams, and a standard feed forward neural network architecture (LeCun et al., 2015; Schmidhuber, 2015).

Besides using these algorithms without added semantic features as a baseline, we also compare the proposed approach to a hierarchical attention network (Yang et al., 2016) and compare the proposed semantic enrichment method to the doc2vec-based semantic enrichment, in which we concatenate doc2vec embeddings (Le & Mikolov, 2014) of size 256 to the features of the overall best performing classifier SVM (Martinc et al.). This is a semantic enrichment framework that is of central focus of this thesis, since symbolic BON features are combined with neural doc2vec features, and its performance can therefore serve as an indicator of how promising is the research of neuro-symbolic hybrid classifiers, on which we focus in the thesis.

Adding semantic features (tax2vec or doc2vec) to BON features improves the performance in five out of six cases (MBTI being the only dataset where semantic enrichment is not working). Tax2vec performs better than doc2vec-based enrichment on three out of five data sets and in most cases works the best when only a small number of semantic features is added (10 or 25). Unsurprisingly, the biggest improvement can be observed on the smallest dataset, PAN 2016 (Age), where 10 semantic features improved the classifiers' performance by about 7% for SVM (generic). The results from Table 2.2 also indicate that the employment of neuro-symbolic hybrid classifier might be beneficial, since the addition of doc2vec features to the BON features outperforms all other configurations on two (PAN (Gender) and Drugs (side)) out of five datasets.

We present the results of a different feature selection strategy in the form of critical distance diagrams (see Figure 2.2), which show average ranks of algorithms across all datasets. Friedman multiple test comparisons followed by the Nemenyi post-hoc correction (Demšar, 2006) were calculated to determine whether performance of different classifiers differs significantly. This is indicated with a red line that connects classifiers that are not statistically significantly different from each other at a confidence level of 95%.

On average, the best performance was achieved using the rarest terms heuristic for

Table 2.2: Effect of the added semantic features to classification performance of different algorithms on six evaluation sets in terms of micro F_1 score. The results in the table correspond to the best performing combination of a classifier and feature selection heuristic, which was in the majority of cases "rarest terms" or "Closeness centrality".

# Semantic	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	HILSTM	0.422	0.752	0.407	0.833	0.443	0.514
0	SVM (Martinc et al.)	0.417	0.814	$\boxed{0.682}$	0.983	0.468	0.503
0	SVM (generic)	0.424	0.751	0.556	0.967	0.445	0.462
256 (doc2vec)	SVM (Martinc et al.)	0.422	$\boxed{0.817}$	0.675	0.979	0.416	$\boxed{0.523}$
30 (tax2vec)	DNN	0.400	0.511	0.182	0.353	0.400	0.321
10 (tax2vec)	SVM (Martinc et al.)	0.445	0.815	0.679	$\boxed{0.996}$	$\boxed{0.47}$	0.506
	SVM (generic)	$\boxed{0.502}$	0.781	0.556	0.972	0.445	0.469
25 (tax2vec)	SVM (Martinc et al.)	0.454	0.814	0.681	0.984	0.468	0.500
	SVM (generic)	0.484	0.755	0.554	0.967	0.449	0.466
50 (tax2vec)	SVM (Martinc et al.)	0.439	0.814	0.681	0.983	0.462	0.499
	SVM (generic)	0.444	0.751	0.554	0.963	0.446	0.463
100 (tax2vec)	SVM (Martinc et al.)	0.424	0.816	0.678	0.984	0.466	0.496
	SVM (generic)	0.422	0.749	0.551	0.958	0.443	0.46
500 (tax2vec)	SVM (Martinc et al.)	0.383	0.797	0.662	0.975	0.45	0.477
	SVM (generic)	0.400	0.724	0.532	0.909	0.424	0.438
1000 (tax2vec)	SVM (Martinc et al.)	0.368	0.783	0.647	0.964	0.436	0.466
	SVM (generic)	0.373	0.701	0.512	0.851	0.407	0.420

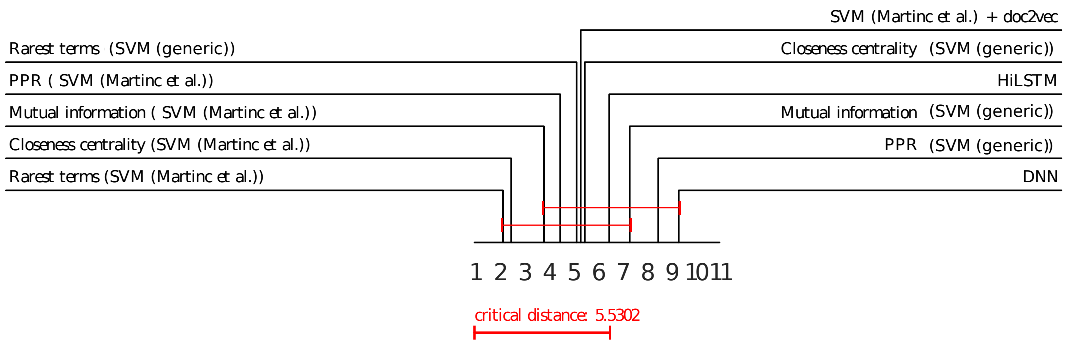


Figure 2.2: Average ranks of different classifiers employing different feature selection algorithms.

feature selection with an SVM (Martinc et al.) classifier. Mutual information and Personalized PageRank heuristic worked the worst on average. While it is worth noting that on average, using doc2vec semantic enrichment performs worse than tax2vec semantic enrichments employing the SVM (Martinc et al.) classifier, the results still indicate that the type of semantic enrichment, in which BON symbolic features are combined with neural embedding features, does outperform purely neural (HiLSTM and DNN) baselines. The results presented in Table 2.2 and Figure 2.2 therefore encourage further development of hybrid approaches. The entire study with all the details about the methodology, experiments and results is enclosed below.

Computer Speech & Language 65 (2021) 101104

Contents lists available at ScienceDirect

Computer Speech & Language

journal homepage: www.elsevier.com/locate/csl

tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification

Blaž Škrlj[a,b,*], Matej Martinc[a,b], Jan Kralj[a], Nada Lavrač[a,c], Senja Pollak[a]

[a] *Jožef Stefan Institute, Jamova 39, Ljubljana 1000, Slovenia*
[b] *Jožef Stefan International Postgraduate School, Jamova 39, Ljubljana 1000, Slovenia*
[c] *University of Nova Gorica, Glavni trg 8, Vipava 5271, Slovenia*

ARTICLE INFO

Article History:
Received 31 January 2019
Revised 19 April 2020
Accepted 21 April 2020
Available online 29 April 2020

Keywords:
taxonomies
vectorization
text classification
short documents
feature construction
semantic enrichment

ABSTRACT

The use of background knowledge is largely unexploited in text classification tasks. This paper explores word taxonomies as means for constructing new semantic features, which may improve the performance and robustness of the learned classifiers. We propose tax2vec, a parallel algorithm for constructing taxonomy-based features, and demonstrate its use on six short text classification problems: prediction of gender, personality type, age, news topics, drug side effects and drug effectiveness. The constructed semantic features, in combination with fast linear classifiers, tested against strong baselines such as hierarchical attention neural networks, achieves comparable classification results on short text documents. The algorithm's performance is also tested in a few-shot learning setting, indicating that the inclusion of semantic features can improve the performance in data-scarce situations. The tax2vec capability to extract corpus-specific semantic keywords is also demonstrated. Finally, we investigate the semantic space of potential features, where we observe a similarity with the well known Zipf's law.

1. Introduction

In text mining, document classification refers to the task of classifying a given text document into one or more categories based on its content (Sebastiani 2002). Given an input set of labeled text documents, a text classifier is expected to learn to associate the patterns appearing in the documents to the document labels. Deep learning approaches (Devlin et al. 2019) have recently become a standard in natural language-related learning tasks, demonstrating good performance on a variety of different classification tasks, including sentiment analysis of tweets (Tang et al. 2015) and news categorization (Kusner et al. 2015). Despite achieving state-of-the-art performance on many tasks, deep learning is not yet optimized for situations, where the number of documents in the training set is low or when the documents contain very little text (Rangel et al. 2017).

Semantic data mining denotes a data mining approach where domain ontologies are used as background knowledge in the data mining process (Ławrynowicz 2017). Semantic data mining approaches have been successfully applied to association rule learning (Angelino et al. 2017), semantic subgroup discovery (Vavpetič and Lavrač 2013); (Perovšek et al. 2015), data visualization (Adhikari et al. 2016) and text classification (Scott and Matwin 1998). Provision of semantic information allows the learner to use features on a higher semantic level, possibly enabling better data generalizations. The semantic information is commonly represented as relational data in the form of taxonomies or ontologies. Development of approaches that leverage such

*Corresponding author.
E-mail address: blaz.skrlj@ijs.si (B. Škrlj).

https://doi.org/10.1016/j.csl.2020.101104

information remains a lively research topic in several fields, including biology (Kim et al. 2018); (Chang et al. 2015), sociology (Freeman 2017) and natural language processing (Wang et al. 2017).

This paper contributes to semantic data mining by using word taxonomies as means for semantic enrichment by constructing new features, with the goal to improve the performance and robustness of the learned classifiers. In particular, it addresses classification of short or incomplete documents, which is useful in a large variety of text classification tasks. Short text is characterized by shortness in the text length, and sparsity in the terms presented, which results in the difficulty in managing and analyzing them based on the bag-of-words representation only. Short texts can be found everywhere, such as search snippets, product reviews and similar (Chen et al. 2011). For example, in author profiling, the task is to recognize the author's characteristics such as age or gender (Rangel et al. 2014), based on a collection of author's text samples. Here, the effect of data size is known to be an important factor, influencing classification performance (Rangel et al. 2016). A frequent text type for this task are tweets, where a collection of tweets from the same author is considered a single document, to which a label must be assigned. The fewer instances (tweets) per user we need, the more powerful and useful the approach. Learning from only a handful of tweets can lead to preliminary detection of bots in social networks, and is hence of practical importance (Chu et al. 2012, 2010). In a similar way, this holds true for nearly any kind of text classification task. For example, for classifying news into a specific topic, using only snippets or titles may be preferred due to non-availability of entire news texts or for increasing the processing speed. Moreover, in biomedical applications, (Grässer et al. 2018) tried to predict drug's side effects and effectiveness from patients' short commentaries, while (Boyce et al. 2012) investigated the use of short user comments to assess drug-drug interactions.

It has been demonstrated that deep neural networks in general need a large amount of information in order to learn complex classifiers, i.e. they require a large training set of documents. For example, the recently introduced BERT neural network architecture (Devlin et al. 2019) consisting of many hidden layers was trained on the whole Wikipedia. It was also shown that the state-of-the-art models do not perform well when incomplete (or scarce) information is used as input (Cho et al. 2015). On the other hand, promising results regarding zero-shot (Socher et al. 2013) and few-shot (Snell et al. 2017) learning were recently achieved.

This paper proposes a novel approach named *tax2vec*, where semantic information available in taxonomies is used to construct semantic features that can improve classification performance on short texts. In the proposed approach, features are constructed automatically and remain *interpretable*. We believe that tax2vec could help explore and understand how external semantic information can be incorporated into existing (black-box) machine learning models, as well as help to explain what is being learned.

This work is structured as follows. Following the theoretical preliminaries and the related work necessary to understand how semantic background knowledge can be used in learning, presented in Section 2, we continue with the description of the proposed tax2vec methodology in Section 3. In Section 4, we describe the experimental setting used to test the methodology. In Section 5, we present the results of experiments, including the evaluation of the qualitative properties of features constructed using tax2vec, and extensive classification benchmark tests. Section 6 discusses the properties of the resulting semantic space and the explainability of the proposed tax2vec algorithm. Implementation and availability of tax2vec is addressed in Section 7. The paper concludes with a summary and prospects for further work in Section 8. For completeness, Appendix A includes a detailed description of the Personalized PageRank algorithm, while Appendix B presents an example segmentation of news articles into paragraphs, forming short documents of interest for this study. Finally, Appendix C contains an additional ablation study regarding the impact of feature numbers on the classifier performance.

2. Background and related work

In this section we present the theoretical preliminaries and selected related work, which served as the basis for the proposed tax2vec approach. We begin by explaining different levels of semantic context and the rationale behind the proposed approach.

2.1. Semantic context

Document classification is highly dependent on *document representation*. In simple bag-of-words representations, the frequency (or a similar weight such as term frequency-inverse document frequency—tf-idf) of each word or *n*-gram is considered as a separate feature. More advanced representations group words with similar meaning together. Such approaches include Latent Semantic Analysis (Landauer 2006), Latent Dirichlet Allocation (Blei et al. 2003), and more recently word embeddings (Mikolov et al. 2013). It has been previously demonstrated that context-aware algorithms significantly outperform the naive learning approaches (Cagliero and Garza 2013). We refer to such semantic context as the *first-level context*.

Second-level context can be introduced by incorporating *background knowledge* (e.g., ontologies) into a learning task, which can lead to improved interpretability and performance of classifiers, learned e.g., by rule learning (Vavpetič and Lavrač 2013) or random forests (Xu et al. 2018). In text mining, Elhadad et al. (2018) present an ontology-based web document classifier, while Kaur and Kumar (2018) propose a clustering-based algorithm for document classification that also benefits from knowledge stored in the underlying ontologies. Cagliero and Garza (2013) present a custom classification algorithm that can leverage taxonomies and demonstrate on a case study of geospatial data that such information can be used to improve the learner's classification performance. Use of hypernym-based features for classification tasks has been considered previously. For example, hypernym-based features were used in rule learning by the Ripper rule learning algorithm (Scott and Matwin 1998). Moreover, it was also demonstrated that the use of hypernym-based features constructed from WordNet significantly impacts the classifier performance (Mansuy and Hilderman 2006).

2.2. Feature construction and selection

When unstructured data is used as input, it is common to explore the options of feature construction. Even though recently introduced deep neural network based approaches operate on simple word indices (or byte-pair encoded tokens) and thus eliminate the need for manual construction of features, such alternatives are not necessarily the optimal approach when vectorizing the background knowledge in the form of taxonomies or ontologies. Features obtained by training a neural network are inherently non-symbolic and as such do not present any added value to the developer's understanding of the (possible) causal mechanisms underlying the learned classifier (Bunge 2017); (Pearl 2009). In contrast, understanding the semantic background of a classifier's decision can shed light on previously not observed second-level context vital to the success of learning, rendering otherwise incomprehensible models easier to understand.

Definition 1 (Feature construction). *Given an unstructured input consisting of n documents, a feature construction algorithm outputs a matrix* $F \in \mathbb{R}^{n \times \alpha}$, *where* α *denotes the predefined number of features to be constructed.*

In practical applications, features are constructed from various data sources, including texts (Stańczyk and Jain 2015), graphs (Kakisim and Sogukpinar 2019); (Škrlj et al. 2019b), audio recordings and similar data (Tomašev et al. 2015). With the increasing computational power at one's disposal, automated feature construction methods are becoming prevalent. Here, the idea is that given some criterion, the feature constructor outputs a set of features selected according to the criterion. For example, the tf-idf feature construction algorithm, applied to a given document corpus, can automatically construct hundreds of thousands of n-gram features in a matter of minutes on an average of-the-shelf laptop.

Many approaches can thus output too many features to be processed in a reasonable time, and can introduce additional noise, which renders the task of learning even harder. To solve this problem, one of the known solutions is *feature selection*.

Definition 2 (Feature selection). *Let* $F \in \mathbb{R}^{n \times \alpha}$ *represent the feature matrix (as defined above), obtained during automated feature construction. A feature selection algorithm transforms matrix F to a matrix* $F' \in \mathbb{R}^{n \times d}$, *where d represents the number of desired features after feature selection.*

Feature selection thus filters out the (unnecessary) features, with the aim of yielding a compact, information-rich representation of the unstructured input. There exist many approaches to feature selection. They can be based on the individual feature's information content, correlation, significance etc. (Chandrashekar and Sahin 2014). Feature selection is, for example, relevant in biological data sets, where only a handful of the key gene markers are of interest, and can be identified by assessing the impact of individual features on the target space (Hira and Gillies 2015).

2.3. Learning from graphs and relational information

In this section we briefly discuss the works that influenced the development of the proposed approach. One of the most elegant ways to learn from graphs is by transforming them into propositional tables, which are a suitable input for many down-stream learning algorithms. Recent attempts to vectorization of graphs include the node2vec (Grover and Leskovec 2016) algorithm for constructing features from homogeneous networks; its extension metapath2vec (Dong et al. 2017) for heterogeneous networks; its symbolic version SGE (Škrlj et al. 2019b); the mol2vec (Jaeger et al. 2018) vectorization algorithm for molecular data; the struc2vec (Ribeiro et al. 2017) graph vectorization algorithm based on homophily relations between nodes, and more. All these approaches (apart from SGE) are sub-symbolic, as the obtained vectorized information (embeddings) are not interpretable. Similarly, recently introduced graph-convolutional neural networks also yield local node embeddings, which take node feature vectors into account (Kipf and Welling 2017); (Hamilton et al. 2017).

In parallel to graph-based vectorization, approaches which tackle the problem of learning from relational databases have also been developed. Symbolic (interpretable) approaches for this vectorization task, known under the term propositionalization, include RSD (Železnỳ and Lavrač 2006), a rule-based algorithm which constructs relational features; and wordification (Perovšek et al. 2015), an approach for unfolding relational databases into bag-of-words representations. The approach, described in the following sections, relies on some of the key ideas initially introduced in the mentioned works on propositionalization, as taxonomies are inherently relational data structures.

3. The tax2vec approach

In this section we outline the proposed tax2vec approach. We begin with a general description of classification from short texts, followed by the key features of tax2vec, which offer solutions to some of the currently not well explored issues in text mining.

3.1. The rationale behind tax2vec

In general text classification tasks, deep learning approaches have outperformed other classifiers (Devlin et al. 2019). However, in classification tasks involving short documents (tweets, opinions, etc.), particularly where the number of instances is low, deep learners are still outperformed by simpler classifiers, such as SVMs (Rangel et al. 2019). This observation was a motivation

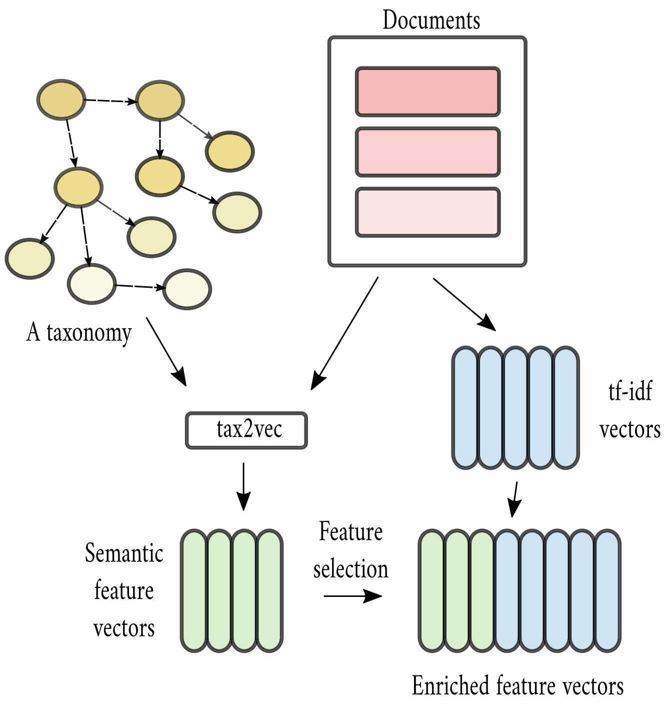


Fig. 1. Schematic representation of tax2vec, combined with standard tf-idf representation of documents. Note that darker nodes in the taxonomy represent more general terms.

for the development of the tax2vec algorithm, proposed in this paper. Compared to non-symbolic node vectorization algorithms discussed in the previous section, tax2vec uses hypernyms as potential features directly and thus makes the process of feature construction and selection possible without the loss of classifier's *interpretability*.

We present the proposed tax2vec algorithm for semantic feature vector construction that can be used to enrich the feature vectors constructed by the established text processing methods such as tf-idf. The tax2vec algorithm takes as input a labeled or unlabeled corpus of *n* documents and a word taxonomy. It outputs a matrix of *semantic feature vectors* in which each row represents a semantics-based vector representation of one input document. Example use of tax2vec in a common language processing pipeline is shown in Figure 1. Note that the obtained semantic feature vectors serve as additional features in the final, vectorized representation of a given corpus.

Let us first explore how parts of the WordNet taxonomy (Miller 1995; Fellbaum 1998) related to the training corpus can be used for the construction of novel features, as such background knowledge can be applied in virtually every English text-based learning setting, as well as for many other languages (Gonzalez-Agirre et al. 2012).

3.2. *Deriving semantic features*

The tax2vec approach implements a two-step semantic feature construction process. First, a document-specific taxonomy is constructed, then a term-weighting scheme is used for feature construction.

3.2.1. *Document-based and corpus-based taxonomy construction*

In the first step of the tax2vec algorithm, a corpus-based taxonomy is constructed from the input document corpus. In this section we describe how the words from individual documents of a corpus are mapped to terms of the WordNet taxonomy to construct a *document-based taxonomy* by focusing on semantic structures, derived exclusively from the *hypernymy* relation between words. Individual document-based taxonomies are then merged into a joint *corpus-based taxonomy*.

When constructing a document-based taxonomy, each word is mapped to the hypernym WordNet taxonomy. This results in a tree-like structure, which spans from individual words to higher-order semantic concepts. For example, given the word monkey, one of its mappings in the WordNet hypernym taxonomy is the term *mammal*, which can be further mapped to e.g., *animal* etc., eventually reaching the most general term, i.e. *entity*.

In order to construct the mapping, the first problem to be solved is *word-sense disambiguation*. For example, the word *bank* has two different meanings, when considered in the following two sentences:

$$
\begin{aligned}
&Synset('entity.n.01')\\
&\rightarrow Synset('abstraction.n.06')\\
&\rightarrow Synset('relation.n.01')\\
&\rightarrow Synset('part.n.01')\\
&\rightarrow Synset('substance.n.01')\\
&\rightarrow Synset('chemical\_element.n.01')\\
&\rightarrow Synset('astatine.n.01')
\end{aligned}
$$

Fig. 2. Example hypernym path extracted for word "astatine", where the $\rightarrow$ corresponds to the "hypernym of" relation (the majority of hypernym paths end with the "entity" term, as it represents one of the most general objects in the taxonomy).

River bank was enforced. | National bank was robbed.

There are many approaches to word-sense disambiguation (WSD). We refer the reader to Navigli (2009) for a detailed overview of the WSD methodology.

In tax2vec, we use Lesk (Basile et al. 2014), a standard WSD algorithm, to map each disambiguated word to the corresponding term in the WordNet taxonomy. The identified term is then associated with a path in the WordNet taxonomy leading from the given term to the root of the taxonomy. Example hypernym path (with WordNet-style notation), extracted for word "astatine", is shown in Figure 2.

By finding a hypernym path to the root of the taxonomy for all words in the input document, a *document-based taxonomy* is constructed, which consists of all hypernyms of all words in the document. After constructing the document-based taxonomy for all the documents in the corpus, the taxonomies are joined into a *corpus-based taxonomy*.

Note that processing each document and constructing the document-based taxonomy is entirely independent from other documents, allowing us to process the documents in parallel and join the results only when constructing the joint corpus-based taxonomy.

3.2.2. *Semantic feature construction*

During the construction of a document-based taxonomy, document-level term counts are calculated for each term. For each word t and document D, we count the number $f_{t,D}$ of times the word or one of its hypernyms appeared in a given document D.

The obtained counts can be used for feature construction directly: each term t from the corpus-based taxonomy is associated with a feature, and a document-level term count is used as the feature value. The current implementation of tax2vec weights the feature values using the double normalization tf-idf metric. For term t, document D and user-selected normalization factor K, feature value tf-idf(t,D,K) is calculated as follows (Manning et al. 2008):

$$
\mathrm{tf}-\mathrm{idf}(t,D,K)=\underbrace{\left(K+(1-K)\frac{f_{t,D}}{\max_{\{t'\in D\}}f_{t',D}}\right)}_{\text{Weighted term frequency}}\cdot\underbrace{\log\left(\frac{N}{n_t}\right)}_{\substack{\text{Inverse}\\ \text{document frequency}}} \tag{1}
$$

where $f_{t,D}$ is the term frequency, normalized by $\max_{\{t'\in D\}}f(t',D)$, which corresponds to the raw count of the most common hypernym of words in the document; value N represents the total number of documents in the corpus, n_t denotes the number of document-based taxonomies the hypernym appears in (i.e. the number of documents that contain a hyponym of t). Note that the term frequencies are normalized with respect to the most frequently occurring term to prevent a bias towards longer documents. In the experiments the normalization constant K was set to 0.5.

3.3. *Feature selection*

The problem with the above presented approach is that all hypernyms from the corpus-based taxonomy are considered, and therefore, the number of columns in the feature matrix can grow to tens of thousands of terms. Including all these terms in the learning process introduces unnecessary noise, and unnecessarily increases the spatial complexity. This leads to the need of feature selection (see Definition 2 in Section 2.2) to reduce the number of features to a user-defined number (a free parameter specified as part of the input). We next describe the scoring functions of feature selection approaches considered in this work.

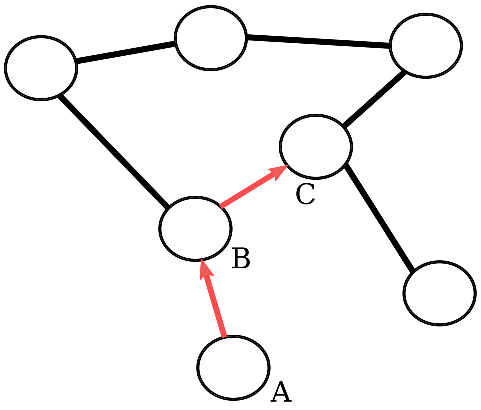


Fig. 3. An example shortest path. The path colored red represents the smallest number of edges needed to reach node C from node A.

As part of tax2vec, we implemented both supervised (Mutual Information - MI and Personalized PageRank - PPR), as well as unsupervised (Betweenness centrality - BC and term count-based selection) feature selection methods, discussed below. Note that the feature selection process is conducted *exclusively* on the semantic space (i.e. on the mapped WordNet terms).

Feature selection by term counts. Intuitively, the rarest terms are the most document-specific and could provide additional information to the classifier. This is addressed in tax2vec by the simplest heuristic, used in the algorithm: a term-count based heuristic that simply takes overall counts of all hypernyms in the corpus-based taxonomy, sorts them in ascending order according to their frequency of occurrence and takes the top d.

Feature selection using term betweenness centrality. As the constructed corpus-specific taxonomy is not necessarily the same as the WordNet taxonomy, the graph-theoretic properties of individual terms within the corpus-based taxonomy could provide a reasonable estimate of a term's importance. The proposed tax2vec implements the betweenness centrality (BC) (Brandes 2001) measure of individual terms as the scoring measure. The betweenness centrality is defined as:

$$BC(t) = \sum_{u \neq v \neq t} \frac{\sigma_{uv}(t)}{\sigma_{uv}}; \tag{2}$$

where σ_{uv} corresponds to the number of shortest paths (see Figure 3) between nodes u and v, and $\sigma_{uv}(t)$ corresponds to the number of paths that pass through term (node) t. Intuitively, betweenness measures the t's importance in the corpus-based taxonomy. Here, the terms are sorted in a descending order according to their betweenness centrality, and again, the top d terms are used for learning.

Feature selection using mutual information. The third heuristic, mutual information (MI) (Peng et al. 2005), aims to exploit the information from the labels, assigned to the documents used for training. The MI between two random discrete variables represented as vectors F_i and Y (i.e. the i-th hypernym feature and a target binary class) is defined as:

$$MI(F_i, Y) = \sum_{x,y \in \{0,1\}} p(F_i = x, Y = y) \cdot \log_2 \left(\frac{p(F_i = x, Y = y)}{p(F_i = x) \cdot p(Y = y)} \right) \tag{3}$$

where $p(F_i = x)$ and $p(Y = y)$ correspond to marginal distributions of the joint probability distribution of F_i and Y. Note that for this step, tax2vec uses the binary feature representation, where the tf-idf features are rounded to the closest integer value (either 0 or 1). This way, only well represented features are taken into account. Further, tax2vec uses one-hot encodings of target classes, meaning that each target class vector consists exclusively of zeros and ones. For *each* of the target classes, tax2vec computes the mutual information (MI) between *all* hypernym features (i.e. matrix X) and a given class. Hence, for each target class, a vector of mutual information scores is obtained, corresponding to MI between individual hypernym features and a given target class.

Finally, tax2vec sums the MI scores obtained for each target class to obtain the final vector, which is then sorted in descending order. The first d hypernym features are used for learning. At this point tax2vec yields the selected features as a sparse matrix, maintaining the spatial complexity amounting to the number of float-valued non-zero entries.

Personalized PageRank-based hypernym ranking. Advances by Kralj et al. (2019) and Kralj (2017) in learning using extensive background knowledge for rule induction explored the use of Personalized PageRank (PPR) algorithm for node subset selection in semantic search space exploration. In tax2vec, we use the same idea to prioritize (score) hypernyms in the corpus-based taxonomy. In this section, we first briefly describe the Personalized PageRank algorithm and then describe how it is applied in tax2vec.

The PPR algorithm takes as an input a network and a set of starting nodes in the network and returns a vector assigning a score to each node in the input network. The scores of nodes are calculated as the stationary distribution of the positions of a random walker that starts its walk on one of the starting nodes and, in each step, either randomly jumps from a node to one of its neighbors (with probability p, set to 0.85 in our experiments) or jumps back to one of the starting nodes (with probability $1-p$). Detailed description of the PPR used in tax2vec is given in Appendix A. The PPR algorithm is used in tax2vec as follows:

(a) Identify a set of hypernyms in the corpus-based taxonomy, to which the words in the input corpus map to in the first step of tax2vec (described in Section 3.2.1).
(b) Run the PPR algorithm on the corpus-based taxonomy, using the hypernyms identified in step 1 as the starting set.
(c) Use the top d best ranked hypernyms as candidate features.

Note that this heuristics offers *global* node ranks with respect to the corpus used.

3.4. The tax2vec algorithm

All the aforementioned steps form the basis of tax2vec, outlined in Algorithm 1. First, tax2vec iterates through the given labeled document corpus in parallel (lines 3–7). For each document, *MaptoTaxonomy* method identifies a set of disambiguated words and determines their corresponding terms in taxonomy $\mathfrak{T}$ (i.e. WordNet) using method m (i.e. Lesk). Term counts are stored for later use (*storeTermCounts*), and the taxonomy, derived from a given document (*doc*) is added to the corpus taxonomy $\mathfrak{T}_{CORPUS}$. Once traversed, the terms present in $\mathfrak{T}_{CORPUS}$ represent potential *features*. Term counts, stored for each document are aggregated into n vectors, where n is the number of documents in the corpus. The result of this step is a real-valued, sparse matrix (vecSpace), where columns represent all possible terms from $\mathfrak{T}_{CORPUS}$. In the following step, feature selection is conducted. Here, graph-based methods (e.g., BC and PPR) identify top d terms based on $\mathfrak{T}_{CORPUS}$'s properties (lines 9–12), and non-graph methods (e.g., MI) is used directly on the sparse matrix to select which d features are the most relevant (lines 13–15). Finally, *selectedFeatures*, a matrix of selected semantic features is returned.

Note that in practice, tax2vec also stores the inverse document frequencies. We omit the description of this step for readability purposes.

Algorithm 1. tax2vec

```
Data: Training set documents $D$, training document labels $Y_{tr}$,
      WordNet taxonomy $\mathfrak{T}$, word-to-taxonomy mapping $m$, feature
      selection heuristic $h$, number of selected features $d$
1  $\mathfrak{T}_{\text{CORPUS}}$ ← empty structure;
2  termCounts ← empty structure;
3  for $doc \in D$ (in parallel) do
4      $\mathfrak{T}_{\text{DOCUMENT}}$ ← MaptoTaxonomy($doc, \mathfrak{T}, m$);
5      Add storeTermCounts($\mathfrak{T}_{\text{DOCUMENT}}$) to termCounts;
6      Add $\mathfrak{T}_{\text{DOCUMENT}}$ to $\mathfrak{T}_{\text{CORPUS}}$;
7  end
8  vecSpace ← tf-idf(constructTfVectors($D, \mathfrak{T}_{\text{CORPUS}}$,termCounts));
9  if $h$ is graph-based then
10     topTerms ← selectFeatures(h, $\mathfrak{T}_{\text{CORPUS}}$, d, optional $Y_{tr}$);
11     selectedFeatures ← select topTerms from vecSpace;
12 end
13 else
14     selectedFeatures← selectFeaturesDirectly(h, vecSpace,d ,$Y_{tr}$);
15 end
16 return selectedFeatures;
Result: $d$ new feature vectors in sparse vector format.
```

3.5. Handling noise

Numerous data sets, including contemporary social media data sets, can be noisy and as such hard to handle by a learning system. We next discuss how distinct parts of tax2vec potentially handle noise in the data, including typos, incomplete and missing words and uncommon characters.

During the initial step of the semantic space construction, tax2vec conducts document-level word disambiguation in order to semantically characterize a given token (word). During this step, any tokens that are not present in the taxonomy will be ignored. Further, as word disambiguation requires a certain word window to operate, this hyperparameter can be used to control the size of context considered by tax2vec. In this work, however, we did not explicitly address the problem of invalid tokens in a given token's neighborhood, yet observed that small window sizes (two and three) offered reasonably robust performance.

Even though disambiguation with Lesk offers the initial *semantic pruning* capabilities, the tax2vec algorithm can further address potential noise as follows. As the user can determine the depth in the WordNet taxonomy that will be considered as the starting point for semantic space construction, potentially too specific terms can be avoided if necessary.

Finally, in the third step, tax2vec conducts *feature selection*. This part of the algorithm is responsible for *filtering* redundant and non-informative terms that could be considered as noise. We tested both supervised, as well as unsupervised feature selection methods, exploring whether additional information about class labels helps with term pruning. Apart from the semantic pruning and selection strategies discussed above, links, mentions and hashtags can be removed to further reduce the noise in social media texts (as mentioned in the description of the SVM implementation by Martinc et al. (2017) in Section 4.2).

We believe all three steps to some extent address how noise is being handled. However, it is expected that additional grammar correction and text normalization could serve as a complementary step to offer improved performance on social media texts.

4. Experimental setting

This section presents the experimental setting used in testing the performance of tax2vec in document classification tasks. We begin by describing the data sets on which the method was tested. Next, we describe the classifiers used to assess the use of features constructed using tax2vec, along with the baseline approaches. We continue by describing the metrics used to assess classification performance, and the description of the experiments.

4.1. Data sets

We tested the effects of features produced with tax2vec on six different class labeled text data sets summarized in Table 1, intentionally chosen from different domains.

The first three data sets are composed of short documents from social media, where we consider classification of tweets.

PAN 2017 (Gender) data set. Given a set of tweets per user, the task is to predict the user's gender[1] (Rangel et al. 2017).

MBTI (Myers-Briggs personality type) data set. Given a set of tweets per user, the task is to predict to which personality class a user belongs[2], first discussed in (Myers 1962).

PAN 2016 (Age) data set. Given a set of tweets per user, the classifier should predict the users's age range[3] (Rangel et al. 2016).

Next, we consider a news articles data set by which we test the potential of the method also on longer documents, while for few shot learning experiments (Section 5.3), we transform the setting to short text documents by using only few paragraphs per article and test whether competitive performance to full-text-based classification can be obtained.

BBC news data set. Given a news article (composed of a number of paragraphs)[4], the goal is to assign to it a topic from a list of topic categories[5] (Greene and Cunningham 2006).

We also consider two biomedical data sets related to drug consumption. Here, the same training instances in the form of short user commentaries were used to predict two different targets.

Drug side effects. This data set links user opinions to side effects of a drug they are taking as treatment. The goal is to predict the side effects prior to experimental measurement (Grässer et al. 2018).[6]

Drug effectiveness. Similarly to side effects (previous data set), the goal of this task is to predict drug effectiveness (Grässer et al. 2018).

[1] https://pan.webis.de/clef17/pan17-web
[2] https://www.kaggle.com/datasnaek/mbti-type/kernels
[3] https://pan.webis.de/clef18/pan18-web
[4] Split to paragraphs according to the double new line is presented in Appendix B.
[5] https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv
[6] http://archive.ics.uci.edu/ml/datasets

Table 1
Data sets used for experimental evaluation of tax2vec's impact on learning. Note that MNS corresponds to the maximum number of text segments (max. number of tweets or comments per user or number of news paragraphs as presented in Appendix B).

Data set (target)	Classes	Words	Unique words	Documents	MNS	Average tokens per segment
PAN 2017 (Gender)	2	5169966	607474	3600	102	14.23
MBTI (Personality)	16	11832937	372811	8676	89	27.98
PAN 2016 (Age)	5	943880	178450	402	202	13.17
BBC news	5	902036	58128	2225	76	70.39
Drugs (Side effects)	4	385746	27257	3107	3	41.47
Drugs (Overall effect)	4	385746	27257	3107	3	41.47

4.2. The classifiers used

As tax2vec serves as a preprocessing method for data enrichment with semantic features, arbitrary classifiers can use the resulting semantic features for learning. Note that in the experiments, the final feature space is composed of both semantic and non-semantic (original) features, i.e., the final feature set used for learning is formed *after* the semantic features have been constructed and selected, by concatenating the original features and the semantic features. We use the following learners:

PAN 2017 approach. An SVM-based approach that relies heavily on the method proposed by Martinc et al. (2017) for the author profiling task in the PAN 2017 shared task (Rangel et al. 2017). This method is based on sophisticated hand-crafted features calculated on different levels of preprocessed text including optional social media text cleaning (e.g., Twitter hashtag, mentions, url replacement with filler tokens). The following features were used:

tf-idf weighted word unigrams calculated on lower-cased text with stopwords removed;
tf-idf weighted word bigrams calculated on lower-cased text with punctuation removed;
tf-idf weighted word bound character tetragrams calculated on lower-cased text;
tf-idf weighted punctuation trigrams (the so-called beg-punct (Sapkota et al. 2015), in which the first character is punctuation but other characters are not) calculated on lower-cased text;
tf-idf weighted suffix character tetragrams (the last four letters of every word that is at least four characters long (Sapkota et al. 2015)) calculated on lower-cased text;
emoji counts of the number of emojis in the document, counted by using the list of emojis created by Novak et al. (2015),[7]; this feature is only useful if the data set in question contains emojis;
document sentiment using the above-mentioned emoji list that contains the sentiment of a specific emoji, used to calculate the sentiment of the entire document by simply adding the sentiment of all the emojis in the document; this feature is only useful if the data set in question contains emojis;
character flood counts calculated by the number of times that three or more identical character sequences appear in the document;

In contrast to the original approach proposed (Martinc et al. 2017), we do not use POS tag sequences as features and a Logistic regression classifier is replaced by a Linear SVM. Here, we experimented with the regularization parameter C, for which values in range {1, 20, 50, 100, 200} were tested. This SVM variant is from this point on referred to as "SVM (Martinc et al.)". As this feature construction pipeline consists of too many parameters, we were not able to perform extensive grid search due to computational complexity. Thus, we did not experiment with feature construction parameters, and kept the configuration proposed in the original study.

Linear SVM with automatic feature construction. The second learner is a libSVM linear classifier (Chang and Lin 2011), trained on a predefined number of word and character level n-grams, constructed using Scikit-learn's *TfidfVectorizer* method. To find the best setting, we varied the SVM's C parameter in range {1, 20, 50, 100, 200}, the number of word features between {10000, 50000, 100000, 200000} and character features between {0, 30}[8]. Note that the word features were sorted by decreasing frequency. Here, we considered (word) n-grams of lengths between two and six. This SVM variation is from this point on referred to as "SVM (generic)". The main difference between "SVM (generic)" and "SVM (Martinc et al.)" is that the latter approach also considers punctuation-based and suffix-based features. Further, it is capable of constructing features that represent document sentiment, which was proven to work well for social media data sets (e.g., tweets). Finally, Martinc's approach also accounts for character repetitions and has a parameter for social-media text cleaning in preprocessing. Note that for both SVM approaches we fine-tuned the hyperparameter C, as is common when employing SVMs. The hyperparameter's values govern how penalized the learner is for a miss-classified instance, which is a property that was shown to vary across data sets (see for example (Meyer et al. 2003)).

[7] http://kt.ijs.si/data/Emoji_sentiment_ranking/
[8] In Figure C1 (Appendix C), the reader can observe the results of the initial experiments on the number of word features that led to selection of this hyperparameter range.

Hierarchical attention networks (HILSTM). The first neural network baseline is the recently introduced hierarchical attention network (Yang et al. 2016). Here, we performed a grid search over {64, 128, 256} hidden layers sizes, embedding sizes of {128, 256, 512}, batch sizes of {8, 24, 52} and number of epochs {5, 15, 20, 30}. For detailed explanation of the architecture, please refer to the original contribution (Yang et al. 2016). We discuss the best-performing architecture in Section 5 below.

Deep feedforward neural networks. As tax2vec constructs feature vectors, we also attempted to use them as inputs for a standard feedforward neural network architecture (LeCun et al. 2015); (Schmidhuber 2015). Here, we performed a grid search across hidden layer settings: {(128, 64), (10, 10, 10)} (where for example (128,64) corresponds to a two hidden layer neural network, where in the first hidden layer there are 128 neurons and 64 in the second), batch sizes {8, 24, 52} and the number of training epochs {5, 15, 20}.[9]

4.3. Semantic features

In addition to the semantic features constructed by tax2vec, doc2vec-based semantic features (Le and Mikolov 2014) were used as a baseline in order to allow for a simple comparison between two semantic feature construction approaches. They were concatenated with the features constructed by Martinc et al.'s SVM approach described in Section 4.2, in order to compare the benefits merging the BoW-based representations with a different type of semantic features (embedding-based ones). We set the embedding dimension to 256, as it was shown that lower dimensional embeddings do not perform well (Pennington et al. 2014).

4.4. Description of the experiments

The experiments were set up as follows. For the drug-related data sets, we used the splits given in the original paper (Grässer et al. 2018). For other data sets, we trained the classifiers using stratified 90%: 10% splits. For each classifier, 10 such splits were obtained. The measure used in all cases is F_1, where for the multiclass problems (e.g., MBTI), we use the micro-averaged F_1. All experiments were repeated five times using different random seeds. The features obtained using tax2vec are used in combination with SVM classifiers, while the other classifiers are used as baselines.[10]

5. Classification results

In this section we provide the results obtained by conducting the experiments outlined in the previous section. We begin by discussing the overall classification performance with respect to different heuristics used. Next, we discuss how tax2vec augments the learner's ability to classify when the number of text segments per user is reduced.

5.1. Classification performance evaluation

The F_1 results are presented in Table 2. The first observation is that combining BoW-based representations with semantic features (tax2vec or doc2vec) leads to performance improvements in five out of six cases (MBTI being the only data set where no improvement is detected). Tax2vec outperforms doc2vec-based vectors in three out of five data sets (PAN 2016 (Age), BBC News and Drugs (effect)), while doc2vec-based features outperform tax2vec on two data sets (PAN 2017 (gender) and Drugs (Side)).

When it comes to tax2vec, up to 100 semantic features aid the SVM learners to achieve better accuracy. The most apparent improvement can be observed for the case of PAN 2016 (Age) data set, where the task was to predict age. Here, 10 semantic features notably improved the classifiers' performance (up to approximately 7% for SVM (generic)). Further, a minor improvement over the state-of-the-art was also observed on the PAN 2017 (Gender) data set and the BBC news categorization (see results for SVM (Martinc et al.)). Hierarchical attention networks outperformed all other learners for the task of side effects prediction, yet semantics-augmented SVMs outperformed neural models when general drug effects were considered as target classes. Similarly, no performance improvements were offered by tax2vec on the MBTI data set.

We now present the classification results in the form of critical distance diagrams, shown in Figures 4, 5 and 6. The diagrams show average ranks of different algorithms according to the (micro) F_1 measure. A red line connects groups of classifiers that are not statistically significantly different from each other at a confidence level of 5%. The significance levels are computed using Friedman multiple test comparisons followed by Nemenyi post-hoc correction (Demšar 2006). For each data set, we selected the best performing parametrization (hyperparameter settings). The best (on average) performing C parameter for both SVM models was 50. The number of features that performed the best for all hyperparameter settings of the SVM (generic) considered in this study is 100,000. The HILSTM architecture's topology varied between data sets, yet we observed that the best results were obtained when more than 15 epochs of training were conducted, combined with the hidden layer size of 64 neurons, where the size of the attention layer was of the same dimension.

[9] The two deep architectures were implemented using TensorFlow (Abadi et al. 2015), and trained using a Nvidia Tesla K40 GPU. We report the best result for top 30 semantic features with the rarest terms heuristic.

[10] Note that simple feedforward neural networks could also be used in combination with hypernym features—we leave such computationally expensive experiments for further work.

Table 2
Effect of the added semantic features to classification performance, where all text segments (tweets/comments per user or segments per news article) are used. The best performing feature selection heuristic for the majority of top performing classifiers was "rarest terms" or "Closeness centrality", indicating that only a handful of hypernyms carry added value, relevant for classification. Note that the results in the table correspond to the best performing combination of a classifier and a given heuristic.

# Semantic	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	HILSTM	0.422	0.752	0.407	0.833	0.443	0.514
0	SVM (Martinc et al.)	0.417	0.814	$\boxed{0.682}$	0.983	0.468	0.503
0	SVM (generic)	0.424	0.751	0.556	0.967	0.445	0.462
256 (doc2vec)	SVM (Martinc et al.)	0.422	$\boxed{0.817}$	0.675	0.979	0.416	$\boxed{0.523}$
30 (tax2vec)	DNN	0.400	0.511	0.182	0.353	0.400	0.321
10 (tax2vec)	SVM (Martinc et al.)	0.445	0.815	0.679	$\boxed{0.996}$	$\boxed{0.47}$	0.506
	SVM (generic)	$\boxed{0.502}$	0.781	0.556	0.972	0.445	0.469
25 (tax2vec)	SVM (Martinc et al.)	0.454	0.814	0.681	0.984	0.468	0.500
	SVM (generic)	0.484	0.755	0.554	0.967	0.449	0.466
50 (tax2vec)	SVM (Martinc et al.)	0.439	0.814	0.681	0.983	0.462	0.499
	SVM (generic)	0.444	0.751	0.554	0.963	0.446	0.463
100 (tax2vec)	SVM (Martinc et al.)	0.424	0.816	0.678	0.984	0.466	0.496
	SVM (generic)	0.422	0.749	0.551	0.958	0.443	0.460
500 (tax2vec)	SVM (Martinc et al.)	0.383	0.797	0.662	0.975	0.450	0.470
	SVM (generic)	0.400	0.724	0.532	0.909	0.424	0.438
1000 (tax2vec)	SVM (Martinc et al.)	0.368	0.783	0.647	0.964	0.436	0.466
	SVM (generic)	0.373	0.701	0.512	0.851	0.407	0.420

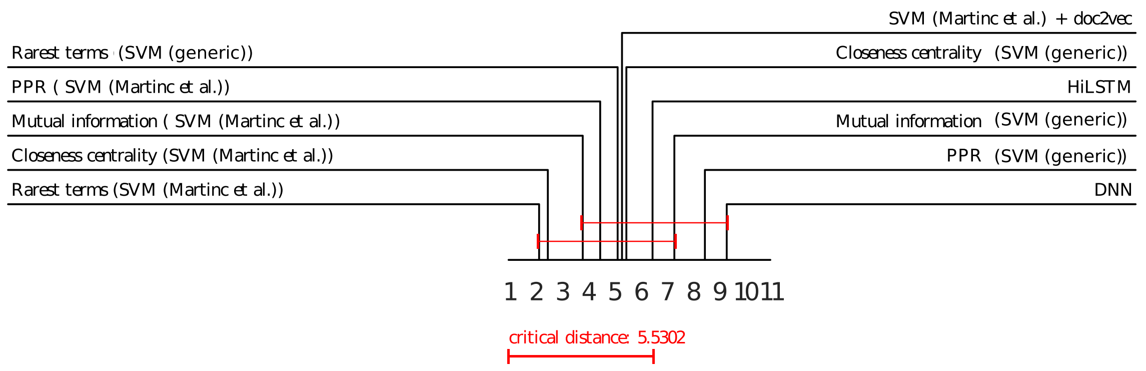


Fig. 4. Average overall classifier ranks. The top (on average) performing classifier is an SVM (Martinc et al.) classifier augmented with semantic features, selected using either simple frequency counts or closeness centrality.

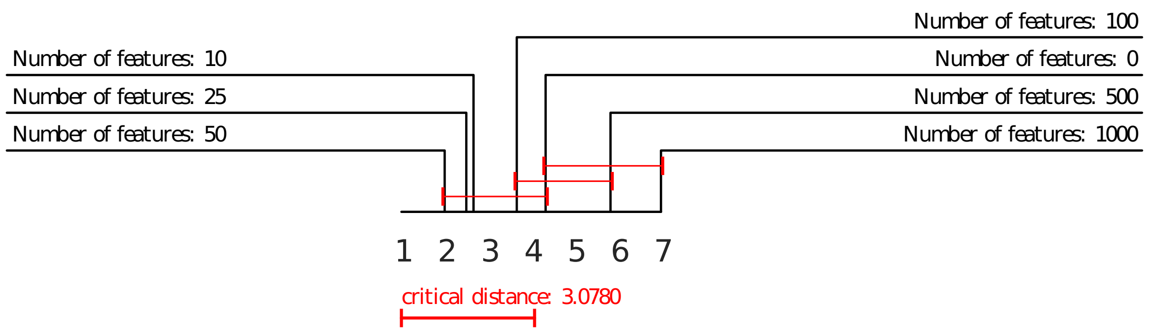


Fig. 5. Effect of semantic features on average classifier rank. Up to 100 semantic features positively affects the classifiers' performance.

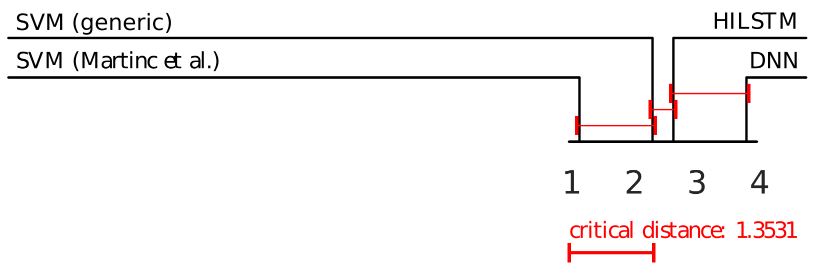


Fig. 6. Overall model performance. SVMs dominate the short text classification. The diagram shows performance averaged over all data sets, where the best model parameterizations (see Table 2) were used for comparison.

In terms of feature selection, the following can be observed (Figure 4). On average, the best performance was obtained when rarest terms heuristic was considered (first and fifth rank). Further, rarest terms, as well as the Personalized PageRank performed better (on average) than mutual information, which can be considered as a baseline in this comparison. The results indicate that myopic feature selection is not optimal when considering novel semantic features. We can also observe that on average the configuration with doc2vec semantic features (SVM (Martinc et al.) + doc2vec) performs worse (ranked as sixth) than all other configurations with SVM (Martinc et al.).

In Figure 5, the reader can observe the performances *of all learners*, averaged w.r.t. to the number of semantic features. The drawn diagram indicates that adding 10, 25 or 50 features to a classifier perform similarly well, however, as also discussed in the previous paragraph, the performance drops when larger semantic space is considered.

Finally, in Figure 6 it can be observed that the overall performance of Martinc et al.'s SVMs is the best, followed by generic SVMs, as well as HILSTMs. We believe such performance drop with deep neural networks in general is due to concatenation of documents prior to learning, and as only a fixed sequence length can be considered, potentially large parts of the token space were neglected during learning. A similar result was, for example observed in the most recent PAN competition (Martinc et al. 2019).

5.2. Few-shot (per instance) learning

As discussed in the introductory sections, one of the goals of this paper was also to explore the setting, where only a handful of text segments per user are considered. Even though such setting is not strictly a few-shot learning (Snell et al. 2017), reducing the number of text segments per instance (e.g., user) aims to simulate a setting where there is limited information available. In Table 3, we present the results for the setting, where only (up to) 10 text segments (e.g., tweets or paragraphs in a given news article) were used for training.

The segments were sampled randomly. Only a single text segment per user was considered for the medical texts, as they consist of at max of three commentaries. Similarly, as the BBC news data set consists of news article-genre pairs, we split the news

Table 3
Effect of added semantic features to classification performance—few shot learning.

Semantic (tax2vec)	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	SVM (Martinc et al.)	0.378	0.617	0.288	0.977	0.468	0.503
	SVM (generic)	0.429	0.554	0.225	0.936	0.445	0.462
10	SVM (Martinc et al.)	0.39	0.616	$\boxed{0.292}$	$\boxed{0.981}$	0.47	0.503
	SVM (generic)	0.429	0.557	0.225	0.948	0.444	0.464
25	SVM (Martinc et al.)	0.429	$\boxed{0.618}$	0.288	0.979	0.465	0.5
	SVM (generic)	$\boxed{0.439}$	0.562	0.226	0.933	0.445	0.458
50	SVM (Martinc et al.)	0.402	0.617	0.288	0.974	0.474	$\boxed{0.504}$
	SVM (generic)	0.42	0.557	0.225	0.919	0.442	0.46
100	SVM (Martinc et al.)	0.382	0.614	0.286	0.974	$\boxed{0.476}$	0.493
	SVM (generic)	0.411	0.552	0.223	0.906	0.437	0.457
500	SVM (Martinc et al.)	0.359	0.604	0.276	0.959	0.465	0.471
	SVM (generic)	0.365	0.548	0.22	0.8	0.419	0.435
1000	SVM (Martinc et al.)	0.34	0.59	0.266	0.925	0.442	0.46
	SVM (generic)	0.359	0.535	0.213	0.704	0.412	0.417

article to paragraphs, which we randomly sampled. The rationale for such sampling is to be able to evaluate tax2vec's performance when, for example, only a handful of paragraphs are available (e.g., only the lead).

We observe that tax2vec based features improve the learners' performance on all of the data sets, albeit by a small margin. The results indicate that adding semantic information improves the performance as only a handful of text segments does not necessarily contain the relevant information.

5.3. Few-shot learning results

We next discuss the results of few-shot learning, as to our knowledge this type of experiments were not conducted before in combination with semantic feature construction methods. The first observation is, semantic features indeed offer more consistent performance improvements than those observed in Table 2, where incremental improvements were not observed on all data sets. In a few-shot learning scenario, however, on all data sets, the inclusion of semantic space either resulted in similar or better performance, indicating a consistent positive effect on the learning in a limited setting. The differences in learner's performance vary around 1% improvement. For example, a 1% improvement was observed for PAN 2016 (Age), BBC News and MBTI data sets.

We finally comment on the classification performance when considering the BBC data set when comparing to reported state-of-the-art. The observed results ($\geq 98\%$) are competitive to neural approaches, such as for example as reported in Asim et al. (2019), where similar span of accuracy was observed. Furthermore, doc2vec-based models have been observed to perform similarly (Trieu et al. 2017). The results of this work indicate that by considering smaller number of paragraphs (instead of whole documents), competitive performance can be observed on the BBC data set.

5.4. Interpretation of results

In this section we explain the intuition behind the effect of semantic features on the classifier's performance. Note that the best performing SVM models consisted of thousands of tf-idf word and character level features, yet only up to 100 semantic features, when added, notably improved the performance. This effect can be understood via the way SVMs learn from high-dimensional data. With each new feature, we increase the dimensionality of the feature space. Even a single feature, when added, potentially impacts the hyperplane construction. Thus, otherwise problem-irrelevant features can become relevant when novel features are added. We believe that adding semantic features to (raw) word tf-idf vector space introduces new information, crucial for successful learning, and potentially aligns the remainder of features so that the classifier can better separate the points of interest.

Table 4
Most informative features with respect to the target class (ranked by MI)—Classes represent news topics (BBC) and different age intervals (PAN 2016 (Age)). Individual target classes are sorted according to a descending mutual information with respect to a given feature.

		Sorted target class-mutual information pairs				
Semantic feature	Average MI	Class 1	Class 2	Class 3	Class 4	Class 5
BBC News data set						
tory.n.03	0.057	politics:0.14	entertainment:0.05	business:0.03	sport:0.01	x
movie.n.01	0.059	business:0.14	politics:0.04	entertainment:0.04	sport:0.02	x
conservative.n.01	0.061	politics:0.15	entertainment:0.05	business:0.03	sport:0.01	x
vote.n.02	0.061	business:0.15	entertainment:0.04	politics:0.04	sport:0.02	x
election.n.01	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0	x
topology.n.04	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0	x
mercantile_establishment.n.01	0.068	politics:0.17	business:0.07	entertainment:0.03	sport:0.01	x
star_topology.n.01	0.069	politics:0.17	business:0.07	entertainment:0.03	sport:0.01	x
rightist.n.01	0.074	politics:0.18	business:0.06	entertainment:0.04	sport:0.01	x
marketplace.n.02	0.087	entertainment:0.22	business:0.06	politics:0.05	sport:0.01	x
PAN (Age) data set						
hippie.n.01	0.007	25-34:0.01	35-49:0.01	18-24:0.0	65-xx:0.0	50-64:0.0
ceremony.n.03	0.007	25-34:0.01	35-49:0.01	18-24:0.01	65-xx:0.0	50-64:0.0
resource.n.02	0.008	50-64:0.02	18-24:0.01	25-34:0.0	65-xx:0.0	35-49:0.0
draw.v.07	0.008	25-34:0.02	35-49:0.01	50-64:0.01	65-xx:0.0	18-24:0.0
observation.n.02	0.008	25-34:0.02	35-49:0.01	50-64:0.01	65-xx:0.0	18-24:0.0
wine.n.01	0.008	35-49:0.02	25-34:0.01	18-24:0.01	50-64:0.01	65-xx:0.0
suck.v.02	0.008	25-34:0.02	50-64:0.02	35-49:0.0	65-xx:0.0	18-24:0.0
sleep.n.03	0.008	25-34:0.02	50-64:0.02	35-49:0.0	65-xx:0.0	18-24:0.0
recognize.v.09	0.009	25-34:0.02	35-49:0.02	18-24:0.0	50-64:0.0	65-xx:0.0
weather.v.04	0.009	25-34:0.02	50-64:0.02	35-49:0.0	18-24:0.0	65-xx:0.0
invention.n.02	0.009	25-34:0.02	35-49:0.01	18-24:0.01	50-64:0.0	65-xx:0.0
yankee.n.03	0.01	50-64:0.02	18-24:0.01	25-34:0.01	35-49:0.0	65-xx:0.0

The other explanation for the notable differences in predictive performance is possibly related to small data set sizes, where only a handful of features can be of relevance and thus notably impact a given classifier's performance. We next discuss the impact of the number of selected semantic features on performance.

5.5. How large semantic space should be considered?

Tables 3 and 4 show that a relatively small number of semantic features are needed for potential performance gains. Note that the number of semantic features that need to be considered is around ≤ 100 in most of the cases. The results indicate that a relatively small proportion of the semantic space carries relevant (additional) information, whereas the remainder potentially introduces noise that degrades the performance. Note that in the limit every term from the taxonomy derived from a given corpus could be considered. In such a scenario, many terms would be irrelevant and would only introduce noise. The experiments conducted in this paper indicate that the threshold for the number of features is in the order of hundreds, yet not more features.

6. Qualitative assessment and explainability of tax2vec

This section discusses the properties of the resulting semantic space in Section 6.1, which is followed by a discussion on the explainability of the proposed tax2vec algorithm in Section 6.2.

6.1. Analysis of the resulting semantic space

In this section we discuss the qualitative properties of the obtained corpus-based taxonomies. We present the results concerning hypernym frequency distributions, as well as the overall structure of an example corpus-based taxonomy.

As the proposed approach is entirely symbolic—each feature can be traced back to a unique hypernym—we explored the feature space qualitatively by exploring the statistical properties of the induced taxonomy using graph-statistical approaches. Here, we modeled hypernym frequency distributions to investigate possible similarity with the Zipf's law (Piantadosi 2014). The analysis was performed using the Py3plex library (Škrlj et al. 2019a). We also visualized the document-based taxonomy of the PAN 2016 (Age) data set using Cytoscape (Shannon et al. 2003).

The examples in this section are all based on the corpus-based taxonomy, constructed from the PAN 2016 (Age) data set. The results of fitting various heavy-tailed distributions to the hypernym frequencies are given in Figure 7.

We fitted power law, truncated normal, log-normal and exponential distributions to the hypernym frequency data. For detailed overview of the distributions we refer the reader to (Foss et al. 2011). One of the key properties we researched was whether the underlying hypernym distribution is exponential or not, as non-exponential distributions indicate similarity with the well known Zipf's law (Piantadosi 2014). The hypernym corpus-based taxonomy is visualized in Figure 8.

Here, each node represents a hypernym obtained in word-to-hypernym mapping phase of tax2vec. The edges represent the hypernymy relation between a given pair of hypernyms.

We next present the results of modeling the corpus-based hypernym frequency distributions. The two functions representing the best fit to hypernym frequency distributions are indeed the power law and the truncated power law. As similar behavior is observed for word frequency in documents (Piantadosi 2014), we believe hypernym distributions are a natural extension, as naturally, if a high-frequency word maps to a given hypernym, the hypernym will be relatively more common with respect to the occurrence of other hypernyms.

We observe that multiple connected components of varying sizes emerge. There exists only a single largest connected component, which consists of more general noun hypernyms, such as *entity* and similar. Interestingly, many smaller components also emerged, indicating parts of the word vector space could be mapped to very specific, disconnected parts of the WordNet taxonomy. Some examples of small disconnected components include (one component per line), indicating also verb-level semantics can be captured and taken into account:

$'spot.v.02', 'discriminate.v.03'\ 'homestead.v.01', 'settle.v.21'$
$'smell.v.05', 'perceive.v.02', 'understand.v.02'$
$'dazzle.v.01', 'blind.v.01'$
$'romance.v.02','adore.v.01','care\_for.v.02','love.v.03','love.v.01'$
$'surrender.v.01', 'yield.v.12', 'capitulate.v.01'$

6.2. Explainability of tex2vec

As discussed in the previous sections, tax2vec selects a set of hypernyms according to a given heuristic and uses them for learning. One of the key benefits of such approach is that the selected semantic features can easily be inspected, hence potentially offering interesting insights into the semantics, underlying the problem at hand.

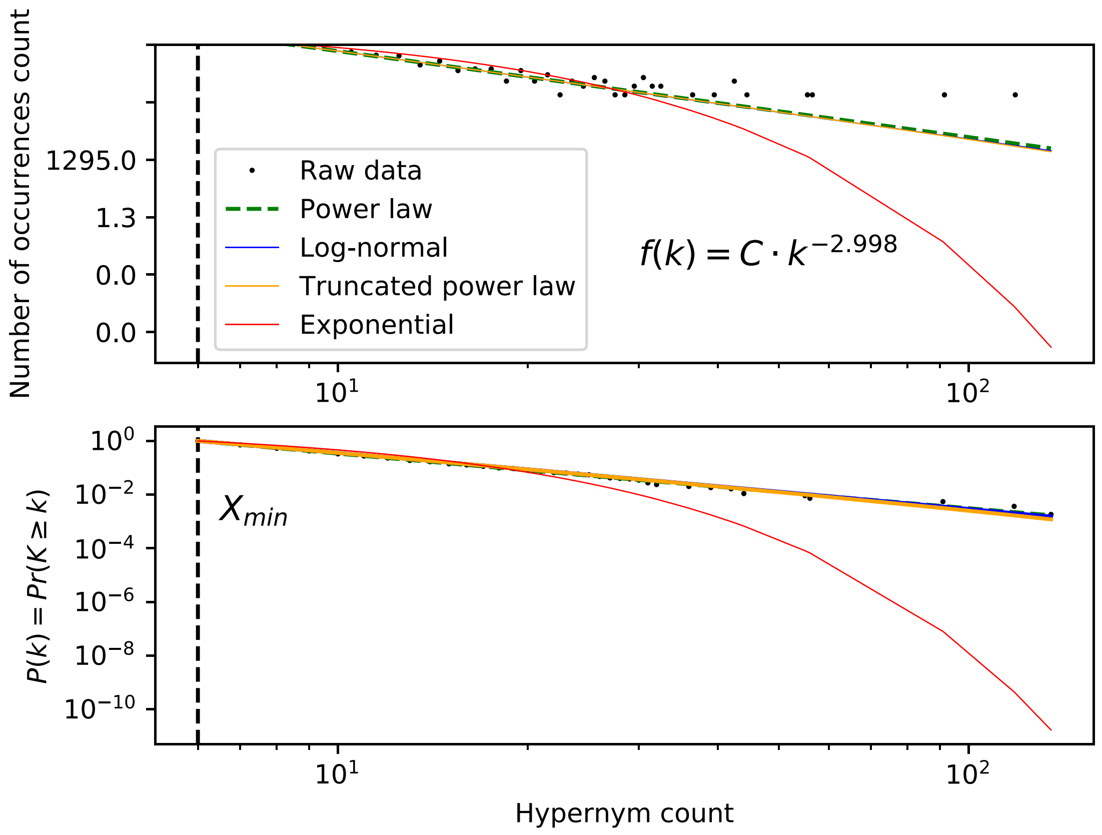


Fig. 7. Hypernym frequency distribution for the PAN 2016 (Age) data set. The equation above the upper plot denotes the coefficients of a power law distribution (C is a constant). In real world phenomena, the exponent of the rightmost expression was observed to range between ≈ 2 and ≈ 3, indicating the hypernym structure of the feature space is subject to a heavy-tailed (possibly best fit—power law) distribution. The X_{min} denotes the hypernym count, after which notable differences in hypernym counts—scale free behavior is observed. Such distribution is to some extent expected, as some hypernyms are more general than others, and thus present in more document-hypernym mappings.

We discuss here a set of 30 features which emerged as relevant according to the “mutual information” heuristic when the BBC News and PAN 2016 (Age) data sets were considered. Here, tax2vec was trained on 90% of the data, the rest was removed (test set). The features and their corresponding mutual information scores are shown in Table 4.

We can observe that the “sport” topic (BBC data set) is not well associated with the prioritized features. On the contrary, terms such as “rightist” and “conservative” emerged as relevant for classifying into the “politics” class. Similarly, “marketplace” for example, appeared relevant for classifying into the “entertainment” class. Even more interesting associations emerged when the same feature ranking was conducted on the PAN 2016 (Age) data set. Here, terms such as “resource” and “wine” were relevant for classifying middle-aged (“wine”) and older adult (“resource”) populations. Note that the older population (65-xx class) was not associated with any of the hypernyms. We believe the reason for this is that the number of available tweets decreases with age.

We repeated a similar experiment (BBC data set) using the “rarest terms” heuristic. The terms which emerged are:

’problem.n.02’, ’question.n.02’, ’riddle.n.01’, ’salmon.n.04’, ’militia.n.02’, ’orphan.n.04’, ’taboo.n.01’, ’desertion.n.01’, ’dearth.n.02’, ’outfitter.n.02’, ’scarcity.n.01’, ’vasodilator.n.01’, ’dilator.n.02’, ’fluoxetine.n.01’, ’high blood pressure.n.01’, ’amlodipine besylate.n.01’, ’drain.n.01’, ’imperative mood.n.01’, ’fluorescent.n.01’, ’veneer.n.01’, ’autograph.n.01’, ’oak.n.02’, ’layout.n.01’, ’wall.n.01’, ’firewall.n.03’, ’workload.n.01’, ’manuscript.n.02’, ’cake.n.01’, ’partition.n.01’, ’plasterboard.n.01’

Even if the feature selection method is unsupervised (not directly associated to classes), we can immediately observe that the features correspond to different topics, raging from medicine (e.g., high blood presure), politics (e.g., militia), food (e.g., cake) and more, indicating that the rarest hypernyms are indeed diverse and as such potentially useful for the learner.

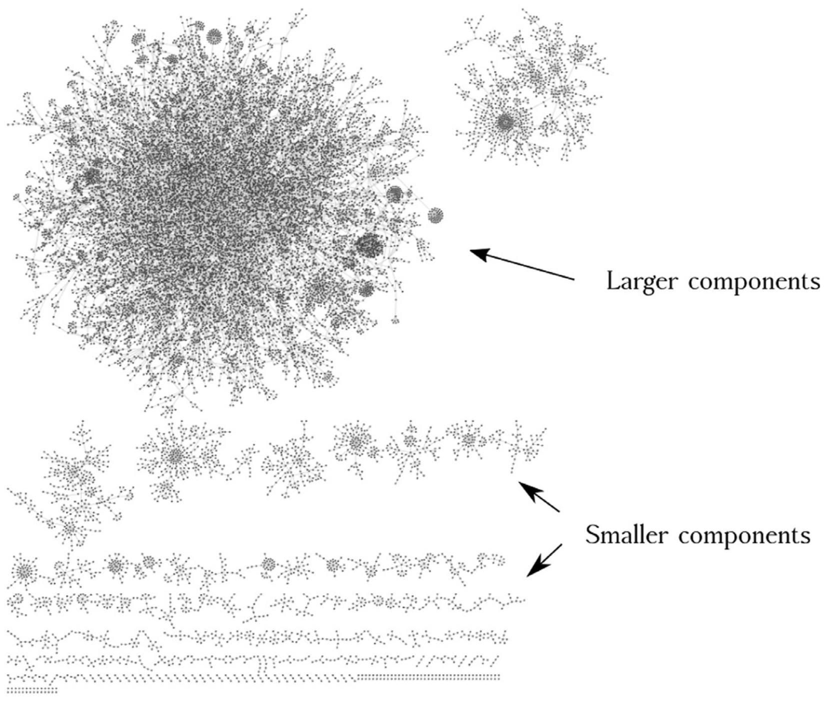


Fig. 8. Topological structure of the hypernym space, induced from the PAN 2016 (Age) data set. Multiple connected components emerged, indicating not all hypernyms map to the same high-level concepts. Such segmentation is data set-specific, and can also potentially provide the means to compare semantic spaces of different data sets. It can be observed that the obtained space is organized in multiple separate components. The largest are drawn at the topmost part of the figure, whereas the smaller ones at the bottom. Such segmentation corresponds to generalizations based on different parts of speech, e.g., nouns and verbs.

The results suggest that tax2vec could potentially also be used to inspect the semantic background of a given data set directly, regardless of the learning task. We believe there are many potential uses for the obtained features, including the following, to be addressed in further work.

- Concept drift detection, i.e. topics change over time; could it be qualitatively detected?
- Topic domination, i.e. what type of topic is dominant with respect to e.g., a geographical region inspected?
- What other learning tasks can benefit by using second level semantics? Can the obtained features be used, for example, for fast keyword search?

7. Implementation and availability

The tax2vec algorithm is implemented in Python 3, where Multiprocessing[11], SciPy (Jones et al. (2001–) and Numpy (Walt et al. 2011) libraries are used for fast (sparse), vectorized operations and parallelism.

