# TEXT MINING FOR CROSS-DOMAIN KNOWLEDGE DISCOVERY

Matjaž Juršič

**Doctoral Dissertation**
**Jožef Stefan International Postgraduate School**
**Ljubljana, Slovenia, May 2013**

**Evaluation Board**:

*Assist. Prof. Dr. Tomaž Erjavec,* Chair, Jožef Stefan Institute, Ljubljana, Slovenia

*Prof. Dr. Tanja Urbančič,* Member, University of Nova Gorica, Nova Gorica, Slovenia

*Prof. Dr. Hannu Toivonen,* Member, University of Helsinki, Helsinki, Finland

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL


Matjaž Juršič

# TEXT MINING FOR CROSS-DOMAIN KNOWLEDGE DISCOVERY

## Doctoral Dissertation

# RUDARJENJE BESEDIL ZA MEDDOMENSKO ODKRIVANJE ZNANJA

## Doktorska disertacija

*Supervisor*: Prof. Dr. Nada Lavrač

*Co-Supervisor*: Prof. Dr. Bojan Cestnik

Ljubljana, Slovenia, May 2013

# Contents

# Abstract

One of the prevailing tendencies in science is research over-specialization, resulting in deep but relatively isolated islands of knowledge. Due to the huge amounts of scientific information produced at an increasingly fast pace, it has become difficult to follow even the specific literature limited to a single domain of specialization. On the other hand, it is well known that many complex problems require knowledge from different domains to be combined in generating their solutions. However, scientific literature all too often remains limited and cited only within a single research sub-community, which makes searching for cross-domain scientific connections even harder. The problem of isolation of scientific domains on the one hand and the vast amount of available knowledge on the other hand are clear motivations for the work presented in this thesis.

The thesis proposes a solution to cross-domain knowledge discovery, which alleviates the stated problem of over-specialization by helping scientists to find promising pathways when combining different domains of interest. In particular, we deal with the problem of cross-domain knowledge discovery from text, when a scientist already assumes two topics or domains that need to be addressed jointly. In such cases, our methodology suggests the most adequate terms for bridging the two domains of interest and gives hints concerning the most promising pathways, potentially leading to new discoveries.

This work presents four main contributions to the problem of cross-domain knowledge discovery through text mining. First, an important text preprocessing step, word lemmatization, was significantly improved in terms of accuracy and efficiency. Second, a new CrossBee methodology was developed for discovering and ranking bridging terms, according to their potential to lead to new cross-domain scientific discoveries. Third, the CrossBee methodology was implemented in an executable workflow in a novel browser-based workflow construction and execution platform ClowdFlows, which enables methodology adaptation and reuse. Finally, yet importantly, an online web application was developed; it has an advanced user interface, which supports the end-user when applying the developed CrossBee methods to the domains of choice.

The first main contribution of this thesis is a novel multi-lingual lemmatization engine. We developed LemmaGen, a general purpose, accurate and very efficient publicly available lemmatizer trained on large lexicons of multiple languages. LemmaGen contains also a learning engine, which can be retrained to effectively generate lemmatizers for new languages.

The second contribution of this work is the CrossBee (Cross-Context Bisociation Explorer) methodology for cross-domain knowledge discovery by means of discovering and ranking domain bridging terms. We present specially designed heuristics that assign a quality estimate, a bisociation score, to each bridging term candidate. Furthermore, the methodology is ensemble-based, employing a set of heuristics to ensure robustness and stable performance across a varie-

ty of different datasets. Next, this ensemble is used for bridging term identification and ranking for cross-domain knowledge discovery from texts.

The third contribution of this thesis is the implementation of the LemmaGen and CrossBee modules as reusable software components in a complex ClowdFlows workflow. This enables experiment repeatability, software reuse, workflow adaptation and augmentation with new modules, ensuring the system's sustainability and reuse by the interested research community.

The CrossBee methodology is implemented in an online system, which presents the fourth important contribution of this thesis. In addition to ranking bridging terms in accordance with the developed methodology, CrossBee web application provides additional functionality that helps the scientist not only to find bridging hypotheses but also to check supportive evidence for them. CrossBee user interface supports user's creativity, as there are many different views and support tools the user can take advantage of when navigating the data.

The novel methodology for cross-domain knowledge discovery using text mining presented in this thesis and implemented in CrossBee has been trained by simulating the discoveries previously published in the literature-based discovery area, rediscovering the connections between migraine and magnesium domains. The approach was tested on the problem of rediscovering connections between autism and calcineurin domains. Finally, this work concludes with a critical evaluation of the developed methodology and interfaces, while discussing possible improvements and directions for further work.

# Povzetek

Čedalje večja specializacija raziskav, znanstvenikov in znanstvenih področij je prevladujoči trend na področju znanosti. Povečujoča specializacija vodi v poglobljene, a hkrati zelo izolirane otoke znanja, čeprav je znano, da mnogo kompleksnih problemov zahteva združevanje znanj z različnih področij. Ker se znanstvene informacije kopičijo čedalje hitreje, je dandanes težko slediti že eni sami znanstveni disciplini, iskati meddomenske povezave pa je še neprimerno težji problem. Poleg tega ostajajo znanstvene discipline ponavadi zaprte in omejene, saj se avtorji objavljenih prispevkov velikokrat medsebojno citirajo zgolj znotraj ožje skupnosti, ki pripada samo določeni znanstveni disciplini. Problem izolacije znanstvenih področij na eni strani in velike količine razpoložljivega a neuporabljenega znanja na drugi strani predstavlja motivacijo za delo, predstavljeno v tej doktorski disertaciji.

Osrednji namen disertacije je razviti metodo za odkrivanje meddomenskega znanja, ki bi znanstvenikom pomagala najti obetavne smeri pri povezovanju domen in s tem ponudila pristop k rešitvi problema prevelike specializacije. V delu se ukvarjamo z rešitvijo, ki temelji na uporabi metod rudarjenja besedil za odkrivanje meddomenskih povezav, v primeru ko znanstvenik že ve, kateri dve domeni ga pravzaprav zanimata in bi ju rad povezal. Naša metodologija tu predlaga najboljše besede oz. koncepte, ki nakazujejo na potencialno najboljše načine premostitve dveh doslej ločenih domen, kar lahko vodi do novih odkritij.

Kot odgovor na predstavljene izzive predstavi disertacija štiri glavne prispevke, ki z uporabo tehnik rudarjenja besedil pomagajo pri odkrivanju meddomenskega znanja. Prvi prispevek je znatna hitrostna in kakovostna izboljšava postopka lematizacije besed, kar predstavlja pomemben del priprave podatkov za nadaljnje rudarjenje besedil. Nato smo razvili novo metodologijo imenovano CrossBee, ki nam pomaga pri odkrivanju in ocenjevanju kvalitete besed s stališča njihovega bisociativnega potenciala za odkrivanje novega znanja med domenami. Sledi implementacija CrossBee metodologije v obliki delotokov v spletnem okolju za gradnjo in izvajanje delotokov ClowdFlows. Zadnji pomembnejši prispevek disertacije je implementacija naprednega spletnega uporabniškega vmesnika, ki z uporabo CrossBee metodologije uporabniku pomaga pri iskanju povezav med izbranimi domenami.

Prvi cilj našega dela je novo večjezično lematizacijsko orodje LemmaGen, ki je javno dostopno ter primerno za splošno uporabo. Lastnosti LemmaGena so še velika učinkovitost in točnost lematizacije besed. Prav tako smo izdelali modele za vrsto evropskih jezikov ter omogočili uporabnikom enostavno dodajanje jezikov na podlagi učenja modelov na novih podatkih, ki jih posreduje uporabnik.

Drugi cilj tega dela, ki predstavlja tudi jedro doktorske disertacije, je bil razvoj nove metodologije, ki z uporabo tehnik rudarjenja besedil pomaga pri odkrivanju meddomenskih zakonitosti v podatkih. Metodologija se imenuje CrossBee (Cross-Context Bisociation Explorer / raziskovalec meddomenskih bisociativnih povezav) in je bila razvita z uporabo novih, posebej zasnovanih hevristik za odkrivanje bisociativnega potenciala vsake besede v besedilu. Nadalje

smo te hevristike združili v ansambel hevristik, kar je privedlo do povečane robustnosti in stabilnosti metode pri uporabi na drugih množicah podatkov. Ansambel hevristik je bil uporabljen kot osnovna metoda za iskanje in ocenjevanje bisociativnega potenciala besed oz. konceptov in služi kot podlaga za meddomensko odkrivanje znanja.

Tretji cilj je bil osredotočen na zasnovo in implementacijo posameznih gradnikov celotne CrossBee metodologije v obliki delotokov v spletnem okolju za gradnjo in izvajanje delotokov ClowdFlows. Takšna implementacija omogoča ponovljivost poskusov, ponovno uporabo posameznih gradnikov, prilagajanje delotokov, izboljševanje delotokov z novimi moduli, zagotavljanje trajnosti razvitih metod ter možnost, da zainteresirani uporabnik sam ponovno uporabi sistem.

Četrti cilj je bila implementacija CrossBee metodologije v istoimenski spletni aplikaciji, ki poleg urejanja besed oz. konceptov glede na njihov bisociativni potencial omogoča tudi številne druge predstavitve podatkov, ki pomagajo uporabniku tako pri iskanju hipotez o meddomenskih povezavah, kot tudi pri preverjanju podpornih trditev za odkrite hipoteze. CrossBee uporabniški vmesnik je zasnovan tako, da podpira uporabnikovo ustvarjalnost, saj obstaja veliko različnih pogledov na podatke in podpornih orodij, ki jih lahko uporabnik s pridom izkoristi pri preiskovanju predlaganih rešitev.

Nova metodologija za odkrivanje meddomenskega znanja s pomočjo rudarjenja besedil CrossBee je bila ocenjena v okviru simulacije odkrivanja že znanih in v znanstveni literaturi objavljenih meddomenskih zakonitosti. S tem postopkom smo metodologijo razvili na podatkih meddomenskih povezav med migreno in magnezijem, preizkusili pa smo jo na iskanju meddomenskih povezav med avtizmom in kalcineurinom. Pričujoča doktorska disertacija se zaključi s kritično oceno razvite metodologije in vmesnika CrossBee ter predlaga nabor možnih izboljšav za nadaljnje delo na tem področju.

# Abbreviations

| | |
|---|---|
| AI | Artificial Intelligence |
| ADC | Annotated Document Corpus |
| AUC | Area Under Curve |
| BISON | BISOciation Networks for creative information discovery |
| BoW | Bag-of-Words |
| CRISP-DM | CRoss Industry Standard Process for Data Mining |
| CrossBee | CROSS context Bisociation ExplorEr |
| CS | Centroid Similarity |
| CSF | Creative Systems Framework |
| CST[1] | Creativity Support Tools |
| CST[2] | Center for SprogTeknologi (a lemmatizer name) |
| DM | Data Mining |
| ELRA | European Language Resources Association |
| FAQ | Frequently Asked Questions |
| GNU | GNU's Not Unix |
| HCI | Human-Computer Interaction |
| HTML | Hyper Text Markup Language |
| IE | Information Extraction |
| IR | Information Retrieval |
| KDD | Knowledge Discovery in Databases |
| LATINO | Link Analysis and Text mINing toolbOx |
| LBD | Literature Based Discovery |
| LDC | Linguistic Data Consortium |
| LGPL | GNU Lesser General Public License |
| LSI | Latent Semantic Indexing |
| MeSH | Medical Subject Headings |
| MSD | MorphoSyntactic Description |
| Multext | Multilingual text tools and corpora |
| Multext-East | Multext for central and Eastern European languages |
| NLP | Natural Language Processing |
| PoS | Part-of-Speech |
| RDR | Ripple Down Rule |
| RF | Random Forest |
| ROC | Receiver Operating Characteristic |
| ROCCH | ROC Convex Hull |
| SEO | Search Engine Optimizations |
| SOAP | Simple Object Access Protocol, |
| SQL | Structured Query Language |
| SVM | Support Vector Machine |
| TF-IDF | Term Frequency – Inverse Document Frequency |
| TM | Text Mining |
| UI | User Interface |
| UMLS | Unified Medical Language System |
| URL | Uniform Resource Locator |
| WSDL | Web Service Definition Language |

# 1 Introduction

This chapter places our work into a broader context of data mining, provides the rationale for choosing the given research topic and states the thesis contributions to science. It first provides an outline and the background of the topic addressed, followed by the motivation of the work, explicit problem statement, the presentation of the research hypothesis, the research goals, and the scientific contributions of the thesis. It concludes by presenting the structure of the rest of the thesis.

## 1.1 Cross-domain knowledge discovery

Data mining (Hand et al., 2001, Han and Kamber, 2006) is a field of study in computer science, and artificial intelligence in particular. Data mining is the main step of broader process of Knowledge Discovery in Databases (KDD) (Fayyad et al., 1996, Cios et al., 2007). The aim of data mining is to find classification models and/or novel, interesting and useful patterns in data either for predictive or descriptive purposes. As such, data mining lightly overlaps with the field of statistics. Text mining (Feldman and Sanger, 2007) is a subfield of data mining, which focuses on discovering new information from text document collections, mainly by using techniques from data mining, machine learning, natural language processing, and information retrieval. Inspired by recent research in bisociative knowledge discovery (Berthold, ed., 2012), this thesis addresses a specific text analysis problem, referred to as *cross-domain knowledge discovery*, that focuses on the problem of extracting novel and valuable information at the intersection of initially separate domains.

The key aim of this thesis is to develop a new methodology and tools for cross-domain knowledge discovery by, building upon approaches from two distinct areas: on the one hand, computational creativity research (Colton and Wiggins, 2012) together with the notion of bisociations introduced by Koestler (1964), which provide a theoretical background for our research, and on the other hand, text mining and in particular literature-based discovery (Swanson, 1986a, Weeber et al., 2001), which serves as a practical basis for our developments.

More specifically, the theoretical grounds of this thesis lie in the creativity research literature on domain-crossing associations, called *bisociations*, that were introduced already by Arthur Koestler in his book *The act of creation* (Koestler, 1964). According to Koestler, a bisociation is a result of a mind process when making completely new associations between concepts from contexts (domains, categories, or classes) that are usually considered separate. Using the concept of bisociation enables us to formulate the overall goal of this thesis as the development of a methodology for bisociation discovery, since bisociations can be considered to represent novel and valuable cross-domain links.

The idea of bisociation was revived and put into a new perspective of automated knowledge discovery as the core underlying concept of the European project BISON (BISOciation Net-

works for creative information discovery, 2008–2011, Online reference [36]). In the project, the initial approach relied on transforming all the available data into the form of a large, heterogeneous information network, where a bisociation was defined as an indirect link among network nodes, which indicates new, yet unexplored knowledge in the data. This thesis, in contrast to the BISON's approach, relies on a different feature-vector-based document representation, and proposes a methodology for retrieving potential bisociations directly from text documents.

From the perspective of practical implementation, our approach to bisociation discovery resembles the approach developed in the field of literature-based discovery (LBD). LBD was introduced by Don R. Swanson (1986a) who investigated the idea of searching for new findings from public knowledge and discovered actual examples of hidden knowledge linking two previously separate domains (Swanson, 1986b). In LBD, authors try to solve problems which are very similar to bisociation discovery, however, the vocabulary and underlying theories for describing the problem and processes are different. According to the LBD terminology (Weeber et al., 2001), the specific type of knowledge discovery problems addressed in this thesis is referred to as a *closed discovery problem* of linking two separate domains of interest.

In a closed discovery setting, one starts with two predefined separate domains, with a research hypothesis that the two domains can be connected in some novel and valuable way. In the core step of closed discovery, the search for supportive evidence for this hypothesis is performed by investigating the available literatures about the two domains. As suggested by Swanson (1988), this can be done by identifying interesting bridging terms appearing in both literatures, which bear the potential of indirectly connecting the two domains under investigation. Although being time-consuming, searching for terms appearing in both literatures is not the main problem. The main issue that also motivated the research presented in this thesis is the fact that a list of terms shared by the two literatures can be very long. Estimating which of the terms have higher potential for stimulating discoveries is an interesting research question, important for practical applications.

In summary, while the creativity research and bisociative knowledge discovery research provide a theoretical background, the LBD research provides a more practical insight into the types of problems that were handled in the past. In this thesis we combine both aspects to make further advances in those areas.

## 1.2 Motivation

As the world has been moving into a new era of globalization over the last decades, information that was once scarce and jealously guarded has now become increasingly available. This paradigm shift—supported by the advances in new online technologies—is even more obvious in recent years with the raise of initiatives for sharing data, knowledge and resources in projects like Wikipedia, Open source development, Creative Commons, Open education, and many others. Although the principle of sharing ideas and resources is not new in the scientific community, it is now increasingly applied due to the Open access scientific publication policy.

On the other hand, a long known problem in science is its ongoing fragmentation into smaller and smaller scientific subdisciplines in order to cope with increasing amounts of information. Even though the divide-and-conquer principle worked well in the past, nowadays overspecialized professionals often seek ways to reconnect the disconnected scientific disciplines since they have realized that great new discoveries can result from domain-crossing explorations.

This thesis tries to combat the resulting situation of constant increase in publicly available data and knowledge on the one hand and very limited capabilities of humans to deal with all this knowledge on the other hand. We present a new methodology which tries to alleviate this problem by providing a partially automated extraction of interesting information and consequently helping the experts who—in their search for new findings—have to deal with vast amounts of scientific information from unrelated domains.

To further exemplify our motivation and the path that we chose to follow towards the overall aim, we cite Don R. Swanson, the originator of the field of literature-based discovery. After more than 20 year of experience, after having founded the field and after numerous highly cited articles, Swanson (2008) wrote:

> *The goal of a literature-based discovery system should be to stimulate human creativity in order to produce a plausible and testable hypothesis stated in a form suitable for publication in the subject field studied, where it is then open to testing, criticism, review, and stimulation of further research.*

Accordingly, developing a methodology and providing a creativity stimulating system is the guideline for the research presented in this thesis.

## 1.3 Problem statement

This thesis addresses the task of using text mining for cross-domain knowledge discovery with the aim of discovering bisociations. This overall goal can be further divided into four subproblems addressed in the thesis: (a) improving text preprocessing, (b) designing the methodology for discovering bisociations, (c) implementing the methodology in a modular way by creating reusable software components, and (d) implementing a creativity stimulating user interface to support the bisociative knowledge discovery process.

The first part of the problem addressed in this thesis is the problem of improving a selected text preprocessing step, i.e., lemmatization. Lemmatization is a preprocessing step for a number of language engineering and text mining tasks, among which there is also the cross-domain bisociation discovery task. The problem of lemmatization deals with finding the normalized forms of inflected words appearing in text. A similar method to lemmatization is word stemming; however, we find lemmatization to be more appropriate for our task since it generates terms in a normalized format (as can be found in dictionaries), which is more appropriate for hypothesis generation. The provided solution to lemmatization has to be efficient, accurate and extensible.

The second part of the problem (considered to be the core part of the problem addressed in this thesis) is to develop a novel methodology with all the needed ingredients for corpus-based bisociation discovery, along with theoretical justifications, implementation, and evaluation. The methodology should be as much as possible domain independent and clearly specified through a sequence of well-defined steps assisting bisociation discovery. The addressed problem is in the field of literature-based discovery known as the closed discovery problem: when having two separate domains in the form of two sets of text documents, the task is to find bridging terms or concepts, which stimulate the scientific creativity in order to generate hypotheses about novel and valuable connections linking both domains.

The third part of the problem concerns the implementation of the designed methodology. The goal is to make the developed software components reusable by a wider community of interested researchers. To achieve reusability, the methodology needs to be modular, and the environment needs to support user-friendly composition of the implemented building blocks. Due to the specified requirements, the implementation needs to be compatible with at least one of the so-called workflow execution environments where the user is able to experiment with different settings and layouts of the methodology.

The fourth part of the problem revolves around designing and implementing a creativity stimulating user interface, which should support the user in searching and specifying domain bridging hypotheses as well as support efficient validation of these hypotheses. The user interface should comply with the basic principles of creativity supporting software tools by providing many different views on the data and propose different exploration paths to choose from.

In summary, the four sub-problems are combined into the problem of creating a modular methodology implemented as a set of reusable components along with a creativity stimulating user interface for the purpose of helping users to find new discoveries at the intersection of otherwise separate domains.

## 1.4 Hypothesis and goals

The hypothesis investigated in this thesis states that the use of a domain-independent methodology comprising text mining approaches and creativity supporting interfaces facilitates the extraction of existing, but hidden and till now unexamined knowledge in the form of bisociative links connecting two separate collections of documents. These links across domains can indicate, for example, a solution to a problem in one domain, which can be applied in another domain, but was not yet explored by scientists because of their high specialization in their own domain of expertise. The discovered bisociations may trigger new ideas, which could further lead to new discoveries in a given domain.

The main hypothesis contains two assumptions which are confirmed in this work. The first assumption is that bisociative terms, which indicate novel links between domains and have not been previously explored, indeed exist in documents. The second assumption states that the bisociative terms exhibit some distinct property, which enables us to find and promote them. Such property could be, for instance, a slightly different statistical distribution of domain occurrences of a bisociative concept in comparison to domain occurrence distribution of a non-bisociative concept.

The main goals of the thesis are split into four main areas which correspond to the four parts of the problem statement provided in the previous section, i.e., improved lemmatization, a new methodology, a modular implementation of the methodology, and a creativity supporting online user interface. The four overall goals can be further elaborated as follows:

1. Improvement of selected text representation and preprocessing techniques to better suit the goal of bisociation discovery.
   (a) Improved multi-lingual lemmatization algorithm LemmaGen.
   (b) Training and evaluation of the algorithm on a large, publicly available Multext and Multext-East corpora and comparison with other publicly available lemmatizers.

   (c) Implementation of the lemmatization algorithm in various applicable forms, as a software library, web services, and reusable components in an online workflow management system.
   (d) Creation and maintenance of the website that offers the basic information about lemmatization and LemmaGen system, serves downloadable open source libraries with short instructions for use, and provides a working demo of the algorithm lemmatizing selected user provided texts.
2. Development of a new methodology for cross-domain knowledge discovery from text corpora.
   (a) State-of-the-art overview of bisociation discovery, text mining and literature-based discovery, with a focus on cross-domain knowledge discovery.
   (b) The design and development of a new domain-independent methodology for cross-domain knowledge discovery using text mining techniques.
   (c) Evaluation of the methodology using examples of successful past cross-domain discoveries.
3. Implementation of a modular methodology in the form of reusable software components and workflows.
   (a) Methodology conceptualization to support modular implementation as a workflow comprising of independent and reusable building blocks.
   (b) Implementation of components of the developed methodology as widgets in an online workflow management and execution environment ClowdFlows (Kranjc et al., 2012).
   (c) Implementation of these components in other reusable forms, i.e., as web services and a software library of components.
   (d) Implementation of the entire methodology as a workflow in ClowdFlows.
4. The design and implementation of a web application supporting creative cross-domain knowledge discovery.
   (a) A state of the art review of creativity supporting user interfaces design.
   (b) Design and implementation of web application *CrossBee* (Cross-context Bisociation ExplorEr) supporting the cross-domain knowledge discovery process.
   (c) Development and/or adaptation of visualization techniques supporting the bisociation discovery process.
   (d) Testing, evaluation and modification of the system in cooperation with the end users in order to adapt the system to their actual discovery process.
   (e) Evaluation and application of the methodology using the online system in a new problem domain.

## 1.5 Scientific contributions

The scientific contributions of this thesis are as follows:

1. Adaptation and improvement of existing core text mining and text representation techniques to suit the specific goal of bisociation discovery. The main contribution is the developed LemmaGen engine for multi-lingual lemmatization, which is the key text preprocessing step enabling the application of the whole methodology on datasets in different languages.

2. Definition of a bisociation discovery task in the context of cross-domain information extraction and literature mining.
3. Development of the domain-independent methodology for bisociative knowledge discovery, which takes as input two document collections (from two separate domains) and outputs a ranked list of terms of which the top ones have higher probability of representing bisociative links between the two domains.
4. Implementation of the methodology in several forms, in particular as a workflow of reusable components in an online workflow execution environment ClowdFlows. The other two implementations are an open source software library and a publicly available collection of web services.
5. Design and development of an online web application CrossBee which not only implements the cross-domain knowledge discovery methodology but also includes a sophisticated user interface enabling users to explore their bisociative hypotheses by efficiently navigating through the data. CrossBee has been designed by considering some principles of creativity supporting tools and therefore stimulates the user's creativity even further, compared to the basic methodology approach.

The scientific contributions of this work were published in the following papers, and book chapters:

- Advances in lemmatization:
  - Juršič, M.; Mozetič, I.; Lavrač, N. Learning ripple down rules for efficient lemmatization. In: *Proceedings of the 10th International Conference Information Society* **1**, 206–209 (IJS, Ljubljana, Slovenia, 2007).
  - Juršič, M.; Mozetič, I.; Erjavec, T.; Lavrač, N. LemmaGen: Multilingual Lemmatization with Induced Ripple-Down Rules. *Journal of Universal Computer Science* **16**, 1190–1214 (2010).
- Methodology for cross-domain knowledge discovery:
  - Juršič, M.; Lavrač, N.; Mozetič, I.; Podpečan, V.; Toivonen, H. Constructing information networks from text documents. In: *Proceedings of the Workshop on Explorative Analytics of Information Networks at ECML PKDD.* 23–26 (ECML/PKDD Organization Committee, Bled, Slovenia, 2009).
  - Juršič, M.; Mozetič, I.; Grčar, M.; Cestnik, B.; Lavrač, N. Identification of concepts bridging diverse biomedical domains. In: *BMC bioinformatics supplements (short paper)* **11**, 4.1–4.2 (BMC Bioinformatics, London, UK, 2010).
  - Lavrač, N.; Sluban, B.; Juršič, M. Cross-Domain Literature Mining through Outlier Document and Bridging Concept Detection. In: *Proceedings of the Workshop on Analysis of Complex Networks at ECML PKDD.* 35–47 (ACNE Committee, Barcelona, Spain, 2010).
  - Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating outliers for cross-context link discovery. In: *Artificial intelligence in medicine: proceedings.* Lecture Notes in Computer Science **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).
  - Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the Power of Outliers for Cross-Domain Literature Mining. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 325–337 (Springer, Heidelberg, Germany, 2012).

  - Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging Concept Identification for Constructing Information Networks from Text Documents. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 66–90 (Springer, Heidelberg, Germany, 2012).
  - Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Bisociative Literature Mining by Ensemble Heuristics. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 338–358 (Springer, Heidelberg, Germany, 2012).
- Bisociativity supporting visualization and exploration web application:
  - Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Cross-domain literature mining: Finding bridging concepts with CrossBee. In: *Proceedings of the 3rd International Conference on Computational Creativity.* 33–40 (University College Dublin, Dublin, Ireland, 2012).
  - Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. HCI Empowered Literature Mining for Cross-Domain Knowledge Discovery. In: *Proceedings of the International Conference on Human Factors in Computing & Informatics.* Lecture Notes in Computer Science **7947**, 124–135 (SouthCHI Committee, Maribor, Slovenia, 2013).

## 1.6 Thesis structure

The thesis is structured as follows: Chapters 1, 2 and 3 present the overview of this thesis, the background work and the employed background technologies, Chapter 4 provides an overview of the methodology developed in this thesis, Chapters 5 and 6 are the core methodological parts presenting the LemmaGen and CrossBee methodologies, Chapter 7 illustrates the developed user interfaces and implementations, and Chapter 8 concludes the thesis by providing extensive discussion and directions for further work. Detailed structure is described below.

Chapter 1 places our work into a broader context of data mining and provides the motivation, problem statement, hypothesis, goals and scientific contributions of this work.

Chapter 2 introduces the background of this thesis and describes the related work. In summary, the following areas/problems are presented: text mining, lemmatization, computational creativity, bisociation research, creative information discovery framework, creativity support tools and literature mining. The chapter concludes by unifying the two perspectives—a bisociation-based theoretical perspective and a text mining practical perspective—and provides a common framework for this thesis.

Chapter 3 offers a basic introduction to the background technologies—i.e., document acquisition, typical text preprocessing steps, and term/document representation—that are used as important ingredients of this work.

Chapter 4 provides a conceptual and procedural top-level overview and a workflow-based representation of the methodology for cross-domain knowledge discovery through text mining.

Chapter 5 presents in detail the first core part of this thesis—the LemmaGen approach to lemmatization. Section 5.1 introduces the basic concepts, Section 5.2 defines the problem of lemmatization, Section 5.3 describes the LemmaGen input and output formats, Sections 5.4 and 5.5 present the LemmaGen learning and lemmatization algorithms respectively, Section 5.6 describes the application and evaluation of LemmaGen on thirteen different European language datasets and compares LemmaGen to other available lemmatizers, and finally, Section 5.7

sketches the implementation of the LemmaGen approach in an online workflow execution environment ClowdFlows.

Chapter 6 presents in detail the second core part of this thesis—the CrossBee approach to bridging term discovery, i.e., the mechanism for term scoring and ranking, based on their bisociation potential. Section 6.1 introduces the base heuristics as the basic ingredients of the methodology, Section 6.2 evaluates the base heuristics using the gold standard dataset, Section 6.3 shows the approach of using an ensemble of heuristics instead of a single heuristic for bisociativity scoring, Section 6.4 presents the evaluation of the ensemble-based term scoring functionality, and finally, Section 6.5 illustrates the implementation of the CrossBee methodology as a detailed ClowdFlows workflow.

Chapter 7 shows the two web applications, LemmaGen and CrossBee, and provides a systematic categorization of the multiple implementations relevant to this thesis.

Chapter 8 concludes this thesis by summarizing the main contributions and proposing many directions for further work, presented along with an extensive discussion about the current state of the research presented in this thesis.

# 2 Background and related work

In this chapter, we introduce the background of this thesis and describe the related work. First, we introduce the area of text mining and present important key concepts, with the emphasis on lemmatization, which is thoroughly presented due to being one of the contributions of this work. Next, we introduce computational creativity as the theoretical background of our research. In particular, we show the fundamentals of creativity research via bisociations, present the details of bisociations formalized in the creative information discovery framework, and link our work to the field of creativity support tools. Next, we present the area of literature mining as the text mining application area which shares many goals with our work. Finally, we unify these two perspectives—a bisociation-based theoretical perspective and a text mining practical perspective—and provide a common framework for this thesis.

## 2.1 Text mining

Text mining is an area of research that deals with digging knowledge out of unstructured textual resources for goals like document retrieval, categorization, summarization, automated ontology construction, question answering, sentiment analysis, visualization, and others, mainly with the purpose of solving (textual) information overload problem. Text mining (sometimes also referred to as text data mining or text analytics) is usually described as a subfield of data mining (Han and Kamber, 2006), nevertheless, it can be also perceived as a more multidisciplinary field drawing its techniques from data mining, machine learning, natural language processing (NLP), corpus-based computational linguistics, information retrieval (IR), information extraction (IE) and knowledge management (Feldman and Sanger, 2007).

From the historical perspective, automated processes that resemble nowadays text mining were first studied already in 1950's with the purpose of document summarization (Luhn, 1958). However, the main principles from IR that are now considered to be the foundations of text mining were developed in 1970's and 80's (Salton and Buckley, 1988; Jones, 2003). Some of these main principles are, for example, IDF measure for term specificity (Jones, 1972), vector space document model, TF-IDF term weighting, and, cosine document similarity (Salton et al., 1975; Robertson and Jones, 1976).

From a procedural perspective, at the most abstract level, a text mining process usually follows the CRISP-DM reference model for data mining (Shearer, 2000). The CRISP-DM process model proposes six phases when working on a data mining project: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Text mining can be distinguished from general data mining by special procedures applied in the data preparation phase. In text mining, the data preparation phase needs to convert unstructured or poorly structured texts into organized data, structured as a table of instances (rows) with attributes (columns). Such a table of instances can be in the modeling phase used by many standard or

slightly adopted data mining algorithms which uncover various pieces of information about the data, for example, implicit grouping of instances, classification rules, categorization of new instances and similar.

### 2.1.1 Key text mining concepts

The key concepts in text mining are a corpus or a document collection, a single document, and document features (Feldman and Sanger, 2007). A *document collection* is any grouping of text documents in which we want to discover patterns. Even though sizes may vary from a couple of hundreds to millions of documents, it is usually true that from the IE perspective more is better. Similarly, a single *document* may vary in size from a single sentence to a whole book, and represents a single textual data unit within a collection. To enable the data mining modeling phase, each document needs to be represented as a set of *document features* it contains. In the most general case, when one is dealing with totally unstructured text, features are derived from text using only text preprocessing methods; however, a feature may be any information that can be assigned to a document (e.g., author, date of publication, document length, assigned keywords, and similar).

Most commonly used document features are words, terms and concepts that appear in text. *Words* are the simplest features which carry no additional semantic value and represent exact words tokenized from text. One step higher on the semantic scale are *terms* which are sets of one or more words occurring together and are already normalized, filtered and grouped together by using various term extraction methods. The features with the most semantic value are *concepts* which are not only terms occurring in the text but also some related, but not necessary mentioned notions (e.g., a document describing some specific car may not actually include the word "car", however, the concept feature "car" could be assigned to such a document). One of the important characteristics of the described document features is the fact that they are usually *sparse* (Feldman and Sanger, 2007) meaning that most of the features in a vector have zero weight. This sparsity is due to the fact that there are many different features (words, terms, or concepts) in the document collection; yet, a single document contains only a small subset of them.

Since high quality features are hard to acquire, all possible methods that could improve this process should be used at this point. The approach that usually helps the most, consists of incorporating background knowledge about the documents and their domain. The most straightforward technique to incorporate background knowledge is to use a controlled vocabulary. A controlled vocabulary is a lexicon of all terms that are relevant in a given domain. Here we can see a major difference when processing general documents as compared to scientific documents. For many scientific domains, there exists not only a controlled vocabulary, but also a pre-annotation for a lot of scientific articles. In this case, one can easily create feature vectors, since the terms as well as the concepts are already predefined. Other interesting approaches to identifying concepts include methods such as KeyGraph (Ohsawa et al., 1998), which extracts terms and concepts with minimal assumptions or background knowledge. Yet, another alternative is using domain ontologies that could be, for example, semi-automatically retrieved by a combination of tools such as TermExtractor (Sclano and Velardi, 2007) and OntoGen (Fortuna et al., 2008).

The main task of the data preparation phase (in text mining typically called the preprocessing step) is to convert a collection of documents into a table of instances, thus converting every document into a table row representation of an instance. The typical approach is to use the vector space model (Salton et al., 1975) where every document feature represents one attribute of an instance (one dimension in the vector space which maps to one table column). Thus, the preprocessing step is of extreme importance in text mining as the dataset is being constructed out of unstructured data. Feldman and Sanger (2007) even state that text mining is to a large degree defined by the elaborate preprocessing techniques.

The approach presented in this thesis focuses on the preprocessing step of the text mining process; therefore we will not be thoroughly describing the data mining steps that follow. In a nutshell, as long as the document feature sparseness is handled correctly, the majority of classical data mining approaches can be used after preprocessing is finished. At this point, we will also not go into detail about the standard preprocessing techniques as a specific application heavily depends on a concrete text mining problem. However, in Chapter 3 we present the actual preprocessing techniques used in our methodology.

This thesis bases its techniques on term-based document features. Nevertheless, we may sometimes name them as concepts when we want to emphasize that they may or should represent a concept observed by a domain expert. As mentioned above, preprocessing text in order to retrieve terms requires normalizing words, grouping them and filtering out non-relevant constructs. One of the important contributions of this thesis is the improvement of a specific word normalization procedure called lemmatization, which produces dictionary forms of words from inflected word forms found in text. Lemmatization is part of the data preparation phase of the methodology, which in fact defines the dataset that will be used in all the later stages.

### 2.1.2 Specific text preprocessing step: Lemmatization

Lemmatization is a valuable preprocessing step for a large number of language engineering and text mining tasks. It is the process of finding the normalized forms of wordforms, i.e., inflected words as they appear in text. For example, in English the lemma of wordforms *dogs* is *dog*, of *wolves* is *wolf*, of *sheep* is *sheep*, of *looking* is *look* and of *took* is *take*.

Traditionally, lemmatization rules were hand-crafted. However, machine learning approaches to morphological analysis and lemmatization became an increasingly interesting research subject. Machine learned lemmatizers are robust and can handle out-of-vocabulary words; they can be trained on large datasets and thus have large coverage; and it is easier and cheaper to acquire training datasets than rules. For example, a researcher in multilingual language processing, who needs the functionality of lemmatization, can acquire lexicons for many languages via the European Language Resources Association ELRA (Online reference [37]) or the U.S. based Linguistic Data Consortium LDC (Online reference [38]), train the learner on them, and use the resulting lemmatizers for text processing.

The problem of stemming and lemmatization was addressed already in the 1960's (Lovins, 1968). Hand-crafted morphological analyzers (which, as a side-effect, could also produce lemmas of wordforms) have been developed for a number of languages. Due to the existence of high-coverage, precise and fast hand-crafted analyzers, such as the well-known Porter stemmer (Porter, 1980), which is considered a de facto standard for English, and its successor, the multilingual Snowball stemmer (Online reference [44]), stemming and lemmatization were often

taken as solved problems. However, these systems (often using various methods such as finite state automata or transducers to compress hand-crafted rules into a resulting language model) have several shortcomings: lemmatizers do not do well on out-of-vocabulary words, and they are expensive to construct. There are still languages without such an infrastructure, they are difficult to adapt to language varieties, and are quite often not publicly available, such as the stemmer for Slovene described in (Popovič and Willett, 1990).

Traditional hand-coded rules in grammars of natural languages typically obey the Elsewhere condition (Kiparsky, 1973): "In cases where more than one rule is applicable, the most specific rule should apply". Hence, traditional hand-coded lemmatization rules are ordered, with exceptions coming first, followed by more general rules. This principle has been followed also in most machine learning approaches to learning lemmatization rules: a rule induction system ATRIS (Mladenić, 1993, 2002a), if-then classification rules and Naive Bayes (Mladenić, 2002b), a first-order rule learning system CLOG (Manandhar et al., 1998), the CST lemmatizer (Dalianis and Jongejan, 2006), and the Ripple Down Rule (RDR) learning approach (Plisson et al., 2008).

The last three of these systems deserve a special mention. The first-order decision list learner CLOG (Manandhar et al., 1998), described and evaluated in (Erjavec and Džeroski, 2004; Džeroski and Erjavec, 2000; Erjavec and Sárossy, 2006), relies on having information from a part-of-speech (PoS) tagger; at a cost of lower efficiency, PoS tagging information allows CLOG to attain a high accuracy (note, however, that such a tagger is not available for all languages). The CST lemmatizer (Dalianis and Jongejan, 2006) is one of the few trainable lemmatizers that is available for download (see Online reference [11]). Therefore, we were able to directly compare its results with the results obtained by our lemmatizer, LemmaGen. The RDR lemmatizer (Plisson et al., 2008) is another publicly available (see Online reference [12]), which was—like LemmaGen—inspired by the Ripple Down Rule learning methodology for the GARVAN-ES1 expert system maintenance (Compton and Jansen, 1988), where the idea was that new rules are incrementally added to the system when new examples of decisions are available. However, new examples might contradict the already existing rules, therefore exceptions to the original rules have to be added as well. When executed, rules are "fired" top-down until the most specific applicable rule fires, thus obeying the Elsewhere principle mentioned above.

## 2.2 Computational creativity

The core research of this thesis is based on the notion of creativity which is used as a fundamental concept to provide the theoretical rationale of the developed methodology. In this section, we present the theoretical background supporting the idea that creative knowledge discovery is in its essence tightly connected to the idea of domain-crossing connections.

Creativity is one of human's basic mental abilities. Yet, the concept of creativity has proven to be very difficult to capture. One of the reasons for non-existing commonly accepted definition of creativity is also the large number of fields which share an active interest into the creativity paradigm and span from social sciences to natural sciences, e.g., philosophy, sociology, education, literature, arts, cognitive science, linguistics, psychology, technology, and others. In addition, the scientists whose research concerns creativity, more often than not drop the explicit definition of creativity from their publications as shown on a selection of 90 papers about creativity by Plucker et al. (2004). Moreover, the definitions of creativity that researchers provide are very diverse; hence, a comprehensive overview is beyond the scope of this thesis.

In the following sections, we provide some definitions of creativity, which are the most relevant to this work. Most emphasis is given to the Arthur Koestler's notion of creativity as bisociation as well as its formalization in the context of creative information exploration paradigm, which represents the definition of the creativity used in this work. Besides the concept of bisociation, we present also a few other attempts to define creativity or at least the means through which the creativity could be achieved or evaluated. Most of these definitions come from the emerging field of computational creativity and from the community researching creativity support tools. In both of these fields, authors try to define creativity from the computational perspective and are, therefore, useful for the conceptualization of our work.

## 2.3 Koestler's bisociative creativity

Koestler's idea of creativity presents the foundation of our work. It was conceived already in the 1960's and published in the book *The Act of Creation* (Koestler, 1964), where Koestler investigates the principle of creativity by studying three human cognitive processes, i.e., comic inspiration, scientific discovery, and artistic originality. He theorizes that the logic patterns of these creative processes are similar in all three cases and that only emotional climate is different in each case. To represent the shared pattern of creativity, Koestler coined a term *bisociation* which is meant to characterize the basic process occurring in a person's mind when being creative or similarly, when being presented with a creative act. Due to the frequent usage and relative importance of the concept of bisociation in this thesis, we describe this rather philosophical idea in more detail in the following paragraphs.

Koestler's notion of bisociation is about making a new interaction between two "independent matrices of perception or reasoning". Translated to our language: a bisociation is a new piece of knowledge connecting two independent domains in a novel way. This is exactly what we want to achieve in our work. Depending on the type of the interaction/connection—i.e., matrices collision, matrices fusion or matrices confrontation—and based on the emotional climate, the resulting effects may be laughter, new scientific discovery or aesthetic experience, respectively. For the purpose of this thesis, we are only interested in new scientific discoveries across independent domains and therefore, in the domain fusing type of bisociative interaction.