As performing a grid search over several parameters is computationally expensive, the majority of the experiments were conducted using the SLING supercomputing architecture.[12]

We developed a stand-alone library that relatively seamlessly fits into existing text mining workflows, hence the Scikit-learn's model syntax was adopted (Pedregosa et al. 2011). The algorithm is first initiated as an object:

$$\text{vectorizer} = \text{tax2vec}(\text{heuristic}, \text{number of features})$$

followed by standard *fit* and *transform* calls:

$$\text{new\_features} = \text{vectorizer.fit\_transform}(\text{corpus}, \text{ optional labels})$$

Such implementation offers fast prototyping capabilities, needed ubiquitously in the development of learning algorithms and executable NLP and text mining workflows.

The proposed tax2vec approach is freely available as a Python 3 library at https://github.com/SkBlaz/tax2vec, which includes also the installation instructions.

[11] https://docs.python.org/2/library/multiprocessing.html
[12] http://www.sling.si/sling/

8. Conclusions and future work

In this work we propose tax2vec, a parallel algorithm for taxonomy-based enrichment of text documents. Tax2vec first maps the words from individual documents to their hypernym counterparts, which are considered as candidate features and weighted according to a normalized tf-idf metric. To select only a user-specified number of relevant features, tax2vec implements multiple feature selection heuristics, which select only the potentially relevant features. The sparse matrix of constructed features is finally used alongside the bag-of-words document representations for the task of text classification, where we study its performance on small data sets, where both the number of text segments per user, as well as the number of overall users considered are small.

The tax2vec approach considerably improves the classification performance especially on data sets consisting of tweets, but also on the news. The proposed implementation offers a simple-to-use API, which facilitates inclusion into existing text preprocessing workflows.

As the next step, the tax2vec will be tested on SMS spam data (Delany et al. 2012), which is another potentially interesting short text data set where taxonomy-based features could improve performance and help the user better understand what classifies as spam (and what not).

One of the drawbacks we plan to address is the support for arbitrary directed acyclic multigraphs—structures commonly used to represent background knowledge. Support for such knowledge would offer a multitude of applications in e.g., biology, where gene ontology and other resources which annotate entities of interest are freely available.

In this work we focus on BoW representation of documents, yet we believe tax2vec could also be used along Continuous Bag-of-Words (CBoW) models. We leave such experimentation for further work.

Even though we use Lesk for the disambiguation task, we believe recent advancements in neural disambiguation (Iacobacci et al. 2016) could also be a "drop-in" replacement for this part of tax2vec. We leave the exploration of such options for further work.

In this work we explored how WordNet could be adapted for scalable feature construction, however tax2vec is by no means limited to manually curated relational (hierarchical) structures. As part of the further work, we believe feature construction based on *knowledge graphs* could also be an option.

The abundance of neural embedding methods introduced in the recent years can be complementary to tax2vec. Understanding how the performance can be improved by jointly using both tax2vec's features and neural network-based ones is a potential interesting research opportunity. Further, in NLP setting, not much attention is devoted to this topic, thus we believe these results offer new trajectories for few-shot learning research.

Other further work considers joining the tax2vec features with existing state-of-the-art deep learning approaches, such as the hierarchical attention networks, which are—according to this study—not very suitable for learning on scarce data sets. We believe that the introduction of semantics into deep learning could be beneficial for both performance, as well as the interpretability of currently poorly understood black-box models.

Finally, as the main benefit of tax2vec is its explanatory power, we believe it could be used for fast keyword search; here, for example, new news or articles could be used as inputs, where the ranked list of semantic features could be directly used as candidate keywords.

Acknowledgements

We would first like to thank the reviewers for insightful comments that improved this paper. The work of the first author was funded by the Slovenian Research Agency through a young researcher grant. The work of other authors was supported by the Slovenian Research Agency (ARRS) core research programme *Knowledge Technologies* (P2-0103), an ARRS funded research project *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078) and European Union's Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media), and under grant agreement No 769661, SAAM project. We also gratefully acknowledge the support of NVIDIA Corporation for the donation of Titan-XP GPU. The initial version of this paper was written while Senja Pollak was a research fellow at the Usher Institute, University of Edinburgh.

Appendix A. Personalized PageRank algorithm

The Personalized PageRank (PPR) algorithm is described below. Let V represent the nodes of the corpus-based taxonomy. For each node $u \in V$, a feature vector is computed by calculating the stationary distribution of a random walk, starting at node u. The stationary distribution is approximated by using power iteration, where the i-th component of the approximation in the k-th iteration is computed as

$$\gamma_u(i)^{(k+1)} = \alpha \cdot \sum_{j \to i} \frac{\gamma_u(j)^{(k)}}{d_j^{out}} + (1-\alpha) \cdot v_u(i); k = 1, 2, \cdots \tag{A.1}$$

The number of iterations k is increased until the stationary distribution converges to the stationary distribution vector (PPR value for node i). In the above equation, α is the damping factor that corresponds to the probability that a random walk follows a randomly chosen outgoing edge from the current node rather than restarting its walk. The summation index j runs over all nodes of the network that have an outgoing connection toward j, (denoted as $j \to i$ in the sum), and d_j^{out} is the out degree of node d_j. The term $v_u(i)$ is the restart distribution that corresponds to a vector of probabilities for a walker's return to the starting node u, i.e. $v_u(u) = 1$ and $v_u(i) = 0$ for $i \neq u$. This vector guarantees that the walker will jump back to the starting node u in case of a restart.[13]

Appendix B. Example document split

While for the data sets consisting of tweets and short comments, the number of segments in a document corresponds to the number of tweets or comments by a user, in the news data set, we varied the size of the news (to create short documents) by splitting the news into paragraphs (we denote such paragraph splits with |||). An example of segmentation of a news from the BBC data set[14] is listed below.

The decision to keep interest rates on hold at 4.75% earlier this month was passed 8-1 by the Bank of England's rate-setting body, minutes have shown.||| One member of the Bank's Monetary Policy Committee (MPC) - Paul Tucker - voted to raise rates to 5%. The news surprised some analysts who had expected the latest minutes to show another unanimous decision. Worries over growth rates and consumer spending were behind the decision to freeze rates, the minutes showed. The Bank's latest inflation report, released last week, had noted that the main reason inflation might fall was weaker consumer spending.||| However, MPC member Paul Tucker voted for a quarter point rise in interest rates to 5%. He argued that economic growth was picking up, and that the equity, credit and housing markets had been stronger than expected.||| The Bank's minutes said that risks to the inflation forecast were "sufficiently to the downside" to keep rates on hold at its latest meeting. However, the minutes added: "Some members noted that an increase might be warranted in due course if the economy evolved in line with the central projection". Ross Walker, UK economist at Royal Bank of Scotland, said he was surprised that a dissenting vote had been made so soon. He said the minutes appeared to be "trying to get the market to focus on the possibility of a rise in rates". "If the economy pans out as they expect then they are probably going to have to hike rates." However, he added, any rate increase is not likely to happen until later this year, with MPC members likely to look for a more sustainable pick up in consumer spending before acting.

This news article is split by a parser into the following four segments (and in short document setting only one paragraph is used to represent the document).

- The decision to keep interest rates on hold at 4.75% earlier this month was passed 8-1 by the Bank of England's rate-setting body, minutes have shown.
- One member of the Bank's Monetary Policy Committee (MPC) - Paul Tucker - voted to raise rates to 5%. The news surprised some analysts who had expected the latest minutes to show another unanimous decision. Worries over growth rates and consumer spending were behind the decision to freeze rates, the minutes showed. The Bank's latest inflation report, released last week, had noted that the main reason inflation might fall was weaker consumer spending.
- However, MPC member Paul Tucker voted for a quarter point rise in interest rates to 5%. He argued that economic growth was picking up, and that the equity, credit and housing markets had been stronger than expected.
- The Bank's minutes said that risks to the inflation forecast were"sufficiently to the downside" to keep rates on hold at its latest meeting. However, the minutes added: "Some members noted that an increase might be warranted in due course if the economy evolved in line with the central projection." Ross Walker, UK economist at Royal Bank of Scotland, said he was surprised that a dissenting vote had been made so soon. He said the minutes appeared to be "trying to get the market to focus on the possibility of a rise in rates." "If the economy pans out as they expect then they are probably going to have to hike rates." However, he added, "any rate increase is not likely to happen until later this year, with MPC members likely to look for a more sustainable pick up in consumer spending before acting."

Appendix C. Impact of different number of features across data sets

Impact of the number of features on the F1 performance of the SVM (generic) classifier are shown in Figure C1.

[13] Note that if the binary vector were instead composed exclusively of ones, the iteration would compute the global PageRank vector, and Equation A.1 would correspond to the standard PageRank iteration.
[14] https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv

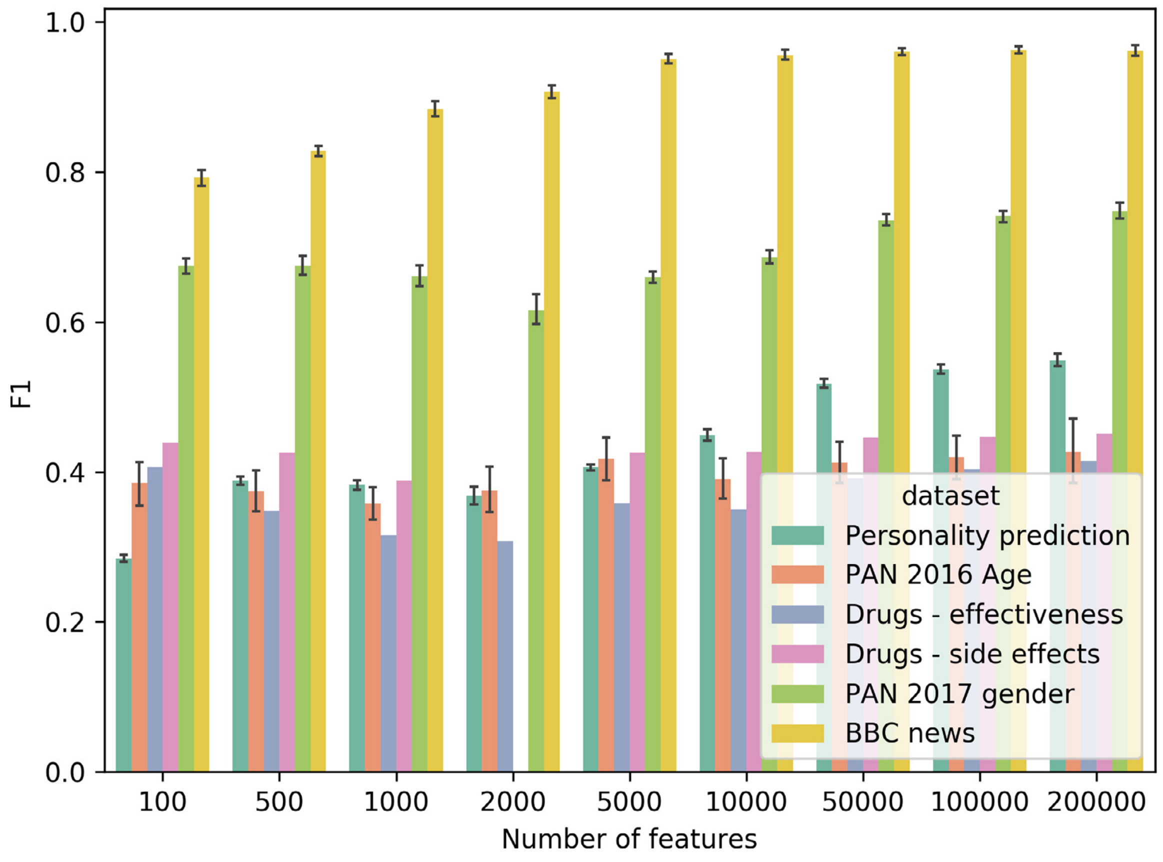


Fig. C1. Impact of the number of features used by the SVM (generic) on the F1 performance. The best performances were observed for feature numbers (word tokens) $\geq$ 10,000, hence these feature numbers were considered in the more expensive experiment stage with semantic vectors.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Man&x00E9;, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.
Adhikari, P.R., Vavpetič, A., Kralj, J., Lavrač, N., Hollmén, J., 2016. Explaining mixture models through semantic pattern mining and banded matrix visualization. Machine Learning 105 (1), 3–39.
Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., Rudin, C., 2017. Learning certifiably optimal rule lists. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 35–44.
Asim, M.N., Khan, M.U.G., Malik, M.I., Dengel, A., Ahmed, S., 2016. A robust hybrid approach for textual document classification. arXiv preprint arXiv:1909.05478.
Basile, P., Caputo, A., Semeraro, G., 2014. An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pp. 1591–1600.
Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent dirichlet allocation. Journal of machine Learning research 3 (1), 993–1022.
Boyce, R., Gardner, G., Harkema, H., 2012. Using natural language processing to identify pharmacokinetic drug-drug interactions described in drug package inserts. In: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing. Association for Computational Linguistics, pp. 206–213.
Brandes, U., 2001. A faster algorithm for betweenness centrality. The Journal of Mathematical Sociology 25 (2), 163–177.
Bunge, M., 2017. Causality and Modern Science. Routledge.
Cagliero, L., Garza, P., 2013. Improving classification models with taxonomy information. Data & Knowledge Engineering 86, 85–101.
Chandrashekar, G., Sahin, F., 2014. A survey on feature selection methods. Computers & Electrical Engineering 40 (1), 16–28.
Chang, C.-C., Lin, C.-J., 2011. Libsvm: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2 (3), 27.
Chang, S., Han, W., Tang, J., Qi, G.-J., Aggarwal, C.C., Huang, T.S., 2015. Heterogeneous network embedding via deep architectures. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 119–128.
Chen, M., Jin, X., Shen, D., 2011. Short text classification improved by learning multi-granularity topics. Twenty-Second International Joint Conference on Artificial Intelligence.
Cho, J., Lee, K., Shin, E., Choy, G., Do, S., 2016. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?. arXiv preprint arXiv:1511.06348.
Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S., 2010. Who is tweeting on twitter: human, bot, or cyborg? In: Proceedings of the 26th annual computer security applications conference. ACM, pp. 21–30.
Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S., 2012. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? IEEE Transactions on Dependable and Secure Computing 9 (6), 811–824.
Delany, S.J., Buckley, M., Greene, D., 2012. Sms spam filtering: Methods and data. Expert Systems with Applications 39 (10), 9899–9908.
Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine learning research 7 (Jan), 1–30.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186.
Dong, Y., Chawla, N.V., Swami, A., 2017. Metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 135–144.
Elhadad, M.K., Badran, K.M., Salama, G.I., 2018. A novel approach for ontology-based feature vector generation for web text document classification. International Journal of Software Innovation (IJSI) 6 (1), 1–10.
Fellbaum, C. (Ed.), 1998. WordNet: An Electronic Lexical Database. MIT Press, Cambridge, MA.
Foss, S., Korshunov, D., Zachary, S., et al., 2011. An introduction to Heavy-tailed and Subexponential Distributions. 6. Springer.
Freeman, L.C., 2017. Research Methods in Social Network Analysis. Routledge.
Gonzalez-Agirre, A., Laparra, E., Rigau, G., 2012. Multilingual central repository version 3.0: upgrading a very large lexical knowledge base. In: Proceedings of the 6th Global WordNet Conference (GWC 2012), Matsue.online
Grässer, F., Kallumadi, S., Malberg, H., Zaunseder, S., 2018. Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In: Proceedings of the 2018 International Conference on Digital Health. ACM, New York, NY, USA, pp. 121–125.
Greene, D., Cunningham, P., 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In: Proceedings of the 23rd International Conference on Machine learning (ICML'06). ACM Press, pp. 377–384.
Grover, A., Leskovec, J., 2016. Node2vec: Scalable feature learning for networks. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 855–864.
Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 1024–1034.
Hira, Z.M., Gillies, D.F., 2015. A review of feature selection and feature extraction methods applied on microarray data. Advances in bioinformatics 2015.
Iacobacci, I., Pilehvar, M.T., Navigli, R., 2016. Embeddings for word sense disambiguation: An evaluation study. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, 1, pp. 897–907.
Jaeger, S., Fulle, S., Turk, S., 2018. Mol2vec: Unsupervised machine learning approach with chemical intuition. Journal of Chemical Information and Modeling 58 (1), 27–35.
Kakisim, A.G., Sogukpinar, I., 2019. Unsupervised binary feature construction method for networked data. Expert Systems with Applications 121, 256–265.
Kaur, R., Kumar, M., 2018. Domain ontology graph approach using markov clustering algorithm for text classification. International Conference on Intelligent Computing and Applications. Springer, pp. 515–531.
Kim, C., Yin, P., Soto, C.X., Blaby, I.K., Yoo, S., 2018. Multimodal biological analysis using NLP and expression profile. 2018 New York Scientific Data Summit (NYSDS), pp. 1–4.
Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. International Conference on Learning Representations (ICLR), p. online.
Kralj, J., 2017. Heterogeneous Information Network Analysis for Semantic Data Mining: Doctoral Dissertation Ph.D. thesis.
Kralj, J., Robnik-Šikonja, M., Lavrač, N., 2019. NetSDM: Semantic data mining with network analysis. Journal of Machine Learning Research 20 (32), 1–50.
Kralj Novak, P., Smailović, J., Sluban, B., Mozetič, I., 2015. Sentiment of emojis. PLOS ONE 10 (12), 1–22. https://doi.org/10.1371/journal.pone.0144296.
Kusner, M., Sun, Y., Kolkin, N., Weinberger, K., 2015. From word embeddings to document distances. International Conference on Machine Learning, pp. 957–966.
Landauer, T.K., 2006. Latent Semantic Analysis. Wiley Online Library.
Ławrynowicz, A., 2017. Semantic Data Mining: An Ontology-based Approach. 29. IOS Press.
Le, Q., Mikolov, T., 2014. Distributed representations of sentences and documents. International conference on machine learning, pp. 1188–1196.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436.
Manning, C.D., Raghavan, P., Schütze, H., 2008. Scoring, term weighting, and the vector space model. Cambridge University Press, pp. 100–123.Ch. first
Mansuy, T.N., Hilderman, R.J., 2006. Evaluating wordnet features in text classification models. FLAIRS Conference, pp. 568–573.
Martinc, M., Škrlj, B., Pollak, S., 2019. Fake or not: Distinguishing between bots, males and females. In: Proceedings of the Tenth International Conference of the CLEF Association (CLEF 2019), p. online.
Martinc, M., Škrjanec, I., Zupan, K., Pollak, S., 2017. PAN 2017: Author profiling - gender and language variety prediction. Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum. CEUR Workshop Proceedings, p. 1866. online.
Meyer, D., Leisch, F., Hornik, K., 2003. The support vector machine under test. Neurocomputing 55 (1), 169–186. https://doi.org/10.1016/S0925-2312(03)00431-4.
Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 3111–3119.
Miller, G.A., 1995. Wordnet: A lexical database for english. Commun. ACM 38 (11), 39–41.
Myers, I. B., 1962. The Myers-Briggs type indicator: Manual.
Navigli, R., 2009. Word sense disambiguation: A survey. ACM Comput. Surv. 41 (2), 10:1–10:69.
Pearl, J., 2009. Causality. Cambridge University Press.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al., 2011. Scikit-learn: Machine learning in Python. Journal of machine learning research 12 (Oct), 2825–2830.
Peng, H., Long, F., Ding, C., 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence 27 (8), 1226–1238.
Pennington, J., Socher, R., Manning, C., 2014. Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical methods in Natural Language Processing (EMNLP), pp. 1532–1543.
Perovšek, M., Vavpetič, A., Cestnik, B., Lavrač, N., 2013. A wordification approach to relational data mining. International Conference on Discovery Science. Springer, pp. 141–154.
Perovšek, M., Vavpetič, A., Kranjc, J., Cestnik, B., Lavrač, N., 2015. Wordification: Propositionalization by unfolding relational data into bags of words. Expert Systems with Applications 42 (17-18), 6442–6456.
Piantadosi, S.T., 2014. Zipf's word frequency law in natural language: A critical review and future directions. Psychonomic Bulletin & Review 21 (5), 1112–1130.
Rangel, F., Rosso, P., Cappellato, L., Ferro, N., Müller, H., Losada, D., 2019. Overview of the 7th author profiling task at pan 2019: Bots and gender profiling. CLEF, p. online.
Rangel, F., Rosso, P., Chugur, I., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daelemans, W., 2014. Overview of the 2nd author profiling task at PAN 2014. Working Notes Papers of the CLEF 2014, pp. 1–30.
Rangel, F., Rosso, P., Potthast, M., Stein, B., 2017. Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. Working Notes Papers of the CLEF.
Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., Stein, B., 2016. Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al., pp. 750–784.
Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R., 2017. Struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, USA, pp. 385–394.
Sapkota, U., Bethard, S., Montes-y-Gómez, M., Solorio, T., 2015. Not all character n-grams are created equal: A study in authorship attribution. NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015, pp. 93–102.
Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural networks 61, 85–117.
Scott, S., Matwin, S., 1998. Text classification using wordnet hypernyms. Usage of WordNet in Natural Language Processing Systems.

Sebastiani, F., 2002. Machine learning in automated text categorization. ACM Comput. Surv. 34 (1), 1–47.
Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T., 2003. Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Research 13 (11), 2498–2504.
Škrlj, B., Kralj, J., Lavrač, N., 2019a. Py3plex: A library for scalable multilayer network analysis and visualization. Complex Networks and Their Applications VII. Springer International Publishing, Cham, pp. 757–768.
Škrlj, B., Lavrač, N., Kralj, J., 2019b. Symbolic graph embedding using frequent pattern mining. In: Kralj Novak, P., Šmuc, T., Džeroski, S. (Eds.), Discovery Science. Springer International Publishing, Cham, pp. 261–275.
Snell, J., Swersky, K., Zemel, R., 2017. Prototypical networks for few-shot learning. Advances in Neural Information Processing Systems, pp. 4077–4087.
Socher, R., Ganjoo, M., Manning, C.D., Ng, A., 2013. Zero-shot learning through cross-modal transfer. In: Proceedings of the Advances in neural information processing systems, pp. 935–943.
Stańczyk, U., Jain, L.C., 2015. Feature selection for Data and Pattern Recognition. Springer.
Tang, D., Qin, B., Liu, T., 2015. Document modeling with gated recurrent neural network for sentiment classification. In: Proceedings of the 2015 conference on empirical methods in natural language processing, pp. 1422–1432.
Tomašev, N., Buza, K., Marussy, K., Kis, P.B., 2015. Hubness-aware classification, Instance Selection and Feature Construction: Survey and Extensions to Time-series. Feature selection for data and pattern recognition. Springer, pp. 231–262.
Trieu, L.Q., Tran, H.Q., Tran, M.-T., 2017. News classification from social media using twitter-based doc2vec model and automatic query expansion. In: Proceedings of the Eighth International Symposium on Information and Communication Technology, pp. 460–467.
Vavpetič, A., Lavrač, N., 2013. Semantic subgroup discovery systems and workflows in the sdm-toolkit. The Computer Journal 56 (3), 304–320.
Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al., 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. Nature methods 17 (3), 261–272.
Walt, S.v. d., Colbert, S.C., Varoquaux, G., 2011. The numpy array: a structure for efficient numerical computation. Computing in Science & Engineering 13 (2), 22–30.
Wang, J., Wang, Z., Zhang, D., Yan, J., 2017. Combining knowledge with deep convolutional neural networks for short text classification. In: Proceedings of IJCAI, 350, p. online.
Xu, N., Wang, J., Qi, G., Huang, T.S., Lin, W., 2018. Ontological random forests for image classification. Computer Vision: Concepts, Methodologies, Tools, and Applications. IGI Global, pp. 784–799.
Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E., 2016. Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1480–1489.
Železnỳ, F., Lavrač, N., 2006. Propositionalization-based relational subgroup discovery with RSD. Machine Learning 62 (1-2), 33–63.

2.5 Combining Symbolic Features with Neural Semantic Features

As was already mentioned above, one of the main deficiencies of AP approaches based on BON features is their inability to model sequential information and their consequential limited grasp of semantic information. On the other hand, neural approaches are very successful at modelling semantics of text but lack an effective weighting scheme capable of determining the importance of specific words (and the information they carry) in the document.

To alleviate the above-mentioned problems of both methods, in Martinc and Pollak (2019) we combined sophisticated feature engineering techniques used in traditional approaches to text classification with the neural automatic feature construction, in order to achieve synergy between these two feature types. **To put it differently, in this section we aim to achieve the stated goal G1 and confirm the hypothesis H2.**

In this study, the focus was on the task of discriminating between similar languages and the proposed method was tested by distinguishing between varieties of 8 different languages. According to the research on neural approaches towards author profiling presented in Section 2.2, we opted to use character-level CNNs, which proved efficient in previous work on discriminating between similar languages. CNNs are able to extract the most important character sequences of a text by employing a max-over-time pooling operation (Collobert et al., 2011b). These sequences resemble character n-grams that were used in nearly every winning approach in the past shared task, which partially explains the good performance of CNNs on previous language variety classification tasks. Additionally, CNNs preserve the order in which the text areas with high predictive power appear in the text.

To compensate for the lack of an effective weighting scheme and global document/corpus-level information available to the network, besides feeding the network a character sequence, from which convolutional features are generated, we propose to feed the network an additional input in the form of a *TF-IDF/BM25 matrix*. Same as in the proposed tax2vec approach above, here we also employ early simple fusion and the TF-IDF/BM25 matrix is concatenated to the flattened (changed from a two-dimensional to a one dimensional vector) convolutional features. The resulting concatenation is fed to a set of fully connected feed forward layers and activation layers responsible for producing the final probability distribution over language variety classes. The entire architecture of the system is presented in Figure 2.3.

The proposed architecture is a hybrid between a traditional feature engineering approach, which relies on different kinds of weighted BON features, and a convolutional approach to text classification. Combining two distinct text classification methods allows the architecture to leverage character-level and more global document/corpus-level information.

The method was tested on eight languages, each of them containing several varieties. We report results on the DSL Corpus Collection (DSLCC) v4.0 (Tan et al., 2014) used in VarDial 2017 (Zampieri et al., 2017), which contains six language groups. We also tested the methodology on two smaller corpora, the Arabic Dialect Identification Corpus (ADIC) used in a VarDial 2016 ADI shared task (Malmasi et al., 2016) and the German Dialect Identification Corpus (GDIC) used in a VarDial 2018 GDI shared task (Zampieri et al., 2018) in order to determine how dataset size and characteristics affect the competitiveness of the proposed system.

For experiments on the DSLCC v4.0, we chose to use a two-step approach, as first proposed by Goutte et al. (2014), where in the first step the general classifier is trained to identify the language group for every specific document. In the second step, six different classification models are trained, one for each language group. After a document is classified as belonging to a specific language group by the general classifier in the first step, it is assigned to the appropriate classifier, which generates the final language variety prediction.

The results of the experiments on the DSLCC v4.0 are presented in Table 2.3. While distinguishing between different language groups (*All-language groups (TF-IDF)* and *All-language groups (BM25)* rows in Table 2.3) is trivial for the system, which achieves almost

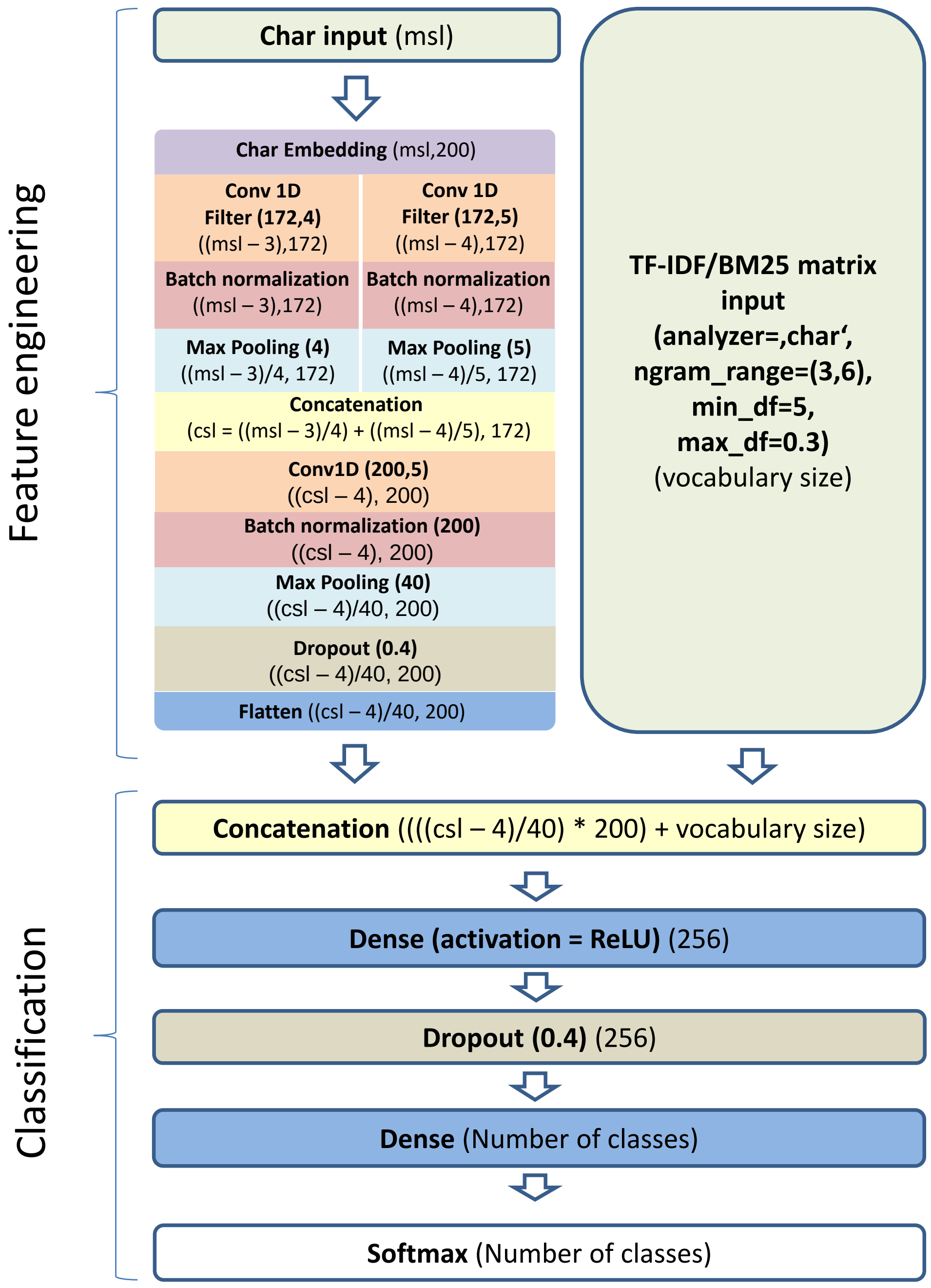

Feature engineering
Classification
Char input (msl)
Char Embedding (msl,200)
Conv 1D Filter (172,4) ((msl – 3),172)
Conv 1D Filter (172,5) ((msl – 4),172)
Batch normalization ((msl – 3),172)
Batch normalization ((msl – 4),172)
Max Pooling (4) ((msl – 3)/4, 172)
Max Pooling (5) ((msl – 4)/5, 172)
Concatenation (csl = ((msl – 3)/4) + ((msl – 4)/5), 172)
Conv1D (200,5) ((csl – 4), 200)
Batch normalization (200) ((csl – 4), 200)
Max Pooling (40) ((csl – 4)/40, 200)
Dropout (0.4) ((csl – 4)/40, 200)
Flatten ((csl – 4)/40, 200)
TF-IDF/BM25 matrix input (analyzer=,char‘, ngram_range=(3,6), min_df=5, max_df=0.3) (vocabulary size)
Concatenation ((((csl – 4)/40) * 200) + vocabulary size)
Dense (activation = ReLU) (256)
Dropout (0.4) (256)
Dense (Number of classes)
Softmax (Number of classes)

Figure 2.3: Architecture of the proposed neuro-symbolic language variety classifier: layer names and input parameters are written in bold, layer output sizes are written in normal text, *msl* stands for maximum sequence length and *csl* stands for concatenated sequence length.

Table 2.3: Results of the proposed neuro-symbolic language variety classifier on the DSLCC v4.0 for different language groups, as well as for the discrimination between language groups (All-language groups). Also the results for all language varieties (All-language varieties) are provided, for which a comparison with the official VarDial 2017 winners is made. Results for both weighting schemes, TF-IDF and BM25, are reported separately.

Language group (weighting)	F_1 (weighted)	F_1 (micro)	F_1 (macro)	Accuracy
All-language groups (TF-IDF)	**0.9981**	**0.9981**	**0.9980**	**0.9981**
All-language groups (BM25)	0.9979	0.9979	0.9980	0.9980
Spanish (TF-IDF)	**0.9136**	**0.9140**	**0.9136**	**0.9140**
Spanish (BM25)	0.9042	0.9047	0.9042	0.9047
Slavic (TF-IDF)	0.8645	0.8650	0.8645	0.8650
Slavic (BM25)	**0.8752**	**0.8753**	**0.8752**	**0.8753**
Farsi (TF-IDF)	0.9685	0.9685	0.9685	0.9685
Farsi (BM25)	**0.9690**	**0.9690**	**0.9690**	**0.9690**
French (TF-IDF)	**0.9570**	**0.9570**	**0.9570**	**0.9570**
French (BM25)	0.9545	0.9545	0.9545	0.9545
Malay and Indonesian (TF-IDF)	0.9855	0.9855	0.9855	0.9855
Malay and Indonesian (BM25)	**0.9860**	**0.9860**	**0.9860**	**0.9860**
Portuguese (TF-IDF)	**0.9480**	**0.9480**	**0.9480**	**0.9480**
Portuguese (BM25)	0.9460	0.9460	0.9460	0.9460
All-language varieties (TF-IDF)	**0.9310**	**0.9312**	**0.9310**	**0.9312**
All-language varieties (BM25)	0.9304	0.9305	0.9304	0.9305
VarDial 2017 winner (Bestgen, 2017)	0.9271	0.9274	0.9271	0.9274

perfect weighted F_1 score, results for the second step of the two-step classification approach show that the difficulty of distinguishing language varieties within different language groups varies. The system had most difficulties with distinguishing between different Slavic languages, where it achieved by far the worst results in terms of weighted F_1 no matter the weighting scheme implemented. The second most difficult were Spanish varieties and the system had the least problems with distinguishing between Malay and Indonesian languages.

When it comes to comparing two weighting schemes, the differences are rather small for all varieties, suggesting that the choice of a weighting regime does not have a large impact on the performance of the system. Overall (rows *All-language varieties (TF-IDF)* and *All-language varieties (BM25)* in Table 2.3), the neural network outperforms the SVM-based approach used by the winners of the shared task by about 0.4 percentage points according to all measures when TF-IDF weighting scheme is used.

The results of the proposed approach on the ADIC and GDIC corpora in comparison to the winners of the VarDial ADI 2016 and VarDial GDI 2018 shared tasks are presented in Table 2.4. While the proposed system performs well on the ADIC dataset, which is much smaller (in terms of number of documents per class) than the corpora in the DSLCC v4.0.0, beating the state-of-the-art by a narrow margin when TF-IDF weighting is used, it does not compare favorably to the state-of-the-art on the GDIC dataset, where it performs almost six percentage points lower than the HeLI method (T. S. Jauhiainen et al., 2018a) in terms of macro F_1 score. This is in line with the hypothesis that neural and hybrid approaches are more sensitive to the amount of data available and do not perform better than SVMs and other symbolic approaches on smaller datasets.

Table 2.4: Results of the proposed neuro-symbolic language variety classifier on the ADIC and GDIC. Results for both weighting schemes, TF-IDF and BM25, are reported separately. The results in bold indicate the best performing system for a specific dataset when a baseline is available.

Language group (weighting)	F_1 (weighted)	F_1 (micro)	F_1 (macro)	Accuracy
ADIC (TF-IDF)	**0.5152**	0.5123	0.5147	**0.5123**
ADIC (BM25)	0.5090	0.5097	0.5067	0.5097
VarDial ADI 2016 winner (Malmasi & Zampieri, 2016)	0.5132	/	/	0.5117
GDIC (TF-IDF)	0.6281	0.6294	0.6280	0.6294
GDIC (BM25)	0.6289	0.6311	0.6289	0.6311
VarDial GDI 2018 winner (T. S. Jauhiainen et al., 2018a)	/	/	**0.6860**	/

Table 2.5: Results of the ablation study. Column *CNN F_1 (weighted)* presents performance of the system in terms of weighted F_1 if only CNN-based features are used, column *BON F_1 (weighted)* presents performance of the system if only TF-IDF-weighted BON features are used and column *All F_1 (weighted)* presents the performance when these two types of features are combined.

Language group	All F_1 (weighted)	CNN F_1 (weighted)	BON F_1 (weighted)
DSLCC v4.0			
All-language groups	**0.9981**	0.9971	0.9976
Spanish	**0.9136**	0.8599	0.8863
Slavic	**0.8645**	0.8300	0.8594
Farsi	**0.9685**	0.9465	0.9610
French	**0.9570**	0.9325	0.9420
Malay and Indonesian	0.9855	0.9560	**0.9875**
Portuguese	**0.9480**	0.8994	0.9434
All-language varieties	**0.9310**	0.8935	0.9199
ADIC	0.5152	0.3971	**0.5177**
GDIC	**0.6281**	0.6059	0.6190

In Martinc and Pollak (2019) we also conduct an ablation study to determine the contribution of the neural and symbolic features. The results of the study are presented in Table 2.5. For these experiments, we first removed parts of the system dealing with the convolutional processing of the character sequence input to measure the contribution of weighted BON features. Second, we removed the symbolic TF-IDF/BM25 matrix input to measure the contribution of the CNN-generated neural features (only TF-IDF weighting was employed in the ablation study).

Note that a classifier with symbolic features (BON classifier) in all cases outperforms the classifier with neural features (CNN classifier). The difference in performance is the largest in the case of ADIC, where the difference is almost eleven percentage points. The difference in performance is the smallest for the French language variety classification in the DSLCC v4.0. corpus, only around one percentage point. By combining both types of features, we manage to outperform the BON classifier on all but one language group in the DSLCC v4.0 and also on the GDIC corpus. The synergy effect between two feature types is the largest in case of the Spanish language variety. The entire study with all the details is enclosed below.

Natural Language Engineering (2019), **25**, pp. 607–632
doi:10.1017/S1351324919000299

CAMBRIDGE
UNIVERSITY PRESS

ARTICLE

Combining *n*-grams and deep convolutional features for language variety classification

Matej Martinc[1*] and Senja Pollak[1,2]

[1]Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia and
[2]Usher Institute of Population Health Sciences and Informatics, Edinburgh Medical School, Usher Institute, University of Edinburgh, Edinburgh, UK
[*]Corresponding author. Email: matej.martinc@ijs.si

(Received 05 November 2018; revised 12 April 2019; accepted 12 May 2019; first published online 18 July 2019)

Abstract
This paper presents a novel neural architecture capable of outperforming state-of-the-art systems on the task of language variety classification. The architecture is a hybrid that combines character-based convolutional neural network (CNN) features with weighted bag-of-*n*-grams (BON) features and is therefore capable of leveraging both character-level and document/corpus-level information. We tested the system on the Discriminating between Similar Languages (DSL) language variety benchmark data set from the VarDial 2017 DSL shared task, which contains data from six different language groups, as well as on two smaller data sets (the Arabic Dialect Identification (ADI) Corpus and the German Dialect Identification (GDI) Corpus, from the VarDial 2016 ADI and VarDial 2018 GDI shared tasks, respectively). We managed to outperform the winning system in the DSL shared task by a margin of about 0.4 percentage points and the winning system in the ADI shared task by a margin of about 0.2 percentage points in terms of weighted F1 score without conducting any language group-specific parameter tweaking. An ablation study suggests that weighted BON features contribute more to the overall performance of the system than the CNN-based features, which partially explains the uncompetitiveness of deep learning approaches in the past VarDial DSL shared tasks. Finally, we have implemented our system in a workflow, available in the ClowdFlows platform, in order to make it easily available also to the non-programming members of the research community.

Keywords: language variety; author profiling; text classification; convolutional neural network; bag-of-n-grams

1. Introduction

Author profiling (AP), which deals with learning about the demographics of a person based on the text she or he produced, is becoming a strong trend in the field of natural language processing (NLP). Tasks such as age, gender, and language variety prediction (automatic distinction between similar dialects or languages) are becoming increasingly popular, in part also because of the marketing potential of this research. Most AP research communities are centered around a series of scientific events and shared tasks on digital text forensics, the two most popular being the evaluation campaign VarDial (Varieties and Dialects)[a] (Zampieri *et al.* 2014), focused on tasks related to the study of linguistic variation, and an event called PAN (Uncovering Plagiarism, Authorship, and Social Software Misuse)[b], which first took place in 2011 and was followed by a series of shared tasks organized since 2013 (Rangel *et al.* 2013).

[a]http://corporavm.uni-koeln.de/vardial/sharedtask.html
[b]http://pan.webis.de/

Table 1. Winning systems for AP classification tasks in PAN AP and VarDial DSL shared tasks (language variety tasks in bold)

Year	VarDial (DSL – closed track)	PAN (AP)
2014	**SVM** + **BON** (Goutte, Léger, and Carpuat (2014))	LIBLINEAR[5] + BON (López-Monroy *et al.* 2014)
2015	**SVM** + **BON** (Malmasi and Dras 2015)	LIBLINEAR[5] + BON (Alvarez-Carmona *et al.* 2015)
2016	**SVM** + **BON** (Çöltekin and Rama 2016)	SVM + BON (Vollenbroek *et al.* 2016)
2017	**SVM** + **BON** (Bestgen 2017)	**SVM** + **BON** (Basile *et al.* 2017)

While deep learning approaches are gradually taking over different areas of NLP, the best approaches to AP still use more traditional classifiers and require extensive feature engineering (Rangel, Rosso, Potthast *et al.* 2017). This fact can be clearly seen if we look at the architectures used by the teams winning the AP shared tasks in recent years. Table 1 presents the winning approaches to the VarDial Discriminating between Similar Languages(DSL) shared tasks and PAN AP (gender, age, personality, and language variety prediction) tasks between 2014 and 2017[c]. In fact, six out of eight winning teams used one or an ensemble of Support Vector machine (SVM) classifiers and bag-of-*n*-grams (BON) features[d] for classification (two other winning teams used a LIBLINEAR classifier[e] and BON features), and when it comes to the task of DSL (all VarDial DSL tasks and PAN 2017 AP task), SVM classifiers with BON features have been used by all of the winning teams. The best ranking system that employed a deep learning architecture was developed by Miura *et al.* (2017) and ranked fourth in the PAN 2017 AP shared task.

The main contribution of this paper is to demonstrate that it is possible to build a neural architecture capable of achieving state-of-the-art results in the field of AP, and more specifically on the task of DSL. The proposed neural system is unique in a sense that it combines sophisticated feature engineering techniques used in traditional approaches to text classification with the newer neural automatic feature construction in order to achieve synergy between these two feature types. Experiments were conducted on eight distinct language varieties. First, we report results on the DSL Corpus Collection (DSLCC) v4.0 (Tan *et al.* 2014) used in VarDial 2017 (Zampieri *et al.* 2017), which was chosen because of its size (with 294,000 documents it is by far the largest corpus used in the presented shared tasks) and because it contains six different language groups, which also allows to explore the possibility of building a generic architecture that would discriminate well between languages in many different language groups without any language-group-specific parameter tweaking. Second, we report results on two much smaller corpora, the Arabic Dialect Identification Corpus (ADIC) used in a VarDial 2016 ADI shared task (Malmasi *et al.* 2016*b*) and the German Dialect Identification Corpus (GDIC) used in a VarDial 2018 GDI shared task (Zampieri *et al.* 2018) in order to determine how data set size and characteristics affect the competitiveness of the proposed system. Finally, we want to encourage the reproducibility of results and offer a larger research community (including linguists and social scientists) an easy out-of-the-box way of using our system. Therefore, we have not only published our code online (`http://source.ijs.si/mmartinc/NLE_2017`) but also implemented the architecture in the clowd-based visual programming system ClowdFlows (Kranjc, Podpečan, and Lavrač (2012)).

[c]VarDial evaluation campaign 2018 was not included because there was no DSL shared task. PAN 2018 gender classification task is not included because the gender classification task dealt with determining the gender of the author from both text and image data.

[d]The term BON features is used in a broader sense here, covering features such as bag-of-words, character, and word BON and bag-of-part-of-speech tags.

[e]It is unclear from the system description papers by López-Monroy *et al.* (2014) and Alvarez-Carmona *et al.* (2015) whether linear SVM or logistic regression classifier was used.

The paper is structured as follows. Section 2 addresses the related work on text classification in the field of AP. Section 3 describes the architecture of the proposed neural classification system in detail, while in Section 4 we report on our experimental setup. Results of the experiments and an error analysis are presented in Section 5, followed by an ablation study in Section 6. Section 7 presents the implementation of our approach in the ClowdFlows platform and finally, the conclusions and directions for further work are presented in Section 8.

2. Related work

The most popular approach to language variety classification usually relies on BON features and SVM classifiers (see Table 1). Bestgen (2017), the winner of the VarDial 2017 DSL task, used an SVM classifier with character *n*-grams, capitalized word character *n*-grams, *n*-grams of part-of-speech (POS) tags, and global statistics (proportions of capitalized letters, punctuation marks, spaces, etc.) features. *N*-grams had sizes from one to seven and different feature configurations were used for different language groups. The novelty of this approach was the use of the BM25 weighting scheme (Robertson and Zaragoza 2009) instead of the traditional term frequency-inverse document frequency (TF-IDF). BM25 (also called Okapi BM25) is a version of TF-IDF with some modifications made to each of the two components (term frequency and inverse document frequency) that, most importantly, allow it to take into account the length of the document. The classical TF-IDF formula is

$$TF - IDF = tf * log(\frac{N}{df})$$

where tf is the number of terms in the document, N is the number of documents in the corpus, and df the number of documents that contain the term. On the other hand, the formula for BM25 is the following:

$$BM25 = \frac{tf}{tf + k_1 * (1 - b + b * \frac{dl}{dl - avg_{dl}})} * log(\frac{N - df + 0.5}{df + 0.5})$$

where k_1 is a free parameter for tuning the asymptotic maximum of the term frequency component of the equation, dl is a document length, avg_{dl} an average length of a document in the corpus, and b a free parameter for fine-tuning the document length normalization part of the equation. While Bestgen (2017) showed in his experiments that the choice of the weighting scheme does impact the performance of the classifier, the general employment of different weighting schemes by the best performing systems in past shared tasks (Zampieri *et al.* 2017) suggests that feature weighting in general is positively correlated with gains in classification performance.

A very similar SVM-based system but with simpler features (just word unigrams, bigrams and, character three- to five-grams) was used by the winners of the PAN 2017 competition Basile *et al.* (2017). The authors of the paper also discovered that adding more complex features into the model actually negatively affected its performance. An SVM ensemble with almost identical features (word unigrams and character one- to six-grams) was also used by the winners of the VarDial 2016 ADI task Malmasi *et al.* (2016*a*). An even more minimalistic SVM-based approach was proposed by the winners of the VarDial 2016 DSL competition (Çöltekin and Rama 2016), who used only character three- to seven-grams as features. The authors also report on the failed attempt to build two neural networks capable of beating the results achieved by the SVM, first one being the FastText model proposed by Joulin *et al.* (2016) and the second one a hierarchical model based on character and word embeddings. Another attempt of tackling the task with a neural approach was reported by Criscuolo and Aluisio (2017). They ranked ninth with a hybrid configuration composed of a word-level multi-layer-perceptron model and a character-level Naive Bayes model. They also experimented with a word-level convolutional neural network (CNN), which performed slightly worse than their hybrid classifier.

There have also been some quite successful attempts of tackling the language variety prediction with neural networks. Miura *et al.* (2017) ranked fourth in the PAN 2017 shared task by using a system consisting of a recurrent neural network layer, a CNN layer, and an attention mechanism. In a set of VarDial 2018 evaluation campaign tasks, Ali tackled the tasks of distinguishing between four different Swiss German dialects (Ali 2018*a*), five Arabic dialects (Ali 2018*b*), and five closely related languages from the Indo-Aryan language family (Ali 2018*c*), ranking second in the first and second task and fourth in the third task, respectively. The system is based on character-level CNNs and recurrent networks. The one-hot encoded input sequence of characters enters the network through the recurrent GRU layer used as an embedding layer. Next is the convolutional layer with different filter sizes, ranging from two to seven, which is followed by a batch normalization, max-pooling, dropout, and finally a softmax layer used for calculating the probability distribution over the labels.

While neural networks were not a frequent choice in VarDial DSL 2017 (Zampieri *et al.* 2017), in the VarDial DSL 2016 shared task (Malmasi *et al.* 2016*b*) three teams used some form of CNN. Belinkov and Glass (2016) used a character-level CNN and ranked sixth out of seven teams, achieving more than six percentage points lower accuracy than the winning system. A somewhat more sophisticated system was employed by Bjerva (2016), who combined CNN with recurrent units, developing a so-called residual network that takes as input sentences represented at a byte level. He ranked fifth in the competition. A third team called *Uppsala* used a word-level CNN but did not submit a report about their approach.

Two rear occasions when an SVM-based system did not win in a language variety classification shared task occurred at VarDial 2018 GDI and Indo-Aryan Language Identification (ILI) tasks, where Jauhiainen *et al.* beat the nearest competition by a large margin of four percentage points (Jauhiainen, Jauhiainen, and Lindén (2018a)) and more than five percentage points (Jauhiainen, Jauhiainen, and Lindén (2018b)), respectively. Their Helsinki language identification (HeLI) method with adaptive language modeling was in both cases calculated on character four-grams. The HeLI system was, however, outperformed by a margin of almost five percentage points at the VarDial 2018 Discriminating between Dutch and Flemish in Subtitles task by an SVM-based system proposed by Çöltekin, Rama, and Blaschke (2018).

3. System architecture

Research presented in Section 2 indicates that using character-level CNNs might be the most promising neural approach to the task of DSL. CNNs are able to identify important parts of a text sequence by employing a max-over-time pooling operation (Collobert *et al.* 2011), which keeps only the character sequences with the highest predictive power in the text. These sequences of predefined lengths resemble character *n*-grams, which were used in nearly every winning approach in the past shared task, but the CNN approach also has the advantage over the traditional BON approaches that it preserves the order in which these text areas with high predictive power appear in the text.

On the other hand, its main disadvantage could be the lack of an effective weighting scheme that would be capable of determining how specific these character sequences are for every input document. The data are fed into a neural classifier in small batches; therefore, it is impossible for it to obtain a somewhat global view on the data and its structure, which is encoded in the more traditional TF-IDF (or BM25) weighted input matrix. Another intuition that might explain the usefulness of weighting schemes for the specific task of language variety classification is related to named entities, for which it was shown in the past shared tasks that they in many cases reflect the origin of the text (Zampieri *et al.* 2015). The hypothesis is that these entities are quite rare and somewhat document specific and are therefore given large weights by different weighting schemes, encouraging the classifier to pay attention to them. The importance of choosing an effective weighting scheme on the task of DSL is also emphasized in the research by Bestgen

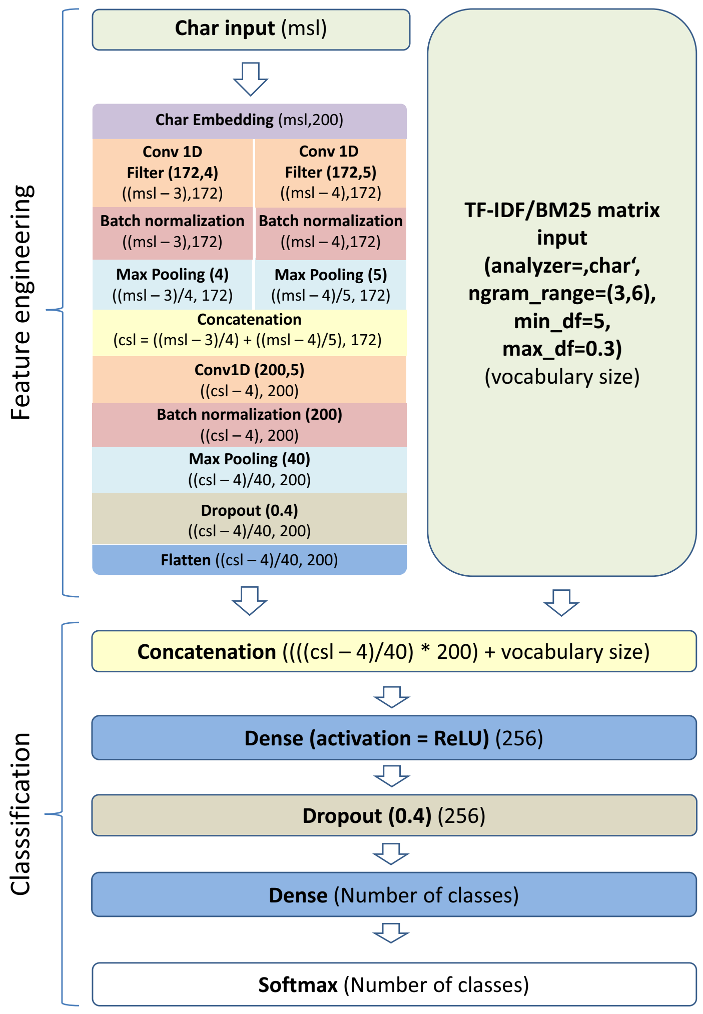

Feature engineering
Classification
Char input (msl)
Char Embedding (msl,200)
Conv 1D Filter (172,4) ((msl – 3),172)
Conv 1D Filter (172,5) ((msl – 4),172)
Batch normalization ((msl – 3),172)
Batch normalization ((msl – 4),172)
Max Pooling (4) ((msl – 3)/4, 172)
Max Pooling (5) ((msl – 4)/5, 172)
Concatenation (csl = ((msl – 3)/4) + ((msl – 4)/5), 172)
Conv1D (200,5) ((csl – 4), 200)
Batch normalization (200) ((csl – 4), 200)
Max Pooling (40) ((csl – 4)/40, 200)
Dropout (0.4) ((csl – 4)/40, 200)
Flatten ((csl – 4)/40, 200)
TF-IDF/BM25 matrix input (analyzer=‚char‘, ngram_range=(3,6), min_df=5, max_df=0.3) (vocabulary size)
Concatenation ((((csl – 4)/40) * 200) + vocabulary size)
Dense (activation = ReLU) (256)
Dropout (0.4) (256)
Dense (Number of classes)
Softmax (Number of classes)

Figure 1. System architecture: layer names and input parameters are written in bold, layer output sizes are written in normal text, *msl* stands for maximum sequence length, and *csl* stands for concatenated sequence length.

(2017), the winner of the VarDial 2017 DSL task, who managed to gain some performance boost by replacing the TF-IDF weighting scheme with BM25.

Our architecture (visualized in Figure 1) builds on these findings from the literature and is in its essence an effective hybrid between a traditional feature engineering approach, which relies on different kinds of BON features, and a newer neural feature engineering approach to text classification. This combination of two distinct text classification architectures is capable of leveraging character-level and more global document/corpus-level information and achieving synergy between these two data flows. The main idea is to improve on standard CNN approaches by adding an additional input to the network that would overcome the lack of an effective weighting scheme. Therefore, the text is fed to the network in the form of two distinct inputs (as presented in Figure 1):

- *Char input*: Every document is converted into a numeric character sequence (every character is represented by a distinct integer) of length corresponding to the number of characters in the longest document in the train set (zero value padding is added after the document character sequence and truncating is also performed at the end of the sequence if the document in the validation or test set is too long).
- *TF-IDF/BM25 matrix*: We explore the effect of two distinct weighting schemes on the performance of the classifier; therefore, input data set is converted into a matrix of either TF-IDF

or BM25 weighted features with a *TfidfVectorizer* from ScikitLearn (Pedregosa *et al.* 2011) or our own implementation of the *BM25Vectorizer*. The matrix is calculated on character n-grams of sizes three, four, five, and six with a minimum document frequency of five and appearing in at most 30% of the documents in the train set. Sublinear term frequency scaling is applied in the term frequency calculation when *TfidfVectorizer* is used and for BM25 weighting parameters b and k_1 are set to 0.75 and 1.2, respectively, same as in Bestgen (2017).

The architecture for processing *Char input* is a relatively shallow character-level CNN with randomly initialized embeddings of size $msl \times 200$, where msl stands for *maximum sequence length*. Assuming that w is a convolutional filter, b is a bias, and f a nonlinear function (a rectified linear unit (*ReLU*) in our case), a distinct character n-gram feature $\mathrm{c_i}$ is produced for every possible window of h characters $x_{i:i+h-1}$ in the document according to the convolutional equation:

$$c_i = f(w \cdot x_{i:i+h-1} + b)$$

In the first step, we employ two parallel convolutional layers (one having a window of size four and the other of size five), each of them having 172 convolutional filters. These layers return two feature maps of size $(msl - ws + 1) \times 172$, where ws is the window size. Batch normalization and max-over-time pooling operations (Collobert *et al.* 2011) are applied on both feature maps in order to filter out features with low predictive power. These operations produce two matrices of size $(msl - ws + 1)/mws \times 172$, where sizes of max-pooling windows (mws) correspond to convolution window sizes. Output matrices are concatenated and the resulting matrix is fed into a second convolutional layer with 200 convolutional filters and window size five. Batch normalization and max-over-time pooling are applied again and after that we conduct a dropout operation on the output of the layer, in which 40% of input units are dropped in order to reduce overfitting. Finally, the resulting output is flattened (changed from a two-dimensional to a one-dimensional vector) and passed to a *Concatenation* layer, where it is concatenated with the input *TF-IDF/BM25 matrix*. The resulting concatenation is passed on to a fully connected layer (*Dense*) with a *ReLU* activation layer and dropout is conducted again, this time on the concatenated vectors. A final step is passing the resulting vectors to a dense layer with a *Softmax* activation, responsible for producing the final probability distribution over language variety classes.

4. Experimental setup

This section describes the data sets and the methodology used in our experiments.

4.1 Data

All experiments were conducted on three corpora described in Table 2:

- **DSLCC v4.0 (Tan *et al.* 2014)**[f]: the corpus used in the VarDial 2017 DSL shared task. The corpus contains 294,000 short excerpts of news texts divided into 6 distinct language groups (Slavic, Indonesian and Malay, Portuguese, Spanish, French, and Farsi) and covering 14 language varieties in total: Bosnian, Croatian and Serbian; Malay and Indonesian; Persian and Dari; Canadian and Hexagonal French; Brazilian and European Portuguese; Argentine, Peninsular, and Peruvian Spanish. Each language contains 20,000 documents for training (out of which 2000 are to be used as a validation set) and 1000 for testing.
- **ADIC (Ali *et al.* 2015)**[g]: the corpus used in the VarDial 2016 ADI shared task. It contains transcribed speech in Modern Standard Arabic, Egyptian, Gulf, Levantine, and North African

[f]The corpus is publicly available at `http://ttg.uni-saarland.de/resources/DSLCC/`
[g]The corpus is publicly available at `http://alt.qcri.org/resources/ArabicDialectIDCorpus/varDial_DSL_shared_task_2016_subtask2/`

Table 2. DSLCC v4.0, ADIC and GDIC corpora

DSLCC v4.0					
Language/Variety	Class	Train inst.	Train tokens	Test inst.	Test tokens
Bosnian	bs	20,000	716,537	1000	35,756
Croatian	hr	20,000	845,639	1000	42,774
Serbian	sr	20,000	777,363	1000	39,003
Indonesian	id	20,000	800,639	1000	39,954
Malay	my	20,000	591,246	1000	29,028
Brazilian Portuguese	pt-BR	20,000	907,657	1000	45,715
European Portuguese	pt-PT	20,000	832,664	1000	41,689
Argentine Spanish	es-AR	20,000	939,425	1000	42,392
Castilian Spanish	es-ES	20,000	1,000,235	1000	50,134
Peruvian Spanish	es-PE	20,000	569,587	1000	28,097
Canadian French	fr-CA	20,000	712,467	1000	36,121
Hexagonal French	fr-FR	20,000	871,026	1000	44,076
Persian	fa-IR	20,000	824,640	1000	41,900
Dari	fa-AF	20,000	601,025	1000	30,121
Total		280.000	8,639,459	14,000	546,790
ADIC					
Egyptian	EGY	1578	85,000	315	13,000
Gulf	GLF	1672	65,000	256	14,000
Levantine	LAV	1758	66,000	344	14,000
Modern Standard	MSA	999	49,000	274	14,000
North African	NOR	1612	52,000	351	12,000
Total		7619	317,000	1540	67,000
GDIC					
Bern	BE	4956	35,962	1191	12,013
Basel	BS	4921	36,965	1200	9802
Lucerne	LU	4593	38,328	1186	11,372
Zurich	ZH	4834	36,919	1175	9610
Total		19,304	148,174	4,752	42,797

dialects. Speech excerpts were taken from a multi-dialectical corpus containing broadcast, debate and discussion programs from Al Jazeera. Altogether 7619 documents were used for training (out of which 10% were used for validation) and 1540 documents for testing.

- **GDIC (Samardzic, Scherrer, and Glaser (2016))**: the corpus used in the VarDial 2018 GDI shared task. Texts were extracted from the ArchiMob corpus of Spoken Swiss German[h], which contains 34 oral interviews with people speaking Bern, Basel, Lucerne, and Zurich Swiss German dialects. A total of 19,304 documents were used for training (out of which 10% were used for validation) and 4752 for testing.

[h]The ArchiMob corpus is publicly available at `https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html`

4.2 Methodology

For experiments in the DSLCC v4.0 we chose to use a two-step approach, as first proposed by Goutte, Léger, and Carpuat (2014):

(1) The general classifier is trained to identify the language group for every specific document. For this step, the input TF-IDF/BM25 matrix is calculated only on the word bound character *n*-grams[i] of sizes three, four, and five with a minimum document frequency of five and appearing in at most 30% of the documents in the train set. This configuration produces a TF-IDF/BM25 matrix of smaller size than if the configuration for the TF-IDF/BM25 matrix, described in Section 3, was used. This size reduction was chosen because distinguishing between different language groups is not a difficult problem, therefore, this parameter reduction does not influence performance but it reduces the execution time.

(2) We train six different classification models, one for each language group. After being classified as belonging to a specific language group by the general classifier in Step 1, the documents are assigned to the appropriate classifier for predicting the final language variety.

Since NLP tools and resources such as POS taggers, pretrained word embeddings, word dictionaries, and tokenizers might not exist for some underresourced languages, we also believe that an architecture which does not require language-specific resources and tools, apart from the training corpus, might be more useful and easier to use in real-life applications. For this reason, our system does not require any additional resources and the conducted preprocessing procedure is light[j].

We show (see Section 5) that the proposed architecture is generic enough to outperform the winning approach of VarDial 2017 on all of the language groups without any language-group-specific parameter or architecture tweaking. In contrast, most of the approaches of the VarDial 2017 DSL shared task resorted to language-group-specific optimization, as getting even the slightest possible performance boost by employing this tactic was important due to the competitive nature of shared tasks.

For the experiments on the smaller ADIC and GDIC data sets, we use the same hyperparameter configuration and TD-IDF/BM25 features as for the six classification models for specific language groups in the DSLCC v4.0 corpus because we want to explore the relation between model performance and data set size. The hypothesis is that the performance of traditional SVM approaches would be less affected by smaller data set size than neural approaches.

We conducted an extensive grid search on the DSLCC v4.0 in order to find the best hyperparameters for the model. All combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and presented in Section 3:

- Learning rates: 0.001, **0.0008**, 0.0006, 0.0004, 0.0002
- Number of parallel convolutions with different filter sizes: [3] [4], [3,4], **[4,5]**, [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], [3,4,5,6], [4,5,6,7], [3,4,5,6,7]
- Character embedding sizes: 100, **200**, 400
- Dense layer sizes: 128, **256**, 512
- Dropout values: 0.2, 0.3, **0.4**, 0.5
- Number of convolutional filters in the first convolution step: 156, **172**, 200
- Number of convolutional filters in the second convolution step: 156, 172, **200**

[i]Word-bound character *n*-grams are made only from text inside word boundaries, for example, a sequence *this is great* would produce a word-bound character 4-gram sequence *this, is__, grea, reat*, in which _ stands for empty space character.

[j]We only replace all email addresses in the text with *EMAIL* tokens and all URLs with *HTTPURL* tokens by employing regular expressions. Even if this might not be relevant to all of the corpora, we keep the preprocessing unchanged for all the settings.

- Size of a max-pooling window in the second convolution step: 10, 20, **40**, 60
- BON n sizes: [3] [4] [3,4], [4,5], [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], **[3,4,5,6]**, [4,5,6,7], [3,4,5,6,7]
- Minimum document frequency of an n-gram in the TF-IDF/BM25 matrix: [2], **[5]**, [10]
- BM25 b parameter: 0.5, **0.75**, 1.0
- BM25 k_1 parameter: 1.0, **1.2**, 1.4

The hyperparameters, which influenced the performance of the network the most, were the learning rate, CNN filter sizes, size of the max-pooling window, BON n size, and a minimum document frequency of n-grams. Too many parallel convolutions, small sizes of the max-pooling window, and low minimum document frequency of n-grams showed tendency toward overfitting, especially when used together in combination. In general, we noticed quite a strong tendency toward overfitting no matter the hyperparameter combination, which could be to some extent the consequence of feeding a high-dimensional TF-IDF/BM25 matrix to the network, which greatly increases the number of network parameters. We noticed that a combination of a relatively small learning rate and a large dropout worked best to counter this tendency.

Another thing we noticed is that using exactly the same configurations of convolutional filter sizes and n-gram sizes negatively affected the performance, which was slightly improved when the configurations did not completely overlap. The hypothesis is that synergy between two data flows is less effective if the information in these two data flows is too similar. The validation set results did however show that configurations containing 4- and 5-grams and filter sizes of 4 and 5 in general worked better than other configurations for DSLCC v4.0 classification; therefore, these configurations were used in both data flows despite the overlap.

We use the Python Keras library (Chollet 2015) for the implementation of the system. For optimization, we use an Adam optimizer (Kingma and Ba 2015) with a learning rate of 0.0008. For each language variety in the DSLCC v4.0, the model is trained on the train set for 20 epochs and tested on the validation set after every epoch. The models trained on the ADIC and GDIC data sets are trained for 80 epochs due to longer convergence time on less data. The model with the best performance on the validation set is chosen for the test set predictions.

5. Results

First we present results on the DSLCC v4.0, which is (as it is the largest and covers the largest number of language varieties) the main focus of this study, then we present results on ADIC and GDIC and finally, we present findings of the error analysis conducted on the misclassified Slavic documents of the DSLCC v4.0 corpus.

5.1 Results on the DSLCC v4.0

Table 3 presents the results achieved by our neural classifier in comparison to the winner of the VarDial 2017 DSL shared task (Bestgen 2017) in terms of weighted F1, micro F1, macro F1, and accuracy measures.

The first step of the two-step classification approach, distinguishing between different language groups (*All-language groups (TF-IDF)* and *All-language groups (BM25)* rows in Table 3), proved trivial for the system, which achieved almost perfect weighted F1 score and misclassified only 27 documents out of 14,000 in the test set when TF-IDF weighting scheme was used and 29 documents when BM25 weighting scheme was used. If we look at the confusion matrices for language group classification (Figures 2 and 3), both models had most difficulties distinguishing between Spanish and Portuguese language groups. Ten Spanish texts were misclassified as Portuguese but on the other hand, only one Portuguese document was misclassified as Spanish when TF-IDF weighting scheme was used. With BM25 weights, the classifier misclassified nine

Table 3. Results of the proposed language variety classifier on the DSLCC v4.0 for different language groups, as well as for the discrimination between language groups (All-language groups). Also the results for all language varieties (All-language varieties) are provided, for which a comparison with the official VarDial 2017 winners is made. Results for both weighting schemes, TF-IDF and BM25, are reported separately

Language group (weighting)	F1 (weighted)	F1 (micro)	F1 (macro)	Accuracy
All-language groups (TF-IDF)	**0.9981**	**0.9981**	**0.9980**	**0.9981**
All-language groups (BM25)	0.9979	0.9979	0.9980	0.9980
Spanish (TF-IDF)	**0.9136**	**0.9140**	**0.9136**	**0.9140**
Spanish (BM25)	0.9042	0.9047	0.9042	0.9047
Slavic (TF-IDF)	0.8645	0.8650	0.8645	0.8650
Slavic (BM25)	**0.8752**	**0.8753**	**0.8752**	**0.8753**
Farsi (TF-IDF)	0.9685	0.9685	0.9685	0.9685
Farsi (BM25)	**0.9690**	**0.9690**	**0.9690**	**0.9690**
French (TF-IDF)	**0.9570**	**0.9570**	**0.9570**	**0.9570**
French (BM25)	0.9545	0.9545	0.9545	0.9545
Malay and Indonesian (TF-IDF)	0.9855	0.9855	0.9855	0.9855
Malay and Indonesian (BM25)	**0.9860**	**0.9860**	**0.9860**	**0.9860**
Portuguese (TF-IDF)	**0.9480**	**0.9480**	**0.9480**	**0.9480**
Portuguese (BM25)	0.9460	0.9460	0.9460	0.9460
All-language varieties (TF-IDF)	**0.9310**	**0.9312**	**0.9310**	**0.9312**
All-language varieties (BM25)	0.9304	0.9305	0.9304	0.9305
VarDial 2017 winner Bestgen (2017)	0.9271	0.9274	0.9271	0.9274

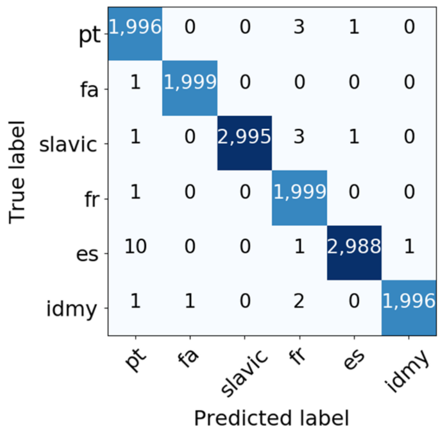


Figure 2. Confusion matrix for language group classification (TF-IDF weighting scheme).

Spanish documents as Portuguese and four Portuguese documents as Spanish. The analysis also reveals some surprising mistakes, such as that three Slavic documents and two documents from the Indonesian and Malay language group were misclassified as French with TF-IDF weighting and four documents from the Indonesian and Malay language group, three Spanish, and three French documents were classified as Slavic with BM25 weighting. A closer inspection of misclassified documents also reveals that these documents are in general much shorter (average word length is 9.74 and 10.17 when TF-IDF and BM25 are used respectively) than an average document in the Slavic sub-corpus (39.06 words long) and very likely contain some misleading named entities (e.g., a Slavic document, which was misclassifed as Spanish when TF-IDF weighting was used, contains the following text: *Caffe - Pizzeria ""BELLA DONNA"" u DOC-u*).