Figure 2.1 illustrates two examples of creativity based on bisociations by sketching two intersecting planes $M_1$ and $M_2$ (two domains or in Koestler words two "independent matrices of perception") and a new creation (e.g., idea or discovery) $L$ at the intersection of both domains. Koestler uses the concept of bisociation (sketched as a connection bridging the intersecting planes) to explain all his real-life examples of creativity among which he presents also scientific discoveries.

Probably the most well-known example of a creative discovery is the story of Greek scholar Archimedes who was given a task to determine whether the crown made for his king was made of pure gold or not. No matter how hard Archimedes tried to find the solution, he was always returning to the source problem of measuring the volume of the irregularly shaped object (see right hand image in Figure 2.1 where the path on $M_1$ plane represents futile quest for the answer in the $M_1$ domain). However, when he took a bath and saw how water rose as he got in (changing his domain of reasoning to the independent $M_2$ plane of taking a bath and to the physics connected to this) he instantly found the solution. The story further says that Archimedes, due to excitement, ran outside naked crying "Eureka!" and later he proved—using the

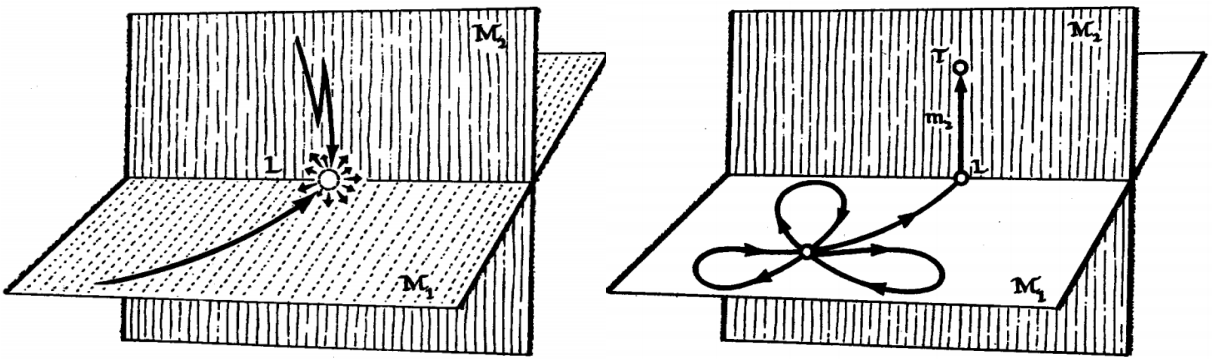



Figure 2.1: *Koestler's visual presentation of bisociation.* Two among many representations of bisociation provided by Koestler (1964). The left figure illustrates an example of humorous bisociation in the moment of grasping the same situation $L$ from two different perspectives $M_1$ and $M_2$. The right figure gives an example of a new scientific discovery by taking a problem from domain $M_1$ (where one cannot find a solution) and connecting it through an idea $L$ to a new domain $M_2$ where a solution $\tau$ to the original problem can be found by following the path of reasoning $m_2$.

discovered principle—that the crown, indeed, was not made of pure gold. Besides the Archimedes story, Koestler presents also many other examples of similar scientific discoveries, which he attributes to the bisociative pattern, e.g., Louis Pasteur discovering the mechanism of vaccination, André-Marie Ampère and Karl Friderich Gauss and their mathematical discoveries, Friedrich August von Kekulé "dreaming" about organic molecule structures, and many others.

Koestler concludes that "creative act is not an act of creation in the sense of the Old Testament. It does not create something out of nothing; it uncovers, selects, re-shuffles, combines, synthesizes already existing facts, ideas, faculties, skills. The more familiar the parts, the more striking the new whole". The presented Koestler's statement is one of the main guidelines for our work since it directly supports our hypothesis assuming that cross-domain links may contain novel knowledge if selected, re-shuffled, combined and synthesized using the right methodology.

### 2.3.1 Creative information discovery framework

The almost half a century old Koestler's notion of bisociation was revived in 2007 (Berthold et al., 2008) and became the fundamental research paradigm of the European project BISON (BISOciation Networks for Creative Information Discovery, see Online reference [36]). For the purpose of this project, the concept of bisociation was re-examined in the context of the framework for creative information exploration. Furthermore, Dubitzky et al. (2012) and Berthold (2012) have upgraded the Koestler's definition to be mathematically more precise and to suit the needs of the computational creative information discovery paradigm.

The following summarized concepts are originally provided by Dubitzky et al. (2012) and are used in the subsequent bisociation definition. They are important for this work and some of them are extensively used in the following sections.

- *An intelligent agent or a reasoner (R)* is somebody or something (e.g., a program), which makes reasoning inside one or more knowledge bases using the available concepts.
- *A concept* denotes a single cognitive unit of meaning, sometimes also referred to as "unit of knowledge". It is normally associated with a corresponding representation or encoding in a particular language. Concepts are usually representations of many real-world entities

(like objects, ideas or events), which share similar properties (features, dimensions), therefore, can be effectively represented by only one concept definition instead of the whole set of all known entities. For example, the concept of a "book" represents all book entities so that an intelligent agent does not need to think about each and every known book when reasoning about this concept.

- *A domain* is perceived as a formal or common sense topic, subject area, field of interest, for example, scientific discipline, social, cultural, economic or political topic, game, common pattern of activities, and so on.
- *A domain theory* $D_i$ defines a set of concepts that are associated with a particular domain $i$. A domain theory represents a commonly accepted notion of a domain, which is shared across a peer group and is therefore fixed or changing very slowly. This is especially true for historically established, mature domains with large communities, e.g., traditional scientific fields.
- *An agent-specific knowledge base* $K_i^R$ is a subset of a domain theory $D_i$, i.e., $K_i^R \subseteq D_i$ containing the concepts from $D_i$, which are available to an intelligent agent $R$ at the time point $t$ when reasoning about the domain $i$. Each intelligent agent $R$ has exactly one knowledge base for a given domain at a specific time point, which is either empty, if the agent knows no concept in that domain, or non-empty otherwise.
- *A habitual incompatibility* is a relation between two knowledge bases $K_i^R$ and $K_j^R$ when there is no concept $c; c \in K_i^R \bigwedge c \in K_j^R$, that is perceived simultaneously in both knowledge bases at a given point in time $t$.

Using these concepts, Dubitzky et al. (2012) formally defined the notion of bisociation for the needs of computational creative information discovery paradigm as follows:

- A *bisociation* is an event at a given point in time $t$, when all concepts related to a concrete problem are perceived simultaneously in (beforehand) habitually incompatible knowledge bases $K_i^R$ and $K_j^R$.

Bisociation, as specified by Dubitzky et al. (2012), is the formal definition and the model for cross-context knowledge discovery, which this thesis builds upon. Let us compare the definition of bisociation to the definition of the more widely known concept of association that is provided using the same terminology:

- An *association* is an event at a given point in time $t$, when all concepts related to a concrete problem are perceived simultaneously in a knowledge base $K_i^R$.

Obviously, the only difference between association and bisociation is the origin of the simultaneously perceived concepts originating respectively from a single knowledge base or from two habitually incompatible knowledge bases. Thorough examination also reveals that a single bisociation exists in one point in time only, as, in the next point in time, the knowledge bases $K_i^R$ and $K_j^R$ become habitually compatible due to the established connection. We conclude the presentation of these background definitions with our own observation: a bisociation that proves itself valuable tends to grow into a future association when domain theories $D_i$ and $D_j$ (i.e., communities supporting these domain theories) recognize the importance of the novel insight provided by this bisociation.

### 2.3.2 Computational creativity research

Bisociation, as described in the previous two sections, is only one among many alternative definitions of creativity. By following our primary goal, i.e., cross-domain knowledge discovery through creativity, we are interested also in other characterizations, which enable us to encode such creativity into a programmable strategy (e.g., a heuristic). Some ideas on how to do this, or at least how to evaluate it, can be found in the literature from the emerging multidisciplinary field of *computational creativity*, which is a subfield of *artificial intelligence* (AI) (Colton and Wiggins, 2012). Computational creativity focuses on computational systems exhibiting behaviors, which an unbiased observer would deem to be creative and whose creations (artifacts and ideas) are novel and valuable. The tasks of understanding and modeling creativity may prove to be one of the hardest tasks in AI (Colton and Wiggins, 2012).

The dominating perception of the creativity theory in the computational creativity community is provided by Margaret Boden, whose book *The Creative Mind: Myths and Mechanisms* (Boden, 2004) attained almost a biblical status in the field (Forth et al., 2010). The majority of contributions to the field, especially those developing theoretical grounds, cite Boden's work. Forth et al. (2010) also state that Boden's theory is attractive as it takes a modern and more formal approach to cognitive science, compared to e.g., Koestler's theory.

We present the Boden's theory of creativity as it is central in the community of computational creativity to which our work is related. In the methodology evaluation part of this thesis, we provide a more extensive discussion on how our work fits in the field of computational creativity. There are three main perspectives from which we can observe the theory of creativity by Boden: the first is the definition itself, the second is looking at the types of creativity, and the last is the categorization of creative processes. In the next three paragraphs, we present these perspectives.

Boden (2004) defines creativity as "the ability to come up with ideas or artifacts that are *new*, *surprising* and *valuable*". The majority of authors agree and build on top of this definition, however, some of them (e.g., Higgins, 2005; Wiggins, 2006) leave out the "surprising" statement. For example, Wiggins (2006) claims that surprise is more a property of a receiver of a creative output: it is an emotion generated by the novelty of the output. For our purposes, we also use only "new and valuable" in the definition. We believe that every artifact or idea, which is new and highly valuable, will generate surprise in an interested receiver—if the novelty is so obvious that it does not generate surprise, then why did it not exist before, especially since it is supposed to be of high value. Based on this definition, we show in the evaluation that the outputs of our methodology are both new and valuable; therefore, the developed system may be considered to be creative.

Furthermore, Boden (1998, 2004) defines two types of creativeness: P-creativity (P for psychological) representing the creativity from the creator's viewpoint—generated artifacts and ideas that are new to the creator—and H-creativity (H for historical) representing creativity from the viewpoint of the society—generated artifacts and ideas that have arisen for the first time in the human history. We share the opinion that AI should primary concentrate on P-creativity (Boden, 1998), hence also our approach will be evaluated in terms of P-creativity as shown later. Nevertheless, in the sciences outside the creativity research typically only H-creativity is appreciated as rediscovering old ideas is considered a bad practice caused by the lack of background research.

When it comes to types of creative processes or, in other words, the forms of creativity, Boden (2004) distinguishes three forms:

- Combinatorial creativity involves making unfamiliar combinations of familiar ideas. "The novel combination requires a rich store of knowledge in the person's mind, and many different ways of moving around within it."
- Exploratory creativity enables formation of a completely new thing but using the principles that are known and established. An example would be the creation of a new painting by using very well known (e.g., school taught) style of painting. So the creations are actually instances which obey well-defined rules of its object's type.
- Transformational creativity is perceived as the hardest as it moves the boundaries of allowed procedures how things can be created. An example of this could be an invention of a new style of painting that would become highly valued over time.

Note that, by definition, the approach based on bisociations (Koestler, 1964; Dubitzky et al., 2012) adopted as our primary theoretical background, is categorized as a form of combinational creativity. Nonetheless, Koestler confidently explains many real-life examples of creativity, which actually fit into either exploratory or transformational form of creativity. This shows that the exact relationship between the Koestler's and Boden's models is complex and, above all, beyond the scope of our research.

There are some efforts to take the Boden's theory even further towards a formal definition (e.g., Wiggins, 2006; Ritchie, 2012). These researches relate to the Boden's theory similarly as the formalization of Dubitzky et al. (2012) relates to the Koestler's theory. Especially Wiggins (2006) and his elaborate formalization of the Creative Systems Framework (CSF) provides a very valuable approach to make greater use of creativity theories in practice when designing actual systems. We note the value of CSF applicability to our work. This would certainly help us better evaluate our system from the perspective of computational creativity as well as possibly guide our research into more "creative" directions. Nevertheless, we leave this formalization for future work.

With this we conclude the introduction of the computational creativity field, which will be referenced later in the evaluation section of this thesis. However, some authors in the field of computational creativity admit (e.g., Veale et al., 2006, 2010) that even though there is an abundance of theories of creativity to choose from, when it comes to building real experimental creative systems, usually, theory inevitably takes a back-seat as it provides little technical details. Similar observation is true for our work where the underlying bisociation theory serves mainly as a guidance and a conceptual framework for the study, investigating whether cross-domain knowledge discovery can provide valuable outcomes when applied in the context of literature mining.

### 2.3.3 Creativity support tools

To develop a methodology and the corresponding interfaces resulting in a truly creative system is a very ambitious goal, especially since creativity itself can be understood in many different ways. However, even the less ambitious goal of developing systems which empower the user to become more creative, is already a non-trivial problem, addressed by the community of researchers developing *creativity support tools*. This section provides a short introduction to the

field and shows why our methodology and the developed user interface align well with this background.

Creativity support tools (CST) are a topic of research inside the field of human-computer interaction (HCI), which is an interdisciplinary field at the intersection of computer science, behavioral sciences and domain specific sciences. The goal of HCI is to improve the interactions between the users and the computers by designing user interfaces, which minimize the barrier between human's cognitive model of the task they want to accomplish and the computer's understanding of this task. Traditionally, HCI focuses on developing productivity support tools for the users, so that they can perform their work more rapidly, effectively, and with fewer errors (Shneiderman, 2009).

In addition to increasing productivity, CST research deals with a greater challenge of enabling users to also increase their creative outputs. CST enhance the users capability to make discoveries or inventions from early stages of gathering information, hypothesis generation, and initial production, to the later stages of refinement, validation, and dissemination (Shneiderman, 2007). This fits completely with our intention when designing the user interface and the methodology presented in this thesis. The following paragraphs present both an informal as well as the formal creativity support tools design guidelines. A more detailed relation between these definitions and our implementation of the interface is established in the evaluation section of this thesis where we show how well our work complies with the CST paradigm.

The following citation from Resnick et al. (2005) informally summarizes the aims in designing CST: "Our goal is to develop improved software and user interfaces that empower users to be not only more productive, but more innovative. Potential users of these interfaces include software and other engineers, diverse scientists, product and graphic designers, architects, educators, students, and many others. Enhanced interfaces could enable more effective searching of intellectual resources, improved collaboration among teams, and more rapid discovery processes. These advanced interfaces should also provide potent support in hypothesis formation, speedier evaluation of alternatives, improved understanding through visualization, and better dissemination of results. For creative endeavors that require composition of novel artifacts (e.g., computer programs, scientific papers, engineering diagrams, symphonies, artwork), enhanced interfaces could facilitate exploration of alternatives, prevent unproductive choices, and enable easy backtracking."

Shneiderman (2007, 2009) provides a structured set of CST design principles, which we consider a formal set and use it to evaluate our solution later in this thesis:

- *Support exploration.* To be successful at discovery and innovation, users should have access to improved search services providing rich mechanisms for organizing search results by ranking, clustering, and partitioning with ample tools for annotation, tagging, and marking.
- *Enable collaboration.* While the actual discovery moments in innovation can be very personal, the processes that lead to them are often highly collaborative.
- *Provide rich history-keeping.* The benefits of rich history-keeping are that users have a record of which alternatives they have tried, they can compare the many alternatives, and they can go back to earlier alternatives to make modifications. History-keeping on computers has even more benefits, such as sharing interesting cases and creating repeatable processes that can be run on new data.

- *Design with low thresholds, high ceilings, and wide walls.* CST should have steep learning curve for novices (low threshold), yet provide sophisticated functionality that experts need (high ceilings), and also deliver a wide range of supplementary services to choose from (wide walls).

Although we have presented the CST field and tried to establish relevant links to our designed interfaces, we need to repeat a note expressed by Shneiderman et al. (2006): "CST is a research topic with high risk but potentially very high payoff". Therefore, despite the fact that a good creativity-supporting interface would indeed be very useful in our use case, there is still high probability that the provided solution would not meet the required criteria. However, we believe the user interface we designed, i.e., the web application CrossBee, indeed meets the majority of the specified design principles.

## 2.4 Literature mining

Up until now, we presented the selected of theoretical backgrounds supporting the idea that effective cross-domain knowledge discovery can be achieved using creativity via the bisociation principle. However, besides a theoretical framework and some design ideas, little can be gained from studding creativity fields alone. This section introduces the next important aspect needed to develop a practical solution for cross-domain knowledge discovery, i.e., the practical approaches to retrieving knowledge from multiple domains. The field that deals with this problem is called *literature-based discovery* and was initiated by Don R. Swanson in the 1980's.

### 2.4.1 Swanson's model of information retrieval

In 1986, Don R. Swanson initiated an idea which gradually evolved into an entire field of research. The idea was about undiscovered public knowledge (Swanson, 1986a) which is assumingly hidden in huge scientific literature databases due to the separation of scientific disciplines. Public knowledge is supposed to be present but hidden in the form of non-interactive (not intersecting) clusters, even though there inherently exist many logical relations among them. Observing from different perspectives, this can be perceived either as the problem of professional overspecialization, as the inability of coping with increasing pace of producing overwhelming amounts of information, or as the problem of lacking a good information retrieval (IR) system that would enable finding relevant cross-domain information. Swanson proposes to advance the latter by sketching some principles, which can be used to systematically uncover this knowledge.

Swanson placed his theory to test by applying it on to the medical domain to find such connections. The first connection—a blood viscosity and circulatory disorder—was found between Raynaud's disease and dietary fish oil (Swanson, 1986b) and the challenging question was: could fish oil be used to treat Raynaud's disease? The second example of undiscovered public knowledge he proposed was a link between migraine headache and magnesium deficiency (Swanson, 1988). Later, many clinical tests were performed (e.g., Digiacomo et al., 1989; Weaver, 1990) and majority of them have proven both connections as viable (Swanson, 1990, 1993). This success resulted in further research; in the following years five other similar examples were published (Swanson and Smalheiser, 1996) and various extensions of the methodology were developed by the original authors and other researchers.

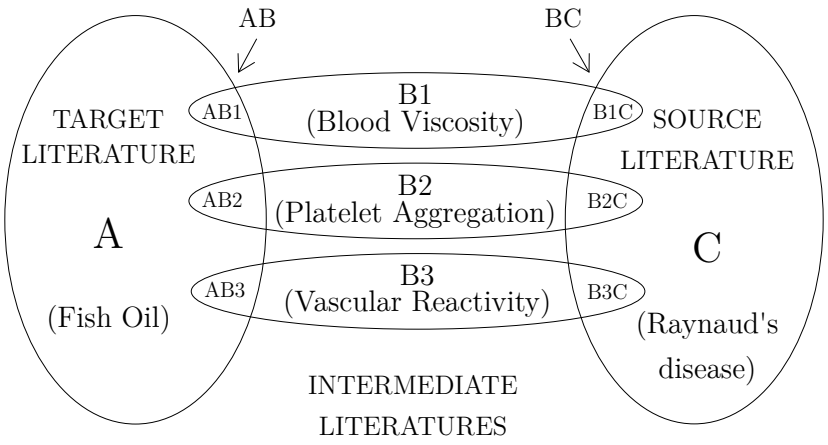



Figure 2.2: *Venn diagram representing Swanson's first discovery.* The so-called ABC literature discovery model overlaid with Raynaud's disease to fish oil connection discovery. Adaptation of a diagram from (Swanson and Smalheiser, 1997)

Further research involved specifying more general approaches of the search procedures for uncovering hidden public knowledge and designing a set of interactive software and database search strategies, collectively called ARROWSMITH (Swanson and Smalheiser, 1997; Smalheiser and Swanson, 1998). Especially the latest online implementation (Smalheiser et al., 2009) of ARROWSMITH resembles our methodology and interface greatly, therefore, we provide a more detailed comparison in the evaluation section of this thesis. Swanson and colleagues found many prosperous links that were beforehand hidden in the literature (Swanson and Smalheiser, 1996) and also found some older examples of methodologies resembling very much to literature-based discovery (Swanson, 2008), which were of extraordinary importance to science. Nevertheless, Swanson (2008) admits that literature-based discovery has its limitations, independent of specific computer techniques used to implement it, since information retrieval is, in essence, incomplete and the size of recorded knowledge base is far beyond human capacity to assimilate.

Methodologically, Swanson's approach is based on finding sets of complementary literatures that are mutually isolated and non-interactive (they do not share authors, cite each other and are not co-cited) but are somehow related or connected. To formalize how these connections could be found, Swanson (1986b) designed the ABC model, which can be explained by the following example. Let the literature about agent A report that *A causes B* and the complementary literature about disease C report that *B influences C*. In the case, when literatures A and C are also non-interactive, one may conclude that a plausible relation between A and C through B presents a novel knowledge unknown to either of the disciplines studying A or C.

A practical example of the ABC principle is sketched in Figure 2.2, which shows an adaptation of a diagram by Swanson and Smalheiser (1997) that is commonly cited (e.g., by Weeber et al., 2001; Chen, 2003; Petrič et al., 2009) and shows the ABC literature-based discovery principle overlaid by Swanson's first discovery about Raynaud's disease. The diagram shows the ABC model at providing evidence that the source of the problem C (Raynaud's disease) can be connected to possible solution A (fish oil) through intermediate literatures B1 (blood viscosity), B2 (platelet aggregation), and B3 (vascular reactivity), even though A and C initially do not intersect (are non-interactive).

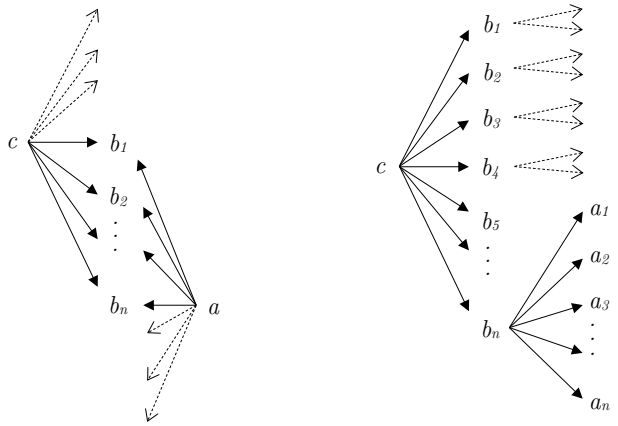



Figure 2.3: *Closed (left) and open (right) discovery process.* As defined by Weeber et al. (2001).

The data source in the majority of Swanson's experiments—and experiments of his followers including our work—is MEDLINE (see Online reference [39]), the world's largest bibliographic database maintained by the U.S. National Library of Medicine, which currently (January 2013) contains over 20 million journal citation and abstracts from biomedical and life sciences literature from around the world.

### 2.4.2 Literature-based discovery

Continuing the description of the research presented in the previous section, literature-based discovery (LBD) is thus the area of research initiated by the Swanson's idea of undiscovered public knowledge. Swanson (2008) sees LBD as a mechanism to support and enhance human research by focusing on finding promising pairs of scientific articles that can serve as a stimulus for identifying associated literature structures.

The standard classification of approaches in LBD is provided by Weeber et al. (2001) who defines *open* and *closed discovery* processes as an integral part of LBD. In a closed discovery setting, the researcher postulates in advance which are the two literatures (A and C) of interest. The closed discovery setting (sketched in Figure 2.3 left) helps to find the intermediate arguments supporting the ABC relation modeling. The supporting concepts or terms are usually based on the arguments from literature B, and are therefore called *B-terms*. This naming enables us to sometimes use the expression *bridging terms* when referring to B-terms, as the role of B-terms is to bridge two separate domains. On the other hand, in open discovery (sketched in Figure 2.3 right), the researcher starts from only one domain, usually from the problem domain C, and tries to find a suitable solution by revealing connection with a separate non-interactive literature A, which is yet to be discovered by using intermediate B-terms. Open discovery can be seen as scientific hypothesis generation while closed discovery represents hypothesis testing and validation (Weeber et al., 2001).

Open and closed discovery have been widely investigated and many LBD support tools have been developed:

- ARROWSMITH was one of the first semi-automatic procedures developed to support LBD. Initially, it was envisioned as a set of interactive search strategies (Swanson and Smalheiser, 1997; Smalheiser and Swanson, 1998) along with a web platform designed to

assist the user in both open and closed discovery. Later, a new version named "ARROWSMITH two-node search interface" (Smalheiser et al., 2006, 2009) was developed, focusing only on closed discovery. It has a much more sophisticated user interface which simplifies and automates the search for interesting B-terms. ARROWSMITH is tightly connected to the MEDLINE database and uses a collection of cleverly engineered heuristics to provide a short and ordered list of potential B-terms, which is finally provided to the user for exploration.

- LBD system was an early approach to develop a system supporting open discovery. In a series of articles, Gordon and Lindsay (1996) first developed the software which implements open discovery using standard text mining approaches and measures. Next, Gordon and Dumais (1998) improved it with Latent Semantic Indexing, whereas, in later experiments Lindsay and Gordon (1999) tested how lexical statistics influence the discovery process. Gordon et al. (2001) tried the developed techniques on a database other than MEDLINE by experimenting with World Wide Web as the data source. Unfortunately, we are not aware of any working publicly accessible online or standalone version of the software produced by this extensively discussed research using advanced text mining methods.
- BITOLA is an open and closed biomedical discovery support, which is available online and uses a heterogeneous approach by incorporating many available information sources to enrich the discovery process. A common ground for all the versions of BITOLA is to use association rules as a relation measure and fast association rule mining (Agrawal et al., 1996) for discovering these relations. The first version (Hristovski et al., 2001) was a standalone application that incorporated of controlled vocabulary thesaurus MeSH (Online reference [41]) and a collection of knowledge sources and tools UMLS (Online references [42]) to improve its results. The next version (Hristovski et al., 2005) was web-based and included additional background knowledge about the chromosomal locations of diseases and genes from various sources and therefore became even more accurate for such discoveries. Furthermore, Hristovski et al. (2006, 2008) incorporated also two external NLP systems that provide semantic relations, which further enhanced the discovery process.
- RaJoLink is another system supporting both the open and closed discovery, and is specialized to the biomedical domain employing the MEDLINE database (Petrič et al., 2009; Urbančič et al., 2009). RaJoLink has been successfully applied to uncover novel relations in the domain of autism (Urbančič et al., 2007; Petrič, 2009; Macedoni-Lukšič et al., 2011). The system is a standalone application but connects to MEDLINE to automatically retrieve documents. It uses the principle of term rarity (in contrast to other approaches) to guide the search towards promising literature A to be connected to C to solve the problem. It also uses MeSH to filter them. Lately, the authors focused on the role of document outliers (Petrič et al., 2012a) and the bisociativity principle (Petrič et al., 2012b) to improve their methodology.
- LitLinker was designed by Pratt and Yetisgen-Yildiz, having open discovery in mind. Initially (Pratt and Yetisgen-Yildiz, 2003), LitLinker used knowledge-based methodology, natural-language processing as well as association rules (Agrawal et al., 1996) to mine the biomedical literature for new links. Later versions (Yetisgen-Yildiz and Pratt, 2006) dropped natural language processing due to being computationally too expensive.

LitLinker shares with BITOLA not only the association rule principle but also the incorporation of MeSH and UMLS to limit the search space. For finding the best B-terms, LitLinker applies a statistical approach that is based on the background distribution of term probabilities and extensively uses a medical knowledge base to prune uninteresting correlations. LitLinker is implemented as an online system; however, it is no longer publicly accessible.

- Other related systems, which are not discussed in detail since they are either unavailable, are very specialized or are not well documented, include Manjal (Sehgal et al., 2008), IRIDESCENT (Wren et al., 2004), eVOC (Tiffin et al., 2005), and G2D (Perez-Iratxeta et al., 2005).

Even though the majority of research in the two decades of LBD research is based on the same ABC principle, there is still a great variety of different approaches to the problem (Weeber et al., 2005; Jensen et al., 2006). We observe that most of the LBD research represents applied works and only lately has the community tried to develop a common LBD framework. Sehgal et al. (2008) proposed a four dimensional system, which allows specification of the four key methodological concepts: object, links, inference methods and additional data. Sehgal et al. (2008) also reviewed much of the existing literature and showed how each fits into the framework. Sehgal et al. state that such a framework would be helpful for establishing common evaluation measures and gold standard datasets in LBD.

Yetisgen-Yildiz and Pratt (2008, 2009) report that existing evaluation metrics usually comprise measuring precision, recall and F-measure, drawing Receiver Operating Characteristic (ROC) curves, or using probabilities showing that some methodology is better than random. Regarding the evaluation approaches, authors try to (a) replicate Swanson's experiments, which is the most common approach, (b) statistically evaluate how many discoveries made by their system in the past became realized later, (c) use expert opinion as evaluation, or (d) publish their findings in the medical literature domain. Yetisgen-Yildiz and Pratt (2009) created the evaluation methodology based on option (b), which is meant to help the researchers to compare approaches and ultimately help to improve the performance of LBD systems overall.

We conclude the presentation of this very relevant field of LBD by summarizing some thoughts from Smalheiser and Torvik (2008), which could easily be expressed also in the context of creative information discovery (Dubitzky et al., 2012). Though there are many flavors of LBD, they all share a more ambitious agenda than simply to extract or process the information present in a given text. Much of the research in text mining tries to identify explicit relationships present in text, however, LBD goes even further to identify relationships that are not only implicit but have been previously unnoticed. Thus, the field of LBD has set its goals on a very (perhaps impossibly) high standard: true, novel, non-trivial and generally cross-disciplinary scientific discoveries.

Table 2.1: *Unifying Koestler's and Swanson's models of creative knowledge discovery.* This table is adopted from the table originally published by Lavrač et al. (2010).

| Koestler's bisociation framework | | Swanson's ABC literature-based discovery |
|---|---|---|
| frames of reference (domains) M1 and M2 | ≈ | literatures A and C |
| self-consistent domain | ≈ | all available literature about a concept |
| habitually incompatible domains | ≈ | mutually isolated, non-interactive literatures |
| perceive simultaneously M1 and M2 | ≈ | study intermediate literatures AB and BC |
| bisociative link L in M1∩M2 | ≈ | connected arguments from B in A ∩ C |
| bisociative frames of reference | ≈ | closed literature-based discovery |

## 2.5 Unifying the computational creativity and text mining frameworks

As first postulated by Lavrač et al. (2010) and later recognized by Petrič et al. (2012a) and Juršič et al. (2012c), the background theories about creativity on one hand and text mining on the other hand can be, to some extent, combined into a unified framework. The two main conceptual frameworks presented in previous sections, the Koestler's theory of creativity through bisociations and the Swanson's ABC model for literature-based discovery have very similar underlying idea.

Starting with the Koestler's model, we observe that the emphasis on finding a bisociation is in perceiving of a situation or idea, L, in two self-consistent but habitually incompatible frames of reference, M1 and M2 (Koestler, 1964). Very similarly, Swanson (1990) claims that an unknown but possibly valuable connection can be discovered by reading two mutually isolated (non-interactive) literatures A and C and joining together their respective arguments. Indeed, both authors state similar claims but using a different vocabulary, e.g., two habitually incompatible frames of reference M1 and M2 may clearly represent two mutually isolated and non-interactive literatures A and C. Lavrač et al. (2010) identified this relation and provided a unified framework represented in a systematic table that summarizes the unification of both theories. For the purpose of this thesis, a slightly adjusted representation of the unified framework is provided in Table 2.1.

With this unification of the two core theories of this thesis we conclude the presentation of the related work. We presented the aspects that guided our research and the background work related to this thesis. The unifying theory, discussed in the last section, serves us not only as a theoretical guidance—which is mainly the case of the section about creativity review—but also as a practical guidance with concrete examples of methodologies already developed in the rich area of LBD. Yet, starting from two such diverse fields as creativity research and literature-based discovery and developing our system by building on both traditions, provides an excellent baseline for new developments presented in this thesis.

# 3 Text mining components

This chapter explains the background technologies that are used as ingredients of our methodology for bisociative knowledge discovery. The first section introduces several document acquisition options. The next section presents some typical text preprocessing steps and aligns them with our work. The last section concerns term and document representation, including weighting models and similarity measures from the area of text analytics, which are extensively used in the methodology part of this thesis.

## 3.1 Document acquisition

For the study presented in this thesis, we use only one data source, i.e., PubMed (see Online reference [40]), which is a service providing access to more than 22 million (January 2013) life sciences and biomedical citations of which the largest subset presents the MEDLINE database. In our research, PubMed was used to retrieve the datasets for the migraine-magnesium and autism-calcineurin domain pairs of documents. However, when experimenting with other domains, we identified the following text document acquisition scenarios, which are also supported in the developed tools:

- Loading locally stored files in various application dependent formats—this is the traditional setting in data mining; however, it usually requires large amounts of partly manual work for transforming the data between different formats.
- Acquiring documents using the SOAP web services (e.g., PubMed uses a SOAP web service interface to access their database).
- Selecting documents from the SQL databases.
- Crawling the internet for gathering documents from web pages (e.g., from Wikipedia).
- Collecting documents from snippets returned from web search engines.

## 3.2 Text preprocessing

This section describes text preprocessing techniques based on standard text mining concepts, which need to be explained for better understanding specific parts of our methodology. By explaining various aspects of preprocessing, this section extends the description of basic text mining concepts and terminology introduced in Section 2.1.1.

Technically, our implementation employs the *LATINO* (*Link analysis and text mining toolbox,* see Online reference [33]) software library for the text preprocessing needs. This library contains a majority of elementary text preprocessing procedures as well as a large number of advanced procedures used in text mining, e.g., clustering and classification algorithms.

Preprocessing is a very important part of any form of knowledge extraction from text documents. Its main task is the transformation of unstructured data from text documents into a predefined well-structured data representation. As shown below, preprocessing is inevitability

very tightly coupled with the extraction of terms. In our case, the actual B-term candidates are also defined in preprocessing. The subsequent scoring and ranking, however, orders the terms and removes the majority of lower ranked terms from the term set.

In general, the task of preprocessing is to construct a set of relevant *features* from text documents. The set of all features selected for a given document collection is called a *representational model*. Each document is represented by a vector of numerical values, one for each feature of the selected representational model. Using this construction, we get the most standard text mining document representation, called *feature vectors*, where each numeric component of a vector is related to a feature and represents a weight related to the importance of the feature in the selected document. Typically in such vectors, the majority of weights are equal to zero. This is the characteristic of text-based feature vectors (Feldman and Sanger, 2007), which are therefore often referred to as *sparse vectors*. The goal of preprocessing is to extract a feature vector for each document from a given document collection.

Commonly used document features are characters, words, terms and concepts (see Section 2.1.1 for more explanation). In the rest of this thesis, we do not distinguish between words, terms or concepts; we refer to all of them as terms.

The most common document representation model in text mining is the *Bag-of-Words* or *BoW* model (Feldman and Sanger, 2007). It uses all terms as features, therefore the dimension of the feature space is equal to the number of different terms in all of the documents.

A standard collection of preprocessing techniques (Feldman and Sanger, 2007) is listed below, together with a set of functionalities implemented in our methodology:

- *Tokenization*: the continuous character stream is broken into meaningful tokens, usually words or terms when a controlled vocabulary is available. Our system uses a standard Unicode tokenizer, which mainly follows the Unicode Standard Annex (see Online reference [43]) for Unicode text segmentation.
- *Stopword removal*: stopwords are predefined words from a language that usually carry no relevant semantic information (e.g., articles, prepositions, conjunctions etc.); the usual practice is to ignore them when building a feature set. Our implementation uses a predefined list of stopwords—common lists that are included in our library are taken from Snowball (see Online reference [44]).
- *Stemming or lemmatization*: the process that converts each word/token into the morphologically neutral form. The following alternatives have been made available in our system: Snowball stemmers, the Porter stemmer (Porter, 1980), and the one that we use, the LemmaGen lemmatization system (presented in Chapter 5).
- *Part-of-speech (PoS) tagging*: the annotation of words with the appropriate PoS tags based on the context in which they appear. PoS tagging is currently not used in our approach.
- *Syntactic parsing*: performs a full syntactic analysis of sentences according to a given grammar. Usually shallow (not full) parsing is used since it can be efficiently applied to large text corpora. Syntactic parsing is currently not used in our approach.
- *Term extraction*: methods that identify which words should be extracted as terms and which should not. Term extraction by grouping words into terms using n-grams (an n-gram is a sequence of n items from a given sequence) has been implemented in our approach. The argumentation why, for the purpose of this work, we cannot apply more advanced term extraction is provided in the discussion chapter of this thesis.

| Documents | Extracted terms |
|---|---|
| $doc_1$ | $ent_1,\ ent_2,\ ent_3$ |
| $doc_2$ | $ent_3,\ ent_4,\ ent_4$ |
| $doc_3$ | $ent_1,\ ent_2,\ ent_2,\ ent_5$ |
| $doc_4$ | $ent_1,\ ent_1,\ ent_1,\ ent_3,\ ent_4,\ ent_4$ |

Original documents and extracted terms.

| Feature space | $ent_1$ | $ent_2$ | $ent_3$ | $ent_4$ | $ent_5$ |
|---|---|---|---|---|---|
| $doc_1$ | $w^f_{1:1}$ | $w^f_{1:2}$ | $w^f_{1:3}$ | | |
| $doc_2$ | | | $w^f_{2:3}$ | $w^f_{2:4}$ | |
| $doc_3$ | $w^f_{3:1}$ | $w^f_{3:2}$ | | | $w^f_{3:5}$ |
| $doc_4$ | $w^f_{4:1}$ | | $w^f_{4:3}$ | $w^f_{4:4}$ | |

Sparse matrix of documents: $w^f_{x:y}$ denotes the weight (in the feature space) of the term $y$ in the feature vector of document $x$.

| Document space | $doc_1$ | $doc_2$ | $doc_3$ | $doc_4$ |
|---|---|---|---|---|
| $ent_1$ | $w^d_{1:1}$ | | $w^d_{1:3}$ | $w^d_{1:4}$ |
| $ent_2$ | $w^d_{2:1}$ | | $w^d_{2:3}$ | |
| $ent_3$ | $w^d_{3:1}$ | $w^d_{3:2}$ | | $w^d_{3:4}$ |
| $ent_4$ | | $w^d_{4:2}$ | | $w^d_{4:4}$ |
| $ent_5$ | | | $w^d_{5:3}$ | |

Sparse matrix of terms: $w^d_{x:y}$ denotes the weight (in the document space) of document $y$ in the document vector of term $x$.

Figure 3.1: *Example of conversion between the feature and the document space.*

## 3.3 Term and document representation

The design choice of the methodology presented in this thesis is that candidate bridging terms will be the features of documents, i.e., the terms as defined in Section 2.1.1 and Section 3.2.

Candidate bridging terms need to be represented in a way which enables efficient calculation of different distance measures between the terms. We chose a representation in which a term is described by a set (vector) of documents in which it appears. In the same manner as documents are represented as sparse vectors of features (terms), the terms can also be represented as sparse vectors of documents. Take an illustrative example in Figure 3.1 where term $ent_1$ is present in documents $doc_1$, $doc_3$ and $doc_4$ and hence its feature vector consists of all these documents (with appropriate weights). By analogy to the original vector space—the *feature space*—the newly created vector space is named a *document space*.

Note that if we write document vectors in the form of a matrix, then the conversion between the feature space and the document space is performed by simply transposing the matrix (see Figure 3.1). Although we have developed the methodology for candidate B-term extraction, the analogy between the feature space and the document space allows us to use the developed approach also for candidate "B-document" extraction. However, practical experiments and conceptual reasoning of this kind of interpretation has not been done yet.

| | $ent_1$ | $ent_2$ | $ent_3$ | $ent_4$ | $ent_5$ |
|---|---|---|---|---|---|
| $doc_1$ | 1 | 1 | 1 | | |
| $doc_2$ | | | 1 | 1 | |
| $doc_3$ | 1 | 1 | | | 1 |
| $doc_4$ | 1 | | 1 | 1 | |

Binary weight

| | $ent_1$ | $ent_2$ | $ent_3$ | $ent_4$ | $ent_5$ |
|---|---|---|---|---|---|
| $doc_1$ | 1 | 1 | 1 | | |
| $doc_2$ | | | 1 | 2 | |
| $doc_3$ | 1 | 2 | | | 1 |
| $doc_4$ | 3 | | 1 | 2 | |

Term occurrence

| | $ent_1$ | $ent_2$ | $ent_3$ | $ent_4$ | $ent_5$ |
|---|---|---|---|---|---|
| $doc_1$ | $^1/_3$ | $^1/_3$ | $^1/_3$ | | |
| $doc_2$ | | | $^1/_3$ | $^2/_3$ | |
| $doc_3$ | $^1/_4$ | $^2/_4$ | | | $^1/_4$ |
| $doc_4$ | $^3/_6$ | | $^1/_6$ | $^2/_6$ | |

Term frequency

| | $ent_1$ | $ent_2$ | $ent_3$ | $ent_4$ | $ent_5$ |
|---|---|---|---|---|---|
| $doc_1$ | $(^1/_3)\cdot\log(^4/_3)$ | $(^1/_3)\cdot\log(^4/_2)$ | $(^1/_3)\cdot\log(^4/_3)$ | | |
| $doc_2$ | | | $(^1/_3)\cdot\log(^4/_3)$ | $(^2/_3)\cdot\log(^4/_2)$ | |
| $doc_3$ | $(^1/_4)\cdot\log(^4/_3)$ | $(^2/_4)\cdot\log(^4/_2)$ | | | $(^1/_4)\cdot\log(^4/_1)$ |
| $doc_4$ | $(^3/_6)\cdot\log(^4/_3)$ | | $(^1/_6)\cdot\log(^4/_3)$ | $(^2/_6)\cdot\log(^4/_2)$ | |

TF-IDF: term frequency – inversed document frequency

Figure 3.2: *Document vector feature weighting models.* Based on the example from Figure 3.1.

### 3.3.1 Weighting models

We identified the four most typical weighting models for assigning weights to features:

- *Binary*: a feature weight is 1 if the corresponding feature is present in the document, or zero otherwise.
- *Term occurrence*: a feature weight is equal to the number of occurrences of this feature. This weight might be sometimes better than a simple binary value since frequently occurring features are likely to be more relevant to the given text.
- *Term frequency*: a weight is derived from the term occurrence by dividing the vector by the sum of all vector's weights. The reasoning is similar to term occurrence, with normalization with respect to document size.
- *TF-IDF*: Term Frequency-Inverse Document Frequency is the most common scheme for weighting features. It is usually defined as:

$$w_{x:y}^{TFIDF} = \mathrm{TermFreq}(\mathrm{doc}_x, \mathrm{ent}_y) \cdot \log(\mathrm{N}/\mathrm{DocFreq}(ent_y)), \tag{1}$$

  where $\mathrm{TermFreq}(\mathrm{doc}_x, \mathrm{ent}_y)$ is the frequency of feature $\mathrm{ent_y}$ inside document $\mathrm{doc_x}$ (equivalent to term frequency defined in the bullet point above), N is the number of all documents and $\mathrm{DocFreq}(\mathrm{ent_y})$ is the number of documents that contain $\mathrm{ent_y}$. The reasoning behind the TF-IDF measure is to lower the weight of features that appear in many documents as this is usually an indication of them being less important (e.g., stopwords). The quality of this approach was established by numerous usages in solutions to various problems in text mining (e.g., Jones, 2003; Feldman and Sanger, 2007).

These four methods can be further modified by vector normalization (dividing each vector so that the length—usually the Euclidian or Manhattan length—of the vector is 1). If and when this should be done, depends on several factors: one of them is the decision which distance measure will be used in the next heuristic specification step. If the cosine similarity is used, a pre-normalization of the vectors is irrelevant, as this is also done during the distance calculation. Figure 3.2 shows the four measures in practice, where the documents are taken from the first table in Figure 3.1. Weights are calculated for the feature space and are not normalized.

### 3.3.2 Similarity measures

The most common measures in vector spaces, which are used extensively in Chapter 6 and especially in Section 6.1.1:

- Dot product:

$$\text{DotProd}(vec_x, vec_y) = \sum_{i=0}^{M} w_{x:i} w_{y:i}, \tag{2}$$

  where $M$ is the number of all the entities.
- Cosine similarity:

$$\text{CosSim}(vec_x, vec_y) = \frac{\text{DotProd}(vec_x, vec_y)}{|vec_x| \cdot |vec_y|}, \tag{3}$$

  which is the dot product normalized by the length of the two vectors. In the cases where the vectors are already normalized, the cosine similarity is identical to the dot product.

In general, the choice of a suitable distance measure should be tightly connected to the choice of the weighting model. Some of the combinations are very suitable and have understandable interpretations or were experimentally evaluated as useful, while others are less appropriate. For the purpose of this thesis we use the following combinations of the most common pairs of weighting model and distance measure:

- *TF-IDF weighting* and *cosine similarity*: this is the standard combination for computing the similarity in the feature space.
- *Binary weighting and dot product distance*: if this is used in the document space, the result is the co-occurrence measure which counts the number of documents where two terms appear together.
- *Term occurrence weighting* and *dot product distance*: this is another measure of co-occurrence of terms in the same documents. Compared to the previous measure, this one considers also multiple co-occurrences of two terms inside a document and gives them a greater weight in comparison with the case where each appears only once inside the same document.

With this section we conclude the presentation of fundamental text preprocessing and representation principles required for the methodology developed in this thesis. The presented concepts form the foundation of understanding the methodology for B-term ranking, especially the definition of heuristic functions, presented in detail in Chapter 6.

# 4 Methodology overview

This chapter provides an overview of the methodology for cross-domain knowledge discovery through text mining. It introduces the conceptual overview of the methodology, present arguments for using a workflow-based representation of the methodology, and provides a top-level overview of the methodology for cross-domain literature mining.

## 4.1 Conceptual overview

At the conceptual level, our methodology provides a solution to the problem of closed discovery as defined in literature mining by Weeber et al. (2001). We shortly introduced the closed discovery concept already in the introductory part of this thesis (Section 1.1 and 1.3) and described it more extensively in the overview of literature-based discovery (Section 2.4.2). In summary, when dealing with a closed discovery problem, we are confronted with two sets of literatures, typically in the form of scientific documents from two domains, named literature A and literature C. The final goal is to find intermediate arguments which suggest new relations between these—previously separated—domains A and C. The supporting arguments, typically terms from the literatures, are according to the Swanson's ABC model called B-terms. Consequently, a simplified formulation of the overall goal of the methodology is to provide suggestions (e.g., scores and rankings) regarding the quality of potential bridging terms (B-terms) extracted from two sets of literatures of interest A and C.

The input to the methodology for B-term identification and ranking consists of two sets of documents, one for each domain. Input documents can either be in the standard form of a running text, e.g., titles and abstracts of scientific documents, or in the form of partly preprocessed text, e.g., text with already recognized named entities. The output of the procedure is a scored and ranked list of all identified interesting terms. The terms are ranked according to their *bisociation scores,* which are the estimates of the potential that the evaluated term is indeed a B-term, which can trigger a bisociation. Our solution to the presented problem of B-term identification and ranking is based on the following three conceptual steps:

1. *Extract candidate B-terms*: Preprocess input documents by employing standard approaches from text mining to extract the useful information present in raw texts. The original documents are transformed into the BoW feature vector representation, whereas the terms are represented in the document vector representation (see Chapter 3 for details). Both representations enable various heuristic calculations of the next step to be done efficiently. The terms extracted in this step are utilized as candidate B-terms in the following step.
2. *Score candidate B-terms*: Take the list of candidate B-terms and evaluate their B-term potential by calculating a *bisociation score* for each term from the list. This step (see Chapter 6 for details) is performed in two phases:

(a) *Evaluate the base heuristics*: Based on the document vector representation of terms and some other properties of terms and documents, use specially designed base heuristic functions to score the terms (see Section 6.1 for details).

(b) *Evaluate the ensemble heuristic*: Base heuristics scores are integrated into one ensemble heuristic score per term, which represents the final output of the scoring candidate B-terms step and is used as the estimate of the term's bisociation potential (see Section 6.3 for details).

3. *Output a ranked list of B-terms*: Order the terms according to descending values of the calculated bisociation scores and return the ranked list of scored terms. Given that this step is done in a rich environment supporting B-term hypotheses testing and exploration, it is not trivial. This final procedural step presents the core functionality of the developed CrossBee system (see Section 6.5 for details).

Besides the three conceptual steps sketched above, there are also other processes (e.g., document acquisition, outlier document detection, and others) that need to be performed in order to achieve the actual working implementation of the proposed methodology. While the rest of this chapter describes a procedural overview of the methodology; Section 6.5 presents the actual implementation of every individual step of the methodology and its components inside the ClowdFlows workflow management and execution environment (see Kranjc et al., 2012 and Online references [34] and [35]).

## 4.2 Workflow-based representation of the methodology

In this section, as well as throughout this thesis, we use workflow diagrams to present the methodology. At this point, all the presented workflow diagrams may be observed only from the conceptual viewpoint. Nevertheless, all of them represent executable workflows composed in the online cloud-based workflow composition environment ClowdFlows.

Every methodology or process can be transformed into a workflow-based representation in many different ways. Usually, in complex methodologies like ours, it is not easy to conceive a workflow, which would be on the one hand conceptually correct and on the other hand technically implementable in an efficient manner. Furthermore, the conceptual design to a large extent defines the implementation and tradeoffs in terms of component reusability, performance, ease of use, configurability, and other systems parameters. When designing our workflow-based representation of the methodology, a top priority was to create a workflow, which conceptually aligns with the methodology, using as much as possible reusable components but at the same time not sacrificing the performance too much.

An important aspect for understanding any kind of workflow is the structure of data that is passed between the workflow components. Although data and data structures may seem to be a technical aspect, it is quite the opposite. By data, we refer to the contents and not to the format or internal organization of data. The data that is passed between the components is one of the key factors defining the design of the components and consequently, of the workflow as a whole. This is why there is also emphasis on explaining the data and not only the components when describing the workflows.

The technical implementation of some of the presented components—mainly text preprocessing and outlier detection—is based heavily on LATINO (Online reference [33]) software

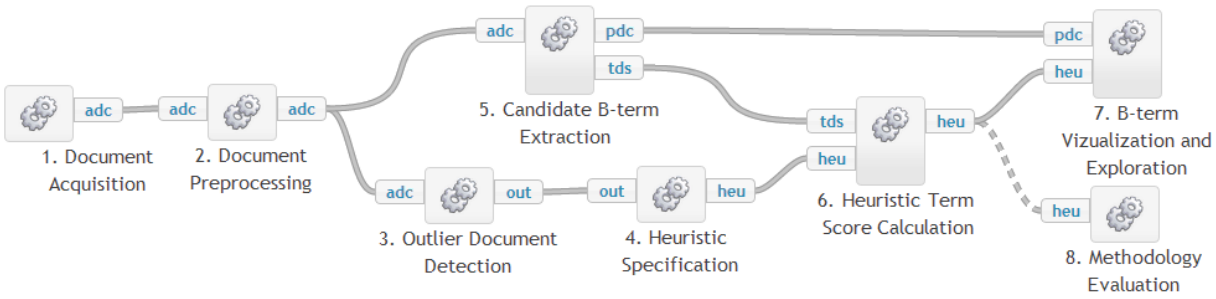



Figure 4.1: *Methodological steps of the overall cross-domain literature mining process.*

library. Nevertheless, packing the library in the form of workflow components on the one hand and developing numerous completely new components for the purpose of this work on the other hand was a very challenging task. Consequently, the conceptual designs and the technical implementation of the presented components and workflows are considered as one of the key contributions of this thesis.

## 4.3 Top-level view of the methodology

The top-most level overview of the methodology, shown in Figure 4.1, consists of the following steps: document acquisition, document preprocessing, outlier document detection, heuristics specification, candidate B-term extraction, heuristic term score calculation and B-term visualization and exploration. An additional ingredient in this top-level overview is the methodology evaluation step; as it is not directly part of the methodology, its connection is drawn as a dashed line in Figure 4.1. The procedural description of the workflow is the following:

1. *Document acquisition* is the first step of the methodology. Its goal is to acquire documents of the two domains, label them with domain labels and pack both domains together into the annotated document corpus (ADC) format.
2. The *document preprocessing* step is responsible for applying standard text preprocessing to the document corpus. The main parts are tokenization, stopword labeling and token lemmatization.
3. The *outlier document detection* step is used for detecting outlier documents that are needed by subsequent heuristic specification. The output is a list (or multiple lists in case when many outlier detection methods are used) of outlier documents.
4. The *heuristic specification* step serves as a highly detailed specification of the heuristics to be used for B-term ranking. The user specifies one or more heuristics which are later applied to evaluate the B-term candidates. Furthermore, each individual heuristic can be hierarchically composed of other heuristics; therefore an arbitrary complex list of heuristics can be composed in this step.
5. The *candidate B-term extraction* step takes care of extracting the terms which are later scored by the specified heuristics. As described in Section 3.2, our process uses a simple n-gram term definition approach and there are several parameters which define the extraction, e.g., the maximal number of tokens to be joined together as a term, minimal term corpus frequency, and similar. The first output is a list of all candidate B-terms (term dataset, TDS) along with their respective document vectors (see Section 3.3 for details). The second output is a parsed document corpus (PDC) which includes information

about the input documents from ADC as well as the exact data on how each document was parsed. This data is needed by the CrossBee web application when displaying the documents as it needs to be able to exactly locate specific words inside a document, e.g., to color or emphasize them.

6. *Heuristic calculation* is methodologically the most important step. It takes the list of extracted B-term candidates and the list of specified heuristics and calculates a heuristic score for each candidate term for each heuristic. The output is structurally still a list of heuristics, however now each of them contains a score for each candidate B-term.
7. *B-term visualization and exploration* is the final step of the methodology. It has two main functionalities. It can either take the heuristically scored terms, rank the terms, and output the terms in the form of a table, or it can take the heuristically scored terms along with the parsed document corpus and send both to the CrossBee web application for advanced visualization and exploration. Besides improved bridging concept identification and ranking, CrossBee also provides various content presentations which further speed up the process of bisociation exploration. These presentations include side-by-side document inspection, emphasis of interesting text fragments, and uncovering similar documents.
8. An additional *methodology evaluation* step was introduced during the development of the methodology. Its purpose is to calculate and visualize various metrics that were used to assess the quality of the methodology. Requirement to use these facilities is to have the actual B-terms as gold standard B-terms available for the domains under investigation. The methodology was evaluated on two problems: the standard migraine-magnesium problem well-known in literature mining, and a more recent autism-calcineurin literature mining problem.

With this, we conclude the overview of the methodology. Firstly, we presented the conceptual overview, next we explained the background behind workflow-based representation of the methodology, which was followed by the top-level overview of the methodology. A more detailed description of the entire methodology for cross-domain knowledge discovery from separate literature sources is presented in Section 6.5 which provides exact descriptions of individual procedural steps along with the specifications of inputs and outputs for each step and the workflow implementation details. The following chapter presents the first main contribution of this thesis—the LemmaGen lemmatization engine—which is one of the key components utilized in the preprocessing step of the methodology that was summarized in this chapter,

# 5 LemmaGen approach to lemmatization

This chapter presents the LemmaGen learning engine, its evaluation and lemmatizers for twelve European languages. Section 5.1 introduces some basic concepts and definitions needed to clearly state the problem in Section 5.2. Section 5.3 describes LemmaGen input and output formats, aimed at generating rules in an understandable decision structure, which also enables efficient lemmatization. Section 5.4 describes how this structure of rules is automatically constructed from the lemmatization examples by LemmaGen. Section 5.5 shows how the lemmatization using constructed rules is performed. Section 5.6 describes the application of LemmaGen to 13 different datasets, compares the results with two other publicly available lemmatization rule learners, and evaluates their performance in terms of lemmatization accuracy, efficiency, and applicability of the approach to different languages. Finally, Section 5.7 presents the implementation of the LemmaGen learning and lemmatization engines as workflows in an online workflow execution environment ClowdFlows.

## 5.1 Stemming and lemmatization

Lemmatization is the process of finding the normalized forms of words appearing in text. It is a useful preprocessing step for a number of language engineering and text mining tasks, and especially important for languages with rich inflectional morphology.

We first introduce some definitions. A *wordform* is the (inflected) form of the word as it appears in a running text, e.g., *wolves*. This wordform can be morphologically analyzed into its stem *wolf-* and ending *-s*. As evident from the example, phonological and morphological factors can influence how the abstract stem and ending are combined to arrive at the wordform. These factors are especially complex in languages with heavy inflection, such as Slovene and other Slavic languages, where stems can combine with many different endings, in a many-to-many relation, and the selection of the appropriate ending for a given stem and how they combine into a wordform can depend on a whole range of factors, from phonological to semantic. Two methods are typically used to abstract away from the variability of wordforms in preprocessing of texts (e.g., text mining, search, information extraction and retrieval): stemming and lemmatization. These two methods are introduced in this section.

The first method, *stemming*, is popular in information extraction and retrieval, and essentially reduces a wordform to an invariant stem that semantically identifies it. This method often collapses different word-classes (parts-of-speech) and does not, in general, produce a surface form. So, for example, the wordforms *computer, computing, computes, computable, computed* would all be typically stemmed to *comput*.

The second method, *lemmatization*, transforms a wordform to its canonical (normalized) form, the lemma, which corresponds to a headword in a dictionary. This canonical form is a particular wordform, which, by convention, serves to identify an abstract word. The distinction

between stems and lemmas is not so important in English, where the stem is often identical to the lemma, but is much more obvious for example in Slavic languages. For example, in English the lemma of the wordform *dogs* is *dog*, of *wolves* is *wolf*, and of *sheep* is *sheep*. On the other hand, in Slovene, for example, the wordform *ovce* (genitive of *sheep*) has the stem *ovc-*, while the lemma form, by convention the nominative singular, is *ovca*. In contrast to stemming, lemmatization is more selective (a single stem can have more than one lemma, e.g., verbal and nominal) and results in an intuitively understandable form of the word. It is also more difficult: not only does the word ending need to be removed from the wordform, but, in general, the correct ending corresponding to the lemma has to be added, as illustrated by the examples we used for stemming: the wordforms *computing, computes, computed* should be lemmatized to *compute*; *computer* to *computer*; and *computable* to *computable*.

Lemmatization is also faced with the problem of ambiguity: a wordform and especially that of an unknown word can have multiple possible lemmas. So, for example, the Slovene wordform *hotela* can be lemmatized as *hotel* (the noun *hotel*), or *hoteti* (the verb *to want*). Which is the correct lemma depends on the context that the wordform appears in. The task of morphosyntactic disambiguation (i.e., determining if a certain wordform in the text is a noun or a verb, and also its other inflectional properties) is the domain of part-of-speech (PoS) taggers, or, more accurately, morphosyntactic taggers. By using the information provided by such a tagger, a lemmatizer is in a much better position to correctly predict the lemma form. However, as shown in this chapter, even without such information, a lemmatizer can still achieve high lemmatization accuracy. Moreover, since our approach does not need morphosyntactic information, it can be used also to lemmatize texts that are not part of complete sentences (e.g., contents of Short Message Service, web queries, etc.).

## 5.2 Problem description

The goal of this work is to construct an efficient multilingual lemmatization engine and to make it publicly available under the GNU open source license. To this end, we have developed a system, named LemmaGen, consisting of a learning algorithm for automatic generation of lemmatization rules in the form of Ripple Down Rules (RDR) (Compton and Jansen, 1990; Srinivasan et al., 1991) and an algorithm for efficient lemmatization using the generated rules. This chapter presents both, the LemmaGen learning engine and lemmatizers for twelve European languages, induced from eight Multext-East (Erjavec, 2010) and five Multext (Multilingual Text Tools and Corpora, Ide and Véronis, 1994) lexicons (see also Online references [18], [19], and [20]).

The proposed LemmaGen approach assumes the availability of a lexicon of lemmatized words. The main idea of our approach is: lemmatize a new—so far unseen—wordform in the same way as the most similar wordform in the lexicon was lemmatized. This can be achieved by *transforming* the problem of *lemmatization* to the problem of *classification*: How to find the correct *class* (transformation) for the current wordform. Correspondingly, the lemmatization problem is translated to a problem of finding the most appropriate class, i.e., the one of the most similar wordform in the lexicon from which the lemmatizer is trained.

What remains is to define an appropriate measure of similarity between two wordforms. As a measure of similarity, we chose the length of the suffixes shared by the two wordforms. For example, the similarity of wordforms *computable* and *compute* is 1 (only the last *-e* is the

same), while the similarity of wordforms *computable* and *permutable* is 6 since they both share the *-utable* suffix. This notion of similarity makes sense because our target languages (mostly) belong to the Indo-European language group where inflection is commonly expressed by suffixes. In these languages, the words that have similar suffixes usually behave alike when inflected, and as a consequence, their wordforms are similar when considering suffixes.

The concept of lemmatization by similarity can be extended to several additional strategies, which in general increase the accuracy. Here we state just the two included in our approach. First, one must resolve the ambiguity when an unknown wordform has equal similarity to more than one wordform from the lexicon. Considering there is no additional information, we choose the most frequent class from the set of all similar wordform classes. In other words, we complement the similarity measure of wordforms with the class frequency. Second, it can also happen that one wordform has equal similarity to two wordforms with the same class frequency. In this case, we look for the second most similar wordform to decide which class to prefer. These are some of the modifications of the similarity measure, which proved to improve the overall accuracy of our lemmatization approach.

The main advantages of the developed learner and multilingual lemmatizer are the following. Firstly, LemmaGen implements a general, non-incremental RDR algorithm, specially tuned to learn from examples with a string-like structure. Secondly, it is a language independent learning engine, at least for inflectional languages. Consequently, it has already been applied to generate lemmatizers for twelve different European languages. Next, the format of induced rules is human readable and can be executed very efficiently (in constant time) when used for lemmatization of new words/texts. The evaluation of LemmaGen on the corresponding lexicons shows that LemmaGen outperforms the lemmatizers generated by two alternative approaches, RDR and CST, both in terms of accuracy and efficiency.

To best of our knowledge, LemmaGen is the most efficient publicly available lemmatizer trained on large lexicons of multiple languages. The learning algorithm and the induced lemmatizers have been made freely available (Online references [3] and [5]) under the GNU open source license. Note also that the LemmaGen learning engine can be retrained on additional languages, and could, most likely, be generalized to deal with other machine learning problems involving string processing.

## 5.3 Knowledge representation

This section describes the LemmaGen input (training examples) and output (lemmatization rules) data structures.

### 5.3.1 Representation of training examples

The training data for lemmatization is generally represented in the form of pairs (Wordform, Lemma). However, for the sake of training by LemmaGen, training examples are actually represented as pairs (Wordform, Class) where the Class label is the transformation which replaces the wordform suffix by a suffix of the lemma (see Table 5.1). It is worth pointing out that our method can be used also for stemming. One only has to replace the training pairs (Wordform, Lemma) with (Wordform, Stem). Thus, even though we concentrate on lemmatization, one can easily switch to stemming by changing the training data.

```
IF bird THEN flies EXCEPT
   IF young bird THEN doesn't fly
      ELSE IF penguin THEN doesn't fly EXCEPT
         IF penguin in airplane THEN flies
ELSE IF airplane THEN flies
```

Figure 5.1: *A simple Ripple Down Rule structure.*

Table 5.1 illustrates the variability of suffixes of wordforms of Slovene, a language with high inflectional complexity. Note however, that even in English there are many different suffixes of wordforms sharing the same stem. Table 5.2 lists a selected set of wordforms from the Multext English lexicons for lemmas starting with string *writ-,* which are used also in the experiments of Section 5.6. The entries in the lexicon have the form of triplets (Wordform, Lemma, MSD), where MSD stands for the morphosyntactic description of the wordform, i.e., a feature structure giving the PoS and other morphosyntactic attributes of the wordform.

### 5.3.2 Representation of lemmatization rules

This section describes the output of the learning algorithm. We first describe the standard Ripple Down Rule (RDR) structure and show how lemmatization rules can be expressed using it. Then, a description of our refinement of the RDR structure is given.

#### Standard RDR format

Ripple Down Rules (RDRs) (Compton and Jansen, 1990; Srinivasan et al., 1991) were originally used for *incremental* knowledge acquisition and maintenance of rule-based systems. Compared to standard if-then classification rules, RDRs resemble decision lists of the `IF-THEN-ELSE` form (Rivest, 1987). However, the main idea of RDRs is that the most general rules are constructed first, and later, as counter examples are encountered, exceptions to the rules are added (`EXCEPT` branches) in an iterative, incremental rule building process. Consequently, RDRs form a tree-like decision structure: rules and their exceptions are ordered, and the first condition that is satisfied and has no exceptions, fires the corresponding consequent. Figure 5.1 shows a simple Ripple Down Rule describing flying properties of birds and objects.

A small—but realistic—RDR for wordform lemmatization, constructed incrementally from examples in Table 5.2, is shown in Figure 5.2. The individual rules in the RDR structure are ordered and need to be interpreted sequentially. A valuable feature of RDR rules is also that one can attach examples to individual clauses. For instance, in a rule such as

```
IF suffix("ote") THEN transform("ote"-->"ite") EG wrote
```

where the additional `EG` keyword lists an example from the training set that caused the creation of the decision branch. This feature of RDRs turns out to be helpful for better understanding of complex rules.

#### Refined RDR Format

In the case of learning lemmatization rules, training examples in the form of large lexicons are readily available. Therefore, there is no need for a learning algorithm to be incremental. A compact RDR structure, including all the exceptions, can be computed by a non-incremental algorithm. In this section, we show how the original RDR representation can be refined in order to improve the readability and efficiency of RDRs for lemmatization.

Table 5.1: *Examples of lemmatization classes.* Examples of class labels, which are used to form training examples (Wordform, Class) for lemmatization of English and Slovene words. Note a great variability of wordform endings for a single Slovene word *brati* (meaning *to read* in English).

| | Wordform | Lemma | Class (WordformSuffix --> LemmaSuffix) |
|---|---|---|---|
| English | | | |
| 1 | dogs | dog | ("s" --> "") |
| 2 | wolves | wolf | ("ves" --> "f") |
| 3 | sheep | sheep | ("" --> "") |
| 4 | looking | look | ("ing" --> "") |
| 5 | took | take | ("ook" --> "ake") |
| Slovene | | | |
| 1 | brat | brati | ("" --> "i") |
| 2 | brala | brati | ("la" --> "ti") |
| 3 | beri | brati | ("eri" --> "rati") |
| 4 | berete | brati | ("erete" --> "rati") |

Table 5.2: *Example from the English lexicon.* Triplets of the form (Wordform, Lemma, MSD), where MSD stands for the wordform morphosyntactic description. This is the set of all lemmas starting with *writ-*, as they appear in the Multext English lexicon (used in the experiments of Section 5.6).

| Wordform | Lemma | MSD |
|---|---|---|
| write | write | Vvb--- |
| write-off | write-off | Ncns- |
| write-offs | write-off | Ncnp- |
| writer | writer | Ncfs- |
| writer | writer | Ncms- |
| writers | writer | Ncfp- |
| writers | writer | Ncmp- |
| writes | write | Vvfps3 |
| writhe | writhe | Vvb--- |
| writhe | writhe | Vvfpp- |
| writhe | writhe | Vvfps1 |
| writhe | writhe | Vvfps2 |
| writhed | writhe | Vvfs-- |
| writhed | writhe | Vvps-- |
| writhes | writhe | Vvfps3 |
| writhing | writhe | Vvpp-- |
| writing | write | Vvpp-- |
| writing | writing | Ncns- |
| writings | writing | Ncnp- |
| writs | writ | Ncnp- |
| written | write | Vvps-- |
| wrote | write | Vvfs-- |

```
1      IF suffix("") THEN transform(""-->"") EXCEPT
1.1       IF suffix("ote") THEN transform("ote"-->"ite")
1.2       ELSE IF suffix("ten") THEN transform("ten"-->"e")
1.3       ELSE IF suffix("s") THEN transform("s"-->"")
1.4       ELSE IF suffix("g") THEN transform(""-->"") EXCEPT
1.4.1        IF suffix("hing") THEN transform("ing"-->"e")
1.5       ELSE IF suffix("d") THEN transform("d"-->"")
1.6       ELSE IF suffix("e") THEN transform(""-->"")
1.7       ELSE IF suffix("r") THEN transform(""-->"")
1.8       ELSE IF suffix("f") THEN transform(""-->"")
1.9       ELSE IF suffix("t") THEN transform(""-->"")
```

Figure 5.2: *Initial RDR structure.* A RDR structure, constructed by incremental learning algorithm RDR (Plisson et al., 2008) from examples in Table 5.2.

```
1      IF suffix("") THEN transform(""-->"") EXCEPT
1.1       IF suffix("ote") THEN transform("ote"-->"ite")
1.2       ELSE IF suffix("ten") THEN transform("ten"-->"e")
1.3       ELSE IF suffix("s") THEN transform("s"-->"")
1.4       ELSE IF suffix("g") THEN transform(""-->"") EXCEPT
1.4.1        IF suffix("hing") THEN transform("ing"-->"e")
1.5       ELSE IF suffix("d") THEN transform("d"-->"")
1.6       ELSE IF suffix("e") THEN transform(""-->"")
1.7       ELSE IF suffix("r") THEN transform(""-->"")
1.8       ELSE IF suffix("f") THEN transform(""-->"")
1.9       ELSE IF suffix("t") THEN transform(""-->"")
```

Figure 5.3: *Refined RDR structure.* A RDR tree structure, constructed by LemmaGen from English words in Table 5.2.

Focusing on a single if-then rule inside the rule structure in Figure 5.2, we can see that it has the following form:

- A rule condition is the suffix of a wordform, which fires the rule, e.g., *ote* in Rule 1.1.
- A rule consequent is a class, i.e., a transformation of the form *WordformSuffix* `-->` *LemmaSuffix*, e.g., *ote*`-->`*ite* in Rule 1.1.

Utilizing the original RDR structure for lemmatization has some disadvantages considering our two main objectives, efficiency and readability. Firstly, efficiency suffers because one is not able to detect directly which exception applies to the wordform that is currently being lemmatized. Therefore, at each level of the tree, one must test conditions of all the exceptions until either the one that applies is found or the end of the list is encountered. These lists of exceptions can be long (i.e., up to 200 for real lexicons) and their processing can take a relatively large amount of time. The readability also suffers due to the long lists of exceptions and sequential triggering of rules. For example, in Figure 5.2, Rule 1.6 is more general than Rule 1.1 and has to be checked *after* Rule 1.1 for correct interpretation.

To solve the problems of efficiency and readability, we have refined the original RDR structure, illustrated in Figure 5.3. Note that, from now on, in the `IF-THEN-ELSE` rule format, we will omit the `ELSE` keyword in order to increase the readability of rules.

The refined RDR structure was achieved by imposing equal-similarity constraint that solves both problems and can be elegantly expressed using the similarity measure defined in Section 5.2: the similarity among conditions (suffixes) inside one exception list must be the same for all possible pairs. For instance, in Figure 5.3, the similarity among conditions of the exception list of the top level rule (*-d*, *-ote*, *-ing*, *-ten*, *-s*) is zero. On the other hand, if we focus on the exception list of the top level rule in Figure 5.2, we can notice that in the RDR structure, the equal-similarity constraint is violated: similarities among different suffixes are not the same: while the similarity between suffixes 1.1 *-ote* and 1.2 *-ten* is 0, the similarity between 1.1 *-ote* and 1.6 *-e* is 1.

## Assertion

Conditions in the same exception list of refined RDR structure satisfy the following constraints:

1. All the suffixes share the same ending ($k-1$ characters).
2. The first character that is different among suffixes (the $k^{th}$ character from the right) is different for all the suffixes, therefore it disjunctively separates all the suffixes.

One can always choose such $k$ that the first statement holds, e.g., if the suffixes share no common ending, then $k$ is set to 1. Subsequently, when $k$ is identified, we must only prove that the second statement holds; namely, that there are no two suffixes having the same character at the position $k$ from the end. This assertion is proved by contradiction; if two such suffixes exist, the similarity between them is greater (at least $k$) than between the others $(k-1)$. This violates the equal-similarity constraint, therefore no such two suffixes exist. With this we have proved that the proposed assertion holds if the equal-similarity constraint is obeyed.

Using the above assertion, we can derive the following properties of the refined RDR structure.

1. It can always be decided which sub-rule fires on a specific wordform by examining just one character (the $k^{th}$) of a wordform suffix. In combination with a proper implementation (e.g., a hash table), the RDR structure can be traversed very efficiently.
2. Exceptions are disjunctive. Consequently, there is always at most one sub-rule that fires for a specific wordform. Moreover, all the rules dealing with similar suffixes are grouped together; therefore the readability is considerably improved.
3. The maximal number of exceptions of one rule is limited by the number of characters in the alphabet. As conditions are disjunctive with respect to the character at the $k^{th}$ position, the number of disjunctive conditions cannot be greater than the number of characters in an alphabet. For the majority of European languages, this number is less than 30 (note that in the original RDR structure constructed by the incremental RDR learner there can be several hundreds of exceptions in constructed rules). This also contributes to improved efficiency and readability.

To summarize, the proposed refined RDR structure overcomes two weaknesses of original RDRs, improving their readability and enabling efficient execution in lemmatization tasks.

## 5.4 The LemmaGen learning algorithm

This section describes the LemmaGen learning algorithm, which learns lemmatization rules in the form of a refined RDR structure. The learning algorithm is efficient and language independent since it can be used for training on new lexicons of lemmatized wordforms.

There are two main properties of the learning algorithm. Firstly, in contrast to most of the RDR algorithms, it is not an incremental learner. Secondly, it is very efficient and has a low time complexity as compared to the original RDR learners. These two properties—non-incrementality and efficiency—are the core improvements of the LemmaGen learning algorithm.

The pseudocode of the algorithm is given in Figure 5.4 and 5.5: the top level function and recursive learning of a rule, respectively. To describe the algorithms, we use an object-oriented representation of the recursive RDR structure. Each rule has three components: a condition, a class, and a (possibly empty) list of exceptions:

`RDR ::= IF` *rule.condition* `THEN` *rule.class* `EXCEPT` *rule.exceptions*

where:

- *rule.condition* is a wordform suffix,
- *rule.class* is a transformation to be applied to a wordform, and
- *rule.exceptions* ::= nil | RDR+ (list of RDRs).

```
1.1 function LearnRDR(examplesList)
1.2    sortedExamplesList = Sort(examplesList, "reverse dictionary sort")
1.3    entireRDR = LearnRecursive(sortedExamplesList)
1.4    return entireRDR
```

Figure 5.4: *Main learning function.* Top level function of the LemmaGen learning algorithm.

The input to learning is a list of training examples, and the output is a RDR structure. The top level function, *LearnRDR* Figure 5.4, just sorts the examples by their wordforms with character strings reversed (line 1.2), and invokes recursive rule learning (function *LearnRecursive*) on the sorted list of examples (line 1.3).

The *LearnRecursive* function Figure 5.5 is the core of the learning algorithm. The function assumes that the examples are sorted by their wordforms, and that each recursive invocation deals with example wordforms which share increasingly longer suffixes. The recursion stops when all the remaining wordforms in the *currentExamplesList* are equal or when there remains just a single example in this list. The *LearnRecursive* function is executed as follows:

- It first finds the common suffix of all examples from the *currentExampleList*. *commonSuffix* can be determined by just comparing the suffixes of the first and last wordforms (line 2.2) because the *currentExamplesList* is sorted.
- Next, it initializes the main variable *currentRule* to an empty RDR structure (line 2.4), assigns to its condition the *commonSuffix* (line 2.5), and to its class the most frequent class of the current set of examples (line 2.6). When determining the most frequent class (function *MostFreqClass*), it ignores such classes `X-->Y` in which `X` is more specific than the current *commonSuffix*.
- The list of exceptions (*currentRule.exceptions*) is constructed in the part of the pseudocode between lines 2.7 and 2.19.
  - Line 2.8 sets the index to start grouping of examples into subsets with longer common suffixes (longer than current *commonSuffix*). The subsets are identified iteratively in the *for* loop (line 2.9). These subsets will later be used to form new exceptions of the current rule.
  - First, two adjacent wordforms are extracted from *currentExamplesList* (lines 2.10 and 2.11). Then the character which distinguishes the suffixes of the two wordforms needs to be determined. It is the first character left of the common suffix of the two wordforms; in the pseudocode, the position of this "distinguishing" character is determined by subtracting the length of the common suffix from the wordlength of each of the two wordforms (lines 2.12 and 2.13).
  - Once the subgroup of examples with a longer common suffix is determined (through identifying the two wordforms with different "distinguishing" character, i.e., when the condition 2.14 is no longer satisfied), the exception to the current rule is constructed (2.17) by recursively calling the *LearnRecursive* procedure for the current group of examples (a subset of *currentExamplesList* from position *start* to current position *i*). Finally, the created exception is added to the list of exceptions of the *currentRule* (line 2.18).

The pseudocode is an abstraction of the actual implementation and does not deal with the details of boundary conditions. It also omits some heuristics which optimize the resulting RDR structure by taking into account the subsumption between the rules.

```
2.1   function LearnRecursive(currentExamplesList)
2.2      commonSuffix = EqualSuffix(currentExamplesList.First().wordform,
                            currentExamplesList.Last().wordform)
2.3      lengthCommonSuffix = StringLength(commonSuffix)
2.4      currentRule = RDR()
2.5      currentRule.condition = commonSuffix
2.6      currentRule.class = MostFreqClass(currentExamplesList)
2.7      currentRule.exceptions = nil
2.8      start = 1;
2.9      for i = 1 to Length(currentExamplesList)-1
2.10        wf1 = currentExamplesList[i].wordform
2.11        wf2 = currentExamplesList[i+1].wordform
2.12        charPosition1 = StringLength(wf1) - lengthCommonSuffix
2.13        charPosition2 = StringLength(wf2) - lengthCommonSuffix
2.14     if (GetChar(wf1,charPosition1) == GetChar(wf2,charPosition2))
2.15        continue
2.16     else
2.17        exceptionRule = LearnRecursive(currentExamplesList[start..i])
2.18        currentRule.exceptions.Append(exceptionRule)
2.19        start = i+1
2.20     return currentRule
```

Figure 5.5: *Recursive learning function.* A function of the algorithm for learning a RDR structure (top level rule and all exception subrules) from a current list of examples.

Theoretical time complexity of the algorithm is $O(n \cdot m)$ where $n$ is the number of all distinct training examples in *examplesList* and $m$ is the length of the longest wordform in the examples. There are two main parts of the algorithm, which need to be considered for time complexity calculation. Firstly, there is lexicographic sorting of the examples (line 1.2 in Figure 5.4); the time complexity of sorting is in general $O(n \log(n))$, however, since we are dealing with strings of limited length, we can use radix sort which has the time complexity of $O(n \cdot m)$. The second part refers to learning of a RDR (line 1.3), specifically the function *LearnRecursive(examplesList)*. The worst-case time complexity here is again $O(n \cdot m)$—the algorithm tests each word in the list of $n$ examples and each time progresses by at least one character, therefore at most $m$ such repetitions can happen for a word. Thus, we conclude that also the overall time complexity is $O(n \cdot m)$. Furthermore, as $m$ is usually fixed for a given language, it can be considered a constant and the actual time complexity of the learning is $O(n)$, i.e., it is linear with respect to the number of examples.

## 5.5 The LemmaGen lemmatization algorithm

This section describes the LemmaGen lemmatization algorithm which uses the refined RDR structure to assign lemmas to words of a given language. The efficiency of this algorithm is due to the design and compactness of the refined RDR structure.

In the following, we present how the learned RDR structure is applied to the classification of new words during lemmatization. Figure 5.6 shows the pseudocode of the lemmatization algorithm. The code is simple, although one point needs some elaboration: how to (effectively) choose the next exception (if it exists):

- First, the position of a character (*keyChar*) that disjunctively separates the exceptions inside the wordform is retrieved (line 1.3). The variable *charPosition* is calculated by sub-

```
1.1 function Lemmatize(wordform, rootRule)
1.2    charPosition = StringLength(wordform) - StringLength(rootRule.condition)
1.3    keyChar = GetChar(wordform,charPosition)
1.4    if (rootRule.exceptions.Exist(keyChar))
1.5       exceptionRule = rootRule.exceptions.Get(keyChar)
1.6       return Lemmatize(wordform, exceptionRule)
1.7    else
1.8       lemma = rootRule.class.Transform(wordform)
1.9       return lemma
```

Figure 5.6: *The lemmatization function.* The function of the lemmatization algorithm that implements a recursive descent through the RDR structure until there is no exception to the current rule.

tracting the length of the current rule suffix from the length of the wordform (line 1.2). In such a way we get the $k^{th}$ character of the wordform (as defined in Section 5.3.2).

- Using the $k^{th}$ character as the key, the exception is directly retrieved from the hash table *rootRule.exceptions* if it exists (lines 1.4 and 1.5). The key in the hash table is just one (distinguishing) character, while the value is the exception-sub-rule that corresponds to the given character.
- With this procedure one retrieves exception rules along the tree path until there are no more exceptions. When the last one, i.e., the most specific one is found, it is used to lemmatize the wordform by applying the assigned transformation to it (line 1.8).

The worst-case time complexity of the lemmatizing algorithm is $O(m)$, where $m$ is the depth of the RDR tree, i.e., in the worst case the length of the longest wordform from the examples. However, since the longest training wordform is limited, lemmatization can be performed in constant time $O(1)$.

## 5.6 Application on the Multext-East and Multext data

This section describes the application of LemmaGen learner to induce lemmatization rules for all the languages for which training lexicons are available through European Union projects Multext (Ide and Véronis, 1994) (Multilingual Text Tools and Corpora) and Multext-East (Erjavec, 2010) (Multext for Central and Eastern European Languages). The lexicons were used to automatically learn lemmatizers for different languages. The accurate and efficient publicly available lemmatizers for 12 European languages represent an important contribution of this thesis.

### 5.6.1 Multext-East and Multext lexicons

There were altogether 13 lexicons for 12 European languages available for learning (there are two different training sets available for English). The sizes and properties of these training sets are listed in Table 5.3. Some observations can be made from this table:

- The sizes of the lexicons differ very much across languages, and the expectation would be that the larger the lexicon (training set), the better the achieved lemmatization accuracy.
- Each lexicon contains records in the form of triples (Wordform, Lemma, MSD), where MSD stands for the wordform morphosyntactic description.
- The number of distinct MSDs for a language may seem to indicate its inflectional complexity; however, it should be noted that the differences in their numbers are also due to

Table 5.3: *Lexicons' properties.* Sizes and properties of the Multext-East and Multext lexicons, in terms of numbers of records, different wordforms, lemmas and MSDs, as well as average numbers of lemmas per wordform, lemmas per pair (Wordform, MSD), and the percentage of records where the wordform is identical to the lemma.

| | | | | | Lemmas per | | WF= |
|---|---|---|---|---|---|---|---|
| Language | Records | Wordforms | Lemmas | MSDs | WF | WF-MSD | Lemma |
| Multext-East | | | | | | | |
| Slovene | 557,970 | 198,083 | 16,352 | 2,083 | 1.0444 | 1.0008 | 5.75% |
| Serbian | 20,294 | 16,809 | 8,355 | 906 | 1.0288 | 1.0025 | 25.08% |
| Bulgarian | 55,200 | 40,708 | 22,790 | 338 | 1.1017 | 1.0058 | 28.92% |
| Czech | 184,628 | 56,795 | 23,030 | 1,428 | 1.0464 | 1.0062 | 39.36% |
| English | 71,784 | 48,309 | 27,343 | 135 | 1.0208 | 1.0007 | **61.08%** |
| Estonian | 135,094 | 89,128 | 46,747 | 642 | **1.1540** | 1.0079 | 24.88% |
| Hungarian | 64,042 | 50,908 | 27,991 | 619 | 1.1219 | **1.0138** | 29.45% |
| Romanian | 428,194 | 352,003 | 39,275 | 616 | 1.0453 | 1.0010 | 11.77% |
| Multext | | | | | | | |
| English | 66,216 | 43,273 | 22,794 | 133 | 1.0184 | 1.0006 | 58.17% |
| French | 306,795 | 231,734 | 29,319 | 380 | 1.0166 | 1.0014 | 11.71% |
| German | 233,858 | 50,085 | 10,485 | 227 | 1.0319 | 1.0024 | 22.11% |
| Italian | 145,530 | 115,614 | 8,877 | 247 | 1.0636 | 1.0042 | 6.47% |
| Spanish | 510,709 | 474,150 | 13,232 | 264 | 1.0069 | 1.0010 | **2.73%** |

different MSD design principles employed for various languages. So, for example, Slovene has significantly more MSDs than Serbian or Czech, although the languages are inflectionally of comparable complexity. The main reason for this is a very detailed pronoun typology for Slovene, which leads to over one thousand MSDs for pronouns alone.