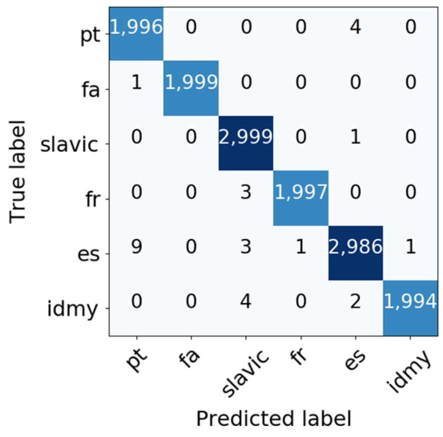


Figure 3. Confusion matrix for language group classification (BM25 weighting scheme).

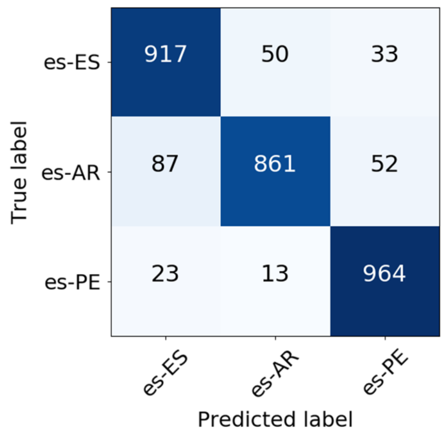


Figure 4. Confusion matrix for Spanish language varieties classification (TF-IDF weighting scheme).

Results for the second step of the two-step classification approach indicate that the difficulty of distinguishing language varieties within different language groups varies. The system had most difficulties with distinguishing between different Slavic languages, where it achieved by far the worst results with an weighted F1 of 0.8645 when TF-IDF weighting scheme was employed and about one percentage point better results when BM25 weighting was used. The second most difficult were Spanish varieties. We should point out that this comes as no surprise, since Slavic and Spanish languages groups were the only two groups that contained three varieties, while the other groups in DSLCC v4.0 contained two varieties. The system had least problems with distinguishing between Malay and Indonesian languages.

When it comes to comparing two weighting schemes, there is no clear overall winner. The biggest differences in performance are on Spanish varieties, where TF-IDF weighting outperforms BM25 by about one percentage point according to every measure, and on Slavic varieties, where BM25 weighting outperforms TF-IDF by a very similar margin. The differences on other varieties are smaller, ranging from 0.005 on Farsi and Malay and Indonesian varieties to 0.020 on Portuguese varieties.

Confusion matrices for specific language varieties enable a more thorough analysis of the results. For Spanish varieties (Figures 4 and 5), the system had most problems distinguishing between Argentine and Castilian Spanish. The second most common mistake no matter the weighting scheme was classifying Argentine Spanish as the Peruvian variety of Spanish. On the other hand, Peruvian Spanish was the easiest to classify by the system, with altogether only 36 (TF-IDF weighting) and 37 (BM25 weighting) misclassified instances.

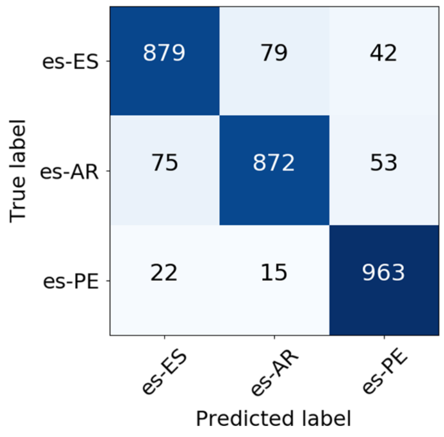


Figure 5. Confusion matrix for Spanish language varieties classification (BM25 weighting scheme).

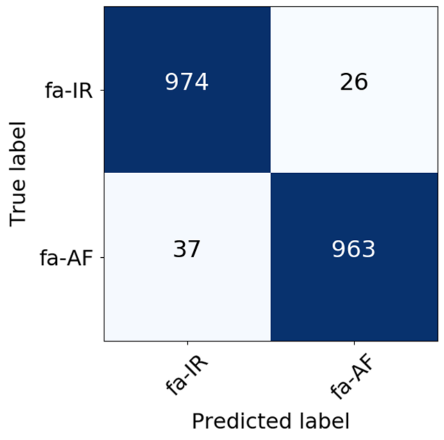


Figure 6. Confusion matrix for Farsi language varieties classification (TF-IDF weighting scheme).

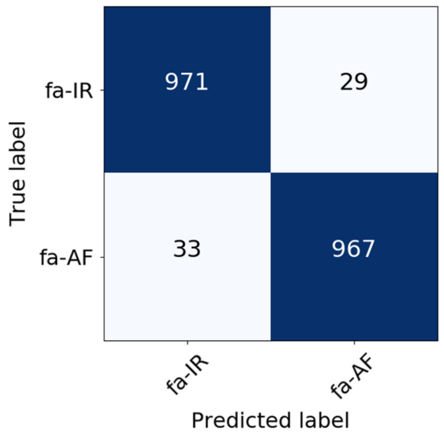


Figure 7. Confusion matrix for Farsi language varieties classification (BM25 weighting scheme).

The system performed well for all binary predictions (Figures 6 and 7, Figures 8 and 9, Figures 10 and 11, Figures 12 and 13) and the difference in performance between two weighting schemes are small. Out of these confusion matrices, the most unbalanced with regard to false predictions is the confusion matrix for Indonesian and Malay variety (Figure 10), where twice as many Indonesian documents were classified as Malay than the other way around when TF-IDF weighting was used. Although, as mentioned before, distinguishing between Indonesian and Malay was the least difficult task for the classifier and altogether only 29 and 28 instances were misclassified when TF-IDF and BM25 weighting were used, respectively.

For Slavic languages (Figures 14 and 15), the hardest problem for the system was distinguishing between Croatian and Bosnian, with 113 Bosnian documents being classified as Croatian and 112 Croatian documents being classified as Bosnian when TF-IDF weighting was used and with 113

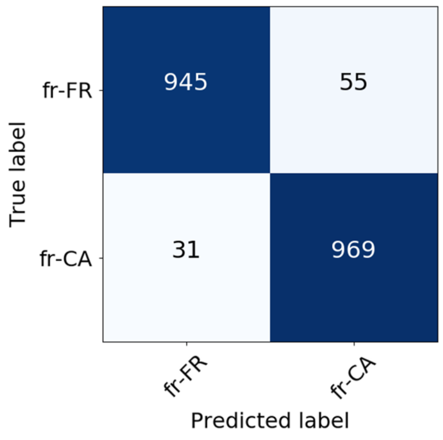


Figure 8. Confusion matrix for French language varieties classification (TF-IDF weighting scheme).

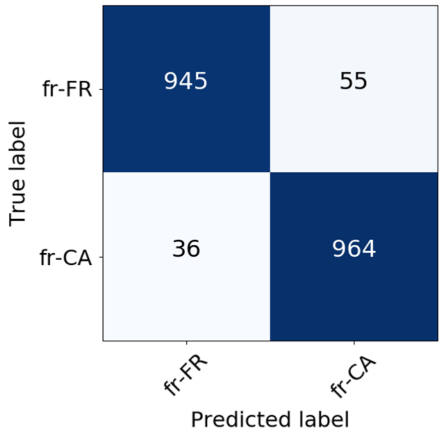


Figure 9. Confusion matrix for French language varieties classification (BM25 weighting scheme).

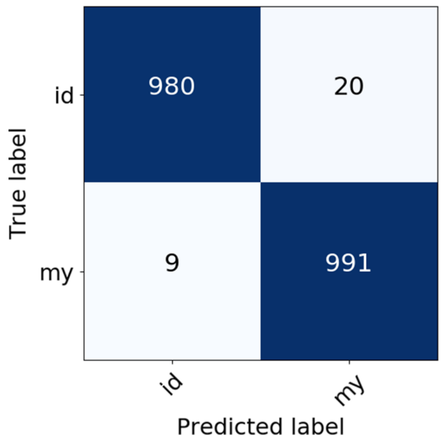


Figure 10. Confusion matrix for Indonesian and Malay variety classification (TF-IDF weighting scheme).

Bosnian documents being classified as Croatian and 99 Croatian documents being classified as Bosnian when BM25 weighting was employed. Distinguishing between Bosnian and Serbian was also not trivial for the classifier no matter the weighting scheme, with 94 Bosnian documents being misclassified as Serbian and 66 Serbian documents misclassified as Bosnian when TF-IDF weighting scheme was deployed and 73 Bosnian documents being misclassified as Serbian and vice versa when BM25 weighting was used. On the other hand, distinguishing between Serbian and Croatian is a much easier problem, with altogether only 20 (TF-IDF weighting) and 16 (BM25 weighting) documents being misclassified.

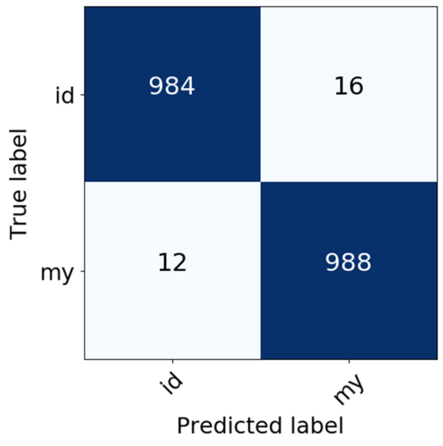


Figure 11. Confusion matrix for Indonesian and Malay variety classification (BM25 weighting scheme).

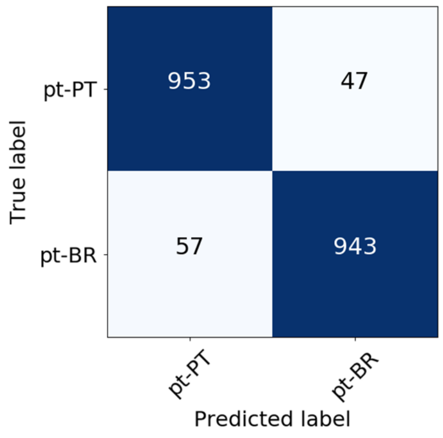


Figure 12. Confusion matrix for Portuguese language varieties classification (TF-IDF weighting scheme).

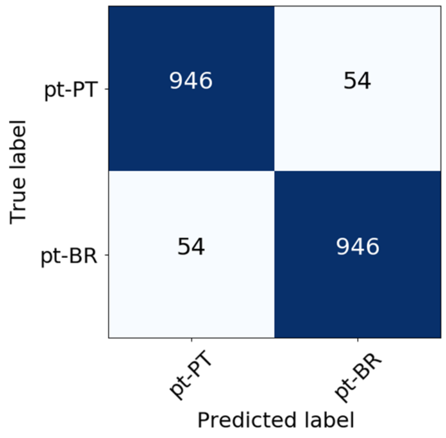


Figure 13. Confusion matrix for Portuguese language varieties classification (BM25 weighting scheme).

Overall (rows *All-language varieties (TF-IDF)* and *All-language varieties (BM25)* in Table 3), the neural network outperforms the SVM-based approach used by the winners of the shared task by about 0.4 percentage points according to all measures when TF-IDF weighting scheme is used. BM25 weighting performs slightly worse but still outperforms state of the art by about 0.35 percentage points margin. Our results therefore differ from the study conducted by Bestgen (2017), the winner of the shared task, where he reported improvement in performance for all but one language group when TF-IDF weighting is replaced by BM25. It should, however, be noted that these improvements were only reported on the validation set and no comparison between weighting schemes was done on the official test set.

There were no available reported results for individual language groups on the official test set, therefore we provide a comparison with the VarDial 2017 DSL winning team on the validation set,

Table 4. Accuracy comparison of our system to the VarDial 2017 DSL winners on validation sets

Language group (weighting)	Our system	VarDial 2017 winner	Improvement (%)
Spanish (TF-IDF)	**0.9180**	0.8970	2.10
Spanish (BM25)	**0.9202**	0.9030	1.72
Slavic (TF-IDF)	**0.8663**	0.8445	2.18
Slavic (BM25)	**0.8670**	0.8506	1.64
Farsi (TF-IDF)	**0.9685**	0.9598	0.87
Farsi (BM25)	**0.9720**	0.9632	0.88
French (TF-IDF)	**0.9588**	0.9396	1.92
French (BM25)	**0.9590**	0.9472	1.18
Malay and Indonesian (TF-IDF)	**0.9863**	0.9835	0.28
Malay and Indonesian (BM25)	**0.9875**	0.9827	0.48
Portuguese (TF-IDF)	**0.9440**	0.9299	1.41
Portuguese (BM25)	**0.9428**	0.9355	0.73

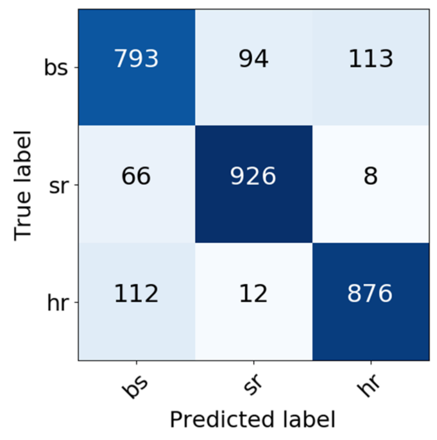


Figure 14. Confusion matrix for Slavic language varieties classification (TF-IDF weighting scheme).

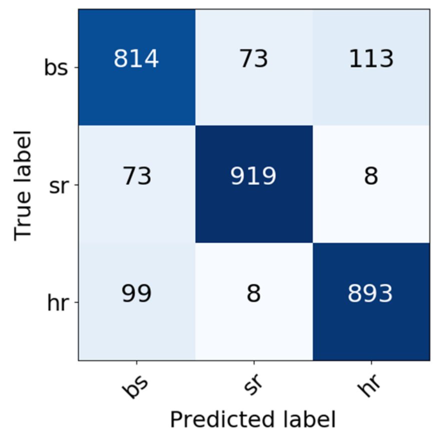


Figure 15. Confusion matrix for Slavic language varieties classification (BM25 weighting scheme).

as the author (Bestgen 2017) reports them when presenting the benefits of the weighting scheme BM25 (in their Table 3 on p. 119). Note, however, that the results report on a slightly simplified system, as for the weighting scheme comparison, the author used only character *n*-grams features. Comparison results are presented in Table 4.

Table 5. Results of the proposed language variety classifier on the ADIC and GDIC. Results for both weighting schemes, TF-IDF and BM25, are reported separately

Language group (weighting)	F1 (weighted)	F1 (micro)	F1 (macro)	Accuracy
ADIC (TF-IDF)	**0.5152**	0.5123	0.5147	**0.5123**
ADIC (BM25)	0.5090	0.5097	0.5067	0.5097
VarDial ADI 2016 winner Malmasi *et al.* (2016*a*)	0.5132	/	/	0.5117
GDIC (TF-IDF)	0.6281	0.6294	0.6280	0.6294
GDIC (BM25)	0.6289	0.6311	0.6289	0.6311
VarDial GDI 2018 winner Jauhiainen *et al.* (2018*a*)	/	/	**0.6860**	/

Our system performs better than the simplified version of the VarDial 2017 DSL shared task winning system on all language groups. When TF-ID weighting is used by both systems, the differences vary from around two percentage points on Spanish, Slavic, and French language groups, to about 1.5 percentage point difference on the Portuguese language group, and finally, to only 0.28 percentage point difference on Malay and Indonesian, which are the easiest languages to distinguish for both of the classifiers. When BM25 weighting scheme is used, the differences are smaller, ranging from about 1.5 percentage point on Spanish and Slavic to about 0.5 percentage point on Malay and Indonesian.

Interestingly, when it comes to comparing both weighting schemes only on validation sets, the influence on the performance of our system when BM25 weighting is used is quite consistent with the influence reported by Bestgen (2017). By using BM25 weighting, the performance is improved on five out of six language groups, same as in Bestgen (2017), although the language groups are not the same: in Bestgen (2017) performance is not improved on the Malay and Indonesian language group while we report no improvement on Portuguese. However, these improvements at least in our case do not translate well to performance improvements on the official test set.

5.2 Results on ADIC and GDIC

Table 5 presents the results achieved by our neural classifier on the ADIC and GDIC corpora in comparison to the winners of the VarDial ADI 2016 and VarDial GDI 2018 shared tasks. The system manages to improve on the state of the art on the ADIC by a small margin of about 0.2 percentage point according to the weighted F1 score when TF-IDF weighting is used, even though the ADIC contains more than 10 times less documents per class than the language varieties in the DSLCC v4.0. By using BM25 weighting, the performance of the classifier is about 0.6 and 0.2 percentage points worse in terms of accuracy and weighted F1 score. On the other hand, the results on the GDIC are almost six percentage points lower than the current state-of-the-art HeLI method (Jauhiainen *et al.* 2018*a*) in terms of macro F1 score. Our system also performed worse than the SVM-based system proposed by Çöltekin *et al.* (2018) and a recurrent neural network proposed by Ali (2018a), which achieved macro F1 scores of 0.646 and 0.645, respectively. We can also observe that BM25 weighting slightly improves the performance according to all the criteria. Results on ADIC and GDIC corpora are somewhat in line with the initial hypothesis that neural approaches are more affected by a small data set size than more traditional SVM approaches. Previous SVM-based state of the art on the ADIC corpora is outperformed by a smaller margin than the DSLCC v4.0 state of the art and the proposed system performs worse than the second ranked SVM system (2018) on the GDIC corpus.

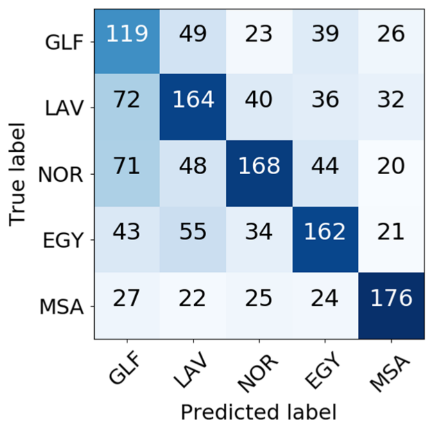


Figure 16. Confusion matrix for Arabic language varieties classification (TF-IDF weighting scheme).

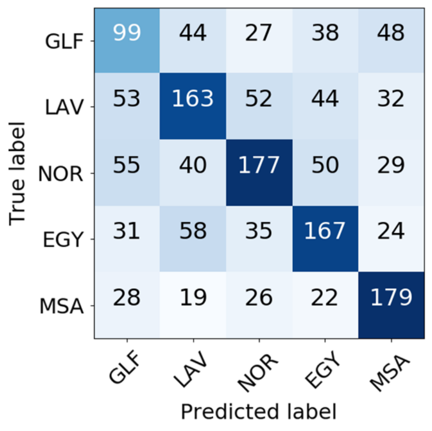


Figure 17. Confusion matrix for Arabic language varieties classification (BM25 weighting scheme).

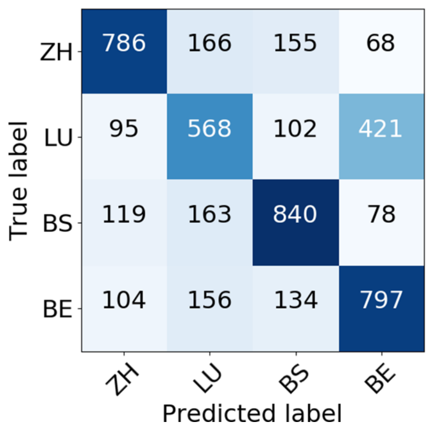


Figure 18. Confusion matrix for German language varieties classification (TF-IDF weighting scheme).

Confusion matrices for the ADIC (Figures 16 and 17) show that the Modern Standard Arabic is the easiest to classify no matter the weighting scheme. We can also see that if BM25 weighting is used, the classifier struggles much more with the Gulf dialect, correctly classifying only 99 out of 256 instances, than if TF-IDF weighting is used, in which case it correctly classifies 119 instances.

Confusion matrices for the GDIC (Figures 18 and 19) show that the choice of the weighting scheme does not have as big of an influence on the performance of the classifier as in the case of ADIC. No matter the weighting scheme, by far the most common mistake was misclassifying the Lucerne dialect as a Bern dialect. Interestingly, the opposite mistake of misclassifying Bern dialect

Table 6. Results of the error analysis on 405 misclassified Slavic documents

Group	Num. doc.	Prop. of doc.	Avg. doc. length
No named entities	144	0.36	26.94
Misleading named entities	70	0.17	40.96
Clarifying named entities	41	0.10	34.96
Unrelated named entities	150	0.37	33.17
All misclassified	405	1.00	32.48

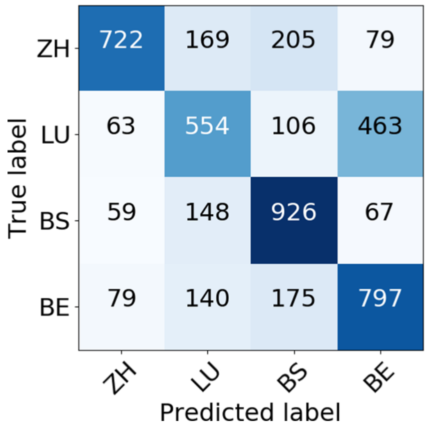


Figure 19. Confusion matrix for German language varieties classification (BM25 weighting scheme).

as Lucerne dialect is much rarer, which might be connected to some extent to the fact that the train set contains 328 more documents for the Bern dialect than for the Lucerne dialect.

5.3 Error analysis

We conducted a manual error analysis on the misclassified Slavic documents[k] in order to get a clearer picture about what kind of documents are the hardest to classify. Misclassified documents were manually grouped into four classes according to the number and type of named entities found in the document:

- **No named entities**: Documents without any named entities.
- **Misleading named entities**: Documents containing any named entities (e.g., names of regions, cities, public figures) originating from a country with the official language variety corresponding to one of the two possible incorrect language varieties (e.g., a document labeled as Serbian containing the word *Zagreb*, which is the capital of Croatia, would be put into this class).
- **Clarifying named entities**: Documents containing named entities originating from a country with the official language variety being the correct language variety and containing no misleading entities.
- **Unrelated named entities**: Documents containing only named entities that are not originating from any of the countries speaking target language varieties (e.g., a document containing only the named entity *Budapest* would be classified into this category).

Results of the analysis are presented in Table 6. Results show that a large portion of misclassified documents (73%) either contain no named entities (36%) or contain only unrelated named entities (37%), which might make them harder to classify, although we cannot claim that for sure, since we

[k]Error analysis was conducted on documents misclassified by the system that employed TF-IDF weighting scheme.

Table 7. Results of the ablation study. Column *CNN F1 (weighted)* presents performance of the system in terms of weighted F1 if only CNN-based features are used, column *BON F1 (weighted)* presents performance of the system if only TF-IDF-weighted BON features are used and column *All F1 (weighted)* presents the performance when these two types of features are combined

Language group	All F1 (weighted)	CNN F1 (weighted)	BON F1 (weighted)
DSLCC v4.0			
All-language groups	**0.9981**	0.9971	0.9976
Spanish	**0.9136**	0.8599	0.8863
Slavic	**0.8645**	0.8300	0.8594
Farsi	**0.9685**	0.9465	0.9610
French	**0.9570**	0.9325	0.9420
Malay and Indonesian	0.9855	0.9560	**0.9875**
Portuguese	**0.9480**	0.8994	0.9434
All-language varieties	**0.9310**	0.8935	0.9199
ADIC	0.5152	0.3971	**0.5177**
GDIC	**0.6281**	0.6059	0.6190

do not know the distribution of these classes across the entire test set. About 17% of the documents on the other hand contain misleading named entities that could influence the classifier prediction. There are also 41 documents (10%) containing only clarifying named entities that would be easily classified correctly by any human annotator with some basic background knowledge about Serbia, Bosnia, and Croatia. This suggests that there is still some room for improvement for the developed classifier.

Another finding is that misclassified documents are in average shorter (32.48 words long) than an average document from a Slavic language group (39.18 words long), suggesting that shorter documents are harder to classify by the classifier due to less available information. We can also see that the only group containing documents with similar length as the whole test set are documents containing misleading named entities (40.96 words long), which suggests that the classifier does somewhat rely on named entities during the prediction process.

6. Ablation study

The main novelty of our approach is the combination of weighted BON features with CNN-generated character features in the neural architecture. We carried out an ablation study in order to determine the contribution of these two types of features in the overall performance. To measure the contribution of weighted BON features, we removed the part of the system that deals with the convolutional processing of the character sequence input (the left side of the feature engineering part sketched in Figure 1). On the other hand, we removed the TF-IDF/BM25 matrix input in order to determine the contribution of the CNN-generated character features. Only TF-IDF weighting was used in the ablation study. The results of the study are presented in Table 7.

In all cases, classifier with only TF-IDF-weighted BON features (BON classifier) performs better than the classifier with only CNN-based features (CNN classifier), which also raises questions about the established deep learning paradigm that in a large majority of cases relies only on the automatically generated neural features. In DSLCC v4.0, the difference in performance is the largest in the case of Portuguese language variety classification, measuring more than four percentage points. If we ignore the language group classification, which is apparently trivial for all

Table 8. Results of the error analysis on Slavic documents misclassified by the BON classifier and correctly classified by the CNN classifier and on Slavic documents misclassified by the CNN classifier and correctly classified by the BON classifier

Group	Num. doc.	Prop. of doc.	Avg. doc. length
BON misclassifed			
No named entities	57	0.31	27.14
Misleading named entities	17	0.09	38.18
Clarifying named entities	28	0.15	33.14
Unrelated named entities	83	0.45	35.04
All	185	1.00	32.61
CNN misclassifed			
No named entities	81	0.30	31.43
Misleading named entities	36	0.13	45.67
Clarifying named entities	58	0.21	35.53
Unrelated named entities	99	0.36	34.98
All	274	1.00	35.45

three versions of the system, the difference in performance is the smallest for the French language variety classification, only around one percentage point.

By combining both types of features, we manage to surpass the performance of the BON classifier on all language groups in the DSLCC v4.0 but the Malay and Indonesian pair. Here, the BON classifier beats the classifier with the combination of both types of features by a small margin of 0.2 percentage points. The synergy effect is the largest in case of Spanish language variety, where we improve the performance of the BON classifier by almost three percentage points. Overall performance of the classifier on all the languages is improved by about one percentage point in comparison to the BON classifier.

Results on smaller data sets are somewhat hard to generalize. In the case of ADIC, the performance gap between BON and CNN is almost 11 percentage points. The bad performance of the CNN classifier in this case also most likely outweighs any positive synergy effect, causing the classifier that uses a combination of both feature types to perform slightly (by about 0.3 percentage points) worse than the BON classifier (which is therefore a new state-of-the-art classifier for the ADIC data set). In the case of GDIC, the performance gap is smaller (about 1.3 percentage points) and there is some synergy effect between the two classifiers.

In order to determine what types of texts are better predicted with the BON classifier and what types of text are better predicted with the CNN classifier, we performed the same error analysis as in Section 5.3 on 185 Slavic documents, which were correctly classified by the CNN classifier and misclassified by the BON classifier, and on 274 documents which were correctly classified by the BON classifier and misclassified by the CNN classifier. Results are presented in Table 8. We can see that on average both of these documents are shorter (32.61 and 35.45 words long) than an average document in the Slavic sub-corpus (39.18 words long). Similar share of documents with no named entities was misclassified by both classifiers but there are differences in shares when it comes to other classes. Both BON and CNN classifiers performed the worst on documents containing only unrelated named entities but the share of these documents in the overall distribution of misclassified documents is much bigger for the BON classifier (0.45 vs. 0.36). On the other hand, documents containing clarifying named entities represent a smaller share in the distribution of documents misclassified by the BON classifier (0.15 vs. 0.21). These results are in accordance with the hypothesis that the BON classifier relies to a larger extent on named entities than the CNN classifier. The share of documents with misleading named entities is the smallest in distributions for both classifiers, which was not the case in the error analysis in Section 5.3 (see Table 6), where the smallest share presented documents with only clarifying named entities. This suggests that both classifiers struggle with these documents and are in most cases misclassified by both classifiers; therefore (as this ablation study is focused on the differences between the BON and CNN classifiers), these documents were not manually analyzed.

7. Workflow for language variety classification

The AP—and larger NLP—community encourages reproducibility of results and code sharing[l]; therefore, our source code is published at http://source.ijs.si/mmartinc/NLE_2017/. Since AP is also a very interdisciplinary field, we also believe it is important to make our tools available to the users outside of the programming community (e.g., linguists or social scientists) with lower level of technical skills.

In our previous work (Martinc and Pollak 2018), we have already implemented a set of pretrained gender classification models into a cloud-based visual programming platform ClowdFlows (http://clowdflows.org) (Kranjc *et al.* 2012). These tools can be used out-of-the-box and are therefore appropriate for the less tech savy members of the AP community. The ClowdFlows platform employs a visual programming paradigm in order to simplify the representation of complex data mining procedures into visual arrangements of their building blocks. Its graphical user interface is designed to enable the users to connect processing components (i.e., widgets) into executable pipelines (i.e., workflows) on a design canvas, reducing the complexity of composition and execution of these workflows. The platform also enables online sharing of the composed workflows.

We took all our pretrained models for language variety classification (six models for six language groups and the general model for distinguishing between different language groups from the DSLCC v4.0, and German and Arabic models used for ADIC and GDIC classification) and packed them in a widget *Language Variety Classifier*. The widget takes a Pandas dataframe (McKinney 2011) containing the corpus as an input and returns a dataframe with an additional column with predicted language/language variety labels. The user needs to define the name of the column containing text documents as a parameter and choose the language group (or language parameter value *all* in order to use the general classifier) according to the input text.

Workflow in Figure 20 (available at http://clowdflows.org/workflow/13322/) is a ClowdFlows implementation of the two-step approach described in Section 4.2 for the language variety classification, illustrated on the DSLCC v4.0 test set. The corpus is loaded from a CSV file with two columns (one for texts and one for true labels) with the help of the *Load corpus from CSV* widget and passed on to the *Language variety classifier* widget, which predicts general language groups for all the texts. The *Filter corpus* widgets are used to split the corpus according to the predicted language group labels. Each of the slices is then fed into six different *Language variety classifier* widgets responsible for intra-language group classification. They output a Pandas dataframe with an additional column containing the predicted variety labels for each corpus slice. The corpus is reassembled with the help of the *Concatenate corpora* widget. The reassembled corpus and the six sub-corpora are then fed into seven *Calculate F1 and accuracy* widgets, which are in fact subprocess widgets[m], each of them containing a subprocess for calculating the accuracy and weighted F1 score[n] of the classification. The results of the classification are written to a table with the help of an *Evaluation results to table* widget. We have presented a repeatable and

[l]For example, this is the Github repository for the PAN shared task: https://github.com/pan-webis-de

[m]More information about the different types of widgets in the ClowdFlows platform is available at the ClowdFlows documentation page https://clowdflows.readthedocs.io/en/latest/.

[n]The results produced by the workflow vary very slightly from the results reported in Section 4 because Theano (Bergstra *et al.* 2011) is used as Keras backend in the ClowdFlows platform instead of Tensorflow (Abadi *et al.* 2016), which is used for producing the results reported in Section 4.

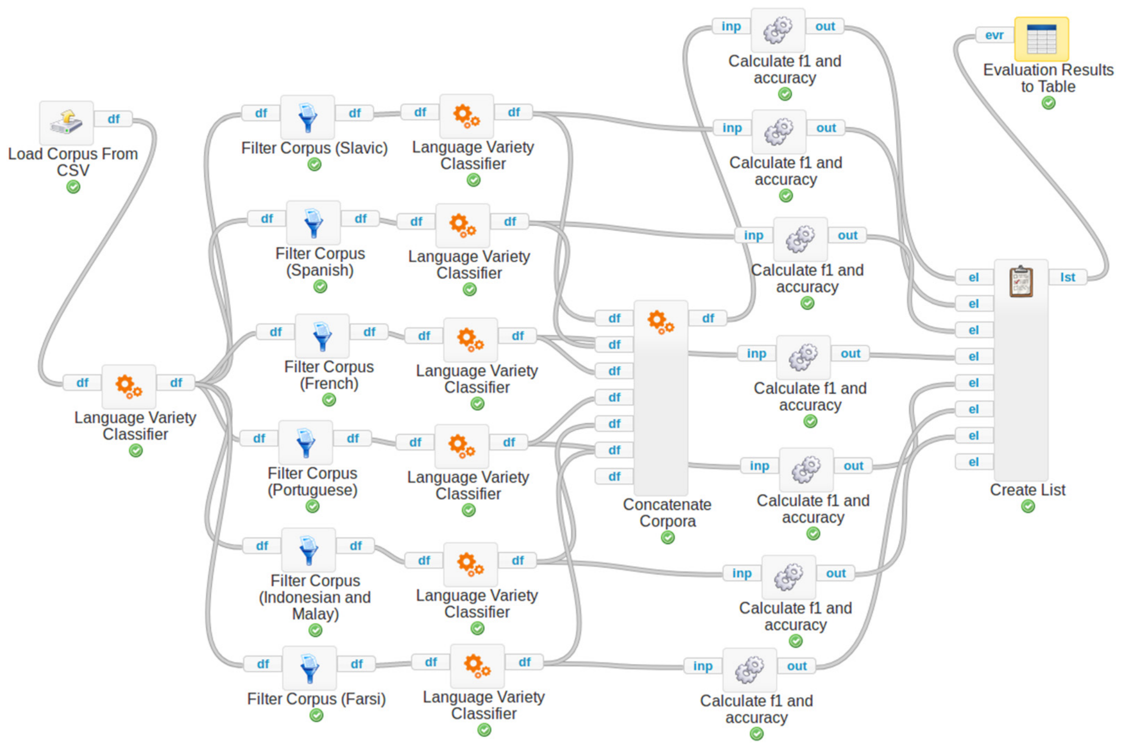

Load Corpus From CSV
Language Variety Classifier
Filter Corpus (Slavic)
Filter Corpus (Spanish)
Filter Corpus (French)
Filter Corpus (Portuguese)
Filter Corpus (Indonesian and Malay)
Filter Corpus (Farsi)
Language Variety Classifier
Concatenate Corpora
Calculate f1 and accuracy
Evaluation Results to Table
Create List
df
inp
out
el
lst
evr

Figure 20. ClowdFlows implementation of the two-step approach for the language variety classification on the DSLCC v4.0. Workflow is publicly available at `http://clowdflows.org/workflow/13322/`.

transparent evaluation workflow, which can be easily tested on novel test sets, but note that the Language variety classifier widget can also be used in novel workflows, for assigning the language of unlabeled text segments. The simplest use would be to input a file with text that user wants to label in a CSV format and connect it to the two-step language classification widgets in order to obtain the labeled corpus (`http://clowdflows.org/workflow/13670/`).

8. Discussion and conclusions

In this paper, we present an original neural language variety classifier. The main novelty is the architecture that is capable of leveraging character-level and more global document/corpus-level information by combining weighted BON features with character-based CNN features. The system was tested on the DSLCC v4.0, ADIC and GDIC corpora, used in the VarDial shared tasks, and managed to outperform state-of-the-art approaches developed in the scope of the shared task on two (including on the benchmark DSLCC v4.0) out of three corpora. An ablation study shows that weighted BON features generally contribute more than CNN-based features. This is in accordance with the previous results in the AP shared tasks where BON-based classification systems were always the winners. On the other hand, our experiments showed that replacing TF-IDF weighting with BM25 weighting in most cases does not improve performance, which is not in accordance with the previous research (Bestgen 2017). Our system is also openly available as a workflow in the ClowdFlows platform for less tech savy members of the AP community.

The experiments on the DSLCC v4.0 have shown that building a neural architecture outperforming the popular SVM BON classification combination on the language variety task is possible, although the performance gains are not very large. With some additional language-group-specific parameter tweaking the performance could be improved, but we decided against this idea in order to preserve the generic nature of the common architecture, which is currently capable of producing state-of-the-art predictions for six different language groups.

The system also proved to be competitive on the much smaller ADIC corpus (minimally outperforming state of the art) but failed to achieve competitive performance on GDIC (where the winning system HeLI was proposed by Jauhiainen *et al.* (2018a)).

We can speculate why this is the case. The results of the error analysis indicate a deterioration in performance of the proposed system on shorter documents. On the other hand, results of the VarDial 2018 shared tasks suggest that the performance of the HeLI system deteriorates less on shorter texts in comparison to other systems participating in shared tasks, since it ranked first on GDIC, where the documents are on average nine words long, and in the Vardial 2018 ILI shared task, where the task was to classify sentences[o], but only ranked fifth in the VarDial 2018 Discriminating between Dutch and Flemish in Subtitles task where the average document was 34.64 words long. Another hypothesis is that the proposed system is more reliant on named entities than the HeLI system, and therefore performs worse on GDIC, since this is the only corpus that does not contain news excerpts or news channel transcripts but transcripts of interviews with the dialect speakers and supposedly contains less named entities. We plan to test these hypotheses in the future work. We might also be able to boost the performance of our system on the GDIC data set by adjusting hyperparameters in order to make the network better suited for the classification of much shorter documents in the GDIC corpus, since currently a lot of data (e.g., *n*-grams that appear in less than five documents, character sequences filtered out by an aggressive max pooling ...) is discarded.

Small performance gains over the current state of the art also raise a question, how much better can automatic discrimination between similar languages actually get? The only study about the theoretical limit of the classification performance on the DSLCC that we are aware off was conducted by Goutte *et al.* (2016) on the DSLCC v2.0 used in the Vardial 2015 DSL shared task, which partially overlaps with the DSLCC v4.0 (Slavic, Malay and Indonesian, and Portuguese parts of the corpus are the same). First, they measured the upper bound on accuracy by taking all the predictions generated by all the systems which participated in the shared task and combining them using ensemble fusion methods such as plurality voting and Oracle. In the plurality voting, the label with most votes (i.e., the label predicted by most systems) is selected as correct and the conducted experiments showed that small improvements (of about 0.5 percentage point) over the best single system can be achieved. The Oracle method for determining the upper-bound performance on the other hand assigns the correct class label for an instance if at least one system classified the instance correctly. This gave them a very optimistic potential accuracy upper boundary of 99.83%.

In order to determine if the instances misclassified by the Oracle method can be correctly classified by humans, they conducted additional evaluation experiments. As it turns out, the difficulty of classification varies across different language groups. Discriminating between the three Slavic languages (Bosnian, Croatian, and Serbian) proved to be the most difficult. For 5 out of 12 instances misclassified by the Oracle method, none of the 6 annotators was able to correctly classify them. On these 12 examples the mean annotator accuracy was 16.66%, which is in fact 16.67% below the random baseline of 33.33%. On the other hand, discriminating between Brazilian and European Portuguese proved more feasible and the mean annotator accuracy on the misclassified instances was 67.50%, 17.50% above the 50% baseline.

This suggests that, at least for some varieties, the upper bound of automatic variety classification has not yet been reached, since our method achieved only 94.80% accuracy on the Portuguese language group. The conducted error analysis (see Section 5.3) on Slavic language varieties also showed that 10% of misclassified documents contained only clarifying named entities; therefore, any human annotator with some basic knowledge about Serbia, Bosnia, and Croatia would be able to classify them correctly without too much difficulty. This would suggest that further improvements on automatic language variety classification are possible, perhaps by employing

[o]We were unable to obtain the average document length for this data set since the number of tokens in the data set is not published in the Vardial 2018 report (Zampieri *et al.* 2018).

transfer learning techniques (Devlin *et al.* 2018) that would provide the classifier with the needed background information. We plan to test the transfer learning approach in the future.

CNNs have been so far the most successful neural architecture for language variety classification but the conducted ablation study shows that the produced features do have some deficiencies that make them less successful than weighted BON features. As shown, the proposed approach of feeding an additional weighted BON matrix into the network does partially compensate for these deficiencies on the language variety classification tasks but further work of exploring the synergy effects of combining automatically generated neural features and weighted features on a number of different NLP tasks and neural architectures is still needed. Feeding the sparse weighted BON matrix into the network does, however, have a drawback of drastically increasing the number of network parameters, which tends to lead to overfitting and increased computational costs. We managed to minimize these negative side effects mostly by an extensive use of dropout and by removing *n*-grams with low document frequencies from the input matrix, but perhaps a somewhat more efficient solution would be to avoid feeding the BON matrix to the neural classifier altogether. Therefore in future work, we plan to propose methods by which we would inject global document/corpus-level information into CNN-based features directly, in order to fix their current deficiencies. In that way combining them with the features that are the result of the more traditional feature engineering would no longer be required. Another option we also plan to explore is building heterogeneous ensembles of traditional SVM BOW-based models and CNNs and see if the performance gains are comparable to the proposed system.

Another line of future research will deal with building better and more useful tools for users with lower level of technical skills. Currently, the ClowdFlows platform does not support training of new neural classification models due to high level of resource consumption of these operations which would negatively affect the scalability of the platform, and since it does not yet support graphics processing unit (GPU) acceleration, which would allow for training of the models in a more reasonable time. The newer version of the ClowdFlows platform, on which the work has already begun, will address all these deficiencies and will allow for training of neural classification models on new varieties and therefore increase the overall usefulness of the system.

Acknowledgements. This paper is supported by European Union's Horizon 2020 research and innovation program under grant agreement No 825153—project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media) and grant agreement No 76966—project SAAM (Supporting Active Ageing through Multimodal coaching). The authors also acknowledge the financial support from the Slovenian Research Agency for research core funding for the program Knowledge Technologies (No P2-0103) and for the project TermFrame—Terminology and Knowledge Frames across Languages (No J6-9372). The results of this paper reflect only the author's view and the Commission is not responsible for any use that may be made of the information it contains. The Titan Xp used for this research was donated by the NVIDIA Corporation.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J. and Kudlur, M. (2016). Tensorflow: A system for large-scale machine learning. In *OSDI,* vol. 16, pp. 265–283.

Ali, A., Dehak, N., Cardinal, P., Khurana, S., Yella, S.H., Glass, J. and Renals, S. (2015). Automatic dialect detection in arabic broadcast speech. In *Proceedings of Interspeech*, pp. 2934–2938. San Francisco, USA: ISCA.

Ali, M. (2018a). Character level convolutional neural network for German dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 172–177. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Ali, M. (2018b). Character level convolutional neural network for Arabic dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 122–127. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Ali, M. (2018c). Character level convolutional neural network for Indo-Aryan language identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 283–287. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Alvarez-Carmona, M.A., López-Monroy, A.P., Montes-y-Gómez, M., Villasenor-Pineda, L. and Escalante, H.J. (2015). INAOE's participation at PAN'15: Author profiling task. In *Working Notes Papers of the CLEF*. Toulouse, France: CEUR Workshop Proceedings.

Belinkov, Y. and Glass, J. (2016). A character-level convolutional neural network for distinguishing similar languages and dialects. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 145–152. Osaka, Japan: The COLING 2016 Organizing Committee.

Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H. and Nissim, M. (2017). N-gram: New Groningen author-profiling model. In *CLEF 2017 Evaluation Labs and Workshop - Working Notes Papers*. Dublin, Ireland: CEUR Workshop Proceedings.

Bergstra, J., Bastien, F., Breuleux, O., Lamblin, P., Pascanu, R., Delalleau, O. and Bengio, Y. (2011). Theano: Deep learning on GPUs with python. In *NIPS 2011, BigLearning Workshop, Granada, Spain*, vol. 3, pp. 1–48.

Bestgen, Y. (2017). Improving the character n-gram model for the DSL task with BM25 weighting and less frequently used feature sets. In *proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 115–123. Valencia, Spain: Association for Computational Linguistics.

Bjerva, J. (2016). Byte-based language identification with deep convolutional networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 119–125. Osaka, Japan: The COLING 2016 Organizing Committee.

Chollet, F. (2015). Keras: Deep learning library for theano and tensorflow. `https://keras.io`.

Cianflone, A. and Kosseim, L. (2017). N-gram and neural language models for discriminating similar languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 243–250. Osaka, Japan: The COLING 2016 Organizing Committee.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K. and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research* **12**, 2493–2537.

Çöltekin, Ç. and Rama, T. (2016). Discriminating similar languages with linear SVMs and neural networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 15–24. Osaka, Japan: The COLING 2016 Organizing Committee.

Çöltekin, Ç., Rama, T. and Blaschke, V. (2018). Tübingen-Oslo team at the VarDial 2018 evaluation campaign: An analysis of n-gram features in language variety identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 55–65. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Criscuolo, M. and Aluisio, S.M. (2017). Discriminating between similar languages with word-level convolutional neural networks. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 124–130. Valencia, Spain: Association for Computational Linguistics.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Goutte, C., Léger, S. and Carpuat, M. (2014). The NRC system for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pp. 139–145. Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Goutte, C., Léger, S., Malmasi, S. and Zampieri, M. (2016). Discriminating similar languages: Evaluations and explorations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pp. 1800–1807. Portorož, Slovenia: European Language Resources Association.

Jauhiainen, T., Jauhiainen, H. and Lindén, K. (2018a). HeLI-based experiments in Swiss German dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 254–262. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Jauhiainen, T., Jauhiainen, H. and Lindén, K. (2018b). Iterative language model adaptation for Indo-Aryan language identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 66–75. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2016). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 427–431. Valencia, Spain: Association for Computational Linguistics.

Kingma, D.P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*. San Diego, California, USA: DBLP.

Kranjc, J., Podpečan, V. and Lavrač, N. (2012). ClowdFlows: A cloud based scientific workflow platform. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 816–819. Bristol, UK: Springer.

López-Monroy, A.P., Montes-y-Gómez, M., Escalante, H.J. and Pineda, L.V. (2014). Using intra-profile information for author profiling. In *CLEF (Working Notes)*, pp. 1116–1120. Sheffield, UK: CEUR Workshop Proceedings.

Malmasi, S. and Dras, M. (2015). Language identification using classifier ensembles. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pp. 35–43. Hissar, Bulgaria: Association for Computational Linguistics.

Malmasi, S. and Zampieri, M. (2016a). Arabic dialect identification in speech transcripts. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 106–113. Osaka, Japan: The COLING 2016 Organizing Committee.

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A. and Tiedemann, J. (2016b). Discriminating between similar languages and arabic dialect identification: A report on the third DSL shared task. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 1–14. Osaka, Japan: The COLING 2016 Organizing Committee.

Martinc, M. and Pollak, S. (2018). Reusable workflows for gender prediction. In *Language Resources and Evaluation Conference (LREC 2018) Proceedings*, pp. 515–520. Miyazaki, Japan: European Language Resources Association.

McKinney, W. (2011). Pandas: A foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, pp. 1–9.

Miura, Y., Taniguchi, T., Taniguchi, M. and Ohkuma, T. (2017). Author profiling with word + character neural attention network. In *CLEF (Working Notes)*. Dublin, Ireland: CEUR Workshop Proceedings.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. and Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research* **12**, 2825–2830.

Rangel, F., Rosso, P., Koppel, M., Stamatatos, E. and Inches, G. (2013). Overview of the author profiling task at PAN 2013. In *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, pp. 352–365. Valencia, Spain: Springer.

Rangel Pardo, F.M., Celli, F., Rosso, P., Potthast, M., Stein, B. and Daelemans, W. (2015). Overview of the 3rd author profiling task at PAN 2015. In *CLEF 2015 Evaluation Labs and Workshop Working Notes Papers*, pp. 1–8. Toulouse, France: CEUR Workshop Proceedings.

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M. and Stein, B. (2016). Overview of the 4th author profiling task at PAN 2016: Cross-genre evaluations. In Balog, K. et al. (ed.) *Working Notes Papers of the CLEF 2016 Evaluation Labs*. CEUR Workshop Proceedings, pp. 750–784. Évora, Portugal: CEUR Workshop Proceedings.

Rangel, F., Rosso, P., Potthast, M. and Stein, B. (2017). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. In *Working Notes Papers of the CLEF*. Dublin, Ireland: CEUR Workshop Proceedings.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval* **3**(4), 333–389.

Samardzic, T., Scherrer, Y. and Glaser, E. (2016). Archimob-a corpus of spoken Swiss German. In *Proceedings of LREC 2016*, pp. 4061–4066. Portorož, Slovenia: European Language Resources Association.

Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P. and Barrón-Cedeño, A. (2014). Overview of the author identification task at PAN 2014. In *CLEF 2014 Evaluation Labs and Workshop Working Notes Papers, Sheffield, UK, 2014*, pp. 1–21. Sheffield, UK: CEUR Workshop Proceedings.

Tan, L., Zampieri, M., Ljubešic, N. and Tiedemann, J. (2014). Merging comparable data sources for the discrimination of similar languages: The DSL corpus collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, pp. 11–15. Reykjavik, Iceland: European Language Resources Association.

Vollenbroek, M.B., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J. and Nissim, M. (2016). Gronup: Groningen user profiling. In *Notebook for PAN at CLEF*, pp. 846–857. Évora, Portugal: CEUR Workshop Proceedings.

Zampieri, M., Tan, L., Ljubešić, N. and Tiedemann, J. (2014). A report on the DSL shared task 2014. In *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, pp. 58–67. Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Zampieri, M., Tan, L., Ljubešić, N., Tiedemann, J. and Nakov, P. (2015). Overview of the DSL shared task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pp. 1–9. Hissar, Bulgaria: Association for Computational Linguistics.

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J. and Aepli, N. (2017). Findings of the VarDial evaluation campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 1–15. Valencia, Spain: Association for Computational Linguistics.

Zampieri, M., Malmasi, S., Nakov, P., Ali, A., Shon, S., Glass, J. and Van der Lee, C. (2018). Language identification and morphosyntactic tagging: The second VarDial evaluation campaign. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 1–17. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Cite this article: Martinc M and Pollak S. Combining *n*-grams and deep convolutional features for language variety classification. *Natural Language Engineering* **25**, 607–632. https://doi.org/10.1017/S1351324919000299

2.6 Final Remarks

As was already stated in Section 1, the AP research described in this chapter started several years ago, when neural networks were scarcely used due to deficiencies, such as the lack of interpretability, high resource demand, and a lack of an efficient weighting scheme. In this chapter, we have shown that some of these limitations of purely neural approaches can be avoided by using a hybrid neuro-symbolic approach, which improves semantic modelling and potentially, by feeding the neural network BON-based symbolic features, which are easy to interpret, also interpretability[9]. The high resource demand was not addressed in our work in AP, and, as we explain in section 5.2, this remains an important deficiency of the proposed strategy.

The recent trends in the field of AP go towards the usage of purely neural approaches due to several important improvements presented by the novel architectures and learning paradigms. The transformer architecture (Vaswani et al., 2017) improved semantic modelling by leveraging the transfer learning paradigm and by proposing the novel multi-head attention mechanism that determines the attention the model should pay to a specific word and therefore works as an efficient weighting scheme, and, as we show in Section 4.3, can also offer some insights into the inner workings of the model, improving interpretability. Transformers, which are these days massively employed for a range of different NLP tasks, improved the deficiencies of neural approaches that we tried to tackle with our proposed approach, therefore it is not surprising that they have been recently employed in several AP shared tasks (Chakravarthi et al., 2021; Gaman et al., 2020). Interestingly though, they are not dominating in these competitions. In the latest VarDial competition (Chakravarthi et al., 2021), non-neural symbolic models won both language (Romanian and Dravidian) identification tasks, and a team, which experimented with both types of models (a symbolic Naive Bayes classifier and two types of transformers, multilingual BERT (Devlin et al., 2019) and XLM-RoBERTa (Conneau et al., 2020)), reported that the symbolic model outperformed transformers.

This indicates that at least for some AP tasks and languages state-of-the-art neural architectures should not be the default choice, since simple symbolic models still offer comparable performance. We can therefore conclude that while the trends towards neural models is clear, these models should be further improved and adapted for some specific tasks and languages, perhaps by employing some of the strategies that we present in this thesis, e.g. by feeding these models additional BON or taxonomy-based features.

[9]Although, admittedly, the interpretation aspect was not the main focus of our studies in AP.

Chapter 3

Unsupervised Readability Prediction With a Combination of Neural Language Model Statistics and Shallow Lexical Indicators

In this chapter, we present a novel methodology for determining the readability of texts. In Section 3.1, we explain what is the readability of a text and how to measure it, and in Section 3.2, we discuss existing approaches to text readability. Section 3.3 constitutes the core of the chapter, in which we summarize our approach towards readability prediction that combines statistics derived from neural language models and symbolic lexical indicators. This chapter also contains enclosed journal publication *Supervised and unsupervised neural approaches to text readability* (Martinc, Pollak, & Robnik-Šikonja, 2021), which describes the proposed readability approach in detail.

3.1 Introduction

Readability represents a relation between a text and the cognitive load of a reader to comprehend it. It is a complex phenomenon that is not only influenced by text properties, such as syntactic and lexical complexity, and discourse cohesion, but also by the background knowledge of a reader (Crossley et al., 2017). Nevertheless, this complexity tends to be ignored by traditional measures for readability detection, which in most cases focus only on shallow lexical and syntactic features, such as word and sentence length (Davison & Kantor, 1982).

These formulas work quite well in practice but have faced harsh criticism concerning their reductionism, weak statistical basis, subjectivity and cultural specificity (Crossley et al., 2017). To face this criticism, more recently unsupervised approaches for readability detection based on newer NLP techniques have been devised. These approaches also leverage high-level textual features for readability modelling, such as semantic and discursive properties of texts, and therefore address some of the deficiencies of traditional measures. Nevertheless, they tend to perform worse than much simpler traditional readability formulas (Todirascu et al., 2016).

Another option is to tackle the readability in a supervised way and consider it as a classification or regression task. These approaches in most cases still rely on extensive feature engineering (Petersen & Ostendorf, 2009; Schwarm & Ostendorf, 2005; Vajjala & Meurers, 2012) and generally yield better results than unsupervised approaches. They nevertheless require labelled readability datasets for training, which are scarce. It has also been shown that the transferability of these approaches between different corpora and languages is limited (Filighera et al., 2019; Xia et al., 2016).

Interestingly, tackling the readability of the text with neural methods is still quite rare, even though some approaches do exist (Filighera et al., 2019; Nadeem & Ostendorf, 2018). Furthermore, while features for readability detection based on traditional n-gram language models can be found in many of the readability studies (Petersen & Ostendorf, 2009; Schwarm & Ostendorf, 2005; Xia et al., 2016), no study employs neural language models for the task at hand, even though language modelling has been drastically improved with these types of models (Mikolov et al., 2011).

In our study (Martinc, Pollak, & Robnik-Šikonja, 2021), we propose a novel unsupervised approach to readability measurement that combines symbolic features, i.e., shallow lexical and syntactic indicators of readability, with neural language model statistics. The main advantage of this approach is that it requires no labelled training set and can therefore also be used in languages without labelled resources. We show that the approach is transferable across different languages and domains since it is capable of contextualizing the readability due to the trainable nature of the neural language model. We also demonstrate that the proposed measure of readability, which we named RSRS (ranked sentence readability score), has good correlation with true readability scores.

3.2 Related Work

In this section we focus on the work related to the proposed unsupervised approach towards readability detection. For a more comprehensive overview of methods for readability detection, which also covers supervised approaches, we refer the reader to our paper (Martinc, Pollak, & Robnik-Šikonja, 2021), which is encapsulated below.

Traditional readability formulas try to construct an interpretable statistical formulation with a good correlation to human understanding of readability. The simplest example of this type of formula would be average sentence length (ASL). Other more comprehensible formulas also consider other statistical factors, such as word length and word difficulty. For example, the Gunning fog index (Gunning, 1952) (GFI) tries to estimate the years of formal education a person needs to understand the text. It is calculated according to the following equation:

$$\text{GFI} = 0.4(\frac{totalWords}{totalSentences} + 100\frac{longWords}{totalSentences}),$$

where *longWords* are words longer than 7 characters. Higher values of the index indicate lower readability.

Other formulas, such as Flesch reading ease (Kincaid et al., 1975) (FRE), Flesch-Kincaid grade level (Kincaid et al., 1975) (FKGL), Automated readability index (Smith & Senter, 1967) (ARI), Dale-Chall readability formula (Dale & Chall, 1948) (DCRF) and SMOG grade (Simple Measure of Gobbledygook) (Mc Laughlin, 1969) consider very similar statistics and are also in most cases employed for educational purposes.

Another distinct group of readability features are the so-called discourse cohesion features, which are either related to the notion of **coherence**, defined as the "semantic property of discourse, based on the interpretation of each sentence relative to the interpretation of other sentences" (Van Dijk, 1977) or to the notion of **cohesion**, defined as "a property of text represented by explicit formal grammatical ties (discourse connectives) and lexical ties that signal how utterances or larger text parts are related to each other". Studies that focus on coherence investigate if a specific text can be interpreted as a coherent message (and not as a set of unconnected words) by for example analysing words that express connectives (e.g., *because, consequently, as a result*, etc.) (Sheehan et al., 2014). On the other hand, studies that focus on the notion of cohesion focus on co-reference and anaphoric chain properties, entity density and entity cohesion features, lexical cohesion measures, and POS tag-based cohesion features (Todirascu et al., 2016). Proposed measures of cohesion are for example average length of reference chains, the total number of all/unique entities per document, frequency of content word repetition across adjacent sentences, or the ratio of pronoun and article parts-of-speech.

Lexico-semantic features have also been employed in many studies, since knowledge about the difficulty of a specific vocabulary, which these features measure, is an important aspect of reading comprehension (Collins-Thompson, 2014). The common features in this group are **Type-token ratio (TTR)**, which measures the ratio between the number of unique words and the total number of words in a text, and word and character n-grams (Vajjala & Meurers, 2012; Xia et al., 2016), which are used as features in many supervised approaches towards readability detection.

Another important feature group are syntactic features, which measure the grammatical complexity of the text. Most used features are the so-called **parse tree features**, such as for example average parse tree height, and **grammatical relations features**, such as for example the longest distance in the grammatical relation sets generated by the parser.

Finally, the features most relevant for our study are language model features. Language models try to predict a probability distribution of words from the fixed size vocabulary V, for word $w_{\mathrm{t+1}}$, given the historical sequence $w_{\mathrm{1:t}} = [w_1, ..., w_t]$. A metric called perplexity (PPL) is usually used to measure the performance of the language model, i.e, the lower the perplexity score, the better the language model predicts the words in a document. Perplexity is defined in the equation below:

$$\mathrm{PPL} = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 m(w_i)}, \tag{3.1}$$

where N is the length of the sequence and $m(w_i)$ is the probability assigned to word w_i by the language model m.

Neural language models tend to outperform n-gram language models by a high margin, since they employ much larger contextual window than n-gram language, which are limited to a small contextual window usually of up to five previous words (Mikolov et al., 2011). Nevertheless, even though they have this clear advantage in terms of performance, we are unaware of any approach that would use neural language models for measuring readability of a text. N-gram language models are on the other hand employed in several studies. In the study by Schwarm and Ostendorf (2005), one n-gram language model was trained for each readability class c in the training dataset. Statistics derived from these language models, such as perplexity and likelihood ratio, are used as features in the supervised classification approach towards readability detection. This approach of training distinct n-gram language models on each readability class and then using language model statistics for classification was also proposed in Petersen and Ostendorf (2009) and Xia et al. (2016).

3.3 Combining Neural Language Model Statistics and Shallow Lexical Indicators

The main question we investigate in our research is whether we can develop a robust new readability formula that will outperform traditional readability formulas by combining symbolic lexical sophistication indicators with neural language model statistics. **To put it differently, in this section we aim to achieve the stated goal G2 and confirm**

the hypothesis H3.

In contrast to the approach proposed in Chapter 2, here we employ late complex fusion. The main reason for this decision lies in the fact that the proposed approach is unsupervised, therefore strictly speaking there is no prediction model to which we could feed the (readability) features, as in the case of early fusion. We also refer to the proposed combination as complex, since symbolic and neural readability indicators are combined within a complex readability formula, and not just by a simple concatenation of distinct representations. The latter would be unfeasible, since neither the neural nor the symbolic readability indicators are in the vector format appropriate for concatenation.

In the proposed readability formula, a neural language model is used as a standalone unsupervised readability predictor. This differs from a procedure proposed in the related work, where separate language model is trained for each readability class (Petersen & Ostendorf, 2009; Xia et al., 2016), and allows us to exploit language model statistics in an unsupervised setting. We claim that this is possible when n-gram language model is replaced with a neural one with a much larger contextual window that spans across sentences, allowing the model to learn high-level textual properties, such as long-distance dependencies (Jawahar et al., 2019), in order to minimize negative log loss (NLL) during training. We also propose to train the language model on a large corpus, exposing it to chunks of text with different levels of complexity. We hypothesize that the model will to some extent be able to distinguish between these levels and return a lower perplexity for more readable text and a higher perplexity for text containing complex and rare language structures and unusual vocabulary.

The proposed new readability formula Ranked Sentence Readability Score (RSRS) relies on two assumptions:

- Shallow indicators of text complexity, such as the length of a sentence, have a good correlation with the readability of a text. Combining them with neural language model statistics could improve the unsupervised readability prediction.

- The perplexity scores returned by a language model for each text are an *unweighted* sum of perplexities of words in the predicted sequence. Using this sum directly might result in low correlation with text readability, since in reality a small amount of unreadable words might have a big impact on the overall readability of the text. The correlation of language model obtained statistics with the readability might therefore be improved by assigning larger weights to such unreadable words.

The procedure for calculating the RSRS is described below. First, we employ the default sentence tokenizer from the NLTK library (Bird & Loper, 2004) to split a text into sentences. We leverage a neural language model to obtain a readability estimation for each word in a specific context, i.e., by computing the word negative log-likelihood (WNLL) for each word in the sentence:

$$\text{WNLL} = -(y_t \log y_p + (1 - y_t) \log (1 - y_p))$$

In the formula above, y_p denotes the probability predicted by the neural language model that a word in a text occurs after a specific historical sequence, and y_t denotes the empirical distribution for a specific position in the sentence. Since y_t is always one, the WNLL formula can be simplified into:

$$\text{WNLL} = \log y_p$$

After that, the words in the sentence are sorted in an ascending order according to their WNLL scores, and the ranked sentence readability score (RSRS) is calculated accordingly:

$$\text{RSRS} = \frac{\sum_{i=1}^{S} \sqrt{i} \cdot \text{WNLL}(i)}{S} \tag{3.2}$$

S denotes the sentence length and i represents the rank of a word in a sentence according to its WNLL value. In order to obtain the best balance between allowing all words to contribute equally to the overall readability of the sentence and allowing only the least readable words to affect the overall readability of the sentence, the word rank is put under the square root. We calculate the average of all the RSRS scores in the text to get the readability score for an entire text.

RSRS leverages symbolic lexical sophistication indicators through index weighting, which determines the contribution of each word to the overall readability score and assures that less readable words contribute more. This is similar to the counts of long and difficult words in some traditional readability formulas, for example in GFI. Longer sentences also influence the value of RSRS, since the square roots of the word rank weights become larger with increased sentence length. In traditional formulas such as GFI, FRE, FKGL, ARI, and DCRF, the same effect is achieved by calculating the ratio between the total number of words and the total number of sentences.

RSRS score also avoids the reductionism of traditional readability formulas by considering neural language model statistics, which carry high-level structural and semantic information. The WNLL score computed for each word depends on the context in which the word appears in, and the assumption is that words appearing in more complex grammatical and lexical contexts will have a higher WNLL. Presumably, WNLL will also consider semantics of the text, since documents with semantics dissimilar to the documents in the language model training set will likely have a negative effect on the performance of the language model. This means that by training the language model on a text from appropriate language and genre, one can customize and personalize the RSRS for a specific task.

We employ three neural architectures for language modelling, a recurrent language model (RLM) based on long short-term memory (LSTM) architecture proposed by Y. Kim et al. (2016), a temporal convolutional network (TCN) proposed by Bai et al. (2018), and Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), which employs a *masked language model* objective for training, where a predefined percentage of randomly chosen words from the input word sequence is masked, and the objective is to predict these masked words from the unmasked context.

For English, we train language models on three datasets with different readabilities, Wiki-normal (containing articles from English Wikipedia), Wiki-simple (containing articles from Simple Wikipedia), and Wiki-balanced (containing a balanced amount of articles from English Wikipedia and Simple Wikipedia). See Table 3.1 for dataset statistics. For Slovenian, we only train the language model on a KRES-balanced corpus (Logar et al., 2012), containing a balanced amount of text from children magazines, teenager magazines, and magazines targeting adult audiences. We also test the viability of using a language model trained on a large general corpus for readability prediction. For English, we employ the English BERT language model, trained on large corpora (Google Books Corpus (Goldberg & Orwant, 2013) and Wikipedia) of about 3300M words containing mostly texts for adult English speakers and for Slovenian, we employ the CroSloEngual BERT model trained on corpora from three languages, Slovenian (1.26G words), Croatian (1.95G words), and English (2.69G words) (Ulčar & Robnik-Šikonja, 2020a).

We test the proposed approach on three English (Newsela (Xu et al., 2015), OneStopEnglish (Vajjala & Lučić, 2018) and WeeBit (Vajjala & Meurers, 2012)) and one

Table 3.1: Readability classes, number of documents, tokens per specific readability class and average tokens per document in each readability corpus.

Readability class	#documents	#tokens	#tokens per doc.
		Wikipedia	
simple	130,000	10,933,710	84.11
balanced	130,000	10,847,108	83.44
normal	130,000	10,719,878	82.46
		OneStopEnglish	
beginner	189	100,800	533.33
intermediate	189	127,934	676.90
advanced	189	155,253	820.49
All	567	383,987	677.23
		WeeBit	
age 7-8	600	77,613	129.35
age 8-9	600	100,491	167.49
age 9-10	600	159,719	266.20
age 10-14	600	89,548	149.25
age 14-16	600	152,402	254.00
All	3,000	579,773	193.26
		Newsela	
2nd grade	224	74,428	332.27
3rd grade	500	197,992	395.98
4th grade	1,569	923,828	588.80
5th grade	1,342	912,411	679.89
6th grade	1,058	802,057	758.09
7th grade	1,210	979,471	809.48
8th grade	1,037	890,358	858.59
9th grade	750	637,784	850.38
10th grade	20	19,012	950.60
11th grade	2	1,130	565.00
12th	1,853	1,833,781	989.63
All	9,565	7,272,252	760.30
		KRES-balanced	
balanced	/	2,402,263	/
		Slovenian SB	
1st-ps	69	12,921	187.26
2nd-ps	146	30,296	207.51
3rd-ps	268	62,241	232.24
4th-ps	1,007	265,242	263.40
5th-ps	1,186	330,039	278.28
6th-ps	959	279,461	291.41
7th-ps	1,470	462,551	314.66
8th-ps	1,844	540,944	293.35
9th-ps	2,154	688,149	319.47
1st-hs	1,663	578,694	347.98
2nd-hs	590	206,147	349.40
3rd-hs	529	165,845	313.51
4th-hs	45	14,313	318.07
All	11,930	3,636,843	304.85

Table 3.2: Ranking (lower is better) of measures on English and Slovenian datasets sorted by the average rank on all datasets for which the measure is available.

Measure	WeeBit	OneStopEnglish	Newsela	Slovenian SB
RLM RSRS-simple	4	4	4	/
TCN RSRS-balanced	11	2	2	**1**
RLM RSRS-balanced	5	5	5	3
GFI	**1**	6	10	4
TCN RSRS-simple	12	**1**	3	/
ASL	3	12	**1**	7
FKGL	2	8	9	5
RLM RSRS-normal	6	7	6	/
TCN RSRS-normal	13	3	7	/
ARI	7	9	8	8
SMOG	8	11	11	2
DCRF	10	13	13	6
FRE	9	14	12	9
TCN perplexity-simple	16	10	15	/
TCN perplexity-balanced	15	15	16	11
BERT RSRS	14	18	14	12
RLM perplexity-balanced	18	17	17	10
RLM perplexity-simple	19	16	18	/
TCN perplexity-normal	17	19	20	/
BERT perplexity	20	21	21	13
RLM perplexity-normal	21	20	19	/

Slovenian school books (SB) test dataset. Besides differing in language, the corpora also differ in terms of semantic differences between different readability classes (Newsela and OneStopEnglish datasets are parallel corpora and WeeBit and Slovenian SB datasets are not), length of documents (Newsela and OneStopEnglish datasets on average contain much longer documents than other two datasets), genre (OneStopEnglish and Newsela datasets contain news articles, WeeBit is made of educational articles, and the Slovenian SB dataset is composed of school books), and target audience (OneStopEnglish targets adult English as a second language (ESL) learners and other datasets target children of different ages).

The ranking of different measures on English and Slovenian datasets are presented in Table 3.2. Besides ranking for the proposed RSRS score, we also report ranking for several traditional readability measures (GFI, FRE, FKGL, ARI, DCRF, SMOG, and average sentence length (ASL)) and investigate how the measured perplexities of language models correlate with the readability labels in the gold-standard corpora. The correlation is measured with the Pearson correlation coefficient (ρ) and a larger positive correlation denotes a better performance for all measures except the FRE readability measure. Here, a larger negative correlation suggests a better performance, since this formula assigns higher scores to more readable texts.

The results in terms of ranking of measures across different datasets are presented in Table 3.2. In general, RSRS measures with different language models trained on datasets with different readabilites offer competitive performance. RSRS-simple and RSRS-balanced measures offer the most robust performance across datasets from different genres and languages. On average the best ranked measure, RSRS-simple, ranked fourth on all English corpora. The second best measure, TCN RSRS-balanced, which was also employed on Slovenian SB, ranked first on Slovenian SB and second on OneStopEnglish and Newsela, but did not do well on WeeBit, where the discrepancy in readability between the language model train and test sets was too large. A bit more consistent was RLM RSRS-balanced measure, ranking fifth on all English corpora and third on Slovenian SB.

The results suggest that the readability of the language model training set is important. If the training set on average contains more complex texts than the majority of texts in the test set, as in the case of language models trained just on the Wiki-normal corpus (and also BERT), the correlation between readability and perplexity worsens, since language models trained on more complex language structures learn how to handle these difficulties. According to these results, the average readability of the training corpus should fit somewhere in the middle or even slightly below the middle of the readability range of the testing corpus. On the other hand, the choice of a neural architecture for language modelling (RLM or TCN) does not appear to have a big influence on the results.

Table 3.2 also shows that shallow readability predictors can give inconsistent results on datasets from different genres and languages, as can be seen on the example of AVS, which ranked first on Newsela and twelfth on OneStopEnglish. Another example is the SMOG measure, which performed very well on the Slovenian SB corpus (rank 2) but ranked twice as eleventh and once as eighth on the English corpora. On average, GFI measure ranked the best, ranking first on WeeBit, sixth on OneStopEnglish, tenth on Newsela, and fourth on Slovenian SB.

The perplexity-based measures show low correlation with the ground truth readability scores and perform even worse than the traditional readability measures. Perplexity is therefore not a good indicator of readability and should therefore not be used alone for readability prediction. Nevertheless, neural language model statistics do contain quite useful information when combined with other symbolic indicators of lexical sophistication. This is shown by the good performance of TCN RSRS and RLM RSRS and is especially useful when readability analysis needs to be conducted on a variety of different datasets.

The journal paper covering the methodology, experiments and results in detail is enclosed below.