- The ratio of the number of wordforms against the number of distinct lemmas should indicate the size of inflectional paradigms of various languages, and also give an indication of their inflectional complexity. However, this is not overall the case, as different principles were adopted in constructing the lexica: some languages (i.e., Multext languages, Slovene, and Romanian) include the complete paradigms for each lemma, while the others include only entries of wordforms actually attested in a reference corpus.
- The number of lemmas per wordform and per wordform-MSD pair indicate the upper bound on lemmatization accuracy. With the ratio greater than 1, a lemmatizer will be, even theoretically, unable to always generate the correct lemma given a wordform or wordform-MSD pair. As can be seen, this problem can be quite severe if the lemmatizer operates only on wordforms; in the worst case (Estonian), the ambiguity is over 15%. If the MSD is taken into account, some ambiguity still remains, but is, even in the worst case (Hungarian), under 1.4%.
- The percentage of entries where the wordform is identical to the lemma (WF=Lemma in Table 5.3) gives the accuracy of a baseline lemmatizer, which would simply assume that the lemma is always identical to the wordform. As can be seen, this approach would already give about 60% accuracy for English. However, with other languages the situation corresponds less well with intuitions, again due to the differences in design criteria discussed above. Lexicons including complete inflectional paradigms have a much lower percentage, as lemma-identical wordforms will tend to occur much more frequently in corpora than in paradigms. Especially surprising is the very low number for Spanish—inspection of the lexicon reveals that the lexicon contains a large number of verbs, which, in Spanish, have a large number of different wordforms, leading to fewer than 3% of lemma-identical wordforms.

In summary, the statistics over the lexicons have as much to do with the decisions made in the design of the MSD sets and what wordforms to include in the lexica, as with the morphological complexity of the various languages.

### 5.6.2 Learners used in the LemmaGen evaluation

We have compared the performance of LemmaGen to the performance of other available lemmatization rule learners: CST (Dalianis and Jongejan, 2006) and RDR (Plisson et al., 2008).

CST (Dalianis and Jongejan, 2006) is one of the few trainable lemmatizers that is available for download (Online reference [11]), and therefore we were able to directly compare its results with the results obtained by our lemmatizer, LemmaGen. The other publicly available lemmatizer, RDR (see Plisson et al., 2008 and Online reference [12]) was—like LemmaGen—inspired by the Ripple Down Rule learning methodology (Compton and Jansen, 1988), implementing the idea of iterative ruleset construction: new rules are incrementally added to the system when new examples of decisions are made available.

### 5.6.3 Experimental settings

For training and testing experiments we used 5-fold cross validation repeated 20 times. Five-fold cross validation is adequate for our experiments as it is estimated that on the average, for all the twelve languages, approximately 80% of all the words of a given language are present in the lexicon. The procedure is repeated 20 times to ensure the stability of statistical evaluation measures, e.g., average accuracy, standard deviation.

The three learning algorithms were tested in two different experimental settings. Recall that in the original lexicons, the records are formed of triples (Wordform, Lemma, Morphosyntactic description). The two settings differ in terms of whether they use the morphosyntactic descriptions (MSD) assigned to the pairs wordform-lemma or not.

- In the first experimental setting, the MSDs were not used. This is the more difficult experimental setting, mimicking the situation in applications working on raw text data (e.g., web page analysis, information retrieval, document clustering and classification, etc.), where stemming and lemmatization are standardly used in the preprocessing of text documents and in which data is usually not (manually or automatically) annotated by morphosyntactic tags. This realistic setting does not need, apart from tokenization and case normalization, any additional processing. However, the achieved accuracies are lower than those achieved in the second setting.
- In the second experimental setting, MSD information was used in the following way: for every language, the available training set was split into separate training sets, one per MSD. For Slovene, this amounts to 2,083 training sets, for Serbian 906 training sets, etc. (c.f. MSDs column of Table 5.3). While being more complex for experiments, due to the separation into numerous separate training sets, the learning task in each training sets is simpler. The main reason is that this separation by itself eliminates many ambiguities: the same wordform string may have several lemmas, but if these are separated into different training sets for different MSDs, the ambiguities are automatically resolved. In this experimental setting, the overall classification accuracy is expected to be higher than in the first one. In practice, using MSD information means that in order to lemmatize a

Table 5.4: *Accuracy testing without MSD.* Comparison of accuracies achieved by RDR, CST and LemmaGen in the experimental setting without using the MSD information. In lexicons/sets where an algorithm significantly outperformed the other two, we mark it by bold accuracy of such outperformer (e.g., LemmaGen outperformed RDR and CST on Serbian training set but not on the Slovene).

| | Training Set | | | Test Set | | |
|---|---|---|---|---|---|---|
| Language | RDR | CST | LemmaGen | RDR | CST | LemmaGen |
| Multext-East | | | | | | |
| Slovene | 99.8 ±0.01 | 99.9 ±0.00 | 99.9 ±0.00 | 93.2 ±0.16 | 91.1 ±0.18 | **93.4** ±0.16 |
| Serbian | 99.7 ±0.03 | 99.7 ±0.03 | **99.8** ±0.02 | 85.2 ±0.60 | 83.0 ±0.63 | **86.1** ±0.61 |
| Bulgarian | 99.4 ±0.02 | 99.5 ±0.02 | 99.5 ±0.02 | 93.7 ±0.22 | 84.0 ±0.40 | **94.1** ±0.21 |
| Czech | 99.4 ±0.03 | 99.4 ±0.03 | **99.5** ±0.03 | 90.0 ±0.35 | 89.3 ±0.36 | **90.6** ±0.35 |
| English | 99.7 ±0.02 | 99.9 ±0.01 | 99.9 ±0.01 | 97.5 ±0.12 | 95.4 ±0.23 | **97.7** ±0.17 |
| Estonian | 99.1 ±0.02 | 99.2 ±0.02 | **99.3** ±0.02 | 90.3 ±0.22 | 80.4 ±0.29 | **90.8** ±0.21 |
| Hungarian | 98.7 ±0.03 | 98.8 ±0.04 | **98.9** ±0.03 | 92.1 ±0.25 | 79.9 ±0.34 | **92.3** ±0.24 |
| Romanian | 99.7 ±0.00 | 99.9 ±0.00 | 99.9 ±0.00 | 88.3 ±0.10 | 83.0 ±0.14 | **88.6** ±0.11 |
| Multext | | | | | | |
| English | 99.8 ±0.01 | 99.9 ±0.01 | 99.9 ±0.01 | 97.8 ±0.16 | 96.0 ±0.19 | **98.0** ±0.13 |
| French | 99.6 ±0.01 | 99.8 ±0.01 | 99.8 ±0.01 | 96.8 ±0.09 | 94.8 ±0.13 | **97.1** ±0.08 |
| German | 99.7 ±0.01 | 99.7 ±0.01 | 99.8 ±0.01 | 96.2 ±0.17 | 95.4 ±0.20 | **96.3** ±0.16 |
| Italian | 99.3 ±0.02 | 99.6 ±0.01 | 99.6 ±0.01 | 95.4 ±0.14 | 88.3 ±0.23 | **95.7** ±0.14 |
| Spanish | 99.8 ±0.00 | 99.9 ±0.00 | 99.9 ±0.00 | 98.1 ±0.05 | 97.5 ±0.05 | **98.4** ±0.04 |

text, it needs to be first tagged with such MSDs. We do not address this issue here, but see the experiments with CLOG (Erjavec and Džeroski, 2004).

### 5.6.4 Experimental results without using MSDs

In these experiments, only pairs (Wordform, Lemma) from the lexicons were used. LemmaGen was compared to RDR and CST.

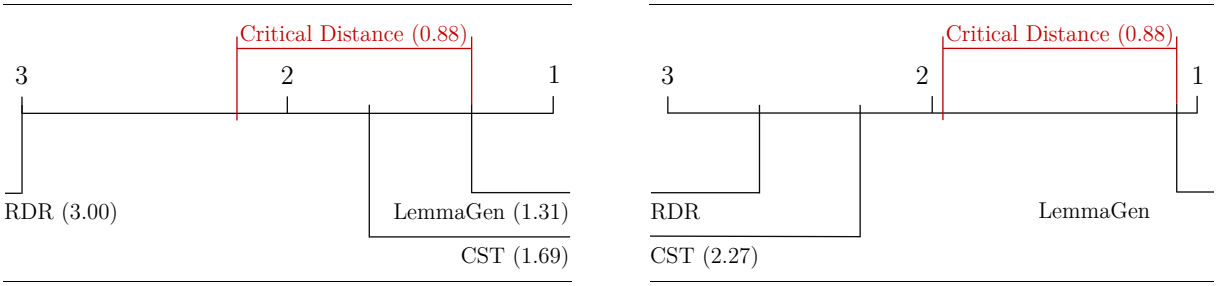
#### Accuracy testing

Results of the comparison are given in Table 5.4. Some further explanations are required before analyzing the results table.

- Accuracy: Lemmatization assigns a transformation (class) to a wordform. If there are $p$ correctly lemmatized wordforms, and $n$ is the total number of wordforms, then the accuracy $acc = {}^{p}/_{n}$.
- Accuracy was averaged separately on training and test sets over 100 runs (20 times 5 fold cross validation). When constructing a test set, we made sure that no two words with the same wordform and lemma appeared in both the training and test set in the same validation step.
- The pairwise differences in the accuracy were tested for statistical significance by a paired Wilcoxon signed-ranks test (Wilcoxon, 1945). When the highest accuracy is significantly different from the rest (for a $p$-value = 0.05), this is indicated by a bold value in the table.

#### Algorithm ranking in terms of accuracy

The goal of our experimental evaluation is to verify a hypothesis that LemmaGen performs significantly better in comparison to RDR and CST, over multiple datasets. The widely used t-test is usually inappropriate and statistically unsafe for such a comparison (Demšar, 2006).




(a)Training Set (b) Test Set

Figure 5.7: *Visualization of accuracy testing without MSD visualization.* Bonferroni-Dunn post-hoc tests for the first experiment using average ranks of algorithms on data from Table 5.4.

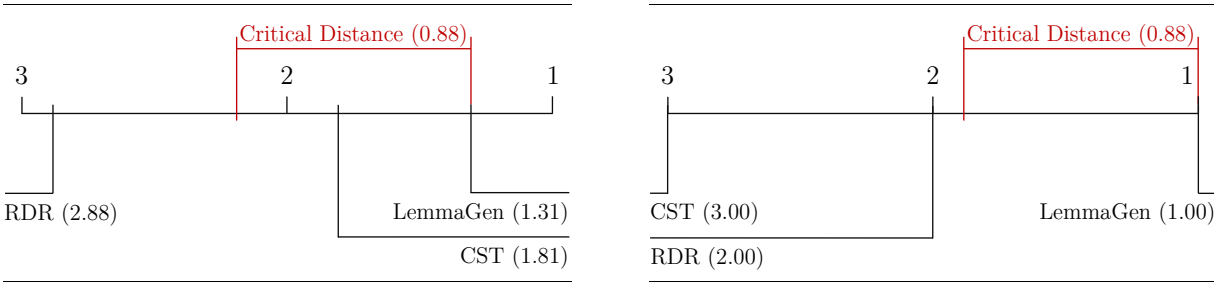
To compare two classifiers over multiple datasets, the Wilcoxon signed-ranks test (Wilcoxon, 1945) is recommended. However, in our case, we want to compare three classifiers in terms of their accuracies. To test the significance of differences between multiple means, a common statistical method is the well-known ANOVA (Fisher, 1959) and its non-parametric counterpart, the Friedman test (Friedman, 1940).

We applied the Friedman test and its corresponding Bonferroni-Dunn (Dunn, 1961) post-hoc test. The Friedman test ranks the algorithms for each dataset separately, the best performing algorithm getting the rank of 1, the second best rank 2, etc. In the case of ties (we used accuracies computed to a precision of just one decimal point), average ranks are assigned. The Friedman test then compares the average ranks of the algorithms. The null-hypothesis states that all the algorithms are equivalent and so their ranks should be equal. If the null-hypothesis is rejected, we can proceed with a post-hoc test.

- The Nemenyi test (Nemenyi, 1963) should be used to compare classifiers to each other.
- In our case, however, we want to compare other classifiers (RDR and CST) to our control classifier (LemmaGen). Consequently, the Bonferroni-Dunn (Dunn, 1961) test is used since it has a much greater power when all classifiers are compared only to a control classifier and not between themselves (Demšar, 2006).

The Bonferroni-Dunn test computes the critical distance (in our case 0.88 for the given $p$ value 0.05) between a classifier and the control classifier, and concludes that the accuracy of the two classifiers is significantly different if the corresponding average ranks differ by at least the critical distance.

The results of the Bonferroni-Dunn post-hoc tests are graphically represented by a simple diagram. Figure 5.7 shows the two results of the analysis of the accuracies from Table 5.4. On




(a)Training Set (b) Test Set

Figure 5.8: *Visualization of accuracy testing with MSD.* Bonferroni-Dunn post-hoc tests for the second experiment using average ranks of algorithms on data from Table 5.5.

Table 5.5: *Accuracy testing with MSD.* Comparison of accuracies achieved by RDR, CST and LemmaGen in the experimental setting using the MSD information.

| | Training Set | | | Test Set | | |
|---|---|---|---|---|---|---|
| Language | RDR | CST | LemmaGen | RDR | CST | LemmaGen |
| Multext-East | | | | | | |
| Slovene | 99.8 ±0.01 | 99.9 ±0.00 | 99.9 ±0.00 | 93.2 ±0.16 | 91.1 ±0.18 | **93.4** ±0.16 |
| Serbian | 99.7 ±0.03 | 99.7 ±0.03 | **99.8** ±0.02 | 85.2 ±0.60 | 83.0 ±0.63 | **86.1** ±0.61 |
| Bulgarian | 99.4 ±0.02 | 99.5 ±0.02 | 99.5 ±0.02 | 93.7 ±0.22 | 84.0 ±0.40 | **94.1** ±0.21 |
| Czech | 99.4 ±0.03 | 99.4 ±0.03 | **99.5** ±0.03 | 90.0 ±0.35 | 89.3 ±0.36 | **90.6** ±0.35 |
| English | 99.7 ±0.02 | 99.9 ±0.01 | 99.9 ±0.01 | 97.5 ±0.12 | 95.4 ±0.23 | **97.7** ±0.17 |
| Estonian | 99.1 ±0.02 | 99.2 ±0.02 | **99.3** ±0.02 | 90.3 ±0.22 | 80.4 ±0.29 | **90.8** ±0.21 |
| Hungarian | 98.7 ±0.03 | 98.8 ±0.04 | **98.9** ±0.03 | 92.1 ±0.25 | 79.9 ±0.34 | **92.3** ±0.24 |
| Romanian | 99.7 ±0.00 | 99.9 ±0.00 | 99.9 ±0.00 | 88.3 ±0.10 | 83.0 ±0.14 | **88.6** ±0.11 |
| Multext | | | | | | |
| English | 99.8 ±0.01 | 99.9 ±0.01 | 99.9 ±0.01 | 97.8 ±0.16 | 96.0 ±0.19 | **98.0** ±0.13 |
| French | 99.6 ±0.01 | 99.8 ±0.01 | 99.8 ±0.01 | 96.8 ±0.09 | 94.8 ±0.13 | **97.1** ±0.08 |
| German | 99.7 ±0.01 | 99.7 ±0.01 | 99.8 ±0.01 | 96.2 ±0.17 | 95.4 ±0.20 | **96.3** ±0.16 |
| Italian | 99.3 ±0.02 | 99.6 ±0.01 | 99.6 ±0.01 | 95.4 ±0.14 | 88.3 ±0.23 | **95.7** ±0.14 |
| Spanish | 99.8 ±0.00 | 99.9 ±0.00 | 99.9 ±0.00 | 98.1 ±0.05 | 97.5 ±0.05 | **98.4** ±0.04 |

the axis of each diagram we plot the average rank of the algorithms. The lowest (best) ranks are to the right. We also show the critical distance on the top, and connect the algorithms that are not significantly different. From the results, we can draw two conclusions. On the training sets, LemmaGen is significantly better than RDR, but not significantly different from CST. However, on the test sets, LemmaGen is in terms of accuracy significantly better than both CST and RDR.

### 5.6.5 Experimental results when using MSDs

In these experiments, MSD information was used, and datasets were separated by language and by MSD. LemmaGen was again compared to RDR and CST.

#### Accuracy testing

The methodology for testing for second experimental setting is the same as in case of first experimental setting, therefore all the results from Table 5.5 are directly comparable to Table 5.4. We notice an overall increase of accuracies in this setting as expected, since MSD tags can be used by algorithms to disambiguate between ambiguous wordforms.

#### Algorithms ranking in terms of accuracy

The ranking methodology is again the same as in the first experimental setting and the conclusions are also similar. In terms of accuracy, LemmaGen is, on training sets, significantly better than RDR, but not significantly different from CST. However, on the test sets, LemmaGen is significantly better than CST and RDR, as shown in Figure 5.8.

### 5.6.6 Accuracy comparison by language

In this section, we briefly compare and discuss the accuracy results achieved by LemmaGen on the test sets according to language, with the results summarized in Table 5.6.

Table 5.6: *Comparison of LemmaGen accuracy by language.* The first column gives the language, the second the number of different MSDs in the lexicon, the third and fourth columns the accuracy and rank for the first experiment with LemmaGen, and the fifth and sixth the accuracy and rank for the second experiment. The rows are sorted according to the rank of the second experiment (when MSDs were used).

| Language | MSDs | Experiment 1 | | Experiment 2 | |
|---|---|---|---|---|---|
| | | Accuracy | Rank | Accuracy | Rank |
| Spanish | 264 | 95.4 | 1 | 98.4 | 1 |
| English MTE | 133 | 92.4 | 2 | 98.0 | 2 |
| English MT | 135 | 90.8 | 3 | 97.7 | 3 |
| French | 380 | 87.5 | 4 | 97.1 | 4 |
| German | 227 | 82.7 | 5 | 96.3 | 5 |
| Italian | 247 | 80.5 | 6 | 95.7 | 6 |
| Bulgarian | 338 | 70.4 | 11 | 94.1 | 7 |
| Slovene | 2,083 | 79.8 | 7 | 93.4 | 8 |
| Hungarian | 619 | 72.3 | 10 | 92.3 | 9 |
| Estonian | 643 | 63.3 | 13 | 90.8 | 10 |
| Czech | 1,428 | 78.3 | 8 | 90.6 | 11 |
| Romanian | 616 | 73.1 | 9 | 88.6 | 12 |
| Serbian | 906 | 65.3 | 12 | 86.1 | 13 |

The accuracy rank of a language corresponds much better to the traditional notion of inflection complexity of languages and language families than do most statistics over the lexicons, with weakly inflecting West-European languages coming first, followed by heavily inflecting Slavic languages. The two non-Indo-European (and non-inflecting, but rather agglutinating) languages, Hungarian and Estonian are grouped together and appear among the Slavic languages.

There are, however, some outliers. Spanish should appear close to Italian, and the fact that it is first seems especially surprising as only about 3% of the wordforms in the lexicon are identical to their lemmas (c.f. Table 5.3), meaning that some morphological operation needs to be performed on practically every wordform. The explanation seems to be, as discussed in Section 5.6.1, that Spanish contains full paradigms for a large number of verbs, but while these paradigms contain many wordforms, the relation of the wordforms to the lemmas is quite regular and simple.

The first Slavic language is, unsurprisingly, Bulgarian, as it has lost almost all its nominal inflection, and is by far the least inflecting among the Slavic languages covered. It is interesting to note that Bulgarian, while being ranked seventh in the second experiment, was only the eleventh in the first—meaning that the lemma of its wordforms is very difficult to predict without recourse to the MSD. Based on the accuracy rank, the next is Slovene, which is also, to a certain extent, an outlier, as it would be expected to perform worse compared to Czech and Serbian. This is, as mentioned in Section 5.6.1, largely due to the different principles in constructing the lexica—Slovene contains full inflectional paradigms, while Czech and Serbian have only the wordforms attested in the reference corpus used to construct the lexicon. They thus cover fewer wordforms per lemma, making for a sparser—and more irregular—dataset.

Next come the two agglutinating languages, Hungarian and Estonian, which have significantly different properties than inflectional ones—but, it would seem from the results, are still simpler than Slavic ones when the construction principles of the lexicons (taking only corpus wordforms and not full paradigms) are the same. After Czech comes, somewhat surprisingly, Romanian, which should intuitively be simpler than Slavic languages, although more complex

Table 5.7: *Efficiency Testing.* Comparison of the efficiencies between RDR, CST and LemmaGen.

| Language | Training time per word [µs] | | | Lemmatization time per word [µs] | | |
|---|---|---|---|---|---|---|
| | RDR | CST | LemmaGen | RDR | CST | LemmaGen |
| Multext-East | | | | | | |
| Slovene | 14.3 | 87.8 | 3.9 | 3.2 | 4.6 | 0.6 |
| Serbian | 9.7 | 58.6 | 5.8 | 2.5 | 5.2 | 0.7 |
| Bulgarian | 22.2 | 72.5 | 4.6 | 5.5 | 5.3 | 0.7 |
| Czech | 10.8 | 63.1 | 3.4 | 2.3 | 4.2 | 0.6 |
| English | 8.2 | 57.4 | 3.2 | 2.0 | 4.8 | 0.6 |
| Estonian | 14.3 | 118.7 | 4.8 | 3.0 | 6.0 | 0.7 |
| Hungarian | 10.5 | 79.2 | 4.5 | 2.2 | 6.0 | 0.7 |
| Romanian | 120.7 | 210.2 | 6.7 | 46.2 | 7.2 | 0.7 |
| Multext | | | | | | |
| English | 7.6 | 52.1 | 3.1 | 1.8 | 4.7 | 0.6 |
| French | 16.6 | 92.5 | 4.6 | 3.7 | 6.2 | 0.7 |
| German | 12.0 | 74.9 | 3.1 | 2.7 | 4.2 | 0.7 |
| Italian | 9.2 | 86.9 | 4.6 | 2.0 | 6.1 | 0.7 |
| Spanish | 17.4 | 92.4 | 4.7 | 4.0 | 7.6 | 0.7 |

than other Romance ones. An inspection of the lexicon reveals that at least a part of the reason is in their treatment of abbreviations and bound wordforms. The first are often expanded (e.g., "ADN" has the lemma "acid dezoxiribonucleic"), while the second exhibit changes also in the first part of the word, rather than only in the suffix (e.g., "-mbarcasem" has the lemma "îmbarca" and "amorul-propriu" the lemma "amor-propriu"). Such cases cannot be adequately covered by a lemmatizer that operates on word suffixes.

The worst ranked is Serbian—but this is most likely due to the much smaller lexicon (training set) than is available for the other languages.

### 5.6.7 Efficiency testing

Table 5.7 shows the results of efficiency testing. Numbers in the table represent the average time (in µs) needed to learn (Training time) and lemmatize (Lemmatization time) one wordform. Hence, the table values should be read as: The lower, the better. We do not provide ranking tests for efficiency results, since it is obvious that LemmaGen is by far more efficient than the other two algorithms.

## 5.7 LemmaGen workflows and modules

This section presents the procedural overview of training and using a lemmatizer inside the ClowdFlows workflow management and execution environment (see Kranjc et al., 2012 and Online reference [34]). The presentation is divided into two parts, Figure 5.9 presents a workflow for training a lemmatizer and Figure 5.10 presents a workflow for utilizing a lemmatizer. For the actual working online executable demo of the presented two workflows, see Online reference [35].

As presented in this chapter, the prerequisite step for lemmatization is lemmatizer training, in other words, a model (RDR tree) for lemmatization needs to be build. Algorithmic aspects of this process (especially component [1.3]) are presented in details in Section 5.4. The procedural steps illustrated in Figure 5.9 are the following:

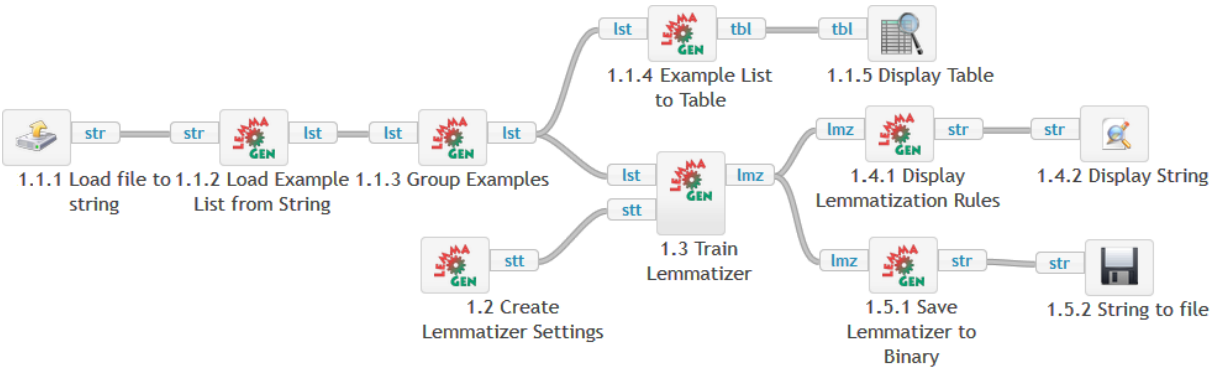



Figure 5.9: *Example of a workflow presenting a lemmatizer training.*

[1.1] load and prepare training examples to be used in the training process,
- [1.1.1] load a file with examples (in a string representation),
- [1.1.2] parse the string to get the actual training examples which are in the form (wordform, lemma, MSD, frequency) where MSD and Frequency are not required to be present,
- [1.1.3] group the examples based on the required training setting, e.g., if MSD tags should be ignored, then all the examples have same wordform and lemma but different MSD can be joined into a single example,
- [1.1.4] transform the grouped examples into a table to be displayed (optional component),
- [1.1.5] verify the examples to inspect if the loading and preparation steps proceeded as planned (optional component),

[1.2] set various lemmatizer training parameters, e.g., how to consider MDS tags, whether to include also example into the model, should the prefix lemmatization rules be learned, etc.,

[1.3] perform—in this thesis developed (see Section 5.4)—algorithm for the actual training of a lemmatizer,

[1.4] visualize the trained lemmatizer model (RDR tree),
- [1.4.1] create a textual (string) representation of the RDR tree lemmatizer model,
- [1.4.2] display the string,

[1.5] save the model description to a file for the future use,
- [1.5.1] serialize the model description into a string,
- [1.5.2] save the string to a file.

The described procedure builds and saves the lemmatization model trained on a given training data. The serialization and saving of the model (see component [1.5.2] above) is required for the elimination of the training phase when the user only wants to apply the lemmatization. Component [2.1.1] in Figure 5.10 and component [2.3.1] in Figure 6.12 (see Section 6.5.2) depend on loading such saved serialized model. Nevertheless, from the procedural perspective, the whole training procedure could just as well be repeated everywhere, where the lemmatizer is required—the lemmatization results would be the same.

Figure 5.10 presents an example of a typical workflow that makes use of a trained lemmatizer. Section 5.5 explains the algorithm working in the background—more specifically in the

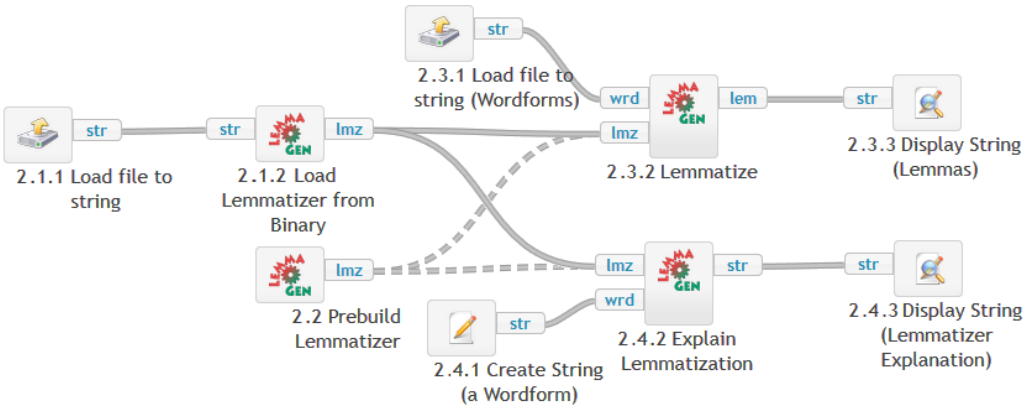



Figure 5.10: *Example of a workflow presenting the usage of a lemmatizer.*

background of component [2.3.2]. The procedural description of the lemmatization process inside the ClowdFlows environment is the following:

[2.1] create a lemmatizer from a saved model,
  [2.1.1] load a serialized model from a file (produced by component [1.5.2] in Figure 5.9),
  [2.1.2] deserialize the model and utilize it to create a new lemmatizer,
[2.2] create a lemmatizer by using the ClowdFlows included pre-trained models for lemmatization (optional component replacing the step in [2.1]),
[2.3] lemmatize wordforms loaded from a file,
  [2.3.1] load the wordforms to be lemmatized from a file,
  [2.3.2] perform the lemmatization of the given wordforms (using the algorithm from Section 5.5),
  [2.3.3] visualize the string with the produced lemmas,
[2.4] produce the explanation of the lemmatization process,
  [2.4.1] create a wordform of interest,
  [2.4.2] apply the lemmatizer to explain the model's decisions when lemmatizing the given wordform,
  [2.4.3] visualize the string describing the lemmatization decisions.

Note the dashed lines in Figure 5.10, connecting component [2.2] with the rest of the workflow. The two dotted lines denote optional connections which replace the usage of the two components in step [2.1]. Component [2.2] presents a simplified creation of lemmatizers by using pre-trained lemmatization models included in ClowdFlows containing 20 models for 18 different languages. Consequently, the user initiated model training presented in Figure 5.9 is not required when only the functionality of lemmatization is requested by the user.

Given the two workflows from Figures 5.9 and 5.10 and combining them into a single workflow brings us to an interesting new scenario, where the user can easily experiment with various training data and lemmatizer training settings. If the lemmatization part of the workflow is supplied with the same data as the training part (or a subset of the data, e.g., using classical data mining cross validation approach), then the user can retrieve various training success measures, e.g., accuracy on 10-fold cross validation. By this approach, new models can be created and their performance can be evaluated.

With these procedural aspects of utilizing a lemmatizer, we conclude the chapter presenting one of the main contributions of this thesis, i.e., the development, evaluation and usage of the LemmaGen lemmatization engine. The following chapter presents the core part of this thesis—the CrossBee methodology for cross-domain knowledge discovery—which utilizes the LemmaGen engine for its text preprocessing needs.

# 6 CrossBee approach to bridging term discovery

This chapter presents the key part of the developed CrossBee methodology, i.e., the mechanism for scoring and ranking the candidate B-terms. The problem addressed is in the field of literature-based discovery known as the closed discovery problem: when having two separate domains in the form of two sets of text documents, the task is to find bridging terms or concepts, which stimulate the scientific creativity in order to generate hypotheses about novel and valuable connections linking both domains. We first present the developed simple heuristic-based approach to bridging term discovery followed by its critical evaluation and description of a more elaborate ensemble approach which is at the end shown to perform well and outperforms the simple approach.

In detail, this chapter is structured as follows: Section 6.1 introduces the heuristics that are used as the basic ingredients of the methodology and establishes the notation and terminology used throughout this chapter. Section 6.2 presents the evaluation of the constructed base heuristics on the well-known literature mining dataset about connections between migraine and magnesium; the section concludes by discussing the drawbacks and potential benefits of using only base heuristics for term bisociative quality evaluation and sketches some possible improvements of the methodology. Section 6.3 presents the approach used to overcome the discussed drawbacks of base heuristics using an ensemble of heuristics to score potential B-terms. Next, Section 6.4 presents the evaluation of the ensemble-based CrossBee methodology on two domains: the first is the already mentioned migraine-magnesium domain, and the second is a more recent domain about connections between autism and calcineurin. Finally, Section 6.5 presents the implementation of the CrossBee methodology as a workflow of reusable components in an online workflow execution environment ClowdFlows.

## 6.1 Base heuristics

We define heuristics as functions that numerically evaluate the term's quality by assigning it a *bisociation score* (measuring the potential that a term is actually a B-term). For the definition of an appropriate set of heuristics, we define a set of special (mainly statistical) properties of terms, which aim at separating B-terms from regular terms; thus, these heuristics can also be viewed as advanced term statistics.

All heuristics operate on the data retrieved from the documents in text preprocessing. Ranking all the terms using the scores calculated by an ideal heuristic should result in finding all the B-terms together at the top of a ranked list. This is an ideal scenario, which is not realistic; however, ranking by heuristic scores should at least increase the proportion of B-terms at the top of the ranked term list.

Formally, a heuristic is a function with two inputs, i.e., a set of domain labeled documents $D$ and a term $t$ appearing in these documents, and one output, i.e., a score that correlates with the term's bisociation score.

In this thesis, we use the following notation: to state that the bisociation score $b$ is equal to the result of a heuristic named $heurX$, we can write it as $b = heurX\ (D, t)$. However, since the set of input documents is static when dealing with a concrete dataset, we can—for the sake of simplicity—omit the set of input documents from a heuristic notation and use only $b = heurX\ (t)$. Whenever we need to explicitly specify the set of documents to which the function is applied (never needed for a heuristic, but sometimes needed for auxiliary functions used in the formula for the heuristic), we write it as $funcX_D(t)$. For specifying the function's input document set, we have two options: either use $D_u$ that stands for the (union) set of all the documents from all the domains, or use $D_n : n \in \{1..N\}$, which stands for the set of documents from the given domain $n$. In general, the following statement holds: $D_u = \bigcup_{n=1}^{N} D_n$ where $N$ is the number of domains. In the most common scenario, when there are exactly two distinct domains, we also use the notation $D_A$ for $D_1$ and $D_C$ for $D_2$, since we introduced $A$ and $C$ as representatives of the initial and the target domain in the closed discovery setting introduced in Section 2.4.2. Due to a large number of heuristics and auxiliary functions, we use the so-called camel casing multi-word naming scheme for easier distinction; names are formed by word concatenation and capitalization of all non-first words (e.g., *freqProdRel* and *tfidfProduct*).

It is worth noting that all the designed heuristics are symmetric in the domains, as switching the order of the domains (which domain is the initial domain and which is the target) does not affect the calculated value of a heuristic. By allowing asymmetric heuristics, the approach loses the possibility to generalize it to more than two domains.

We divided the heuristics into different sets for easier explanation; however, most of the described heuristics work fundamentally in a similar way—they all manipulate solely the data present in term and document vectors and derive the terms' bisociation score. The exceptions to this are the outlier-based heuristics, which first evaluate outlier documents and only later use the information from the term vectors for B-term evaluation. We can thus define three sets of base heuristics: BoW-based, outlier-based and the baseline heuristics. The following sections describe each set in more detail.

### 6.1.1 Heuristics based on BoW representation

The BoW-based heuristics manipulate the data present in term and document vectors (see Section 3.3 for details about representation) to derive the terms' bisociation score. They can be divided to: frequency-based, TF-IDF-based, and similarity-based.

#### Frequency-based heuristics

We first define two auxiliary functions:

- $countTerm_D(t)$: counts the number of occurrences of term $t$ in a document set $D$ (called term frequency in TF-IDF related contexts),
- $countDoc_D(t)$: counts the number of documents in which term $t$ appears in a document set $D$, (called document frequency in TF-IDF related contexts).

We define the following base heuristics:

- $freqTerm(t) = countTerm_{D_u}(t)$: term frequency in the two domains,

- $freqDoc(t) = countDoc_{D_u}(t)$: document frequency in the two domains,
- $freqRatio(t) = \frac{countTerm_{D_u}(t)}{countDoc_{D_u}(t)}$: term to document frequency ratio,
- $freqDomnRatioMin(t) = \min\left(\frac{countTerm_{D_1}(t)}{countTerm_{D_2}(t)}, \frac{countTerm_{D_2}(t)}{countTerm_{D_1}(t)}\right)$: minimum of term frequencies ratio of the two domains,
- $freqDomnProd(t) = countTerm_{D_1}(t) \cdot countTerm_{D_2}(t)$: product of term frequencies of the two domains,
- $freqDomnProdRel(t) = \frac{countTerm_{D_1}(t) \cdot fcountTerm_{D_2}(t)}{countTerm_{D_u}(t)}$: product of term frequencies of the two domains relative to the term frequency in all domains.

## TF-IDF-based heuristics

TF-IDF is a standard measure of term's importance in a document, which is used heavily in text mining research (see Equation 1 on page 28 for detailed TF-IDF definition). In the following heuristic definitions, we use the following auxiliary functions:

- $tfidf_{\mathrm{d}}(t)$ stands for TF-IDF weight of a term $t$ in a document $d$,
- $tfidf_D(t)$ represents TF-IDF weight of a term in the centroid vector of all the documents $d: d \in D$, where centroid vector is defined as an average of all document vectors and thus presents an average document of the document collection $D$.

Heuristics based on TF-IDF are listed below:

- $tfidfSum(t) = \sum_{d \in D_u} tfidf_d(t)$: sum of all TF-IDF weights of a term in the two domains; this heuristic is analogous to $freqTerm(t)$,
- $tfidfAvg(t) = \frac{\sum_{d \in D_u} tfidf_d(t)}{freq\_doc_{D_u}(t)}$: average TF-IDF weights of a term across all domains,
- $tfidfDomnProd(t) = tfidf_{D_1}(t) \cdot tfidf_{D_2}(t)$: product of a term TF-IDF weights in the two domains,
- $tfidfDomnSum(t) = tfidf_{D_1}(t) + tfidf_{D_2}(t)$: sum of a term TF-IDF weights in the two domains.

## Similarity-based heuristics

Another approach to construct a relevant heuristic measure is to use the cosine similarity measure. We start by creating a representational model as a document space and by converting terms into document vectors (see Section 3.3). Next, we get the centroid vectors for both domains in the document space representation. Finally, we apply TF-IDF weighting on top of all the newly constructed vectors and centroids. We define the following auxiliary function:

- $simCos_D(t)$: calculates the cosine similarity of the document vector of term $t$ and the document vector of a centroid of documents $d \in D$.

The base heuristics are the following:

- $simAvgTerm(t) = simCos_{D_u}(t)$: similarity to an average term—the distance from the center of the cluster of all terms,
- $simDomnProd(t) = simCos_{D_1}(t) \cdot simCos_{D_2}(t)$: product of a term similarity to the centroids of the two domains,
- $simDomnRatioMin(t) = \min\left(\frac{simCos_{D_1}(t)}{simCos_{D_2}(t)}, \frac{simCos_{D_2}(t)}{simCos_{D_1}(t)}\right)$: minimum of a term frequencies ratio of the two domains.

### 6.1.2 Outlier-based heuristics

Outlier detection is an established area of data mining (Hodge and Austin, 2004). Conceptually, an outlier is an unexpected event, entity or—in our case—a document. We are especially interested in outlier documents since they frequently embody new information that is often hard to explain in the context of existing knowledge. Moreover, in data mining, an outlier is occasionally a primary object of study as it can potentially lead to the discovery of new knowledge. These assumptions are well aligned with the bisociation potential that we wish to optimize, thus, we have constructed several heuristics that harvest the information possibly residing in outlier documents.

We concentrate on a specific type of outliers, i.e., domain outliers, which are the documents that tend to be more similar to the documents of the opposite domain than to those of their own domain. The techniques that we use to detect outlier documents (Sluban and Lavrač, 2010; Lavrač et al., 2010; Sluban et al., 2011, 2012) is based on using classification algorithms to detect outlier documents. First we train a classification model for each domain and afterwards classify all the documents using the trained classifier. The documents that are misclassified—according to their domain of origin—are declared as outlier documents, since according to the classification model they do not belong to their domain of origin.

We defined three different outlier sets of documents based on three classification algorithms utilized. These outlier sets are:

- $D_{CS}$: documents misclassified by the Centroid Similarity (CS) classifier,
- $D_{RF}$: documents misclassified by the Random Forest (RF) classifier,
- $D_{SVM}$: documents misclassified by the Support Vector Machine (SVM) classifier.

Centroid similarity is a basic classifier model implemented in our system. It classifies each document to the domain whose centroid's TF-IDF vector is the most similar to the document's TF-IDF vector. The description of the other two classification models is beyond the scope of this thesis, as we used external procedures to retrieve these outlier document sets; the detailed description is provided by Sluban et al. (2012).

For each outlier set we defined two heuristics: the first counts the frequency of a term in an outlier set and the second computes the relative frequency of a term in an outlier set compared to the relative frequency of a term in the whole dataset. The resulting heuristics are listed below:

- $outFreqCS(t) = countTerm_{D_{CS}}(t)$: term frequency in the CS outlier set,
- $outFreqRF(t) = countTerm_{D_{RF}}(t)$: term frequency in the RF outlier set,
- $outFreqSVM(t) = countTerm_{D_{SVM}}(t)$: term frequency in the SVM outlier set,
- $outFreqSum(t) = countTerm_{D_{CS}}(t) + countTerm_{D_{RF}}(t) + countTerm_{D_{SVM}}(t)$: sum of term frequencies in all three outlier sets,
- $outFreqRelCS(t) = \frac{countTerm_{D_{CS}}(t)}{countTerm_{D_u}(t)}$: relative frequency in the CS outlier set,
- $outFreqRelRF(t) = \frac{countTerm_{D_{RF}}(t)}{countTerm_{D_u}(t)}$: relative frequency in the RF outlier set,
- $outFreqRelSVM(t) = \frac{countTerm_{D_{SVM}}(t)}{countTerm_{D_u}(t)}$: relative frequency in the SVM outlier set,
- $outFreqRelSum(t) = \frac{countTerm_{D_{CS}}(t)+countTerm_{D_{RF}}(t)+countTerm_{D_{SVM}}(t)}{countTerm_{D_u}(t)}$: sum of relative term frequencies in all three outlier sets.

### 6.1.3 Baseline heuristics

We have two other heuristics, which are supplementary and serve as a baseline for the others. The auxiliary functions used in their calculation are:

- $randNum()$: returns random number from the interval (0,1) regardless of the term under investigation,
- $inBoth(t)$: 1 if a term $t$ appears in both domains and 0 otherwise.

The two baseline heuristics are:

- $random(t) = randNum()$ : random baseline heuristic,
- $appearInAllDomn(t) = inBoth(t) + (randNum())/2$ : is a better baseline heuristic which can separate two classes of terms, the ones that appear in both domains and the ones that appear only in one. The terms that appear only in one domain have a strictly lower heuristic score than those that appear in both. The score inside of these two classes is still random.

## 6.2 Base heuristics evaluation

This section presents the evaluation of the heuristics defined in the previous section. First, we describe the evaluation procedure, next we present the domain used in the evaluation, and finally show the results of the evaluation along with the discussion of the results.

### 6.2.1 Evaluation procedure

Experimental data for the evaluation of heuristics' quality is required to contain at least the following two ingredients: a set of documents from two domains and a "gold standard" list of B-terms associated with these two domains. Given this data, one is able to identify the true B-terms and evaluate how well the heuristics are able to promote these B-terms compared to the rest of the terms. In this context, promoting means that a heuristic predicts high score for the term which puts the term on the top of the list of terms ordered by their predicted scores. Ideally, all true B-terms should have high scores and all the other terms should have low scores. Nevertheless, typically a list of B-terms is not exhaustive for the domain pair, which means that only a subset of all the bisociative terms between the domains is actually listed in the gold standard list.

Consequently, the evaluation measure to be used should not penalize a heuristic, which promotes also some of the other, non-listed, terms as long as it also promotes the actual B-terms. The reasoning is that the promoted terms may also be B-terms, but were until now undiscovered and therefore not listed. Based on this reasoning, a ranking/ordering metric should be used to evaluate the heuristic performance. Rank metrics depend only on the order of the terms based on prediction scores ignoring the actual value of the predicted scores. If the ordering is preserved, it makes no difference if the predicted scores change (Caruana and Niculescu-Mizil, 2004).

There are several alternatives among ranking metrics, e.g., area under the ROC curve (AUC), average precision (APR), precision/recall break-even point (BEP) (Caruana and Niculescu-Mizil, 2004). We use ROC (Receiver Operating Characteristic) curve and AUC (Area

Under ROC curve) to compare the performance of heuristics. Our decision is simply based on the fact that the ROC metric is well known and there exists a large volume of easily accessible research on the ROC and AUC metrics (e.g., Provost and Fawcett, 2001). Additionally, the supplementary ROC convex hull (ROCCH) analysis can be utilized to choose the optimal set of heuristics (Fawcett, 2006; Flach and Wu, 2005) to be employed together to better predict term's bisociation score by exploiting the synergy (performed in Section 6.3.2).

For the purpose of our evaluation, the ROC curves are constructed in the following way:

- Sort all the terms by their descending heuristic score.
- Starting from the beginning of the term list, do the following for each term: if a term is a B-term, then draw one vertical line segment (up) on the ROC curve, else draw one horizontal line segment (right) in the ROC space.
- Sometimes, a heuristic outputs the same score for many terms and therefore we cannot sort them uniquely. Among all the terms with the same bisociation score $b$, let $\mathrm{b}_b$ be the number of terms that are B-terms and $\mathrm{nb}_b$ the number of non-B-terms. We then draw a line from the current point $p$ to the point $p + (\mathrm{nb}_b, \mathrm{b}_b)$. In this way, we may produce slanted lines if such an equal scoring term set contains both B-terms and non B-terms.

Using the above procedure, we get one ROC curve for each heuristic. The ROC space is defined by its two axes. The ROC's vertical axis scale goes from zero to the number of B-terms and the horizontal goes from zero to the number of non B-terms. Such approach, using absolute numbers on the axes, is uncommon when applying ROC curves for the analysis. Typically, normalized values are used and both axes span from 0 to 1. Nevertheless, our goal is to keep the absolute numbers since they are very valuable when interpreting the ROC curve, e.g., we are able to quickly identify how many B-terms are among the best $n$ ranked terms. Otherwise—besides the labels on both axes—using the absolute numbers does not change the shape of the curve or the interpretation in any way.

AUC is defined as the percentage of the area under curve—the area under the curve is divided by the area of the whole ROC space. If a heuristic is perfect (it detects all the B-terms and ranks them at the top of the ordered list), we get a curve that goes first just up and then just right with an AUC of 100%. The worst possible heuristic sorts all the terms randomly regardless of being a B-term or not and achieves AUC of 50%. The heuristic *random* is represented by the diagonal in the ROC space. Interestingly, achieving AUC below 50% is not necessary bad. It just means that the ranking procedure was applied incorrectly, i.e., that the ranked list should be inversed, and the resulting achieved AUC is above 50%.

The fact that some heuristics output the same score for many terms can produce different sorted lists and thus different performance estimates for the same heuristic on the same dataset. In the case of such equal scoring term sets, the inner sorting is random (which indeed produces different performance estimates). However, the ROCs constructed by the instructions above correspond to an average ROC over all possible random inner sortings. Besides AUC, we list also the interval of AUC, which tells how much each heuristic varies among the best and the worst sorting of a possibly existing equal scoring term set. Preferable are the heuristics with a smaller interval, which implies that they produce smaller and fewer equal scoring sets.

### 6.2.2 The migraine-magnesium dataset

This section describes the dataset used to evaluate the base heuristics' potential for successful B-term ranking. The dataset that we used is the well-researched *migraine-magnesium* domain pair, which was introduced by Swanson (1988) and was later explored by Swanson (1990), Weeber et al. (2001), Swanson et al. (2006), Petrič et al. (2009) and others. In the literature-based discovery process, Swanson managed to find more than 60 pairs of articles connecting the migraine domain with the magnesium deficiency via 43 different B-terms. In our evaluation, we tried to rediscover these B-terms stated by Swanson to connect the two domains (see Table 6.1).

The dataset contains scientific paper titles which were retrieved by querying the PubMed database with the keyword *"migraine"* for the migraine domain and with the keyword *"magnesium"* for the magnesium domain. An additional condition to the query was the publishing date, which was limited to before the year 1988, since Swanson's original experiment—which we want to reproduce—also considered only articles published before that year. The query resulted in 8,058 titles of the average length of 11 words (2,425 from the migraine domain and 5,633 from the magnesium domain). We preprocessed the dataset using the standard procedures described in Section 3.2 and by additionally specifying terms as n-grams of maximum length 3 (max. three words were combined to form a term) with minimum occurrence 2 (each n-gram had to appear at least twice to be promoted to a term). Using these preferences, we produced a dataset containing 13,525 distinct terms or 1,847 distinct terms that appear at least once in each domain; both numbers include also all the 43 terms that Swanson marked as B-terms. Respectively, the final numbers of non B-terms are 13,483 and 1,804, as will be observable in following figures. An average document in the dataset consists of 12 terms and 394 (4.89%) documents contain at least one B-term.

### 6.2.3 Comparison of the heuristics

This section presents the results of the comparison of base heuristics on the magnesium-migraine dataset. Table 6.2 presents the results of the AUC analysis for all the heuristics, while Figure 6.1 plots the ROC curves for the best heuristic and the two baselines. We show these two results in order to demonstrate the important property of the researched problem and data, which remained yet unobserved. The fact is that all the 43 B-terms appear in both domains; therefore, it is reasonable for the evaluation that the heuristics are also aware of this fact. In other words, all the terms that do not appear in both domains (11,678 terms) should be discarded and not used in bisociation scoring and ranking since the basic property of B-terms—at least in this dataset—is that they appear in both datasets.

The fact presented in the previous paragraph can be suspected already from Table 6.2 where we observe very good results with relatively small AUC differences among the heuristics. However, the issue becomes obvious from Figure 6.1 which shows that even the baseline heuristic *appearInAllDomn* is able to retrieve all the true B-terms before the position 2,000 (or 1,804 to be exact). The same goes for *outFreqRelSum* as well as for all the other heuristics, which are not plotted in Figure 6.1. Consequently, it is irrelevant to plot the ROC curve and to calculate the AUC for positions beyond 1,804, or in general, beyond the position of the last non B-term that appears in both domains.

Table 6.1: *Migraine-magnesium B-terms.* As identified by Swanson et al. (2006).

| | | | | | |
|---|---|---|---|---|---|
| 1 | 5 ht | 16 | convulsive | 31 | prostaglandin |
| 2 | 5 hydroxytryptamine | 17 | coronary spasm | 32 | prostaglandin e1 |
| 3 | 5 hydroxytryptamine receptor | 18 | cortical spread depression | 33 | prostaglandin synthesis |
| 4 | anti aggregation | 19 | diltiazem | 34 | reactivity |
| 5 | anti inflammatory | 20 | epilepsy | 35 | seizure |
| 6 | anticonvulsant | 21 | epileptic | 36 | serotonin |
| 7 | antimigraine | 22 | epileptiform | 37 | spasm |
| 8 | arterial spasm | 23 | hypoxia | 38 | spread |
| 9 | brain serotonin | 24 | indomethacin | 39 | spread depression |
| 10 | calcium antagonist | 25 | inflammatory | 40 | stress |
| 11 | calcium blocker | 26 | nifedipine | 41 | substance p |
| 12 | calcium channel | 27 | paroxysmal | 42 | vasospasm |
| 13 | calcium channel blocker | 28 | platelet aggregation | 43 | verapamil |
| 14 | cerebral vasospasm | 29 | platelet function | | |
| 15 | convulsion | 30 | prostacyclin | | |

Table 6.2: *AUC analysis of the heuristics.* Comparison of the results of all the defined heuristics ordered by the AUC quality. The first column states the name of the heuristic; the second displays a percentage of the area under the ROC curve; and the last is the interval of AUC.

| Heuristic | AUC | Interval | | | |
|---|---|---|---|---|---|
| *outFreqRelSum* | 95.33% | 0.35% | *freqDomnProdRel* | 93.71% | 0.40% |
| *outFreqRelRF* | 95.24% | 0.55% | *simDomnRatioMin* | 93.58% | 0.00% |
| *outFreqRelSVM* | 95.06% | 1.26% | *tfidfSum* | 93.58% | 0.00% |
| *outFreqRelCS* | 94.96% | 1.30% | *tfidfDomnProd* | 93.47% | 0.39% |
| *outFreqSum* | 94.96% | 0.70% | *freqDomnProd* | 93.42% | 0.44% |
| *tfidfAvg* | 94.87% | 0.00% | *freqRatio* | 93.35% | 5.23% |
| *outFreqRF* | 94.73% | 1.53% | *appearInAllDomn* | 93.31% | 6.69% |
| *outFreqSVM* | 94.70% | 2.06% | *simDomnProd* | 93.27% | 0.00% |
| *outFreqCS* | 94.67% | 1.80% | *freqTerm* | 93.20% | 0.50% |
| *freqDomnRatioMin* | 94.36% | 0.62% | *freqDoc* | 93.19% | 0.50% |
| *tfidfDomnSum* | 93.85% | 0.35% | *simAvgTerm* | 92.71% | 0.00% |
| | | | *random* | 50.00% | 50.00% |

Table 6.3: *Corrected AUC analysis of the heuristics.* AUC analysis corrected for the fact that only 1,847 terms are used as B-term candidates.

| Heuristic | AUC | Interval | | | |
|---|---|---|---|---|---|
| *outFreqRelSum* | 65.10% | 2.62% | *freqDomnProdRel* | 52.99% | 2.99% |
| *outFreqRelRF* | 64.43% | 4.11% | *simDomnRatioMin* | 52.02% | 0.00% |
| *outFreqRelSVM* | 63.08% | 9.42% | *tfidfSum* | 52.02% | 0.00% |
| *outFreqRelCS* | 62.33% | 9.72% | *tfidfDomnProd* | 51.20% | 2.91% |
| *outFreqSum* | 62.33% | 5.23% | *freqDomnProd* | 50.83% | 3.29% |
| *tfidfAvg* | 61.66% | 0.00% | *freqRatio* | 50.30% | 39.09% |
| *outFreqRF* | 60.62% | 11.43% | *appearInAllDomn* | 50.00% | 50.00% |
| *outFreqSVM* | 60.39% | 15.40% | *simDomnProd* | 49.70% | 0.00% |
| *outFreqCS* | 60.17% | 13.45% | *freqTerm* | 49.18% | 3.74% |
| *freqDomnRatioMin* | 57.85% | 4.63% | *freqDoc* | 49.11% | 3.74% |
| *tfidfDomnSum* | 54.04% | 2.62% | *simAvgTerm* | 45.52% | 0.00% |
| | | | *random* | 6.69% | 50.00% |

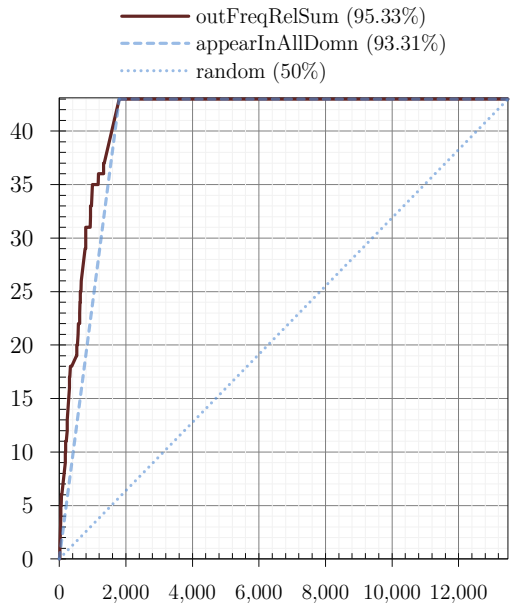



Figure 6.1: *ROC analysis of the best performing heuristic and the two baselines.* ROC curve of the selected best heuristic *outFreqRelSum* along with the baseline heuristic *random* and improved baseline heuristic *appearInAllDomn* on ranking the 43 B-terms among all 13,525 terms.

From this point on, all the presented results are presented based on the reasoning that only terms appearing in both domains can become B-term candidates. In the currently observed migraine-magnesium dataset, this results in only 1,847 B-term candidates. Consequently, the result presentation changes significantly since the problem of ranking the filtered list of produced better B-term candidates is much harder, compared to the problem of ranking all the terms. Figure 6.2 presents the same results for the same heuristics as Figure 6.1 but in the discussed changed perspective. The same transformation was also applied to Table 6.2 and resulted in Table 6.3. Though the presented results are the same, the difference in the perspective is obvious.

The first look at the numeric result comparison (Table 6.3) reveals the following:

- The overall AUC results of all heuristics is in the range of from approx. 45% to 65%, which is at the first sight not very impressive.
- The improved baseline heuristic, *appearInAllDomn,* has 50% AUC, which is as expected.
- Some heuristics' AUC is below 50%, which reveals that they contain some information about correct B-term scoring, but are defined in the opposite way. If their scores were inversed, their AUC would rise above 50%, e.g., *simAvgTerm* would achieve 54.48%.
- Some heuristics, including the best performing ones, have a relatively high AUC interval, which means that they output the same score for many terms.
- Overall, the outlier-based heuristics seem to perform the best.

Observing the results in Table 6.3, followed by the detailed ROC analysis described below, we select the best heuristic to be used as the heuristic for providing bisociation scores of terms. The chosen heuristic is *outFreqRelSum*, the first from the list in Table 6.3, due to the fact that it has the highest AUC and especially since it shows a low uncertainty. In other words, it has a small AUC interval, which means that it better defines the position of B-terms and we do not need to rely so much on random sorting of equal scoring term sets. We also assume it to be less volatile across different domains since it actually represents cooperation (the sum) of the three other well performing heuristics: *outFreqRelRF*, *outFreqRelSVM*, and *outFreqRelCS*.

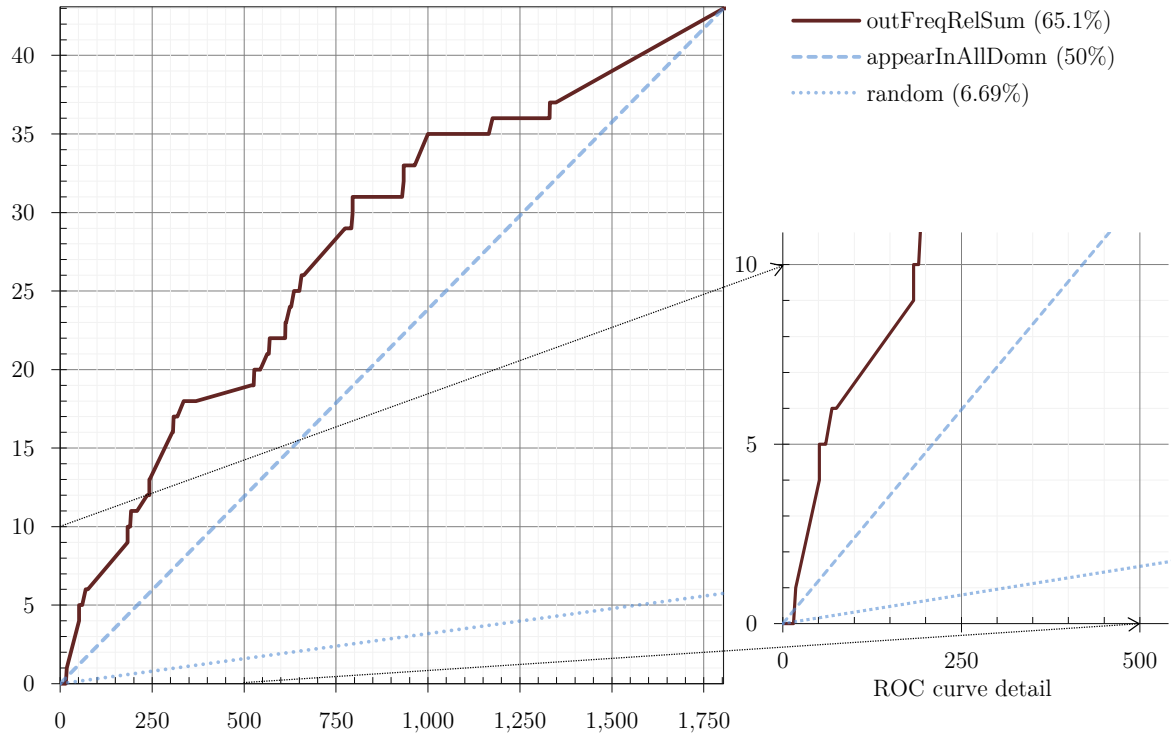



Figure 6.2: *ROC analysis of the best performing heuristic and the two baselines.* The same results for the same heuristics as in Figure 6.1 but using only 1,847 terms (appearing in both domains) as B-Term candidates. The left hand side chart shows the view on the whole ROC space, while the right hand side show a zoom-in view on the top 500 terms and the first 10 B-terms.

Detailed ROC curve analysis of the chosen heuristic (see Figure 6.2) shows that our heuristic is better than the improved baseline heuristic, but—at a first sight—not as much as we would hope for. However, when examined carefully (see right hand side of Figure 6.2) we perceive that the *outFreqRelSum* heuristic has an extremely steep incline at the beginning of the curve, which is much steeper than the incline of the baseline heuristics. This means that the chosen heuristic is able to detect B-terms at the beginning of the ordered list much faster than the baseline. The steep incline can be seen also in other heuristics, shown in Figures 6.3 and 6.4.

Detailed analysis of the initial steep incline property of the *outFreqRelSum* heuristic results in the following observations. At the position of the second gridline on axis x on the right hand side of Figure 6.2 (by the term 50 in the ordered list of terms), the chosen heuristic is able to detect already 4-5 B-terms while the baseline heuristic only approximately one. Similarly, we notice that at the 200$^{\text{th}}$ term, the baseline heuristics detects 5 B-terms while *outFreqRelSum* detects already 11 and at the 300$^{\text{th}}$ term the ratio is 7:16. If we follow the curve further (on the left hand side of Figure 6.2), we see a decrease in relative difference; nevertheless, at the 1000$^{\text{th}}$ term, the ratio is still 24:35, even though the performance here is not so important as the performance at the beginning of the curve. The presented behavior at the beginning of the curve is highly appreciated especially from the point of view of the expert who needs to go through such an ordered list of terms and detect potential B-terms. In such a setting, our goal is to push as much as possible of valuable B-terms to the very beginning of the list, even if other B-terms are dispersed evenly across the entire terms list.

Figure 6.3 shows the comparison of outlier-based heuristics. Although overall performance is similar, we again observe differences in initial incline property where *outFreqRelRF* is the best.

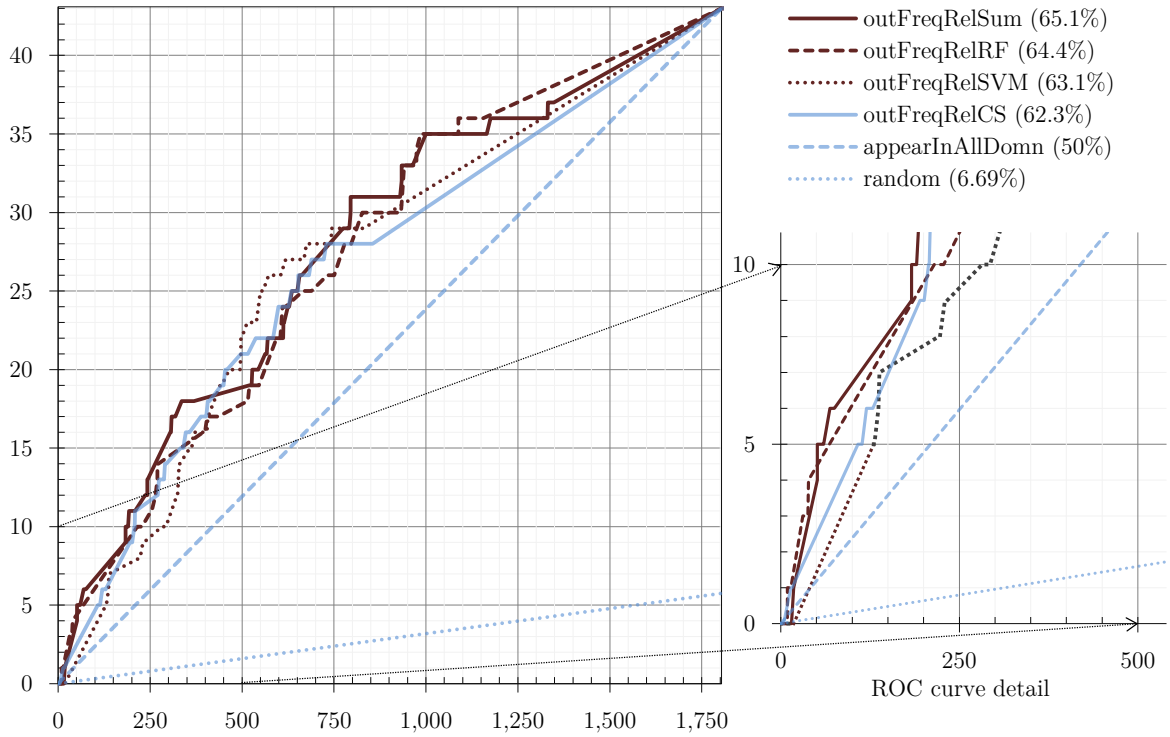



Figure 6.3: *Outlier heuristics ROC comparison.* ROC curves of the outlier-based set of heuristics.

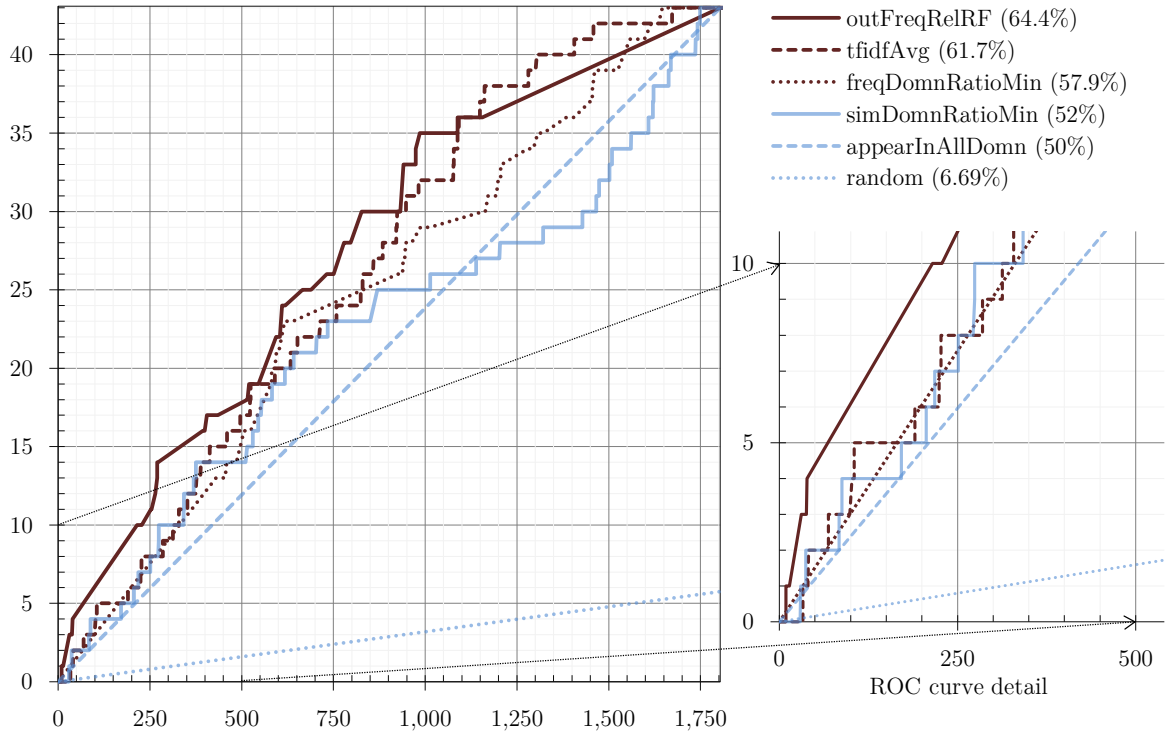



Figure 6.4: *Best performing heuristics ROC comparison.* ROC curves of the best-performing heuristics—one from each set (based on: frequency, TF-IDF, similarity, outliers) along with both baseline heuristics on detecting the B-terms among 1,847 candidates.

Even though we found the best heuristic from the outlier set, we are still interested in how the heuristics from the other sets performed. This comparison is shown in Figure 6.4 where we present the best performing heuristic from each set of heuristics. Notice the outlier heuristic *outFreqRelRF* that undoubtedly wins. It is harder to establish an order between the other three heuristics. The undesired property is exposed by *simDomnRatioMin* where the ROC curve shows worse performance than *appearInAllDomn* at the right side of the curve; however, this would be tolerable if there was improved performance at the beginning of the curve. The conclusion for the heuristics, besides the outlier ones, is that even though they are slightly better than the baseline heuristic we are not able to decide which outperforms the others.

### 6.2.4 Open issues concerning the use of base heuristics

Overall, the results of the evaluation are beneficial for the insight into heuristic performance on the examined migraine-magnesium dataset. The conclusion is that it is hard to promote B-terms in a ranked list of terms by observing only the terms' statistical properties in the documents. However, we managed to construct a well performing heuristic which is based on the relative frequency of a term in three outlier document sets which are retrieved using the three classifiers: Centroid Similarity, Random Forest, and, Support Vector Machine. The conclusion of our evaluation is well aligned with the results presented by Sluban et al. (2012) and Petrič et al. (2012b).

Nevertheless, we identified another problem of the proposed B-term ranking when we began to experimentally apply the selected single heuristic to new domains, not limited to migraine and magnesium. The problem arises from the fact that the process of selection of a single heuristic is prone to overfitting the training dataset (in our case migraine-magnesium), which results in heuristics' performance instability across other datasets. As long as our experiments were performed only on a single dataset, i.e., the migraine-magnesium dataset, the results of the selected single heuristic (i.e., *outFreqRelSum,* which proved to be the best heuristic on this dataset) were stable, even if we used various modifications of data preprocessing, removed random documents from the set, randomly deleted words from documents or did some other data perturbations.

## 6.3 CrossBee ensemble heuristic

In the CrossBee methodology proposed so far we estimated the bisociation potential of a term by calculating its bisociation score using different base heuristics—as presented in Section 6.1—which are indeed good candidates for providing bisociation scores. Although the evaluation of base heuristics on the migraine-magnesium dataset in Section 6.2 shows some promising properties, we are not yet satisfied with the results. The discussion in Section 6.2.4 shows the pitfalls of using the base heuristics' scores directly and explains why the choice of the right heuristic for a particular domain is far from trivial. A solution to ease the stated problem, proposed in this section, is to combine multiple heuristics into an ensemble heuristic, i.e., a heuristic that combines the results of multiple base heuristics into one aggregated result, which in less sensitive to the variability of domain characteristics.

Ensemble learning is a known approach used in machine learning for combining predictions of multiple models into one final prediction. It is well known (Dietterich, 2000) that the result-

ing ensemble model is more accurate than any of the individual models used to build it as long as the models are similarly accurate, are better than random, and their errors are uncorrelated. There is a wide variety of known and well tested ensemble techniques, e.g., bagging, boosting, majority voting, random forest, naïve Bayes, etc. (Rokach, 2009). However, these approaches are usually used for the problem of classification while the core problem presented in this work is ranking. Nevertheless, with the rise of the areas like information retrieval and search engines' web page rankings; the ensemble ranking is also gaining research attention (e.g., Dwork et al., 2001; Hoi and Jin, 2008).

One possible—and probably the most typical—approach to designing an ensemble heuristic from a set of base heuristics consists of two steps. In the first step, the task is to select member heuristics for the ensemble heuristic using standard data mining approaches like feature selection. In the second step, equation discovery is used to obtain an optimal combination of member heuristics. The advantage of such approach is that the ensemble creation does not require manual intervention. Therefore, we performed several experiments with this approach; however, the results of an ensemble were even more overfitted to the training domain. Consequently, we decided to manually—based on experiences and experimentation—select appropriate base heuristics and construct an ensemble heuristic. As the presentation of numerous experiments, which support our design decisions, is beyond the scope of this thesis, we describe only the final solution, along with some reasoning about choosing the heuristics.

### 6.3.1 Ensemble construction

The ensemble heuristic, which we constructed based on the experiments, is constructed from two parts: the *ensemble voting score* and the *ensemble position score,* which are summed together to give the final *ensemble score*.

The *ensemble voting score* ($s_t^{vote}$) of a given term $t$ is an integer, which denotes how many base heuristics voted for the term. Each selected base heuristic $h_i$ gives one vote ($s_{t_j,h_i}^{vote} = 1$) to each term, which is in the first third in its ranked list of terms and zero votes to all the other terms ($s_{t_j,h_i}^{vote} = 0$). The voting threshold one third (1/3) was set empirically grounded on the evaluation of the ensemble heuristic on the migraine-magnesium domain and is based on the number of the terms, which appear in both domains (not one third of all the terms). Formally, the ensemble voting score of a term $t_j$ that is at position $p_j$ in the ranked list of $n$ terms is computed as a sum of individual heuristics' voting scores:

$$s_{t_j}^{vote} = \sum_{i=1}^{k} s_{t_j,h_i}^{vote} = \sum_{i=1}^{k} \begin{cases} 1\text{: } p_j < n/3, \\ 0\text{: } otherwise \end{cases}. \tag{4}$$

Therefore, each term can get a score $s_{t_j}^{vote} \in \{0, 1, 2, \ldots, k\}$, where $k$ is the number of base heuristics used in the ensemble.

The *ensemble position score* ($s_t^{pos}$) is calculated as an average of position scores of individual base heuristics. For each heuristic $h_i$, the term's *position score* $s_{t_j,h_i}^{pos}$ is calculated as $(n - p_j)/n$, which results in position scores being in the interval [0,1). For an ensemble of $k$ heuristics, the ensemble position score is computed as an average of individual heuristics' position scores:

$$s^{pos}_{t_j} = \frac{1}{k}\sum_{i=1}^{k} s^{pos}_{t_j,h_i} = \frac{1}{k}\sum_{i=1}^{k} \frac{(n-\mathrm{p}_j)}{n}. \quad (5)$$

The final *ensemble score* is computed as:

$$s_t = \; s^{vote}_t + s^{pos}_t. \quad (6)$$

Using the proposed construction we make sure that the integer part of the ensemble score always presents the ensemble vote score, while the ensemble score's fractional part always presents the ensemble position score. An ensemble position score is strictly lower than 1, therefore a term with a lower ensemble voting score can never have a higher final ensemble score than a term with a higher ensemble voting score.

Note that at the first sight, our method of constructing the ensemble score looks rather intricate. An obvious way to construct an ensemble score of a term could be simply to sum together individual base heuristics scores; however, the calculation of the ensemble score by our method is well justified by extensive experimental results on the migraine-magnesium dataset.

The described method for ensemble score calculation is illustrated in Figure 6.5. In the upper left table, the base heuristics scores are shown for each term. The next table presents terms ranked according to the base heuristics scores. From this table, the voting and position scores are calculated for every term based on its position, as shown in the upper right table. For example, all terms at position 2, i.e., $t_1$, $t_6$, and $t_6$, get voting score 1 and position score 4/6. The central table below shows the exact equation how these individual base heuristics' voting and position scores are combined for each term. The table at the bottom displays the list of terms ranked by the calculated ensemble scores.

| | Base scores | | |
|---|---|---|---|
| Term | $h_1$ | $h_2$ | $h_3$ |
| $t_1$ | 0.93 | 0.46 | 0.33 |
| $t_2$ | 0.26 | 0.15 | 0.10 |
| $t_3$ | 0.51 | 0.22 | 0.79 |
| $t_4$ | 0.45 | 0.84 | 0.73 |
| $t_5$ | 0.41 | 0.15 | 0.11 |
| $t_6$ | 0.99 | 0.64 | 0.74 |

Base heuristic scores

| | Base ranking | | |
|---|---|---|---|
| Pos. | $h_1$ | $h_2$ | $h_3$ |
| 1 | $t_6$ | $t_4$ | $t_3$ |
| 2 | $t_1$ | $t_6$ | $t_6$ |
| 3 | $t_3$ | $t_1$ | $t_4$ |
| 4 | $t_4$ | $t_3$ | $t_1$ |
| 5 | $t_5$ | $t_2$ | $t_5$ |
| 6 | $t_2$ | $t_5$ | $t_2$ |

Terms ranked by base heuristics

| | Voting score | Position score |
|---|---|---|
| Pos. | $s^{vote}_{t_j,h_i}$ | $s^{pos}_{t_j,h_i}$ |
| 1 | 1 | $(6-1)/6 = 5/6$ |
| 2 | 1 | $(6-2)/6 = 4/6$ |
| 3 | 0 | $(6-3)/6 = 3/6$ |
| 4 | 0 | $(6-4)/6 = 2/6$ |
| 5 | 0 | $(6-5)/6 = 1/6$ |
| 6 | 0 | $(6-6)/6 = 0/6$ |

Voting and position scores based on positions in the ranked lists

| | Voting score sum | | | | | | + | Pos. score average = | | | | | | | Ensemble score | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ( | $\mathrm{s}^{\mathrm{vote}}_{\mathrm{t_j,h_1}}$ | + | $\mathrm{s}^{\mathrm{vote}}_{\mathrm{t_j,h_2}}$ | + | $\mathrm{s}^{\mathrm{vote}}_{\mathrm{t_j,h_3}}$ | ) + ( | $\mathrm{s}^{\mathrm{pos}}_{\mathrm{t_j,h_1}}$ | + | $\mathrm{s}^{\mathrm{pos}}_{\mathrm{t_j,h_2}}$ | + | $\mathrm{s}^{\mathrm{pos}}_{\mathrm{t_j,h_3}}$ | )/k = | $\mathrm{s}^{\mathrm{vote}}_{\mathrm{t_j}}$ | + | $\mathrm{s}^{\mathrm{pos}}_{\mathrm{t_j}}$ | = | $\mathrm{s}_{\mathrm{t_j}}$ |
| $s_{t_1}$ | = ( | 1 | + | 0 | + | 0 | ) + ( | 4/6 | + | 3/6 | + | 2/6 | )/3 = | 1 | + | 9/18 | = | 1.50 |
| $s_{t_2}$ | = ( | 0 | + | 0 | + | 0 | ) + ( | 0/6 | + | 1/6 | + | 0/6 | )/3 = | 0 | + | 1/18 | = | 0.06 |
| $s_{t_3}$ | = ( | 0 | + | 0 | + | 1 | ) + ( | 3/6 | + | 2/6 | + | 5/6 | )/3 = | 1 | + | 10/18 | = | 1.56 |
| $s_{t_4}$ | = ( | 0 | + | 1 | + | 0 | ) + ( | 2/6 | + | 5/6 | + | 3/6 | )/3 = | 1 | + | 10/18 | = | 1.56 |
| $s_{t_5}$ | = ( | 0 | + | 0 | + | 0 | ) + ( | 1/6 | + | 0/6 | + | 1/6 | )/3 = | 0 | + | 2/18 | = | 0.11 |
| $s_{t_6}$ | = ( | 1 | + | 1 | + | 1 | ) + ( | 5/6 | + | 4/6 | + | 4/6 | )/3 = | 3 | + | 13/18 | = | 3.72 |

Calculation of ensemble heuristic score

| |
|---|
| $t_6$ (3.72), [$t_2$, $t_3$] (1.56), $t_1$ (1.50), $t_5$ (0.11), $t_2$ (0.06) |

Ranked list of terms produced by the ensemble

Figure 6.5: *Example of ensemble construction.* Illustrated on a simple example with six terms and three heuristics. The last table states the result—the ranked list of terms.

### 6.3.2 Selecting base heuristics for the ensemble

Another important decision when constructing the ensemble is the selection of base heuristics. Table 6.3 shows the results that influenced our decision, which base heuristics to select. The measure used for heuristic performance comparison is AUC. Our final set of heuristics included in the ensemble is the following:

- *outFreqRelRF*
- *outFreqRelSVM*
- *outFreqRelCS*
- *outFreqSum*
- *tfidfDomnSum*
- *freqRatio*

Our initial idea was to choose one (possibly the best performing) heuristic from each set. The rationale behind the idea was to include the top performing heuristics that are as independent as possible. In such a way, the combined information provided by the constructed ensemble was expected to be higher than the information contributed by the individual heuristics. However, certain additional decisions were made to maximize ensemble performance on the migraine-magnesium dataset as well as due to trying not to overfit this dataset:

- The first observation (see Table 6.3) is that all outlier heuristics based on relative term frequency, i.e., *outFreqRelRF*, *outFreqRelSVM*, and, *outFreqRelCS* perform very well. Actually the only heuristic that is better is the *outFreqRelSum*, which is the combination of all these three. As we want to emphasize the power of this best performing set, we include all three heuristics into the ensemble instead of only *outFreqRelSum*. So they get more votes and a chance to over-vote some other—not so well performing—heuristics.
- A representative heuristic of the second outlier heuristic set, based on absolute term frequency, is *outFreqSum,* which is not only the best performing heuristic among non-relative outlier heuristics, but it also integrates the votes of other three heuristics from this set and is therefore the best candidate.
- Representatives of BoW heuristics based on frequency and TF-IDF were chosen in a way that tries to avoid overfitting the migraine-magnesium dataset. The reasoning is based on the ROC convex hull (ROCCH) analysis (Flach and Wu, 2005; Fawcett, 2006), which selects the base heuristics depending on the shape of the ROC curve. The best heuristics are those, whose behavior in ROC space is complementary and together forming the maximal area under ROCCH. Based on this reasoning we chose *freqRatio* and *tfidfDomnSum heuristics.*
- We completely discarded all the heuristics of the type similarity, as their performance is in the range of the baseline heuristic *appearInAllDomn*.

Using the set of heuristics selected in this section and the equations for combining them into an ensemble heuristic presented in the previous section, we are ready to test the ensemble methodology on a new domain. The next section presents this evaluation.