Supervised and Unsupervised Neural Approaches to Text Readability

Matej Martinc
Jožef Stefan Institute, Ljubljana, Slovenia
Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
matej.martinc@ijs.si

Senja Pollak
Jožef Stefan Institute, Ljubljana, Slovenia
senja.pollak@ijs.si

Marko Robnik-Šikonja
University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
marko.robnik@fri.uni-lj.si

We present a set of novel neural supervised and unsupervised approaches for determining the readability of documents. In the unsupervised setting, we leverage neural language models, whereas in the supervised setting, three different neural classification architectures are tested. We show that the proposed neural unsupervised approach is robust, transferable across languages, and allows adaptation to a specific readability task and data set. By systematic comparison of several neural architectures on a number of benchmark and new labeled readability data sets in two languages, this study also offers a comprehensive analysis of different neural approaches to readability classification. We expose their strengths and weaknesses, compare their performance to current state-of-the-art classification approaches to readability, which in most cases still rely on extensive feature engineering, and propose possibilities for improvements.

1. Introduction

Readability is concerned with the relation between a given text and the cognitive load of a reader to comprehend it. This complex relation is influenced by many factors, such as a degree of lexical and syntactic sophistication, discourse cohesion, and background knowledge (Crossley et al. 2017). In order to simplify the problem of measuring readability, traditional readability formulas focused only on lexical and syntactic features

Submission received: 26 July 2019; revised version received: 22 November 2020; accepted for publication: 18 December 2020.

https://doi.org/10.1162/COLI_a_00398

expressed with statistical measurements, such as word length, sentence length, and word difficulty (Davison and Kantor 1982). These approaches have been criticized because of their reductionism and weak statistical bases (Crossley et al. 2017). Another problem is their objectivity and cultural transferability, since children from different environments master different concepts at different ages. For example, a word *television* is quite long and contains many syllables but is well-known to most young children who live in families with a television.

With the development of novel natural language processing (NLP) techniques, several studies attempted to eliminate deficiencies of traditional readability formulas. These attempts include leveraging high-level textual features for readability modeling, such as semantic and discursive properties of texts. Among them, cohesion and coherence received the most attention, and several readability predictors based on these text features have been proposed (see Section 2). Nevertheless, none of them seems to predict the readability of the text as well as much simpler readability formulas mentioned above (Todirascu et al. 2016).

With the improvements in machine learning, the focus shifted once again, and most newer approaches consider readability as being a classification, regression, or a ranking task. Machine learning approaches build prediction models to predict human assigned readability scores based on several attributes and manually built features that cover as many text dimensions as possible (Schwarm and Ostendorf 2005; Petersen and Ostendorf 2009; Vajjala and Meurers 2012). They generally yield better results than the traditional readability formulas and text cohesion–based methods but require additional external resources, such as labeled readability data sets, which are scarce. Another problem is the transferability of these approaches between different corpora and languages, because the resulting feature sets do not generalize well to different types of texts (Xia, Kochmar, and Briscoe 2016; Filighera, Steuer, and Rensing 2019).

Recently, deep neural networks (Goodfellow, Bengio, and Courville 2016) have shown impressive performance on many language-related tasks. In fact, they have achieved state-of-the-art performance in all semantic tasks where sufficient amounts of data were available (Collobert et al. 2011; Zhang, Zhao, and LeCun 2015). Even though very recently some neural approaches toward readability prediction have been proposed (Nadeem and Ostendorf 2018; Filighera, Steuer, and Rensing 2019), these types of studies are still relatively scarce, and further research is required in order to establish what type of neural architectures are the most appropriate for distinct readability tasks and data sets. Furthermore, language model features designed to measure lexical and semantic properties of text, which can be found in many of the readability studies (Schwarm and Ostendorf 2005; Petersen and Ostendorf 2009; Xia, Kochmar, and Briscoe 2016), are generated with traditional *n*-gram language models, even though language modeling has been drastically improved with the introduction of neural language models (Mikolov et al. 2011).

The aim of the present study is two-fold. First, we propose a novel approach to readability measurement that takes into account neural language model statistics. This approach is unsupervised and requires no labeled training set but only a collection of texts from the given domain. We demonstrate that the proposed approach is capable of contextualizing the readability because of the trainable nature of neural networks and that it is transferable across different languages. In this scope, we propose a new measure of readability, RSRS (ranked sentence readability score), with good correlation with true readability scores.

Second, we experiment to find how different neural architectures with automatized feature generation can be used for readability classification and compare their performance to state-of-the-art classification approaches. Three distinct branches of neural architectures—recurrent neural networks (RNN), hierarchical attention networks (HAN), and transfer learning techniques—are tested on four gold standard readability corpora with good results.

The article is structured as follows. Section 2 addresses the related work on readability prediction. Section 3 offers a thorough analysis of data sets used in our experiments, and in Section 4, we present the methodology and results for the proposed unsupervised approach to readability prediction. The methodology and experimental results for the supervised approach are presented in Section 5. We present conclusions and directions for further work in Section 6.

2. Related Work

Approaches to the automated measuring of readability try to find and assess factors that correlate well with human perception of readability. Several indicators, which measure different aspects of readability, have been proposed in the past and are presented in Section 2.1. These measures are used as features in newer approaches, which train machine learning models on texts with human-annotated readability levels so that they can predict readability levels on new unlabeled texts. Approaches that rely on an extensive set of manually engineered features are described in Section 2.2. Finally, Section 2.3 covers the approaches that tackle readability prediction with neural classifiers. Besides tackling the readability as a classification problem, several other supervised statistical approaches for readability prediction have been proposed in the past. They include regression (Sheehan et al. 2010), Support Vector Machine (SVM) ranking (Ma, Fosler-Lussier, and Lofthus 2012), and graph-based methods (Jiang, Xun, and Qi 2015), among many others. We do not cover these methods in the related work because they are not directly related to the proposed approach.

2.1 Readability Features

Classical readability indicators can be roughly divided into five distinct groups: traditional, discourse cohesion, lexico-semantic, syntactic, and language model features. We describe them below.

2.1.1 Traditional Features. Traditionally, readability in texts was measured by statistical readability formulas, which try to construct a simple human-comprehensible formula with a good correlation to what humans perceive as the degree of readability. The simplest of them is average sentence length (ASL), though they take into account various other statistical factors, such as word length and word difficulty. Most of these formulas were originally developed for the English language but are also applicable to other languages with some modifications (Škvorc et al. 2019).

The Gunning fog index (Gunning 1952) (GFI) estimates the years of formal education a person needs to understand the text on the first reading. It is calculated with the following expression:

$$\mathrm{GFI} = 0.4\left(\frac{totalWords}{totalSentences} + 100\frac{longWords}{totalSentences}\right)$$

where *longWords* are words longer than 7 characters. Higher values of the index indicate lower readability.

Flesch reading ease (Kincaid et al. 1975) (FRE) assigns higher values to more readable texts. It is calculated in the following way:

$$\mathrm{FRE} = 206.835 - 1.015\left(\frac{totalWords}{totalSentences}\right) - 84.6\left(\frac{totalSyllables}{totalWords}\right)$$

The values returned by the Flesch-Kincaid grade level (Kincaid et al. 1975) (FKGL) correspond to the number of years of education generally required to understand the text for which the formula was calculated. The formula is defined as follows:

$$\mathrm{FKGL} = 0.39\left(\frac{totalWords}{totalSentences}\right) + 11.8\left(\frac{totalSyllables}{totalWords}\right) - 15.59$$

Another readability formula that returns values corresponding to the years of education required to understand the text is the Automated Readability Index (Smith and Senter 1967) (ARI):

$$\mathrm{ARI} = 4.71\left(\frac{totalCharacters}{totalWords}\right) + 0.5\left(\frac{totalWords}{totalSentences}\right) - 21.43$$

The Dale-Chall readability formula (Dale and Chall 1948) (DCRF) requires a list of 3,000 words that fourth-grade US students could reliably understand. Words that do not appear in this list are considered difficult. If the list of words is not available, it is possible to use the GFI approach and consider all the words longer than 7 characters as difficult. The following expression is used in calculation:

$$\mathrm{DCRF} = 0.1579\left(\frac{difficultWords}{totalWords} * 100\right) + 0.0496\left(\frac{totalWords}{totalSentences}\right)$$

The SMOG grade (Simple Measure of Gobbledygook) (McLaughlin 1969) is a readability formula originally used for checking health messages. Similar to FKGL and ARI, it roughly corresponds to the years of education needed to understand the text. It is calculated with the following expression:

$$\mathrm{SMOG} = 1.0430\sqrt{numberOfPolysyllables\frac{30}{totalSentences}} + 3.1291$$

where the *numberOfPolysyllables* is the number of words with three or more syllables.

We are aware of one study that explored the transferability of these formulas across genres (Sheehan, Flor, and Napolitano 2013), and one study that explored transferability across languages (Madrazo Azpiazu and Pera 2020). The study by Sheehan, Flor, and Napolitano (2013) concludes that, mostly due to vocabulary specifics of different genres, traditional readability measures are not appropriate for cross-genre prediction, because they underestimate the complexity levels of literary texts and overestimate that of educational texts. The study by Madrazo Azpiazu and Pera (2020), on the other hand, concludes that the readability level predictions for translations of the same text are rarely consistent when using these formulas.

All of the above-mentioned readability measures were designed for the specific use on English texts. There are some rare attempts to adapt these formulas to other languages (Kandel and Moles 1958) or to create new formulas that could be used on languages other than English (Anderson 1981).

To show a multilingual potential of our approach, we address two languages in this study, English and Slovenian, a Slavic language with rich morphology and orders of magnitude fewer resources compared to English. For Slovenian, readability studies are scarce. Škvorc et al. (2019) researched how well the above statistical readability formulas work on Slovenian text by trying to categorize text from three distinct sources: children's magazines, newspapers and magazines for adults, and transcriptions of sessions of the National Assembly of Slovenia. Results of this study indicate that formulas that consider the length of words and/or sentences work better than formulas that rely on word lists. They also noticed that simple indicators of readability, such as percentage of adjectives and average sentence length, work quite well for Slovenian. To our knowledge, the only other study that employed readability formulas on Slovenian texts was done by Zwitter Vitez (2014). In that study, the readability formulas were used as features in the author recognition task.

2.1.2 Discourse Cohesion Features. In the literature, we can find at least two distinct notions of discourse cohesion (Todirascu et al. 2016). First is the notion of **coherence**, defined as the "semantic property of discourse, based on the interpretation of each sentence relative to the interpretation of other sentences" (Van Dijk 1977). Previous research that investigates this notion tries to determine whether a text can be interpreted as a coherent message and not just as a collection of unrelated sentences. This can be done by measuring certain observable features of the text, such as the repetition of content words or by analysis of words that explicitly express connectives (*because, consequently, as a result*, etc.) (Sheehan et al. 2014). A somewhat more investigated notion, due to its easier operationalization, is the notion of **cohesion**, defined as "a property of text represented by explicit formal grammatical ties (discourse connectives) and lexical ties that signal how utterances or larger text parts are related to each other."

According to Todirascu et al. (2016), we can divide cohesion features into five distinct classes, outlined below: co-reference and anaphoric chain properties, entity density and entity cohesion features, lexical cohesion measures, and part of speech (POS) tag-based cohesion features. **Co-reference and anaphoric chain properties** were first proposed by Bormuth (1969), who measured various characteristics of anaphora. These features include statistics, such as the average length of reference chains or the proportion of various types of mention (noun phrases, proper names, etc.) in the chain. **Entity density** features include statistics such as the total number of all/unique entities per document, the average number of all/unique entities per sentence, and so forth. These features were first proposed in Feng, Elhadad, and Huenerfauth (2009) and Feng et al. (2010), who followed the theoretical line from Halliday and Hasan (1976) and Williams (2006). **Entity cohesion** features assess relative frequency of possible transitions between syntactic functions played by the same entity in adjacent sentences (Pitler and Nenkova 2008). **Lexical cohesion measures** include features such as the frequency of content word repetition across adjacent sentences (Sheehan et al. 2014), a Latent Semantic Analysis (LSA)-based feature for measuring the similarity of words and passages to each other proposed by Landauer (2011), or a measure called Lexical

Tightness (LT), suggested by Flor, Klebanov, and Sheehan (2013), defined as the mean value of the Positive Normalized Pointwise Mutual Information (PMI) for all pairs of content-word tokens in a text. The last category is **POS tag-based cohesion features**, which measure the ratio of pronoun and article parts-of-speech, two crucial elements of cohesion (Todirascu et al. 2016).

Todirascu et al. (2016), who analyzed 65 discourse features found in the readability literature, concluded that they generally do not contribute much to the predictive power of text readability classifiers when compared with the traditional readability formulas or simple statistics such as sentence length.

2.1.3 Lexico-semantic Features. According to Collins-Thompson (2014), vocabulary knowledge is an important aspect of reading comprehension, and lexico-semantic features measure the difficulty of vocabulary in the text. A common feature is **Type-token ratio (TTR)**, which measures the ratio between the number of unique words and the total number of words in a text. The length of the text influences TTR; therefore, several corrections, which produce a more unbiased representation, such as Root TTR and Corrected TTR, are also used for readability prediction.

Other frequently used features in classification approaches to readability are ***n*-gram lexical features**, such as word and character n-grams (Vajjala and Meurers 2012; Xia, Kochmar, and Briscoe 2016). While **POS-based lexical features** measure lexical variation (i.e., TTR of lexical items such as nouns, adjectives, verbs, adverbs, and prepositions) and density (e.g., the percentage of content words and function words), **word list-based features** use external psycholinguistic and Second Language Acquisition (SLA) resources, which contain information about which words and phrases are acquired at the specific age or English learning class.

2.1.4 Syntactic Features. Syntactic features measure the grammatical complexity of the text and can be divided into several categories. **Parse tree features** include features such as an average parse tree height or an average number of noun- or verb-phrases per sentence. **Grammatical relations features** include measures of grammatical relations between constituents in a sentence, such as the longest/average distance in the grammatical relation sets generated by the parser. **Complexity of syntactic unit features** measure the length of a syntactic unit at the sentence, clause (any structure with a subject and a finite verb), and T-unit level (one main clause plus any subordinate clause). Finally, **coordination and subordination features** measure the amount of coordination and subordination in the sentence and include features such as a number of clauses per T-unit or number of coordinate phrases per clause, and so on.

2.1.5 Language Model Features. The standard task of language modeling can be formally defined as predicting a probability distribution of words from the fixed size vocabulary V, for word w_{t+1}, given the historical sequence $w_{1:t} = [w_1, \dots, w_t]$. To measure its performance, traditionally a metric called perplexity is used. A language model m is evaluated according to how well it predicts a separate test sequence of words $w_{1:N} = [w_1, \dots, w_N]$. For this case, the perplexity (PPL) of the language model $m()$ is defined as:

$$\mathrm{PPL} = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 m(w_i)} \tag{1}$$

where $m(w_i)$ is the probability assigned to word w_i by the language model m, and N is the length of the sequence. The lower the perplexity score, the better the language model predicts the words in a document—that is, the more predictable and aligned with the training set the text is.

All past approaches for readability detection that use language modeling leverage older *n*-gram language models rather than the newer neural language models. Schwarm and Ostendorf (2005) train one *n*-gram language model for each readability class *c* in the training data set. For each text document *d*, they calculate the likelihood ratio according to the following formula:

$$LR(d,c) = \frac{P(d|c)P(c)}{\sum_{\bar{c}\neq c} P(d|\bar{c})P(\bar{c})}$$

where $P(d|c)$ denotes the probability returned by the language model trained on texts labeled with class c, and $P(d|\bar{c})$ denotes probability of d returned by the language model trained on the class $\bar{c}$. Uniform prior probabilities of classes are assumed. The likelihood ratios are used as features in the classification model, along with perplexities achieved by all the models.

In Petersen and Ostendorf (2009), three statistical language models (unigram, bigram and trigram) are trained on four external data resources: Britannica (adult), Britannica Elementary, CNN (adult), and CNN abridged. The resulting 12 *n*-gram language models are used to calculate perplexities of each target document. It is assumed that low perplexity scores calculated by language models trained on the adult level texts and high perplexity scores of language models trained on the elementary/abridged levels would indicate a high reading level, and high perplexity scores of language models trained on the adult level texts and low perplexity scores of language models trained on the elementary/abridged levels would indicate a low reading level.

Xia, Kochmar, and Briscoe (2016) train 1- to 5-gram word-based language models on the British National Corpus, and 25 POS-based 1- to 5-gram models on the five classes of the WeeBit corpus. Language models' log-likelihood and perplexity scores are used as features for the classifier.

2.2 Classification Approaches Based on Feature Engineering

The above approaches measure readability in an unsupervised way, using the described features. Alternatively, we can predict the level of readability in a supervised way. These approaches usually require extensive feature engineering and also leverage many of the features described earlier.

One of the first classification approaches to readability was proposed by Schwarm and Ostendorf (2005). It relies on a SVM classifier trained on a WeeklyReader corpus,[1] containing articles grouped into four classes according to the age of the target audience. Traditional, syntactic, and language model features are used in the model. This approach was extended and improved upon in Petersen and Ostendorf (2009).

Altogether, 155 traditional, discourse cohesion, lexico-semantic, and syntactic features were used in an approach proposed by Vajjala and Lučić (2018), tested on a recently published OneStopEnglish corpus. Sequential Minimal Optimization (SMO) classifier with the linear kernel achieved the classification accuracy of 78.13% for three readability classes (elementary, intermediate, and advanced reading level).

1 `http://www.weeklyreader.com`.

A successful classification approach to readability was proposed by Vajjala and Meurers (2012). Their multilayer perceptron classifier is trained on the WeeBit corpus (Vajjala and Meurers 2012) (see Section 3 for more information on WeeBit and other mentioned corpora). The texts were classified into five classes according to the age group they are targeting. For classification, the authors use 46 manually crafted traditional, lexico-semantic, and syntactic features. For the evaluation, they trained the classifier on a train set consisting of 500 documents from each class and tested it on a balanced test set of 625 documents (containing 125 documents per each class). They report 93.3% accuracy on the test set.[2]

Another set of experiments on the WeeBit corpus was conducted by Xia, Kochmar, and Briscoe (2016), who conducted additional cleaning of the corpus because it contained some texts with broken sentences and additional meta-information about the source of the text, such as copyright declaration and links, strongly correlated with the target labels. They use similar lexical, syntactic, and traditional features as Vajjala and Meurers (2012) but add language modeling (see Section 2.1.5 for details) and discourse cohesion-based features. Their SVM classifier achieves 80.3% accuracy using the 5-fold crossvalidation. This is one of the studies where the transferability of the classification models is tested. The authors used an additional CEFR (Common European Framework of Reference for Languages) corpus. This small data set of CEFR-graded texts is tailored for learners of English (Council of Europe 2001) and also contains 5 readability classes. The SVM classifier trained on the WeeBit corpus and tested on the CEFR corpus achieved the classification accuracy of 23.3%, hardly beating the majority classifier baseline. This low result was attributed to the differences in readability classes in both corpora, since WeeBit classes are targeting children of different age groups, and CEFR corpus classes are targeting mostly adult foreigners with different levels of English comprehension. However, this result is a strong indication that transferability of readability classification models across different types of texts is questionable.

Two other studies that deal with the multi-genre prospects of readability prediction were conducted by Sheehan, Flor, and Napolitano (2013) and Napolitano, Sheehan, and Mundkowsky (2015). Both studies describe the problem in the context of the TextEvaluator Tool (Sheehan et al. 2010), an online system for text complexity analysis. The system supports multi-genre readability prediction with the help of a two-stage prediction workflow, in which first the genre of the text is determined (as being informational, literary, or mixed) and after that its readability level is predicted with an appropriate genre-specific readability prediction model. Similarly to the study above, this work also indicates that using classification models for cross-genre prediction is not feasible.

When it comes to multi- and crosslingual classification, Madrazo Azpiazu and Pera (2020) explore the possibility of a crosslingual readability assessment and show that their methodology called CRAS (Crosslingual Readability Assessment Strategy), which includes building a classifier that uses a set of traditional, lexico-semantic, syntactic, and discourse cohesion-based features works well in a multilingual setting. They also show that classification for some low resource languages can be improved by including documents from a different language into the train set for a specific language.

2 Later research by Xia, Kochmar, and Briscoe (2016) called the validity of the published experimental results into question; therefore, the reported 93.3% accuracy might not be the objective state-of-the-art result for readability classification.

2.3 Neural Classification Approaches

Recently, several neural approaches for readability prediction have been proposed. Nadeem and Ostendorf (2018) tested two different architectures on the WeeBit corpus regression task, namely, sequential Gated Recurrent Unit (GRU) (Cho et al. 2014) based RNN with the attention mechanism and hierarchical RNNs (Yang et al. 2016) with two distinct attention types: a more classic attention mechanism proposed by Bahdanau, Cho, and Bengio (2014), and multi-head attention proposed by Vaswani et al. (2017). The results of the study indicate that hierarchical RNNs generally perform better than sequential. Nadeem and Ostendorf (2018) also show that neural networks can be a good alternative to more traditional feature-based models for readability prediction on texts shorter than 100 words, but do not perform that competitively on longer texts.

Another version of a hierarchical RNN with the attention mechanism was proposed by Azpiazu and Pera (2019). Their system, named Vec2Read, is a multi-attentive RNN capable of leveraging hierarchical text structures with the help of word and sentence level attention mechanisms and a custom-built aggregation mechanism. They used the network in a multilingual setting (on corpora containing Basque, Catalan, Dutch, English, French, Italian, and Spanish texts). Their conclusion was that although the number of instances used for training has a strong effect on the overall performance of the system, no language-specific patterns emerged that would indicate that prediction of readability in some languages is harder than in others.

An even more recent neural approach for readability classification on the cleaned WeeBit corpus (Xia, Kochmar, and Briscoe 2016) was proposed by Filighera, Steuer, and Rensing (2019), who tested a set of different embedding models, word2vec (Mikolov et al. 2013), the uncased Common Crawl GloVe (Pennington, Socher, and Manning 2014), ELMo (Peters et al. 2018), and BERT (Devlin et al. 2019). The embeddings were fed to either a recurrent or a convolutional neural network. The BERT-based approach from their work is somewhat similar to the BERT-based supervised classification approach proposed in this work. However, one main distinction is that no fine-tuning is conducted on the BERT model in their experiments (i.e., the extraction of embeddings is conducted on the pretrained BERT language model). Their best ELMo-based model with a bidirectional LSTM achieved an accuracy of 79.2% on the development set, slightly lower than the accuracy of 80.3% achieved by Xia, Kochmar, and Briscoe (2016) in the 5-fold crossvalidation scenario. However, they did manage to improve on the state of the art by an ensemble of all their models, achieving the accuracy of 81.3%, and the macro averaged F_1-score of 80.6%.

A somewhat different neural approach to readability classification was proposed by Mohammadi and Khasteh (2019), who tackled the problem with deep reinforcement learning, or more specifically, with a deep convolutional recurrent double dueling Q network (Wang et al. 2016) using a limited window of 5 adjacent words. GloVe embeddings and statistical language models were used to represent the input text in order to eliminate the need for sophisticated NLP features. The model was used in a multilingual setting (on English and Persian data sets) and achieved performance comparable to the state of the art on all of the data sets, among them also on the Weebit corpus (accuracy of 91%).

Finally, a recent study by Deutsch, Jasbi, and Shieber (2020) used predictions of HAN and BERT models as additional features in their SVM model that also utilized a set of syntactic and lexico-semantic features. Although they did manage to improve the performance of their SVM classifiers with the additional neural features, they concluded that additional syntactic and lexico-semantic features did not generally improve the predictions of the neural models.

3. Data Sets

In this section, we first present the data sets used in the experiments (Section 3.1) and then conduct their preliminary analysis (Section 3.2) in order to assess the feasibility of the proposed experiments. Data set statistics are presented in Table 1.

3.1 Data Set Presentation

All experiments are conducted on four corpora labeled with readability scores:

- **The WeeBit corpus**: The articles from WeeklyReader[3] and BBC-Bitesize[4] are classified into five classes according to the age group they are targeting. The classes correspond to age groups 7–8, 8–9, 9–10, 10–14, and 14–16 years. Three classes targeting younger audiences consist of articles from WeeklyReader, an educational newspaper that covers a wide range of nonfiction topics, from science to current affairs. Two classes targeting older audiences consist of material from the BBC-Bitesize Web site, containing educational material categorized into topics that roughly match school subjects in the UK. In the original corpus of Vajjala and Meurers (2012), the classes are balanced and the corpus contains altogether 3,125 documents, 625 per class. In our experiments, we followed recommendations of Xia, Kochmar, and Briscoe (2016) to fix broken sentences and remove additional meta information, such as copyright declaration and links, strongly correlated with the target labels. We reextracted the corpus from the HTML files according to the procedure described in Xia, Kochmar, and Briscoe (2016) and discarded some documents because of the lack of content after the extraction and cleaning process. The final corpus used in our experiments contains altogether 3,000 documents, 600 per class.
- **The OneStopEnglish corpus** (Vajjala and Lučić 2018) contains aligned texts of three distinct reading levels (beginner, intermediate, and advanced) that were written specifically for English as Second Language (ESL) learners. The corpus was compiled over the period 2013–2016 from the weekly news lessons section of the language learning resources `onestopenglish.com`. The section contains articles sourced from the Guardian newspaper, which were rewritten by English teachers to target three levels of adult ESL learners (elementary, intermediate, and advanced). Overall, the document-aligned parallel corpus consists of 189 texts, each written in three versions (567 in total). The corpus is freely available.[5]

3 `http://www.weeklyreader.com`.
4 `http://www.bbc.co.uk/bitesize`.
5 `https://zenodo.org/record/1219041`.

- **The Newsela corpus** (Xu, Callison-Burch, and Napoles 2015): We use the version of the corpus from 29 January 2016 consisting of altogether 10,786 documents, out of which we only used 9,565 English documents. The corpus contains 1,911 original English news articles and up to four simplified versions for every original article, that is, each original news article has been manually rewritten up to 4 times by editors at Newsela, a company that produces reading materials for pre-college classroom use, in order to target children at different grade levels and help teachers prepare curricula that match the English language skills required at each grade level. The data set is a document-aligned parallel corpus of original and simplified versions corresponding to altogether eleven different imbalanced grade levels (from 2nd to 12th grade).
- **Corpus of Slovenian school books (Slovenian SB):** In order to test the transferability of the proposed approaches to other languages, a corpus of Slovenian school books was compiled. The corpus contains 3,639,665 words in 125 school books for nine grades of primary schools and four grades of secondary school. It was created with several aims, like studying different quality aspects of school books, extraction of terminology, and linguistic analysis. The corpus contains school books for 16 distinct subjects with very different topics ranging from literature, music, and history to math, biology, and chemistry, but not in equal proportions, with readers being the largest type of school books included.

  Whereas some texts were extracted from the Gigafida reference corpus of written Slovene (Logar et al. 2012), most of the texts were extracted from PDF files. After the extraction, we first conduct some light manual cleaning on the extracted texts (removal of indices, copyright statements, references, etc.). Next, in order to remove additional noise (tips, equations, etc.), we apply a filtering script that relies on manually written rules for sentence extraction (e.g., a text is a sentence if it starts with an uppercase and ends with an end-of-sentence punctuation) to obtain only passages containing sentences. Final extracted texts come without structural information (where does a specific chapter end or start, which sentences constitute a paragraph, where are questions, etc.), since labeling the document structure would require a large amount of manual effort; therefore we did not attempt it for this research.

  For supervised classification experiments, we split the school books into chunks 25 sentences long, in order to build a train and test set with a sufficient number of documents.[6] The length of 25 sentences was chosen due to size limitations of the BERT classifier, which can be fed documents that contain up to 512 byte-pair tokens (Kudo and Richardson 2018),[7] which on average translates to slightly less than 25 sentences.

6 Note that this chunking procedure might break the text cohesion and that topical similarities between chunks from the same chapter (or paragraphs) might have a positive effect on the performance of the classification. However, because the corpus does not contain any high-level structural information (e.g., the information about paragraph or chapter structure of a specific school book), no other more refined chunking method is possible.

7 Note that the BERT tokenizer uses byte-pair tokenization (Kudo and Richardson 2018), which in some cases generates tokens that correspond to sub-parts of words rather than entire words. In the case of Slovenian SB, 512 byte-pair tokens correspond to 306 word tokens on average.

Table 1
Readability classes, number of documents, tokens per specific readability class, and average tokens per document in each readability corpus.

Readability class	#documents	#tokens	#tokens per doc.
	Wikipedia		
simple	130,000	10,933,710	84.11
balanced	130,000	10,847,108	83.44
normal	130,000	10,719,878	82.46
	OneStopEnglish		
beginner	189	100,800	533.33
intermediate	189	127,934	676.90
advanced	189	155,253	820.49
All	567	383,987	677.23
	WeeBit		
age 7–8	600	77,613	129.35
age 8–9	600	100,491	167.49
age 9–10	600	159,719	266.20
age 10–14	600	89,548	149.25
age 14–16	600	152,402	254.00
All	3,000	579,773	193.26
	Newsela		
2nd grade	224	74,428	332.27
3rd grade	500	197,992	395.98
4th grade	1,569	923,828	588.80
5th grade	1,342	912,411	679.89
6th grade	1,058	802,057	758.09
7th grade	1,210	979,471	809.48
8th grade	1,037	890,358	858.59
9th grade	750	637,784	850.38
10th grade	20	19,012	950.60
11th grade	2	1,130	565.00
12th	1,853	1,833,781	989.63
All	9,565	7,272,252	760.30
	KRES-balanced		
balanced	/	2,402,263	/
	Slovenian SB		
1st-ps	69	12,921	187.26
2nd-ps	146	30,296	207.51
3rd-ps	268	62,241	232.24
4th-ps	1,007	265,242	263.40
5th-ps	1,186	330,039	278.28
6th-ps	959	279,461	291.41
7th-ps	1,470	462,551	314.66
8th-ps	1,844	540,944	293.35
9th-ps	2,154	688,149	319.47
1st-hs	1,663	578,694	347.98
2nd-hs	590	206,147	349.40
3rd-hs	529	165,845	313.51
4th-hs	45	14,313	318.07
All	11,930	3,636,843	304.85

Language models are trained on large corpora of texts. For this purpose, we used the following corpora.

- **Corpus of English Wikipedia** and **Corpus of Simple Wikipedia** articles: We created three corpora for the use in our unsupervised English experiments:[8]
  - **Wiki-normal** contains 130,000 randomly selected articles from the Wikipedia dump, which comprise 489,976 sentences and 10,719,878 tokens.
  - **Wiki-simple** contains 130,000 randomly selected articles from the Simple Wikipedia dump, which comprise 654,593 sentences and 10,933,710 tokens.
  - **Wiki-balanced** contains 65,000 randomly selected articles from the Wikipedia dump (dated 26 January 2018) and 65,000 randomly selected articles from the Simple Wikipedia dump. Altogether the corpus comprises 571,964 sentences and 10,847,108 tokens.
- **K**RES-balanced: The KRES corpus (Logar et al. 2012) is a 100 million word balanced reference corpus of Slovenian language: 35% of its content is books, 40% periodicals, and 20% Internet texts. From this corpus we took all the available documents from two children's magazines (Ciciban and Cicido), all documents from four teenager magazines (Cool, Frka, PIL plus, and Smrklja), and documents from three magazines targeting adult audiences (Življenje in tehnika, Radar, City magazine). With these texts, we built a corpus with approximately 2.4 million words. The corpus is balanced in a sense that about one-third of the sentences come from documents targeting children, one-third is targeting teenagers, and the last third is targeting adults.

3.2 Data Set Analysis

Overall, there are several differences between our data sets:

- **Language:** As already mentioned, we have three English (Newsela, OneStopEnglish and WeeBit), and one Slovenian (Slovenian SB) test data set.
- **Parallel corpora vs. unaligned corpora:** Newsela and OneStopEnglish data sets are parallel corpora, which means that articles from different readability classes are semantically similar to each other. On the other hand, WeeBit and Slovenian SB data sets contain completely different articles in each readability class. Although this might not affect traditional readability measures, which do not take semantic information into account, it might prove substantial for the performance of classifiers and the proposed language model-based readability measures.

8 English Wikipedia and Simple Wikipedia dumps from 26 January 2018 were used for the corpus construction.

- **Length of documents:** Another difference between Newsela and OneStopEnglish data sets on one side, and WeeBit and Slovenian SB data set on the other, is the length of data set documents. Newsela and OneStopEnglish data sets contain longer documents, on average about 760 and 677 words long, and documents in the WeeBit and Slovenian SB corpora are on average about 193 and 305 words long, respectively.
- **Genre:** OneStopEnglish and Newsela data sets contain news articles, WeeBit is made of educational articles, and the Slovenian SB data set is composed of school books. For training of the English language models, we use Wikipedia and Simple Wikipedia, which contain encyclopedia articles, and for Slovene language model training, we use the KRES-balanced corpus, which contains magazine articles.
- **Target audience:** OneStopEnglish is the only test data set that specifically targets adult ESL learners and not children, as do other test data sets. When it comes to data sets used for language model training, KRES-balanced corpus is made of articles that target both adults and children. The problem with Wikipedia and Simple Wikipedia is that no specific target audience is addressed because articles are written by volunteers. In fact, using Simple Wikipedia as a data set for the training of simplification algorithms has been criticized in the past because of its lack of specific simplification guidelines, which are based only on the declarative statement that Simple Wikipedia was created for "children and adults who are learning the English language" (Xu, Callison-Burch, and Napoles 2015). This lack of guidelines also contributes to the decrease in the quality of simplification according to Xu, Callison-Burch, and Napoles (2015), who found that the corpus can be noisy and that half of its sentences are not actual simplifications but rather copied from the original Wikipedia.

This diversity of the data sets limits ambitions of the study to offer general conclusions true across genres, languages, or data sets. On the other hand, it offers an opportunity to determine how the specifics of each data set affect each of the proposed readability predictors and also to determine the overall robustness of the applied methods.

Although many aspects differ from one data set to another, there are also some common characteristics across all the data sets, which allow using the same prediction methods on all of them. These are mostly connected to the common techniques used in the construction of the readability data sets, no matter the language, genre, or target audience of the specific data set. The creation of parallel simplification corpora (i.e., Newsela, OneStopEnglish, and Simple Wikipedia) generally involves three techniques, splitting (breaking a long sentence into shorter ones), deletion (removing unimportant parts of a sentence), and paraphrasing (rewriting a text into a simpler version via reordering, substitution, and occasionally expansion) (Feng 2008). Even though there might be some subtleties involved (because what constitutes simplification for one type of user may not be appropriate for another), how these techniques are applied is rather general. Also, although there is no simplification used in the non-parallel corpora (WeeBit, Slovenian SB), the contributing authors were nevertheless instructed to write the text for a specific target group and adapt the writing style accordingly. In most cases, this leads to the same result (e.g., shorter, less complex sentences and simpler vocabulary used in texts intended for younger or less fluently speaking audiences).

The claim of commonality between data sets can be backed up by the fact that even traditional readability indicators correlate quite well with human assigned readability, no matter the specific genre, language, or purpose of each data set. Results in Table 2 demonstrate this point by showcasing readability scores of traditional readability formulas from Section 2.1.1. We can see that the general pattern of increased difficulty on all data sets and for all indicators—larger readability scores (or in the case of FRE, smaller) are assigned to those classes of the data set that contain texts written for older children or more advanced ESL learners. This suggests that multi-data set, multi-genre, and even multilingual readability prediction is feasible on the set of chosen data sets, even if only the shallow traditional readability indicators are used.

However, the results do indicate that cross-genre or even cross-data set readability prediction might be problematic because the data sets do not cover the same readability range according to the shallow prediction formulas (and also ground truth readability labels). For example, documents in the WeeBit 14–16 age group have scores very similar to the Newsela 6th grade documents, which means that a classifier trained on the WeeBit corpus might have a hard time classifying documents belonging to higher Newsela grades since the readability of these documents is lower than for the most complex documents in the WeeBit corpus according to all of the shallow readability indicators. For this reason, we opted not to perform any supervised cross-data set or cross-genre experiments. Nevertheless, the problem of cross-genre prediction is important in the context of the proposed unsupervised experiments, because the genre discrepancy between the data sets used for training the language models and the data sets on which the models are used might influence the performance of the proposed language model-based measures. A more detailed discussion on this topic is presented in Section 4.2.

The analysis in Table 2 also confirms the findings by Madrazo Azpiazu and Pera (2020), who have shown that crosslingual readability prediction with shallow readability indicators is problematic. For example, if we compare the Newsela corpus and Slovenian SB corpus, which both cover roughly the same age group, we can see that for some readability indicators (FRE, FKGL, DCRF, and ASL) the values are on entirely different scales.

4. Unsupervised Neural Approach

In this section, we explore how neural language models can be used for determining the readability of the text in an unsupervised way. In Section 4.1, we present the neural architectures used in our experiments; in Section 4.2, we describe the methodology of the proposed approach; and in Section 4.3, we present the conducted experiments.

4.1 Neural Language Model Architectures

Mikolov et al. (2011) have shown that neural language models outperform n-gram language models by a high margin on large and also relatively small (less than 1 million tokens) data sets. The achieved differences in perplexity (see Equation (1)) are attributed to a richer historical contextual information available to neural networks, which are not limited to a small contextual window (usually of up to 5 previous words) as is the case of n-gram language models. In Section 2.1.5, we mentioned some approaches that use

Table 2
Scores of traditional readability indicators from Section 2.1.1 for specific classes in the readability data sets.

Class	GFI	FRE	FKGL	ARI	DCRF	SMOG	ASL
			Wikipedia				
simple	11.80	62.20	8.27	14.08	11.40	11.40	16.90
balanced	13.49	56.17	9.70	15.86	12.53	12.53	19.54
normal	15.53	49.16	11.47	18.06	13.89	13.89	23.10
			WeeBit				
age 7–8	6.91	83.41	3.82	8.83	7.83	7.83	10.23
age 8–9	8.45	76.68	5.34	10.33	8.87	8.87	12.89
age 9–10	10.30	69.88	6.93	12.29	10.01	10.01	15.69
age 10–14	9.94	75.35	6.34	11.20	9.67	9.67	16.64
age 14–16	11.76	66.61	8.09	13.56	10.81	10.81	18.86
			OneStopEnglish				
beginner	11.79	66.69	8.48	13.93	11.05	11.05	20.74
intermediate	13.83	59.68	10.19	15.98	12.30	12.30	23.98
advanced	15.35	54.84	11.54	17.65	13.22	13.22	26.90
			Newsela				
2nd grade	6.11	85.69	3.27	8.09	7.26	7.26	9.26
3rd grade	7.24	80.92	4.27	9.30	7.94	7.94	10.72
4th grade	8.58	76.05	5.40	10.50	8.88	8.88	12.72
5th grade	9.79	71.76	6.47	11.73	9.68	9.68	14.81
6th grade	11.00	67.46	7.53	12.99	10.47	10.47	16.92
7th grade	12.11	62.71	8.54	14.12	11.26	11.26	18.46
8th grade	13.05	60.37	9.38	15.19	11.83	11.83	20.81
9th grade	14.20	55.00	10.46	16.37	12.70	12.70	22.17
10th grade	14.15	55.70	10.60	16.50	12.83	12.83	23.33
11th grade	15.70	56.41	11.05	16.96	12.77	12.77	24.75
12th grade	14.52	55.58	10.71	16.70	12.79	12.79	23.69
			KRES-balanced				
balanced	12.72	29.20	12.43	14.88	14.08	14.08	15.81
			Slovenian SB				
1st-ps	9.54	31.70	10.38	11.72	11.12	11.12	7.63
2nd-ps	9.49	34.90	10.11	11.34	11.26	11.26	8.37
3rd-ps	10.02	32.89	10.61	11.78	11.80	11.80	9.31
4th-ps	10.96	30.29	11.18	12.84	12.39	12.39	10.40
5th-ps	11.49	28.13	11.62	13.33	12.79	12.79	11.02
6th-ps	13.20	20.10	12.84	14.57	13.61	13.61	11.45
7th-ps	12.94	22.97	12.61	14.52	13.64	13.64	12.24
8th-ps	13.48	18.12	13.09	14.78	13.71	13.71	11.32
9th-ps	13.69	19.26	13.13	15.07	13.94	13.94	12.27
1st-hs	15.12	12.66	14.33	16.22	14.96	14.96	13.62
2nd-hs	15.13	15.13	13.90	15.83	14.67	14.67	13.49
3rd-hs	14.76	13.09	14.00	15.62	14.44	14.44	12.57
4th-hs	14.66	14.39	13.64	15.54	14.03	14.03	11.62

n-gram language models for readability prediction. However, we are unaware of any approach that would use deep neural network language models for determining the readability of a text.

In this research, we utilize three neural architectures for language modeling. First are RNNs, which are suitable for modeling sequential data. At each time step t, the input vector x_t, and hidden state vector h_{t-1} are fed into the network, producing the next hidden vector state h_t with the following recursive equation:

$$h_t = f(Wx_t + Uh_{t-1} + b)$$

where f is a nonlinear activation function, W and U are matrices representing weights of the input layer and hidden layer, and b is the bias vector. Learning long-range input dependencies with plain RNNs is problematic because of vanishing gradients (Bengio, Simard, and Frasconi 1994); therefore, in practice, modified recurrent networks, such as Long Short-Term Memory networks (LSTMs) are used. In our experiments, we use the LSTM-based language model proposed by Kim et al. (2016). This architecture is adapted to language modeling of morphologically rich languages, such as Slovenian, by utilizing an additional character-level convolutional neural network (CNN). The convolutional level learns a character structure of words and is connected to the LSTM-based layer, which produces predictions at the word level.

Bai, Kolter, and Koltun (2018) introduced a new sequence modeling architecture based on convolution, called temporal convolutional network (TCN), which is also used in our experiments. TCN uses causal convolution operations, which make sure that there is no information leakage from future time steps to the past. This and the fact that TCN takes a sequence as an input and maps it into an output sequence of the same size makes this architecture appropriate for language modeling. TCNs are capable of leveraging long contexts by using a very deep network architecture and a hierarchy of dilated convolutions. A single dilated convolution operation F on element s of the 1-dimensional sequence x can be defined with the following equation:

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d\cdot i}$$

where $f : 0, \ldots k-1$ is a filter of size k, d is a dilation factor, and $s - d \cdot i$ accounts for the direction of the past. In this way, the context taken into account during the prediction can be increased by using larger filter sizes and by increasing the dilation factor. The most common practice is to increase the dilation factor exponentially with the depth of the network.

Recently, Devlin et al. (2019) proposed a novel approach to language modeling. Their BERT uses both left and right context, which means that a word w_t in a sequence is not determined just from its left sequence $w_{1:t-1} = [w_1, \ldots, w_{t-1}]$ but also from its right word sequence $w_{t+1:n} = [w_{t+1}, \ldots, w_{t+n}]$. This approach introduces a new learning objective, a *masked language model*, where a predefined percentage of randomly chosen words from the input word sequence is masked, and the objective is to predict these masked words from the unmasked context. BERT uses a transformer neural network architecture (Vaswani et al. 2017), which relies on the self-attention mechanism. The distinguishing feature of this approach is the use of several parallel attention layers, the so-called attention heads, which reduce the computational cost and allow the system to attend to several dependencies at once.

All types of neural network language models, TCN, LSTM, and BERT, output softmax probability distribution calculated over the entire vocabulary, and present the probabilities for each word given its historical (and in the case of BERT also future) sequence. Training of these networks usually minimizes the negative log-likelihood (NLL) of the training corpus word sequence $w_{1:n} = [w_1, \dots, w_n]$ by backpropagation through time:

$$\text{NLL} = -\sum_{i=1}^{n} \log P(w_i | w_{1:i-1}) \tag{2}$$

In the case of BERT, the formula for minimizing NLL also uses the right-hand word sequence:

$$\text{NLL} = -\sum_{i=1}^{n} \log P(w_i | w_{1:i-1}, w_{i+1:n})$$

where w_i are the *masked words*.

The following equation, which is used for measuring the perplexity of neural language models, defines the relationship between perplexity (PPL, see Equation (1)) and NLL (Equation (2)):

$$\text{PPL} = e^{(\frac{\text{NLL}}{N})}$$

4.2 Unsupervised Methodology

Two main questions we wish to investigate in the unsupervised approach are the following:

- Can standalone neural language models be used for unsupervised readability prediction?
- Can we develop a robust new readability formula that will outperform traditional readability formulas by relying not only on shallow lexical sophistication indicators but also on neural language model statistics?

4.2.1 Language Models for Unsupervised Readability Assessment. The findings of the related research suggest that a separate language model should be trained for each readability class in order to extract features for successful readability prediction (Petersen and Ostendorf 2009; Xia, Kochmar, and Briscoe 2016). On the other hand, we test the possibility of using a neural language model as a standalone unsupervised readability predictor.

Two points that support this kind of usage are based on the fact that neural language models tend to capture much more information compared to the traditional *n*-gram models. First, because *n*-gram language models used in the previous work on readability detection were in most cases limited to a small contextual window of up to five words, their learning potential was limited to lexico-semantic information (e.g., information about the difficulty of vocabulary and word *n*-gram structures in the text), and information about the text syntax. We argue that due to much larger contextual information of the neural models (e.g., BERT leverages sequences of up to 512 byte-pair tokens), which spans across sentences, the neural language models also learn high-level textual properties, such as long-distance dependencies (Jawahar, Sagot, and Seddah 2019), in order to minimize NLL during training. Second, *n*-gram models in past readability research have only been trained on the corpora (or, more specifically, on parts of the corpora) on which they were later used. In contrast, by training the neural models on large general corpora, the model also learns semantic information, which can be transferred when the model is used on a smaller test corpus. The success of this knowledge transfer is, to some extent, dependent on the genre compatibility of the train and test corpora.

A third point favoring greater flexibility of neural language models relies on the fact that no corpus is a monolithic block of text made out of units (i.e., sentences, paragraphs, and articles) of exactly the same readability level. This means that a language model trained on a large corpus will be exposed to chunks of text with different levels of complexity. We hypothesize that, due to this fact, the model will to some extent be able to distinguish between these levels and return a lower perplexity for more standard, predictable (i.e., readable) text. Vice versa, complex and rare language structures and vocabulary of less readable texts would negatively affect the performance of the language model, expressed via larger perplexity score. If this hypothesis is correct, then ideally, the average readability of the training corpus should fit somewhere in the middle of the readability spectrum of the testing corpus.

To test these statements, we train language models on Wiki-normal, Wiki-simple, and Wiki-balanced corpora described in Section 3. All three Wiki corpora contain roughly the same amount of text, in order to make sure that the training set size does not influence the results of the experiments. We expect the following results:

- **Hypothesis 1:** Training the language models on a corpus with a readability that fits somewhere in the middle of the readability spectrum of the testing corpus will yield the best correlation between the language model's performance and readability. According to the preliminary analysis of our corpora conducted in Section 3.2 and results of the analysis in Table 2, this ideal scenario can be achieved in three cases: (i) if a language model trained on the Wiki-simple is used on the Newsela corpora, (ii) if a language model trained on the Wiki-balanced corpus is used on the OneStopEnglish corpus, and (iii) if the model trained on the KRES-balanced corpus is used on the Slovenian SB corpus, despite the mismatch of genres in these corpora.
- **Hypothesis 2:** The language models trained only on texts for adults (Wiki-normal) will show higher perplexity on texts for children (WeeBit and Newsela) because their training set did not contain such texts; this will negatively affect the correlation between the language model's performance and readability.
- **Hypothesis 3:** Training the language models only on texts for children (Wiki-simple corpus) will result in a higher perplexity score of the language model when applied to adult texts (OneStopEnglish). This will positively affect the correlation between the language model's performance and readability. However, this language model will not be able to reliably distinguish between texts for different levels of adult ESL learners, which will have a negative effect on the correlation.

To further test the viability of the unsupervised language models as readability predictors and to test the limits of using a single language model, we also explore the possibility of using a language model trained on a large general corpus. The English BERT language model was trained on large corpora (Google Books Corpus [Goldberg and Orwant 2013] and Wikipedia) of about 3,300M words containing mostly texts for adult English speakers. According to hypothesis 2, this will have a negative effect on the correlation between the performance of the model and readability.

Because of the large size of the BERT model and its huge training corpus, the semantic information acquired during training is much larger than the information acquired by the models we train on our much smaller corpora, which means that there is a greater possibility that the BERT model was trained on some text semantically similar to the content in the test corpora and that this information can be successfully transferred. However, the question remains, exactly what type of semantic content does the BERT's training corpus contain? One hypothesis is that its training corpus contains more content specific for adult audiences and less content found in the corpora for children. This would have a negative effect on the correlation between the performance of the model and readability on the WeeBit corpus. Contrarily, because the two highest readability classes in the WeeBit corpus contain articles from different scientific fields used for the education of high school students, which can contain rather specific and technical content that is unlikely to be common in the general training corpus, this might influence a positive correlation between the performance of the model and readability. Newsela and OneStopEnglish, on the other hand, are parallel corpora, which means that the semantic content in all classes is very similar; therefore the success or failure of semantic transfer will most likely not affect these two corpora.

4.2.2 Ranked Sentence Readability Score. Based on the two considerations below, we propose a new Ranked Sentence Readability Score (RSRS) for measuring the readability with language models.

- The shallow lexical sophistication indicators, such as the length of a sentence, correlate well with the readability of a text. Using them besides statistics derived from language models could improve the unsupervised readability prediction.
- The perplexity score used for measuring the performance of a language model is an *unweighted* sum of perplexities of words in the predicted sequence. In reality, a small number of unreadable words might drastically reduce the readability of the entire text. Assigning larger weights to such words might improve the correlation of language model scores with the readability.

The proposed readability score is calculated with the following procedure. First, a given text is split into sentences with the default sentence tokenizer from the NLTK library (Bird and Loper 2004). In order to obtain a readability estimation for each word in a specific context, we compute, for each word in the sentence, the word negative log-likelihood (WNLL) according to the following formula:

$$\text{WNLL} = -(y_t \log y_p + (1 - y_t) \log (1 - y_p))$$

where y_p denotes the probability (from the softmax distribution) predicted by the language model according to the historical sequence, and y_t denotes the empirical distribution for a specific position in the sentence, that is, y_t has the value 1 for the word in the vocabulary that actually appears next in the sequence and the value 0 for all the other words in the vocabulary. Next, we sort all the words in the sentence in ascending order according to their WNLL score, and the ranked sentence readability score (RSRS) is calculated with the following expression:

$$\mathrm{RSRS} = \frac{\sum_{i=1}^{S} \sqrt{i} \cdot \mathrm{WNLL}(i)}{S} \tag{3}$$

where S denotes the sentence length and i represents the rank of a word in a sentence according to its WNLL value. The square root of the word rank is used for proportionally weighting words according to their readability because initial experiments suggested that the use of a square root of a rank represents the best balance between allowing all words to contribute equally to the overall readability of the sentence and allowing only the least readable words to affect the overall readability of the sentence. For out-of-vocabulary words, square root rank weights are doubled, because these rare words are, in our opinion, good indicators of non-standard text. Finally, in order to obtain the readability score for the entire text, we calculate the average of all the RSRS scores in the text. An example of how RSRS is calculated for a specific sentence is shown in Figure 1.

The main idea behind the RSRS score is to avoid the reductionism of traditional readability formulas. We aim to achieve this by including high-level structural and semantic information through neural language model–based statistics. The first assumption is that complex grammatical and lexical structures harm the performance of the language model. Since WNLL score, which we compute for each word, depends on the context in which the word appears in, words appearing in more complex grammatical and lexical contexts will have a higher WNLL. The second assumption is that the semantic information is included in the readability calculation: Tested documents with semantics dissimilar to the documents in the language model training set will negatively affect the performance of the language model, resulting in the higher WNLL score for words with unknown semantics. The trainable nature of language models allows for customization and personalization of the RSRS for specific tasks,

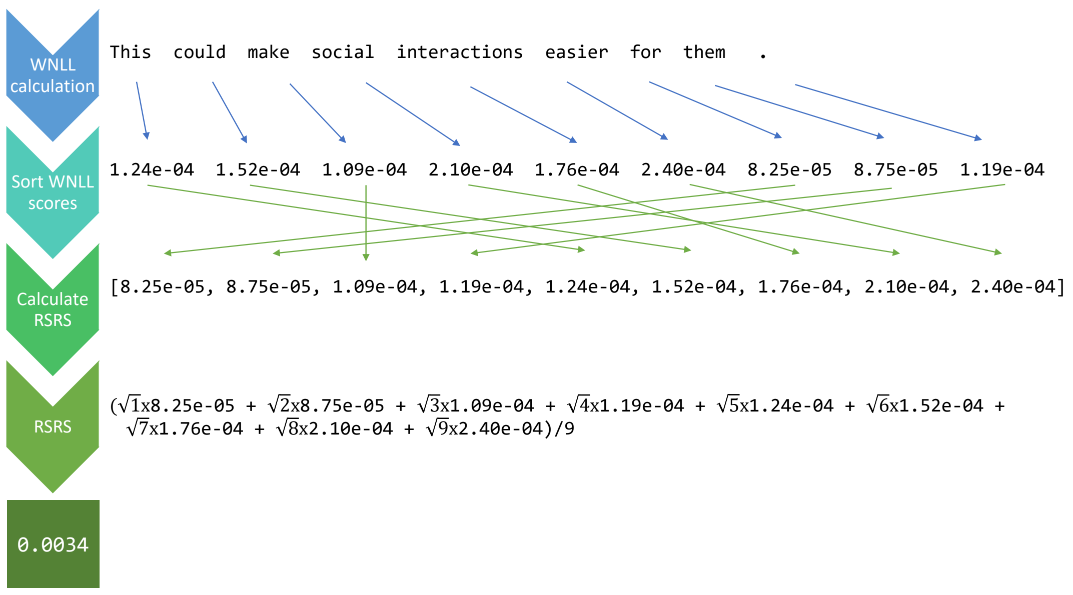

WNLL calculation
Sort WNLL scores
Calculate RSRS
RSRS
0.0034
This could make social interactions easier for them .
1.24e-04 1.52e-04 1.09e-04 2.10e-04 1.76e-04 2.40e-04 8.25e-05 8.75e-05 1.19e-04
[8.25e-05, 8.75e-05, 1.09e-04, 1.19e-04, 1.24e-04, 1.52e-04, 1.76e-04, 2.10e-04, 2.40e-04]
(√1x8.25e-05 + √2x8.75e-05 + √3x1.09e-04 + √4x1.19e-04 + √5x1.24e-04 + √6x1.52e-04 + √7x1.76e-04 + √8x2.10e-04 + √9x2.40e-04)/9

Figure 1
The RSRS calculation for the sentence *This could make social interactions easier for them.*

topics, and languages. This means that RSRS will alleviate the problem of cultural non-transferability of traditional readability formulas.

On the other hand, the RSRS also leverages shallow lexical sophistication indicators through the index weighting scheme, which ensures that less readable words contribute more to the overall readability score. This is somewhat similar to the counts of long and difficult words in the traditional readability formulas, such as GFI and DCRF. The value of RSRS also increases for texts containing longer sentences, since the square roots of the word rank weights become larger with increased sentence length. This is similar to the behavior of traditional formulas such as GFI, FRE, FKGL, ARI, and DCRF, where this effect is achieved by incorporating the ratio between the total number of words and the total number of sentences into the equation.

4.3 Unsupervised Experiments

For the presented unsupervised readability assessment methodology based on neural language models, we first present the experimental design followed by the results.

4.3.1 Experimental Design. Three different architectures of language models (described in Section 4.1) are used for experiments: a temporal convolutional network (TCN) proposed by Bai, Kolter, and Koltun (2018), a recurrent language model (RLM) using character-level CNN and LSTM proposed by Kim et al. (2016), and an attention-based language model, BERT (Devlin et al. 2019). For the experiments on the English language, we train TCN and RLM on three Wiki corpora.

To explore the possibility of using a language model trained on a general corpus for the unsupervised readability prediction, we use the BERT-base-uncased English language model, a pretrained uncased language model trained on BooksCorpus (0.8G words) (Zhu et al. 2015) and English Wikipedia (2.5G words). For the experiments on Slovenian, the corpus containing just school books is too small for efficient training of language models; therefore TCN and RLM were only trained on the KRES-balanced corpus described in Section 3. For exploring the possibility of using a general language model for the unsupervised readability prediction, a pretrained CroSloEngual BERT model trained on corpora from three languages, Slovenian (1.26G words), Croatian (1.95G words), and English (2.69G words) (Ulčar and Robnik-Šikonja 2020), is used. The corpora used in training the model are a mix of news articles and a general Web crawl.

The performance of language models is typically measured with the perplexity (see Equation (1)). To answer the research question of whether standalone language models can be used for unsupervised readability prediction, we investigate how the measured perplexity of language models correlates with the readability labels in the gold-standard WeeBit, OneStopEnglish, Newsela, and Slovenian SB corpora described in Section 3. The correlation to these ground truth readability labels is also used to evaluate the performance of the RSRS measure. For performance comparison, we calculate the traditional readability formula values (described in Section 2) for each document in the gold-standard corpora and measure the correlation between these values and manually assigned labels. As a baseline, we use the average sentence length (ASL) in each document.

The correlation is measured with the Pearson correlation coefficient (ρ). Given a pair of distributions X and Y, the covariance cov, and the standard deviation σ, the formula for ρ is:

$$\rho_{x,y} = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

A larger positive correlation signifies a better performance for all measures except the FRE readability measure. As this formula assigns higher scores to better-readable texts, a larger negative correlation suggests a better performance of the FRE measure.

4.3.2 Experimental Results. The results of the experiments are presented in Table 3. The ranking of measures on English and Slovenian data sets are presented in Table 4.

The correlation coefficients of all measures vary drastically between different corpora. The highest ρ values are obtained on the Newsela corpus, where the best performing measure (surprisingly this is our baseline—the average sentence length) achieves the ρ of 0.906. The highest ρ on the other two English corpora are much lower. On the WeeBit corpus, the best performance is achieved by GFI and FKGL measures (ρ of 0.544), and on the OneStopEnglish corpus, the best performance is achieved with the proposed TCN RSRS-simple (ρ of 0.615). On the Slovenian SB, the ρ values are higher, and the best performing measure is TCN RSRS score-balanced with ρ of 0.789.

The perplexity-based measures show a much lower correlation with the ground truth readability scores. Overall, they perform the worst of all the measures for both languages (see Table 4), but we can observe large differences in their performance across different corpora. Although there is either no correlation or low negative correlation between perplexities of all three language models and readability on the WeeBit corpus, there is some correlation between perplexities achieved by RLM and TCN on OneStopEnglish and Newsela corpora (the highest being the ρ of 0.566 achieved by TCN perplexity-simple on the Newsela corpus). The correlation between RLM and

Table 3
Pearson correlation coefficients between manually assigned readability labels and the readability scores assigned by different readability measures in the unsupervised setting. The highest correlation for each corpus is marked with bold typeface.

Measure/Data set	WeeBit	OneStopEnglish	Newsela	Slovenian SB
RLM perplexity-balanced	−0.082	0.405	0.512	0.303
RLM perplexity-simple	−0.115	0.420	0.470	/
RLM perplexity-normal	−0.127	0.283	0.341	/
TCN perplexity-balanced	0.034	0.476	0.537	0.173
TCN perplexity-simple	0.025	0.518	0.566	/
TCN perplexity-normal	−0.015	0.303	0.250	/
BERT perplexity	−0.123	−0.162	−0.673	−0.563
RLM RSRS-balanced	0.497	0.551	0.890	0.732
RLM RSRS-simple	0.506	0.569	0.893	/
RLM RSRS-normal	0.490	0.536	0.886	/
TCN RSRS-balanced	0.393	0.601	0.894	**0.789**
TCN RSRS-simple	0.385	**0.615**	0.894	/
TCN RSRS-normal	0.348	0.582	0.886	/
BERT RSRS	0.279	0.384	0.674	0.126
GFI	**0.544**	0.550	0.849	0.730
FRE	−0.433	−0.485	−0.775	−0.614
FKGL	**0.544**	0.533	0.865	0.697
ARI	0.488	0.520	0.875	0.658
DCRF	0.420	0.496	0.735	0.686
SMOG	0.456	0.498	0.813	0.770
ASL	0.508	0.498	**0.906**	0.683

Table 4
Ranking (lower is better) of measures on English and Slovenian data sets sorted by the average rank on all data sets for which the measure is available.

Measure	WeeBit	OneStopEnglish	Newsela	Slovenian SB
RLM RSRS-simple	4	4	4	/
TCN RSRS-balanced	11	2	2	**1**
RLM RSRS-balanced	5	5	5	3
GFI	**1**	6	10	4
TCN RSRS-simple	12	**1**	3	/
ASL	3	12	**1**	7
FKGL	2	8	9	5
RLM RSRS-normal	6	7	6	/
TCN RSRS-normal	13	3	7	/
ARI	7	9	8	8
SMOG	8	11	11	2
DCRF	10	13	13	6
FRE	9	14	12	9
TCN perplexity-simple	16	10	15	/
TCN perplexity-balanced	15	15	16	11
BERT RSRS	14	18	14	12
RLM perplexity-balanced	18	17	17	10
RLM perplexity-simple	19	16	18	/
TCN perplexity-normal	17	19	20	/
BERT perplexity	20	21	21	13
RLM perplexity-normal	21	20	19	/

TCN perplexity measures and readability classes on the Slovenian SB corpus is low, with RLM perplexity-balanced showing the ρ of 0.303 and TCN perplexity-balanced achieving ρ of 0.173.

BERT perplexities are negatively correlated with readability, and the negative correlation is relatively strong on Newsela and Slovenian school books corpora (ρ of −0.673 and −0.563, respectively), and weak on WeeBit and OneStopEnglish corpora. As BERT was trained on corpora that are mostly aimed at adults, the strong negative correlation on Newsela and Slovenian SB corpora seem to suggest that BERT language models might actually be less perplexed by the articles aimed at adults than the documents aimed at younger audiences. This is supported by the fact that the negative correlation is weaker on the OneStopEnglish corpus, which is meant for adult audiences, and for which our analysis (see Section 3.2) has shown that it contains more complex texts according to the shallow readability indicators.

Nevertheless, the weak negative correlation on the WeeBit corpus is difficult to explain as one would expect a stronger negative correlation because the same analysis showed that WeeBit contains the least complex texts out of all the tested corpora. If this result is connected with the successful transfer of the semantic knowledge, it supports the hypothesis that the two classes containing the most complex texts in the WeeBit corpus contain articles with rather technical content that perplex the BERT model. However, the role of the semantic transfer should also dampen the negative correlation on the Slovenian SB, which is a non-parallel corpus and also contains rather technical educational content meant for high-school children. Perhaps the transfer is less successful for Slovenian since the Slovenian corpus on which the CroSloEngual BERT was trained is smaller than the English corpora used for training of English BERT.

Although further experiments and data are needed to pinpoint the exact causes for the discrepancies in the results, we can still conclude that using a single language model trained on general corpora for unsupervised readability prediction of texts for younger audiences or English learners is, at least according to our results, not a viable option.

Regarding our expectations that performance of the language model trained on a corpus with average readability that fits somewhere in the middle of the readability spectrum of the testing corpus would yield the best correlation with manually labeled readability scores, it is interesting to look at the differences in performance between TCN and RLM perplexity measures trained on Wiki-normal, Wiki-simple, and Wiki-balanced corpora. As expected, the correlation scores are worse on the WeeBit corpus, since all classes in this corpus contain texts that are less complex than texts in any of the training corpora. On the OneStopEnglish corpus, both Wiki-simple perplexity measures perform the best, which is unexpected, since we would expect the balanced measure to perform better. On the Newsela corpus, RLM perplexity-balanced outperforms RLM perplexity-simple by 0.042 (which is unexpected), and TCN perplexity-simple outperforms TCN perplexity-balanced by 0.029, which is according to the expectations. Also, according to the expectation is the fact, that both Wiki-normal perplexity measures are outperformed by a large margin by Wiki-simple and Wiki-balanced perplexity measures on the OneStopEnglish and the Newsela corpora. Similar observations can be made with regard to RSRS, which also leverages language model statistics. On all corpora, the performance of Wiki-simple RSRS measures and Wiki-balanced RSRS measures is comparable, and these measures consistently outperform Wiki-normal RSRS measures.

These results are not entirely compatible with hypothesis 1 in Section 4.2.1 that Wiki-balanced measures would be most correlated with readability on the OneStopEnglish corpus and that Wiki-simple measures would be most correlated with readability on the Newsela corpus. Nevertheless, training the language models on the corpora with readability in the middle of the readability spectrum of the test corpus seems to be an effective strategy, because the differences in performance between Wiki-balanced and Wiki-simple measures are not large. On the other hand, the good performance of the Wiki-simple measures supports our hypothesis 3 in Section 4.2.1, that training the language models on texts with the readability closer to the bottom of the readability spectrum of the test corpus for children will result in a higher perplexity score of the language model when applied to adult texts, which will have a positive effect on the correlation with readability.

The fact that positive correlation between readability and both Wiki-simple and Wiki-balanced perplexity measures on the Newsela and OneStopEnglish corpora is quite strong supports the hypothesis that more complex language structures and vocabularies of less readable texts would result in a higher perplexity on these texts. Interestingly, strong correlations also indicate that the genre discrepancies between the language model train and test sets do not appear to have a strong influence on the performance. Whereas the choice of a neural architecture for language modeling does not appear to be that crucial, the readability of the language model training set is of utmost importance. If the training set on average contains more complex texts than the majority of texts in the test set, as in the case of language models trained just on the Wiki-normal corpus (and also BERTs), the correlation between readability and perplexity disappears or even gets reverted, since language models trained on more complex language structures learn how to handle these difficulties.

The low performance of perplexity measures suggests that neural language model statistics are not good indicators of readability and should therefore not be used alone for readability prediction. Nevertheless, the results of TCN RSRS and RLM RSRS suggest that language models contain quite useful information when combined with other shallow lexical sophistication indicators, especially when readability analysis needs to be conducted on a variety of different data sets.

As seen in Table 4, shallow readability predictors can give inconsistent results on data sets from different genres and languages. For example, the simplest readability measure, the average sentence length, ranked first on Newsela and twelfth on OneStopEnglish. It also did not do well on the Slovenian SB corpus, where it ranked seventh. SMOG, on the other hand, ranked very well on the Slovenian SB corpus (rank 2) but ranked twice as eleventh and once as eighth on the English corpora. Among the traditional measures, GFI presents the best balance in performance and consistency, ranking first on WeeBit, sixth on OneStopEnglish, tenth on Newsela, and fourth on Slovenian SB.

On the other hand, RSRS-simple and RSRS-balanced measures offer more robust performance across data sets from different genres and languages according to ranks in Table 4. For example, the RLM RSRS-simple measure ranked fourth on all English corpora. The TCN RSRS-balanced measure, which was also used on Slovenian SB, ranked first on Slovenian SB and second on OneStopEnglish and Newsela. However, it did not do well on WeeBit, where the discrepancy in readability between the language model train and test sets was too large. RLM RSRS-balanced was more consistent, ranking fifth on all English corpora and third on Slovenian SB. These results suggest that language model statistics can improve the consistency of predictions on a variety of different data sets. The robustness of the measure is achieved by training the language model on a specific train set, with which one can optimize the RSRS measure for a specific task and language.

5. Supervised Neural Approach

As mentioned in Section 1, recent trends in text classification show the domination of deep learning approaches that internally use automatic feature construction. Existing neural approaches to readability prediction (see Section 2.3) tend to generalize better across data sets and genres (Filighera, Steuer, and Rensing 2019), and therefore solve the problem of classical machine learning approaches relying on an extensive feature engineering (Xia, Kochmar, and Briscoe 2016).

In this section, we analyze how different types of neural classifiers can predict text readability. In Section 5.1, we describe the methodology, and in Section 5.2 we present experimental scenarios and results of conducted experiments.

5.1 Supervised Methodology

We tested three distinct neural network approaches to text classification:

- Bidirectional long short-term memory network (BiLSTM). We use the RNN approach proposed by Conneau et al. (2017) for classification. The BiLSTM layer is a concatenation of forward and backward LSTM layers that read documents in two opposite directions. The max and mean pooling are applied to the LSTM output feature matrix in order to get the maximum and average values of the matrix. The resulting vectors are concatenated and fed to the final linear layer responsible for predictions.

- Hierarchical attention networks (HAN). We use the architecture of Yang et al. (2016) that takes hierarchical structure of text into account with the two-level attention mechanism (Bahdanau, Cho, and Bengio 2014; Xu et al. 2015) applied to word and sentence representations encoded by BiLSTMs.
- Transfer learning. We use the pretrained BERT transformer architecture with 12 layers of size 768 and 12 self-attention heads. A linear classification head was added on top of the pretrained language model, and the whole classification model was fine-tuned on every data set for three epochs. For English data sets, the BERT-base-uncased English language model is used, while for the Slovenian SB corpus, we use the CroSloEngual BERT model trained on Slovenian, Croatian, and English (Ulčar and Robnik-Šikonja 2020).[9]

We randomly shuffle all the corpora, and then Newsela and Slovenian SB corpora are split into a train (80% of the corpus), validation (10% of the corpus), and test (10% of the corpus) sets. Because of the small number of documents in OneStopEnglish and WeeBit corpora (see description in Section 3), we used five-fold stratified crossvalidation on these corpora to get more reliable results. For every fold, the corpora were split into the train (80% of the corpus), validation (10% of the corpus), and test (10% of the corpus) sets. We employ Scikit StratifiedKFold,[10] both for train-test splits and five-fold crossvalidation splits, in order to preserve the percentage of samples from each class.

BiLSTM and HAN classifiers were trained on the train set and tested on the validation set after every epoch (for a maximum of 100 epochs). The best performing model on the validation set was selected as the final model and produced predictions on the test sets. BERT models are fine-tuned on the train set for three epochs, and the resulting model is tested on the test set. The validation sets were used in a grid search to find the best hyperparameters of the models. For BiLSTM, all combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below:

- Batch size: **8**, 16, 32
- Learning rates: 0.00005, **0.0001**, 0.0002, 0.0004, 0.0008
- Word embedding size: 100, **200**, 400
- LSTM layer size: **128**, 256
- Number of LSTM layers: 1, **2**, 3, 4
- Dropout after every LSTM layer: 0.2, **0.3**, 0.4

For HAN, we tested all combinations of the following hyperparameter values (the best combination is written in bold):

- Batch size: 8, **16**, 32
- Learning rates: 0.00005, **0.0001**, 0.0002, 0.0004, 0.0008

9 Both models are available through the Transformers library `https://huggingface.co/transformers/`.
10 `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html`.,

- Word embedding size: 100, **200**, 400
- Sentence embedding size: **100**, 200, 400

For BERT fine-tuning, we use the default learning rate of 0.00002. The input sequence length is limited to 512 byte-pair tokens, which is the maximum supported input sequence length.

We used the same configuration for all the corpora and performed no corpus specific tweaking of classifier parameters. We measured the performance of all the classifiers in terms of accuracy (in order to compare their performance to the performance of the classifiers from the related work), weighted average precision, weighted average recall, and weighted average F_1-score.[11] Since readability classes are ordinal variables (in our case ranging from 0 to n = number of classes−1), not all mistakes of classifiers are equal; therefore we also utilize the Quadratic Weighted Kappa (QWK) measure, which allows for mispredictions to be weighted differently, according to the cost of a specific mistake. Calculation of the QWK involves three matrices containing observed scores, ground truth scores, and the weight matrix scores, which in our case correspond to the distance d between the classes c_i and c_j and is defined as $d = |c_i - c_j|$. QWK is therefore calculated as:

$$\text{QWK} = 1 - \frac{\sum_{i=1}^{c} \sum_{j=1}^{c} w_{ij} x_{ij}}{\sum_{i=1}^{c} \sum_{j=1}^{c} w_{ij} m_{ij}} \tag{4}$$

where c is the number of readability classes and w_{ij}, x_{ij}, and m_{ij} are elements in the weight, observed, and ground truth matrices, respectively.