Table 6.4: *Autism-calcineurin B-terms.* Terms identified by Petrič et al. (2009) as B-terms for the autism-calcineurin dataset.

| | | | | | |
|---|---|---|---|---|---|
| 1 | synaptic | 6 | bcl 2 | 11 | 22q11 2 |
| 2 | synaptic plasticity | 7 | type 1 diabetes | 12 | maternal hypothyroxinemia |
| 3 | calmodulin | 8 | ulcerative colitis | 13 | bombesin |
| 4 | radiation | 9 | asbestos | | |
| 5 | working memory | 10 | deletion syndrome | | |

## 6.4 CrossBee evaluation

This section presents the evaluation of the base and ensemble heuristics used in the CrossBee methodology. The key result of this evaluation is the assessment how well the proposed ensemble heuristic performs when ranking the terms. Furthermore, we are interested in the ranking from the perspective of the domain expert who acts as the end-user of the CrossBee system (the system is presented in Section 6.5). From the expert's point of view, the ROC curves and AUC statistics—as used in the evaluation of base heuristics in Section 6.2—are not the most crucial information about the quality of a single heuristic. However, in general it still holds that a better ROC curve reflects a better heuristic. Usually the user is interested in questions like: (a) how many B-terms are likely to be found among the first $n$ terms in a ranked list (where $n$ is a selected number of terms the expert is willing to inspect, e.g., 5, 20 or 100), or (b) how much one can trust a heuristic if a new dataset is explored.

To evaluate the final methodology based on the constructed ensemble of base heuristics, we applied it to two problems or datasets. The first one is once again the migraine-magnesium dataset (Swanson, 1988, 1990), which we used already in Section 6.2 and represents a gold standard in literature mining. The second dataset, which acts as a test case to show the domain independence of the methodology, is the autism-calcineurin domain pair introduced and researched by Urbančič et al. (2007) and Petrič et al. (2009).

### 6.4.1 Experimental setting

The evaluation was performed based on two datasets (or two domain pairs, since each dataset consists of two domains), which can be viewed as a training and test dataset. The training dataset is the dataset we employed when developing the methodology, i.e., for creating a set of base heuristics in Section 6.1 as well as for creating the ensemble heuristic presented in Section 6.3. The results of the evaluation on the training dataset are important, but needs to be interpreted carefully due to a danger of overfitting the dataset. The test dataset is used for the evaluation of the methodology.

As the training data, we used the migraine-magnesium domain pair, which was presented in detail in Section 6.2.2. Additionally, Table 6.5 provides the summary of the dataset's most relevant attributes in comparison to the test dataset.

For the test, dataset we used the *autism-calcineurin* domain pair, which was introduced and initially researched by Urbančič et al. (2007) and later also in Petrič et al. (2009, 2012a). Like Swanson, Petrič et al. (2009) also discovered the B-terms, 13 in total (listed in Table 6.4), whose importance in connecting autism to calcineurin (a protein phosphatase) is discussed and confirmed by the domain expert. In the view of searching for B-terms, this dataset has a differ-

Table 6.5: *Datasets statistics.* Comparison of some statistical properties of the two datasets used in the experiments.

| | | Migraine-magnesium | Autism-calcineurin |
|---|---|---|---|
| Acquisit. | Source | PubMed | PubMed |
| | Query terms | "migraine", "magnesium" | "autism", "calcineurin" |
| | Additional conditions | Year < 1988 | / |
| | Part of paper used | Title | Abstract |
| Docum. Statistics | Number | 8,058 (2,415-5,633) | 15,243 (9,365-5,878) |
| | Doc. with B-term | 394 (4.89%) | 1672 (10.97%) |
| | Avg. words per doc. | 11 | 180 |
| | Outliers (CS/SVM/RF) | (505/362/896) | (377/292/142) |
| Term Statistic | Avg. term per doc. | 7 | 173 |
| | Distinct terms | 13,525 | 322,252 |
| | B-term candidates | 1,847 | 78,805 |
| | Defined B-terms | 43 | 13 |

ent dimensionality compared to the migraine-magnesium dataset. On the one hand, it has only approximately one fourth of the B-terms defined, while on the other hand, it contains more than 40 times as many potential B-term candidates. Therefore, the ratio between B-terms and candidate terms is substantially lower—approximately by factor 160, i.e., the chance to find a B-term among the candidate terms if picking it at random is 160 times lower in the autism-calcineurin dataset than in the magnesium-migraine dataset. Consequently, finding the actual B-terms in the autism-calcineurin dataset is much more difficult compared to the migraine-magnesium dataset.

Table 6.5 states some properties for comparing the two datasets used in the evaluation. One of the major differences between the datasets is the length of an average document since only the titles were used in the migraine-magnesium dataset, while the full abstracts were used in the autism-calcineurin case—in order to match the settings of the experiments in the original research (Swanson, 1988; Urbančič et al., 2007) on these two datasets. Consequently, also the number of distinct terms and B-term candidates is much larger in the case of the autism-calcineurin dataset. Nevertheless, the preprocessing of both datasets was the same with the exception of outlier document identification. For the needs of RF and SVM outlier-based heuristics, we used the outlier documents identified by Sluban et al. (2012) since we did not implement RF and SVM classifiers ourselves. Thus, our outlier heuristics results are completely aligned with the results provided in (Sluban et al., 2012) for both datasets; however, Sluban et al. used slightly different document preprocessing for each of the two datasets. Table 6.5 also shows the exact number of outliers identified in each dataset. We can inspect higher numbers in the migraine-magnesium dataset, which points to the problem of harder classification of documents in this dataset; this is also partly due to shorter texts.

### 6.4.2 Results in the migraine-magnesium dataset

Figure 6.6 shows the comparison of ranking performance for the ensemble and all the base heuristics on the migraine-magnesium dataset. The heuristics are ordered by their AUC. Black dots along with percentages show the heuristic's AUC performance. Gray bars around the AUC central point show the interval of a heuristic's AUC result, explained below.

The property of heuristics having AUC on the interval and not as a fixed value is due to the fact that some heuristics do not produce unambiguous rankings, e.g., see equal ensemble scores for terms $t_2$ and $t_3$ in Figure 6.5. This issue and its handling was explained in Section 6.2.

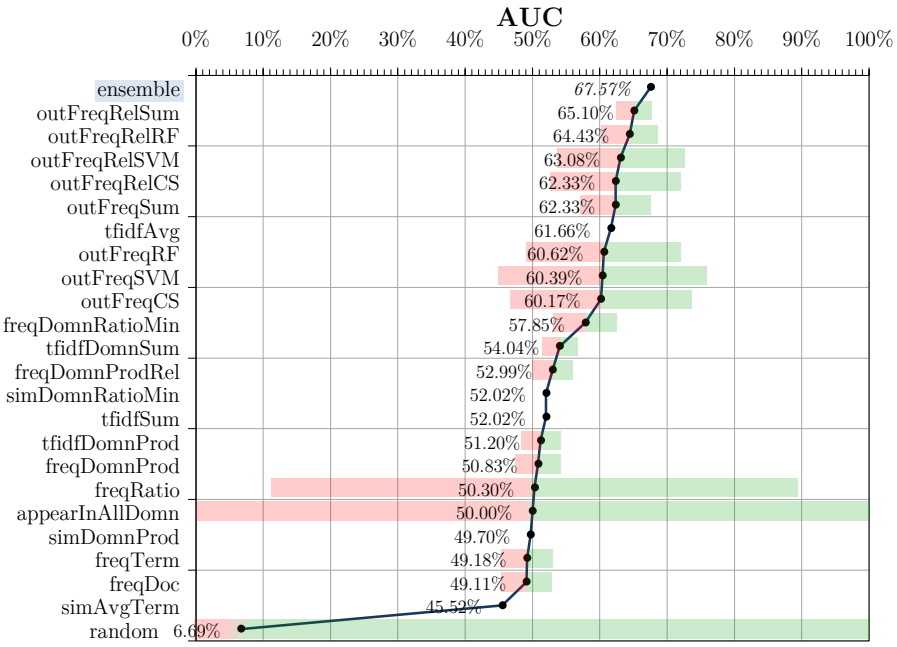



Figure 6.6: *Migraine-magnesium AUC results.* Graphical representation of the AUC measure for all the individual heuristics and the ensemble heuristic on the migraine-magnesium dataset.

Figure 6.6 shows no surprises among the base heuristics, since the results are equal to those presented in Table 6.3 of Section 6.2.3. However, when focusing on the ensemble heuristic, we notice that it is better in both, higher AUC value and lower AUC interval compared to all the other heuristics. We constructed the ensemble using also two not so well performing heuristics (*tfidfDomnSum* and *freqRatio*) in order to avoid overfitting the ensemble on the training domain. This could have a negative effect to the ensemble performance; however, the ensemble performance was not seriously affected, which provides evidence of right decisions made when designing the ensemble. The training dataset was indeed used for designing the ensemble; therefore, the notion in the previous statement is expected.

As stated in the introduction of this section, we are mostly interested in the heuristics quality from the end user's perspective. Such evaluation of heuristics quality is shown in Figure 6.7, where the length of colored bars indicates how many B-terms were found among the first 5, 20, 100, 500 and 2000 terms on the ranked list of terms produced by a heuristic. We can see that the ensemble finds one B-term among the first 5 terms (the darkest gray bar), one B-term—no additional B-terms—among the first 20 terms (no bar), 6 B-terms—5 additional—among the first 100 terms (lighter gray bar), 22 B-terms—16 additional—among first 500 terms (even lighter gray bar) and all the 43 B-terms—21 additional—among the first 2,000 terms (the lightest gray bar). Thus, if the expert limits himself to inspect only the first 100 terms, he will find 6 B-terms in the ensemble list, slightly more than 6 in the *outFreqRelSum* list, 6 in the *outFreqRelRF*, and so on. Results in Figure 6.7 confirm that the ensemble is among the best performing heuristics also from the end-user's perspective. Even though a strict comparison depends also on the threshold of how many terms an expert is willing to inspect, the ensemble is always among the best.

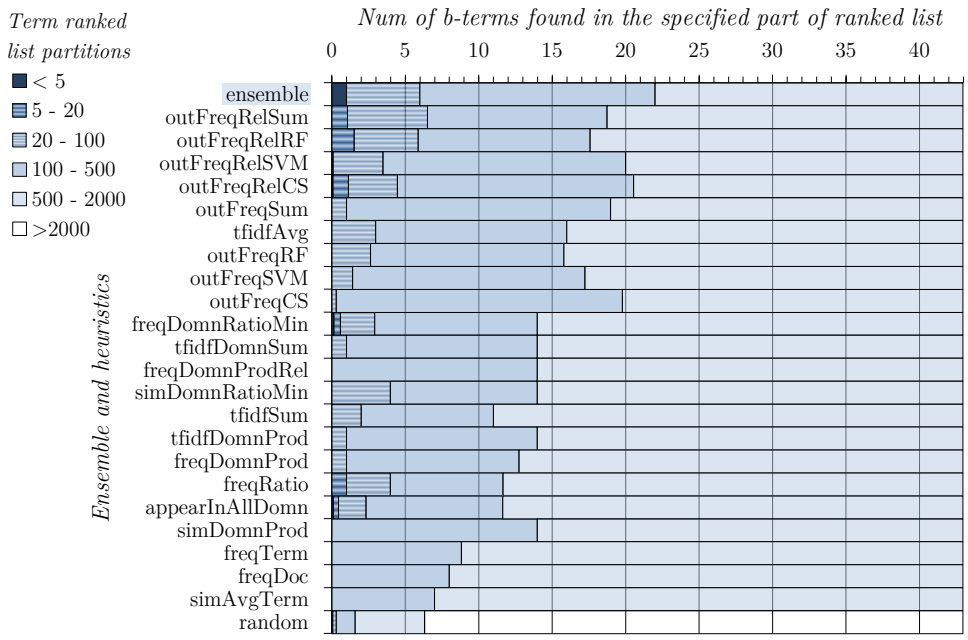



Figure 6.7: *Migraine-magnesium ranking quality results.* Comparison of the ensemble and base heuristics capacity to rank the B-terms at the very beginning of the terms list for the migraine-magnesium dataset.

### 6.4.3 Results in the autism-calcineurin dataset

Figure 6.8 shows how our methodology works on a new independent test dataset, which was not used in the development of the methodology. As discussed, the dimensionality of the autism-calcineurin dataset is considerably different and less favorable compared to the migraine-magnesium dataset. This is evident also when observing Figure 6.8 since the performance of individual base heuristics significantly changes. Some of the originally best performing heuristics, e.g., based on relative frequency in outlier sets are now among the worst and the other types, e.g., TF-IDF-based that were not performing well before, are now among the best. The most important observation is that the ensemble heuristic is still among the best (placed after *freqRatio* and *outFreqSum*) and exposes small uncertainty—note that there is no line marking the AUC interval around the ensemble score. Otherwise, we can notice a slight AUC increase of the best performing heuristics, which is very positive since the candidate term list is much longer now and we expect to find the same number of B-terms much later in the candidate term list compared to the migraine-magnesium dataset.

The last result in this section is the user-oriented visualization of heuristics performance shown in Figure 6.9. This gives us the final argument for the quality of the ensemble heuristic since it outperforms or at least equals to all the other heuristics on the most interesting ranked list lengths (up to 20, 100, 500 terms). The ensemble finds one B-term among 20 ranked terms, 2 among 100 and 3 among 500 ranked terms. At a first sight, this may seem a bad performance, but, note that there are 78,805 candidate terms, which the heuristics have to rank. The evidence of the quality of the ensemble can be understood if we compare it to the *appearInAllDomn* heuristic, which is the baseline heuristic and represents the performance, which is achievable without developing the methodology presented in this work. The *appearInAllDomn* heuristic discovers in average only approximately 0.33 B-terms before position 2000

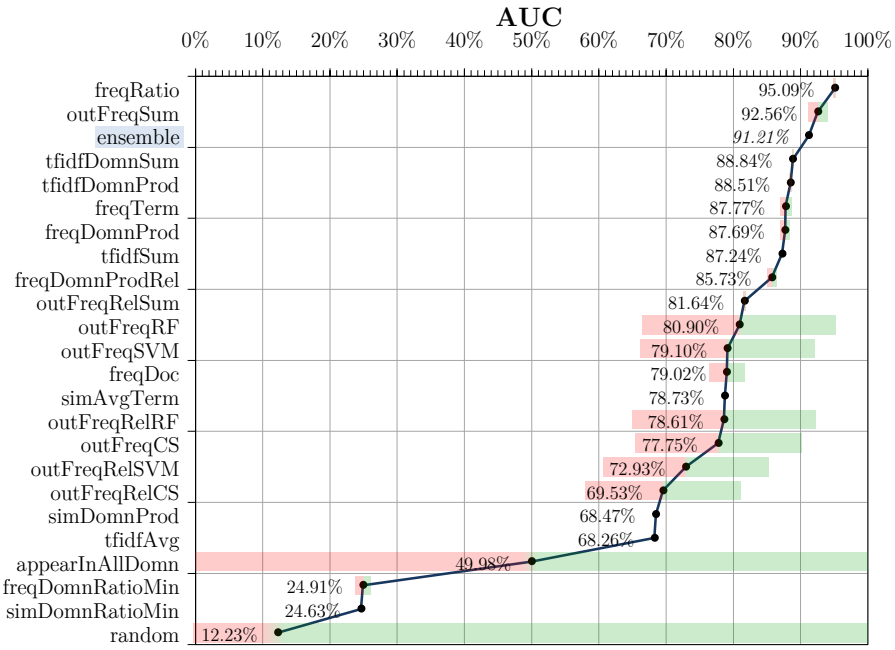



Figure 6.8: *Migraine-magnesium AUC results.* Graphical representation of the ensemble and base heuristics ranking the B-terms at the very beginning of the terms list for the autism-calcineurin dataset. The longer the dark part, the more B-terms a heuristic ranks at the specified partition of the ranked list.

in the ranked list while the ensemble discovers 5; not to mention the shorter term lists where ensemble has even better ratio compared to the *appearInAllDomn* heuristic.

With the user-concerned presentataion of results we conclude the CrossBee methodology evaluation section, which showed that there is great potential for the methodlolgy in terms of supporting the B-term discovery task. The following section presents the methodlolgy from the workflow perspective.

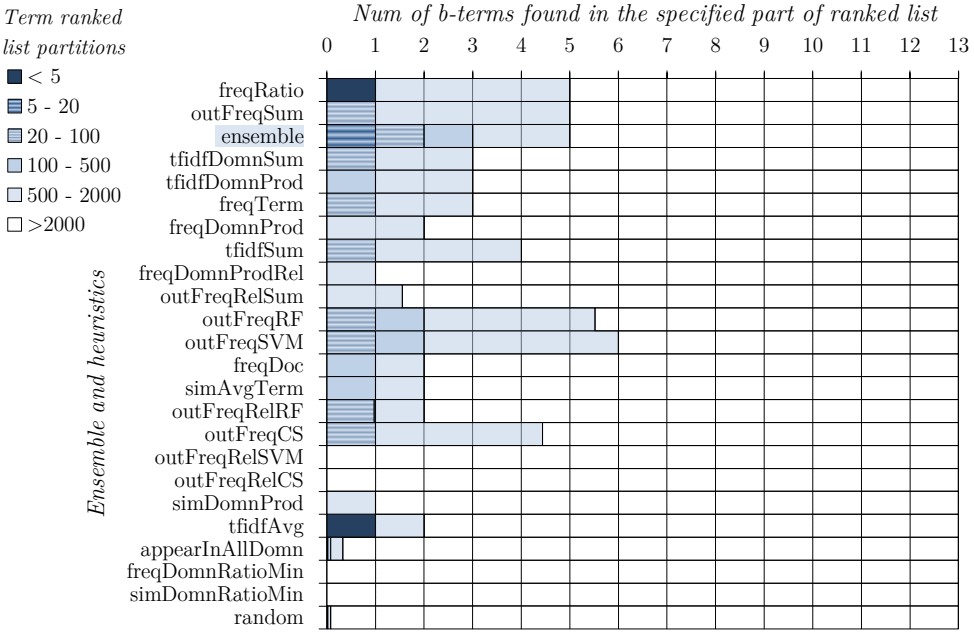



Figure 6.9: *Autism-calcineurin ranking quality results.* Comparison of the ensemble and base heuristics capacity to rank the B-terms at the very beginning of the terms list for the autism-calcineurin dataset.

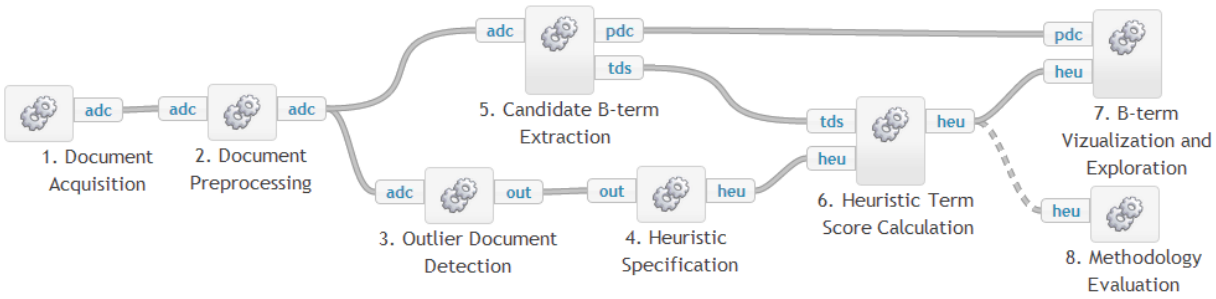



Figure 6.10*: Steps of the overall cross-domain literature mining process.*

## 6.5 CrossBee workflows and modules

Based on the overview of the cross-domain knowledge discovery methodology presented in Section 4.3, this section provides a more detailed description of the entire methodology workflow, together with the details of individual workflow components. Every section below represents one of the steps of the overall workflow, presented in Figure 6.10 which is identical to Figure 4.1 and is repeated here for the sake of clarity. For the link to the online executable version of the presented workflows see Online reference [35].

### 6.5.1 Document acquisition

The first step of the workflow—document acquisition—is shown in Figure 6.11. The output is the annotated document corpus (ADC) consisting of the acquired documents labeled with domain labels. The components are responsible for the following tasks:

[1.1] load literature A into ADC data structure,
  - [1.1.1] load raw text data from a file (this component could be replaced by loading documents from the web or by acquiring them using web services),
  - [1.1.2] build the annotated document corpus from the raw data, i.e., parse the loaded raw text data into a collection of documents,
  - [1.1.3] assign a domain label (e.g., “literature A”, “docsA”, “migraine”) to the documents to enable their identification after merging with literature B,

[1.2] load literature B into the ADC data structure (individual components are aligned with the components [1.1]),

[1.3] merge the two literatures into a single ADC structure,

[1.4] optional check of document acquisition by visual inspection of the created corpus.

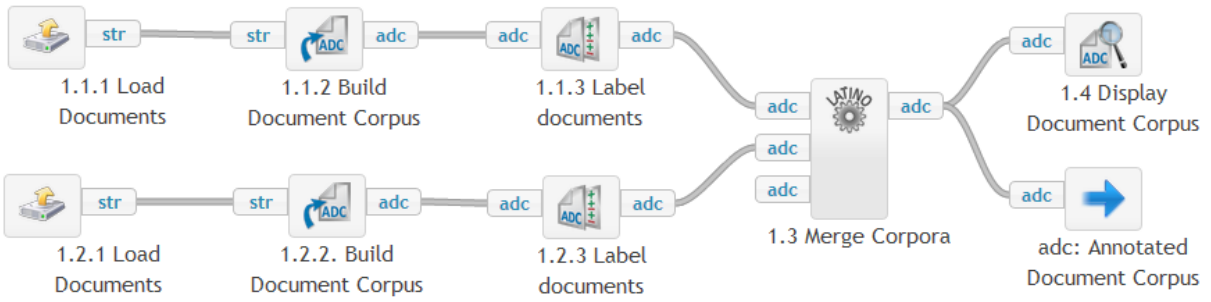



Figure 6.11: *Document acquisition.*

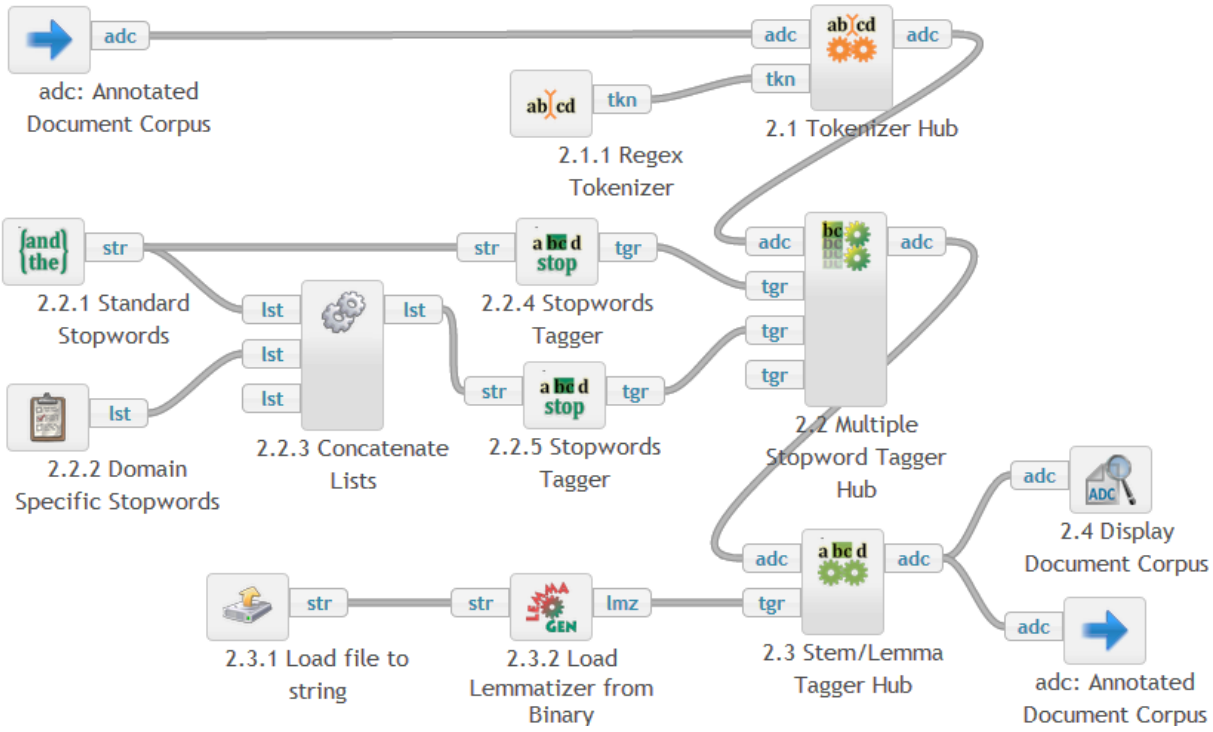



Figure 6.12: *Document preprocessing.*

### 6.5.2 Document preprocessing

Document acquisition presented in the previous section is followed by text preprocessing shown in Figure 6.12. The overall goal of text preprocessing is to tokenize the documents, to detect stopwords and to lemmatize the tokens. The output is structurally equal to the input; however every document in the ADC now contains additional information about tokens, stopwords and lemmas. The specific components perform the following tasks:

[2.1] split documents to tokens (the basic elements for further text processing),
  [2.1.1] create tokenizer object (simple tokenizer of based on regular expressions)

[2.2] tag stopword tokens by using two different stopword taggers (components [2.2.4] and [2.2.5] both tag stopwords in the documents, however the output of the first is used by component [5.1], while the output of the second is used by component [3.1]—the two components require different sets of stopwords to be tagged—component [3.1] requires also the domain specific words to be tagged, due to the outlier detection specifics),
  [2.2.1] load standard English stopwords,
  [2.2.2] define additional domain specific stopwords (these are usually the words used in a query for selecting the domains of interest),
  [2.2.3] concatenate the two lists of stopwords,
  [2.2.4] define the first stopword tagger using the standard English stopwords only (detected stopwords are used in the candidate B-term extraction component [5.1]),
  [2.2.5] define the second stopword tagger using the standard and domain specific stopwords (detected stopwords are used in the document outlier detection component [3.1]),

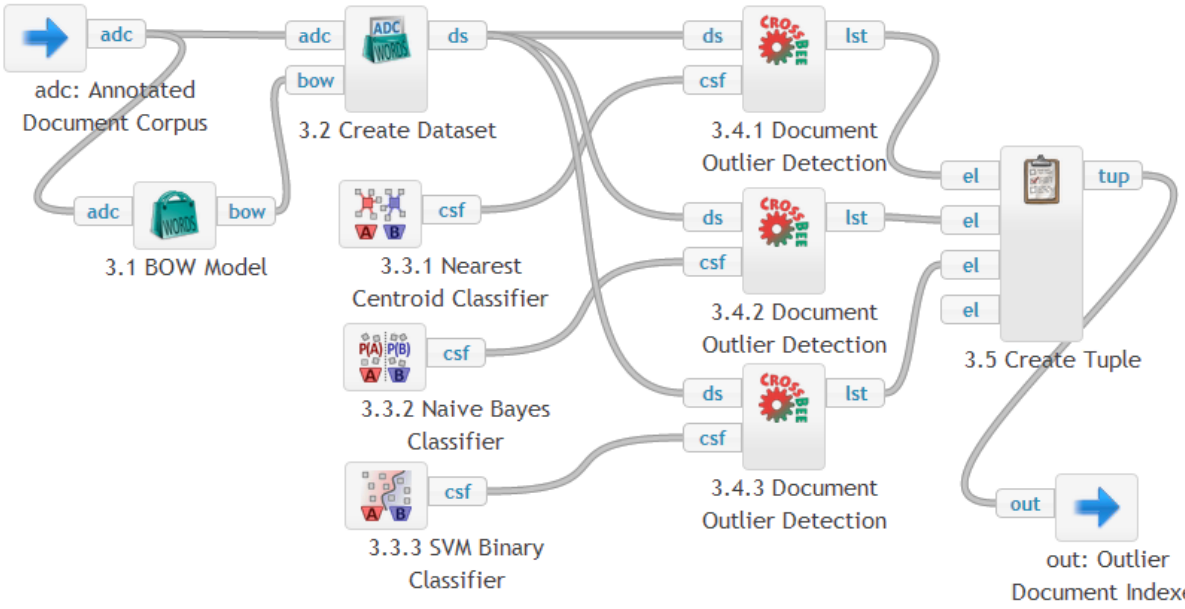



Figure 6.13: *Outlier document detection.*

[2.3] lemmatize tokens using the LemmaGen lemmatizer developed in this thesis (see Section 5),

[2.3.1] load the serialized model definition from a file (the exact workflow for producing this file is presented in Section 5.7, Figure 5.9 where component [1.5.2] saves such a file),

[2.3.2] create a lemmatizer by deserializing the loaded model definition (Section 5.7 shows an alternative to using a file-based model—see component [2.2] in Figure 5.10, which enables the usage of pre-build models included in ClowdFlows),

[2.4] verify the correctness of the whole document preprocessing step by visual inspection of the resulting document corpus (optional component).

### 6.5.3 Outlier document detection

Document outlier detection step is shown in Figure 6.13. The reasoning and idea behind classification-based outlier detection is based on the work of (Sluban et al., 2012) and has been presented in more detail in Section 6.1.2. The outlier document detection procedure is constructed from the following components:

[3.1] construct BoW model for representing documents in the form of feature vectors (see Section 3.2 and 3.3 for more details),

[3.2] using the BoW model to process the documents creating a feature vector representation of every document and produce document dataset,

[3.3] define classification algorithms to be used for outlier detection,

[3.3.1] define the nearest centroid (also known as Rocchio) algorithm for document classification,

[3.3.2] define the naïve Bayes algorithm for document classification,

[3.3.3] define the binary support vector machine (SVM) algorithm for document classification,

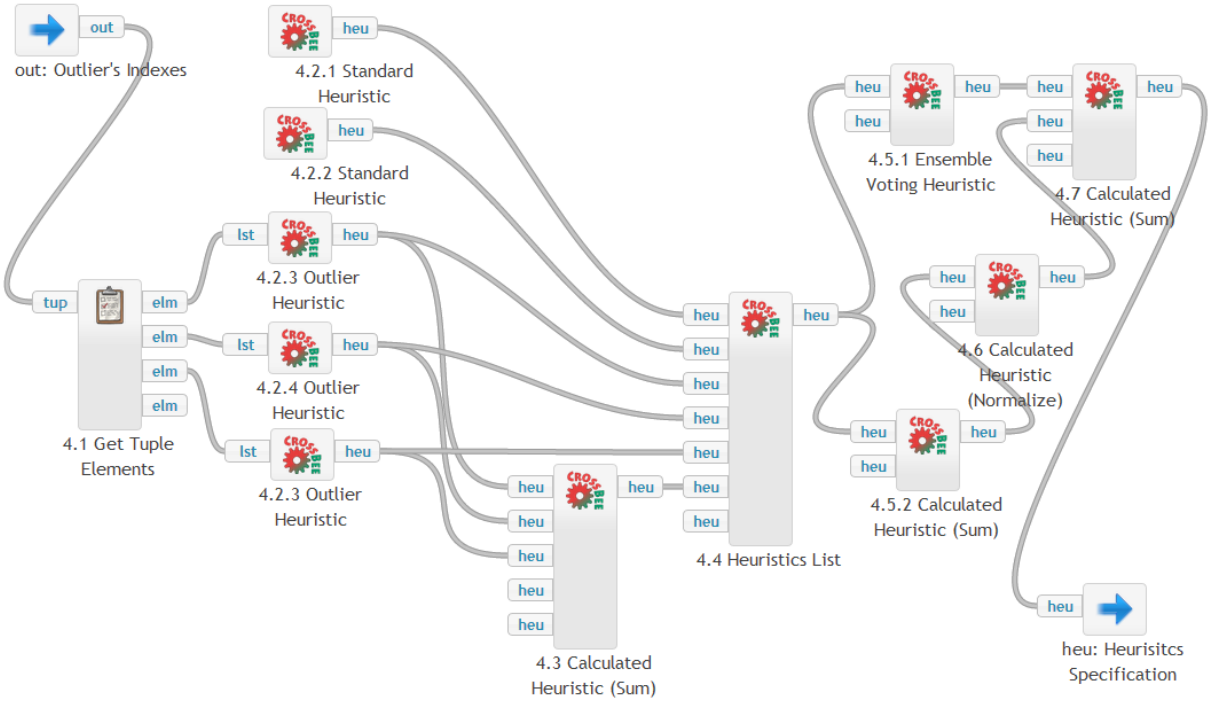



Figure 6.14: *Heuristic specification.*

[3.4] apply classification algorithms in order to retrieve and output document outliers,
  [3.4.1] apply the nearest centroid classifier for outlier document detection,
  [3.4.2] apply the naïve Bayes classifier for outlier document detection,
  [3.4.3] apply the nearest centroid classifier for outlier document detection,

[3.5] pack the results of all three classification-based outlier detection methods into a single transfer object (for the sake of workflow clarity and reusability).

### 6.5.4 Heuristics specification

While the heuristics specification step is the core part of our methodology, this step only specifies which heuristics are selected and how these heuristics should be combined into the ensemble heuristic. The actual calculation is performed later in the heuristic term score calculation step.

The specification displayed in Figure 6.14 is the outcome of our research about the base term heuristics and their combination into the ensemble heuristic presented in Section 6. Which heuristics to use and how to combine them is based on the experiments on the real data that we performed as a part of the research presented in this thesis. The findings resulted in the setting shown in Figure 6.14 for which we believe is a good choice when applied on new data. Nevertheless, the setting and the choice of the base heuristics is fully customizable and can be freely configured to better suit the needs of new applications.

The inputs for this procedure are the lists of detected outliers, which are needed for the outlier-based heuristics specification, while the output is the complex ensemble heuristic, which computes the term bisociation scores. The components in the heuristic specification perform the following tasks:

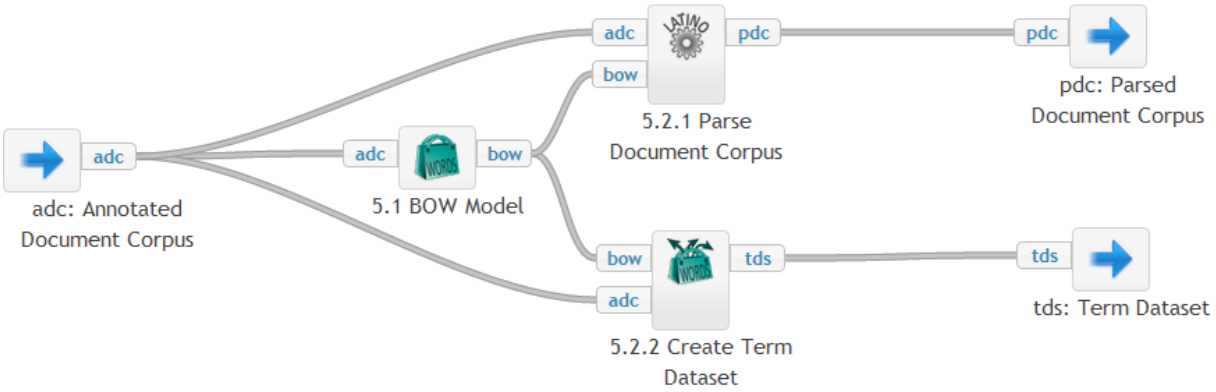



Figure 6.15: *Candidate B-term extraction.*

[4.1] unpack the results of the three classification-based outlier detection methods and get the three lists of outliers (inverse of component [3.5]),

[4.2] define base heuristics (see Section 6.3.2 for details about the base heuristics selection),

- [4.2.1] define TF-IDF-based heuristic *tfidfDomnSum*,
- [4.2.2] define term-frequency-based heuristic *freqRatio*,
- [4.2.3] define outlier-based heuristic *outFreqRelRF*,
- [4.2.4] define outlier-based heuristic *outFreqRelSVM*,
- [4.2.5] define outlier-based heuristic *outFreqRelCS*,

[4.3] define a new outlier-based calculated heuristic *outFreqSum* as a sum of the three included outlier-based heuristics (see Section 6.1.2 for details),

[4.4] create a list of all six base heuristics,

[4.5] combine the six heuristics into a single ensemble heuristic

- [4.5.1] define an ensemble voting heuristic that includes votes of the six heuristics (ensemble voting score, see Equation 4 on page 67)
- [4.5.2] define a calculated heuristic that does the summation of the six heuristics (ensemble position score, see Equation 5 on page 68)

[4.6] define a new calculated heuristic that normalizes the scores to the range between 0 and less than 1,

[4.7] define the final ensemble heuristic by summing the ensemble voting heuristics, which results in the number of terms heuristics' votes in the range from 0 to 6 (integer value), and the calculated normalized sum of heuristics scores in the range from 0 to less than 1 (final ensemble score, see Equation 6 on page 68).

### 6.5.5 Candidate B-term extraction

Another core step of the workflow—candidate B-term extraction—is shown in Figure 6.15. Although it contains only three components, it has a very important and complex goal of transforming the input document corpus into the parsed document corpus as well as the term dataset. The two outputs need to be synchronized in terms of the exact parsing procedure, which is achieved by using the same BoW model that takes care of the parsing. The three components perform the following tasks:

[5.1] construct BoW model for representing documents in the form of feature vectors (see Section 3.2 for more details),

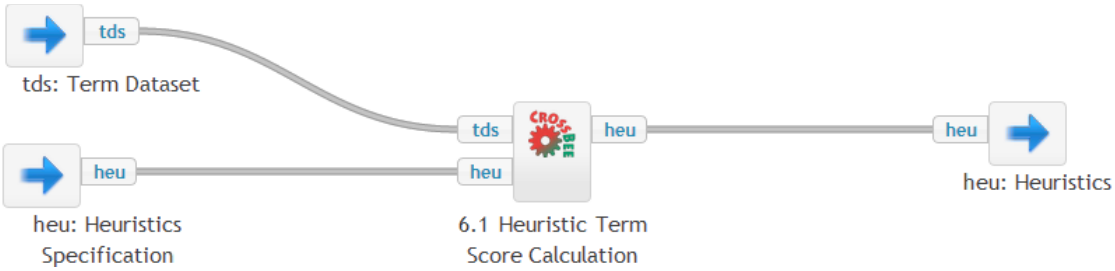



Figure 6.16: *Heuristic term score calculation.*

[5.2] process the input annotated document corpus,
- [5.2.1] parse the documents and capture the information about the exact parsed locations of terms in the text (for the purpose of visualization of documents and the need of highlighting and emphasizing of specific terms),
- [5.2.2] create a list of all terms with additional terms' document vectors, which present vectors of term occurrences in the documents (needed for the advanced heuristic term scores calculation).

### 6.5.6 Heuristic term score calculation

Figure 6.16 shows a structurally simple methodological step of heuristic term score calculation that contains only one component. We could easily skip this encapsulation and use component [6.1] directly at the top level view of the workflow; however we wanted to uniformly present all the steps as well as emphasize this—for the methodology—essential part where the actual heuristic calculation happens.

The inputs to the procedure are the term dataset and the heuristics specification. Based on the information present in the term dataset, the algorithm calculates all the specified heuristics scores for all the terms. The calculation results in the same heuristic structure as defined in Section 6.5.4, however the ensemble heuristic at the top level, as well as all the sub-heuristics, now contain their calculated scores of the terms. The scores of the top-level heuristic are intended to represent terms' bisociation scores and are typically used as a basis for the final term ranking.

### 6.5.7 B-term visualization and exploration

The final part of the workflow, presented in Figure 6.17 deals with the visualization and exploration of the ranked list of B-terms. There are two inputs to this step. The first and the most important is the applied ensemble heuristics, which contain the scores for the extracted candidate B-terms. The second input, the parsed document corpus, is used by the online application for cross-context bisociation exploration CrossBee, which needs the exact information about term extraction from documents, to be able to align the terms back with the original documents in order to visualize them. The goals of the created components are the following:

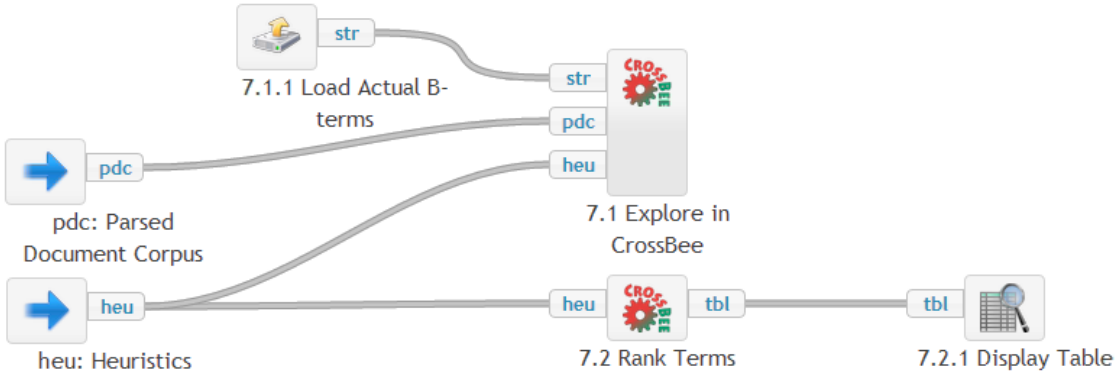



Figure 6.17: *B-term visualization and exploration.*

[7.1] explore the final results in a web application CrossBee, which was designed specifically for the purpose of bisociativity exploration (expressed either through terms or through documents),
  [7.1.1] optional expert specified B-terms may be provided to CrossBee in order to emphasize them in the text and to deliver a feedback about the bisociative quality of the provided ranking,

[7.2] rank the terms
  [7.2.1] display the ranked terms in the form of a table along with their respective scores.