5.2 Supervised Experimental Results

The results of supervised readability assessment using different architectures of deep neural networks are presented in Table 5, together with the state-of-the-art baseline results from the related work (Xia, Kochmar, and Briscoe 2016; Filighera, Steuer, and Rensing 2019; Deutsch, Jasbi, and Shieber 2020). We only present the best result reported by each of the baseline studies; the only exception is Deutsch, Jasbi, and Shieber (2020), for which we present two results, SVM-BF (SVM with BERT features) and SVM-HF (SVM with HAN features) that proved the best on the WeeBit and Newsela corpora, respectively.

On the WeeBit corpus, by far the best performance according to all measures was achieved by BERT. In terms of accuracy, BERT outperforms the second-best BiLSTM by about 8 percentage points, achieving the accuracy of 85.73%. HAN performs the worst on the WeeBit corpus according to all measures. BERT also outperforms the accuracy result reported by Xia, Kochmar, and Briscoe (2016), who used the five-fold crossvalidation setting and the accuracy result on the development set reported by Filighera, Steuer, and Rensing (2019).[12] In terms of weighted F_1-score, both strategies

11 We use the Scikit implementation of the metrics (`https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics`) and set the "average" parameter to "weighted."

12 For the study by Filighera, Steuer, and Rensing (2019), we report accuracy on the development set instead of accuracy on the test set, as the authors claim that this result is more comparable to the results achieved in the crossvalidation setting. On the test set, Filighera, Steuer, and Rensing (2019) report the best accuracy of 74.4%.

Table 5
The results of the supervised approach to readability in terms of accuracy, weighted precision, weighted recall, and weighted F_1-score for the three neural network classifiers and methods from the literature.

Measure/Data set	WeeBit	OneStopEnglish	Newsela	Slovenian SB
Filighera et al. (2019) accuracy	0.8130	–	–	–
Xia et al. (2016) accuracy	0.8030	–	–	–
SVM-BF (Deutsh et al., 2020) F_1	0.8381	–	0.7627	–
SVM-HF (Deutsh et al., 2020) F_1	–	–	0.8014	–
Vajjala et al. (2018) accuracy	–	0.7813	–	–
BERT accuracy	**0.8573**	0.6738	0.7573	0.4545
BERT precision	**0.8658**	0.7395	0.7510	0.4736
BERT recall	**0.8573**	0.6738	0.7573	0.4545
BERT F_1	**0.8581**	0.6772	0.7514	0.4157
BERT QWK	**0.9527**	0.7077	0.9789	**0.8855**
HAN accuracy	0.7520	**0.7872**	**0.8138**	0.4887
HAN precision	0.7534	**0.7977**	**0.8147**	0.4866
HAN recall	0.7520	**0.7872**	**0.8138**	0.4887
HAN F_1	0.7520	**0.7888**	**0.8101**	0.4847
HAN QWK	0.8860	**0.8245**	**0.9835**	0.8070
BiLSTM accuracy	0.7743	0.6875	0.7111	**0.5277**
BiLSTM precision	0.7802	0.7177	0.6910	**0.5239**
BiLSTM recall	0.7743	0.6875	0.7111	**0.5277**
BiLSTM F_1	0.7750	0.6920	0.6985	**0.5219**
BiLSTM QWK	0.9060	0.7230	0.9628	0.7980

that use BERT (utilizing the BERT classifier directly or feeding BERT features to the SVM classifier as in Deutsch, Jasbi, and Shieber [2020]) seem to return similar results. Finally, in terms of QWK, BERT achieves a very high score of 95.27% and the other two tested classifiers obtain a good QWK score close to 90%.

The best result on Newsela is achieved by HAN, achieving the F_1-score of 81.01% and accuracy of 81.38%. This is similar to the baseline SVM-HF result achieved by Deutsch, Jasbi, and Shieber (2020), who fed HAN features to the SVM classifier. BERT performs less competitively on the OneStopEnglish and Newsela corpora. On One-StopEnglish, it is outperformed by the best performing classifier (HAN) by about 10 percentage points, and on Newsela, it is outperformed by about 6 percentage points according to accuracy and F_1 criteria. The most likely reason for the bad performance of BERT on these two corpora is the length of documents in these two data sets. On average, documents in the OneStopEnglish and Newsela corpora are 677 and 760 words long. On the other hand, BERT only allows input documents of up to 512 byte-pair tokens, which means that documents longer than that need to be truncated. This results in the substantial loss of information on the OneStopEnglish and Newsela corpora but not on the WeeBit and Slovenian SB corpora, which contain shorter documents, 193 and 305 words long.

The results show that BiLSTM also has problems when dealing with longer texts, even though it does not require input truncation. This suggests that the loss of context is not the only reason for the non-competitive performance of BERT and BiLSTM, and that the key to the successful classification of long documents is the leveraging of hierarchical information in the documents, for which HAN was built for. The assumption is that this is particularly important in parallel corpora, where the simplified versions of the original texts contain the same message as the original texts, which forces the classifiers not to rely as much on semantic differences but rather focus on structural differences.

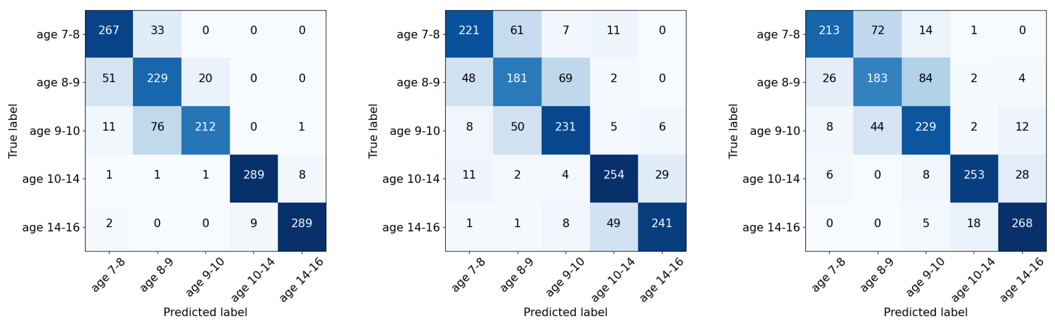
While F_1-scores and accuracies suggest large discrepancies in performance between HAN and two other classifiers on the OneStopEnglish and Newsela corpora, QWK scores draw a different picture. Although the discrepancy is still large on OneStopEnglish, all classifiers achieve almost perfect QWK scores on the Newsela data set. This suggests that even though BERT and BiLSTM make more classification mistakes than HAN, these mistakes are seldom costly on the Newsela corpus (i.e., documents are classified into neighboring classes of the correct readability class). QWK scores achieved on the Newsela corpus by all classifiers are also much higher than the scores achieved on other corpora (except for the QWK score achieved by BERT on the WeeBit corpus). This is in line with the results in the unsupervised setting, where the ρ values on the Newsela corpus were substantially larger than on other corpora.

The HAN classifier achieves the best performance on the OneStopEnglish corpus with an accuracy of 78.72% in the five-fold crossvalidation setting. This is comparable to the state-of-the-art accuracy of 78.13% achieved by Vajjala and Lučić (2018) with their SMO classifier using 155 hand-crafted features. BiLSTM and BERT classifiers perform similarly on this corpus, by about 10 percentage points worse than HAN, according to accuracy, F_1-score, and QWK.

The results on the Slovenian SB corpus are also interesting. In general, the performance of classifiers is the worst on this corpus, with the F_1-score of 52.19% achieved by BiLSTM being the best result. BiLSTM performs by about 4 percentage points better than HAN according to F_1-score and accuracy, while both classifiers achieve roughly the same QWK score of about 80%. On the other hand, BERT achieves lower F_1-score (about 45.45%) and accuracy (41.57%), but performs much better than the other two classifiers according to QWK, achieving QWK of almost 90%.

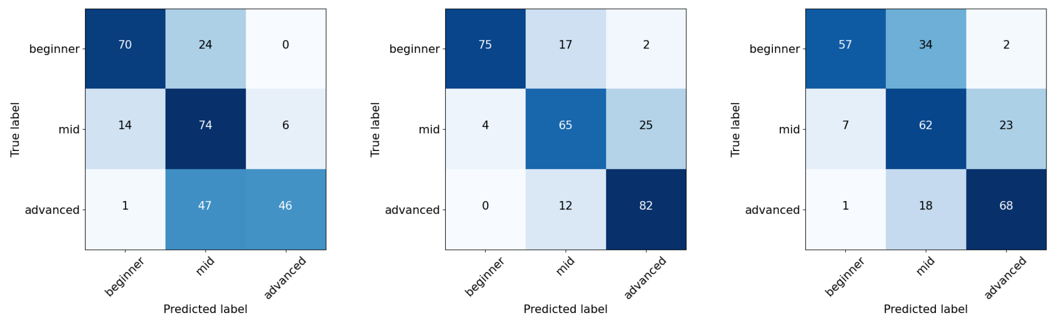
Confusion matrices for classifiers give us a better insight into what kind of mistakes are specific for different classifiers. For the WeeBit corpus, confusion matrices show (Figure 2) that all the tested classifiers have the most problems distinguishing between texts for children 8–9 years old and 9–10 years old. The mistakes where the text is falsely classified into an age group that is not neighboring the correct age group are rare. For example, the best performing BERT classifier misclassified only 16 documents into non-neighboring classes. When it comes to distinguishing between neighboring classes, the easiest distinction for the classifiers was the distinction between texts for children 9–10 years old and 10–14 years old. Besides fitting into two distinct age groups, the documents in these two classes also belong to two different sources (texts for children 9–10 years old consist of articles from WeeklyReader and texts for children 10–14 years old consist of articles from BBC-Bitesize), which suggests that the semantic and writing style dissimilarities between these two neighboring classes might be larger than for other neighboring classes, and that might have a positive effect on the performance of the classifiers.

On the OneStopEnglish corpus (Figure 3), the BERT classifier, which performs the worst on this corpus according to all criteria but precision, had the most problems correctly classifying documents from the advanced class, misclassifying about half of the documents. HAN had the most problems with distinguishing documents from the advanced and intermediate class, while the BiLSTM classifier classified a disproportionate amount of intermediate documents into the beginner class.



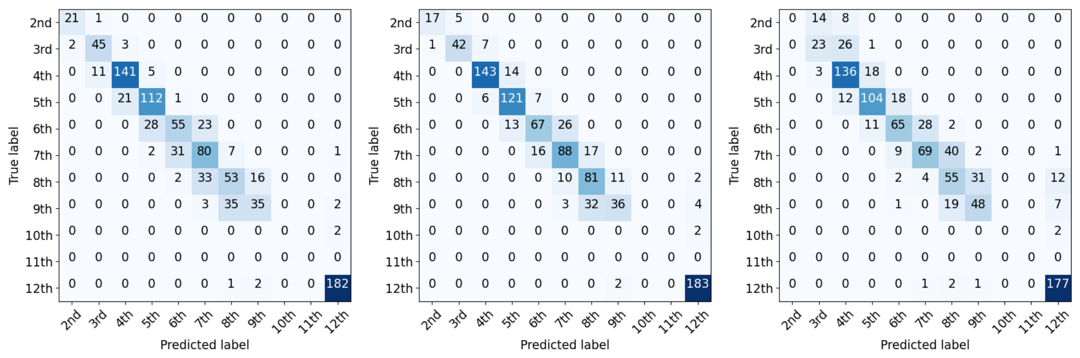
a) BERT b) HAN c) BiLSTM

Figure 2
Confusion matrices for BERT, HAN, and BiLSTM on the WeeBit corpus.



a) BERT b) HAN c) BiLSTM

Figure 3
Confusion matrices for BERT, HAN, and BiLSTM on the OneStopEnglish corpus.



a) BERT b) HAN c) BiLSTM

Figure 4
Confusion matrices for BERT, HAN, and BiLSTM on the Newsela corpus.

Confusion matrices of all classifiers for the Newsela corpus (Figure 4) follow a similar pattern. Unsurprisingly, no classifier predicted any documents to be in two minority classes (10th and 11th grade) with minimal training examples. As the QWK score has already shown, all classifiers classified a large majority of misclassified instances into

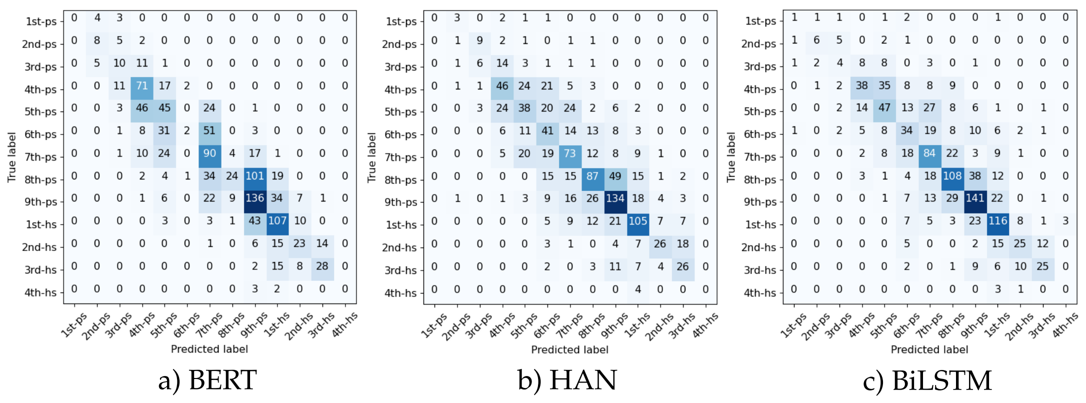


Figure 5
Confusion matrices for BERT, HAN, and BiLSTM on the Slovenian school books corpus.

neighboring classes, and costlier mistakes are rare. For example, the best performing HAN classifier altogether misclassified only 13 examples into non-neighboring classes.

Confusion matrices for the Slovenian SB corpus (Figure 5) are similar for all classifiers. The biggest spread of misclassified documents is visible for the classes in the middle of the readability range (from the 4th-grade of primary school to the 1st-grade of high school). The mistakes, which cause BERT to have lower F_1-score and accuracy scores than the other two classifiers, are most likely connected to the misclassification of all but two documents belonging to the school books for the 6th class of the primary school. Nevertheless, a large majority of these documents were misclassified into two neighboring classes, which explains the high QWK score achieved by the classifier. What negatively affected the QWK scores for HAN and BiLSTM is that the frequency of making costlier mistakes of classifying documents several grades above or below the correct grade is slightly higher for them than for BERT. Nevertheless, even though F_1-score results are relatively low on this data set for all classifiers (BiLSTM achieved the best F_1-score of 52.19%), the QWK scores around or above 80% and the confusion matrices clearly show that a large majority of misclassified examples were put into classes close to the correct one, suggesting that classification approaches to readability prediction can also be reliably used for Slovenian.

Overall, the classification results suggest that neural networks are a viable option for the supervised readability prediction. Some of the proposed neural approaches managed to outperform state-of-the-art machine learning classifiers that leverage feature engineering (Xia, Kochmar, and Briscoe 2016; Vajjala and Lučić 2018; Deutsch, Jasbi, and Shieber 2020) on all corpora where comparisons are available. However, the gains are not substantial, and the choice of an appropriate architecture depends on the properties of the specific data set.

6. Conclusion

We presented a set of novel unsupervised and supervised approaches for determining the readability of documents using deep neural networks. We tested them on several manually labeled English and Slovenian corpora. We argue that deep neural networks are a viable option both for supervised and unsupervised readability prediction and show that the suitability of a specific architecture for the readability task depends on the data set specifics.

We demonstrate that neural language models can be successfully used in the unsupervised setting, since they, in contrast to *n*-gram models, capture high-level textual properties and can successfully leverage rich semantic information obtained from the training data set. However, the results of this study suggest that unsupervised approaches to readability prediction that only take these properties of text into account cannot compete with the shallow lexical sophistication indicators. This is somewhat in line with the findings of the study by Todirascu et al. (2016), who also acknowledged the supremacy of shallow lexical indicators when compared with higher-level discourse features. Nevertheless, combining the components of both neural and traditional readability indicators into the new RSRS (ranked sentence readability score) measure does improve the correlation with human readability scores.

We argue that the RSRS measure is adaptable, robust, and transferable across languages. The results of the unsupervised experiments show the influence of the language model training set on the performance of the measure. While the results indicate that an exact match between the genres of the train and test sets is not necessary, the text complexity of a train set (i.e., its readability), should be in the lower or middle part of the readability spectrum of the test set for the optimal performance of the measure. This indicates that out of the two high-level text properties that the RSRS measure uses for determining readability, semantic information and long-distance structural information, the latter seems to have more effect on the performance. This is further confirmed by the results of using the general BERT language model for the readability prediction, which show a negative correlation between the language model perplexity and readability, even though the semantic information the model possesses is extensive due to the large training set.

The functioning of the proposed RSRS measure can be customized and influenced by choice of the training set. This is the desired property because it enables personalization and localization of the readability measure according to the educational needs, language, and topic. The usability of this feature might be limited for under-resourced languages because a sufficient amount of documents needed to train a language model that can be used for the task of readability prediction in a specific customized setting might not be available. On the other hand, our experiments on the Slovenian language show that a relatively small 2.4 million word training corpus for language models is sufficient to outperform traditional readability measures.

The results of the unsupervised approach to readability prediction on the corpus of Slovenian school books are not entirely consistent with the results reported by the previous Slovenian readability study (Škvorc et al. 2019), where the authors reported that simple indicators of readability, such as average sentence length, performed quite well. Our results show that the average sentence length performs very competitively on English but ranks badly on Slovenian. This inconsistency in results might be explained by the difference in corpora used for the evaluation of our approaches. Whereas Škvorc et al. (2019) conducted experiments on a corpus of magazines for different age groups (which we used for language model training), our experiments were conducted on a corpus of school books, which contains items for sixteen distinct school subjects with very different topics ranging from literature, music, and history to math, biology, and chemistry. As was already shown in Sheehan, Flor, and Napolitano (2013), the variance in genres and covered topics has an important effect on the ranking and performance of different readability measures. Further experiments on other Slovenian data sets are required to confirm this hypothesis.

In the supervised approach to determining readability, we show that the proposed neural classifiers can either outperform or at least compare with state-of-the-art approaches leveraging extensive feature engineering as well as previously used neural models on all corpora where comparison data is available. While the improved performance and elimination of work required for manual feature engineering are desirable, on the downside, neural approaches tend to decrease the interpretability and explainability of the readability prediction. Interpretability and explainability are especially important for educational applications (Sheehan et al. 2014; Madnani and Cahill 2018), where the users of such technology (educators, teachers, researchers, etc.) often need to understand what causes one text to be judged as more readable than the other and according to which dimensions. Therefore in the future, we will explore the possibilities of explaining the readability predictions of the proposed neural classifier with the help of general explanation techniques such as SHAP (Lundberg and Lee 2017), or the attention mechanism (Vaswani et al. 2017), which can be analyzed and visualized and can offer valuable insights into inner workings of the system.

Another issue worth discussing is the trade-off between performance gains we can achieve by employing computationally demanding neural networks on the one side and the elimination of work on the other. For example, on the OneStopEnglish corpus, we report the accuracy of 78.72% when HAN is used, while Vajjala and Lučić (2018) report an accuracy of 78.13% with their classifier employing 155 hand-crafted features. While it might be worth opting for a neural network in order to avoid extensive manual feature engineering, on the other hand, the same study by Vajjala and Lučić (2018) also reports that just by employing generic text classification features, 2–5 character *n*-grams, they obtained the accuracy of 77.25%. Considering this, one might argue that, depending on the use case, it might not be worth dedicating significantly more time, work, or computational resources for an improvement of slightly more than 1%, especially if this also decreases the overall interpretability of the prediction.

The performance of different classifiers varies across different corpora. The major factor proved to be the length of documents in the data sets. The HAN architecture, which tends to be well equipped to handle long-distance hierarchical text structures, performs the best on these data sets. On the other hand, in terms of QWK measure, BERT offers significantly better performance on data sets that contain shorter documents, such as WeeBit and Slovenian SB. As was already explained in Section 5.2, a large majority of OneStopEnglish and Newsela documents need to be truncated in order to satisfy the BERT's limitation of 512 byte-pair tokens. Although it is reasonable to assume that the truncation and the consequential loss of information do have a detrimental effect on the performance of the classifier, the extent of this effect is still unclear. The problem of truncation also raises the question of what is the minimum required length of a text for a reliable assessment of readability and if there exists a length threshold, above which having more text does not influence the performance of a classifier in a significant manner. We plan to assess this in future work thoroughly. Another related line of research we plan to pursue in the future is the use of novel algorithms, such as Longformer (Beltagy, Peters, and Cohan 2020) and Linformer (Wang et al. 2020), in which the attention mechanism scales linearly with the sequence length, making it feasible to process documents of thousands of tokens. We will check if applying these two algorithms on the readability data sets with longer documents can further improve the state of the art.

The other main difference between WeeBit and Slovenian SB data sets on the one hand, and Newsela and OneStopEnglish data sets on the other, is that they are not parallel corpora, which means that there can be substantial semantic differences between the readability classes in these two corpora. It seems that pretraining BERT as a language model allows for better exploitation of these differences, which leads to better performance. However, this reliance on semantic information might badly affect the performance of transfer learning based models on parallel corpora, since the semantic differences between classes in these corpora are much more subtle. We plan to assess the influence of available semantic information on the performance of different classification models in the future.

The differences in performance between classifiers on different corpora suggest that tested classifiers take different types of information into account. Provided that this hypothesis is correct, some gains in performance might be achieved if the classifiers are combined. We plan to test a neural ensemble approach for the task of predicting readability in the future.

While this study mostly focused on multilingual and multi-genre readability prediction, in the future, we also plan to test the cross-corpus, cross-genre, and cross-language transferability of the proposed supervised and unsupervised approaches. This requires new readability data sets for different languages and genres that are currently rare or not publicly available. On the other hand, this type of research will be capable of further determining the role of genre in the readability prediction and might open an opportunity to improve the proposed unsupervised readability score further.

Acknowledgments

The research was financially supported by the European Social Fund and Republic of Slovenia, Ministry of Education, Science, and Sport through project Quality of Slovene Textbooks (KaUČ). The work was also supported by the Slovenian Research Agency (ARRS) through core research programs P6-0411 and P2-0103, and the projects Terminology and Knowledge Frames Across Languages (J6-9372) and Quantitative and Qualitative Analysis of the Unregulated Corporate Financial Reporting (J5-2554). This work has also received funding from the European Union's Horizon 2020 Research and Innovation program under grant agreement no. 825153 (EMBEDDIA). The results of this publication reflect only the authors' views, and the EC is not responsible for any use that may be made of the information it contains.

References

Anderson, Jonathan. 1981. Analysing the readability of English and non-English texts in the classroom with LIX. In *Seventh Australian Reading Association Conference*, pages 1–12, Darwin.

Azpiazu, Ion Madrazo and Maria Soledad Pera. 2019. Multiattentive recurrent neural network architecture for multilingual readability assessment. *Transactions of the Association for Computational Linguistics*, 7:421–436. DOI: `https://doi.org/10.1162/tacl_a_00278`

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Beltagy, Iz, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166. DOI: `https://doi.org/10.1109/72.279181`, PMID: 18267787

Bird, Steven and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions*, page 31, Barcelona. DOI: `https://doi.org/10.3115/1219044.1219075`

Bormuth, John R. 1969. *Development of Readability Analysis*. ERIC Clearinghouse.

Cho, Kyunghyun, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha. DOI: `https://doi.org/10.3115/v1/D14-1179`

Collins-Thompson, Kevyn. 2014. Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*, 165(2):97–135. DOI: `https://doi.org/10.1075/itl.165.2.01col`

Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Conneau, Alexis, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen. DOI: `https://doi.org/10.18653/v1/D17-1070`

Council of Europe, Council for Cultural Co-operation. Education Committee. Modern Languages Division. 2001. *Common European Framework of Reference for Languages: Learning, Teaching, Assessment*. Cambridge University Press.

Crossley, Scott A., Stephen Skalicky, Mihai Dascalu, Danielle S. McNamara, and Kristopher Kyle. 2017. Predicting text comprehension, processing, and familiarity in adult readers: New approaches to readability formulas. *Discourse Processes*, 54(5-6):340–359. DOI: `https://doi.org/10.1080/0163853X.2017.1296264`

Dale, Edgar and Jeanne S. Chall. 1948. A formula for predicting readability: Instructions. *Educational Research Bulletin*, pages 37–54.

Davison, Alice and Robert N. Kantor. 1982. On the failure of readability formulas to define readable texts: A case study from adaptations. *Reading Research Quarterly*, pages 187–209. DOI: `https://doi.org/10.2307/747483`

Deutsch, Tovly, Masoud Jasbi, and Stuart Shieber. 2020. Linguistic features for readability assessment. *arXiv preprint arXiv:2006.00377*. DOI: `https://doi.org/10.18653/v1/2020.bea-1.1`

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, MN.

Feng, Lijun. 2008. Text simplification: A survey, The City University of New York.

Feng, Lijun, Noémie Elhadad, and Matt Huenerfauth. 2009. Cognitively motivated features for readability assessment. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 229–237, Athens. DOI: `https://doi.org/10.3115/1609067.1609092`

Feng, Lijun, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A comparison of features for automatic readability assessment. In *COLING 2010: Posters*, pages 276–284, Beijing.

Filighera, Anna, Tim Steuer, and Christoph Rensing. 2019. Automatic text difficulty estimation using embeddings and neural networks. In *European Conference on Technology Enhanced Learning*, pages 335–348, Delft. DOI: `https://doi.org/10.1007/978-3-030-29736-7_25`

Flor, Michael, Beata Beigman Klebanov, and Kathleen M. Sheehan. 2013. Lexical tightness and text complexity. In *Proceedings of the Workshop on Natural Language Processing for Improving Textual Accessibility*, pages 29–38, Atlanta, GA.

Goldberg, Yoav and Jon Orwant. 2013. A data set of syntactic-ngrams over time from a very large corpus of English books. In *Second Joint Conference on Lexical and Computational Semantics*, pages 241–247, Atlanta, GA.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Gunning, Robert. 1952. *The Technique of Clear Writing*. McGraw-Hill, New York.

Halliday, Michael Alexander Kirkwood and Ruqaiya Hasan. 1976. *Cohesion in English*. Routledge.

Jawahar, Ganesh, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. DOI: `https://doi.org/10.18653/v1/P19-1356`

Jiang, Birong, Endong Xun, and Jianzhong Qi. 2015. A domain independent approach for extracting terms from research papers. In *Australasian Database Conference*, pages 155–166, Melbourne. DOI: `https://doi.org/10.1007/978-3-319-19548-3_13`

Kandel, Lilian and Abraham Moles. 1958. Application de l'indice de flesch à la langue française. *Cahiers Etudes de Radio-Télévision*, 19(1958):253–274.

Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *AAAI*, pages 2741–2749, Phoenix, AZ.

Kincaid, J. Peter, Robert P. Fishburne Jr, Richard L. Rogers, and Brad S. Chissom. 1975. *Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel*. Institute for Simulation and Training, University of Central Florida.

Kudo, Taku and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*. DOI: `https://doi.org/10.18653/v1/D18-2012`, PMID: 29382465

Landauer, Thomas K. 2011. Pearson's text complexity measure, Pearson.

Logar, Nataša, Miha Grčar, Marko Brakus, Tomaž Erjavec, Špela Arhar Holdt, Simon Krek, and Iztok Kosem. 2012. *Korpusi slovenskega jezika Gigafida, KRES, ccGigafida in ccKRES: gradnja, vsebina, uporaba*. Trojina, zavod za uporabno slovenistiko.

Lundberg, Scott M. and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4768–4777, Long Beach, CA.

Ma, Yi, Eric Fosler-Lussier, and Robert Lofthus. 2012. Ranking-based readability assessment for early primary children's literature. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 548–552, Montreal.

Madnani, Nitin and Aoife Cahill. 2018. Automated scoring: Beyond natural language processing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1099–1109, Santa Fe, NM.

Madrazo Azpiazu, Ion and Maria Soledad Pera. 2020. Is crosslingual readability assessment possible? *Journal of the Association for Information Science and Technology*, 71(6):644–656. DOI: `https://doi.org/10.1002/asi.24293`

McLaughlin, G. Harry. 1969. SMOG grading—a new readability formula. *Journal of Reading*, 12(8):639–646.

Mikolov, Tomáš, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černockỳ. 2011. Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth Annual Conference of the International Speech Communication Association*, pages 605–608.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, Florence.

Mohammadi, Hamid and Seyed Hossein Khasteh. 2019. Text as environment: A deep reinforcement learning text readability assessment model. *arXiv preprint arXiv:1912.05957*.

Nadeem, Farah and Mari Ostendorf. 2018. Estimating linguistic complexity for science texts. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 45–55, New Orleans, LA. DOI: `https://doi.org/10.18653/v1/W18-0505`

Napolitano, Diane, Kathleen M. Sheehan, and Robert Mundkowsky. 2015. Online readability and text complexity analysis with TextEvaluator. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 96–100, Denver, CO. DOI: `https://doi.org/10.3115/v1/N15-3020`

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, Doha. DOI: `https://doi.org/10.3115/v1/D14-1162`

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237, New Orleans, LA. DOI: `https://doi.org/10.18653/v1/N18-1202`

Petersen, Sarah E. and Mari Ostendorf. 2009. A machine learning approach to reading level assessment. *Computer Speech & Language*, 23(1):89–106. DOI: `https://doi.org/10.1016/j.csl.2008.04.003`

Pitler, Emily and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 186–195, Honolulu, HI. DOI: `https://doi.org/10.3115/1613715.1613742`

Schwarm, Sarah E. and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Ann Arbor, MI. DOI: `https://doi.org/10.3115/1219840.1219905`

Sheehan, Kathleen M., Michael Flor, and Diane Napolitano. 2013. A two-stage approach for generating unbiased estimates of text complexity. In *Proceedings of the Workshop on Natural Language Processing for Improving Textual Accessibility*, pages 49–58, Atlanta, GA.

Sheehan, Kathleen M., Irene Kostin, Yoko Futagi, and Michael Flor. 2010. Generating automated text complexity classifications that are aligned with targeted text complexity standards. *ETS Research Report Series*, 2010(2):i–44. DOI: `https://doi.org/10.1002/j.2333-8504.2010.tb02235.x`

Sheehan, Kathleen M., Irene Kostin, Diane Napolitano, and Michael Flor. 2014. The TextEvaluator tool: Helping teachers and test developers select texts for use in instruction and assessment. *The Elementary School Journal*, 115(2):184–209. DOI: `https://doi.org/10.1086/678294`

Škvorc, Tadej, Simon Krek, Senja Pollak, Špela Arhar Holdt, and Marko Robnik-Šikonja. 2019. Predicting Slovene text complexity using readability measures. *Contributions to Contemporary History (Spec. Issue on Digital Humanities and Language Technologies*, 59(1):198–220. DOI: `https://doi.org/10.51663/pnz.59.1.10`

Smith, Edgar A. and R. J. Senter. 1967. Automated readability index. *AMRL-TR. Aerospace Medical Research Laboratories (US)*, pages 1–14.

Todirascu, Amalia, Thomas François, Delphine Bernhard, Núria Gala, and Anne-Laure Ligozat. 2016. Are cohesive features relevant for text readability evaluation? In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 987–997, Osaka.

Ulčar, Matej and Marko Robnik-Šikonja. 2020. FinEst BERT and CroSloEngual BERT. In *International Conference on Text, Speech, and Dialogue*, pages 104–111, Brno.

Vajjala, Sowmya and Ivana Lučić. 2018. OneStopEnglish corpus: A new corpus for automatic readability assessment and text simplification. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 297–304, New Orleans, LA.

Vajjala, Sowmya and Detmar Meurers. 2012. On improving the accuracy of readability classification using insights from second language acquisition. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 163–173, Montreal.

Van Dijk, Teun Adrianus. 1977. *Text and Context: Explorations in the Semantics and Pragmatics of Discourse*. Longman London.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, Long Beach, CA.

Wang, Sinong, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Wang, Ziyu, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1995–2003, New York.

Williams, Geoffrey. 2006. Michael Hoey. Lexical priming: A new theory of words and language. *International Journal of Lexicography*, 19(3):327–335. DOI: `https://doi.org/10.1093/ijl/ecl017`

Xia, Menglin, Ekaterina Kochmar, and Ted Briscoe. 2016. Text readability assessment for second language learners. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 12–22, San Diego, CA. DOI: `https://doi.org/10.18653/v1/W16-0502`, PMCID: PMC4879617

Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Richard Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, Lille.

Xu, Wei, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association of Computational Linguistics*, 3(1):283–297. DOI: `https://doi.org/10.1162/tacl_a_00139`

Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, CA. DOI: https://doi.org/10.18653/v1/N16-1174

Zhang, Xiang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657, Montreal.

Zhu, Yukun, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 19–27, Santiago. DOI: https://doi.org/10.1109/ICCV.2015.11

Zwitter Vitez, Ana. 2014. Ugotavljanje avtorstva besedil: primer "trenirkarjev." In *Language Technologies: Proceedings of the 17th International Multiconference Information Society - IS 2014*, pages 131–134, Ljubljana.

3.4 Final Remarks

Same as in the field of AP, recently the trends go towards employment of neural models for readability detection. Our study (Martinc, Pollak, & Robnik-Šikonja, 2021) was one of the first that proposed to use transformers to tackle readability detection in a supervised way, i.e. by modelling it as a multi-class classification problem, instead of using more traditional classifiers with several types of hand-crafted features. Very much in line with our own research, some novel approaches also consider combining these two approaches. For example, in the study by Lee et al. (2021) they test several combinations of neural transformer models and non-neural random forest models and report improvements in performance achieved by this hybrid model on all test datasets.

On the other hand, novel approaches that would tackle readability in an unsupervised way are rare and these new unsupervised approaches usually consider only symbolic features (Ehara, 2021). This might be connected with the fact that other studies confirmed our findings that the correlation between readability and neural language model statistics, such as perplexity, is weak (Miaschip et al., 2020), and that traditional readability formulas, despite their deficiencies, work sufficiently well by just considering shallow lexical indicators. Nevertheless, as we show in our study, language models do carry information that, when meaningfully combined with shallow indicators of readability, can increase performance, robustness and adaptability of the unsupervised measures.

This is especially important for readability detection for less resourced languages without readability datasets on which you could train supervised approaches. Traditional readability formulas have limited applicability, when it comes to less resourced languages, since most of them have been designed specifically for the use on English. And as far as we are aware, for most languages there exist no specific language adapted readability formulas. Further development and employment of unsupervised hybrid approaches is therefore, at least in our opinion, still one of the most perspective strategies for bridging this research gap.

Chapter 4

Keyword Extraction with Transformer-Based Neural Network and Symbolic TF-IDF Statistics

In contrast with the tasks of author profiling and readability prediction, where neural approaches are still not massively employed and tend to perform worse than other approaches based on extensive feature engineering, sequence labelling is a field dominated by neural approaches, which showcased superior performance in comparison to other symbolic approaches on several tasks with a sufficient amount of data, such as NER and POS tagging. Nevertheless, for some sequence labelling tasks and also for some languages, the amount of training data is still insufficient. One of these tasks is keyword extraction, which we explore in this chapter. In Section 4.1, we describe the problem of keyword extraction and outline the proposed approach of tackling the task with a combination of neural and symbolic approaches. In Section 4.2, we describe existing approaches for keywords extraction, in Section 4.3, we describe the proposed neural architecture for the task at hand that requires less labelled resources for training and in Section 4.4, we show how the proposed neural approach can be combined with a symbolic TF-IDF-based approach for keyword extraction in order to improve the recall of the method. In this chapter, there are two enclosed publications, namely a journal paper *TNT-KID: Transformer-based neural tagger for keyword identification* (Martinc, Škrlj, et al., 2021), which describes the proposed neural approach towards keyword extraction in detail, and a workshop paper *Extending neural keyword extraction with TF-IDF tagset matching* (Koloski et al., 2021), which describes how the proposed neural approach can be combined with a symbolic approach.

4.1 Introduction

With keyword extraction, we refer to the automatic extraction of words and phrases that represent crucial semantic aspects of the text and summarize its content. This task is currently gaining traction due to exponential growth in the amount of raw unlabelled textual data which lack metadata that would help with its organization. Therefore, development of algorithms capable of efficient organization, categorization, and summarization of large amounts of text documents has become a necessity (Firoozeh et al., 2020).

While first methods for automated keyword extraction have been developed more than a decade ago (Mihalcea & Tarau, 2004; Witten et al., 2005), the progress has been steady and novel unsupervised approaches, such as YAKE (Campos et al., 2018) and RaKUn (Škrlj et al., 2019) have been proven quite useful and transferable across domains and languages, since they tend to be language agnostic and do not require any training data. On the other hand, there is still a large performance gap between unsupervised approaches and supervised, with the latter being much more efficient due to better semantic modelling and the capability to adapt to the syntax, semantics, content, genre and keyword assignment regime of a specific text. The novel neural algorithms for keyword extraction (J. Chen et al., 2018; Meng et al., 2019; Yuan et al., 2020) achieve excellent performance but require vast amounts of training data and are therefore not transferable to domains and languages lacking large manually labelled resources.

The main foci of our research on the topic of keyword extraction are the following:

- We propose a Transformer-based Neural Tagger for Keyword IDentification (TNT-KID) that requires far less labelled data than other neural approaches and still achieves performance comparable to the state-of-the-art supervised approaches in settings with plenty of training data. We also show that the proposed neural model outperforms other approaches by a large margin when training data is scarce.[1]
- By combining the proposed TNT-KID approach with a more traditional symbolic TF-IDF-based approach towards keyword extraction we manage to drastically improve the recall of the proposed system, which makes the system more usable in news media setting, in which a preferred output is a predefined number of keywords for each input news article.

4.2 Related Work

Related work on keyword detection can be divided into supervised and unsupervised approaches. Traditionally, the task consisted of two steps, extracting keyword candidates from the text according to a number of syntactic and lexical features and ranking these candidates according to different heuristics and selecting the top candidates as keywords (Yuan et al., 2020).

When it comes to supervised approaches, first approaches employed frequency-based statistics such as TF-IDF and the term's position in the text as features for term identification (Medelyan et al., 2009; Witten et al., 2005). These features are fed to the Naive Bayes classifier, which determines for each word or phrase in the text if it is a keyword or not. More recently, the keywords extraction task began to be modelled as sequence labelling (Gollapalli et al., 2017). Some approaches of this type employed Conditional Random Fields (CRF) tagger for keyword extraction, while most relied on neural architectures, in most cases RNNs (Luan et al., 2017).

Recent state-of-the-art approaches decided to tackle the problem as a sequence-to-sequence generation task. They employ a generative model with a recurrent encoder-decoder framework, which is, besides keyword extraction, also capable of finding keywords that do not appear in the text (Meng et al., 2017). Several variants of this approach have been proposed, namely the original CopyRNN method (Meng et al., 2019), CatSeqD proposed by Yuan et al. (2020), who incorporated two diversity mechanisms into the model, a Semi-supervised CopyRNN proposed by Ye and Wang (2018), which, besides the labelled samples, also leverages unlabelled samples, and a so-called CorrRNN proposed by J. Chen et al. (2018), who proposed additional mechanisms that handle repetitions and increase keyphrase diversity.

On the other hand, at least one study employed transformers (Vaswani et al., 2017) for keyword detection. Sahrawat et al. (2020) used several transformer and recurrent architectures (BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), GPT-2 (Radford et al.,

[1]Code is available under the MIT license at https://gitlab.com/matej.martinc/tnt_kid/.

2019), ELMo (Peters et al., 2018), etc.) to generate contextual embeddings that were fed into a bidirectional long short-term memory (BiLSTM) network with an additional conditional random fields layer (BiLSTM-CRF). They also experimented with SciBERT (Beltagy et al., 2019), a BERT version pretrained on a large multi-domain corpus of scientific publications. They observed that this genre-specific pretraining on texts of the same genre as the texts in the keyword datasets used in their experiments improves the performance of the model.

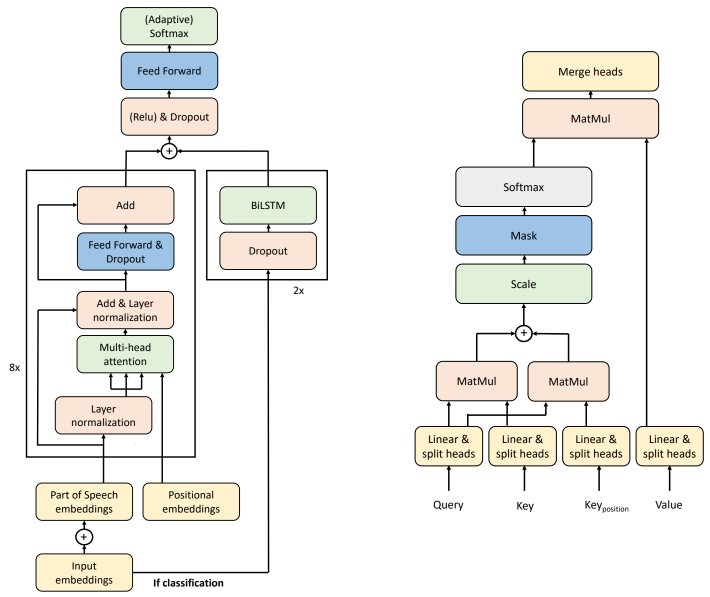
Unsupervised keyword extraction methods can be divided into four main categories, namely statistical, graph-based, embeddings-based, and language model-based methods. Statistical methods use statistical characteristics of text to capture keywords. A state-of-the-art algorithm of this type is YAKE (Campos et al., 2018) that considers features such as casing, position, frequency, relatedness to context and dispersion of a specific term to identify keywords. Graph-based methods, such as TextRank (Mihalcea & Tarau, 2004), Single Rank (Wan & Xiao, 2008) or TopicRank (Bougouin et al., 2013), construct graphs to rank words based on their position in the graph. The most recent graph-based keyword detector is RaKUn (Škrlj et al., 2019) that uses several novel techniques, such as for example expanding the initial lexical graph with the introduction of meta-vertices and employment of graph redundancy filters.

Another branch of unsupervised methods for keyword extraction are embedding-based methods, which consider embedding representations for identifying keywords. These methods can rely on building direct graphs based on embedding-based measures, such as cosine distance between embeddings, where candidate keyphrases are represented as vertices, as in the case of the Key2Vec method proposed by Mahata et al. (2018). Another option is the so-called EmbedRank approach proposed by Bennani-Smires et al. (2018). Here, part of speech (POS)-based patterns are used to extract initial candidate keyphrases and after that each candidate is ranked according to the cosine distance between the candidate and the embedding of the document in which it appears. For representing candidate phrases and documents, Sent2Vec embeddings (Pagliardini et al., 2018) are used.

Language model-based methods, which use language model-derived statistics to extract keywords from text, are less common. An interesting approach was proposed by Tomokiyo and Hurst (2003), who extracted keyphrases by measuring Kullback–Leibler divergence (Vidyasagar, 2010) between several unigram and n-gram language models. This system used two features, namely phraseness, which measures if a given word sequence is in fact a phrase, and informativeness, which measures if the most important ideas in the document are captured in a specific keyphrase.

4.3 Transformer-Based Sequence Labelling Approach to Keyword Extraction

As mentioned above, unsupervised approaches, which in the majority of cases (the exception being embedding-based approaches) rely on symbolic frequency-based features, lack the effectiveness of the neural supervised approaches that nevertheless require a large amount of labelled data for successful training. We propose Transformer-based Neural Tagger for Keyword IDentification (TNT-KID) (Martinc, Škrlj, et al., 2021) that tries to alleviate the problems of neural approaches that require a lot of data and unsupervised symbolic approaches that are less effective (**i.e. in this section we aim to achieve the stated goal G3**). The proposed approach achieves performance comparable to the state-of-the-art supervised approaches while requiring only a fraction of manually labelled data required by other neural approaches. This allows the model to be employed for less resourced languages and domains.



(a) Model architecture. (b) The attention mechanism.

Figure 4.1: TNT-KID's architecture overview.

This is possible due to the transfer learning technique (Howard & Ruder, 2018; Peters et al., 2018), where a model is first trained in an unsupervised way as a language model on a large unlabelled corpus and then fine-tuned on a smaller corpus with manually labelled keywords for keyword detection. While this approach has recently become a well established procedure in the field of NLP, it was rarely tested in the domain-specific setting. In fact, the standard paradigm is that the model should be pretrained on a very large general corpus (e.g., the English BERT model (Devlin et al., 2019) was pretrained on the corpus containing 3,300 million tokens). In contrast, in this work we propose a pretraining on a much smaller unlabelled domain-specific corpora, allowing easier transfer of the model to languages with less textual resources and reducing the time and computer resources required for training. We show that this type of pretraining is sufficient for the system to grasp the semantic information inside the text and adapt to a specific domain, which results in the reduced amount of labelled data required for training.

TNT-KID relies on a modified transformer encoder architecture (Vaswani et al., 2017). This encoder consists of a normalization layer that is followed by a multi-head attention mechanism, another normalization layer and the fully connected feed-forward and dropout layers, same as in the GPT-2 architecture proposed by Radford et al. (2019). The encoder also contains a residual connection around the attention mechanism. During language model pretraining, a language model head consisting of a dropout layer, a feed forward layer and the adaptive softmax layer (Grave et al., 2017) is added on top of the encoder. The language model head is replaced with a token classification head during fine-tuning. This head contains a ReLu non-linearity, a dropout layer, a feed forward classification layer and softmax layer, which returns probability for each token that it is either a keyword (or part of the keyphrase) or not.

We hypothesise that token position is especially important in the keyword identification task (e.g., the probability of the first word in the text being a keyword is much larger than the probability that the last word is a keyword), therefore we propose modifications of the Transformer architecture to better tackle the importance of relation between each token and each position. First, we propose a re-parametrization of the attention mechanism (see Figure 4.1b), which allows the model to better distinguish between the semantic/grammatical and purely positional information and therefore assign attention to some tokens just on the basis of their position. Another modification that affects the modelling of positional information, is the addition of the two-layer randomly initialised encoder, consisting of dropout and two bidirectional long short-term memory (BiLSTM) layers, during the fine-tuning token classification step of the model training.

The third modification aims to decrease the time complexity of the model's pretraining and involves replacing the standard input embedding layer and softmax function with adaptive input representations (Baevski & Auli, 2019) and an adaptive softmax (Grave et al., 2017) during the pretraining language modelling phase. Another design choice that affects the computational complexity of the model is the usage of the Sentencepiece byte-pair encoding tokenization scheme (Kudo & Richardson, 2018), with which the vocabulary is limited to about 32,000 tokens, which has a positive effect on the efficiency of the language model head softmax layer, since the probability distribution only needs to be derived across 32,000 subwords and not an entire word vocabulary.

During language model pretraining, we employ the autoregressive language modelling objective, where the task can be defined as predicting a probability distribution of words from the fixed size vocabulary V, for word w_{t}, given the historical sequence $w_{\mathrm{1:t\text{-}1}} = [w_1, ..., w_{t-1}]$. We show in the ablation study (Martinc, Škrlj, et al., 2021), that this objective results in much bigger performance gains than the masked language modelling objective, first proposed by Devlin et al. (2019), when the size of corpus used for pretraining is limited.

During fine-tuning, we model the keyword tagging task as a binary classification task and the model is trained to predict if a word in the sequence is a keyword or not. Due to the imbalance between two classes (i.e. majority of words in the text tend not to be keywords), we propose a custom classification loss function that considers this imbalance. Probabilities for words in the sequence are aggregated into two distinct sets, one for each class and two negative log losses (NLL) are calculated, one for each probability set. The two NLLs are normalized with the size of the set and summed (see Martinc, Škrlj, et al. (2021) for details). To produce the final set of keywords for each document, tagged words are extracted from the text and duplicates are removed. Note that a sequence of two or more sequential keywords is always interpreted as a multi-word keyphrase. From the resulting set, keyphrases longer than four words and keyphrases containing punctuation (with the exception of dashes and apostrophes) are discarded. The final output is a ranked list, in which the keywords are arranged according to the softmax probability assigned by the model in a descending order.

The proposed approach was tested on seven English datasets with manually labelled keywords from two distinct genres, scientific papers about computer science and news. For the computer science domain we tested the model on KP20k (Meng et al., 2017), Inspec (Hulth, 2003), Krapivin (Krapivin et al., 2009), NUS (Nguyen & Kan, 2007), and SemEval (S. N. Kim et al., 2010) datasets. The model was also tested on three datasets from the news domain, namely KPTimes (Gallina et al., 2019), JPTimes (Gallina et al., 2019) and DUC (Wan & Xiao, 2008) datasets. The statistics about the datasets that are used for training and testing of our models are presented in Table 4.1.

In the pretraining phase, two language models were trained, one on the concatenation of all the texts from the computer science domain and the other on the concatenation of all the texts from the news domain. After that, the trained language models were fine-

Table 4.1: Datasets used for empirical evaluation of keyword extraction algorithms. *No.docs* stands for the number of documents, *Avg. doc. length* stands for average document length in the corpus (in terms of number of words, i.e., we split the text by white-space), *Avg. kw.* stands for average number of keywords per document in the corpus, *% present kw.* stands for the percentage of keywords that appear in the corpus (i.e., percentage of document's keywords that appear in the text of the document) and *Avg. present kw.* stands for the average number of keywords per document that actually appear in the text of the specific document.

Dataset	No. docs	Avg. doc. length	Avg. kw.	% present kw.	Avg. present kw.
Computer science papers					
KP20k-train	530,000	156.34	5.27	62.43	3.29
KP20k-valid	20,000	156.55	5.26	62.30	3.28
KP20k-test	20,000	156.52	5.26	62.55	3.29
Inspec-valid	1500	125.21	9.57	76.92	7.36
Inspec-test	500	121.82	9.83	78.14	7.68
Krapivin-valid	1844	156.65	5.24	54.34	2.85
Krapivin-test	460	157.76	5.74	55.66	3.20
NUS-test	211	164.80	11.66	50.47	5.89
SemEval-valid	144	166.86	15.67	45.43	7.12
SemEval-test	100	183.71	15.07	44.53	6.71
News articles					
KPTimes-train	259,923	783.32	5.03	47.30	2.38
KPTimes-valid	10,000	784.65	5.02	46.78	2.35
KPTimes-test	10,000	783.47	5.04	47.59	2.40
JPTimes-test	10,000	503.00	5.03	76.73	3.86
DUC-test	308	683.14	8.06	96.62	7.79

tuned on each dataset's *validation* sets (see Table 4.1) for a maximum of 10 epochs and then tested on the test set. We compare the TNT-KID approach to several systems for keyword extraction mentioned in Section 4.2 by measuring the $F_1@k$ score, precision@k and recall@k with $k \in \{5, 10\}$ to asses the performance of each model[2]. We use a simple TF-IDF-based keyword extraction (i.e., extraction of k words with the highest TF-IDF from each document) as an unsupervised baseline. For KEA and Maui, we only report results that were available in the related work (KPTimes, JPTimes and DUC corpus) due to bad performance of the algorithms on all the corpora for which results are available. For a supervised baseline, we report results for the unmodified pretrained GPT-2 (Radford et al., 2019) model with a standard feed forward token classification head. The results are presented in Table 4.2.

On average, TNT-KID offers the best performance on the test datasets according to $F_1@5$ and $F_1@10$ and is closely followed by GPT-2 + BiLSTM. The different generative approaches towards keyword extraction (CopyRNN, CatSeqD, Semi-supervised CopyRNN and CorrRNN) offer similar performance according to all criteria. Similar could be said for all the unsupervised approaches, which on average offer a surprisingly homogeneous performance. Overall, supervised neural network approaches outperform all other approaches by a large margin.

TNT-KID performs the best on four datasets in terms of $F_1@10$ but only on one dataset in terms of $F_1@5$. This can be explained by the fact that TNT-KID generally detects more

[2]Note that several measures exist that could be used for the evaluation of keyword extractors. Arguably, measures which also consider the ranking of keywords in a retrieved list, such as mean average precision and mean reciprocal rank, would be more suitable. Nevertheless, we opted to evaluate our approach by measuring $F_1@k$ score, precision@k and recall@k in order to be able to compare our results to a large set of other methods, since these are the three most common evaluation metrics applied in the related work.

Table 4.2: Evaluation of supervised and unsupervised keyword extractors. Results marked with * were obtained by our implementation or reimplementation of the algorithm and results without * were reported in the related work.

	KP20k	Inspec	Krapivin	NUS	SemEval	KPTimes	JPTimes	DUC	Average
				Unsupervised algorithms					
				TfIdf					
F_1@5	0.072	0.160	0.067	0.112	0.088	0.179*	0.266*	0.098*	0.130
F_1@10	0.094	0.244	0.093	0.140	0.147	0.151*	0.229*	0.120*	0.152
				TextRank					
F_1@5	0.181	0.286	0.185	0.230	0.217	0.022*	0.012*	0.120*	0.157
F_1@10	0.151	0.339	0.160	0.216	0.226	0.030*	0.026*	0.181*	0.166
				YAKE					
F_1@5	0.141*	0.204*	0.215*	0.159*	0.151*	0.105*	0.109*	0.106*	0.149
F_1@10	0.146*	0.223*	0.196*	0.196*	0.212*	0.118*	0.135*	0.132*	0.170
				RaKUn					
F_1@5	0.177*	0.101*	0.127*	0.224*	0.167*	0.168*	0.225*	0.189*	0.172
F_1@10	0.160*	0.108*	0.106*	0.193*	0.159*	0.139*	0.185*	0.172*	0.153
				Key2Vec					
F_1@5	0.080*	0.121*	0.068*	0.109*	0.081*	0.126*	0.158*	0.062*	0.101
F_1@10	0.090*	0.181*	0.082*	0.121*	0.126*	0.116*	0.145*	0.078*	0.117
				EmbedRank					
F_1@5	0.135*	0.345*	0.149*	0.173*	0.189*	0.063*	0.081*	0.219*	0.169
F_1@10	0.134*	0.394*	0.158*	0.190*	0.217*	0.057*	0.074*	0.246*	0.184
				Supervised algorithms					
				KEA					
F_1@5	0.046	0.022	0.018	0.073	0.068	/	/	/	/
F_1@10	0.044	0.022	0.017	0.071	0.065	/	/	/	/
				Maui					
F_1@5	0.005	0.035	0.005	0.004	0.011	/	/	/	/
F_1@10	0.005	0.046	0.007	0.006	0.014	/	/	/	/
				Semi-supervised CopyRNN					
F_1@5	0.308	0.326	0.296	0.356	0.322	/	/	/	/
F_1@10	0.245	0.334	0.240	0.320	0.294	/	/	/	/
				CopyRNN					
F_1@5	0.317	0.244	0.305	0.376	0.318	0.406*	0.256*	0.083	0.288
F_1@10	0.273	0.289	0.266	0.352	0.318	0.393	0.246	0.105	0.280
				CatSeqD					
F_1@5	0.348	0.276	**0.325**	**0.374**	**0.327**	0.424*	0.238*	0.063*	0.297
F_1@10	0.298	0.333	0.285	**0.366**	**0.352**	0.424*	0.238*	0.063*	0.295
				CorrRNN					
F_1@5	/	/	0.318	0.361	0.320	/	/	/	/
F_1@10	/	/	0.278	0.335	0.320	/	/	/	/
				GPT-2					
F_1@5	0.275*	0.413*	0.253*	0.318*	0.257*	0.421*	0.331*	0.298*	0.321
F_1@10	0.278*	0.469*	0.253*	0.323*	0.278*	0.423*	0.336*	0.312*	0.334
				GPT-2 + BiLSTM-CRF					
F_1@5	**0.355***	**0.462***	0.287*	0.329*	0.246*	0.478*	**0.386***	**0.333***	0.360
F_1@10	**0.360***	0.524*	0.288*	0.336*	0.274*	0.479*	**0.389***	0.371*	0.378
				TNT-KID					
F_1@5	0.336*	0.460*	0.310*	0.350*	0.283*	**0.485***	0.359*	0.318*	**0.363**
F_1@10	0.338*	**0.536***	**0.320***	0.358*	0.337*	**0.485***	0.361*	**0.373***	**0.389**

keywords than other neural algorithms due to the proposed custom loss function. While this results in better recall and consequentially also in better performance when up to 10 keywords need to be predicted, it also hurts precision of the system, which negatively affects the F_1 score in a setting where only up to 5 keywords need to be predicted.

The performances of TNT-KID and GPT-2 + BiLSTM-CRF are comparable on a large majority of datasets according to both criteria, with the difference being the biggest on the SemEval dataset. It should be noted that TNT-KID employs only 8 attention layers, 8 attention heads and an embedding size of 512 instead of the standard 12 attention layers, 12 attention heads and an embeddings size of 768, which the pretrained GPT-2 model employs. While GPT-2 + BiLSTM-CRF employs a computationally demanding CRF layer, TNT-KID employs an additional BiLSTM encoder during the classification phase, which makes it slower than the unmodified GPT-2. If the BiLSTM-CRF layer is not used, as in the case of the vanilla GPT-2 model with a standard token classification head, the performance of the GPT-2 model becomes much less competitive, even though it still on average manages to outperform all non-transformer-based algorithms.

The journal paper containing the details about the study is enclosed below.

Natural Language Engineering (2021), 1–40
doi:10.1017/S1351324921000127

CAMBRIDGE
UNIVERSITY PRESS

ARTICLE

TNT-KID: Transformer-based neural tagger for keyword identification

Matej Martinc[1, 2,*], Blaž Škrlj[1, 2] and Senja Pollak[1]

[1]Jožef Stefan Institute, Department of Knowledge Technologies, Jamova 39, 1000 Ljubljana, Slovenia and [2]Jožef Stefan International Postgraduate School, Department of Knowledge Technologies, Jamova 39, 1000 Ljubljana, Slovenia
*Corresponding author. E-mail: matej.martinc@ijs.si

(Received 17 March 2020; revised 4 May 2021; accepted 10 May 2021)

Abstract
With growing amounts of available textual data, development of algorithms capable of automatic analysis, categorization, and summarization of these data has become a necessity. In this research, we present a novel algorithm for keyword identification, that is, an extraction of one or multiword phrases representing key aspects of a given document, called Transformer-Based Neural Tagger for Keyword IDentification (TNT-KID). By adapting the transformer architecture for a specific task at hand and leveraging language model pretraining on a domain-specific corpus, the model is capable of overcoming deficiencies of both supervised and unsupervised state-of-the-art approaches to keyword extraction by offering competitive and robust performance on a variety of different datasets while requiring only a fraction of manually labeled data required by the best-performing systems. This study also offers thorough error analysis with valuable insights into the inner workings of the model and an ablation study measuring the influence of specific components of the keyword identification workflow on the overall performance.

Keywords: Keyword extraction; Transfer learning; Transformer architecture

1. Introduction

With the exponential growth in the amount of available textual resources, organization, categorization, and summarization of these data presents a challenge, the extent of which becomes even more apparent when it is taken into account that a majority of these resources do not contain any adequate meta information. Manual categorization and tagging of documents is unfeasible due to a large amount of data, therefore, development of algorithms capable of tackling these tasks automatically and efficiently has become a necessity (Firoozeh *et al.* 2020).

One of the crucial tasks for organization of textual resources is keyword identification, which deals with automatic extraction of words that represent crucial semantic aspects of the text and summarize its content. First automated solutions to keyword extraction have been proposed more than a decade ago (Witten *et al.* 1999; Mihalcea and Tarau 2004) and the task is currently again gaining traction, with several new algorithms proposed in the recent years. Novel unsupervised approaches, such as RaKUn (Škrlj, Repar, and Pollak 2019) and YAKE (Campos *et al.* 2018), work fairly well and have some advantages over supervised approaches, as they are language and genre independent, do not require any training and are computationally undemanding. On the other hand, they also have a couple of crucial deficiencies:

- Term frequency–inverse document frequency (TfIdf) and graph-based features, such as PageRank, used by these systems to detect the importance of each word in the document, are based only on simple statistics like word occurrence and co-occurrence, and are therefore unable to grasp the entire semantic information of the text.

- Since these systems cannot be trained, they cannot be adapted to the specifics of the syntax, semantics, content, genre and keyword assignment regime of a specific text (e.g., a variance in a number of keywords).

These deficiencies result in a much worse performance when compared to the state-of-the-art supervised algorithms (see Table 2), which have a direct access to the gold-standard keyword set for each text during the training phase, enabling more efficient adaptation. Most recent supervised neural algorithms (Chen *et al.* 2018; Meng *et al.* 2019; Yuan *et al.* 2020), therefore, achieve excellent performance under satisfactory training conditions and can model semantic relations much more efficiently than algorithms based on simpler word frequency statistics. On the other hand, these algorithms are resource demanding, require vast amounts of domain-specific data for training, and can therefore not be used in domains and languages that lack manually labeled resources of sufficient size.

In this research, we propose Transformer-Based Neural Tagger for Keyword IDentification (TNT-KID)[a] that is capable of overcoming the aforementioned deficiencies of supervised and unsupervised approaches. We show that while requiring only a fraction of manually labeled data required by other neural approaches, the proposed approach achieves performance comparable to the state-of-the-art supervised approaches on test sets for which a lot of manually labeled training data are available. On the other hand, if training data that is sufficiently similar to the test data are scarce, our model outperforms state-of-the-art approaches by a large margin. This is achieved by leveraging the transfer learning technique, where a keyword tagger is first trained in an unsupervised way as a language model on a large corpus and then fine-tuned on a (usually) small-sized corpus with manually labeled keywords. By conducting experiments on two different domains, computer science articles and news, we show that the language model pretraining allows the algorithm to successfully adapt to a specific domain and grasp the semantic information of the text, which drastically reduces the needed amount of labeled data for training the keyword detector.

The transfer learning technique (Peters *et al.* 2018; Howard and Ruder 2018), which has recently become a well-established procedure in the field of natural language processing (NLP), in a large majority of cases relies on very large unlabeled textual resources used for language model pretraining. For example, a well-known English BERT model (Devlin *et al.* 2019) was pretrained on the Google Books Corpus (Goldberg and Orwant 2013) (800 million tokens) and Wikipedia (2500 million tokens). On the other hand, we show that smaller unlabeled domain-specific corpora (87 million tokens for computer science and 232 million tokens for news domain) can be successfully used for unsupervised pretraining, which makes the proposed approach easily transferable to languages with less textual resources and also makes training more feasible in terms of time and computer resources available.

Unlike most other proposed state-of-the-art neural keyword extractors (Meng *et al.* 2017, 2019; Chen *et al.* 2018; Ye and Wang 2018; Yuan *et al.* 2020), we do not employ recurrent neural networks but instead opt for a transformer architecture (Vaswani *et al.* 2017), which has not been widely employed for the task at hand. In fact, the study by Sahrawat *et al.* (2020) is the only study we are aware of that employs transformers for the keyword extraction task. Another difference between our approach and most very recent state-of-the-art approaches from the related work is also task formulation. While Meng *et al.* (2017), (2019) and Yuan *et al.* (2020) formulate a keyword extraction task as a sequence-to-sequence generation task, where the classifier is trained to generate an output sequence of keyword tokens step by step according to the input sequence and the previous generated output tokens, we formulate a keyword extraction task as a sequence

[a]Code is available under the MIT license at https://gitlab.com/matej.martinc/tnt_kid/.

labeling task, similar as in Gollapalli, Li, and Yang (2017), Luan, Ostendorf, and Hajishirzi (2017) and Sahrawat *et al.* (2020).

Besides presenting a novel keyword extraction procedure, the study also offers an extensive error analysis, in which the visualization of transformer attention heads is used to gain insights into inner workings of the model and in which we pinpoint key factors responsible for the differences in performance of TNT-KID and other state-of-the-art approaches. Finally, this study also offers a systematic evaluation of several building blocks and techniques used in a keyword extraction workflow in the form of an ablation study. Besides determining the extent to which transfer learning affects the performance of the keyword extractor, we also compare two different pretraining objectives, autoregressive language modeling and masked language modeling (Devlin *et al.* 2019), and measure the influence of transformer architecture adaptations, a choice of input encoding scheme and the addition of part-of-speech (POS) information on the performance of the model.

The paper is structured as follows. Section 2 addresses the related work on keyword identification and covers several supervised and unsupervised approaches to the task at hand. Section 3 describes the methodology of our approach, while in Section 4 we present the datasets, conducted experiments and results. Section 5 covers error analysis, Section 6 presents the conducted ablation study, while the conclusions and directions for further work are addressed in Section 7.

2. Related work

This section overviews selected methods for keyword extraction, supervised in Section 2.1 and unsupervised in Section 2.2. The related work is somewhat focused on the newest keyword extraction methods, therefore, for a more comprehensive survey of slightly older methods, we refer the reader to Hasan and Ng (2014).

2.1 Supervised keyword extraction methods

Traditional supervised approaches to keyword extraction considered the task as a two step process (the same is true for unsupervised approaches). First, a number of syntactic and lexical features are used to extract keyword candidates from the text. Second, the extracted candidates are ranked according to different heuristics and the top n candidates are selected as keywords (Yuan *et al.* 2020). One of the first supervised approaches to keyword extraction was proposed by Witten *et al.* (1999), whose algorithm named KEA uses only TfIdf and the term's position in the text as features for term identification. These features are fed to the Naive Bayes classifier, which is used to determine for each word or phrase in the text if it is a keyword or not. Medelyan, Frank, and Witten (2009) managed to build on the KEA approach and proposed the *Maui* algorithm, which also relies on the Naive Bayes classifier for candidate selection but employs additional semantic features, such as, for example, *node degree*, which quantifies the semantic relatedness of a candidate to other candidates, and *Wikipedia-based keyphraseness*, which is the likelihood of a phrase being a link in the Wikipedia.

Wang, Peng, and Hu (2006) was one of the first studies that applied a feedforward neural network classifier for the task at hand. This approach still relied on manual feature engineering and features such as TfIdf and appearance of the keyphrase candidate in the title or heading of the given document. On the other hand, Villmow, Wrzalik, and Krechel (2018) applied a Siamese Long Short-Term Memory (LSTM) network for keyword extraction, which no longer required manual engineering of statistical features.

A more recent supervised approach is the so-called sequence labeling approach to keyword extraction by Gollapalli *et al.* (2017), where the idea is to train a keyword tagger using token-based linguistic, syntactic and structural features. The approach relies on a trained Conditional Random Field (CRF) tagger and the authors demonstrated that this approach is capable of working on-par with slightly older state-of-the-art systems that rely on information from the Wikipedia and citation networks, even if only within-document features are used. In another sequence labeling approach proposed by Luan *et al.* (2017), a sophisticated neural network is built by combing an input layer comprising a concatenation of word, character and part-of-speech embeddings, a bidirectional Long Short-Term Memory (BiLSTM) layer and, a CRF tagging layer. They also propose a new semi-supervised graph-based training regime for training the network.

Some of the most recent state-of-the-art approaches to keyword detection consider the problem as a sequence-to-sequence generation task. The first research leveraging this tactic was proposed by Meng *et al.* (2017), employing a generative model for keyword prediction with a recurrent encoder–decoder framework with an attention mechanism capable of detecting keywords in the input text sequence and also potentially finding keywords that do not appear in the text. Since finding absent keywords involves a very hard problem of finding a correct class in a set of usually thousands of unbalanced classes, their model also employs a copying mechanism (Gu *et al.* 2016) based on positional information, in order to allow the model to find important keywords present in the text, which is a much easier problem.

The approach was further improved by Chen *et al.* (2018), who proposed additional mechanisms that handle repetitions and increase keyphrase diversity. In their system named CorrRNN, the so-called coverage vector is employed to check whether the word in the document has been summarized by previous keyphrases. Also, before the generation of each new keyphrase, preceding phrases are taken into account to eliminate generation of duplicate phrases.

Another improvement was proposed by Ye and Wang (2018), who tried to reduce the amount of data needed for successful training of the model proposed by Meng *et al.* (2017). They propose a semi-supervised keyphrase generation method (in Section 4, we refer to this model as a Semi-supervised CopyRNN), which, besides the labeled samples, also leverages unlabeled samples, that are labeled in advance by syntetic keyphrases obtained with unsupervised keyword extraction methods or by employing a self-learning algorithms. The novel keyword extraction approach proposed by Wang *et al.* (2018) also tries to reduce the amount of needed labeled data. The employed Topic-Based Adversarial Neural Network (TANN) is capable of leveraging the unlabeled data in the target domain and also data from the resource-rich source domain for the keyword extraction in the target domain. They propose a special topic correlation layer, in order to incorporate the global topic information into the document representation, and a set of domain-invariant features, which allow the transfer from the source to the target domain by adversarial training on the topic-based representations.

The study by Meng *et al.* (2019) tried to improve the approach proposed in their previous study (Meng *et al.* 2017) by investigating different ways in which the target keywords can be fed to a classifier during the training phase. While the original system used the so-called *one-to-one* approach, where a training example consists of an input text and a single keyword, the improved model employs a *one-to-seq* approach, where an input text is matched with a concatenated sequence made of all the keywords for a specific text. The study also shows that the order of the keywords in the text matters. The best-performing model from Meng *et al.* (2019), named CopyRNN, is used in our experiments for the comparison with the state of the art (see Section 4). A *one-to-seq* approach has been further improved by Yuan *et al.* (2020), who incorporated two diversity mechanisms into the model. The mechanisms (called *semantic coverage* and *orthogonal regularization*) constrain the overall inner representation of a generated keyword sequence to be semantically similar to the overall meaning of the source text, and therefore force the model to produce diverse keywords. The resulting model leveraging these mechanisms has been named CatSeqD and is also used in our experiments for the comparison between TNT-KID and the state of the art.

A further improvement of the generative approach towards keyword detection has been proposed by Chan *et al.* (2019), who integrated a reinforcement learning (RL) objective into the keyphrase generation approach proposed by Yuan *et al.* (2020). This is done by introducing an adaptive reward function that encourages the model to generate sufficient amount of accurate keyphrases. They also propose a new Wikipedia-based evaluation method that can more robustly evaluate the quality of the predicted keyphrases by also considering name variations of the ground truth keyphrases.

We are aware of one study that tackled keyword detection with transformers. Sahrawat *et al.* (2020) fed contextual embeddings generated using several transformer and recurrent architectures (BERT Devlin *et al.* 2019, RoBERTa Liu *et al.* 2019, GPT-2 Radford *et al.* 2019, ELMo Peters *et al.* 2018, etc.) into two distinct neural architectures, a bidirectional Long Short-Term Memory Network (BiLSTM) and a BiLSTM network with an additional conditional random fields layer (BiLSTM-CRF). Same as in Gollapalli *et al.* (2017), they formulate a keyword extraction task as a sequence labeling approach, in which each word in the document is assigned one of the three possible labels: k_b denotes that the word is the first word in a keyphrase, k_i means that the word is inside a keyphrase, and k_o indicates that the word is not part of a keyphrase.