### 6.5.8 Methodology evaluation

The last set of components presented in Figure 6.18 serves the needs of methodology evaluation. There is only one input to the process—one or more evaluated heuristics—which presents the result of all the preceding methodological steps. However, additional information about the actual B-terms is required in order to assess any kind of quality measures. The components perform the following task:

[8.1] select one or more heuristics of interest from the list of multiple heuristics (selected list of heuristics is used to compare performance side by side in all components),

[8.2] load the actual (expert identified) B-terms, which present the gold standard terms used to evaluate the quality of the methodology,

[8.3] calculate ROC curves and the AUC (Area Under Curve) values (see Section 6.4),
  [8.3.1] display the AUC values in a bar chart,
  [8.3.2] display ROC curves graphically,

[8.4] calculate standard IR measures like precision, recall, and the F measure,
  [8.4.1] compare IR measures in the form of a bar chart,
  [8.4.2] display and compare the F values in the advanced VIPER performance evaluation chart component,

[8.5] calculate other ranking performance measures (see Section 6.4.2 for examples),
  [8.5.1] compare ranking performance measures in the form of a chart,
  [8.5.2] compare ranking performance measures in the form of a table.

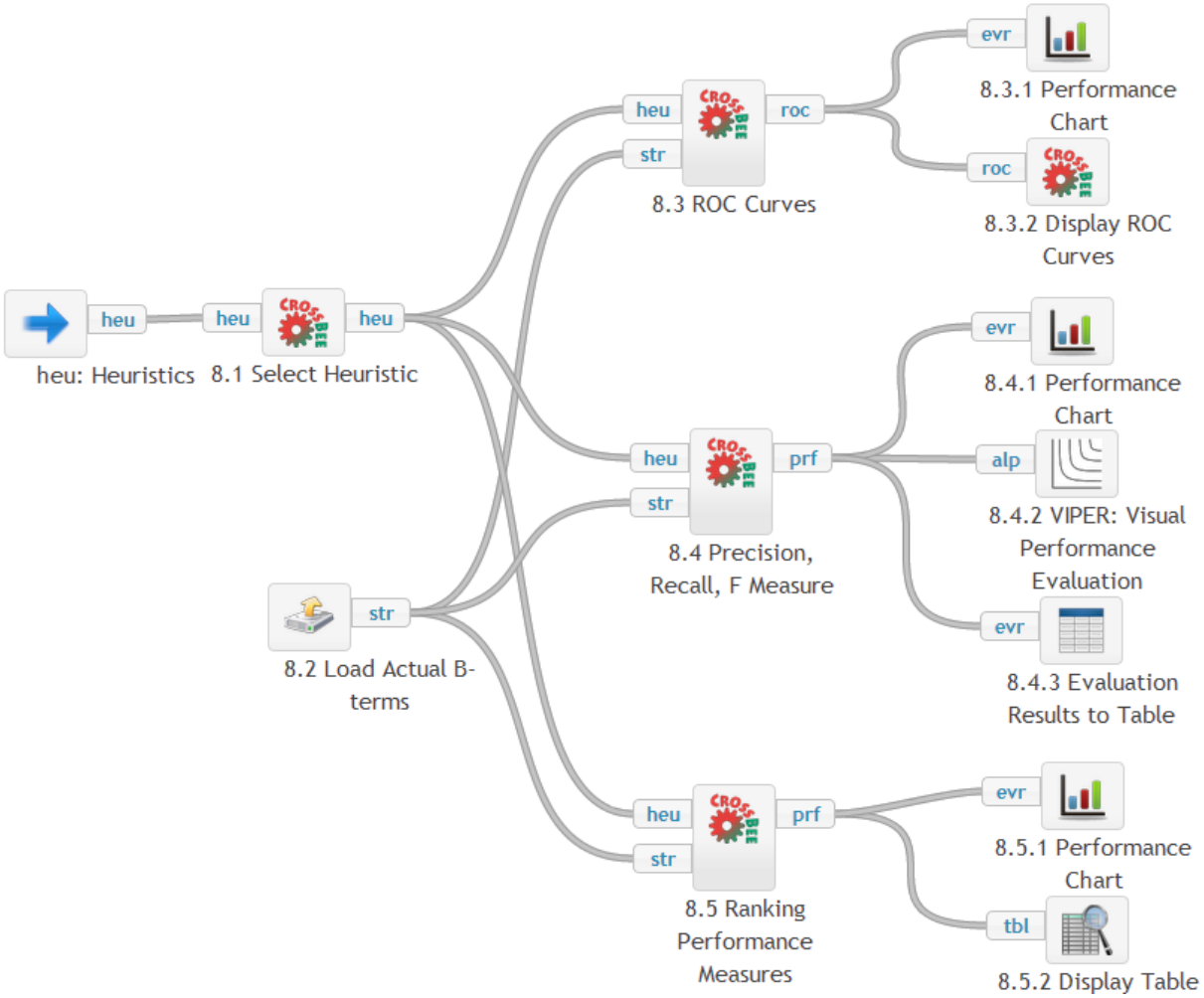



Figure 6.18: *Methodology evaluation.*

The methodology evaluation functionality presented in this section is not part of the actual workflow for cross-domain knowledge discovery; however, it is indispensable when developing such an approach.

With this we conclude the chapter presenting the key part of this thesis—the developed CrossBee methodology, i.e., the mechanism for scoring and ranking the candidate B-terms. The following chapter, among other topics, illustrates how the developed methodology can be further enhanced by connecting it to an advanced online user interface that supports creativity stimulating bisociative exploration.

# 7 Web applications and accessibility

This chapter presents the two web applications developed to support and enhance the methodologies developed in this thesis. Section 7.1 introduces some basic terminology and concepts for describing websites and describes some basic shared characteristics (e.g., design) of the two web applications, Section 7.2 presents the first web application—LemmaGen—which deals with the problem of lemmatization discussed in detail in Chapter 5. Section 7.3 presents the second web application—CrossBee—which implements the methodology and accompanying user interface for bisociation discovery developed in Chapter 6. Section 7.4 concludes by systematically categorizing all implementations relevant to this thesis and states the locations where the developed functionality is available either to download or as a service.

## 7.1 Shared characteristics and terminology

Throughout this chapter as well as the whole thesis we use the term *web page* (or sometimes only *page*) to refer to an online document or resource retrieved by the user in a single request to a specific URL (e.g., http://lemmatise.ijs.si/Software). Furthermore, a *website* is a collection of multiple web pages, usually located under the same domain name (e.g., lemmatise.ijs.si), which contain contextually connected information (e.g., about LemmaGen) linked together by hyperlinks.

When we emphasize that the contents of some website is dynamic we also use the term *web application* (or sometimes only *application*). Dynamic web pages are generated by a *web server* at the time when the user accesses the web page. Conversely, static pages are the documents, which are prepared on a web server in advance (e.g., static HTML documents). Web applications employ dynamically generated contents to provide the user with services based on the user's input, which cannot be statically prepared in advance, e.g., when the user submits a text for lemmatization, a web application needs to respond with a dynamic web page containing the lemmatized user's text.

The two applications—LemmaGen and CrossBee—along with the developed supplementary tool TopicCircle presented in Appendix A are dynamic as described in the paragraph above. Furthermore, excluding description pages in LemmaGen and the applications' *landing pages* (web pages intended to be the starting point of user's website exploration), all the other web pages need to be dynamic since they all perform operations on the user provided data, consequently cannot be static, and thus we refer to them as web applications.

Considering the design of LemmaGen and CrossBee (e.g., see Figure 7.1 and 7.8), we used typical, simple and familiar page layouting principle, which is shared among all the web pages of the two applications. It consists of a header on the top every web page containing the application logo, acknowledgements and a basic navigational menu. Additionally, the left most part of an application contains a detailed hierarchal navigational menu and sometimes also a specific

page dependet menu, e.g., see the download page on the left hand side of Figure 7.1. In many figures we leave out these shared parts of web pages as we want to focus on the specific contents (e.g., see Figure 7.2). Even though the design template is rather classical, it was slightly differentiated in a way to deliver a unique and distinguishing look of the applications.

## 7.2 LemmaGen: Online lemmatization platform

Online user interface supporting the LemmaGen algorithms (Online reference [1]) serves three main purposes: to provide online lemmatization service, to give access to LemmaGen software libraries, and to contain basic information about lemmatization and help about using the libraries. The contents of the first, i.e., the online lemmatization service, are highly dynamic, while the contents of the other two are static. This web application was set up in 2009 and has been actively developed and maintained since. It deserves—along with the published LemmaGen article (Juršič et al., 2010) and a conference paper (Juršič et al., 2007)—many credits for the good reception of the LemmaGen system in the community interested in lemmatization (more concrete evidence for this statement is provided in Section 7.2.4).

Figure 7.1 shows an overview of the LemmaGen web page. For illustrative purposes of this section the text on the figures need not to be readable in all the figures; an interested reader


Figure 7.1: *Overview of the LemmaGen web application.* The right hand side shows the start page of the application (Online reference [1]) with brief information about LemmaGen and lemmatization in general, while the left hand side shows the download page (Online reference [3]) where the user gets detailed information about the libraries and lemmatization models and is provided with the links to download them.

can access the application online through Online reference [1]. The right hand side of Figure 7.1 shows a simple introductory page to lemmatization and LemmaGen in particular, which was intended to be the landing page; however, as observable from our website statistics, popular search engines often point the user directly to the page of his interest (e.g., online lemmatization service) based on the his search query. The left hand side of Figure 7.1 presents a zoomed-out view of the page with downloads and supplementary information—the information about interesting details is provided in the following sections.

Important pages on the LemmaGen website contain rather comprehensive explanation of the topics (e.g., the download page presented on the left side of Figure 7.1). Based on the independent projects that came out of the LemmaGen library without our awareness or help, we are confident to state that the LemmaGen website is sufficiently complete, contains enough information and is self-sufficient for an interested user.

The following sections present the three key aspects of the LemmaGen web application: online lemmatization service, downloadable software and models, and support for the software.

### 7.2.1 Online lemmatization service

The online lemmatization service (Online reference [2]) enables the user to apply the LemmaGen-based lemmatization to the user provided text. This type of browser-based service that

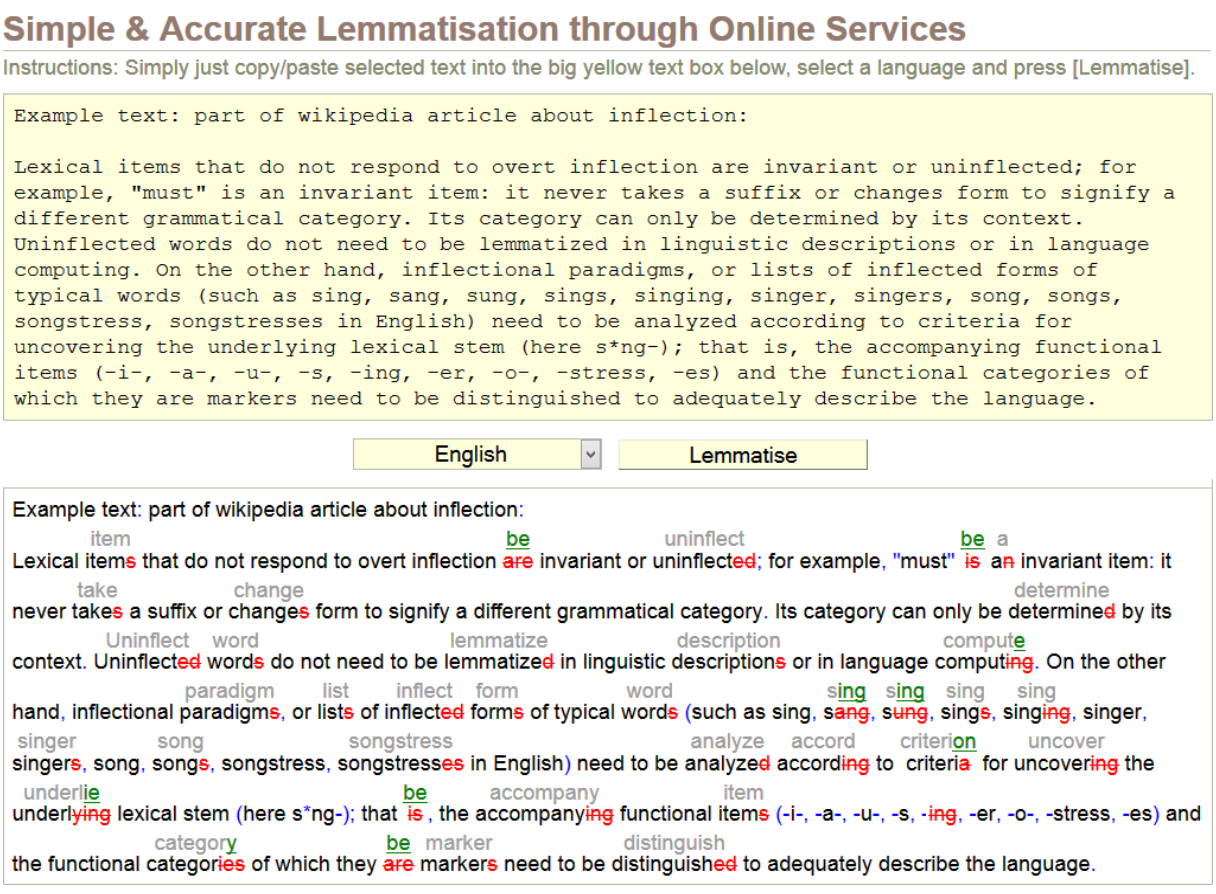



Figure 7.2: *Online lemmatization service demo web page.* The yellow colored elements on the top present the user input and interaction elements, while the text area at the bottom shows the user-friendly formatted output of the service. Note that the blue colored characters present the sections ignored by the tokenizer (e.g., punctuations), whereas the meaning of other colors is obvious from the context.

outputs rich and word-aligned information about lemmas is intended mostly for demo, educational or testing purposes. On the other hand, when the service of lemmatization is to be used in a production environment, we strongly recommend using the provided SOAP web service (Online reference [6]) or using our older and simpler website (Online reference [3]), which works on text snippets as well as on the whole files and outputs clean lemmas without additional lemmatization information and is therefore more appropriate for automatic text processing purposes, where the usage of SOAP web services is not appropriate.

Figure 7.2 shows the default example of lemmatization, available when the user first hits the online lemmatization service web page. The top yellow colored elements present the user modifiable contents; while the lower white colored text area presents the lemmatization result. The user lemmatizes the text by entering or pasting it into the top yellow field, selecting the required language (model for lemmatization) and hitting the "Lemmatise" button (note the website uses UK spelling, thus lemmatise and not lemmatize). The text is sent to the application where it is lemmatized and the user receives the formatted results of lemmatized text in a bottom white field.

The LemmaGen results format was designed specifically for the user's convenience of examining and qualitatively evaluating the results of lemmatization. Observing other similar online services for lemmatization and/or stemming (see Figure 7.3) shows several different approaches for formatting the results:

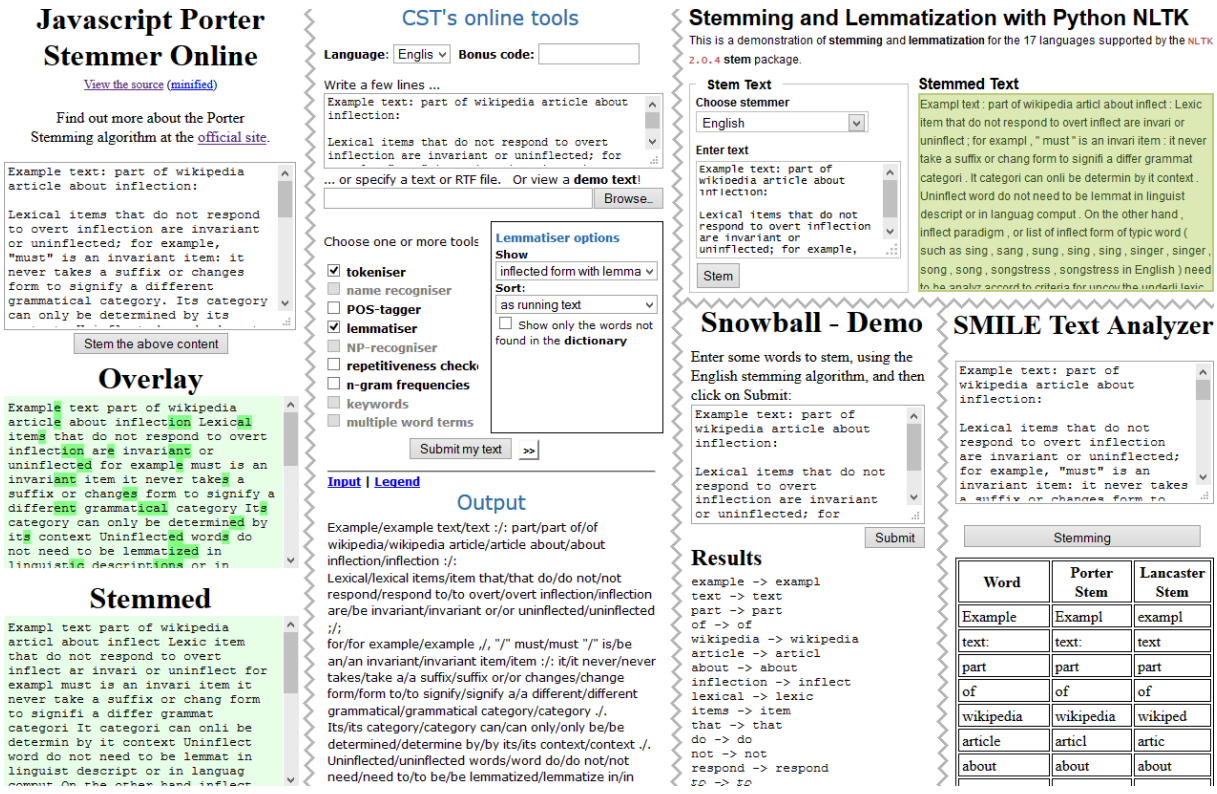



Figure 7.3: *Several top search engine ranked online lemmatization and/or stemming websites.* The web applications are the following (from left to right): Javascript Porter Stemmer Online (Online reference [13]), CST's online tools (Online reference [14]), Stemming and Lemmatization with Python NLTK (Online reference [15]), Snowball-Demo (Online reference [16]), and SMILE Text Analyzer (Online reference [17]). We used the example text from Figure 7.2 to try out all the services.

- Running text with the original wordforms replaced by their stems/lemmas (see Online references [15] and [13]).
- Running text with the original wordforms as well as the lemmas inside the text (see Online reference [14]).
- Running text with marked parts of the wordforms that were stemmed away (see Online reference [13]).
- A table containing the original wordforms and lemmas side by side, while listing all the words vertically as rows of the table (e.g., Online references [16] and [17]).

There are two main issues concerning the presented formats; either the readability or the information about the stemming/lemmatization process is lost. The format of running text with lemmas replacing wordforms is adequate for further processing of such text; however, it is very cumbersome to inspect the results visually, since the user cannot see clearly which changes were introduced by the lemmatizer. In order to extract this information the user needs to read the two texts—the original and the lemmatized—in parallel. On the other hand, the format of a table with wordforms and lemmas enables the user to see the introduced changes clearly, but the readability is lost. Similar holds for the format of running text containing both wordforms and lemmas. The best output format that does not sacrifice the presented issues is the format of running text with marked parts, which are stemmed away; however, this format cannot be applied to lemmatization since the words are not only shortened, but some characters or the whole wordforms may be replaced. Consequently, none of the presented available formats were appropriate for our needs.

Our approach for presenting the result of lemmatization (see bottom part of Figure 7.2) exploits the familiar outlook of a text presented in form of text corrections, which solves both issues mentioned in the previous paragraph. On the one hand we still maintain good readability of the text, while the changes introduced by the lemmatizer are clearly noticeable. Furthermore, the color coding increases the interpretability by coloring the deleted characters with red, the added characters with green, the preserved stems with gray, and the characters ignored by a tokenizer (like punctuation, parentheses and similar) with blue. We claim that such a presentation is superior in terms of the user's examining and evaluating the results, compared to the above described formats used by similar lemmatization and stemming web applications.

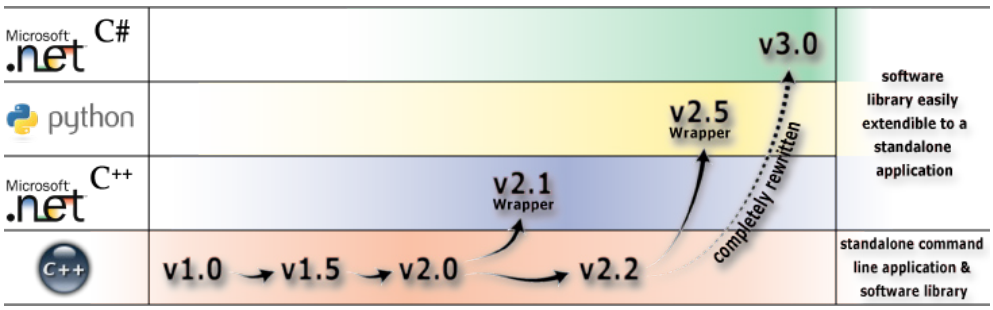



Figure 7.4: *Major versions of LemmaGen.* The software library went through several phases of development, underlying programing languages and types of applications. This figure is taken from the LemmaGen web application (Online reference [3]) and sketches the relationship among the versions of the library.

### 7.2.2 Downloadable libraries and lemmatization models

The LemmaGen software library has a relatively rich development history: it was written in two programming languages and ported to others, it exists in two forms: a software library and a command line application combined with a software library, it was trained on multiple versions of Multext corpora, and it is compiled for multiple operating systems. Figure 7.4 is taken from the download page of the LemmaGen web application and presents some dimensions of complexity arising from developing and supporting many different versions of the library.

In this section we present only a short summary of the produced libraries as the detailed description of them or the development process is beyond the scope of this thesis. Furthermore, we believe it is unnecessary since we provide an extensive description on the web page (Online reference [3]) with downloads, which is shown also on the right hand side of Figure 7.1.

In summary, LemmaGen is implemented in two versions. The first, the older, version is written in C++ programing language and supports not only the usage as a software library but also as a very customizable sets of standalone applications created for all the needs from training the lemmatizer, lemmatization, to testing the accuracies and other supplementary functionalities. The source code of the library comes in a package prepared to be compiled in Windows or Linux operating systems; furthermore, due to the absence of any external library dependencies it is very simple to be compiled also in other systems. The implementation is extremely efficient; however, this version exhibits problems with the transferability of the lemmatization models—they must be learned from the wordform examples in the exactly same text encoding as the lemmatized text will be. This version is not actively developed anymore, but due to the several extensions built on top of it (see Section 7.2.4), we still support and maintain the version by fixing reported bugs.

The second, newer version is written in the C# programing language and contains only the library without any supplementary application built on top of it. Nevertheless, from the implementational perspective this version uses a more modern approach to software development employing strong functionalities and data structures provided by programming frameworks,

| Lexicon Type | Language | Textual Model Tree | Binary Lemmatiser Model | Lexicon Size | |
|---|---|---|---|---|---|
| | | | | Wordforms | Lemmas |
| Multext-East v3 | English | (get) | (get) | 71,784 | 27,467 |
| | Slovene | (get) | (get) | 557,970 | 16,389 |
| | Bulgarian | (get) | (get) | 55,200 | 22,982 |
| | Czech | (get) | (get) | 184,628 | 23,435 |
| | Estonian | (get) | (get) | 135,094 | 46,933 |
| | French | (get) | (get) | 306,795 | 29,446 |
| | Hungarian | (get) | (get) | 64,042 | 28,090 |
| | Romanian | (get) | (get) | 428,194 | 39,359 |
| | Serbian | (get) | (get) | 20,294 | 8,392 |
| Multext | English | (get) | (get) | 66,216 | 22,874 |
| | French | (get) | (get) | 306,795 | 29,446 |
| | German | (get) | (get) | 233,858 | 10,655 |
| | Italian | (get) | (get) | 145,530 | 8,877 |
| | Spanish | (get) | (get) | 510,709 | 13,236 |
| All | All | (get) | (get) | NA | NA |

Figure 7.5: *Multext-East pre-trained models for lemmatization.* Section of the download page providing the LemmaGen lemmatization models trained on the Multext-East version 3 resources.

which result in less code, cleaner code, easier maintainability and extensibility and possibility of improved algorithms. Consequently this library is not only the reimplementation, but contains many enhancement over C++ version like prefix lemmatization rules, better ambiguity resolution heuristics, extending models with MSD information and other enhancements, which primarily increases the accuracy and speed or decrease the size of trained models. The library enables rapid programing and in only few lines of code the user can implement the application he needs—which is the reason why the supplementary lemmatization application in not provided along with the library as in the C++ case. Although the execution environment is slower (C# compiles into intermediate byte-code whose execution is not as efficient as the execution of the C++ natively compiled code) we were able to improve the design of algorithms ensuring a comparable overall execution performance.

The website provides also the pre-trained lemmatizer models for the two presented main versions of the library. The datasets used for training the models are based on the Multext East resources (Online reference [18]).

The C++ LemmaGen is trained upon version 3 of the Multext-East resources (Online reference [19]), which contain resources for 12 languages (see Figure 7.5). Each model file is separately downloadable and the user can include and use only the needed models, which results in a flexible modular design; however, this can represent too much overhead for an inexperienced user.

The C# LemmaGen uses the latest, version 4 resources (Online reference [20]), which contain 16 languages for training the lemmatizers. Combining these with the standard Multext resources we built the lemmatization models for the following 18 languages: Bulgarian, Czech, English, Estonian, French, Hungarian, Macedonian, Polish, Persian Romanian, Russian, Slovak, Slovene, Serbian, Ukrainian, German, Italian, and Spanish. The distribution of these models is different compared to the C++ library since we packed the compressed models directly into the library. The advantage is that the user needs no additional actions and can start using the library right away; on the other hand, the downside is the increased size of the library. However, the user can exclude the models not required by his solution and in this way reduce the size.

More detailed and technically oriented information can be found on the web page (Online reference [3]).

### 7.2.3 LemmaGen help and support

Besides the online lemmatization service and the downloadable contents, the third and the last key part of the LemmaGen web application is to provide help and support to the users interested in lemmatization. This is achieved through several web pages, which contain:

- detailed descriptions of the libraries,
- code examples on how to use the libraries from the developer's perspective,
- downloadable publications regarding the theoretical background,
- contact information.

The first page on the list above was described in detail in the previous section and is available online as seen on the left hand side of Figure 7.1 (Online reference [3]). The third and the fourth pages are rather simple and available as separate pages on the LemmaGen website (Online references [9] and [10]). What is left for this section is a short presentation of the second item, i.e., the examples of using the code libraries. These examples (see Figure 7.6) are provided for the new C# library and contain a complete, step-by-step description of creating an application that starts by downloading the library, contains the complete source code and ends with a working application. We believe such tutorials are extremely valuable to new LemmaGen users and this assumption was already confirmed by some of them through email conversation.

### 7.2.4 Dissemination and impact

This section shows that LemmaGen has been well accepted by the users. We provide some statistics for our website and show several project developed on top of LemmaGen. We are aware that this presentation does not provide a quantitative evaluation measure of success due to the lack of a reference point; however, we believe it provides sufficient evidence about LemmaGen's potential compared to other existing lemmatizers.

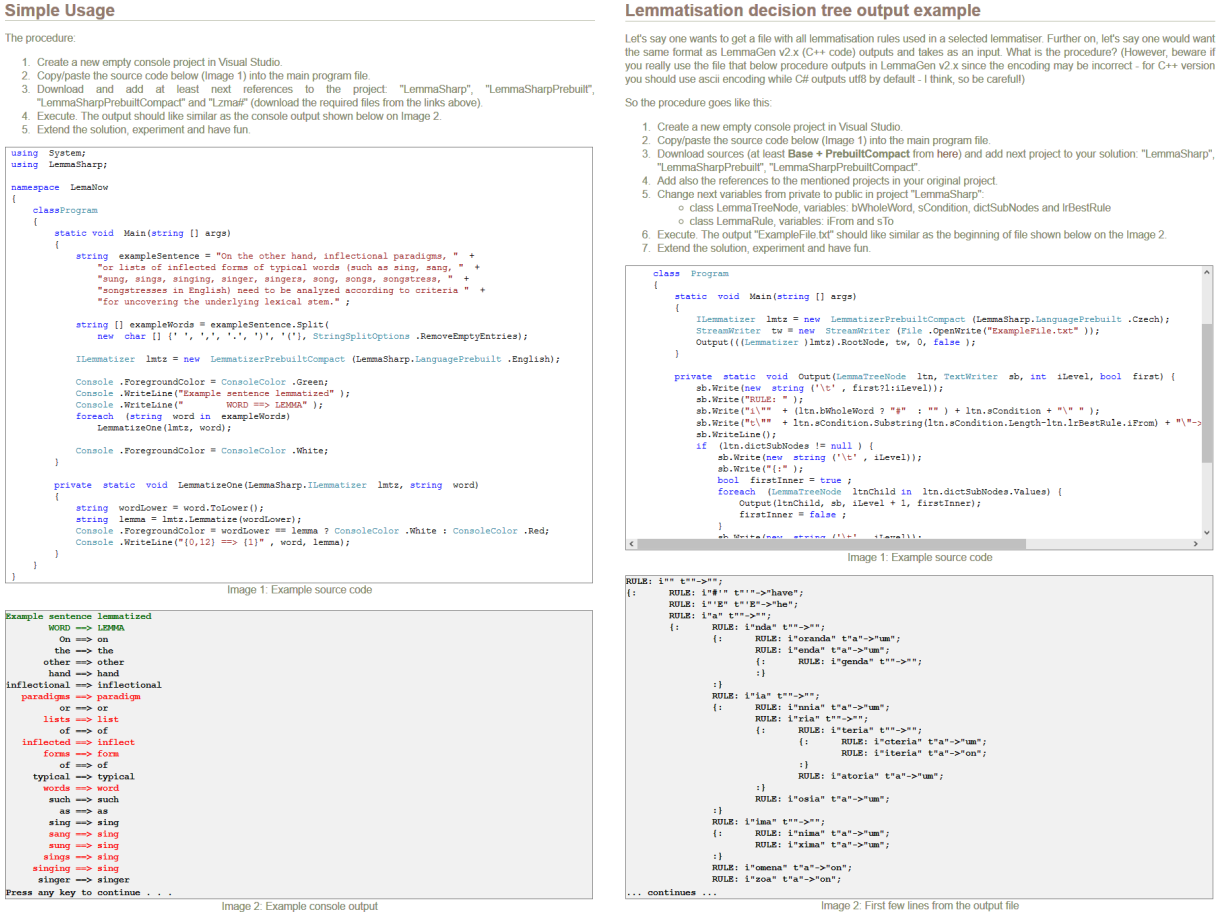



Figure 7.6: *Two examples of web pages providing help on using the LemmaGen library.* The example on the left (Online reference [7]) contains the description and all the required code for a working application that lemmatizes a given text, while the example on the right (Online reference [8]) provides the code for outputting the model of a trained lemmatizer.

The chart in Figure 7.7 presents two measurements from the LemmaGen website traffic logs, i.e., the number of page-views and downloads per month for the period from the start of July 2010 to the end of April 2013. In this period LemmaGen received around 19,000 page-views and 1,500 downloads. Among the page-views, 45% are attributed to the page about the software (Section 7.2.2), 24% to the online lemmatization service page (Section 7.2.1), 17% to the other help pages (Section 7.2.3) and 14% to the main overall landing page. Similarly, analysis of downloads shows that about 10% of downloads are of publications, while the rest is nearly evenly split between downloads of C# LemmaGen library and C++ library along with the lemmatization models. When observing page-views and downloads together we see a strong correlation, which is expected, however the ratio between them is 10:1, which we consider a very positive indication since one out of ten users also downloaded software.

The chart in Figure 7.7 shows the data over time and we are pleased to see a slowly but continuously increasing interest in our research. In average over the whole period there were about 600 page-views per month; however, lately the number of page-views is increasing fast and has risen above 1,000 in 2013. To put these numbers into perspective we used two Google services: Google Trends and Google AdWords (Online reference [45] and [46] respectively)—company Google is currently (in 2013) the reference point in web search engines statistics as it handles by far the largest percent of web queries. Google Trends service is given the keyword of interest and returns a relative index of interest over time, which is calculated on the basis of searches performed by this keyword. Google AdWords, on the other hand, provides an absolute number of global monthly searches that are averaged in the last 12 months. Using these two metrics we can further analyze the data.

Google Trend's provided index is displayed in Figure 7.7 as a black dotted line titled global interest in lemmatization. Though it does not align to any axis (due to its relative nature) it

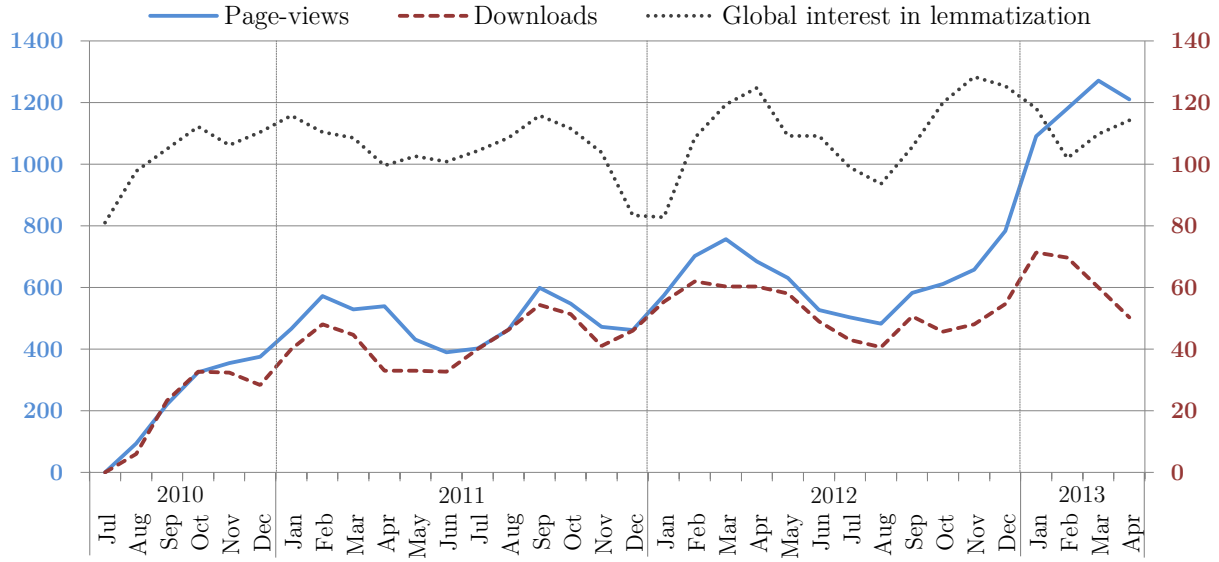



Figure 7.7: *LemmaGen website page-views and downloads logs.* The chart presents the website traffic logs from the start of July 2010 to end of April 2013. The solid blue line aligns with the left vertical axis and represents the number of page-views per month. The dashed red line aligns with the right vertical axis and represents the number of downloads per month. The dotted black line does not align to any axes as it represents a relative global interest in lemmatization over time. All three curves are smoothed with simple moving average over a period of 3 months.

exhibits strong correlation with the two curves retrieved from our logs. This provides us informal evidence that the comparison is correct and that indeed the majority of the users come to our page from the Google search engine—our logs also show this fact. Observing the curves in Figure 7.7 provides us with an indication that the interest in our research in lemmatization is gaining relatively to the whole area of lemmatization.

Google AdWords provides us with the number of global monthly searches of "lemmatization" (and contextually similar terms), which results in 2,900. Approximately including the information about Google market share produces the value 4,400. Given the latest page-view values for LemmaGen, which are in the range around 1,000, and given that an average LemmaGen user visits 4.75 pages per visit, we conclude that LemmaGen receives around 5% of global search traffic connected to lemmatization. However, note that these are relatively rough values and we provide them only as estimations of the impact and not as hard proofs of any kind.

Another possibility of evaluating success of a research project like LemmaGen, which resulted in a practical tool and library, can be investigated by inspecting how many projects utilize the developed library. We searched the web (January 2013) and found nine projects using the LemmaGen library, out of these, two are directly affiliated with us, we were aware about the other two, but we were completely unaware of the additional five. Furthermore, these are obviously only the publicly accessible projects and since we licensed the library under the very liberal license LGPL, we suspect many more exist, which are not published on the web and therefore not searchable. The following list summarizes the projects utilizing the facilities of LemmaGen:

- Obeliks: part-of-speech tagger for Slovene language (Online reference [21]),
- LATINO: link analysis and text mining toolbox (Online reference [33]),
- Orange: data mining toolbox (Online reference [22]),
- Lemmagen-Python-Extension: Python project to expose lemmatization functions of LemmaGen (Online reference [23]),
- Lemmagen4J: LemmaGen rewritten from C# to Java (Online reference [24]),
- slovene_lemmatizer: project about creating Slovene lemmatizer based on C++ version of LemmaGen (Online reference [25]),
- Text-processing-bundle: A bundle containing code related to processing of unstructured sources (Online reference [26]),
- Sparc.TagCloud: a TagCloud library for .NET(Online reference [27]),
- Taverna workflow based on LemmaGen (Online reference [28]).

The most inspiring in the above list are the two projects with the essential purpose of making LemmaGen available in other programming languages, namely, Lemmagen-Python-Extension (Online reference [23]) and Lemmagen4J (Online reference [24]). This provides us with additional indication that the research regarding LemmaGen, which was done in this thesis, is a valuable contribution in the area of stemming and lemmatization. We conclude by stating two citations from the websites using LemmaGen: "extremely efficient, pretty good algorithms used, easily portable via platform invokes/c-library, and completely written in native C++ without using any external libraries" (Online reference [23]), "our library uses the wonderful LemmaGen library to lemmatize the analysed text" (Online reference [27]).

To summarize, the LemmaGen web application is designed to be self-sufficient reference for the underlying LemmaGen library in terms of providing the library downloads, documentation, code tutorials and intuitive online lemmatization service that can be used either for educational of testing purposes. Furthermore, SOAP web services are provided for the needs of production environment lemmatization. We are pleased to see the evidences presented in this section, which indicate that LemmaGen was well accepted by the users dealing with the problem of lemmatization.

## 7.3 CrossBee bisociative literature mining platform

The web application CrossBee is the default user interface to the CrossBee methodology presented in this thesis and an off-the-shelf solution for finding bisociations bridging two domains. As opposed to LemmaGen the CrossBee web application does not offer a downloadable library and documentation distribution or extensive help. From this perspective CrossBee provides the functionality analogous to the LemmaGen's online lemmatization; however in the CrossBee case, the methodology is more complex and requires a collection several web pages and views and not just one simple page as in the LemmaGen case.

Our aim was to create a system that not only suggests bridging terms using the ensemble ranking methodology presented in this thesis but also helps the experts when searching for hidden links that connect two seemingly unrelated domains. To this core we have added supplementary functionalities and content presentations, which make the CrossBee website a user-friendly tool for ranking and exploration of prospective cross-context links. This enables the user not only to spot but also to efficiently investigate discovered cross-domain links.

The CrossBee web application surpasses other types of interfaces particularly in the user-friendliness perspective. The website is designed by considering final users who are not necessary computer scientists or data miners and who prefer fixed sequence of steps, which directly follow the proven methodology. However, the weakness of this approach is the lack of possibility to experiment with different settings as well as the lack of possibility to extend the methodology with new ideas and then compare or evaluate these derived approaches among themselves.

This section presents CrossBee by describing its most important functionality and a typical use case example. The CrossBee website is built on top of the CrossBee library created for the needs of our previous work (Juršič et al., 2012a, 2012b). From this perspective CrossBee is firstly, a functional enhancement and secondly, a wrapping of this functionality into a practical web user interface especially designed for the requirements of bisociation discovery (Online reference [29]). Note that future versions of CrossBee might not be visually identical as the current version presented here. Nevertheless, the core ensemble ranking algorithm, the two datasets and the results presented in this work will remain available in the future by providing a link to the application, data and settings compatible with this thesis in order to ensure the repeatability of the experiments.

### 7.3.1 A typical use case

The most standard use case—as envisioned by CrossBee authors—is the following:

- Prior to starting the process of bisociation exploration, the user needs to prepare the input file of documents. The prescribed format of the input file is kept simple to enable all

users, regardless of their computing skills, to prepare the file of documents of their interest. Each line of the file contains exactly three tab-separated entries: (a) the document identification number, (b) the domain acronym, and (c) the document text. The user starts at the initialization page (see Figure 7.8), which is the main entry point to the system. The minimal required user's input at this point is a file of documents from two domains. The other options available to the user at this point include specifying the exact preprocessing options, specifying the base heuristics to be used in the ensemble, specifying outlier documents identified by an external outlier detection software, (e.g., Sluban et al., 2012), defining the already known B-terms, and others. When the user selects all the desired options he proceeds to the next step.