The study shows that contextual embeddings generated by transformer architectures generally perform better than static (e.g., FastText embeddings Bojanowski *et al.* 2017) and among them, BERT showcases the best performance. Since all of the keyword detection experiments are conducted on scientific articles, they also test SciBERT (Beltagy, Lo, and Cohan 2019), a version of BERT pretrained on a large multi-domain corpus of scientific publications containing 1.14M papers sampled from Semantic Scholar. They observe that this genre-specific pretraining on texts of the same genre as the texts in the keyword datasets slightly improves the performance of the model. They also report significant gains in performance when the BiLSTM-CRF architecture is used instead of BiLSTM.

The neural sequence-to-sequence models are capable of outperforming all older supervised and unsupervised models by a large margin, but do require a very large training corpora with tens of thousands of documents for successful training. This means that their use is limited only to languages (and genres) in which large corpora with manually labeled keywords exist. On the other hand, the study by Sahrawat *et al.* (2020) indicates that the employment of contextual embeddings reduces the need for a large dataset with manually labeled keywords. These models can, therefore, be deployed directly on smaller datasets by leveraging semantic information already encoded in contextual embeddings.

2.2 Unsupervised keyword extraction methods

The previous section discussed recently emerged methods for keyword extraction that operate in a supervised learning setting and can be data-intensive and time consuming. Unsupervised keyword detectors can tackle these two problems, yet at the cost of the reduced overall performance.

Unsupervised approaches need no training and can be applied directly without relying on a gold-standard document collection. In general, they can be divided into four main categories, namely statistical, graph-based, embeddings-based, and language model-based methods:

- Statistical methods, such as KP-MINER (El-Beltagy and Rafea 2009), RAKE (Rose *et al.* 2010), and YAKE (Campos *et al.* 2018), use statistical characteristics of the texts to capture keywords. An extensive survey of these methods is presented in the study by Merrouni, Frikh, and Ouhbi (2020).
- Graph-based methods, such as TextRank (Mihalcea and Tarau 2004), Single Rank (Wan and Xiao 2008) and its extension ExpandRank (Wan and Xiao 2008), TopicRank (Bougouin, Boudin, and Daille 2013), Topical PageRank (Sterckx *et al.* 2015), KeyCluster (Liu *et al.* 2009), and RaKUn (Škrlj *et al.* 2019) build graphs to rank words based on their position in the graph. A survey by Merrouni *et al.* (2020) also offers good coverage of graph-based algorithms.
- Embedding-based methods such as the methods proposed by Wang, Liu, and McDonald (2015a), Key2Vec (Mahata *et al.* 2018), and EmbedRank (Bennani-Smires *et al.* 2018) employ semantic information from distributed word and sentence representations

(i.e., embeddings) for keyword extraction. Thes methods are covered in more detail in the survey by Papagiannopoulou and Tsoumakas (2020).

- Language model-based methods, such as the ones proposed by Tomokiyo and Hurst (2003) and Liu *et al.* (2011), on the other hand use language model-derived statistics to extract keywords from text. The methods are well covered in surveys by Papagiannopoulou and Tsoumakas (2020) and Çano and Bojar (2019).

Among the statistical approaches, the state-of-the-art keyword extraction algorithm is YAKE (Campos *et al.* 2018). It defines a set of features capturing keyword characteristics, which are heuristically combined to assign a single score to every keyword. These features include casing, position, frequency, relatedness to context, and dispersion of a specific term. Another recent statistical method proposed by Won, Martins, and Raimundo (2019) shows that it is possible to build a very competitive keyword extractor by using morpho-syntactic patterns for the extraction of candidate keyphrases and afterward employ simple textual statistical features (e.g., term frequency, inverse document frequency, position measures etc.) to calculate ranking scores for each candidate.

One of the first graph-based methods for keyword detection is TextRank (Mihalcea and Tarau 2004), which first extracts a lexical graph from text documents and then leverages Google's PageRank algorithm to rank vertices in the graph according to their importance inside a graph. This approach was somewhat upgraded by TopicRank (Bougouin *et al.* 2013), where candidate keywords are additionally clustered into topics and used as vertices in the graph. Keywords are detected by selecting a candidate from each of the top-ranked topics. Another method that employs PageRank is PositionRank (Florescu and Caragea 2017). Here, a word-level graph that incorporates positional information about each word occurence is constructed. One of the most recent graph-based keyword detectors is RaKUn (Škrlj *et al.* 2019) that employs several new techniques for graph construction and vertice ranking. First, the initial lexical graph is expanded and adapted with the introduction of meta-vertices, that is, aggregates of existing vertices. Second, for keyword detection and ranking, a graph-theoretic *load centrality* measure is used along with the implemented graph redundancy filters.

Besides employing PageRank on document's words and phrases, there are other options for building a graph. For example, in the CommunityCluster method proposed by Grineva, Grinev, and Lizorkin (2009), a single document is represented as a graph of semantic relations between terms that appear in that document. On the other hand, the CiteTextRank approach (Gollapalli and Caragea 2014), used for extraction of keywords from scientific articles, leverages additional contextual information derived from a citation network, in which a specific document is referenced. Finally, SGRank (Danesh, Sumner, and Martin 2015) and KeyphraseDS (Yang *et al.* 2017) methods belong to a family of the so-called hybrid statistical graph algorithms. SGRank ranks candidate keywords extracted from the text according to the set of statistical heuristics (position of the first occurrence, term length, etc.) and the produced ranking is fed into a graph-based algorithm, which conducts the final ranking. In the KeyphraseDS approach, keyword extraction consists of three steps: candidates are first extracted with a CRF model and a keyphrase graph is constructed from the candidates; spectral clustering, which takes into consideration knowledge and topic-based semantic relatedness, is conducted on the graph; and final candidates are selected through the integer linear programming (ILP) procedure, which considers semantic relevance and diversity of each candidate.

The first keyword extraction method that employed embeddings was proposed by Wang *et al.* (2015a). Here, a word graph is created, in which the edges have weights based on the word co-occurrence and the euclidean distance between word embeddings. A weighted PageRank algorithm is used to rank the words. This method is further improved in Wang, Liu, and McDonald (2015b), where a personalized weighted PageRank is employed together with the pretrained word embeddings. (Mahata *et al.* 2018) suggested further improvement to the approach by introducing domain-specific embeddings, which are trained on multiword candidate phrases extracted from corpus documents. Cosine distance is used to measure the distance between embeddings and a direct graph is constructed, in which candidate keyphrases are represented as vertices. The final ranking is derived by using a theme-weighted PageRank algorithm (Langville and Meyer 2004).

An intriguing embedding-based solution was proposed by Papagiannopoulou and Tsoumakas (2018). Their Reference Vector Algorithm (RVA) for keyword extraction employs the so-called local word embeddings, which are embeddings trained on the single document from which keywords need to be extracted.

Yet, another state-of-the-art embedding-based keyword extraction method is EmbedRank (Bennani-Smires *et al.* 2018). In the first step, candidate keyphrases are extracted according to to the part-of-speech (POS)-based pattern (phrases consisting of zero or more adjectives followed by one or more nouns). Sent2Vec embeddings (Pagliardini, Gupta, and Jaggi 2018) are used for representation of candidate phrases and documents in the same vector space. Each phrase is ranked according to the cosine distance between the candidate phrase and the embedding of the document in which it appears.

Language model-based keyword extraction algorithms are less common than other approaches. Tomokiyo and Hurst (2003) extracted keyphrases by employing several unigram and n-gram language models, and by measuring KL divergence (Vidyasagar 2010) between them. Two features are used in the system: phraseness, which measures if a given word sequence is a phrase, and informativeness, which measures how well a specific keyphrase captures the most important ideas in the document. Another interesting approach is the one proposed by Liu *et al.* (2011), which relies on the idea that keyphrasing can be considered as a type of translation, in which documents are translated into the language of keyphrases. Statistical machine translation word alignment techniques are used for the calculation of matching probabilities between words in the documents and keyphrases.

3. Methodology

This section presents the methodology of our approach. Section 3.1 presents the architecture of the neural model, Section 3.2 covers the transfer learning techniques used, Section 3.3 explains how the final fine-tuning phase of the keyword detection workflow is conducted, and Section 4.3 covers evaluation of the model.

3.1 Architecture

The model follows an architectural design of an original transformer encoder (Vaswani *et al.* 2017) and is shown in Figure 1(a). Same as in the GPT-2 architecture (Radford *et al.* 2019), the encoder consists of a normalization layer that is followed by a multi-head attention mechanism. A residual connection is employed around the attention mechanism, which is followed by another layer normalization. This is followed by the fully connected feedforward and dropout layers, around which another residual connection is employed.

For two distinct training phases, language model pretraining and fine-tuning, two distinct "heads" are added on top of the encoder, which is identical for both phases and therefore allows for the transfer of weights from the pretraining phase to the fine-tuning phase. The language model head predicts the probability for each word in the vocabulary that it appears at a specific position in the sequence and consists of a dropout layer and a feedforward layer, which returns the output matrix of size SL $*$ $|V|$, where SL stands for sequence length (i.e., a number of words in the input text) and $|V|$ stands for the vocabulary size. This is followed by the adaptive softmax layer (Grave *et al.* 2017) (see description below).

During fine-tuning, the language model head is replaced with a token classification head, in which we apply ReLu nonlinearity and dropout to the encoder output, and then feed the output

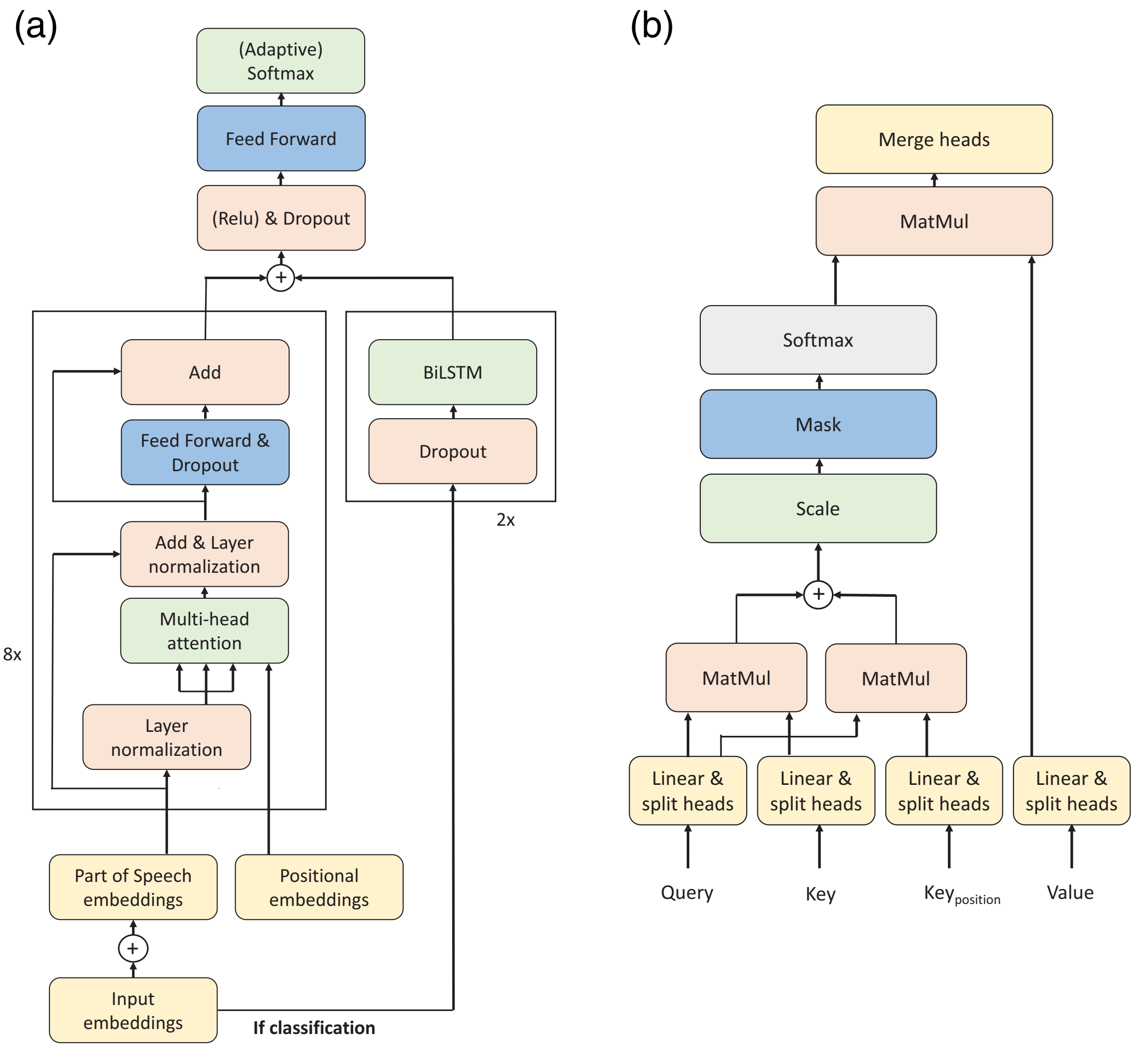


Figure 1. TNT-KID's architecture overview. (a) Model architecture. (b) The attention mechanism.

to the feedforward classification layer, which returns the output matrix of size SL ∗ NC, where NC stands for the number of classes (in our case 2, since we model keyword extraction as a binary classification task, see Section 3.3 for more details). Finally, a softmax layer is added in order to obtain probabilities for each class.

We also propose some significant modifications of the original GPT-2 architecture. First, we propose a re-parametrization of the attention mechanism (see Figure 1(b)). In the original transformer architecture, positional embedding is simply summed to the input embedding and fed to the encoder. While this allows the model to learn to attend by relative positions, the positional information is nevertheless fed to the attention mechanism in an indirect aggregated manner. On the other hand, we propose to feed the positional encoding to the attention mechanism directly, since we hypothesize that this would not only allow modeling of the relative positions between tokens but would also allow the model to better distinguish between the positional and semantic/grammatical information and therefore make it possible to assign attention to some tokens purely on the basis of their position in the text. The reason behind this modification is connected with the hypothesis that token position is especially important in the keyword identification task and with this re-parametrization the model would be capable of directly modeling the importance of relation between each token and each position. Note that we use relative positional embeddings for representing the positional information, same as in Dai *et al.* (2019), where the main idea is to only encode the relative positional information in the hidden states instead of the absolute.

Standard scaled dot-product attention (Vaswani *et al.* 2017) requires three inputs, a so-called *query, key, value* matrix representations of the embedded input sequence and its positional information (i.e., element wise addition of input embeddings and positional embeddings) and the idea is to obtain attention scores (in a shape of an attention matrix) for each relation between tokens inside these inputs by first multiplying *query* (Q) and transposed *key* (K) matrix representations, applying scaling and softmax functions, and finally multiplying the resulting normalized matrix with the *value* (V) matrix, or more formally,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k represents the scaling factor, usually corresponding to the first dimension of the *key* matrix. On the other hand, we propose to add an additional positional input representation matrix K_{position} and model attention with the following equation:

$$\text{Attention}(Q, K, V, K_{\text{pos}}) = \text{softmax}\left(\frac{QK^T + QK^T_{\text{position}}}{\sqrt{d_k}}\right)V$$

Second, besides the text input, we also experiment with the additional part-of-speech (POS) tag sequence as an input. This sequence is first embedded and then added to the word embedding matrix. Note that this additional input is optional and is not included in the model for which the results are presented in Section 4.4 due to marginal effect on the performance of the model in the proposed experimental setting (see Section 6).

The third modification involves replacing the standard input embedding layer and softmax function with adaptive input representations (Baevski and Auli 2019) and an adaptive softmax (Grave, Joulin, Cissé, Grangier and Jégou 2017). While the modifications presented above affect both training phases (i.e., the language model pretraining and the token classification fine-tuning), the third modification only affects the language model pretraining (see Section 3.2). The main idea is to exploit the unbalanced word distribution to form word clusters containing words with similar appearance probabilities. The entire vocabulary is split into a smaller cluster containing words that appear most frequently, a second (usually slightly bigger) cluster that contains words that appear less frequently and a third (and also optional fourth) cluster that contains all the other words that appear rarely in the corpus. During language model training, instead of predicting an entire vocabulary distribution at each time step, the model first tries to predict a cluster in which a target word appears in and after that predicts a vocabulary distribution just for the words in that cluster. Since in a large majority of cases, the target word belongs to the smallest cluster containing most frequent words, the model in most cases only needs to generate probability distribution for less than a tenth of a vocabulary, which drastically reduces the memory requirements and time complexity of the model at the expense of a marginal drop in performance.

We experiment with two tokenization schemes, word tokenization and Sentencepiece (Kudo and Richardson 2018) byte pair encoding (see Section 4 for details) and for these two schemes, we employ two distinct cluster distributions due to differences in vocabulary size. When word tokenization is employed, the vocabulary size tends to be bigger (e.g., reaching up to 600,000 tokens in our experiments on the news corpora), therefore in this setting, we employ four clusters, first one containing 20,000 most frequent words, second one containing 20,000 semi-frequent words, third one containing 160,000 less frequent words, and the fourth cluster containing the remaining least frequent words in the vocabulary.[b] When byte pair encoding is employed, the vocabulary is

[b]The proposed cluster distribution was derived from the distribution proposed by Dai *et al.* (2019), who limited the vocabulary size to about 260,000 tokens and proposed a three-cluster distribution with clusters of size 20,000, 40,000, and 200,000

notably smaller (i.e., containing about 32,000 tokens in all experiments) and the clustering procedure is strictly speaking no longer necessary. Nevertheless, since the initial experiments showed that the performance of the model does not worsen if the majority of byte pair tokens is kept in the first cluster, we still employ the clustering procedure in order to reduce the time complexity of the model, but nevertheless adapt the cluster distribution. We only apply three clusters: the first one contains 20,000 most frequent byte pairs, same as when word tokenization is employed; the second cluster is reduced to contain only 10,000 semi-frequent byte pairs; the third cluster contains only about 2000 least frequent byte pairs.

We also present the modification, which only affects the fine-tuning token classification phase (see Section 3.3). During this phase, a two layer randomly initialized encoder, consisting of dropout and two bidirectional Long Short-Term Memory (BiLSTM) layers, is added (with element-wise summation) to the output of the transformer encoder. The initial motivation behind this adaptation is connected with findings from the related work, which suggest that recurrent layers are quite successful at modeling positional importance of tokens in the keyword detection task (Meng *et al.* 2017; Yuan *et al.* 2020) and by the study of Sahrawat *et al.* (2020), who also reported good results when a BiLSTM classifier and contextual embeddings generated by transformer architectures were employed for keyword detection. Also, the results of the initial experiments suggested that some performance gains can in fact be achieved by employing this modification.

In terms of computational complexity, a self-attention layer complexity is $\mathcal{O}(n^2 * d)$ and the complexity of the recurrent layer is $\mathcal{O}(n * d^2)$, where n is the sequence length and d is the embedding size (Vaswani *et al.* 2017). This means that the complexity of the transformer model with an additional BiLSTM encoder is therefore $\mathcal{O}(n^2 * d^2)$. In terms of the number of operations required, the standard TNT-KID model encoder employs the sequence size of 512, embedding size of 512 and 8 attention layers, resulting in altogether $512^2 * 512 * 8 = 1,073,741,824$ operations. By adding the recurrent encoder with two recurrent bidirectional layers (which is the same as adding four recurrent layers, since each bidirectional layer contains two unidirectional LSTM layers), the number of operations increases by $512 * 512^2 * 4 = 536,870,912$. In practice, this means that the model with the additional recurrent encoder conducts token classification roughly 50% slower than the model without the encoder. Note that this addition does not affect the language model pretraining, which tends to be the more time demanding task due to larger corpora involved.

Finally, we also experiment with an employment of the BiLSTM-CRF classification head on top of the transformer encoder, same as in the approach proposed by Sahrawat *et al.* (2020) (see Section 6 for more details about the results of this experiment). For this experiment, during the fine-tuning token classification phase, the token classification head described above is replaced with a BiLSTM-CRF classification head proposed by Sahrawat *et al.* (2020), containing one BiLSTM layer and a CRF (Lafferty, McCallum, and Pereira 2001) layer.[c] Outputs of the BiLSTM $f = f_1, ..., f_n$ are fed as inputs to a CRF layer, which returns the output score $s(f,y)$ for each possible label sequence according to the following equation:

$$s(f, y) = \sum_{t=1}^{n} \tau_{y_{t-1},y_t} + f_{t,y_t}$$

tokens. To avoid limiting the vocabulary size, we added an additional cluster for very rare tokens. This is still in line with the recommendation by Grave *et al.* (2017), who proposed three–five clusters as an optimal trade-off between the improvement in time complexity and reduction in performance. Since the initial experiments suggested that the reduction in size of the second and third cluster does not hurt the performance of the model (in contrast, reduction of the first cluster does have a detrimental effect on the performance), but does slightly improve the time complexity, second cluster was reduced to 20,000 tokens and the third cluster to 160,000 tokens.

[c]Note that in the experiments in which we employ BiLSTM-CRF, we do not add an additional two layer BiLSTM encoder described above to the output of the transformer encoder.

τ_{y_{t-1},y_t} is a transition matrix representing the transition score from class y_{t-1} to y_t. The final probability of each label sequence score is generated by exponentiating the scores and normalizing over all possible output label sequences:

$$p(y|f) = \frac{exp(s(f, y))}{\sum_{y'} exp(s(f', y'))}$$

To find the optimal sequence of labels efficiently, the CRF layer uses the Viterbi algorithm (Forney 1973).

3.2 Transfer learning

Our approach relies on a transfer learning technique (Howard and Ruder 2018; Devlin *et al.* 2019), where a neural model is first pretrained as a language model on a large corpus. This model is then fine-tuned for each specific keyword detection task on each specific manually labeled corpus by adding and training the token classification head described in the previous section. With this approach, the syntactic and semantic knowledge of the pretrained language model is transferred and leveraged in the keyword detection task, improving the detection on datasets that are too small for the successful semantic and syntactic generalization of the neural model.

In the transfer learning scenario, two distinct pretraining objectives can be considered. First, is the autoregressive language modeling where the task can be formally defined as predicting a probability distribution of words from the fixed size vocabulary V, for word w_t, given the historical sequence $w_{1:t-1} = [w_1, ..., w_{t-1}]$. This pretraining regime was used in the GPT-2 model (Radford *et al.* 2019) that we modified. Since in the standard transformer architecture self-attention is applied to an entire surrounding context of a specific word (i.e., the words that appear after a specific word in each input sequence are also used in the self-attention calculation), we employ obfuscation masking to the right context of each word when the autoregressive language model objective is used, in order to restrict the information only to the prior words in the sentence (plus the word itself) and prevent target leakage (see Radford *et al.* (2019) for details on the masking procedure).

Another option is a masked language modeling objective, first proposed by Devlin *et al.* (2019). Here, a percentage of words from the input sequence is masked in advance, and the objective is to predict these masked words from an unmasked context. This allows the model to leverage both left and right context, or more formally, the token w_t is also determined by sequence of tokens $w_{t+1:n} = [w_{t+1}, ..., w_{t+n}]$. We follow the masking procedure described in the original paper by Devlin *et al.* (2019), where 15% of words are randomly designated as targets for prediction, out of which 80% are replaced by a masked token ($< mask >$), 10% are replaced by a random word and 10% remain intact.

The final output of the model is a softmax probability distribution calculated over the entire vocabulary, containing the predicted probabilities of appearance (P) for each word given its left (and in case of the *masked language modeling objective* also right) context. Training, therefore, consists of the minimization of the negative log loss (NLL) on the batches of training corpus word sequences by backpropagation through time:

$$\text{NLL} = -\sum_{i=1}^{n} \log P(w_i|w_{1:i-1}) \tag{1}$$

While the *masked language modeling* objective might outperform autoregressive language modeling objective in a setting where a large pretraining corpus is available (Devlin *et al.* 2019) due to the inclusion of the right context, these two training objectives have at least to our knowledge never been compared in a setting where only a relatively small domain-specific corpus is

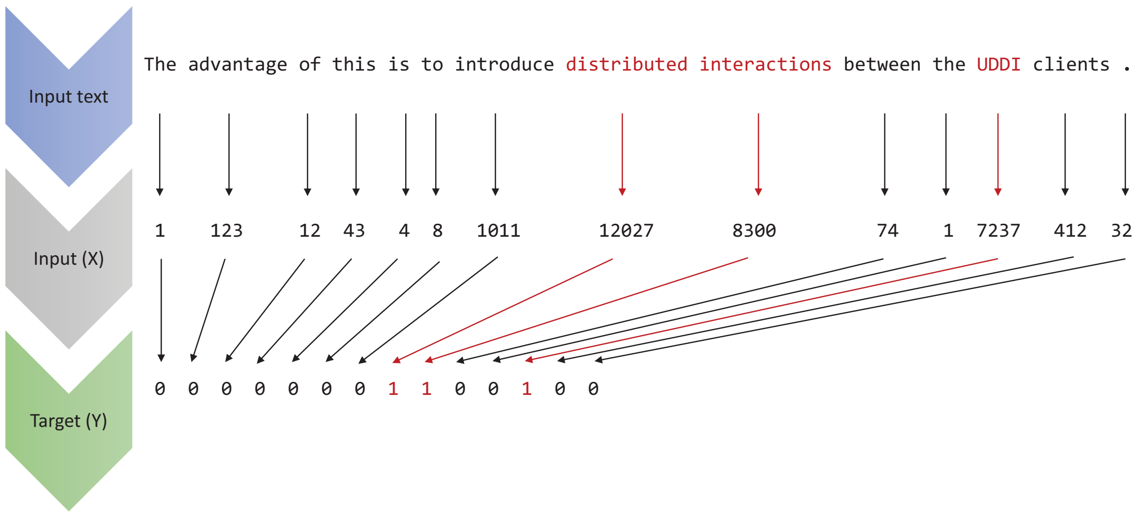


Figure 2. Encoding of the input text "*The advantage of this is to introduce distributed interactions between the UDDI clients.*" with keywords *distributed interactions* and *UDDI*. In the first step, the text is converted into a numerical sequence, which is used as an input to the model. The model is trained to convert this numerical sequence into a sequence of zeroes and ones, where the ones indicate the position of a keyword.

available for the pretraining phase. For more details about the performance comparison of these two pretraining objectives, see Section 6.

3.3 Keyword identification

Since each word in the sequence can either be a keyword (or at least part of the keyphrase) or not, the keyword tagging task can be modeled as a binary classification task, where the model is trained to predict if a word in the sequence is a keyword or not.[d] Figure 2 shows an example of how an input text is first transformed into a numerical sequence that is used as an input of the model, which is then trained to produce a sequence of zeroes and ones, where the positions of ones indicate the positions of keywords in the input text.

Since a large majority of words in the sequence are not keywords, the usage of a standard NLL function (see Equation (1)), which would simply calculate a sum of log probabilities that a word is either a keyword or not for every input word sequence, would badly affect the recall of the model since the majority negative class would prevail. To solve this problem and maximize the recall of the system, we propose a custom classification loss function, where probabilities for each word in the sequence are first aggregated into two distinct sets, one for each class. For example, text "*The advantage of this is to include distributed interactions between the UDDI clients.*" in Figure 2 would be split into two sets, the first one containing probabilities for all the words in the input example which are not keywords (*The, advantage, of, this, is, to, include, between, the, clients,.*), and the other containing probabilities for all the words in the input example that are keywords or part of keyphrases (*distributed, interactions, UDDI*). Two NLLs are calculated, one for each probability set, and both are normalized with the size of the set. Finally, the NLLs are summed. More formally, the loss is computed as follows. Let $W = \{w_i\}_{i=1}^{n}$ represent an enumerated sequence of tokens for which predictions are obtained. Let p_i represent the predicted probabilities for the ith token that it either belongs or does not belong to the ground truth class. The o_i represents the output weight vector of the neural network for token i and j corresponds to the number of classes (two in our

[d]Note that this differs from the sequence labeling approach proposed by Sahrawat *et al.* (2020), where each word in the document is assigned one of three possible labels (see Section 2 for details).

case as the word can be a keyword or not). Predictions are in this work obtained via a log-softmax transform (*first*), defined as follows (for the *i*th token):

$$p_i = \text{lst}(o_i) = \log \frac{\exp(o_i)}{\sum_j \exp(o_j)}.$$

The loss function is comprised from two main parts. Let $K_+ \subseteq W$ represent tokens that are keywords and $K_- \subseteq W$ the set of tokens that are **not** keywords. Note that $|K_- \cup K_+| = n$, that is, the two sets cover all considered tokens for which predictions are obtained. During loss computation, only the probabilities of the ground truth class are considered. We mark them with p_i^+ or p_i^-. Then the loss is computed as

$$L_+ = -\frac{1}{|K_+|} \sum_{w_i \in K_+} p_i^+ \quad \text{and} \quad L_- = -\frac{1}{|K_-|} \sum_{w_i \in K_-} p_i^-.$$

The final loss is finally computed as

$$\text{Loss} = L_+ + L_-.$$

Note that even though all predictions are given as an argument, the two parts of the loss address different token indices (i).

In order to produce final set of keywords for each document, tagged words are extracted from the text and duplicates are removed. Note that a sequence of ones is always interpreted as a multiword keyphrase and not as a combination of one-worded keywords (e.g., *distributed interactions* from Figure 2 is considered as a single multiword keyphrase and not as two distinct one word keywords). After that, the following filtering is conducted:

- If a keyphrase is longer than four words, it is discarded.
- Keywords containing punctuation (with the exception of dashes and apostrophes) are removed.
- The detected keyphrases are ranked and arranged according to the softmax probability assigned by the model in a descending order.

4. Experiments

We first present the datasets used in the experiments. This is followed by the experimental design, evaluation, and the results achieved by TNT-KID in comparison to the state of the art.

4.1 Keyword extraction datasets

Experiments were conducted on seven datasets from two distinct genres, scientific papers about computer science and news. The following datasets from the computer science domain are used:

- **KP20k (Meng *et al.* 2017)**: This dataset contains titles, abstracts, and keyphrases of 570,000 scientific articles from the field of computer science. The dataset is split into train set (530,000), validation set (20,000), and test set (20,000).
- **Inspec (Hulth 2003)**: The dataset contains 2000 abstracts of scientific journal papers in computer science collected between 1998 and 2002. Two sets of keywords are assigned to each document, the controlled keywords that appear in the Inspec thesaurus, and the uncontrolled keywords, which are assigned by the editors. Only uncontrolled keywords are used in the evaluation, same as by Meng *et al.* (2017), and the dataset is split into 500 test papers and 1500 train papers.

- **Krapivin (Krapivin, Autaeu, and Marchese 2009)**: This dataset contains 2304 full scientific papers from computer science domain published by ACM between 2003 and 2005 with author-assigned keyphrases. Four-hundred and sixty papers from the dataset are used as a test set and the others are used for training. Only titles and abstracts are used in our experiments.
- **NUS (Nguyen and Kan 2007)**: The dataset contains titles and abstracts of 211 scientific conference papers from the computer science domain and contains a set of keywords assigned by student volunters and a set of author-assigned keywords, which are both used in evaluation.
- **SemEval (Kim *et al.* 2010)**: The dataset used in the SemEval-2010 Task 5, Automatic Keyphrase Extraction from Scientific Articles, contains 244 articles from the computer science domain collected from the ACM Digital Library. One-hundred articles are used for testing and the rest are used for training. Again, only titles and abstracts are used in our experiments, the article's content was discarded.

From the news domain, three datasets with manually labeled gold-standard keywords are used:

- **KPTimes (Gallina, Boudin, and Daille 2019)**: The corpus contains 279,923 news articles containing editor-assigned keywords that were collected by crawling New York Times news website.[e] After that, the dataset was randomly divided into training (92.8%), development (3.6%) and test (3.6%) sets.
- **JPTimes (Gallina *et al.* 2019)**: Similar as **KPTimes**, the corpus was collected by crawling Japan Times online news portal.[f] The corpus only contains 10,000 English news articles and is used in our experiments as a test set for the classifiers trained on the **KPTimes** dataset.
- **DUC (Wan and Xiao 2008)**: The dataset consists of 308 English news articles and contains 2488 hand-labeled keyphrases.

The statistics about the datasets that are used for training and testing of our models are presented in Table 1. Note that there is a big variation in dataset sizes in terms of number of documents (column *No. docs*), and in an average number of keywords (column *Avg. kw.*) and present keywords per document (columns *Avg. present kw.*), ranging from 2.35 present keywords per document in *KPTimes-valid* to 7.79 in *DUC-test*.

4.2 Experimental design

We conducted experiments on the datasets described in Section 4.1. First, we lowercased and tokenized all datasets. We experimented with two tokenization schemes, word tokenization and Sentencepiece (Kudo and Richardson 2018) byte pair encoding (see Section 6 for more details on how these two tokenization schemes affect the overall performance). During both tokenization schemes, a special $< eos >$ token is used to indicate the end of each sentence. For the best-performing model, for which the results are presented in Section 4.4, byte pair encoding was used. For generating the additional POS tag sequence input described in Section 3.1, which was **not** used in the best-performing model, Averaged Perceptron Tagger from the NLTK library (Bird and Loper 2004) was used. The neural architecture was implemented in PyTorch (Paszke *et al.* 2019).

In the pretraining phase, two language models were trained for up to 10 epochs, one on the concatenation of all the texts from the computer science domain and the other on the concatenation

[e]https://www.nytimes.com.

[f]https://www.japantimes.co.jp.

Table 1. Datasets used for empirical evaluation of keyword extraction algorithms. *No.docs* stands for number of documents, *Avg. doc. length* stands for average document length in the corpus (in terms of number of words, that is, we split the text by white space), *Avg. kw.* stands for average number of keywords per document in the corpus, *% present kw.* stands for the percentage of keywords that appear in the corpus (i.e., percentage of document's keywords that appear in the text of the document), and *Avg. present kw.* stands for the average number of keywords per document that actually appear in the text of the specific document

Dataset	No. docs	Avg. doc. length	Avg. kw.	% present kw.	Avg. present kw.
Computer science papers					
KP20k-train	530,000	156.34	5.27	62.43	3.29
KP20k-valid	20,000	156.55	5.26	62.30	3.28
KP20k-test	20,000	156.52	5.26	62.55	3.29
Inspec-valid	1500	125.21	9.57	76.92	7.36
Inspec-test	500	121.82	9.83	78.14	7.68
Krapivin-valid	1844	156.65	5.24	54.34	2.85
Krapivin-test	460	157.76	5.74	55.66	3.20
NUS-test	211	164.80	11.66	50.47	5.89
SemEval-valid	144	166.86	15.67	45.43	7.12
SemEval-test	100	183.71	15.07	44.53	6.71
News articles					
KPTimes-train	259,923	783.32	5.03	47.30	2.38
KPTimes-valid	10,000	784.65	5.02	46.78	2.35
KPTimes-test	10,000	783.47	5.04	47.59	2.40
JPTimes-test	10,000	503.00	5.03	76.73	3.86
DUC-test	308	683.14	8.06	96.62	7.79

of all the texts from the news domain. Overall the language model train set for computer science domain contained around 87 million tokens and the news train set about 232 million tokens. These small sizes of the language model train sets enable relatively fast training and smaller model sizes (in terms of number of parameters) due to the reduced vocabulary.

After the pretraining phase, the trained language models were fine-tuned on each dataset's *validation* sets (see Table 1), which were randomly split into 80% of documents used for fine-tuning and 20% of documents used for hyperparameter optimization and test set model selection. The documents containing more than 512 tokens are truncated. Next, the documents are sorted according to the token length and split into batches. The documents in each batch are padded with a special $< pad >$ token to the length of the longest document in the batch. Each model was fine-tuned for a maximum of 10 epochs and after each epoch, the trained model was tested on the documents chosen for hyperparameter optimization and test set model selection. The model that showed the best performance (in terms of F1@10 score) was used for keyword detection on the test set. All combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and on average worked best for all the datasets in both domains[g]:

- Learning rates: 0.00005, 0.0001, **0.0003**, 0.0005, 0.001.
- Embedding size: 256, **512**.
- Number of attention heads: 4, **8**, 12.
- Sequence length: 128, 256, **512**.
- Number of attention layers: 4, **8**, 12.

Note that in our experiments, we use the same splits as in related work (Meng *et al.* 2019; Meng *et al.* 2017; Gallina *et al.* 2019) for all datasets with predefined splits (i.e., all datasets with train and validation sets, see Table 1). The exceptions are NUS, DUC and JPTimes datasets with no available predefined validation-test splits. For NUS and DUC, 10-fold cross-validation is used and the model used for keyword detection on the JPTimes-test dataset was fine-tuned on the KPTimes-valid dataset. Another thing to consider is that in the related work by Yuan *et al.* (2020), Meng *et al.* (2017), Gallina *et al.* (2019), Chen *et al.* (2018) and Ye and Wang (2018), to which we are comparing, large datasets KPTimes-train and KP20k-train with 530,000 documents and 260,000 documents, respectively, are used for the classification model training and these trained models are applied on all test sets from the matching domain. On the other hand, we do not train our classification models on these two large train sets but instead use smaller KPTimes-valid and KP20k-valid datasets for training, since we argue that, due to language model pretraining, fine-tuning the model on a relatively small labeled dataset is sufficient for the model to achieve competitive performance. We do however conduct the language model pretraining on the concatenation of all the texts from the computer science domain and the news domain as explained above, and these two corpora also contain texts from KPTimes-train and KP20k-train datasets.

4.3 Evaluation

To asses the performance of the model, we measure F1@*k* score, a harmonic mean between Precision@*k* and Recall@*k*.

In a ranking task, we are interested in precision at rank *k*. This means that only the keywords ranked equal to or better than *k* are considered and the rest are disregarded. Precision is the ratio of the number of correct keywords returned by the system divided by the number of all keywords returned by the system,[h] or more formally:

$$precision = \frac{|correct\ returned\ keywords@k|}{|returned\ keywords|}$$

Recall@*k* is the ratio of the number of correct keywords returned by the system and ranked equal to or better than *k* divided by the number of correct ground truth keywords:

$$recall = \frac{|correct\ returned\ keywords@k|}{|correct\ keywords|}$$

Due to the high variance of a number of ground truth keywords, this type of recall becomes problematic if *k* is smaller than the number of ground truth keywords, since it becomes impossible for the system to achieve a perfect recall. Similar can happen to precision@k, if the number of

[g]Note that the same set of hyperparameters are also used in the pretraining phase.

[h]Note that the number of returned keywords does not necessarily equal K for some of the systems used in our experiments, namely Semi-supervised CopyRNN, CopyRNN, CatSeqD, CorrRNN, GPT-2, GPT-2 with a BiLSTM-CRF classification head and TNT-KID.

keywords in a gold standard is lower than k, and the returned number of keywords is fixed at k. We shall discuss how this affects different keyword detection systems in Section 7.

Finally, we formally define F1@k as a harmonic mean between Precision@k and Recall@k:

$$F1@k = 2 * \frac{P@k * R@k}{P@k + R@k}$$

In order to compare the results of our approach to other state-of-the-art approaches, we use the same evaluation methodology as Yuan *et al.* (2020) and Meng *et al.* (2019), and measure F1@k with k being either 5 or 10. Note that F1@k is calculated as a harmonic mean of macro-averaged precision and recall, meaning that precision and recall scores for each document are averaged and the F1 score is calculated from these averages. Same as in the related work, lowercasing and stemming are performed on both the gold standard and the predicted keywords (keyphrases) during the evaluation and the predicted keyword is considered correct only if the stemmed and lowercased forms of predicted and gold-standard keywords exactly match (i.e., partial matches are considered incorrect). Only keywords that appear in the text of the documents (present keywords)[i] were used as a gold standard and the documents containing no present keywords were removed, in order to make the results of the conducted experiments comparable with the reported results from the related work.

As is pointed out in the study by Gallina, Boudin, and Daille (2020), evaluation and comparison of keyphrase extraction algorithms is not a trivial task, since keyphrase extraction models in different studies are evaluated under different, not directly comparable experimental setups. To make the comparison fair, they recommend the testing of the models on the same datasets, using identical gold-standard keyword sets and employing the same preprocessing techniques and parameter settings. We follow these guidelines strictly, when it comes to the use of identical datasets and gold-standard keyword sets, but somewhat deviate from them when it comes to the employment of identical preprocessing techniques and parameter settings employed for different approaches. Since all unsupervised approaches operate on a set of keyphrase candidates, extracted from the input document, Gallina *et al.* (2020) argues that the extraction of these candidates and other parameters should be identical (e.g., they select the sequences of adjacent nouns with one or more preceding adjectives of length up to five words in order to extract keyword candidates) for a fair comparison between algorithms. On the other hand, we are more interested in comparison between keyword extraction approaches instead of algorithms alone and argue that the distinct keyword candidate extraction techniques are inseparable from the overall approach and should arguably be optimized for each distinct algorithm. Therefore, we employ the original preprocessing proposed by the authors for each specific unsupervised approach and apply hyperparameters recommended by the authors. For the supervised approaches, we again employ preprocessing and parameter settings recommended by the authors (e.g., we employ word tokenization proposed by the authors of the systems for CopyRNN and CatSeqD, and employ GPT-specific byte pair tokenizer for GPT-2 and GPT-2 + BiLSTM-CRF approaches).

Instead of reimplementing each specific keyword extraction approach, we report results from the original studies whenever possible, that is, whenever the original results were reported for the same datasets, gold-standard keyword sets, and evaluation criteria, in order to avoid any possible biased decisions (e.g., the choice of hyperparameter settings not clearly defined in the original paper) and reimplementation mistakes. The results of the reimplementation are only reported for evaluation on datasets missing in the original studies and for algorithms with the publicly available code with clear usage instructions. If that is not the case, or if we were not able to obtain the source code from the original authors, the reimplementation was not attempted, since it is in most cases

[i]Note that scientific and news articles often list keywords that do not appear in the text of the article. For example, an NLP paper would often list "*Text mining*" as a keyword of the paper, even though the actual phrase does not appear in the text of the paper.

almost impossible to reimplement an algorithm accurately just by following the description in the paper (Repar, Martinc, and Pollak 2019).

4.4 Keyword extraction results and comparison to the state of the art

In Table 2, we present the results achieved by TNT-KID and a number of algorithms from the related work on the datasets presented in Table 1. Note that TfIdf, TextRank, YAKE, RaKUn, Key2Vec, and EmbedRank algorithms are unsupervised and do not require any training. KEA, Maui, GPT-2, GPT-2 + BiLSTM-CRF, and TNT-KID were trained on the different *validation* set for each of the datasets, and CopyRNN and CatSeqD were trained on the large KP20k-train dataset for keyword detection on computer science domain, and on the KPTimes-train dataset for keyword detection on the news domain, since they require a large train set for competitive performance. For two other CopyRNN variants, CorrRNN and Semi-supervised CopyRNN, we only report results on science datasets published in Chen *et al.* (2018) and Ye and Wang (2018) respectively, since the code for these two systems is not publicly available. The published results for CorrRNN were obtained by training the model on the KP20k-train dataset. On the other hand, Semi-supervised CopyRNN was trained on 40,000 labeled documents from the KP20k-train dataset and 400,000 documents without labels from the same dataset.

For RaKUn (Škrlj *et al.* 2019) and YAKE (Campos *et al.* 2020), we report results for default hyperparameter settings, since the authors of RaKUn, as well as YAKE's authors claim that a single hyperparameter set can offer sufficient performance across multiple datasets. We used the author's official github implementations[j] in the experiments. For Key2Vec (Mahata *et al.* 2018), we employ the github implementation of the algorithm [k] to generate results for all datasets, since the results in the original study are not comparable due to different set of keywords used (i.e., the keywords are not limited to only the ones that appear in text). Since the published code does not contain a script for the training of domain-specific embeddings trained on multiword candidate phrases, GloVe embeddings (Pennington, Socher, and Manning 2014) with the dimension of 50 are used instead. [l] The EmbedRank results in the original study (Bennani-Smires *et al.* 2018) are also not comparable (again, the keywords in the study are not limited to only the ones that appear in text); therefore, we once again use the official github implementation[m] of the approach to generate results for all datasets and employ the recommended Sent2Vec embeddings (Pagliardini *et al.* 2018) trained on English Wikipedia with the dimension of 700.

For KEA and Maui, we do not conduct additional testing on corpora for which results are not available in the related work (KPTimes, JPTimes, and DUC corpus) due to bad performance of the algorithms on all the corpora for which results are available. Finally, for TfIdf and TextRank, we report results from the related work where available (Yuan *et al.* 2020) and use the implementation of the algorithms from the Python Keyphrase Extraction (PKE) library[n] to generate unavailable results. Same as for RaKUn and YAKE, default hyperparameters are used.

For KEA, Maui, CopyRNN, and CatSeqD, we report results for the computer science domain published in Yuan *et al.* (2020) and for the news domain we report results for CopyRNN published in Gallina *et al.* (2019). The results that were not reported in the related work are results for CatSeqD on KPTimes, JPTimes, and DUC, since this model was originally not tested on these three datasets, and the F1@5 score results for CopyRNN on KPTimes and JPTimes. Again, the author's official github implementations[o] were used for training and testing of both models. The models were trained and tested on the large KPTimes-train dataset with a help of a script supplied

[j]https://github.com/SkBlaz/rakun and https://github.com/LIAAD/yake.
[k] https://github.com/MarkSecada/key2vec.
[l] Note that this might have significant impact on the results.
[m]https://github.com/swisscom/ai-research-keyphrase-extraction.
[n]https://github.com/boudinfl/pke.
[o]https://github.com/memray/OpenNMT-kpg-release.

Table 2. Empirical evaluation of state-of-the-art keyword extractors. Results marked with ∗ were obtained by our implementation or reimplementation of the algorithm and results without ∗ were reported in the related work

	KP20k	Inspec	Krapivin	NUS	SemEval	KPTimes	JPTimes	DUC	Average
				Unsupervised algorithms					
				TfIdf					
F1@5	0.072	0.160	0.067	0.112	0.088	0.179*	0.266*	0.098*	0.130
F1@10	0.094	0.244	0.093	0.140	0.147	0.151*	0.229*	0.120*	0.152
				TextRank					
F1@5	0.181	0.286	0.185	0.230	0.217	0.022*	0.012*	0.120*	0.157
F1@10	0.151	0.339	0.160	0.216	0.226	0.030*	0.026*	0.181*	0.166
				YAKE					
F1@5	0.141*	0.204*	0.215*	0.159*	0.151*	0.105*	0.109*	0.106*	0.149
F1@10	0.146*	0.223*	0.196*	0.196*	0.212*	0.118*	0.135*	0.132*	0.170
				RaKUn					
F1@5	0.177*	0.101*	0.127*	0.224*	0.167*	0.168*	0.225*	0.189*	0.172
F1@10	0.160*	0.108*	0.106*	0.193*	0.159*	0.139*	0.185*	0.172*	0.153
				Key2Vec					
F1@5	0.080*	0.121*	0.068*	0.109*	0.081*	0.126*	0.158*	0.062*	0.101
F1@10	0.090*	0.181*	0.082*	0.121*	0.126*	0.116*	0.145*	0.078*	0.117
				EmbedRank					
F1@5	0.135*	0.345*	0.149*	0.173*	0.189*	0.063*	0.081*	0.219*	0.169
F1@10	0.134*	0.394*	0.158*	0.190*	0.217*	0.057*	0.074*	0.246*	0.184
				Supervised algorithms					
				KEA					
F1@5	0.046	0.022	0.018	0.073	0.068	/	/	/	/
F1@10	0.044	0.022	0.017	0.071	0.065	/	/	/	/
				Maui					
F1@5	0.005	0.035	0.005	0.004	0.011	/	/	/	/
F1@10	0.005	0.046	0.007	0.006	0.014	/	/	/	/
				Semi-supervised CopyRNN					
F1@5	0.308	0.326	0.296	0.356	0.322	/	/	/	/
F1@10	0.245	0.334	0.240	0.320	0.294	/	/	/	/

Table 2. Continued

	KP20k	Inspec	Krapivin	NUS	SemEval	KPTimes	JPTimes	DUC	Average
					CopyRNN				
F1@5	0.317	0.244	0.305	0.376	0.318	0.406*	0.256*	0.083	0.288
F1@10	0.273	0.289	0.266	0.352	0.318	0.393	0.246	0.105	0.280
					CatSeqD				
F1@5	0.348	0.276	**0.325**	**0.374**	**0.327**	0.424*	0.238*	0.063*	0.297
F1@10	0.298	0.333	0.285	**0.366**	**0.352**	0.424*	0.238*	0.063*	0.295
					CorrRNN				
F1@5	/	/	0.318	0.361	0.320	/	/	/	/
F1@10	/	/	0.278	0.335	0.320	/	/	/	/
					GPT-2				
F1@5	0.275*	0.413*	0.253*	0.318*	0.257*	0.421*	0.331*	0.298*	0.321
F1@10	0.278*	0.469*	0.253*	0.323*	0.278*	0.423*	0.336*	0.312*	0.334
					GPT-2 + BiLSTM-CRF				
F1@5	**0.355***	**0.462***	0.287*	0.329*	0.246*	0.478*	**0.386***	**0.333***	0.360
F1@10	**0.360***	0.524*	0.288*	0.336*	0.274*	0.479*	**0.389***	0.371*	0.378
					TNT-KID				
F1@5	0.336*	0.460*	0.310*	0.350*	0.283*	**0.485***	0.359*	0.318*	**0.363**
F1@10	0.338*	**0.536***	**0.320***	0.358*	0.337*	**0.485***	0.361*	**0.373***	**0.389**

by the authors of the papers. Same hyperparameters that were used for KP20k training in the original papers (Meng *et al.* 2019; Yuan *et al.* 2020) were used.

We also report results for the unmodified pretrained GPT-2 (Radford *et al.* 2019) model with a standard feedforward token classification head, and a pretrained GPT-2 with a BiLSTM-CRF token classification head, as proposed in Sahrawat *et al.* (2020) and described in Section 3.1.[p] Note that a pretrained GPT-2 model with a BiLSTM-CRF token classification head in this experiment does not conduct binary classification, but rather employs the sequence labeling procedure from Sahrawat *et al.* (2020) described in Section 2, which assigns words in the text sequence into three classes. For the unmodified pretrained GPT-2 (Radford *et al.* 2019) model and a pretrained GPT-2 with a BiLSTM-CRF token classification head, we apply the same fine-tuning regime as for TNT-KID, that is we fine-tune the models for up to 10 epochs on each dataset's *validation* sets (see Table 1), which were randomly split into 80% of documents used for training and 20% of documents used for the test set model selection. The model that showed the best performance on this set of documents (in terms of F1@10 score) was used for keyword detection on the test set. We use the default hyperparameters (i.e., sequence length of 512, embedding size of 768, learning

[p]We use the implementation of GPT-2 from the Transformers library (https://github.com/huggingface/transformers) and use the Pytorch-crf library (https://pytorch-crf.readthedocs.io/en/stable/) for the implementation of the BiLSTM-CRF token classification head.

rate of 0.00003, 12 attention heads, and a batch size of 8) for both models and the original GPT-2 tokenization regime.

Overall, supervised neural network approaches drastically outperform all other approaches. Among them, TNT-KID performs the best on four datasets in terms of F1@10. It is outperformed by CatSeqD (on NUS and SemEval) or GPT-2+ BiLSTM-CRF (on JPTImes and DUC) on the other four datasets. CatSeqD also performs competitively on KP20k, Krapivin, and KPTimes datasets, but is outperformed by a large margin on three other datasets by both GPT-2 + BiLSTM-CRF and TNT-KID. To be more specific, in terms of F1@10, TNT-KID outperforms the CatSeqD approach by about 20% points on the Inspec dataset, on the DUC dataset, it outperforms CatSeqD by about 30% points, and on JPTimes it outperforms CatSeqD by about 12% points.

The results of CopyRNN, Semi-supervised CopyRNN, and CorrRNN are in a large majority of cases very consistent with CatSeqD. For example, CopyRNN performs slightly better than CatSeqD on DUC and JPTimes, and slightly worse on the other six datasets. Semi-supervised CopyRNN performs slightly worse than CopyRNN on the majority of datasets for which the results are available according to both criteria. On the other hand, CorrRNN slightly outperforms CopyRNN on two out of the three datasets for which the results are available according to both criteria, but is nevertheless still outperformed by CatSeqD on both of these datasets.

Results of TNT-KID are comparable to the results of GPT-2 + BiLSTM-CRF according to both criteria on a large majority of datasets. The difference is the biggest on the SemEval dataset, where the GPT-2 + BiLSTM-CRF is outperformed by TNT-KID by a margin of about 6% points in terms of F1@10. On the other hand, a GPT-2 model with a standard token classification head does perform less competitively on most datasets but still on average outperforms all non-transformer-based algorithms.

In terms of F1@5, GPT-2 + BiLSTM-CRF outperforms TNT-KID on four datasets (KP20k, Inspec, JPTimes, and DUC) and CatSeqD on three (Krapivin, NUS, and SemEval). Nevertheless, in terms of F1@5, TNT-KID offers consistently competitive performance on all datasets and on average still outperforms both of these algorithms. The performances of GPT-2 + BiLSTM-CRF and TNT-KID are comparable on most datasets, with TNT-KID outperforming GPT-2 + BiLSTM-CRF by a relatively small margin on four out of the eight datasets, and GPT-2 + BiLSTM-CRF outperforming TNT-KID on the other four. On average, the performance of these two algorithms in terms of F1@5 is almost identical, with TNT-KID outperforming GPT-2 + BiLSTM-CRF by a very small margin of 0.3% point.

The difference in performance between TNT-KID and the best-performing sequence-to-sequence generation approach towards keyword extraction, CatSeqD, can be partially explained by the difference in training regimes and the fact that our system was designed to maximize recall (see Section 3). Since our system generally detects more keywords than CatSeqD, it tends to achieve better recall, which offers a better performance when up to 10 keywords need to be predicted. On the other hand, a more conservative system that generally predicts less keywords tends to achieve a better precision, which positively affects the F1 score in a setting where only up to five keywords need to be predicted. This phenomenon will be analyzed in more detail in Section 5, where we also discuss the very low results achieved by CatSeqD on the DUC dataset.

When it comes to two other supervised approaches, KEA and Maui, they perform badly on all datasets they have been tested on and are outperformed by a large margin even by all unsupervised approaches. When we compare just unsupervised approaches, EmbedRank and TextRank achieve much better results than the other approaches according to both measures on the Inspec dataset. This is the dataset with the on average shortest documents. On the other hand, both of these algorithms perform uncompetitively in comparison to other unsupervised approaches on two datasets with much longer documents, KPTimes and JPTimes, where RaKUn and TfIdf are the best unsupervised approaches, respectively. Interestingly, EmbedRank and TextRank also achieve the highest F1@10 score out of all unsupervised keyword detectors on the DUC dataset, which

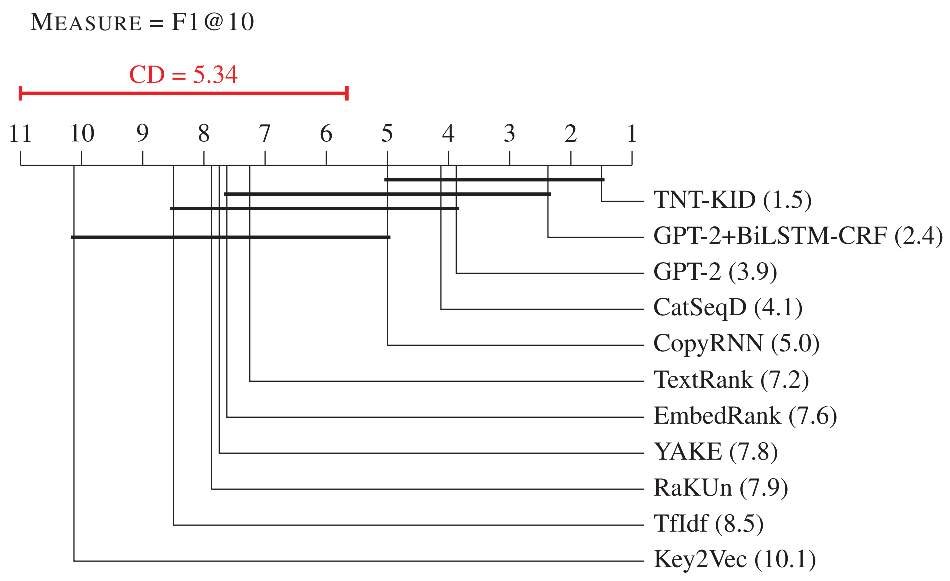


Figure 3. Critical distance diagram showing the results of the Nemenyi test. Two keyword extraction approaches are statistically significantly different in terms of F1@10 if a difference between their ranks (shown in brackets next to the keyword extraction approach name) is larger than the critical distance (CD). If two approaches are connected with a horizontal line, the test did not detect statistically significant difference between the approaches. For the Nemenyi test $\alpha = 0.05$ was used.

also contains long documents. Perhaps, this could be explained by the average number of present keywords, which is much higher for DUC-test (7.79) than for KPTimes-test (2.4) and JPTimes-test (3.86) datasets.

Overall (see row *average*), TNT-KID offers the most robust performance on the test datasets and is closely followed by GPT-2 + BiLSTM. CopyRNN and CatSeqD are very close to each other according to both criteria. Out of unsupervised approaches, on average all of them offer surprisingly similar performance. Even though graph-based and statistical approaches toward unsupervised keyword extraction are more popular than embedding-based approaches, the best overall performance in terms of F1@10 is offered by the embeddings-based approach EmbedRank. On the other hand, the other embedding-based method Key2Vec performs the worst out of all unsupervised approaches according to both criteria. According to the F1@10 score, the second ranked YAKE on average works slightly better than the third ranked TextRank and also in general offers more steady performance, since it gives the most consistent results on a variety of different datasets. Similar could be said for RaKUn, the best ranked unsupervised algorithm according to the F1@5 score.

Statistical comparison of classifiers over multiple datasets (according to the achieved F1@10 score) is conducted according to the procedure proposed in Demšar (2006), that is, with the Friedman test (Friedman 1937), and we were able to reject the null hypothesis, which states that there are no statistically significant differences between the tested keyword extraction approaches. This allowed us to proceed with the *post hoc* Nemenyi test (Nemenyi 1963) to find out which keyword extractors achieve statistically significantly different results. Note that only keyword extraction approaches employed on all the datasets are compared. The results are shown in Figure 3. We can see that the Nemenyi test has detected a significant difference in performance between TNT-KID and unsupervised keyword extractors (Key2Vec, TfIdf, RaKUn, Yake, EmbedRank, and TextRank), but was not strong enough to detect statistically significant differences between the five best supervised approaches.

Examples of the TNT-KID keyword detection are presented in the Appendix.

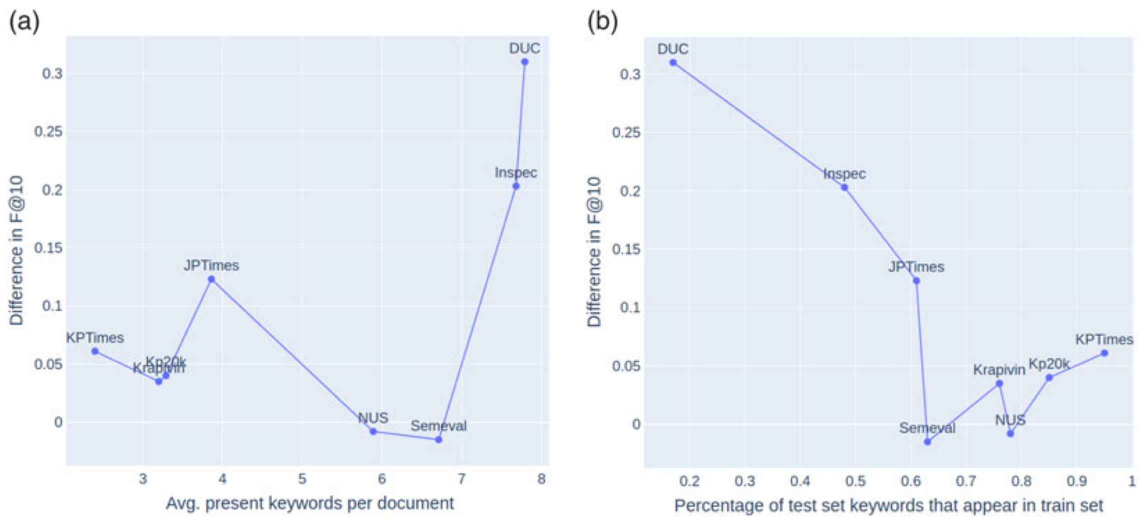


Figure 4. (a) Relation between the average number of present keywords per document for each test dataset and the difference in performance ($F1@10_{\text{TNT-KID}} - F1@10_{\text{CatSeqD}}$). (b) Relation between the percentage of keywords that appear in the train set for each test dataset and the difference in performance ($F1@10_{\text{TNT-KID}} - F1@10_{\text{CatSeqD}}$).

5. Error analysis

In this section, we first analyze the reasons why transformer-based TNT-KID is capable of outperforming other state-of-the-art neural keyword detectors, which employ a generative model, by a large margin on some of the datasets. Second, we gather some insights into the inner workings of the TNT-KID by a visual analysis of the attention mechanism.

5.1 Comparison between TNT-KID and CatSeqD

As was observed in Section 4.4, transformer-based TNT-KID and GPT-2 + BiLSTM-CRF outperform generative models CatSeqD and CopyRNN by a large margin on the Inspec, JPTimes, and DUC datasets. Here, we try to explain this discrepancy by focusing on the difference in performance between the best transformer-based model, TNT-KID, and the best generative model, CatSeqD. The first hypothesis is connected with the statistical properties of the datasets used for training and testing, or more specifically, with the average number of keywords per document for each dataset. Note that CatSeqD is trained on the KP20k-train, when employed on the computer science domain, and on the KPTimes-train dataset, when employed on news. Table 1 shows that both of these datasets do not contain many present keywords per document (KP20k-train 3.28 and KPTimes-train 2.38), therefore, training the model on these datasets conditions it to be conservative in its predictions and to assign less keywords to each document than a more liberal TNT-KID. This gives the TNT-KID a competitive advantage on the datasets with more present keywords per document.

Figure 4(a) shows a correlation between the average number of present keywords per document for each dataset and the difference in performance in terms of F1@10, measured as a difference between an F1@10 score achieved by TNT-KID and an F1@10 score achieved by CatSeqD. The difference in performance is the biggest for the DUC dataset (about 30% points) that on average has the most keywords per document, 7.79, and second biggest for Inspec, in which an average document has 7.68 present keywords.

The above hypothesis explains why CatSeqD offers competitive performance on the KP20k-test, Krapivin-test, NUS-test, and KPTimes-test datasets with similar number of keywords per document than its two train sets, but does not explain the competitive performance of CatSeqD on the SemEval-test set that has 6.71 present keywords per document. Even more importantly, it does not explain the large difference in performance between TNT-KID and CatSeqD on the JPTimes-test. This suggests that there is another factor influencing the performance of some keyword detectors.

The second hypothesis suggests that the difference in performance could be explained by the difference in training regimes and the different tactics used for keyword detection by the two systems. While TNT-KID is fine-tuned on each of the datasets, no fine-tuning is conducted for CatSeqD that needs to rely only on the information obtained during training on the large KP20k-train and KPTimes-train datasets. This information seems sufficient when CatSeqD is tested on datasets that contain similar keywords than the train sets. On the other hand, this training regime does not work for datasets that have less overlapping keywords.

Figure 4(b) supports this hypothesis by showing strong correlation between the difference in performance in terms of F1@10 and the percentage of keywords that appear both in the CatSeqD train sets (KP20k-train and KPTimes-train for computer science and news domain, respectively) and the test datasets. DUC and Inspec datasets have the smallest overlap, with only 17% of keywords in DUC appearing in the KPTimes-train and with 48% of keywords in Inspec appearing in the KP20k-train set. On the other hand, Krapivin, NUS, KP20k and KPTimes, the test sets on which CatSeqD performs more competitively, are the datasets with the biggest overlap, reaching up to 95% for KPTimes-test.

Figure 4(b) also explains a relatively bad performance of CatSeqD on the JPTimes corpus (see Table 2) despite the smaller average number of keywords per document. Interestingly, despite the fact that no dataset-specific fine-tuning for TNT-KID is conducted on the JPTimes corpus (since there is no validation set available, fine-tuning is conducted on the KPTimes-valid), TNT-KID manages to outperform CatSeqD on this dataset by about 13% points. This suggests that a smaller keyword overlap between train and test sets has less of an influence on the TNT-KID and could be explained with the fact, that CatSeqD considers keyword extraction as a generation task and tries to generate a correct keyword sequence, while TNT-KID only needs to tag an already existing word sequence, which is an easier problem that perhaps requires less specific information gained during training.

According to the Figure 4(b), the SemEval-test set is again somewhat of an outlier. Despite the keyword overlap that is quite similar to the one in the JPTimes-test set and despite having a relatively large set of present keywords per document, CatSeqD still performs competitively on this corpus. This points to a hypothesis that there might be another unidentified factor, either negatively influencing the performance of TNT-KID and positively influencing the performance of CatSeqD, or the other way around.

5.2 CatSeqD fine-tuning

According to the results in Section 4.4, supervised approaches to the keyword extraction task tend to outperform unsupervised approaches, most likely due to their ability to adapt to the specifics of the syntax, semantics and keyword labeling regime of the specific corpus. On the other hand, the main disadvantage of most supervised approaches is that they require a large dataset with labeled keywords for training, which are scarce at least in some languages. In this paper, we argue that the main advantage of the proposed TNT-KID approach is that due to its language model pretraining, the model only requires a small labeled dataset in order to fine-tune the language model for the keyword classification task. This fine-tuning allows the model to adapt to each dataset and leads to a better performance of TNT-KID in comparison to CatSeqD, for which no fine-tuning was conducted.

Even though no fine-tuning was conducted in the original CatSeqD study (Yuan *et al.* 2020), one might hypothesize that the performance of CatSeqD could be further improved if the model would be fine-tuned on each dataset, same as TNT-KID. To test this hypothesis, we take the

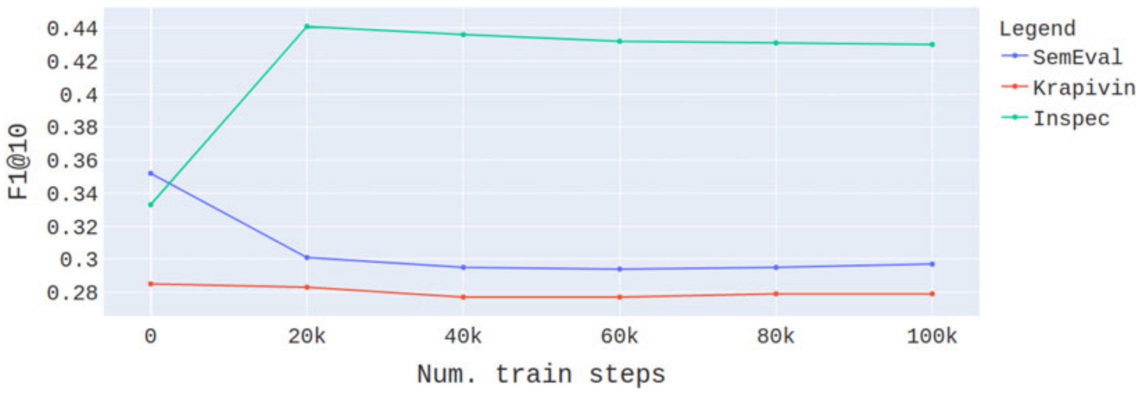


Figure 5. Performance of the KP20k trained CatSeqD model fine-tuned on SemEval, Krapivin and Inspec validation sets and tested on the corresponding test sets, in correlation with the length of the fine-tuning in terms of number of train steps. Zero train steps means that the model was not fine-tuned.

CatSeqD model trained on KP20k, conduct additional training on the SemEval, Krapivin and Inspec validation sets (i.e., all datasets besides KP20k and KPTimes with a validation set), and test these fine-tuned models on the corresponding test sets. Fine-tuning was conducted for up to 100,000 train steps[q] and the results are shown in Figure 5.

Only on one of the three datasets, the Inspec-test set, the performance can be improved by additional fine-tuning. Though the improvement on the Inspec-test set of about 10% points (from 33.5% to 44%) in terms of F1@10 is quite substantial, the model still performs worse than TNT-KID, which achieves F1@10 of 53.6%. The improvement is most likely connected with the fact that the Inspec-test set contains more keywords that do not appear in the KP20k than SemEval and Krapivin-test sets (see Figure 4(b)). Inspec-test set also contains more keywords per document than the other two test sets (7.68 present keywords on average, in comparison to 6.71 present keywords per document in the SemEval-test set and 3.2 in the Krapivin-test set). Since the KP20k train set on average contains only 3.29 present keywords per document, the fine-tuning on the Inspec dataset most likely also adapts the classifier to a more liberal keyword labeling regime.

On the other hand, fine-tuning does not improve the performance on the Krapivin and SemEval datasets. While there is no difference between the fine-tuned and original model on the Krapivin-test set, fine-tuning negatively affects the performance of the model on the SemEval dataset. The F1@10 score drops from about 35% to about 30% after 20,000 train steps. Further fine-tuning does not have any effect on the performance. The hypothesis is that this drop in performance is somewhat correlated with the size of the SemEval validation set, which is much smaller (it contains only 144 documents) than Inspec and Krapivin validation sets (containing 1500 and 1844 documents, respectively), and this causes the model to overfit. Further tests would, however, need to be conducted to confirm or deny this hypothesis.

Overall, 20,000 train steps seem to be enough for model adaptation in each case, since the results show that additional fine-tuning does not have any influence on the performance.

5.3 Dissecting the attention space

One of the advantages of the transformer architecture is its employment of the attention mechanism, that can be analyzed and visualized, offering valuable insights into inner workings of the system and enabling interpretation of how the neural net tackles the keyword identification task.

[q]Same hyperparameters that were used for KP20k training in the original paper (Yuan *et al.* 2020) were used for fine-tuning.

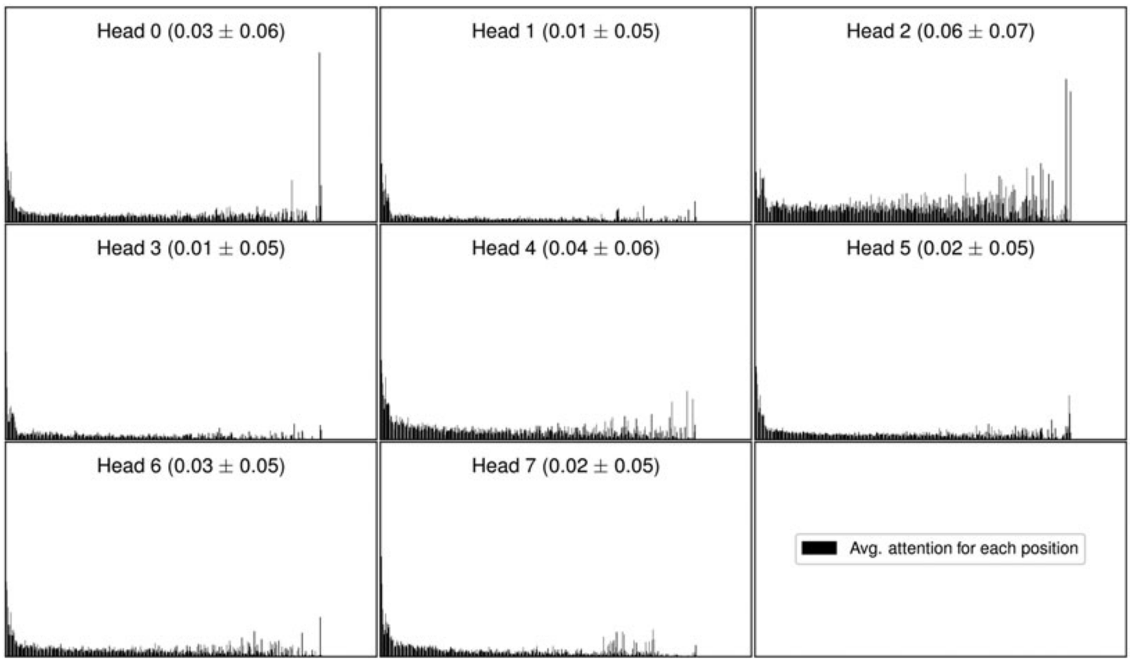


Figure 6. Average attention for each token position in the SemEval corpus across eight attention heads. Distinct peaks can be observed for tokens appearing at the beginning of the document in all eight attention heads.

The TNT-KID attention mechanism consists of multiple attention heads (Vaswani *et al.* 2017)—square matrices linking pairs of tokens within a given text—and we explored how this (activated) weight space can be further inspected via visualization and used for interpretation.

While square attention matrices show importance of the correlations between all tokens in the document for a keyword identification task, we focused only on the diagonals of the matrices, which indicate how much attention the model pays to the "correlation" a specific word has with itself, that is, how important is a specific word for the classification of a specific token as either being a keyword or not. We extracted these diagonal attention scores for eight attention heads of the last out of eight encoders, for each of the documents in the SemEval-test and averaged the scores across an entire dataset by summing together scores belonging to the same position in each head and dividing this sum with the number of documents. Figure 6 shows the average attention score of each of the eight attention heads for each token position. While there are differences between heads, a distinct peak at the beginning of the attention graph can be observed for all heads, which means that heads generally pay more attention to the tokens at the beginning of the document. This suggests that the system has learned that tokens appearing at the beginning of the document are more likely to be keywords (Figure 7 shows the actual keyword count for each position in the SemEval corpus) and once again shows the importance of positional information for the task of keyword identification.

Another insight into how the system works can be gained by analyzing how much attention was paid to each individual token in each document. Figure 8 displays attentions for individual tokens, as well as marks them based on predictions for an example document from the SemEval-test. Green tokens were correctly identified as keywords, red tokens were incorrectly identified as keywords, and less transparency (more color) indicates that a specific token received more attention from the classifier.

Figure 8 shows that at least for this specific document, many tokens that were either correctly or incorrectly classified as keywords did receive more attention than an average token, especially if they appeared at the beginning of the document. There are also some tokens that received a lot

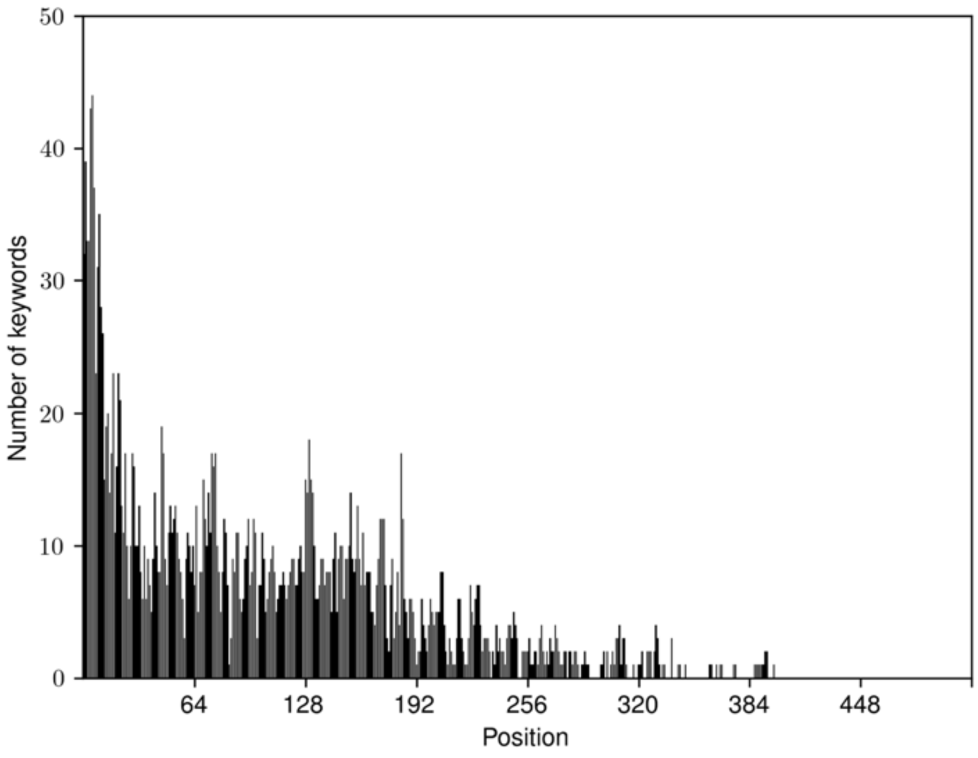


Figure 7. Number of keywords for each token position in the SemEval corpus. Distinct peaks can be observed for positions at the beginning of the document.

of attention and were not classified as keywords, for example, *eos* (end of sentence signs) and also words like *on, is, has, this, etc.* Another interesting thing to notice is the fact that the amount of attention associated with individual tokens that appear more than once in the document varies and is somewhat dependent on the position of the token.[r]

6. Ablation study

In this section, we explore the influence of several technique choices and building blocks of the keyword extraction workflow on the overall performance of the model:

- **Language model pretraining**: assessing whether pretraining positively affects the performance of the keyword extraction and if the improvements are dataset or domain specific.
- **Choice of pretraining regime**: comparison of two pretraining objectives, autoregressive language modeling and masked language modeling are described in Section 3.2.
- **Choice of input tokenization scheme**: comparison of two tokenization schemes, word tokenization and Sentencepiece (Kudo and Richardson 2018) byte pair encoding.
- **Part-of-speech(POS) tags**: assessment whether adding POS tags as an additional input improves the performance of the model.
- **Transformer architecture adaptations**: as was explained in Section 3.1, we propose a re-parametrization of the attention mechanism and in the fine-tuning stage, we add an additional BiLSTM encoder to the output of the transformer encoder. We also experiment with the addition of the BiLSTM+CRF token classification head on top of the model, as was proposed in

[r]Note that Figure 8 is just a motivating example. A more thorough statistical analysis of much more than just one document would be required in order to draw proper conclusions about the behavior of the attention mechanism during keyword identification.

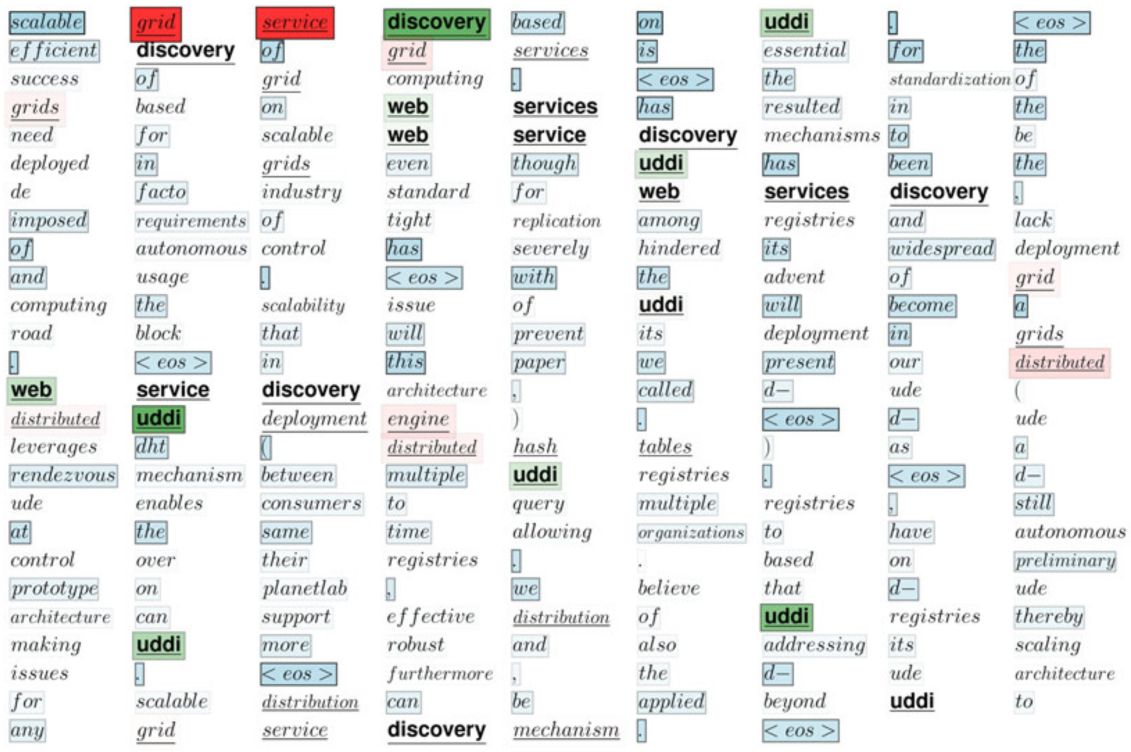

scalable efficient success grids need deployed de imposed of and computing road . web distributed leverages rendezvous ude at control prototype architecture making issues for any
grid discovery of based for in facto requirements autonomous usage the block < eos > service uddi dht mechanism enables the over on can uddi . scalable grid
service of grid on scalable grids industry of control . scalability that in discovery deployment (between consumers same their planetlab support more < eos > distribution service
discovery grid computing web web even standard tight has < eos > issue will this architecture engine distributed multiple to time registries . effective robust furthermore can discovery
based services . services service though for replication severely with of prevent paper .) hash uddi query allowing . we distribution and , be mechanism
on is < eos > has discovery uddi web among hindered the uddi its we called . tables registries multiple organizations . believe of also the applied .
uddi essential the resulted mechanisms has services registries its advent will deployment present d– < eos >) . registries to based that uddi addressing d– beyond < eos >
. for standardization in to been discovery and widespread of become in our ude d– as < eos > . have on d– registries its ude uddi
< eos > the of the be the . lack deployment grid a grids distributed (ude a d– still autonomous preliminary ude thereby scaling architecture to

Figure 8. Attention-colored tokens. Underlined phrases were identified as keywords by the system and bold font indicates that the identification was correct (i.e., the keyphrase appears in the gold standard). Less color transparency indicates stronger attention for the token and the color itself designates that the token was correctly identified as keyword (green), incorrectly identified as keyword (red) or was not identified as keyword by the system (blue).

Sahrawat *et al.* (2020) and described in Section 3.1. Here we assess the influence of these additions on the performance of the model.

Table 3 presents results on all datasets for several versions of the model, a model with no language model pretraining (*noLM*), a model pretrained with an autoregressive language model objective (*LM*), a model pretrained with a masked language model objective (*maskedLM*), a model pretrained with an autoregressive language model objective and leveraging byte pair encoding tokenization scheme (*LM+BPE*), a model pretrained with an autoregressive language model objective and leveraging additional POS tag sequence input (*LM+POS*), a model pretrained with an autoregressive language model objective and a BiLSTM encoder (*LM+BiLSTM*), a model pretrained with an autoregressive language model objective leveraging byte pair encoding tokenization scheme and a BiLSTM encoder, but without the proposed attention mechanism re-parametrization (*LM+BPE+BiLSTM+noAR*), a model pretrained with an autoregressive language model objective leveraging byte pair encoding tokenization scheme and a BiLSTM encoder (*LM+BPE+BiLSTM*), and a model pretrained with an autoregressive language model objective leveraging byte pair encoding tokenization scheme and a BiLSTM+CRF token classification head (*LM+BPE+BiLSTM+CRF*).

On average (see last two rows in Table 3), by far the biggest boost in performance is gained by employing the autoregressive language model pretraining (column *LM*), improving the F1@5 score by about 11% points and the F1@10 score by 12% points in comparison to no language model pretraining (column *noLM*). As expected, the improvements are substantial on two smallest corpora, which by themselves do not contain enough text for the model to obtain sufficient syntactic and semantic knowledge. Large gains are achieved on the NUS test set, where almost an 70% improvement in terms of the F1@10 score can be observed (from 20.98% to 35.59%), and

Table 3. Results of the ablation study. Column *LM+BPE+BiLSTM* represents the results for the model that was used for comparison with other methods from the related work in Section 4.4

	noLM	LM	maskedLM	LM+BPE	LM+POS	LM+BiLSTM	LM+BPE+ BiLSTM+noAR	LM+BPE+ BiLSTM	LM+BPE+ BiLSTM+CRF
					KP20k				
F1@5	0.2490	0.2914	0.2392	0.2919	0.2923	0.3203	0.3301	0.3355	**0.3398**
F1@10	0.2226	0.2876	0.2238	0.2921	0.2866	0.3174	0.3338	0.3378	**0.3443**
					Inspec				
F1@5	0.2833	0.4105	0.2850	0.4122	0.4171	0.4427	0.4389	**0.4595**	0.4514
F1@10	0.3528	0.4959	0.3610	0.5006	0.5028	0.5149	0.5091	**0.5356**	0.5169
					Krapivin				
F1@5	0.1922	0.2559	0.1757	0.2582	0.2611	0.2918	0.3058	**0.3097**	0.2874
F1@10	0.1837	0.2546	0.1859	0.2638	0.2583	0.3006	0.3145	**0.3202**	0.2882
					NUS				
F1@5	0.2034	0.3366	0.2194	0.3443	0.3135	0.3260	0.3319	0.3498	**0.3530**
F1@10	0.2098	0.3559	0.2479	0.3640	0.3481	0.3567	**0.3691**	0.3579	0.3602
					SemEval				
F1@5	0.1565	**0.3018**	0.1643	0.2954	0.2812	0.2957	0.2812	0.2825	0.2580
F1@10	0.2032	0.3374	0.2248	0.3251	**0.3399**	0.3351	0.3057	0.3365	0.3145
					KPTimes				
F1@5	0.2744	0.4433	0.2984	0.4302	0.4389	0.4830	0.4691	**0.4852**	0.4407
F1@10	0.2244	0.4409	0.2555	0.4281	0.4348	0.4810	0.4692	**0.4852**	0.4416
					JPTimes				
F1@5	0.2321	0.3393	0.2405	0.3315	0.3547	**0.3880**	0.3507	0.3590	0.3088
F1@10	0.2272	0.3430	0.2291	0.3349	0.3587	**0.3855**	0.3547	0.3613	0.3102
					DUC				
F1@5	0.2081	0.2751	0.1341	0.2831	0.2912	0.3135	0.2965	**0.3179**	0.2986
F1@10	0.2431	0.3391	0.1751	0.3323	0.3421	0.3706	0.3540	**0.3730**	0.3484
					Average				
F1@5	0.2249	0.3317	0.2196	0.3309	0.3312	0.3576	0.3505	**0.3624**	0.3422
F1@10	0.2334	0.3568	0.2379	0.3551	0.3589	0.3827	0.3763	**0.3884**	0.3655

on the SemEval-test set, where the improvement of 93% in terms of F1@5 can be observed. Not surprisingly, for the KP20k dataset, which has a relatively large validation set used for fine-tuning, we can observe a smaller improvement of about 29% in terms of F1@10. On the other hand, we observe the largest improvement of roughly 96% in terms of F1@10 on the KPTimes-test set, even though the KPTimes validation set used for fine-tuning is quite large. This means that in the language modeling phase the model still manages to obtain knowledge that is not reachable in the fine-tuning phase and can perhaps be partially explained by the fact that all documents are truncated into 512 tokens long sequences in the fine-tuning phase. The KPTimes-valid dataset, used both for language modeling and fine-tuning, has on average of 784.65 tokens per document, which means that more than a third of the document's text is discarded during the fine-tuning phase. This is not the case in the language modeling phase, where all of the text is leveraged.

On the other hand, using the masked language modeling pretraining (column *maskedLM*) objective on average yields a negligible improvement of about 0.5% points in terms of F1@10 score and worsening of about 0.5% points in terms of F1@10 score in comparison to no language model pretraining. It does, however, improve the performance on the two smallest datasets, NUS (by about 4% points in terms of F1@10) and SemEval (by about 2% points in terms of F1@10). More surprisingly, improvement is also substantial on the KPTimes dataset (about 3% points). The large discrepancy in performance between the two different language model objectives can be partially explained by the sizes of the pretraining corpora. By using autoregressive language modeling, the model learns to predict the next word probability distribution for each sequence in the corpus. By using the masked language modeling objective, 15% of the words in the corpus are randomly masked and used as targets for which the word probability distributions need to be predicted from the surrounding context. Even though each training epoch a different set of words is randomly masked, it is quite possible that some words are never masked due to small sizes of the corpora and since we only train the model for up to 10 epochs.

Results show that adding POS tags as an additional input (column *LM+POS*) leads to only marginal performance improvements. Some previous studies suggest that transformer-based models that employ transfer learning already capture sufficient amount of syntactic and other information about the composition of the text (Jawahar, Sagot, and Seddah 2019). Our results therefore support the hypothesis that additional POS tag inputs are somewhat unnecessary in the transfer learning setting but additional experiments would be needed to determine whether this is task/language specific or not.

Another adaptation that does not lead to any significant improvements when compared to the column *LM* is the usage of the byte pair encoding scheme (column *LM+BPE*). The initial hypothesis that motivated the usage of byte pair encoding was that it might help the model's performance by introducing some knowledge about the word composition and by enabling the model to better understand that different forms of the word can represent the same meaning. However, the usage of byte pair encoding might on the other hand also negatively affect the performance, since splitting up words inside a specific keyphrase would make these keyphrases longer in terms of number of words and detecting a longer continuous word sequence as a keyword might represent a harder problem for the model than detecting a shorter one. Nevertheless, usage of byte pair encoding does have an additional positive effect of drastically reducing the vocabulary of the model (e.g., for news articles, this means a reduction from almost 600,000 tokens to about 32,000) and with it also the number of parameters in the model (from about 630 million to about 80 million).

Adding an additional BiLSTM encoder in the fine-tuning stage of a pretrained model (column *LM+BiLSTM*) leads to consistent improvements on almost all datasets and to an average improvement of about 3% points in terms of both F1@5 and F1@10 scores. This confirms the findings from the related work that recurrent neural networks work well for the keyword detection task and also explains why a majority of state-of-the-art keyword detection systems leverage recurrent layers.

We also present a model in which we employed autoregressive language model pretraining, used byte pair encoding scheme and added a BiLSTM encoder (column *LM+BPE+BiLSTM*) that was used for comparison with other methods from the related work in Section 4.4, and the *LM+BPE+BiLSTM+noAR* model, which employs the same pretraining and tokenization regimes, and also has an added BiLSTM encoder, but was nevertheless not adapted for the keyword extraction task by the re-parametrization of the attention mechanism described in Section 3.1.

LM+BPE+BiLSTM outperforms the non-adapted model by a small, yet consistent margin on all but one dataset (on NUS *LM+BPE+BiLSTM+noAR* performs better in terms of F1@10) according to both criteria.

Finally, we also evaluate the tactic proposed by Sahrawat *et al.* (2020), where a BiLSTM+CRF token classification head is added on top of the transformer encoder, which employs the byte pair encoding scheme and autoregressive language model pretraining (column *LM+BPE+BiLSTM+CRF*). The BiLSTM+CRF performs quite well, outperforming all other configurations on two (i.e., on KP20k according to both measures and on NUS accroding to F1@5) datasets. On average, it, however, still performs by more than 2% points worse than LM+BPE+BiLSTM according to both measures. These results suggests that an additional CRF layer is not worth adding to the model when a binary sequence labeling regime is employed, but may nevertheless be useful when classification into more classes needs to be conducted, such as in the case of the labeling regime proposed by Sahrawat *et al.* (2020) described in Section 2.

7. Conclusion and future work

In this research we have presented TNT-KID, a novel transformer-based neural tagger for keyword identification that leverages a transfer learning approach to enable robust keyword identification on a number of datasets. The presented results show that the proposed model offers a robust performance across a variety of datasets with manually labeled keywords from two different domains. By exploring the differences in performance between our model and the best-performing generative model from the related work, CatSeqD by Yuan *et al.* (2020), we manage to pinpoint strengths and weaknesses of each keyword detection tactic (i.e., keyword labeling and keyword generation) and therefore enable a potential user to choose the approach most suitable for the task at hand. By visualizing the attention mechanism of the model, we try to interpret classification decisions of the neural network and show that efficient modeling of positional information is essential in the keyword detection task. Finally, we propose an ablation study which shows how specific components of the keyword extraction workflow influence the overall performance of the model.

The biggest advantage of supervised approaches to keyword extraction task is their ability to adapt to the specifics of the syntax, semantics, content, genre, and keyword tagging regime of the specific corpus. Our results show that this offers a significant performance boost and state-of-the-art supervised approaches outperform state-of-the-art unsupervised approaches on the majority of datasets. On the other hand, the ability of the supervised models to adapt might become limited in cases when the train dataset is not sufficiently similar to the dataset on which keyword detection needs to be performed. This can clearly be seen on the DUC dataset, in which only about 17% of keywords also appear in the KPTimes train set, used for training the generative CopyRNN and CatSeqD models. Here, these two state-of-the-art models perform the worst of all the models tested and as is shown in Section 5.2, this *keywordinees* generalization problem cannot be overcome by simply fine-tuning these state-of-the-art systems on each specific dataset.

On the other hand, TNT-KID bypasses the generalization problem by allowing fine-tuning on very small datasets. Nevertheless, the results on the JPTimes corpus suggest that it also generalizes better than CopyRNN and CatSeqD. Even though all three algorithms are trained on the KPTimes dataset (since JPTimes corpus does not have a validation set),[s] TNT-KID manages to outperform the other two by about 10% points according to the F1@10 and F1@5 criteria despite the discrepancy between train and test set keywords. As already mentioned in Section 5.1, this can be partially explained by the difference in approaches used by the models and the fact that keyword generation is a much harder task than keyword tagging. For keyword generation task to be successful, seeing a sequence that needs to be generated in advance, during training, is perhaps more important, than for a much simpler task of keyword tagging, where a model only needs to decide if

[s]Note that TNT-KID is trained on the validation set, while CopyRNN and CatSeqD are trained on the much larger train set.

a word is a keyword or not. Even though the keyword generators try to ease the task by employing a copying mechanism (Gu *et al.* 2016), the experiments suggest that generalizing *keywordinees* to unseen word sequences still represent a bigger challenge for these models than for TNT-KID.

While the conducted experiments suggest that TNT-KID works better than other neural networks in a setting where previously unseen keywords (i.e., keywords not present in the training set) need to be detected, further experiments need to be devised to evaluate the competitiveness of TNT-KID in a cross-domain setting when compared to unsupervised approaches. Therefore, in order to determine if the model's internal representation of *keywordiness* is general enough to be transferable across different domains, in the future we also plan to conduct some cross-domain experiments.

Another aspect worth mentioning is the evaluation regime and how it affects the comparison between the models. By fine-tuning the model on each dataset, the TNT-KID model learns the optimal number of keywords to predict for each specific dataset. This number is in general slightly above the average number of present keywords in the dataset, since the loss function was adapted to maximize recall (see Section 3). On the other hand, CatSeqD and CopyRNN are only trained on the KP20k-train and KPTimes-train datasets that have less present keywords than a majority of test datasets. This means our system on average predicts more keywords per document than these two systems, which negatively affects the precision of the proposed system in comparison to CatSeqD and CopyRNN, especially at smaller k values. On the other hand, predicting less keywords hurts recall, especially on datasets where documents have on average more keywords. As already mentioned in Section 6, this explains why our model compares better to other systems in terms of F1@10 than in terms of F1@5 and also raises a question how biased these measures of performance actually are. Therefore, in the future, we plan to use other performance measures to compare our model to others.

Overall, the differences in training and prediction regimes between TNT-KID and other neural models imply that the choice of a network is somewhat dependent on the use-case. If a large training dataset of an appropriate genre with manually labeled keywords is available and if the system does not need to predict many keywords, then CatSeqD might be the best choice, even though TNT-KID shows competitive performance on a large majority of datasets. On the other hand, if only a relatively small train set is available and it is preferable to predict a larger number of keywords, then the results of this study suggest that TNT-KID is most likely a better choice.

The conducted study also indicates that the adaptation of the transformer architecture and the training regime for the task at hand can lead to improvements in keyword detection. Both TNT-KID and a pretrained GPT-2 model with a BiLSTM + CRF token classification head manage to outperform the unmodified GPT-2 with a default token classification head by a comfortable margin. Even more, TNT-KID manages to outperform both, the pretrained GPT-2 and the GPT-2 with BiLSTM + CRF, even though it employs only 8 attention layers, 8 attention heads and an embedding size of 512 instead of the standard 12 attention layers, 12 attention heads and an embeddings size of 768, which the pretrained GPT-2 employs. The model on the other hand does employ an additional BiLSTM encoder during the classification phase, which makes it slower than the unmodified GPT-2 but still faster than the GPT-2 with the BiLSTM + CRF token classification head that employs a computationally demanding CRF layer.

The ablation study clearly shows that the employment of transfer learning is by far the biggest contributor to the overall performance of the system. Surprisingly, there is a very noticeable difference between performances of two distinct pretraining regimes, autoregressive language modeling and masked language modeling in the proposed setting with limited textual resources. Perhaps a masked language modeling objective regime could be improved by a more sophisticated masking strategy that would not just randomly mask 15% of the words but would employ a more fine-grained entity-level masking and phrase-level masking, similar as in Zhang *et al.* (2019). This and other pretraining learning objectives will be explored in the future work.

In the future, we also plan to expand the set of experiments in order to also cover other languages and domains. Since TNT-KID does not require a lot of manually labeled data for fine-tuning and only a relatively small domain-specific corpus for pretraining, the system is already fairly transferable to other languages and domains, even to low resource ones. It is especially useful for languages, for which pretrained transformers such as GPT-2, which also perform quite well on the keyword extraction task, do not yet exist. Deploying the system to a morphologically richer language than English and conducting an ablation study in that setting would also allow us to see, whether byte pair encoding and the additional POS tag sequence input would lead to bigger performance boosts on languages other than English.

Finally, another line of research we plan to investigate is a cross-lingual keyword detection. The idea is to pretrain the model on a multilingual corpus, fine-tune it on one language and then conduct zero-shot cross-lingual testing of the model on the second language. Achieving a satisfactory performance in this setting would make the model transferable even to languages with no manually labeled resources.

Acknowledgments. This paper is supported by European Union's Horizon 2020 research and innovation program under grant agreement No. 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). We also acknowledge the project Development of Slovene in a Digital Environment co-financed by the Republic of Slovenia and the European Union under the European Regional Development Fund—the project is being carried out under the Operational Programme for the Implementation of the EU Cohesion Policy in the period 2014–2020. The second author was financed via young research ARRS grant. Finally, the authors acknowledge the financial support from the Slovenian Research Agency for research core funding for the program Knowledge Technologies (No. P2-0103) and the project TermFrame—Terminology and Knowledge Frames across Languages (No. J6-9372).

Financial support. The authors were supported by the European Union's Horizon 2020 research and innovation program under grant agreement No. 825153 (project EMBEDDIA), project Development of Slovene in a Digital Environment co-financed by the Republic of Slovenia and the European Union under the European Regional Development Fund. The second author was financed via young research ARRS grant. Finally, the authors received the financial support from the Slovenian Research Agency for research core funding for the program Knowledge Technologies (No. P2-0103) and the projects TermFrame (No. J6-9372) and CANDAS (No. J6-2581).

References

Baevski A. and Auli M. (2019). Adaptive input representations for neural language modeling. In *7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA. OpenReview.net.

Beltagy I., Lo K. and Cohan A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics, pp. 3615–3620.

Bennani-Smires K., Musat C., Hossmann A., Baeriswyl M. and Jaggi M. (2018). Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium. Association for Computational Linguistics, pp. 221–229.

Bird S. and Loper E. (2004). NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, Barcelona, Spain. Association for Computational Linguistics, pp. 214–217.

Bojanowski P., Grave E., Joulin A. and Mikolov T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* **5**, 135–146.

Bougouin A., Boudin F. and Daille B. (2013). TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Nagoya, Japan. Asian Federation of Natural Language Processing, pp. 543–551.

Campos R., Mangaravite V., Pasquali A., Jorge A., Nunes C. and Jatowt A. (2020). Yake! keyword extraction from single documents using multiple local features. *Information Sciences* **509**, 257–289.

Campos R., Mangaravite V., Pasquali A., Jorge A.M., Nunes C. and Jatowt A. (2018). Yake! collection-independent automatic keyword extractor. In *European Conference on Information Retrieval*, Grenoble, France. Springer, pp. 806–810.

Çano E. and Bojar O. (2019). Keyphrase generation: A multi-aspect survey. In *2019 25th Conference of Open Innovations Association (FRUCT)*, Helsinki, Finland. IEEE, pp. 85–94.

Chan H.P., Chen W., Wang L. and King I. (2019). Neural keyphrase generation via reinforcement learning with adaptive rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics, pp. 2163–2174.

Chen J., Zhang X., Wu Y., Yan Z. and Li Z. (2018). Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics, pp. 4057–4066.

Dai Z., Yang Z., Yang Y., Carbonell J., Le Q. and Salakhutdinov R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics, pp. 2978–2988.

Danesh S., Sumner T. and Martin J.H. (2015). SGRank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, Denver, Colorado. Association for Computational Linguistics, pp. 117–126.

Demšar J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* **7**, 1–30.

Devlin J., Chang M.-W., Lee K. and Toutanova K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics, pp. 4171–4186.

El-Beltagy S.R. and Rafea A. (2009). Kp-miner: A keyphrase extraction system for english and arabic documents. *Information Systems* **34**(1), 132–144.

Firoozeh N., Nazarenko A., Alizon F. and Daille B. (2020). Keyword extraction: Issues and methods. *Natural Language Engineering* **26**(3), 259–291.

Florescu C. and Caragea C. (2017). PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada. Association for Computational Linguistics, pp. 1105–1115.

Forney G.D. (1973). The viterbi algorithm. *Proceedings of the IEEE* 61(3), 268–278.

Friedman M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association* **32**(200), 675–701.

Gallina Y., Boudin F. and Daille B. (2019). KPTimes: A large-scale dataset for keyphrase generation on news documents. In *Proceedings of the 12th International Conference on Natural Language Generation*, Tokyo, Japan. Association for Computational Linguistics, pp. 130–135.

Gallina Y., Boudin F. and Daille B. (2020). Large-scale evaluation of keyphrase extraction models. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, JCDL'20, New York, NY, USA. Association for Computing Machinery, pp. 271–278.

Goldberg Y. and Orwant J. (2013). A dataset of syntactic-ngrams over time from a very large corpus of English books. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, Atlanta, Georgia, USA. Association for Computational Linguistics, pp. 241–247.

Gollapalli S.D. and Caragea C. (2014). Extracting keyphrases from research papers using citation networks. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, Québec City, Québec, Canada. Association for Computing Machinery, pp. 1629–1635.

Gollapalli S.D., Li X.-L. and Yang P. (2017). Incorporating expert knowledge into keyphrase extraction. In *Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, California, USA. Association for Computing Machinery, pp. 3180–3187.

Grave E., Joulin A., Cissé M., Grangier D. and Jégou H. (2017). Efficient softmax approximation for gpus. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, Sydney, Australia. Association for Computing Machinery, pp. 1302–1310.

Grineva M., Grinev M. and Lizorkin D. (2009). Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th International Conference on World Wide Web*, Madrid, Spain. Association for Computing Machinery, pp. 661–670.

Gu J., Lu Z., Li H. and Li V.O. (2016). Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany. Association for Computational Linguistics, pp. 1631–1640.

Hasan K.S. and Ng V. (2014). Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, USA. Association for Computational Linguistics, pp. 1262–1273.

Howard J. and Ruder S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics, pp. 328–339.

Hulth A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP'03, Stroudsburg, PA, USA. Association for Computational Linguistics, pp. 216–223.

Jawahar G., **Sagot B. and Seddah D.** (2019). What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics, pp. 3651–3657.

Kim S.N., **Medelyan O.**, **Kan M.-Y. and Baldwin T.** (2010). Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval'10, Stroudsburg, PA, USA. Association for Computational Linguistics, pp. 21–26.

Krapivin M., **Autaeu A. and Marchese M.** (2009). Large dataset for keyphrases extraction. Technical report, University of Trento.

Kudo T. and Richardson J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations*, Brussels, Belgium. Association for Computational Linguistics, pp. 66–71.

Lafferty J.D., **McCallum A. and Pereira F.C.N.** (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML'01, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc., pp. 282–289.

Langville A.N. and Meyer C.D. (2004). Deeper inside pagerank. *Internet Mathematics* **1**(3), 335–380.

Liu Y., **Ott M.**, **Goyal N.**, **Du J.**, **Joshi M.**, **Chen D.**, **Levy O.**, **Lewis M.**, **Zettlemoyer L. and Stoyanov V.** (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Liu Z., **Chen X.**, **Zheng Y. and Sun M.** (2011). Automatic keyphrase extraction by bridging vocabulary gap. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, Portland, Oregon, USA. Association for Computational Linguistics, pp. 135–144.

Liu Z., **Li P.**, **Zheng Y. and Sun M.** (2009). Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore. Association for Computational Linguistics, pp. 257–266.

Luan Y., **Ostendorf M. and Hajishirzi H.** (2017). Scientific information extraction with semi-supervised neural tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics, pp. 2641–2651.

Mahata D., **Kuriakose J.**, **Shah R.R. and Zimmermann R.** (2018). Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, USA. Association for Computational Linguistics, pp. 634–639.

Medelyan O., **Frank E. and Witten I.H.** (2009). Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore. Association for Computational Linguistics, pp. 1318–1327.

Meng R., **Yuan X.**, **Wang T.**, **Brusilovsky P.**, **Trischler A. and He D.** (2019). Does order matter? an empirical study on generating multiple keyphrases as a sequence. arXiv preprint arXiv:1909.03590.

Meng R., **Zhao S.**, **Han S.**, **He D.**, **Brusilovsky P. and Chi Y.** (2017). Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada. Association for Computational Linguistics, pp. 582–592.

Merrouni Z.A., **Frikh B. and Ouhbi B.** (2020). Automatic keyphrase extraction: A survey and trends. *Journal of Intelligent Information Systems* **54**, 391–424.

Mihalcea R. and Tarau P. (2004). TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain. Association for Computational Linguistics, pp. 404–411.

Nemenyi P. (1963). *Distribution-Free Multiple Comparisons*. PhD Thesis, Princeton University.

Nguyen T.D. and Kan M.-Y. (2007). Keyphrase extraction in scientific publications. In *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, Berlin, Heidelberg: Springer, pp. 317–326.

Pagliardini M., **Gupta P. and Jaggi M.** (2018). Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana, USA. Association for Computational Linguistics, pp. 528–540.

Papagiannopoulou E. and Tsoumakas G. (2018). Local word vectors guiding keyphrase extraction. *Information Processing & Management* **54**(6), 888–902.

Papagiannopoulou E. and Tsoumakas G. (2020). A review of keyphrase extraction. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **10**(2), e1339.

Paszke A., **Gross S.**, **Massa F.**, **Lerer A.**, **Bradbury J.**, **Chanan G.**, **Killeen T.**, **Lin Z.**, **Gimelshein N.**, **Antiga L.**, **Desmaison A.**, **Kopf A.**, **Yang E.**, **DeVito Z.**, **Raison M.**, **Tejani A.**, **Chilamkurthy S.**, **Steiner B.**, **Fang L.**, **Bai J. and Chintala S.** (2019). Pytorch: An imperative style, high-performance deep learning library. *In Advances in Neural Information Processing Systems*, vol. **32**, Vancouver, Canada. Curran Associates, Inc., pp. 8024–8035.

Pennington J., **Socher R. and Manning C.** (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Association for Computational Linguistics, pp. 1532–1543.

Peters M., Neumann M., Iyyer M., Gardner M., Clark C., Lee K. and Zettlemoyer L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana. Association for Computational Linguistics, pp. 2227–2237.

Radford A., Wu J., Child R., Luan D., Amodei D. and Sutskever I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog* **1**(8).

Repar A., Martinc M. and Pollak S. (2019). Reproduction, replication, analysis and adaptation of a term alignment approach. *Language Resources and Evaluation* **54**, 767–800.

Rose S., Engel D., Cramer N. and Cowley W. (2010). Automatic keyword extraction from individual documents. In *Text Mining: Applications and Theory*, pp. 1–20.

Sahrawat D., Mahata D., Kulkarni M., Zhang H., Gosangi R., Stent A., Sharma A., Kumar Y., Shah R.R. and Zimmermann R. (2020). Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. In *Proceedings of European Conference on Information Retrieval (ECIR 2020)*, Lisbon, Portugal. Springer, pp. 328–335.

Škrlj B., Repar A. and Pollak S. (2019). RaKUn: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. In *International Conference on Statistical Language and Speech Processing*, Ljubljana, Slovenia. Springer, pp. 311–323.

Sterckx L., Demeester T., Deleu J. and Develder C. (2015). Topical word importance for fast keyphrase extraction. In *Proceedings of the 24th International Conference on World Wide Web*, New York, NY, USA. Association for Computing Machinery, pp. 121–122.

Tomokiyo T. and Hurst M. (2003). A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment - Volume 18*, Sapporo, Japan. Association for Computational Linguistics, pp. 33–40.

Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser Ł. and Polosukhin I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, Vancouver, Canada. Curran Associates, Inc., pp. 5998–6008.

Vidyasagar M. (2010). Kullback-leibler divergence rate between probability distributions on sets of different cardinalities. In *49th IEEE Conference on Decision and Control (CDC)*, Atlanta, Georgia, USA. IEEE, pp. 948–953.

Villmow J., Wrzalik M. and Krechel D. (2018). Automatic keyphrase extraction using recurrent neural networks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, New York, NY, USA. Springer, pp. 210–217.

Wan X. and Xiao J. (2008). Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, Chicago, Illinois, USA. AAAI Press, pp. 855–860.

Wang J., Peng H., and **Hu J.-s.** (2006). Automatic keyphrases extraction from document using neural network. In *Advances in Machine Learning and Cybernetics*, Guangzhou, China. Springer, pp. 633–641.

Wang R., Liu W. and McDonald C. (2015a). Corpus-independent generic keyphrase extraction using word embedding vectors. In *Proceedings of the Workshop on Deep Learning for Web Search and Data Mining (DL-WSDM)*, Shanghai, China. Association for Computing Machinery, pp. 39–46.

Wang R., Liu W. and McDonald C. (2015b). Using word embeddings to enhance keyword identification for scientific publications. In *Australasian Database Conference*, Melbourne, VIC, Australia. Springer, pp. 257–268.

Wang Y., Liu Q., Qin C., Xu T., Wang Y., Chen E. and Xiong H. (2018). Exploiting topic-based adversarial neural network for cross-domain keyphrase extraction. In *2018 IEEE International Conference on Data Mining (ICDM)*, Singapore. IEEE, pp. 597–606.

Witten I.H., Paynter G.W., Frank E., Gutwin C. and Nevill-Manning C.G. (1999). Kea: Practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, DL'99, Berkeley, California, USA. Association for Computing Machinery, pp. 254–255.

Won M., Martins B. and Raimundo F. (2019). Automatic extraction of relevant keyphrases for the study of issue competition. Technical report, EasyChair.

Yang S., Lu W., Yang D., Li X., Wu C. and Wei B. (2017). Keyphraseds: Automatic generation of survey by exploiting keyphrase information. *Neurocomputing* **224**, 58–70.

Ye H. and Wang L. (2018). Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics, pp. 4142–4153.

Yuan X., Wang T., Meng R., Thaker K., Brusilovsky P., He D. and Trischler A. (2020). One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics, pp. 7961–7975.

Zhang Z., Han X., Liu Z., Jiang X., Sun M. and Liu Q. (2019). ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy. Association for Computational Linguistics*, pp. 1441–1451.

Appendix: Examples of keyword identification

Document 1:

Quantum market games. We propose a quantum-like description of markets and economics. The approach has roots in the recently developed quantum game theory"

Predicted keywords: quantum market games, economics, quantum-like description

True keywords: economics, quantum market games, quantum game theory

Document 2:

Revenue Analysis of a Family of Ranking Rules for Keyword Auctions. Keyword auctions lie at the core of the business models of today's leading search engines. Advertisers bid for placement alongside search results, and are charged for clicks on their ads. Advertisers are typically ranked according to a score that takes into account their bids and potential click-through rates. We consider a family of ranking rules that contains those typically used to model Yahoo! and Google's auction designs as special cases. We find that in general neither of these is necessarily revenue-optimal in equilibrium, and that the choice of ranking rule can be guided by considering the correlation between bidders' values and click-through rates. We propose a simple approach to determine a revenue-optimal ranking rule within our family, taking into account effects on advertiser satisfaction and user experience. We illustrate the approach using Monte Carlo simulations based on distributions fitted to Yahoo! bid and click-through rate data for a high-volume keyword.

Predicted keywords: auction, keyword auctions, keyword, ranking rules, ranking, click through rates, click-through rates, revenue, advertiser, revenue analysis

True keywords: revenue optimal ranking, ranking rule, revenue, advertisement, keyword auction, search engine

Document 3:

Profile-driven instruction-level parallel scheduling with application to super blocks. Code scheduling to exploit instruction-level parallelism (ILP) is a critical problem in compiler optimization research in light of the increased use of long-instruction-word machines. Unfortunately, optimum scheduling is computationally intractable, and one must resort to carefully crafted heuristics in practice. If the scope of application of a scheduling heuristic is limited to basic blocks, considerable performance loss may be incurred at block boundaries. To overcome this obstacle, basic blocks can be coalesced across branches to form larger regions such as super blocks. In the literature, these regions are typically scheduled using algorithms that are either oblivious to profile information (under the assumption that the process of forming the region has fully utilized the profile information), or use the profile information as an addendum to classical scheduling techniques. We believe that even for the simple case of linear code regions such as super blocks, additional performance improvement can be gained by utilizing the profile information in scheduling as well. We propose a general paradigm for converting any profile-insensitive list scheduler to a profile-sensitive scheduler. Our technique is developed via a theoretical analysis of a simplified abstract model of the general problem of profile-driven scheduling over any acyclic code region, yielding a scoring measure for ranking branch instructions.

Predicted keywords: scheduling, profile-driven scheduling, instruction-level parallelism, profile-sensitive scheduler, instruction word machines

True keywords: long-instruction-word machines, scheduling heuristic, compiler optimization, optimum scheduling, abstract model, ranking branch instructions, profile-driven instruction-level parallel scheduling, profile-sensitive scheduler, linear code regions, code scheduling

Document 4:

Forty Years After War, Israel Weighs Remaining Risks. JERUSALEM. It was 1 p.m. on Saturday, 6 October 1973, the day of Yom Kippur, the holiest in the Jewish calendar, and Israel's military intelligence chief, Maj. Gen. Eli Zeira, had called in the country's top military journalists for an urgent briefing. He told us that war would break out at sundown, about 6 p.m., said Nachman Shai, who was then the military affairs correspondent for Israel's public television channel and is now a Labor member of Parliament. Forty minutes later he was handed a note and said, Gentlemen, the war broke out, and he left the room. Moments before that note arrived, according to someone else who was at that meeting, General Zeira had been carefully peeling almonds in a bowl of ice water. The coordinated attack by Egypt and Syria, which were bent on regaining strategic territories and pride lost to Israel in the 1967 war, surprised and traumatized Israel. For months, its leaders misread the signals and wrongly assumed that Israel's enemies were not ready to attack. Even in those final hours, when the signs were unmistakable that a conflict was imminent, Israel was misled by false intelligence about when it would start. As the country's military hurriedly called up its reserves and struggled for days to contain, then repel, the joint assault, a sense of doom spread through the country. Many feared a catastrophe. Forty years later, Israel is again marking Yom Kippur, which falls on Saturday, the anniversary of the 1973 war according to the Hebrew calendar. This year the holy day comes in the shadow of new regional tensions and a decision by the United States of America to opt, at least for now, for a diplomatic agreement rather than a military strike against Syria in response to a deadly chemical weapons attack in the Damascus suburbs on August 21. Israeli newspapers and television and radio programs have been filled with recollections of the 1973 war, even as the country's leaders have insisted that the probability of any new Israeli entanglement remains low and that the population should carry on as normal. For some people here, though, the echoes of the past have stirred latent questions about the reliability of intelligence assessments and the risks of another surprise attack. Any Israeli with a 40-year perspective will have doubts, said Mr. Shai, who was the military's chief spokesman during the Persian Gulf War of 1991, when Israelis huddled in sealed rooms and donned gas masks, shocked once again as Iraqi Scud missiles slammed into the heart of Tel Aviv. Coming after the euphoria of Israel's victory in the 1967 war, when 6 days of fighting against the Egyptian, Jordanian and Syrian Armies left Israel in control of the Sinai Peninsula, the West Bank, Gaza, East Jerusalem, and the Golan Heights, the conflicts of 1973, 1991, and later years have scarred the national psyche. But several former security officials and analysts said that while the risks now may be similar to those of past years in some respects, there are also major differences. In 1991, for example, the United States of America responded to the Iraqi attack by hastily redeploying some Patriot antimissile batteries to Israel from Europe, but the batteries failed to intercept a single Iraqi Scud, tracking them instead and following them to the ground with a thud. Since then, Israel, and the United States of America have invested billions of dollars in Israel's air defenses, with the Arrow, Patriot and Iron Dome systems now honed to intercept short-, medium-, and longer range rockets and missiles. Israelis, conditioned by subsequent conflicts with Hezbollah in Lebanon and Hamas in Gaza and by numerous domestic drills, have become accustomed to the wail of sirens and the idea of rocket attacks. But the country is less prepared for a major chemical attack, even though chemical weapons were used across its northern frontier, in Syria, less than a month ago, which led to a run on gas masks at distribution centers here. In what some people see as a new sign of government complacency at best and downright failure at worst, officials say there are enough protective kits for only 60% of the population, and supplies are dwindling fast. Israeli security assessments rate the probability of any attack on Israel as low, and the chances of a chemical attack as next to zero. In 1973, the failure of intelligence assessments about Egypt and Syria was twofold. They misjudged the countries' intentions and miscalculated their military capabilities. Our coverage of human intelligence, signals intelligence and other sorts was second to none, said Efraim Halevy, a former chief of Mossad, Israel's national intelligence agency. We thought we could initially contain any attack or repulse it within a couple of days. We wrongly assessed the capabilities of the Egyptians and the Syrians. In my opinion, that was the crucial failure. Israel is in a different situation today, Mr. Halevy said. The Syrian armed forces are depleted and focused on fighting their domestic battles, he said. The Egyptian Army is busy dealing with its internal turmoil, including a campaign against Islamic militants in Sinai. Hezbollah, the Lebanese militant group, is heavily involved in aiding President Bashar al-Assad of Syria, while the Iranians, Mr. Halevy said, are not likely to want to give Israel a reason to strike them, not as the aggressor but as a victim of an Iranian attack. Israel is also much less likely to suffer such a colossal failure in assessment, Mr. Halevy said. We have plurality in the intelligence community, and people have learned to speak up, he said. The danger of a mistaken concept is still there, because we are human. But it is much more remote than before. Many analysts have attributed the failure of 1973 to arrogance. There was a disregarding of intelligence, said Shlomo Avineri, a political scientist at Hebrew University and a director general of Israel's Ministry of Foreign Affairs in the mid-1970s. War is a maximization of uncertainties, he said, adding that things never happen the same way twice, and that wars never end the way they are expected to. Like most countries, Israel has been surprised by many events in recent years. The two Palestinian uprisings broke out unexpectedly, as did the Arab Spring and the two revolutions in Egypt. In 1973, logic said that Egypt and Syria would not attack, and for good reasons, said Ephraim Kam, a strategic intelligence expert at the Institute for National Security Studies at Tel Aviv University who served for more than 20 years in military intelligence. But there are always things we do not know. Intelligence is always partial, Mr. Kam said, its gaps filled by logic and assessment. The problem, he said, is that you cannot guarantee that the logic will fit with reality. In his recently published diaries from 1973, Uzi Eilam, a retired general, recalled the sounding of sirens at 2 p.m. on Yom Kippur and his rushing to the war headquarters. Eli Zeira passed me, pale-faced, he wrote, referring to the military intelligence chief, and he said: So it is starting after all. They are putting up planes. A fleeting glance told me that this was no longer the Eli Zeira who was so self-assured.

Predicted keywords: Israel, military, Syria, Egypt

True keywords: Israel, Yom Kippur, Egypt, Syria, military, Arab spring

Document 5:

Abe's 15-month reversal budget fudges cost of swapping people and butter for concrete and guns. The government of Shinzo Abe has just unveiled its budget for fiscal 2013 starting in April. Abe's stated intention was to radically reset spending priorities. He is indeed a man of his word. For this is a budget that is truly awesome for its radical step backward into the past a past where every public spending project would do wonders to boost economic growth. It is also a past where a cheaper yen would bring unmitigated benefits to Japan's exporting industries. None of it is really true anymore. Public works do indeed do wonders in boosting growth when there is nothing there to begin with. But in a mature and well-developed economy like ours, which is already so well equipped with all the necessities of modern life, they can at best have only a one-off effect in creating jobs and demand. And in this globalized day and age, an exporting industry imports almost as much as it exports. No longer do we live in a world where a carmaker makes everything within the borderlines of its nationality. Abe's radical reset has just as much to do with philosophy as with timelines. Three phrases come to mind as I try to put this budget in a nutshell. They are: from people to concrete, from the regions to the center, and from butter to guns. The previous government led by the Democratic Party of Japan declared that it would put people before concrete. No more building of ever-empty concert halls and useless multiple amenity centers where nothing ever happens. More money would be spent on helping people escape their economic difficulties. They would give more power to the regions so they could decide for themselves what was really good and worked for the local community. Guns would most certainly not take precedence over butter. Or rather over the low-fat butter alternatives popular in these more health-conscious times. All of this has been completely reversed in Abe's fiscal 2013 budget. Public works spending is scheduled to go up by more than 15% while subsistence payments for people on welfare will be thrashed to the tune of more than 7%. If implemented, this will be the largest cut ever in welfare assistance. The previous government set aside a lump sum to be transferred from the central government's coffers to regional municipalities to be spent at their own discretion on local projects. This sum will now be clawed back into the central government's own public works program. The planned increase in spending on guns is admittedly small: a 0.8% increase over the fiscal 2012 initial budget. It is nonetheless the first increase of its kind in 11 years. And given the thrashing being dealt to welfare spending, the shift in emphasis from butter to guns is clearly apparent. One of the Abe government's boasts is that it will manage to hold down the overall size of the budget in comparison with fiscal 2012. The other one is that it will raise more revenues from taxes rather than borrowing. True enough on the face of it. But one has to remember the very big supplementary budget that the government intends to push through for the remainder of fiscal 2012. The money for that program will come mostly from borrowing. Since the government is talking about a 15-month budget that seamlessly links up the fiscal 2012 supplementary and fiscal 2013 initial budgets, they should talk in the same vein about the size of their spending and the borrowing needed to accommodate the whole 15-month package. It will not do to smother the big reset with a big coverup.

Predicted keywords: Shinzo Abe, Japan, economy

True keywords: Shinzo Abe, budget

4.4 Combining Neural Keyword Extraction With Symbolic TF-IDF Based Keyword Extraction

As is shown in the previous section, neural state-of-the-art supervised approaches in most cases offer good performance in terms of precision. Nevertheless, the recall of these systems can be problematic for some use cases, since the system is trained to return a similar number of keywords per document as the average number of keywords per document in the train set. For example, in media house environments, automatic keyword extraction systems are used as a recommendation system, which is expected to return a diverse list of keyword candidates (preferably of constant length). This list is manually inspected by a journalist who selects the most suitable candidates. If the train set contains only a few keywords per document (e.g., the Croatian dataset described in Table 4.3 contains only 1.19 keywords per article), the list of candidates returned by the system will in most cases not be comprehensive enough for manual selection of best candidates.

We want to improve the recall of the existing neural supervised keyword extraction system by combining the output of the neural method with an output of the unsupervised symbolic keyword extraction technique. **We also investigate if this hybrid system achieves performance comparable to state-of-the-art according to several evaluation criteria and therefore explore whether the hypothesis H4 should be confirmed or denied.**

The unsupervised symbolic method we opted for is a TF-IDF tagset matching technique, which considers only words and multi-word expressions in the document that appear in a predefined tag/keyword set, ranks them according to their TF-IDF score, and returns the needed amount of best-ranked candidates. The new hybrid system checks the number of keywords returned by the neural approach and if the number is smaller than needed, the list is expanded by l best-ranked keyword candidates returned by the symbolic TF-IDF-based keyword extraction method, where l corresponds to k - m, $k = 10$ and m corresponds to the number of keywords returned by the neural TNT-KID method. According to the typology proposed in Chapter 1, we refer to this combination approach as a late simple fusion. This fusion type was chosen in this specific use case because there is a large discrepancy in performance between the neural and symbolic keyword extraction methods (see results in Table 4.5). The proposed approach is appropriate since it allows us to prioritize the neural method and only add the keywords returned by the TF-IDF-based keyword extraction method to the final keyword list when needed.

The experiments were conducted on four news corpora from the Embeddia project

Table 4.3: Datasets used for empirical evaluation of keyword extraction algorithms.

Language	Total docs	Doc. len	All keywords	% present kw.	present kw.
Train sets					
Croatian	32,223	438.50	3.54	0.32	1.19
Estonian	10,750	395.24	3.81	0.65	2.77
Russian	13,831	392.82	5.66	0.76	4.44
Latvian	13,133	378.03	3.23	0.53	1.69
Test sets					
Croatian	3582	464.39	3.53	0.34	1.26
Estonian	7,747	411.59	4.09	0.69	3.12
Russian	11,475	335.93	5.43	0.79	4.33
Latvian	11,641	460.15	3.19	0.55	1.71

Table 4.4: Distribution of tags provided per language. The media houses provided tagsets for Estonian and Russian, while the tags for Latvian and Croatian were extracted from the train set.

Dataset	**Unique tags**	**Type of tags**
Croatian	21,165	Constructed
Estonian	52,068	Provided
Russian	5,899	Provided
Latvian	4,015	Constructed

Table 4.5: Results on the multi-lingual news datasets.

Model	**P@5**	**R@5**	F_1**@5**	**P@10**	**R@10**	F_1**@10**
Croatian						
TF-IDF(tm)	0.2226	0.4543	0.2988	0.1466	0.5888	0.2347
TNT-KID	0.3296	0.5135	0.4015	0.3167	0.5359	0.3981
BERT + BiLSTM-CRF	**0.4607**	0.4672	**0.4640**	**0.4599**	0.4708	**0.4654**
TNT-KID & TF-IDF(tm)	0.2659	**0.5670**	0.3621	0.1688	**0.6944**	0.2716
Estonian						
TF-IDF(tm)	0.0716	0.1488	0.0966	0.0496	0.1950	0.0790
TNT-KID	**0.5194**	0.5676	**0.5424**	**0.5098**	0.5942	**0.5942**
BERT + BiLSTM-CRF	0.5118	0.4617	0.4855	0.5078	0.4775	0.4922
TNT-KID & TF-IDF(tm)	0.3463	**0.5997**	0.4391	0.1978	**0.6541**	0.3037
Russian						
TF-IDF(tm)	0.1764	0.2314	0.2002	0.1663	0.3350	0.2223
TNT-KID	**0.7108**	0.6007	**0.6512**	**0.7038**	0.6250	**0.6621**
BERT + BiLSTM-CRF	0.6901	0.5467	0.5467	0.6849	0.5643	0.6187
TNT-KID & TF-IDF(tm)	0.4519	**0.6293**	0.5261	0.2981	**0.6946**	0.4172
Latvian						
TF-IDF(tm)	0.2258	0.5035	0.3118	0.1708	0.5965	0.2655
TNT-KID	0.6089	0.6887	**0.6464**	0.6054	0.6960	**0.6476**
BERT + BiLSTM-CRF	**0.6215**	0.6214	0.6214	**0.6204**	0.6243	0.6223
TNT-KID & TF-IDF(tm)	0.3402	**0.7934**	0.4762	0.2253	**0.8653**	0.3575

(Latvian, Estonian, Russian, and Croatian)[3] described in Table 4.3, which contain news articles between 2015 and 2019 and were divided into training and test sets. The symbolic TF-IDF method was constrained to only return keyword candidates from four distinct (one per each language) tagsets containing keywords, which were either provided by the editors of a media house, or constructed from the keywords from the training set (see Table 4.4 for details).

We employ two neural methods for the experiments in this study, namely TNT-KID and BERT + BiLSTM-CRF proposed by Sahrawat et al. (2020). BERT + BiLSTM-CRF approach is very similar to the GPT-2 + BiLSTM-CRF approach described in Section 4.2, the only difference being that the English GPT-2 model is replaced with the multilingual BERT model (more specifically, the 'bert-base-multilingual-uncased' model was used) pretrained on a large corpus consisting of Wikipedias of about 100 languages (Devlin et al., 2019). TNT-KID on the other hand requires additional language model pretraining on the

[3]https://www.clarin.si/repository/xmlui/handle/11356/1403

domain-specific corpus and was therefore first trained as a language model on each training corpus. Both neural models were after that fine-tuned on the training corpus for a specific language for a maximum of up to 10 epochs and tested on the matching test corpus from the same language.

The results of the evaluation on all four languages are presented in Table 4.5. Namely, we present results for the symbolic TF-IDF tagset matching (tm) method, for two neural methods, TNT-KID and BERT+BiLSTM-CRF, and the combination of TNT-KID and TF-IDF(tm).

While the results for TNT-KID and BERT+BiLSTM-CRF neural methods are somewhat comparable, TNT-KID still outperforms BERT+BiLSTM-CRF on three out of four languages in terms of F_1@10. On the other hand, the symbolic TF-IDF tagset matching method for keyword extraction performs much worse than both neural methods according to all criteria. Nevertheless, if we combine the output of the TF-IDF tagset matching method with the output of TNT-KID, we can drastically improve the recall@5 and the recall@10 of the keyword extraction system. The improvement is substantial and consistent across all datasets. However, since the hybrid system always returns 10 keywords, which is much more than the average number of present gold standard keywords in the media partner datasets (see Table 4.3), this also means that the overall measured precision (and consequentially also the F_1 score) of the hybrid system is lower than for the neural systems. Nevertheless, we argue that the improvement in recall at the expanse of the precision is a good trade off if the system is intended to be used as a recommendation system in the media house environment, since it does not take much time for a journalist to manually inspect 10 keyword candidates per article and manually pick the best few candidates (e.g., by clicking on them).

The entire study with all the details about methodology, experiments and results is enclosed below.

Extending Neural Keyword Extraction with TF-IDF tagset matching

Boshko Koloski
Jožef Stefan Institute
Jožef Stefan IPS
Jamova 39, Ljubljana
boshko.koloski@ijs.si

Senja Pollak
Jožef Stefan Institute
Jamova 39, Ljubljana
senja.pollak@ijs.si

Blaž Škrlj
Jožef Stefan Institute
Jožef Stefan IPS
Jamova 39, Ljubljana
blaz.skrlj@ijs.si

Matej Martinc
Jožef Stefan Institute
Jamova 39, Ljubljana
matej.martinc@ijs.si

Abstract

Keyword extraction is the task of identifying words (or multi-word expressions) that best describe a given document and serve in news portals to link articles of similar topics. In this work, we develop and evaluate our methods on four novel data sets covering less-represented, morphologically-rich languages in European news media industry (Croatian, Estonian, Latvian, and Russian). First, we perform evaluation of two supervised neural transformer-based methods, Transformer-based Neural Tagger for Keyword Identification (TNT-KID) and Bidirectional Encoder Representations from Transformers (BERT) with an additional Bidirectional Long Short-Term Memory Conditional Random Fields (BiLSTM CRF) classification head, and compare them to a baseline Term Frequency - Inverse Document Frequency (TF-IDF) based unsupervised approach. Next, we show that by combining the keywords retrieved by both neural transformer-based methods and extending the final set of keywords with an unsupervised TF-IDF based technique, we can drastically improve the recall of the system, making it appropriate for usage as a recommendation system in the media house environment.

1 Introduction

Keywords are words (or multi-word expressions) that best describe the subject of a document, effectively summarise it and can also be used in several document categorization tasks. In online news portals, keywords help with efficient retrieval of articles when needed. Similar keywords characterise articles of similar topics, which can help editors to link related articles, journalists to find similar articles and readers to retrieve articles of interest when browsing the portals. For journalists manually assigning tags (keywords) to articles represents a demanding task, and high-quality automated keyword extraction shows to be one of components in news digitalization process that many media houses seek for.

The task of keyword extraction can generally be tackled in an unsupervised way, i.e., by relying on frequency based statistical measures (Campos et al., 2020) or graph statistics (Škrlj et al., 2019), or with a supervised keyword extraction tool, which requires a training set of sufficient size and from appropriate domain. While supervised methods tend to work better due to their ability to adapt to a specifics of the syntax, semantics, content, genre and keyword assignment regime of a specific text (Martinc et al., 2020a), their training for some less resource languages is problematic due to scarcity of large manually annotated resources. For this reason, studies about supervised keyword extraction conducted on less resourced languages are still very rare. To overcome this research gap, in this paper we focus on supervised keyword extraction on three less resourced languages, Croatian, Latvian, and Estonian, and one fairly well resourced language (Russian) and conduct experiments on data sets of media partners in the EMBEDDIA project[1]. The code for the experiments is made available on GitHub under the MIT license[2].

In media house environments, automatic keyword extraction systems are expected to return a diverse list of keyword candidates (of constant length), which is then inspected by a journalist who

[1]http://embeddia.eu/

[2]https://github.com/bkolosk1/Extending-Neural-Keyword-Extraction-with-TF-IDF-tagset-matching/

manually selects appropriate candidates. While the state-of-the-art supervised approaches in most cases offer good enough precision for this type of usage as a recommendation system, the recall of these systems is nevertheless problematic. Supervised systems learn how many keywords should be returned for each news article on the gold standard train set, which generally contains only a small amount of manually approved candidates for each news article. For example, among the datasets used in our experiments (see Section 3), the Russian train set contains the most (on average 4.44) present keywords (i.e., keywords which appear in the text of the article and can be used for training of the supervised models) per article, while the Croatian test set contains only 1.19 keywords per article. This means that for Croatian, the model will learn to return around 1.19 keywords for each article, which is not enough.

To solve this problem we show that we can improve the recall of the existing supervised keyword extraction system by:

- Proposing an additional TF-IDF tagset matching technique, which finds additional keyword candidates by ranking the words in the news article that have appeared in the predefined keyword set containing words from the gold standard train set. The new hybrid system first checks how many keywords were returned by the supervised approach and if the number is smaller than needed, the list is expanded by the best ranked keywords returned by the TF-IDF based extraction system.

- Combining the outputs of several state-of-the-art supervised keyword extraction approaches.

The rest of this work is structured as follows: Section 2 presents the related work, while Section 3 describes the datasets on which we evaluate our method. Section 4 describes our proposed method with all corresponding steps. The experiment settings are described in Section 5 and the evaluation of the proposed methods is shown in Section 6. The conclusions and the proposed further work are presented in Section 7.

2 Related Work

Many different approaches have been developed to tackle the problem of extracting keywords. The early approaches, such as KP-MINER (El-Beltagy and Rafea, 2009) and RAKE (Rose et al., 2010) rely on unsupervised techniques which employ frequency based metrics for extraction of keywords from text. Formally, aforementioned approaches search for the words w from vocabulary $\mathcal{V}$ that maximize a given metric h for a given text t:

$$\mathrm{kw} = \underset{w \in \mathcal{V}}{\operatorname{argmax}}\, h(w,t).$$

In these approaches, frequency is of high relevance and it is assumed that the more frequent a given word, the more important the meaning this word carries for a given document. Most popular such metrics are the naïve frequency (word count) and the term frequency-inverse document frequency (TF-IDF) (Salton and McGill, 1986).

Most recent state-of-the-art statistical approaches, such as YAKE (Campos et al., 2020), also employ frequency based features, but combine them with other features such as casing, position, relatedness to context and dispersion of a specific term in order to derive a final score for each keyword candidate.

Another line of research models this problem by exploiting concepts from graph theory. Approaches, such as TextRank (Mihalcea and Tarau, 2004), Single Rank (Wan and Xiao, 2008), TopicRank (Bougouin et al., 2013) and Topical PageRank (Sterckx et al., 2015) build a graph G, i.e., a mathematical construct described by a set of vertexes V and a set of edges E connecting two vertices. In one of the most recent approaches called RaKUn (Škrlj et al., 2019), a directed graph is constructed from text, where vertexes V and two words w_i, w_{i+1} are linked if they appear following one another. Keywords are ranked by a shortest path-based metric from graph theory - the load centrality.

The task of keyword extraction can also be tackled in a supervised way. One of the first supervised approaches was an algorithm named KEA (Witten et al., 2005), which uses only TF-IDF and the term's position in the text as features for term identification. More recent neural approaches to keyword detection consider the problem as a sequence-to-sequence generation task (Meng et al., 2017) and employ a generative model for keyword prediction with a recurrent encoder-decoder framework and an attention mechanism capable of detecting keywords in the input text sequence whilst also potentially finding keywords that do not appear in the text.

Finally, the newest branch of models consider keyword extraction as a sequence labelling task and tackle keyword detection with transformers. Sahrawat et al. (2020) fed contextual embeddings generated by several transformer models (BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), GPT-2 (Radford et al., 2019), etc.) into two types of neural architectures, a bidirectional Long short-term memory network (BiLSTM) and a BiLSTM network with an additional Conditional random fields layer (BiLSTM-CRF). Sun et al. (2020) on the other hand proposed BERT-JointKPE that employs a chunking network to identify phrases and a ranking network to learn their salience in the document. By training BERT jointly on the chunking and ranking tasks the model manages to establish balance between the estimation of keyphrase quality and salience.

Another state-of-the-art transformer based approach is TNT-KID (Transformer-based Neural Tagger for Keyword Identification) (Martinc et al., 2020a), which does not rely on pretrained language models such as BERT, but rather allows the user to train their own language model on the appropriate domain. The study shows that smaller unlabelled domain specific corpora can be successfully used for unsupervised pretraining, which makes the proposed approach easily transferable to low-resource languages. It also proposes several modifications to the transformer architecture in order to adapt it for a keyword extraction task and improve performance of the model.

3 Data Description

We conducted experiments on datasets containing news in four languages; Latvian, Estonian, Russian, and Croatian. Latvian, Estonian and Russian datasets contain news from the Ekspress Group, specifically from Estonian Ekspress Meedia (news in Estonian and Russian) and from Latvian Delfi (news in Latvian and Russian). The dataset statistics are presented in Table 2, and the datasets (Pollak et al., 2021) and their train/test splits[3] are publicly available. The media-houses provided news articles from 2015 up to the 2019. We divided them into training and test sets. For the Latvian, Estonian, and Russian training sets, we used the articles from 2018, while for the test set the articles from 2019 were used. For Croatian, the articles from 2019 are arranged by date and split into training and test (i.e., about 10% of the 2019 articles with the most recent date) set. In our study, we also use tagsets of keywords. Tagset corresponds either to a collection of keywords maintained by editors of a media house (see e.g. Estonian tagset), or to a tagset constructed from assigned keywords from articles available in the training set. The type of tagset and the number of unique tags for each language are listed in Table 1.

Dataset	Unique tags	Type of tags
Croatian	21,165	Constructed
Estonian	52,068	Provided
Russian	5,899	Provided
Latvian	4,015	Constructed

Table 1: Distribution of tags provided per language. The media houses provided tagsets for Estonian and Russian, while the tags for Latvian and Croatian were extracted from the train set.

4 Methodology

The recent supervised neural methods are very precise, but, as was already mentioned in Section 1, in same cases they do not return a sufficient number of keywords. This is due to the fact that the methods are trained on the training data with a low number of gold standard keywords (as it can be seen from Table 2). To meet the media partners' needs, we designed a method that complements state-of-the-art neural methods (the TNT-KID method (Martinc et al., 2020b) and the transformer-based method proposed by Sahrawat et al. (2020), which are both described in Section 2) by a tagset matching approach, returning constant number of keywords (k=10).

4.1 Transformer-based Keyword Extraction

Both supervised neural approaches employed in this study are based on the Transformer architecture (Vaswani et al., 2017), which was somewhat adapted for the specific task at hand. Both models are fed lowercased text consisting of the title and the body of the article. Tokenization is conducted by either using the default BERT tokenizer (when BERT is used) or by employing Sentencepiece tokenizer (Kudo and Richardson, 2018) (when TNT-KID is used). While the multilingual BERT model is already pretrained on a large corpus consisting of Wikipedias of about 100 languages (Devlin et al.,

[3]https://www.clarin.si/repository/xmlui/handle/11356/1403

			Avg. Train					Avg. Test				
Dataset	**Total docs**	**Total kw.**	**Total docs**	**Doc len**	**Kw.**	**% present kw.**	**present kw.**	**Total docs**	**Doc len**	**Kw.**	**% present kw.**	**Present kw.**
Croatian	35,805	126,684	32,223	438.50	3.54	0.32	1.19	3582	464.39	3.53	0.34	1.26
Estonian	18,497	59,242	10,750	395.24	3.81	0.65	2.77	7,747	411.59	4.09	0.69	3.12
Russian	25,306	5,953	13,831	392.82	5.66	0.76	4.44	11,475	335.93	5.43	0.79	4.33
Latvian	24,774	4,036	13,133	378.03	3.23	0.53	1.69	11,641	460.15	3.19	0.55	1.71

Table 2: Media partners' datasets used for empirical evaluation of keyword extraction algorithms.

2018), TNT-KID requires an additional language model pretraining on the domain specific corpus.

4.2 TF-IDF(tm) Tagset Matching

In our approach, we first take the keywords returned by a neural keyword extraction method and next complement the returned keyword list by adding the missing keywords to achieve the set goal of k keywords. The added keywords are selected by taking the top-ranked candidates from the TF-IDF tagset matching extraction conducted on the preprocessed news articles and keywords.

4.2.1 Preprocessing

First, we concatenate the body and the title of the article. After that we lowercase the text and remove stopwords. Finally, the text is tokenized and lemmatized with the Lemmagen3 lemmatizer (Juršič et al., 2010), which supports lemmatization for all the languages except Latvian. For Latvian we use the LatvianStemmer [4]. For the stopword removal we used the *Stopwords-ISO* [5] Python library which contained stopwords for all four languages. The final cleaned textual input consists of the concatenation of all of the preprocessed words from the document. We apply the same preprocessing procedure on the predetermined tagsets for each language. The preprocessing procedure is visualized in Figure 1.

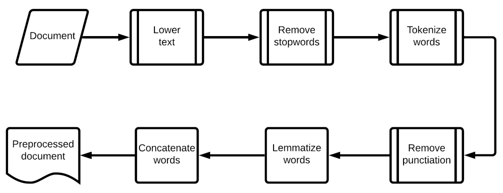


Figure 1: Preprocessing pipeline used for the document normalization and cleaning.

[4]https://github.com/rihardsk/LatvianStemmer
[5]https://github.com/stopwords-iso

4.2.2 TF-IDF Weighting Scheme

The TF-IDF weighting scheme (Salton and McGill, 1986) assigns each word its weight w based on the frequency of the word in the document (term frequency) and the number of documents the word appears in (inverse document frequency). More specifically, TF-IDF is calculated with the following equation:

$$TF-IDF_{\text{i}} = tf_{\text{i,j}} \cdot \log_e(\frac{|D|}{df_i})$$

The formula has two main components:

- *Term-frequency* (tf) that counts the number of appearances of a word in the document (in the equation above, $tf_{\text{i,j}}$ denotes the number of occurrences of the word i in the document j)
- *Inverse-document-frequency* (idf) ensures that words appearing in more documents are assigned lower weights (in the formula above df_i is the number of documents containing word i and $|D|$ denotes the number of documents).

The assumption is that words with a higher TF-IDF value are more likely to be keywords.

4.3 Tagset Matching Keyword Expansion

For a given neural keyword extraction method *N*, and for each document *d*, we select *l* best ranked keywords according to the TF-IDF(tm), which appear in the keyword tagset for each specific dataset. Here, *l* corresponds to *k* - *m*, where $k = 10$ and *m* corresponds to the number of keywords returned by a neural method.

Since some of the keywords in the tagsets provided by the media partners were variations of the same root word (i.e., keywords are not lemmatized), we created a mapping from a root word (i.e., a word lemma or a stem) to a list of possible variations in the keyword dataset. For example, a word *'riigieksam'* (*'exam'*) appearing in the article, could be mapped to three tags in the tagset by the Estonian media house with the same root form *'riigieksam'*: *'riigieksamid'*, *'riigieksamide'* and *'riigieksam'*.

We tested several strategies for mapping the occurrence of a word in the news article to a specific tag in the tagset. For each lemma that mapped to multiple tags, we tested returning a random tag, a tag with minimal length and a tag of maximal length. In the final version, we opted to return the tag with the minimal length, since this tag corresponded to the lemma of the word most often.

5 Experimental Settings

We conducted experiments on the datasets described in Section 3. We evaluate the following methods and combinations of methods:

- **TF-IDF(tm):** Here, we employ the preprocessing and TF-IDF-based weighting of keywords described in Section 4 and select the top-ranked keywords that are present in the tagset.
- **TNT-KID** (Martinc et al., 2020b): For each dataset, we first pretrain the model with an autoregressive language model objective. After that, the model is fine-tuned on the same train set for the keyword extraction task. Sequence length was set to 256, embedding size to 512 and batch size to 8, and we employ the same preprocessing as in the original study (Martinc et al., 2020b).
- **BERT + BiLSTM-CRF** (Sahrawat et al., 2020): We employ an uncased multilingual BERT[6] model with an embedding size of 768 and 12 attention heads, with an additional BiLSTM-CRF token classification head, same as in Sahrawat et al. (2020).
- **TNT-KID & BERT + BiLSTM-CRF**: We extracted keywords with both of the methods and complemented the TNT-KID extracted keywords with the BERT + BiLSTM-CRF extracted keywords in order to retrieve more keywords. Duplicates (i.e., keywords extracted by both methods) are removed.
- **TNT-KID & TF-IDF**: If the keyword set extracted by TNT-KID contains less than 10 keywords, it is expanded with keywords retrieved with the proposed TF-IDF(tm) approach, i.e., best ranked keywords according to TF-IDF, which do not appear in the keyword set extracted by TNT-KID.
- **BERT + BiLSTM-CRF & TF-IDF**: If the keyword set extracted by BERT + BiLSTM-CRF contains less than 10 keywords, it is expanded with keywords retrieved with the proposed TF-IDF(tm) approach, i.e., best ranked keywords according to TF-IDF, which do not appear in the keyword set extracted by BERT + BiLSTM-CRF.
- **TNT-KID & BERT + BiLSTM-CRF & TF-IDF:** the keyword set extracted with the TNT-KID is complemented by keywords extracted with BERT + BiLSTM-CRF (duplicates are removed). If after the expansion the keyword set still contains less than 10 keywords, it is expanded again, this time with keywords retrieved by the TF-IDF(tm) approach.

For TNT-KID, which is the only model that requires language model pretraining, language models were trained on train sets in Table 2 for up to ten epochs. Next, TNT-KID and BERT + BiLSTM-CRF were fine-tuned on the training datasets, which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The documents containing more than 256 tokens are truncated, while the documents containing less than 256 tokens are padded with a special $< \text{pad} >$ token at the end. We fine-tuned each model for a maximum of 10 epochs and after each epoch the trained model was tested on the documents chosen for validation. The model that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set.

6 Evaluation

For evaluation, we employ precision, recall and F1 score. While F1@10 and recall@10 are the most relevant metrics for the media partners, we also report precision@10, precision@5, recall@5 and F1@5. Only keywords which appear in a text (present keywords) were used as a gold standard, since we only evaluate approaches for keyword tagging that are not capable of finding keywords which do not appear in the text. Lowercasing and lemmatization (stemming in the case of Latvian) are performed on both the gold standard and the

[6]More specifically, we use the 'bert-base-multilingual-uncased' implementation of BERT from the Transformers library (`https://github.com/huggingface/transformers`).

Model	**P@5**	**R@5**	**F1@5**	**P@10**	**R@10**	**F1@10**
Croatian						
TF-IDF	0.2226	0.4543	0.2988	0.1466	0.5888	0.2347
TNT-KID	0.3296	0.5135	0.4015	0.3167	0.5359	0.3981
BERT + BiLSTM-CRF	**0.4607**	0.4672	**0.4640**	**0.4599**	0.4708	**0.4654**
TNT-KID & TF-IDF(tm)	0.2659	0.5670	0.3621	0.1688	0.6944	0.2716
BERT + BiLSTM-CRF & TF-IDF(tm)	0.2644	0.5656	0.3604	0.1549	0.6410	0.2495
TNT-KID & BERT + BiLSTM-CRF	0.2940	0.5447	0.3820	0.2659	0.5968	0.3679
TNT-KID & BERT + BiLSTM-CRF & TF-IDF(tm)	0.2648	**0.5681**	0.3612	0.1699	**0.7040**	0.2738
Estonian						
TF-IDF	0.0716	0.1488	0.0966	0.0496	0.1950	0.0790
TNT-KID	**0.5194**	0.5676	**0.5424**	**0.5098**	0.5942	**0.5942**
BERT + BiLSTM-CRF	0.5118	0.4617	0.4855	0.5078	0.4775	0.4922
TNT-KID & TF-IDF(tm)	0.3463	0.5997	0.4391	0.1978	0.6541	0.3037
BERT + BiLSTM-CRF & TF-IDF(tm)	0.3175	0.4978	0.3877	0.1789	0.5381	0.2686
TNT-KID & BERT + BiLSTM-CRF	0.4421	0.6014	0.5096	0.4028	0.6438	0.4956
TNT-KID & BERT + BiLSTM-CRF & TF-IDF(tm)	0.3588	**0.6206**	0.4547	0.2107	**0.6912**	0.3230
Russian						
TF-IDF	0.1764	0.2314	0.2002	0.1663	0.3350	0.2223
TNT-KID	**0.7108**	0.6007	**0.6512**	**0.7038**	0.6250	**0.6621**
BERT + BiLSTM-CRF	0.6901	0.5467	0.5467	0.6849	0.5643	0.6187
TNT-KID & TF-IDF(tm)	0.4519	0.6293	0.5261	0.2981	0.6946	0.4172
BERT + BiLSTM-CRF & TF-IDF(tm)	0.4157	0.5728	0.4818	0.2753	0.6378	0.3846
TNT-KID & BERT + BiLSTM-CRF	0.6226	0.6375	0.6300	0.5877	0.6707	0.6265
TNT-KID & BERT + BiLSTM-CRF & TF-IDF(tm)	0.4622	**0.6527**	0.5412	0.2965	**0.7213**	0.4203
Latvian						
TF-IDF	0.2258	0.5035	0.3118	0.1708	0.5965	0.2655
TNT-KID	0.6089	0.6887	**0.6464**	0.6054	0.6960	**0.6476**
BERT + BiLSTM-CRF	**0.6215**	0.6214	0.6214	**0.6204**	0.6243	0.6223
TNT-KID & TF-IDF(tm)	0.3402	**0.7934**	0.4762	0.2253	0.8653	0.3575
BERT + BiLSTM-CRF & TF-IDF(tm)	0.2985	0.6957	0.4178	0.1889	0.7427	0.3012
TNT-KID & BERT + BiLSTM-CRF	0.4545	0.7189	0.5569	0.4341	0.7297	0.5443
TNT-KID & BERT + BiLSTM-CRF & TF-IDF(tm)	0.3318	0.7852	0.4666	0.2124	**0.8672**	0.3414

Table 3: Results on the EMBEDDIA media partner datasets.

extracted keywords (keyphrases) during the evaluation. The results of the evaluation on all four languages are listed in Table 3.

Results suggest, that neural approaches, TNT-KID and BERT+BiLSTM-CRF offer comparable performance on all datasets but nevertheless achieve different results for different languages. TNT-KID outperforms BERT-BiLSTM-CRF model according to all the evaluation metrics on the Estonian and Russian news dataset. It also outperforms all other methods in terms of precision and F1 score. On the other hand, BERT+BiLSTM-CRF performs better on the Croatian dataset in terms of precision and F1-score. On Latvian TNT-KID achieves top results in terms of F1, while BERT+BiLSTM-CRF offers better precision.

Even though the TF-IDF tagset matching method performs poorly on its own, we can nevertheless drastically improve the recall@5 and the recall@10 of both neural systems, if we expand the keyword tag sets returned by the neural methods with the TF-IDF ranked keywords. The improvement is substantial and consistent for all datasets, but it nevertheless comes at the expanse of the lower precision and F1 score. This is not surprising, since the final expanded keyword set always returns 10 keywords, i.e., much more than the average number of present gold standard keywords in the media partner datasets (see Table 2), which badly affects the precision of the approach. Nevertheless, since for a journalist a manual inspection of 10 keyword candidates per article and manual selection of good candidates (e.g., by clicking on them) still requires less time than the manual selection of keywords from an article, we argue that the improvement of recall at the expanse of the precision is a good trade off, if the system is intended to be used as a recommendation system in the media house environment.

Combining keywords returned by TNT-KID and BERT + BiLSTM-CRF also consistently improves recall, but again at the expanse of lower precision and F1 score. Overall, for all four languages, the best performing method in terms of recall is the TNT-KID & BERT + BiLSTM-CRF & TF-IDF(tm).

7 Conclusion and Future Work

In this work, we tested two state-of-the-art neural approaches for keyword extraction, TNT-KID (Martinc et al., 2020a) and BERT BiLSTM-CRF (Sahrawat et al., 2020), on three less resourced European languages, Estonian, Latvian, Croatian, as well as on Russian. We also proposed a tagset based keyword expansion approach, which drastically improves the recall of the method, making it more suitable for the application in the media house environment.

Our study is one of the very few studies where supervised keyword extraction models were employed on several less resourced languages. The results suggest that these models perform well on languages other than English and could also be successfully leveraged for keyword extraction on morphologically rich languages.

The focus of the study was whether we can improve the recall of the supervised models, in order to make them more useful as recommendation systems in the media house environment. Our method manages to increase the number of retrieved keywords, which drastically improves the recall for all languages. For example, by combing all neural methods and the TF-IDF based approach, we improve on the recall@10 achieved by the best performing neural model, TNT-KID, by 16.81 percentage points for Croatian, 9.70 percentage points for Estonian, 9.63 percentage points for Russian and 17.12 percentage points for Latvian. The resulting method nevertheless offers lower precision, which we will try to improve in the future work.

In the future we also plan to perform a qualitative evaluation of our methods by journalists from the media houses. Next, we plan to explore how adding background knowledge from knowledge databases - lexical (e.g. Wordnet(Fellbaum, 1998)) or factual (e.g. WikiData(Vrandečić and Krötzsch, 2014)) would benefit the aforementioned methods. The assumption is that with the linkage of the text representation and the background knowledge we would achieve a more representative understanding of the articles and the concepts appearing in them, which would result in a more successful keyword extraction.

In traditional machine-learning setting a common practice of combining different classifier outputs to a single output is referred to as stacking. We propose further research on this topic by testing combinations of various keyword extraction models. Finally, we also plan to further improve our unsupervised TF-IDF based keyword extraction method. One way to to do this would be to add the notion of positional encoding, since some of the keywords in the news-media domain often can be found at the beginning of the article and the TF-IDF(tm) does not take this into account while applying the weighting on the matched terms.

8 Acknowledgements

This paper is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No. 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). The third author was financed via young research ARRS grant. Finally, the authors acknowledge the financial support from the Slovenian Research Agency for research core funding for the programme Knowledge Technologies (No. P2-0103) and the project TermFrame - Terminology and Knowledge Frames across Languages (No. J6-9372).

References

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257 – 289.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Samhaa R. El-Beltagy and Ahmed Rafea. 2009. Kp-miner: A keyphrase extraction system for english and arabic documents. *Inf. Syst.*, 34(1):132–144.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Matjaž Juršič, Igor Mozetič, Tomaž Erjavec, and Nada Lavrač. 2010. Lemmagen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, 16(9):1190–1214.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Matej Martinc, Blaž Škrlj, and Senja Pollak. 2020a. Tnt-kid: Transformer-based neural tagger for keyword identification. *arXiv preprint arXiv:2003.09166*.

Matej Martinc, Blaž Škrlj, and Senja Pollak. 2020b. Tnt-kid: Transformer-based neural tagger for keyword identification.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Senja Pollak, Marko Robnik Šikonja, Matthew Purver, Michele Boggia, Ravi Shekhar, Marko Pranjić, Salla Salmela, Ivar Krustok, Tarmo Paju, Carl-Gustav Linden, Leo Leppänen, Elaine Zosa, Matej Ulčar, Linda Freienthal, Silver Traat, Luis Adrián Cabrera-Diego, Matej Martinc, Nada Lavrač, Blaž Škrlj, Martin Žnidaršič, Andraž Pelicon, Boshko Koloski, Vid Podpečan, Janez Kranjc, Shane Sheehan, Hannu Toivonen, Emanuela Boros, Jose Moreno, and Antoine Doucet. 2021. EMBEDDIA tools, datasets and challenges: Resources and hackathon contributions. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20.

Dhruva Sahrawat, Debanjan Mahata, Mayank Kulkarni, Haimin Zhang, Rakesh Gosangi, Amanda Stent, Agniv Sharma, Yaman Kumar, Rajiv Ratn Shah, and Roger Zimmermann. 2020. Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. In *Proceedings of European Conference on Information Retrieval (ECIR 2020)*, pages 328–335.

Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval.

Blaž Škrlj, Andraž Repar, and Senja Pollak. 2019. Rakun: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. In *International Conference on Statistical Language and Speech Processing*, pages 311–323. Springer.

Lucas Sterckx, Thomas Demeester, Johannes Deleu, and Chris Develder. 2015. Topical word importance for fast keyphrase extraction. In *Proceedings of the 24th International Conference on World Wide Web*, pages 121–122.

Si Sun, Chenyan Xiong, Zhenghao Liu, Zhiyuan Liu, and Jie Bao. 2020. Joint keyphrase chunking and salience ranking with bert. *arXiv preprint arXiv:2004.13639*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*, volume 8, pages 855–860.

Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 2005. Kea: Practical automated keyphrase extraction. In *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, pages 129–152. IGI global.

4.5 Final Remarks

We have shown that combining supervised neural state-of-the-art keyword extractors with unsupervised symbolic keyword extractors can be beneficial in some specific use cases. On the other hand, this strategy would be hard to employ in order to improve the overall performance of the system in terms of F_1. The reason for this is the large gap in performance between supervised and unsupervised keyword extractors, which in most cases results in a hybrid system performing somewhere in the middle between a high performing supervised model and a low performing unsupervised model. This means that the development and employment of hybrid keyword detection systems will most likely not be a thriving research trend in the future, but rather a limited effort serving some very specific needs.

The development of supervised hybrid keyword extraction approaches capable of outperforming state-of-the-art neural models is also questionable. As was stated in Section 1, neural networks tend to outperform symbolic approaches by a large margin on all sequence labeling tasks, where sufficient amount of data is available. And, as we have showed in this Chapter, labeled data requirements of the neural system can be reduced by employing the transfer learning paradigm and train a model on a domain specific corpus.

On the other hand, the development of hybrid unsupervised approaches seems a perspective research direction. Novel embedding-based unsupervised keyword extractors that leverage neural representations (Grootendorst, 2020; Mahata et al., 2018) offer performance on pair with the state-of-the-art symbolic models. A hybrid system that would efficiently combine these neural and symbolic representations could therefore push the state-of-the-art for unsupervised keyword extractors even further. These models would be especially useful in domains and languages without sufficient resources for the training of supervised models. Perhaps they could also reduce the gap in performance between supervised and unsupervised models.

Chapter 5

Conclusions and Further Work

In this thesis, novel strategies for tackling problems from three distinct NLP research areas, author profiling, readability detection and keyword extraction, were presented. The common procedure applied for all presented use cases is to combine neural and symbolic text representations. The motivation for this methodology comes from the fact that while neural approaches, which have started to dominate NLP in recent years, have been responsible for major advances in semantic modelling, they still showcase several deficiencies (e.g., neural approaches require a lot of data, are computationally expensive, lack an efficient weighting scheme, etc.) that can be alleviated by combining these methods with symbolic methods. In this chapter, we first summarize the scientific contributions of the research in Section 5.1. After that we discuss the strengths and weaknesses of the proposed approach in Section 5.2 and finally propose several directions for further work in Section 5.3 and instructions on reproduction of the experiments in Section 5.4.

5.1 Summary of Scientific Contributions

We have proposed novel approaches to combine distinct text representations to solve problems from three research areas: author profiling (AP), readability detection, and keyword extraction. By employing this framework, we have managed to improve performance on several use cases. Therefore, we can confirm the general hypothesis H_g, which stated that combining different representations, which carry information about distinct aspects of the text, and establishing synergy between these representations, can lead to a boost in the performance of the NLP system. While the choice of the fusion between representations is dependent on the format of each representation and the task at hand, the conducted experiments indicate several different fusion types can be successfully employed.

First, we managed to achieve goal G1 and improve semantic modelling in the field of AP by proposing two strategies. The first strategy relied on adding symbolic semantic features based on word taxonomies (Škrlj et al., 2021) to traditional BON features (see Section 2.4 for details). In the first step of the algorithm, a document-based taxonomy was constructed from the input document corpus by mapping the words from individual documents of a corpus to terms of the WordNet taxonomy (Miller, 1995). These document-based taxonomies, which model semantic structures derived from the hypernym relation between words, are then merged into a joint corpus-based taxonomy, relations inside which can then be used as additional semantic features. With the extensive experimentation on several datasets we have managed to confirm our hypothesis H1 that semantic modelling can be improved by including background knowledge, such as taxonomies. The conducted experiments also showed that the development of neuro-symbolic approaches might be a viable option for the improvement of semantic modelling, since one of the baselines tested during the experiments, which combined symbolic BON features with neural doc2vec embeddings, showcased good performance.

The second strategy relied on the development of a hybrid algorithm that combines sophisticated feature engineering techniques from traditional approaches to text classification with the newer neural automatic feature construction (Martinc & Pollak, 2019) (see Section 2.5 for details). The neural part of the proposed hybrid neuro-symbolic approach is based on character-level CNNs, which are able to identify important parts of a text sequence by employing a max-over-time pooling operation that keeps only the character sequences with the highest predictive power in the text. The network was modified to enable an additional input, which helps to overcome the lack of an effective weighting scheme. The final architecture requires that the text is fed to the network in the form of two distinct inputs, a standard sequence of characters and an additional BON matrix of TF-IDF weighted features. The approach proved successful in beating the state-of-the-art on the task of language variety prediction, therefore we can confirm the hypothesis H2, which claimed that combining neural and symbolic representations can advance the state-of-the-art on text document classification tasks.

When it comes to readability prediction, we first proposed to use neural language models in an unsupervised way. Traditionally, n-gram language models are employed in the supervised classification setting by training a separate language model for each readability class. It is assumed that low perplexity scores calculated by language models trained on less readable texts and high perplexity scores of language models trained on more readable texts would indicate a high reading level, and high perplexity scores of language models trained on less readable texts and low perplexity scores of language models trained on more readable texts would indicate a low reading level. These language model perplexities are used as features in a supervised readability classification (Petersen & Ostendorf, 2009; Xia et al., 2016). On the other hand, we test the possibility of using a neural language model as a standalone unsupervised readability predictor, since neural language models tend to capture much more information compared to the traditional n-gram models (Martinc, Pollak, & Robnik-Šikonja, 2021) (see Section 3.3 for details).

The statistics derived from the language model (i.e., negative log-likelihood) are combined with shallow symbolic readability indicators that consider simple text statistics, such as sentence length. In this way we obtain a novel readability formula Ranked Sentence Readability Score (RSRS) that is not only based on shallow statistics but also takes background knowledge and discourse cohesion into consideration by leveraging language model statistics. We show that the formula offers good correlation with gold standard readability scores across different genres and languages, therefore we can claim that goal G2, i.e. proposing a new readability formula that offers state-of-the-art performance and can be easily adapted to specific domains and languages, has been achieved. We can also confirm hypothesis H3, which stated that a novel readability measure, which would include background knowledge and discourse cohesion indicators obtained from neural language models in addition to the standard shallow symbolic lexical sophistication indicators, offers better readability estimations than traditional readability formulas.

Finally, we propose a novel approach to keyword extraction. In order to decrease the needed amount of manually labelled data, we propose to leverage the transfer learning technique, where a keyword tagger is first trained in an unsupervised way as a language model on a large corpus and then fine-tuned on a (usually) small-sized corpus with manually labelled keywords. To improve the performance of the model, the transformer architecture was adapted for the specific task at hand by adapting the attention mechanism to directly model the relation between a token and its position, since positional information is especially important for keyword detection (see Section 4.3 for details).

Hypothesis H4 stated that a system for sequence labeling, which combines neural and symbolic text representations, would achieve performance comparable to state-of-the-art, while requiring only a fraction of manually labelled data required by neural approaches. While we did manage to build a system that offers state-of-the-art performance without the need for a large manually labelled dataset (i.e. we can claim that goal G3 was achieved), this system nevertheless uses just neural representations, therefore we cannot confirm hypothesis H4. Even more, if we combine this neural approach with a symbolic unsupervised TF-IDF-based keyword detector, the performance of the system in terms of F_1@10 and F_1@5 scores (i.e., the two most common measures of keyword extractor effectiveness) actually drops. We do show that the proposed neuro-symbolic combination is a good way to drastically improve the recall of the approach and therefore make it suitable to be used as a recommendation system in the news media environment (see Section 4.4 for details).

5.2 Discussing the Strength and Weaknesses of the Proposed Framework

In this thesis, we have compared the proposed neuro-symbolic approach with other state-of-the-art approaches from the related work across several dimensions. While the performance of the method (measured in terms of accuracy, F1, precision, recall, etc.) was usually the main comparison criteria, we have also discussed other important aspects of the approach. Interpretability was discussed in several places, since it enables the user of a method an insight into its functioning. This is a crucial aspect, since it allows easier debugging of a method, transfer of learnings of the method employed into a broader knowledge base and also offers protection against the bias embedded in the algorithm. In Chapter 4, we have discussed the computational complexity of different supervised and unsupervised algorithms for keyword detection, since this aspect affects the ease of usability of a specific method and is also a dimension, along which there is a clear distinction between supervised and unsupervised approaches for keyword detection. Finally, ease of adaptability and transferability was thoroughly discussed in Chapter 3 due to the fact that most readability formulas in the past have been developed for educational purposes in English speaking countries and therefore cannot be directly transferred to other languages and domains. Transferability was also discussed in Chapter 4, where we claimed that state-of-the-art neural approaches from the related work are hard to transfer to less resourced languages and domains due to scarce manually labelled resources of sufficient size.

In this section, we compare the strengths and weaknesses of the proposed methods across the above mentioned dimensions, which reflect different aspects of the methods' functionality. To be more specific, we compare performance, complexity and scalability, ease of adaptability and transferability, and interpretability of the developed methods to the methods that represent state-of-the-art in each specific field (i.e. author profiling, readability, and keyword extraction).

Performance

We have shown on several use cases that we improved the preexisting state-of-the-art by employing the proposed approach of combining neural and symbolic representations. Good performance is therefore a strength of the proposed approach. Nevertheless, in many cases the gains in performance are not substantial and the method can on some specific datasets be outperformed by other baseline approaches.

In the author profiling use case, where the language variety classification was tackled with a combination of CNN and TF-IDF-based features, we have shown that a gain in performance is obtained in a large majority of cases when two distinct feature types are combined (see Table 2.5). Nevertheless, the employment of the method on the ADIC corpus is a counter example that shows that the synergy between two distinct methods will not be successful in some specific scenarios. What makes the ADIC example different from other examples, on which the method works, is the imbalance in the performance of both, neural and symbolic methods. More specifically, the CNN features perform much worse (more than 10 percentage points worse) than the TF-IDF features. This indicates that the synergy can only be achieved if both methods that are combined perform comparably well. If that is not the case, worse performing features will most likely have a negative effect on the overall performance.

Another indication of this can be observed on the use case of readability prediction. Different versions of RSRS scores, which overall offer the best performance, are ranked badly on the WeeBit corpus. On this specific dataset the perplexity-based scores performed the worst, in most cases showing no correlation to the gold standard readability scores. Here, three different traditional readability measures, GFI, ASL and FKGL, outperformed the best ranked RLM RSRS-simple score, which ranked fourth.

The imbalance effect, which influences the success of synergy between neural and non-neural features, is harder to confirm on the keyword extraction, since there we only focused on improving the recall of the keyword extraction method, and the recall cannot be worsened by just expanding the returned list of keyword candidates returned by the TNT-KID methods with candidates returned by the symbolic TF-IDF-based extraction. Nevertheless, the improvement in R@10 in terms of percentage points is the smallest on the Estonian corpus, where the TF-IDF(tm) method performs the worst.

Complexity and Scalability

When it comes to the computational complexity of the proposed methods, it is in most cases worse than the baseline methods, since they tend to combine two distinct approaches, which can be quite complex. This makes the complexity a weakness of the proposed method. In the case of language variety prediction, feeding the additional sparse TF-IDF weighted bag-of-n-grams matrix into the network drastically increases the number of network parameters and consequentially the computational load of the method. The neuro-symbolic approach is also harder to scale than purely neural methods, since TF-IDF matrix's proportions increase linearly with a vocabulary size, meaning that documents coming from very large corpora with a substantial vocabulary are expensive to represent.

Complexity is also the weakness of the system for readability prediction, since using language model statistics for determining readability of the text is far more time and computationally demanding than just employing the baseline traditional readability formulas. On the other hand, applying an additional TF-IDF-based keyword extraction to expand the list of predictions obtained by a neural method does not add much in terms of time and computational complexity, since TF-IDF keyword extraction is much less complex than neural extraction. In the case of keyword extraction, we have also shown that the proposed TNT-KID neural network has less parameters than the best performing baseline, GPT + BiLSTM-CRF, and therefore tends to be less computationally demanding.

Interpretability

Despite recent advances (Lundberg & Lee, 2017), neural classifiers are still not completely interpretable methods. This generally means that if a neural component is added to a specific symbolic system, the system becomes less interpretable, since insight into some parts of the system becomes limited. On the other hand, adding a symbolic component to a neural system improves the overall interpretability of the system, since some of the classification rules that the supervised system has learned, or the classification criteria the unsupervised system has been designed to consider, are easy to inspect and clearly defined.

In the case of language variety classification, we can claim that feeding an additional TF-IDF weighted n-gram matrix into a neural network increases the overall interpretability of the system. On the other hand, the proposed approach is less interpretable than most other state-of-the-art approaches in the field of author profiling, which in most cases rely on purely symbolic methods. Similar can be said for the approach proposed for readability detection, since neural language models used in the proposed RSRS formula have limited interpretability and most baseline methods are symbolic.

When it comes to sequence labelling tasks, where the established approach is the employment of neural models, the usage of a simple TF-IDF-based keyword extractor does help with the interpretability. Since the system allows to determine at least for some keywords why they were chosen, it is more interpretable than most other state-of-the-art systems. The interpretation is still limited since a large amount of returned keywords is extracted by a neural TNT-KID method. While we do show that an insight into the system can be obtained by the visualization of the attention mechanism (Martinc, Škrlj, et al., 2021), this insight is more restricted than for symbolic methods.

For these reasons, we can conclude that the interpretability can be considered a weakness of the method for at least two out of three use cases presented in the thesis: author profiling and readability.

Ease of Adaptability and Transferability

Due to the supervised nature of the proposed approaches, we claim that they can be easily adapted to novel domains and languages. The language variety classification experiments show that the system works well across eight different language groups without any hyperparameter tweaking. There are also no language- or domain-specific features that would make the transfer problematic.

The readability experiments indicate that the proposed RSRS readability measure offers robust performance across several languages and domains, which cannot be said for the traditional readability formulas. This is due to the trainable nature of the language models the score employs, which can be adapted to each specific readability task, domain and language.

Finally, keyword extraction experiments show that the proposed transformer-based approach offers competitive performance across five different languages (English, Croatian, Estonian, Latvian and Russian). Since the approach relies on language model pretraining, it requires much less labelled data than most other state-of-the-art neural approaches. This means that the method can also be employed for keyword extraction in less resourced languages, where large manually labelled keyword datasets are scarce.

Overall, we can conclude that the proposed methods can be easily adapted and transferred across different languages and domains. This means that adaptability and transferability are the strong points of the proposed methods.

5.3 Further Work

While in further work we plan to focus on the weaknesses of the proposed approach, we will still try to improve the performance and transferability of the methods. For example, the error analysis that we conducted on Slavic language varieties, as part of the language variety classification experiments, indicated that the system's performance could be further improved by employing transfer learning techniques (Devlin et al., 2019), since that would allow the model to obtain some background information useful for language variety classification. If during pretraining strong semantic connections could be established between specific named entities and words typical for a specific variety, that would offer a classifier a strong clue to which variety does a specific document, containing these named entities, belong.

For readability, we plan to test the RSRS measure on more gold standard readability corpora from many languages and domains in order to further study the robustness and transferability of the score. This will require new readability datasets which are currently rare or not publicly available. The language model training regime could also be improved by studying how the genre discrepancy between the language model training corpus and the dataset, for which readability needs to be predicted, affects the performance of the score. By studying this and also by determining other influencing factors, we could further improve the performance of the score.

For keyword extraction, we plan to further investigate the influence of language model pretraining on the system's performance. We have shown that there is a noticeable difference between performances of two distinct pretraining regimes, autoregressive language modelling and masked language modelling, when only limited textual resources are available. We plan to investigate this line of work further by trying out different pretraining objectives and by improving the existing objectives (e.g., masking strategy could be further improved by targeting specific words and phrases for masking in order to maximize the model's learning potential). We also plan to test the TNT-KID in a zero-shot cross-lingual keyword detection setting, where the model would be pretrained on a multilingual corpus, fine-tuned on one language and then tested on the second language. If we managed to optimize the performance of the model for this specific setting, the model would become transferable even to languages with no manually labelled resources and could replace unsupervised approaches currently used for these languages.

Another line of experiments we plan to pursue in the scope of further work on the topic of keyword extraction will be aimed at exactly determining how much domain specific data is required for the successful pretraining of the TNT-KID model. We will experiment with training sets of variable sizes and observe the subsequent impact on the performance. Also, we plan to use other performance measures besides $F_1@k$ score, Precision@k and Recall@k to compare our model to others. Currently, TNT-KID compares better to other systems in terms of F1@10 than in terms of F1@5, which raises a question how biased these measures of performance actually are. To make the evaluation more robust, we will apply other measures, such as for example mean average precision and mean reciprocal rank, which also consider the ranking of keywords in the retrieved list.

Decreasing the complexity of the models and improving the scalability of the proposed methods will be one of the two main foci of our further work. For language variety classification, we feed the sparse weighted bag-of-n-grams matrix into the neural network, which drastically increases the number of network parameters and consequentially increases the computational costs. In the existing study, we managed to minimize the size of the sparse matrix without compromising the performance of the system by removing n-grams with low document frequencies from the input matrix but the options for further reducing the symbolic component of the system are limited. Instead, we plan to avoid feeding the bag-of-n-grams matrix to the neural classifier altogether and focus on fixing current deficiencies of CNN neural networks by injecting global document/corpus-level information in a more efficient manner. One option is the integration of the self-attention mechanism (Vaswani et al., 2017) into the CNN network. The attention mechanism can be interpreted as an efficient weighting scheme since, among others, it can also determine how much attention the model should pay to a specific word. An additional hypothetical advantage of integrating the attention mechanism into the network would be improved semantic modelling since the mechanism can also identify important relations between words, which might have been overlooked in the current implementation of the model.

When it comes to readability and keyword extraction, the time and computational complexity of the systems could be decreased by the knowledge distillation techniques, where knowledge encoded in large *teacher* models can be effectively transferred to a smaller *student* model (Sanh et al., 2019). This technique would allow us to decrease the language models used in the RSRS score and the trained keyword extraction models. For readability, we also plan to test out novel transformer-based architectures for language modelling (e.g., a Transformer-X language model proposed by Dai et al. (2019)) that would be trained from scratch on corpora with appropriate readability. These models are much faster than the currently used recurrent and temporal convolutional language models since they allow more operations to run in parallel.

The other main focus of the further work will be increasing the interpretability of the models. The easiest way to increase the interpretability and explainability of the models would be to remove the neural components and rely purely on the symbolic methods, such as for example tax2vec. This would likely have a detrimental effect on the performance of the systems for some use cases. Therefore, a preferable option would be to increase the interpretability of the neural part of the model.

When it comes to language variety classification, the addition of the attention mechanism discussed above would improve the interpretability of the network since the mechanism can be visualized and offers some insight into the decision process of the network, as we have shown in Martinc, Škrlj, et al. (2021). Another option we plan to investigate is the employment of several techniques designed to interpret convolutional networks by inspecting the convolutional filters and max pooling mechanisms in higher convolutional layers (Q. Zhang et al., 2018).

Interpretability and explainability are especially important in the readability research, since systems for determining readability are often used for educational purposes. There, the users (educators, teachers, etc.) need to understand the cause of the readability prediction. While the RSRS score represents an aggregation of word NLL scores across a sentence, a more in-depth analysis and visualization of each specific word-level NLL score would offer a better insight into the system.

Finally, while we have shown that the proposed keyword extraction method is already explainable through an attention mechanism, which allows to identify words and phrases to which the classifier paid the most attention, we plan to further improve the interpretability of the system by employing general explanation techniques such as SHAP (Lundberg & Lee, 2017).

5.4 Implementation and Availability

The source code for all the experiments presented in this dissertation is publicly available under the MIT license in order to assure reproducibility of the conducted research. More specifically, the code is available in the following git repositories:

- Code for experiments presented in Section 2.3 and described in paper **'PAN 2017: Author Profiling - Gender and Language Variety Prediction' (Martinc et al., 2017)**: https://github.com/pan-webis-de/martinc17

- Code for experiments presented in Section 2.4 and described in paper **'tax2vec:**

Constructing Interpretable Features from Taxonomies for Short Text Classification' (Škrlj et al., 2021): https://github.com/SkBlaz/tax2vec

- Code for experiments presented in Section 2.5 and described in paper **'Combining N-grams and Deep Convolutional Features for Language Variety Classification' (Martinc & Pollak, 2019)**: http://source.ijs.si/mmartinc/NLE_2017
- Code for experiments presented in Section 3.3 and described in paper **'Supervised and Unsupervised Neural Approaches to Text Readability' (Martinc, Pollak, & Robnik-Šikonja, 2021)**: https://gitlab.com/matej.martinc/text_readability
- Code for experiments presented in Section 4.3 and described in paper **'TNT-KID: Transformer-based Neural Tagger for Keyword Identification' (Martinc, Škrlj, et al., 2021)**: https://gitlab.com/matej.martinc/tnt_kid/
- Code for experiments presented in Section 4.4 and described in paper **'Extending Neural Keyword Extraction with TF-IDF tagset matching' (Koloski et al., 2021)**: https://github.com/bkolosk1/Extending-Neural-Keyword-Extraction-with-TF-IDF-tagset-matching/

References

Ali, M. (2018a). Character level convolutional neural network for arabic dialect identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, 122–127.

Ali, M. (2018b). Character level convolutional neural network for german dialect identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, 172–177.

Ali, M. (2018c). Character level convolutional neural network for indo-aryan language identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, 283–287.

Baevski, A., & Auli, M. (2019). Adaptive input representations for neural language modeling. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. https://openreview.net/forum?id=ByxZX20qFQ

Baevski, A., Edunov, S., Liu, Y., Zettlemoyer, L., & Auli, M. (2019). Cloze-driven pretraining of self-attention networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5360–5369. https://doi.org/10.18653/v1/D19-1539

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H., & Nissim, M. (2017). N-gram: New groningen author-profiling model. *1866*. http://ceur-ws.org/Vol-1866/paper%5C_71.pdf

Basile, P., Caputo, A., & Semeraro, G. (2014). An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 1591–1600.

Belinkov, Y., & Glass, J. (2016). A character-level convolutional neural network for distinguishing similar languages and dialects. *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, 145–152. https://www.aclweb.org/anthology/W16-4819

Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3615–3620. https://doi.org/10.18653/v1/D19-1371

Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., & Jaggi, M. (2018). Simple unsupervised keyphrase extraction using sentence embeddings. *Proceedings of the 22nd Conference on Computational Natural Language Learning*, 221–229. https://doi.org/10.18653/v1/K18-1022

Bestgen, Y. (2017). Improving the character ngram model for the dsl task with bm25 weighting and less frequently used feature sets. *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, 115–123.

Bird, S., & Loper, E. (2004). NLTK: The natural language toolkit. *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, 31.

Bjerva, J. (2016). Byte-based language identification with deep convolutional networks. *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, 119–125. https://www.aclweb.org/anthology/W16-4816

Bohnet, B., McDonald, R., Simões, G., Andor, D., Pitler, E., & Maynez, J. (2018). Morphosyntactic tagging with a meta-BiLSTM model over context sensitive token encodings. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2642–2652. https://doi.org/10.18653/v1/P18-1246

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146. https://doi.org/10.1162/tacl_a_00051

Bougiatiotis, K., & Krithara, A. (2016). Author profiling using complementary second order attributes and stylometric features (K. Balog, L. Cappellato, N. Ferro, & C. Macdonald, Eds.). *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers*.

Bougouin, A., Boudin, F., & Daille, B. (2013). TopicRank: Graph-based topic ranking for keyphrase extraction. *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, 543–551.

Brandes, U. (2001). A faster algorithm for betweenness centrality. *The Journal of Mathematical Sociology*, *25*(2), 163–177.

Burger, J. D., Henderson, J., Kim, G., & Zarrella, G. (2011). Discriminating gender on twitter. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1301–1309.

Busger op Vollenbroek, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H., & Nissim, M. (2016). Gronup: Groningen user profiling notebook for PAN at clef 2016 (K. Balog, L. Cappellato, N. Ferro, & C. Macdonald, Eds.). *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers*.

Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., & Jatowt, A. (2020). Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, *509*, 257–289.

Campos, R., Mangaravite, V., Pasquali, A., Jorge, A. M., Nunes, C., & Jatowt, A. (2018). Yake! collection-independent automatic keyword extractor. *European Conference on Information Retrieval*, 806–810.

Chakravarthi, B. R., Mihaela, G., Ionescu, R. T., Jauhiainen, H., Jauhiainen, T., Lindén, K., Ljubešić, N., Partanen, N., Priyadharshini, R., Purschke, C., Rajagopal, E., Scherrer, Y., & Zampieri, M. (2021). Findings of the VarDial evaluation campaign 2021. *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects*, 1–11. https://aclanthology.org/2021.vardial-1.1

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

Chen, J., Zhang, X., Wu, Y., Yan, Z., & Li, Z. (2018). Keyphrase generation with correlation constraints. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4057–4066. https://doi.org/10.18653/v1/D18-1439

Chen, Q., Lamoreaux, A., Wang, X., Durrett, G., Bastani, O., & Dillig, I. (2021). Web question answering with neurosymbolic program synthesis. *Proceedings of the 42nd*

ACM SIGPLAN International Conference on Programming Language Design and Implementation, 328–343.

Chowdhury, G. G. (2010). *Introduction to modern information retrieval.* Facet publishing.

Chu, Z., Gianvecchio, S., Wang, H., & Jajodia, S. (2010). Who is tweeting on twitter: Human, bot, or cyborg? *Proceedings of the 26th annual computer security applications conference*, 21–30.

Chu, Z., Gianvecchio, S., Wang, H., & Jajodia, S. (2012). Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *IEEE Transactions on Dependable and Secure Computing*, *9*(6), 811–824.

Collins-Thompson, K. (2014). Computational assessment of text readability: A survey of current and future research. *ITL-International Journal of Applied Linguistics*, *165*(2), 97–135.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011a). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493–2537.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011b). Natural language processing (almost) from scratch. *Journal of machine learning research*, *12*(ARTICLE), 2493–2537.

Çöltekin, Ç. (2020). Dialect identification under domain shift: Experiments with discriminating romanian and moldavian. *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, 186–192.

Çöltekin, Ç., & Rama, T. (2016). Discriminating similar languages with linear svms and neural networks. *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, 15–24.

Çöltekin, Ç., Rama, T., & Blaschke, V. (2018). Tübingen-oslo team at the vardial 2018 evaluation campaign: An analysis of n-gram features in language variety identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, 55–65.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747

Crossley, S. A., Skalicky, S., Dascalu, M., McNamara, D. S., & Kyle, K. (2017). Predicting text comprehension, processing, and familiarity in adult readers: New approaches to readability formulas. *Discourse Processes*, *54*(5-6), 340–359.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., & Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2978–2988. https://doi.org/10.18653/v1/P19-1285

Dale, E., & Chall, J. S. (1948). A formula for predicting readability: Instructions. *Educational research bulletin*, 37–54.

Davison, A., & Kantor, R. N. (1982). On the failure of readability formulas to define readable texts: A case study from adaptations. *Reading research quarterly*, 187–209.

De Raedt, L., Manhaeve, R., Dumancic, S., Demeester, T., & Kimmig, A. (2019). Neuro-symbolic= neural+ logical+ probabilistic. *NeSy'19@ IJCAI, the 14th International Workshop on Neural-Symbolic Learning and Reasoning.*

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, *7*(Jan), 1–30.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Dumitrescu, S., Avram, A.-M., & Pyysalo, S. (2020). The birth of Romanian BERT. *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4324–4328. https://doi.org/10.18653/v1/2020.findings-emnlp.387

Ebersbach, M., Herms, R., & Eibl, M. (2017). Fusion methods for icd10 code classification of death certificates in multilingual corpora. *CLEF (Working Notes)*, 36.

Ehara, Y. (2021). Evaluation of unsupervised automatic readability assessors using rank correlations. *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, 62–72.

Ferrone, L., & Zanzotto, F. M. (2020). Symbolic, distributed, and distributional representations for natural language processing in the era of deep learning: A survey. *Frontiers in Robotics and AI*, *6*. https://doi.org/10.3389/frobt.2019.00153

Filighera, A., Steuer, T., & Rensing, C. (2019). Automatic text difficulty estimation using embeddings and neural networks. *European Conference on Technology Enhanced Learning*, 335–348.

Firoozeh, N., Nazarenko, A., Alizon, F., & Daille, B. (2020). Keyword extraction: Issues and methods. *Natural Language Engineering*, *26*(3), 259–291.

Franco-Salvador, M., Rangel, F., Rosso, P., Taulé, M., & Martít, M. A. (2015). Language variety identification using distributed representations of words and documents. *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Lecture Notes in Computer Science*, 28–40.

Gallina, Y., Boudin, F., & Daille, B. (2019). KPTimes: A large-scale dataset for keyphrase generation on news documents. *Proceedings of the 12th International Conference on Natural Language Generation*, 130–135. https://doi.org/10.18653/v1/W19-8617

Gaman, M., Hovy, D., Ionescu, R. T., Jauhiainen, H., Jauhiainen, T., Lindén, K., Ljubešić, N., Partanen, N., Purschke, C., Scherrer, Y., et al. (2020). A report on the vardial evaluation campaign 2020. *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects.*

Goldberg, Y., & Orwant, J. (2013). A dataset of syntactic-ngrams over time from a very large corpus of english books. *Second Joint Conference on Lexical and Computational Semantics*, 241–247.

Gollapalli, S. D., Li, X.-L., & Yang, P. (2017). Incorporating expert knowledge into keyphrase extraction. *Thirty-First AAAI Conference on Artificial Intelligence.*

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press.

Goutte, C., Léger, S., & Carpuat, M. (2014). The nrc system for discriminating similar languages. *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, 139–145.

Grässer, F., Kallumadi, S., Malberg, H., & Zaunseder, S. (2018). Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. *Proceedings of the 2018 International Conference on Digital Health*, 121–125.

Grave, E., Joulin, A., Cissé, M., Jégou, H., et al. (2017). Efficient softmax approximation for gpus. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1302–1310.

Grootendorst, M. (2020). Keybert: Minimal keyword extraction with bert. https://doi.org/10.5281/zenodo.4461265

Gunning, R. (1952). *The technique of clear writing.* McGraw-Hill, New York.

Heinzerling, B., & Strube, M. (2019). Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 273–291. https://doi.org/10.18653/v1/P19-1027

Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 328–339. https://doi.org/10.18653/v1/P18-1031

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 216–223.

Jauhiainen, T. S., Jauhiainen, H. A., Linden, B. K. J., et al. (2018a). Heli-based experiments in swiss german dialect identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018).*

Jauhiainen, T. S., Jauhiainen, H. A., Linden, B. K. J., et al. (2018b). Iterative language model adaptation for indo-aryan language identification. *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018).*

Jawahar, G., Sagot, B., & Seddah, D. (2019). What does BERT learn about the structure of language? *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3651–3657. https://doi.org/10.18653/v1/P19-1356

Katona, E., Buda, J., & Bolonyai, F. (2021). Using n-grams and statistical features and to identify hate speech spreaders on twitter. *Working Notes Papers of the CLEF.*

Kim, S. N., Medelyan, O., Kan, M.-Y., & Baldwin, T. (2010). Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. *Proceedings of the 5th International Workshop on Semantic Evaluation*, 21–26.

Kim, Y., Jernite, Y., Sontag, D., & Rush, A. M. (2016). Character-aware neural language models. *AAAI*, 2741–2749.

Kincaid, J. P., Fishburne Jr, R. P., Rogers, R. L., & Chissom, B. S. (1975). *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.* Institute for Simulation; Training, University of Central Florida.

Koloski, B., Pollak, S., Škrlj, B., & Martinc, M. (2021). Extending neural keyword extraction with TF-IDF tagset matching. *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, 22–29. https://aclanthology.org/2021.hackashop-1.4

Koppel, M., Argamon, S., & Shimoni, A. R. (2002). Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, *17*(4), 401–412.

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2019). NetSDM: Semantic data mining with network analysis. *Journal of Machine Learning Research*, *20*(32), 1–50.

Krapivin, M., Autaeu, A., & Marchese, M. (2009). *Large dataset for keyphrases extraction* (tech. rep.). University of Trento.

Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 66–71. https://doi.org/10.18653/v1/D18-2012

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. *International conference on machine learning*, 1188–1196.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

Lee, B. W., Jang, Y. S., & Lee, J. H.-J. (2021). Pushing on text readability assessment: A transformer meets handcrafted linguistic features. *arXiv preprint arXiv:2109.12258.*

Li, Y. (2018). Towards improving speech emotion recognition for in-vehicle agents: Preliminary results of incorporating sentiment analysis by using early and late fusion methods. *Proceedings of the 6th International Conference on Human-Agent Interaction*, 365–367.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Logar, N., Grčar, M., Brakus, M., Erjavec, T., Holdt, Š. A., Krek, S., & Kosem, I. (2012). *Korpusi slovenskega jezika gigafida, kres, ccgigafida in cckres: Gradnja, vsebina, uporaba*. Trojina, zavod za uporabno slovenistiko.

López-Monroy, A. P., Montes-y-Gómez, M., Escalante, H. J., & Pineda, L. V. (2014). Using intra-profile information for author profiling. *CLEF (Working Notes)*, 1116–1120.

Luan, Y., Ostendorf, M., & Hajishirzi, H. (2017). Scientific information extraction with semi-supervised neural tagging. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2641–2651. https://doi.org/10.18653/v1/D17-1279

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 4768–4777.

Ma, K., Francis, J., Lu, Q., Nyberg, E., & Oltramari, A. (2019). Towards generalizable neuro-symbolic systems for commonsense question answering. *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, 22–32. https://doi.org/10.18653/v1/D19-6003

Mahata, D., Kuriakose, J., Shah, R., & Zimmermann, R. (2018). Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 634–639.

Maier, W., & Gómez-Rodríguez, C. (2014). Language variety identification in Spanish tweets. *Language Technology for Closely Related Languages and Language Variants*, 25–35.

Malmasi, S., & Zampieri, M. (2016). Arabic dialect identification in speech transcripts. *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, 106–113.

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A., & Tiedemann, J. (2016). Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, 1–14.

Martinc, M., Haider, F., Pollak, S., & Luz, S. (2021). Temporal integration of text transcripts and acoustic features for alzheimer's diagnosis based on spontaneous speech. *Frontiers in Aging Neuroscience*, 299.

Martinc, M., & Pollak, S. (2019). Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, *25*(5), 607–632.

Martinc, M., Pollak, S., & Robnik-Šikonja, M. (2021). Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, *47*(1), 141–179.

Martinc, M., Skrjanec, I., Zupan, K., & Pollak, S. (2017). PAN 2017: Author profiling - gender and language variety prediction. *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland, September 11-14, 2017*, *1866*. http://ceur-ws.org/Vol-1866/paper%5C_78.pdf

Martinc, M., Škrlj, B., & Pollak, S. (2021). Tnt-kid: Transformer-based neural tagger for keyword identification. *Natural Language Engineering*, 1–40.

Mc Laughlin, G. H. (1969). SMOG grading - a new readability formula. *Journal of reading*, *12*(8), 639–646.

Medelyan, O., Frank, E., & Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, 1318–1327.

Meng, R., Yuan, X., Wang, T., Brusilovsky, P., Trischler, A., & He, D. (2019). Does order matter? an empirical study on generating multiple keyphrases as a sequence. *arXiv preprint arXiv:1909.03590*.

Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., & Chi, Y. (2017). Deep keyphrase generation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 582–592. https://doi.org/10.18653/v1/P17-1054

Miaschip, A., Alzettam, C., Brunato, D., Dell'Orletta, F., & Venturi, G. (2020). Is neural language model perplexity related to readability? *Computational Linguistics CLiC-it 2020*, 303.

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into text. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 404–411. https://www.aclweb.org/anthology/W04-3252

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, 3111–3119. https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html

Mikolov, T., Deoras, A., Kombrink, S., Burget, L., & Černocký, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. *Twelfth Annual Conference of the International Speech Communication Association*, 605–608.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, *38*(11), 39–41.

Miura, Y., Taniguchi, T., Taniguchi, M., & Ohkuma, T. (2017a). Author profiling with word+ character neural attention network. *CLEF (Working notes)*.

Miura, Y., Taniguchi, T., Taniguchi, M., & Ohkuma, T. (2017b). Author profiling with word+ character neural attention network. *CLEF (Working notes)*.

Modaresi, P., Liebeck, M., & Conrad, S. (2016). Exploring the effects of cross-genre machine learning for author profiling in PAN 2016. *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers*.

Nadeem, F., & Ostendorf, M. (2018). Estimating linguistic complexity for science texts. *Proceedings of the Thirteenth workshop on innovative use of NLP for building educational applications*, 45–55.

Nguyen, T. D., & Kan, M.-Y. (2007). Keyphrase extraction in scientific publications. *International conference on Asian digital libraries*, 317–326.

Novak, P. K., Smailović, J., Sluban, B., & Mozetič, I. (2015). Sentiment of emojis. *PloS one*, *10*(12), e0144296.

Pagliardini, M., Gupta, P., & Jaggi, M. (2018). Unsupervised learning of sentence embeddings using compositional n-gram features. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 528–540. https://doi.org/10.18653/v1/N18-1049

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of EMNLP*, 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *Proceedings of NAACL-HLT*, 2227–2237.

Petersen, S. E., & Ostendorf, M. (2009). A machine learning approach to reading level assessment. *Computer speech & language*, *23*(1), 89–106.

Pizarro, J. (2019). Using n-grams to detect bots on twitter. *Working Notes Papers of the CLEF.*

Plank, B., & Hovy, D. (2015). Personality traits on Twitter—or—How to get 1,500 personality tests in a week. *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 92–98. https://doi.org/10.18653/v1/W15-2913

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8).

Rangel, F., Celli, F., Rosso, P., Potthast, M., Stein, B., & Daelemans, W. (2015). Overview of the 3rd author profiling task at PAN 2015. *CLEF 2015 Working Notes.*

Rangel, F., Giachanou, A., Ghanem, B. H. H., & Rosso, P. (2020). Overview of the 8th author profiling task at pan 2020: Profiling fake news spreaders on twitter. *CEUR Workshop Proceedings*, *2696*, 1–18.

Rangel, F., & Rosso, P. (2019). Overview of the 7th author profiling task at pan 2019: Bots and gender profiling in twitter. *Working Notes Papers of the CLEF 2019 Evaluation Labs Volume 2380 of CEUR Workshop.*

Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., & Inches, G. (2013a). Overview of the author profiling task at pan 2013. *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, 352–365.

Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., & Inches, G. (2013b). Overview of the author profiling task at pan 2013. *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, 352–365.

Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017a). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. *Working Notes Papers of the CLEF.*

Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017b). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. *Working notes papers of the CLEF*, 1613–0073.

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B. (2016a). Overview of the 4th Author Profiling Task at PAN 2016: Cross-genre evaluations. *CLEF 2016 Working Notes.*

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B. (2016b). Overview of the 4th author profiling task at pan 2016: Cross-genre evaluations. *Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings*, 750–784.

Robertson, S., & Zaragoza, H. (2009). *The probabilistic relevance framework: Bm25 and beyond.* Now Publishers Inc.

Sahrawat, D., Mahata, D., Kulkarni, M., Zhang, H., Gosangi, R., Stent, A., Sharma, A., Kumar, Y., Shah, R. R., & Zimmermann, R. (2020). Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. *Proceedings of European Conference on Information Retrieval (ECIR 2020)*, 328–335.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108.*

Sapkota, U., Bethard, S., Montes-y-Gómez, M., & Solorio, T. (2015). Not all character n-grams are created equal: A study in authorship attribution. *NAACL HLT 2015,* *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 93–102. http://aclweb.org/anthology/N/N15/N15-1010.pdf

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, *61*, 85–117.

Schwarm, S. E., & Ostendorf, M. (2005). Reading level assessment using support vector machines and statistical language models. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 523–530.

Schwartz, H. A., Eichstaedt, J. C., Kern, M. L., Dziurzynski, L., Ramones, S. M., Agrawal, M., Shah, A., Kosinski, M., Stillwell, D., Seligman, M. E., et al. (2013). Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, *8*(9).

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, *27*(3), 379–423.

Sheehan, K. M., Kostin, I., Napolitano, D., & Flor, M. (2014). The TextEvaluator tool: Helping teachers and test developers select texts for use in instruction and assessment. *The Elementary School Journal*, *115*(2), 184–209.

Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., & Pollak, S. (2021). Tax2vec: Constructing interpretable features from taxonomies for short text classification. *Computer Speech & Language*, *65*, 101104.

Škrlj, B., Repar, A., & Pollak, S. (2019). RaKUn: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. *International Conference on Statistical Language and Speech Processing*, 311–323.

Škvorc, T., Lavrač, N., & Robnik-Šikonja, M. (2022). Nesychair: Automatic conference scheduling combining neuro-symbolic representations and constrained clustering. *IEEE Access*, *10*, 10880–10897.

Smith, E. A., & Senter, R. (1967). Automated readability index. *AMRL-TR. Aerospace Medical Research Laboratories (US)*, 1–14.

Susskind, Z., Arden, B., John, L. K., Stockton, P., & John, E. B. (2021). Neuro-symbolic AI: an emerging class of AI workloads and their characterization. *CoRR*, *abs/2109.06133*. https://arxiv.org/abs/2109.06133

Tan, L., Zampieri, M., Ljubešic, N., & Tiedemann, J. (2014). Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, 11–15.

Tillmann, C., Al-Onaizan, Y., & Mansour, S. (2014). Improved sentence-level arabic dialect classification. *Proceedings of the VarDial Workshop*, 110–119.

Todirascu, A., François, T., Bernhard, D., Gala, N., & Ligozat, A.-L. (2016). Are cohesive features relevant for text readability evaluation? *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 987–997.

Tomokiyo, T., & Hurst, M. (2003). A language model approach to keyphrase extraction. *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment*, 33–40.

Ulčar, M., & Robnik-Šikonja, M. (2020a). FinEst BERT and CroSloEngual BERT. *International Conference on Text, Speech, and Dialogue*, 104–111.

Ulčar, M., & Robnik-Šikonja, M. (2020b). Finest bert and crosloengual bert. *International Conference on Text, Speech, and Dialogue*, 104–111.

Vajjala, S., & Lučić, I. (2018). OneStopEnglish corpus: A new corpus for automatic readability assessment and text simplification. *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, 297–304.

Vajjala, S., & Meurers, D. (2012). On improving the accuracy of readability classification using insights from second language acquisition. *Proceedings of the Seventh workshop on building educational applications using NLP*, 163–173.

Van Dijk, T. A. (1977). *Text and context: Explorations in the semantics and pragmatics of discourse*. Longman London.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 5998–6008.

Vellido, A. (2020). The importance of interpretability and visualization in machine learning for applications in medicine and health care. *Neural Computing and Applications*, *32*(24), 18069–18083.

Vidyasagar, M. (2010). Kullback-leibler divergence rate between probability distributions on sets of different cardinalities. *49th IEEE Conference on Decision and Control (CDC)*, 948–953.

Wan, X., & Xiao, J. (2008). Single document keyphrase extraction using neighborhood knowledge. *Proceedings of the AAAI Conference*, *8*, 855–860.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2019). GLUE: A multi-task benchmark and analysis platform for natural language understanding. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. https://openreview.net/forum?id=rJ4km2R5t7

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (2005). Kea: Practical automated keyphrase extraction. *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, 129–152.

Xia, M., Kochmar, E., & Briscoe, T. (2016). Text readability assessment for second language learners. *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, 12–22.

Xu, W., Callison-Burch, C., & Napoles, C. (2015). Problems in current text simplification research: New data can help. *Transactions of the Association of Computational Linguistics*, *3*(1), 283–297.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489.

Ye, H., & Wang, L. (2018). Semi-supervised learning for neural keyphrase generation. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4142–4153. https://doi.org/10.18653/v1/D18-1447

Yuan, X., Wang, T., Meng, R., Thaker, K., Brusilovsky, P., He, D., & Trischler, A. (2020). One size does not fit all: Generating and evaluating variable number of keyphrases. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7961–7975. https://doi.org/10.18653/v1/2020.acl-main.710

Zaidan, O. F., & Callison-Burch, C. (2014). Arabic dialect identification. *Computational Linguistics*, *40*(1), 171–202.

Zampieri, M., & Gebre, B. G. (2012). Automatic identification of language varieties: The case of Portuguese. *Proceedings of KONVENS 2012*, 233–237.

Zampieri, M., Gebre, B. G., & Diwersy, S. (2013). N-gram language models and POS distribution for the identification of Spanish varieties. *Proceedings of TALN 2013*, 580–587.

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J., Scherrer, Y., & Aepli, N. (2017). Findings of the vardial evaluation campaign 2017. *Proceedings of the fourth workshop on NLP for similar languages, varieties and dialects.*

Zampieri, M., Malmasi, S., Nakov, P., Ali, A., Shon, S., Glass, J., Scherrer, Y., Samardžić, T., Ljubešić, N., Tiedemann, J., et al. (2018). Language identification and morphosyntactic tagging. the second vardial evaluation campaign., 1–17.

Zampieri, M., Malmasi, S., Scherrer, Y., Samardžić, T., Tyers, F., Silfverberg, M., Klyueva, N., Pan, T.-L., Huang, C.-R., Ionescu, R. T., et al. (2019). A report on the third vardial evaluation campaign.

Zampieri, M., Tan, L., Ljubešić, N., & Tiedemann, J. (2014a). A report on the dsl shared task 2014. *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, 58–67.

Zampieri, M., Tan, L., Ljubešić, N., & Tiedemann, J. (2014b). A report on the dsl shared task 2014. *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, 58–67.

Zhang, Q., Wu, Y. N., & Zhu, S.-C. (2018). Interpretable convolutional neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8827–8836.

Zhang, X., Zhao, J. J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, 649–657.

Zidarn, R. (2020). *Avtomatsko povzemanje slovenskih besedil z globokimi nevronskimi mrežami* (Master's thesis). University of Ljubljana.

Bibliography

Publications Related to the Thesis

Journal Articles

Martinc, M., Pollak, S., & Robnik-Šikonja, M. (2021). Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, 47(1), 141–179.

Martinc, M., Škrlj, B., & Pollak, S. (2020). TNT-KID: Transformer-based neural tagger for keyword identification. *Natural Language Engineering*, 1–40.

Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., & Pollak, S. (2021). Tax2vec: Constructing interpretable features from taxonomies for short text classification. *Computer Speech & Language*, 65, 101104.

Martinc, M., & Pollak, S. (2019). Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, 25(5), 607–632.

Conference Papers

Martinc, M., Skrjanec, I., Zupan, K., & Pollak, S. (2017). PAN 2017: Author profiling - gender and language variety prediction. *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum*, Dublin, Ireland, 1866.

Koloski, B., Pollak, S., Škrlj, B., & Martinc, M. (2021). Extending neural keyword extraction with TF-IDF tagset matching. *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, online, 22–29. https://aclanthology.org/2021.hackashop-1.4

Other Publications

Journal Articles

Martinc, M., Haider, F., Pollak, S., & Luz, S. (2021). Temporal integration of text transcripts and acoustic features for Alzheimer's diagnosis based on spontaneous speech. *Frontiers in aging neuroscience*, 13, 652647-1-652647-15.

Škrlj, B., Martinc, M., Lavrač, N., & Pollak, S. (2021). autoBOT: evolving neuro-symbolic representations for explainable low resource text classifcation. *Machine learning*, 110, 989–1028.

Repar, A., Martinc, M., & Pollak, S. (2020). Reproduction, replication, analysis and adaptation of a term alignment approach. *Language resources and evaluation*, 54(3), 767-800.

Lavrač, N., Martinc, M., Pollak, S., Pompe Novak, M., & Cestnik, B. (2020). Bisociative literature-based discovery : lessons learned and new word embedding approach. *New generation computing*, 38, 773-800.

Martinc, M., Žnidaršič, M., Lavrač, N., & Pollak, S. (2018). Towards creative software blending : computational infrastructure and use cases. *Informatica : an international journal of computing and informatics*, 42(1), 77-84.

Klemenčič, M., Žnidaršič, M., Vavpetič, A., & Martinc, M. (2017). Erasmus students' involvement in quality enhancement of Erasmus+ mobility through digital ethnography and ErasmusShouts. *Studies in higher education*, 42(5), 925-932.

Conference Papers

Pollak, S., Robnik Šikonja, M., Purver, M., Pranjič, M., Ulčar, M., Martinc, M., Lavrač, N., Škrlj, B., Žnidaršič, M., Pelicon, A., Koloski, B., Podpečan, V., Kranjc, J., et al. (2021). EMBEDDIA Tools, datasets and challenges : resources and hackathon contributions. *Proceedings of EACL hackashop on news media content analysis and automated report generation*, online, 99-109.

Pelicon, A., Shekhar, R., Martinc, M., Škrlj, B., Purver, M., & Pollak, S. (2021). Zero-shot cross-lingual content filtering : offensive language and hate speech detection. *Proceedings of EACL hackashop on news media content analysis and automated report generation*, online, 30-34.

Martinc, M., Perger, N., Pelicon, A., Ulčar, M., Vezovnik, A., & Pollak, S. (2021). EMBEDDIA hackathon report : automatic sentiment and viewpoint analysis of Slovenian news corpus on the topic of LGBTIQ+. In: TOIVONEN, Hannu (ed.), BOGGIA, Michele (ed.). *Proceedings of EACL hackashop on news media content analysis and automated report generation*, online, 121-126.

Montariol, S., Martinc, M., & Pivovarova, L. (2021). Scalable and interpretable semantic change detection. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, online, 4642-4652.

Martinc, M., Montariol, S., Pivovarova, L., & Zosa, E. (2020). Discovery Team at SemEval-2020 Task : context-sensitive embeddings not always better than static for semantic change detection. *Proceedings of the 28th International Conference on Computational Linguistics [and] proceedings of the Fourteenth Workshop on Semantic Evaluation*, Barcelona, Spain, 67-73.

Martinc, M., Škrlj, B., Pirkmajer, S., Lavrač, N., Cestnik, B., Marzidovšek, M., & Pollak, S. (2020). COVID-19 therapy target discovery with context-aware literature mining. *Proceedings of the Discovery Science: 23rd International Conference (DS 2020)*, Thessaloniki, Greece, 109-123.

Martinc, M., & Pollak, S. (2020). Tackling the ADReSS challenge : a multimodal approach to the automated recognition of Alzheimer's dementia. *Proceedings of Interspeech 2020*, Shanghai, China, 2157-2161.

Martinc, M., Kralj Novak, P., & Pollak, S. (2020). Leveraging contextual embeddings for detecting diachronic semantic shift. *Proceeedings of LREC 2020 : Twelfth International Conference on Language Resources and Evaluation*, Marseille, France, 4811-4819.

Lavrač, N., Martinc, M., Pollak, S., & Cestnik, B. (2020). Bisociative literature-based discovery: lessons learned and new prospects. *Proceedings of the Eleventh International Conference on Computational Creativity*, Coimbra, Portugal, 139-145.

Vintar, Š., Grčič-Simeunovič, L., Martinc, M., Pollak, S., & Stepišnik, U. (2020). Mining semantic relations from comparable corpora through intersections of word embeddings. *Proceedings of the LREC 2020 13th Workshop on Building and Using Comparable Corpora*, online, 29-34.

Martinc, M., & Pollak, S. (2020). Pooled LSTM for Dutch cross-genre gender classification. *Proceedings of the Shared Task on Cross-Genre Gender Prediction in Dutch at CLIN29*

(GxG-CLIN29) co-located with the 29th Conference on Computational Linguistics in The Netherlands (CLIN29), Groningen, The Netherlands, 1-9.

Pollak, S., Martinc, M., & Mihurko Poniž, K. (2020). Natural language processing for literary text analysis : word-embeddings-based analysis of Zofka Kveder's Work. *Proceedings of the Workshop on Digital Humanities and Natural Language Processing (DHandNLP 2020), co-located with International Conference on the Computational Processing of Portuguese (PROPOR 2020)*, Évora, Portugal, 1-9.

Martinc, M., Montariol, S., Zosa, E., & Pivovarova, L. (2020). Capturing evolution in word usage: just add more clusters?. *Proceedings of the Web Conference 2020*, Taipei, Taiwani, 343-349.

Pollak, S., Repar., A., Martinc, M., & Podpečan, V. (2019). Karst exploration : extracting terms and definitions from Karst domain corpus. *Proceedings of the eLex 2019 Conference*, Sintra, Portugal, pages 934-956.

Martinc, M., Žnidaršič, M., & Pollak, S. (2019). System for rapid classification and analysis of financial reports. *Proceedings of the 9th Language & Technology Conference "Human Language Technologies as a Challenge for Computer Science and Linguistics"*, Poznań, Poland, 44-48.

Pelicon, A., Martinc, M., & Kralj Novak, P. (2019). Embeddia at SemEval-2019 task : detecting hate with neural network and transfer learning approaches. *Proceedings of 13th Workshops on Semantic Evaluation [co-located with the] 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2019)*, Minneapolis, Minnesota, 600-606.

Martinc, M., Škrlj, B., & Pollak, S. (2019). Who is hot and who is not? Profiling Celebs on Twitter : Notebook for PAN at CLEF 2019. *Working Notes of CLEF 2019: Conference and Labs of the Evaluation Forum*, Lugano, Switzerland.

Martinc M., Škrlj, B., & Pollak, S. (2019). Fake or not: Distinguishing between bots, males and females. *Working Notes of CLEF 2019: Conference and Labs of the Evaluation Forum*, Lugano, Switzerland.

Repar, A., Martinc, M., & Pollak, S. (2018). Machine learning approach to bilingual terminology alignment : reimplementation and adaptation. *Proceedings of the 4REAL 2018 Workshop on Replicability and Reproducibility of Research Results in Science and Technology of Language*, Miyazaki, Japan, 1-8.

Martinc, M., & Pollak, S. (2018). Reusable workflows for gender prediction. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, Miyazaki, Japan, 515-520.

Repar, A., Martinc, M., Žnidaršič, M., & Pollak, S. (2018). BISLON: BISociative SLOgaN generation based on stylistic literary devices. *Proceedings of the Ninth International Conference on Computational Creativity, ICCC 2018*, Salamanca, Spain, 248-255.

Dobrovoljc, K., & Martinc, M. (2018). Er... well, it matters, right? On the role of data representations in spoken language dependency parsing. *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, Brussels, Belgium, 37-46.

Martinc, M, Škrlj, B., & Pollak, S. (2018). Multilingual gender classification with multi-view deep learning : Notebook for PAN at CLEF 2018. In: CAPPELLATO, Linda (ed.). *Working Notes of CLEF 2018 : Conference and Labs of the Evaluation Forum*, Avignon, France.

Biography

Matej Martinc was born on March 13, 1985 in Kranj, Slovenia. First, he studied philosophy and sociology of culture at the Faculty of Arts at the University of Ljubljana, where he defended his bachelor thesis "Not superiority, but equality of cultures at the time of globalization" in 2011. He continued his studies at the University of Ljubljana's Faculty of Computer and Information Science, which he completed with the bachelor thesis "Effective natural language processing with Python" under the supervision of Prof. Dr. Matjaž Kukar in 2015.

After a one-semester Erasmus internship at the Technical University of Cartagena, Spain, where he designed and implemented a system for visualization of the university data, he started working at the Jožef Stefan Institute, Slovenia, and in 2016 enrolled in the PhD programme "Information and Communication Technologies" at the Jožef Stefan International Postgraduate School under the supervision of Assist. Prof. Dr. Senja Pollak.

His research is focused on natural language processing and he authored a number of publications on the topic of author profiling, readability prediction, keyphrase extraction and semantic shift detection. He is currently researching how neural transfer learning approaches and contextual embeddings can be combined with symbolic methods in order to tackle these problems more efficiently. In addition, he has been participating in a number of computer science shared tasks, where he twice ranked as 2nd in author profiling shared tasks (PAN 2017 and CLIN 2019).

He is currently involved in two European Union's Horizon 2020 research projects, EMBEDDIA and SAAM and national projects TermFrame, CANDAS, Formica 2 and RSDO.