- CrossBee starts a computationally very intensive step in which it prepares all the data needed for the fast subsequent exploration phase. During this step the actual text preprocessing, base heuristics, ensemble, bisociation scores and rankings are computed in the way presented in Section 6.3. This step does not require any user's intervention.
- After computation, the user is presented with a ranked list of B-term candidates as seen in Figure 7.9. The list provides the user some additional information including the ensemble's individual base heuristics votes (columns 7-12) and term's domain occurrence statistics in both domains (columns 5 and 6). If the user defines the actual B-terms during the initialization (which is not a realistic scenario when exploring a new domain for the first time) then these B-terms are marked throughout the whole CrossBee session, as seen in Figure 7.9 and Figure 7.10. The user then browses through the list and chooses the term he believes to be promising for finding meaningful connections between domains.
- At this point, the user inspects the actual appearances of the selected term in both do-

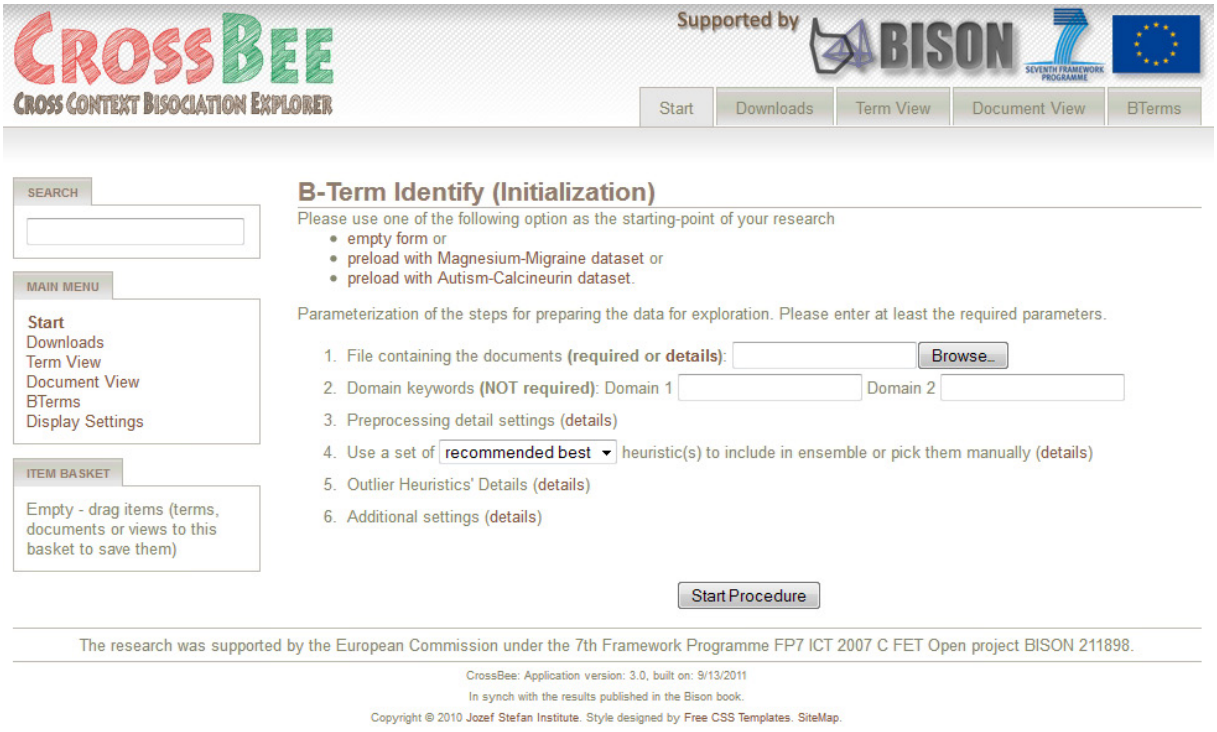
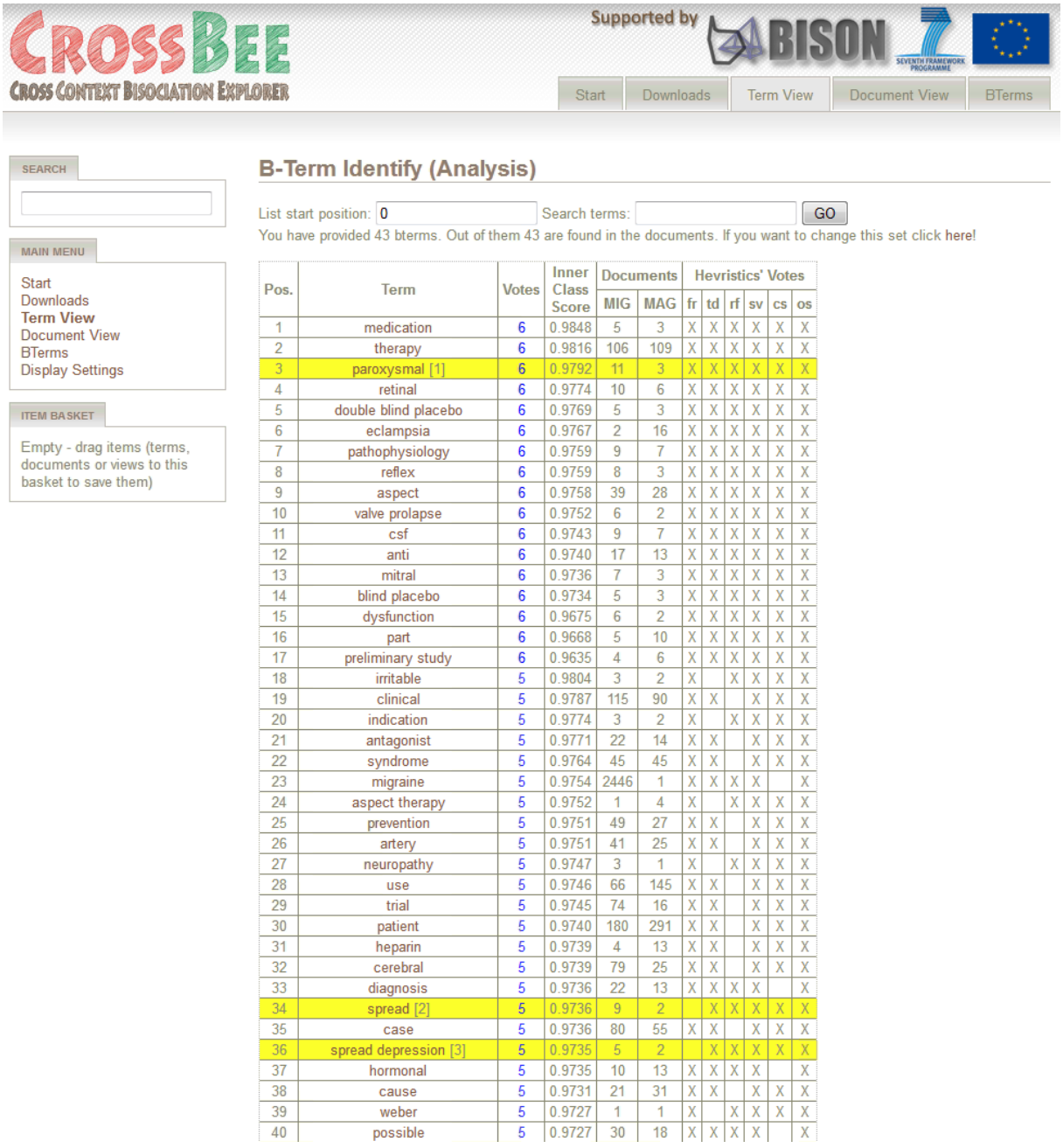



Figure 7.8: *Home page of the CrossBee system.* The user starts an exploration at this point by inputting documents of interest and by tuning the parameters of the system.

mains, using the side-by-side document inspection as shown in Figure 7.10. In this way, he can verify whether his rationale behind selecting this term as a bridging term can be justified based on the contents of the inspected documents.

- Afterwards, the user continues with the exploration by returning to step 3 or by choosing another term in step 4, or concludes the session.

The most important result of the exploration procedure is a proof for a chosen term to be an actual bridge between the two domains, based on supporting facts from the documents. As experienced in sessions with experts, the identified documents are an important result as well,




| Pos. | Term | Votes | Inner Class Score | Documents | | Hevristics' Votes | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIG | MAG | fr | td | rf | sv | cs | os |
| 1 | medication | 6 | 0.9848 | 5 | 3 | X | X | X | X | X | X |
| 2 | therapy | 6 | 0.9816 | 106 | 109 | X | X | X | X | X | X |
| 3 | paroxysmal [1] | 6 | 0.9792 | 11 | 3 | X | X | X | X | X | X |
| 4 | retinal | 6 | 0.9774 | 10 | 6 | X | X | X | X | X | X |
| 5 | double blind placebo | 6 | 0.9769 | 5 | 3 | X | X | X | X | X | X |
| 6 | eclampsia | 6 | 0.9767 | 2 | 16 | X | X | X | X | X | X |
| 7 | pathophysiology | 6 | 0.9759 | 9 | 7 | X | X | X | X | X | X |
| 8 | reflex | 6 | 0.9759 | 8 | 3 | X | X | X | X | X | X |
| 9 | aspect | 6 | 0.9758 | 39 | 28 | X | X | X | X | X | X |
| 10 | valve prolapse | 6 | 0.9752 | 6 | 2 | X | X | X | X | X | X |
| 11 | csf | 6 | 0.9743 | 9 | 7 | X | X | X | X | X | X |
| 12 | anti | 6 | 0.9740 | 17 | 13 | X | X | X | X | X | X |
| 13 | mitral | 6 | 0.9736 | 7 | 3 | X | X | X | X | X | X |
| 14 | blind placebo | 6 | 0.9734 | 5 | 3 | X | X | X | X | X | X |
| 15 | dysfunction | 6 | 0.9675 | 6 | 2 | X | X | X | X | X | X |
| 16 | part | 6 | 0.9668 | 5 | 10 | X | X | X | X | X | X |
| 17 | preliminary study | 6 | 0.9635 | 4 | 6 | X | X | X | X | X | X |
| 18 | irritable | 5 | 0.9804 | 3 | 2 | X | | X | X | X | X |
| 19 | clinical | 5 | 0.9787 | 115 | 90 | X | X | | X | X | X |
| 20 | indication | 5 | 0.9774 | 3 | 2 | X | | X | X | X | X |
| 21 | antagonist | 5 | 0.9771 | 22 | 14 | X | X | | X | X | X |
| 22 | syndrome | 5 | 0.9764 | 45 | 45 | X | X | | X | X | X |
| 23 | migraine | 5 | 0.9754 | 2446 | 1 | X | X | X | X | | X |
| 24 | aspect therapy | 5 | 0.9752 | 1 | 4 | X | | X | X | X | X |
| 25 | prevention | 5 | 0.9751 | 49 | 27 | X | X | | X | X | X |
| 26 | artery | 5 | 0.9751 | 41 | 25 | X | X | | X | X | X |
| 27 | neuropathy | 5 | 0.9747 | 3 | 1 | X | | X | X | X | X |
| 28 | use | 5 | 0.9746 | 66 | 145 | X | X | | X | X | X |
| 29 | trial | 5 | 0.9745 | 74 | 16 | X | X | | X | X | X |
| 30 | patient | 5 | 0.9740 | 180 | 291 | X | X | | X | X | X |
| 31 | heparin | 5 | 0.9739 | 4 | 13 | X | X | | X | X | X |
| 32 | cerebral | 5 | 0.9739 | 79 | 25 | X | X | | X | X | X |
| 33 | diagnosis | 5 | 0.9736 | 22 | 13 | X | X | X | X | | X |
| 34 | spread [2] | 5 | 0.9736 | 9 | 2 | | X | X | X | X | X |
| 35 | case | 5 | 0.9736 | 80 | 55 | X | X | | X | X | X |
| 36 | spread depression [3] | 5 | 0.9735 | 5 | 2 | | X | X | X | X | X |
| 37 | hormonal | 5 | 0.9735 | 10 | 13 | X | X | X | X | | X |
| 38 | cause | 5 | 0.9731 | 21 | 31 | X | X | | X | X | X |
| 39 | weber | 5 | 0.9727 | 1 | 1 | X | | X | X | X | X |
| 40 | possible | 5 | 0.9727 | 30 | 18 | X | X | X | X | | X |

Figure 7.9: *Candidate B-term ranking page.* As displayed by CrossBee after the preprocessing is done. This example shows CrossBee's term list output as ranked by the ensemble heuristic in the migraine-magnesium dataset. The terms marked yellow are the Swanson's B-terms.

as they usually turn out to be a valuable source of information providing a deeper insight into the discovered cross-domain relations.

### 7.3.2 Other CrossBee functionalities

Below we list the most important additional functionalities of the CrossBee system:

- *Document focused exploration* empowers the user to filter and order the documents by various criteria. The user can find it more pleasing to start exploring the domains by reading documents and not browsing through the term lists. The ensemble ranking can be used to propose the user which documents to read by suggesting those with the highest proportion of highly ranked terms.
- *Detailed document view* provides a more detailed presentation of a single document in-

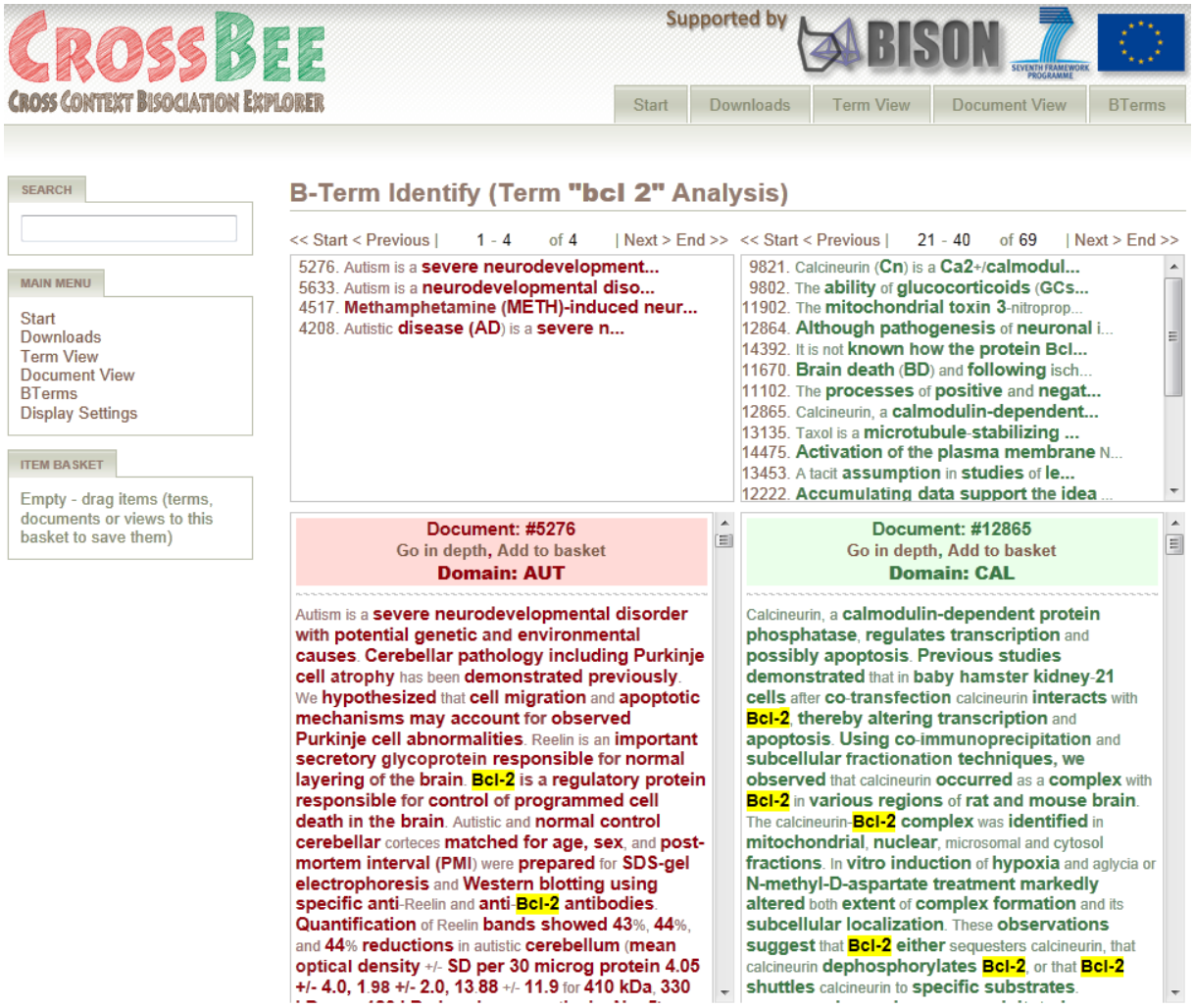



Figure 7.10: *Side-by-side document inspection of potential cross-domain links.* CrossBee supports the inspection of potential cross-domain links by a side-by-side view of documents from the two domains under investigation. The figure presents an example from the autism-calcineurin dataset, showing the analysis of the Bcl-2 term. The presented view enables efficient comparison of documents, the left one from the autism and the right one from the calcineurin domain. The actual displayed documents were reported by Macedoni-Lukšič et al. (2011) as relevant for exploring the relationship between autism and calcineurin.

cluding various term statistics and a similarity graph showing the similarity between this document and other documents from the dataset.

- *Methodology performance analysis* supports the evaluation of the methodology by providing various data, which can be used to measure the quality of the results, e.g., data for plotting the ROC curves.
- *High-ranked term emphasis* marks the terms according to their bisociation score calculated by the ensemble heuristic. When using this feature all high-ranked terms are emphasized throughout the whole application making them easier to spot (note different font sizes in Figure 7.10).
- *B-term emphasis* marks the terms defined as B-terms by the user (note yellow terms in Figure 7.10).
- *Domain separation* is a simple but effective option, which colors all the documents from the same domain with the same color, making an obvious distinction between the documents from the two domains (note different colors in Figure 7.10).
- *User interface (UI) customization* enables the user to decrease or increase the intensity of the following features: high-ranked term emphasis, B-term emphasis and domain separation. In cooperation with the experts, we discovered that some of them do like the emphasizing features while the others do not. Therefore, we introduced the UI customization where everybody can set the intensity of these features by their preferences.

### 7.3.3 CrossBee computational creativity assessment

Since our work originates from Koestler's definition of creative process it naturally satisfies his notion of creativity. However, the concepts of creativity and computational creativity have several definitions as presented in Section 2.2. We argue that our approach can be labeled creative according to at least two of them, introduced by Boden (2004) and Wiggins (2006).

As we described in Section 2.3.2, Boden (2004) defines creativity as "the ability to come up with ideas or artifacts that are new, surprising and valuable." Considering this definition, and given that the main output of our methodology is a ranked list of potentially interesting bridging terms/concepts, we argue that—although we do not produce new concepts—the ranking of potentially interesting bridging concepts itself may represent new, surprising and valuable ideas/artifacts. The proposed approach produces new term rankings, because to the best of our knowledge there are no similar methodologies available. The results are often also surprising, both in terms of unlikeliness (as not commonly used terms may appear at the top of the ranked list) as well as in terms of subjective surprise (as noted by observing the experts using our system). The weakest claim we provide is the notion of value of the system as until now the developed approach did not produce a scientific breakthroughs; however, it triggered novel insights by experts who tested our system. Therefore, we conclude that using Boden's definition, the level of our systems creativity is limited by the value of its results and only the time and the number of users will show how valuable the system is and how valuable its results really are.

Considering computational creativity, Wiggins (2006) proposes the following definition for which he states to be commonly adapted by the AI community: computational creativity refers to the "performance of tasks (by a computer), which, if performed by a human, would be deemed creative." We argue that, although the ranking problem we solve is not something

people usually do, our system can be considered creative according to this definition. Take an analogy with online search engines whose task is finding documents and ranking the search results. We believe that, if such rankings were performed by a human, this could be considered as a very creative process and the person would need to possess enormous knowledge to be able to do this. The final results of our methodology—the insights, which might arise from using our system—could also be considered scientifically creative, even though the ultimate creative act will be performed by the experts using the system and not the system alone. We designed the methodology in a way to enable the expert to be more productive when generating such creative ideas. Therefore, we argue that this added effectiveness of the expert's creativity process originates from the system and its underlying methodology. Hence we believe our system possesses some elements of computational creativity proposed by Wiggins.

The last part of the previous paragraph already demonstrates yet another type of creativity, which is defined by the human-computer interaction (HCI) community dealing with creativity support tools (CST, see Section 2.3.3). As stated by Resnick et al. (2005) the goal in CST is to develop improved software and user interfaces that empower users to be not only more productive, but more innovative. This is exactly the characteristic envisioned at the conception of the CrossBee methodology (to leave the web application aside for a moment). We provide some evidence of our methodology enabling the user to be more innovative already in Section 6.4. We show that our methodology promotes B-terms that should—when observed by the expert—trigger and stimulate the user's ability to make new bisociative links between two domains and consequently, boost the user's creativity. We argue that being creative and being innovative are, for the purpose of this discussion very similar ideas and therefore conclude that our system is considered a creativity supporting tool by Resnick's definition.

A more structured set of design principles for CST provided by Shneiderman (2007, 2009) is described in Section 2.3.3. Shneiderman states that is fundamental for CST to enable collaboration, provide rich history-keeping, design with low thresholds (steep learning curve for novices), high ceilings (sophisticated functionality that experts need), and wide walls (wide range of supplementary services). Current limitation in the CrossBee web application is the rich history-keeping aspect as it is completely unaddressed. Next in line of weakly supported principle is enabling collaboration since CrossBee does not yet enable native online expert collaboration; however, using the tool in pairs of users has been observed to have beneficial effects. Next, high ceilings principle is covered by the elaborated underlying CrossBee methodology suggesting B-terms, which provide sophisticated functionality that experts need. Furthermore, low thresholds are partially addressed by providing a common, well known type of interface—website—so that the user can focus on the problem and not on the interface itself. Additionally, wide walls principle can be considered to be addressed with many possible views on the data that application provides along with supplementary tools like TopicCircle (see Appendix A). Finally, the principle currently the most enforced is certainly the exploration as the efficient navigation among different views of the data is application's base task. In summary, even though the above argument statements are rather loose, we believe that the CrossBee web application shows initial CST orientation (as defined by Shneiderman) and at the same time we acknowledge that there is plenty room for further improvement.

Table 7.1: *Comparison of the implemented interfaces.*

| Property Category | Interface Property | Websites | | Workflow widgets | Webservices | Code libraries | |
|---|---|---|---|---|---|---|---|
| | | CrossBee | LemmaGen | | | CrossBee | LemmaGen |
| Usage | Ease of use | •••• | ••••• | ••• | •• | • | •• |
| | Documentation | •••• | •••• | •••• | ••• | • | •• |
| | Technical knowledge needed | • | • | •• | •••• | ••••• | ••••• |
| Functionality | Flexibility / Configurability | •• | • | •••• | •••• | ••••• | ••••• |
| | Functionality included | ••• | • | •••• | ••• | •••• | •••• |
| | Methodology modification | • | • | •••• | •••• | ••••• | •• |
| | Evaluation support | ••• | • | •••• | •••• | ••••• | ••••• |
| Execution efficiency | Speed | ••• | •• | •• | • | •••• | ••••• |
| | Appropriate for large scale | • | • | •• | • | ••• | ••••• |

## 7.4 Software and services accessibility

This section deals with interfaces to the methodology implementation, i.e., the user and programming interfaces. Let us first introduce the terminology used in this section. We use the word *implementation* to refer to the concrete source code consisting of lines, functions, objects, and other code elements, which deliver the core functionality needed for the methodology execution on real data. On the other hand, the term *interface* is a higher level concept that refers to a point of interaction between the implemented components. Each interface provides one self-sufficient, predefined, methodology specific functionality that is needed to execute the designed process.

We provide four types of interfaces, which differ in their complexity, configurability, ease of use, target users and other parameters that define their applicability in standard usage scenarios. The four types of interfaces are:

- web application,
- interactive workflow and set of workflow components in the ClowdFlows environment,
- set of web services, and
- open source code library.

In addition to these types, we split the provided interfaces also by functionality, as we want to make available not only the CrossBee methodology but the LemmaGen lemmatization engine. Accordingly, we developed the following interfaces enabling the user to access the functionality developed in this thesis:

- CrossBee website (Online reference [29]),
- LemmaGen website (Online reference [1]),
- LemmaGen and CrossBee workflows and workflow components (Online reference [35]),
- SOAP web service equivalents of workflow components (Online references [6] and [30]),
- LemmaGen code library (Online reference [3] and [5]),
- CrossBee code library (Online reference [31]).

The actual interfaces are compared side by side in Table 7.1 where we list some of interface properties and assign qualitative scores 1-5 based on our subjective estimation of how well each

property is represented in each interface. Table 7.1 should be considered informational only, as for instance, the user who has extensive experience in using SOAP web services will probably find this interface easier to use than some other. However, the information presented in this table can still be very useful for a potential user to make a preliminary decision on which interface to choose depending on his needs.

Even though the provided interfaces are very diverse it should be noted that the underlying implementation is the same in all of them. Consequently, the results retrieved from any interface should be the same if we execute matching functionalities with identical input and same parameter settings.

With this presentation of created/provided interfaces to our methodology we conclude the chapter about web applications and accessibility. The following last chapter concludes this thesis by summarizing the main contributions and provides discussion about open questions and suggestions for further work.

# 8 Conclusions and further work

The main goal of this thesis was to propose a solution to the problem of bridging previously separate domains of interest, and suggesting means for finding promising pathways leading to new discoveries. To this end, we developed a methodology and a system for bisociative literature mining, focusing on ensemble-based bridging term identification and ranking. In this chapter, we first summarize the main contributions of this thesis followed by an extensive discussion about the state of presented research and many ideas for further work.

## 8.1 Summary of contributions

In the research presented in this thesis, we addressed the problem of using text mining for cross-domain knowledge discovery. The starting point of this thesis was an in-depth literature survey on topics tightly connected to text mining for cross-domain knowledge discovery, ranging from creativity research, bisociations and bisociative knowledge discovery, text mining, literature-based discovery, computational creativity, to creativity support tools. As a first task, we succeeded to improve an important text preprocessing step—word lemmatization. The developed LemmaGen system proved to be superior in terms of accuracy and efficiency in comparison with other publicly available lemmatizers. As a core contribution of the thesis, we developed a new CrossBee methodology for discovering and ranking bridging terms, according to their potential to lead to new cross-domain scientific discoveries. We implemented the CrossBee methodology in an executable workflow in a novel browser-based workflow construction and execution platform ClowdFlows, which enables methodology adaptation and reuse. Additionally, the methodology evaluation shows promising results when searching for bringing terms in new domains, especially satisfactory are the results which evaluate the methodology performance in terms of the reduced user burden when searching for bridging clues. Moreover, we developed LemmaGen and CrossBee as online web applications with advanced user interface to support the end users when applying the developed methods to the domains of their choice.

The research presented in this thesis was focused on the following four main tasks:

1. An important task was the adaptation and improvement of selected text processing techniques. We have developed LemmaGen (see Chapter 5), a learning algorithm for automatic generation of lemmatization rules in the form of a refined ripple down rule (RDR) tree structure. The algorithm is very efficient and was used to produce accurate lemmatizers for 18 European languages. The main contributions of this thesis regarding the LemmaGen lemmatization engine are the following:
   - Improved compactness and readability of rule representation, by refining the RDR structure, resulting in a condensed rule representation (Section 5.3).
   - Design of a generic, non-incremental RDR learning algorithm with low worst-case time complexity (Section 5.4).

- Efficiency of the LemmaGen lemmatization algorithm empirically validated against two reference lemmatization algorithms. Improved efficiency was achieved by exploiting the design and compactness of the refined RDR structure (Section 5.5).
- Lemmatization rules, induced from Multext-East and Multext training lexica, which improved the accuracy and efficiency of lemmatization for most of the 18 European languages covered (Section 5.6).

2. The core task was the development of the CrossBee methodology for bisociative knowledge discovery, which given two document collections (from two separate domains) produces a ranked list of bridging terms (B-terms). The key feature of the ranked list of terms is that the top of the list should contain terms with a higher probability of representing bisociative links between the given domains (Chapter 6).
   - We developed a number of specially designed heuristic functions that provide a bisociation score quality estimate for each term. These base heuristics can be—based on the type of term features they exploit—divided into the following sets: frequency-based, TF-IDF-based, similarity-based, and outlier-based. (Section 6.1)
   - In the experimental evaluation, we tested a set of heuristics on the migraine-magnesium dataset and discussed the pitfalls of using base heuristics alone (Section 6.2).
   - The main contribution in the CrossBee methodology was the development of the improved ensemble-based heuristic, which employs a set of base heuristics to ensure robustness and stable performance across a variety of different datasets (Section 6.3).
   - We evaluated the ensemble-based methodology on two domains, migraine-magnesium and autism-calcineurin, showing that the proposed CrossBee methodology substantially reduces the end-user burden in terms of the length of the term list he needs to inspect to find some B-terms (Section 6.4).
3. We implemented the CrossBee methodology and the supporting LemmaGen lemmatization engine in various forms (Section 7.4).
   - We implemented the LemmaGen and CrossBee modules as reusable software components in complex ClowdFlows workflows. This enables experiment repeatability, software reuse, workflow adaptation and augmentation with new modules, ensuring the system's sustainability and reuse by the interested research community.
   - We provide SOAP web service counterparts of the developed ClowdFlows components.
   - We implemented the LemmaGen web application which is designed to be self-sufficient reference for the underlying LemmaGen library in terms of providing the library downloads, documentation, code tutorials and intuitive online lemmatization service that can be used either for educational or testing purposes.
   - We wrapped the developed functionality of LemmaGen and CrossBee in form of open source libraries.
4. We designed and implemented an online user interface CrossBee, which provides means for the users to apply the cross-domain knowledge discovery methodology on the domains of his interest. Additionally, it enables the users to explore the bisociative terms by efficient navigating through the data (Section 7.3).

- In addition to bridging terms ranking, CrossBee web application provides many other functionalities which help the scientist not only to find bridging hypotheses but also to check supportive evidence for them. CrossBee's visualization functionalities, in particular its presentation of pairs of documents, which can be inspected in more detail for meaningful relations, can be very helpful.
- We designed CrossBee by considering the principles of creativity supporting tools (e.g., by providing many different views and support tools the user can take advantage of when navigating the data), which should contribute to stimulating the user's creativity.

## 8.2 Discussion and further work

The presented work addressed several diverse research challenges in order to create an integrated modular solution to cross domain knowledge discovery. It is precisely this diversity of various components, which leaves us ample room for further research. The following ideas roughly follow the sequence of the methodological steps presented in the section about the top-level view of the methodology, illustrated in Figure 4.1.

### Text preprocessing

One possibility of extending our work in terms of text preprocessing is to experiment with alternative *term/concept detection approaches*. For this purpose, a simple n-gram approach was used since we were limited with the given gold standard datasets used in evaluation. The bounding condition were the B-terms provided in the dataset—the term detection method is required to detect the terms, which equal the given B-terms, otherwise the evaluation is not possible.

The term extraction methods, which we experimented with, unfortunately eliminated most of the provided B-terms, which disabled the usage of this particular approach. However, when the evaluation/comparison with existing cross-domain knowledge discovery approaches is not in question, we will be able to include either simple approaches like using MeSH controlled vocabulary thesaurus to filter the terms or *advanced term/concept extraction methods* like KeyGraph (Ohsawa et al., 1998), TermExtractor (Sclano and Velardi, 2007), LUIZ (Vintar, 2010), and others. We believe such an enhancement of the methodology would greatly improve the user's qualitative perception of the value of discovered terms, which would definitely lead to increased overall bisociation knowledge discovery performance.

### LemmaGen lemmatization

Another line of further research in the context of preprocessing is inevitably *additional work on the developed LemmaGen* lemmatization engine, which proved to be efficient, scalable, accurate and very well accepted by the users requiring the solution for the problem of lemmatization. The work to be done on LemmaGen falls in three categories: further improve algorithms producing more accurate lemmatization, evaluate and publish the results and disseminate the work through the LemmaGen website. The following three paragraphs present discussion and further work regarding these aspects.

*Improving LemmaGen algorithm accuracy* is among the priorities of further development. The current advantage and at the same time the pitfall of LemmaGen is ignoring the contexts of the words to be lemmatized. This is the advantage as it does not need to have the whole grammatically correct sentences to lemmatize; it works well on query-like word sets and works well also on out of vocabulary words; however some accuracy is lost due to ignoring the context of the words where available. LemmaGen has been successfully integrated into an advanced PoS tagger system Obeliks (see Grčar and Krek, 2012 and Online reference [21]), where LemmaGen is provided with the PoS tag information and consequently works more accurately, however this is quite a complex system to train and use. Therefore, it is in our future agenda to include such contextual information into the LemmaGen algorithm natively; this will however require substantial new research about this topic.

Additionally, *improving the lemmatization models' accuracies* without redesigning the algorithms will include training the models on the wordform-lemma pairs that have additional information about wordform average text occurrence. This would eliminate many errors due to lemmatization ambiguity; nevertheless, we need to gather the resources, which include the wordform average text occurrence, as it is not present in the Multext-East resources. In order to increase the accuracy, it would also be worthwhile to perform linguistic analysis of the LemmaGen rules and improve the algorithm by examining the types of the errors it makes.

*Additional evaluation and publication of LemmaGen* is also among the further work planned for LemmaGen. After the conception of the initial core algorithm presented in this thesis, many extensions have been developed that improve the lemmatization, e.g., additional prefix lemmatization rules, new ambiguity resolution heuristics, MDS considering models, and so on. Preliminary tests show promising results for some of newly developed features, however we need to do extensive evaluation and also describe and publish the achievements which turn out to be valuable.

Our further work includes also testing LemmaGen on other languages; the ultimate test will be the application of LemmaGen to resources of radically different languages, such as Arabic, which will certainly show the *limitations of wordform suffix analysis approach* used in our work. Besides word lemmatization, it would also be interesting to test whether the developed non-incremental RDR learning algorithm can be adapted to learning transformations from other examples with a string-like structure, such as amino acids in the case of proteins, and nucleotides in the case of genes.

*Providing and improving the support on the LemmaGen website* is the final item on our agenda regarding LemmaGen. The website proved to be an excellent dissemination channel for our research especially in terms of applicability, as shown in Section 7.2.4. First, we will update the website with the newest libraries with enhanced functionalities that need to be described and exemplified by new tutorials of usage. Next, due to the large number of existing versions, the unification of versioning and distribution is needed. For our internal development, we already switched to the modern distributed version control system Git (Online reference [47]), which needs to be made also publicly available. During the years of helping the users, we created relatively large questions and answers data, which we will use to set up frequently asked questions (FAQ) web page.

Last but not least, actions to *better promote LemmaGen website* by considering search engine optimizations (SEO) techniques will be done by slightly redesigning the web application to

better rank among search results and by providing links to our page from other high traffic web pages (e.g., Wikipedia article about lemmatization), which will additionally increase the visibility of LemmaGen. This concludes the discussion and further work regarding LemmaGen lemmatization engine and the text preprocessing in general.

## B-term ranking heuristics

Regarding the methodology for bisociative knowledge discovery we have several very stimulating ideas about *base heuristics for B-term ranking*, which we want to explore in the future. First we want to add some new, fundamentally different classes of base heuristics to rank the terms. The ideas for heuristics range from using SVM keywords (SVM trained to separate between domains) as potential B-terms with high score, to exploiting term relationship information provided by latent semantic indexing (LSI) of the domains (Deerwester et al., 1990). There are several similar plans for heuristics, which will bring additional information about the terms, which is currently hidden inside the documents.

Besides the stated classical information extraction (IE) measures, we have also more unconventional ideas for base heuristics; one of which is based on *the notion of B-documents*. Because our approach does not separate between documents and terms (both use the same representation), we can inject also the documents inside the term list before the ranking. After the ranking, the high scoring documents can be marked as B-documents and explored separately or they can be used to construct new heuristic, which will give high scores to the terms appearing in B-documents (this is analogous to the reasoning of outlier heuristics).

Additionally, *newly constructed B-document heuristics* can be added to the ensemble and the whole process can be iterated again, creating yet another set of B-documents and new heuristic. Doing this iteration loop many times will boost some unknown—yet to be researched—properties of the terms/documents. It will be extremely interesting to analyze such boosting scenario in terms of procedures behavior and convergence—will the ranking of terms be stationary after a while or will it diverge?

## B-term ranking ensemble

The most important issue in our further work is to reanalyze the core of the methodology for bisociative knowledge discovery, i.e., *the ensemble heuristic for B-term ranking* construction. Additional research regarding more systematic ways of constructing ensembles of heuristics to predict the bisociation score of a term is required. The controversy of our approach for ensemble construction is the manual construction of ensemble heuristic, even though the ensemble construction is a well-researched topic and many classical approaches like bagging, boosting, majority voting, random forest, and others exist.

Indeed, when being faced with the question of ensemble construction we tried many such approaches, which gave us exceptionally good results. For example, we used Weka's (Witten and Frank, 2005) M5P implementation of the classical regression tree learning algorithm M5' (Quinlan, 1992; Wang and Witten, 1997) for combing all base heuristics into a single function for predicting bisociation score. This step was followed by combining multiple such tree predictors with the Bagging approach (Breiman, 1996) to ensure robustness and the results we got were exceptionally good on the training migraine-magnesium domain. We could easily achieve over 99% AUC performance using the same setting that produced 65% AUC score as presented

in Section 6.2. However, when testing the resulting models on the second, autism-calcineurin, domain the AUC performance was seriously degraded compared to the manually constructed ensemble approach. Even though we realized that the problem is in overfitting the migraine-magnesium dataset, we were unable to resolve this issue.

We assume the reason lies in profoundly different statistical properties of the two datasets as discussed in Section 6.4.1. If we intentionally degraded the quality of learning models in the migraine-magnesium dataset, it resulted in improved accuracy in the autism-calcineurin dataset—there was an implicit inverse relationship between the AUC performances of the two datasets. Since the automatically constructed models which were approximately biased between the two datasets, performed no better than our manually constructed model, we decided to use the latter due to the clarity of the presented work—we can specify clear analytical equation for the ensemble construction as presented in Section 6.3.

Nevertheless, recently we were presented with the work in the field of *transfer learning* (Caruana, 1997; Pan and Yang, 2010), which provides some elaborate approaches for handling cases when the trainng domain has profoundly different feature distribution compared to the target domain. Even more specifically, our problem fits in the framework of *transductive transfer learning,* which is related to *domain adaptation* (Daume and Marcu, 2006), *sample selection bias* (Zadrozny, 2004) and *covariate shift* (Shimodaira, 2000), which all deal with similar problems and using similar assumptions about the data. First preliminary experiments show some promising results, which will be—guided by the presented areas—the top-most priority in the future work regarding the ensemble heuristic methodology.

### Bisociation score

Interpreting the created *bisociation score in a more general context* is another arising and exciting research questions. For instance, if we look at bisociation score, as a more general term importance measure (like TF-IDF), then we can use it to construct a text mining (TM) dataset (a set of documents vectors) whose features will be weighted by our bisociation score. Such dataset can be used in classical TM tasks like clustering and classification. In the case, when the described construction provides unwanted results, then bisociation score can be used as an additional weight along with TF-IDF, e.g., by multiplying the two. To get an overview of the behavior of presented approach we will perform experiments in our TopicCircle online document-clustering platform, which will be applied on the gold standard dataset and tested whether any of the previously published cross-domain relations reappear.

### CrossBee web application

The future work on the *CrossBee web application* will include polishing and improving data visualization and exploration facilities, further boosting the creativity aspect of the UI by consulting the research from the area creativity support tools (CST), and implementing other ideas, which can considerably improve the bisociative discovery process.

One such inspiring idea is adding the *interesting sack* functionality. This sack will be visible on the UI at all times (in all views) and the user will be able to drag and drop any displayed term, document or possibly some other element into the sack. Provided with the information which elements are of great importance to the user, our methodology will be able the recalculate heuristics in the background, create new base heuristics based on the interested documents

and terms and calibrate the ensemble to the specific domain-pair and the user's interest to boost the selected terms and documents. We speculate that such an approach will provide incomparably better results in terms of quality of term ranking from the end-users perspective.

## Methodology evaluation and new applications

Another line of future research will be the *evaluation of methodology* on other gold standard domains. In this respect, the alternative domain, which offers unchanged evaluation approaches is, e.g., the somatomedin C-arginine domain pair with the possibility of B-terms being extracted from the connecting arguments published by Swanson (1990). By using the third dataset like somatomedin Carginine would enable much more in-depth analysis of ensemble heuristics performance and avoiding the possibility of good results on the autism-calcineurin dataset being random.

Furthermore*, new domains outside the clique of literature-based discovery* (LBD) research and not based on PubMed data source need to be included in the methodology development and evaluation. The domains should still include gold standard B-terms, even though the provided terms may not be named B-terms (as this is LBD terminology), but the semantic value of such terms should still be the notion of bridging separate domains. Such outside domain is needed since the LBD research and PubMed can be considered a single domain when observing from a broader perspective. Only fundamentally different domains can prove the true robustness and domain independence of the methodology, as for now, we can only claim that the methodology is independent of domains inside biomedical scientific literature.

Regardless of additional extensions and testing described above, the created methodology for bisociative knowledge discovery is pointless if not *applied to a concrete bisociative knowledge discovery process, which the actual users/experts are interested in.* This is the last but not the least item of our future work regarding the approach established in this thesis. During the development of the CrossBee methodology and web application, we cooperated tightly with the domain experts from the area of biology; however, the focus was mostly biased on our part of research to develop the bisociative methodology, which the experts would be able to use without any external help. Although the experts were often positively surprised by the high ranking terms provided by CrossBee, and they confirmed receiving new insights about bridging the domains of their interest, the concrete research and publication, which would unambiguously confirm our approach, have not been achieved yet. This is by far the most important item in our future agenda regarding the research developed in this thesis.

# Acknowledgements

The path leading to this dissertation was, like most of such paths, full of ups and downs, mixed with strange crossroads, sleepless nights and countless cups of coffee, nevertheless, I had the great luck to be surrounded by exceptional people who supported and encouraged me to continue this journey regardless of the many difficulties encountered.

My first debt of gratitude goes to my supervisor Nada Lavrač. Her inspirational ideas and willingness to generously devote large amounts of her time made the realization of my PhD possible. Without her scientific excellence and broad knowledge I could have never reached the goals of this research. Similar gratitude goes to my co-supervisor Bojan Cestnik for creatively participating in numerous brainstorming debates and providing many stimulating ideas that ended up to be the building blocks of this thesis.

Many thanks go to the members of my PhD committee: Tanja Urbančič, Tomaž Erjavec, and Hannu Toivonen. I am thankful for their fast and precise evaluation and for providing very relevant comments which undoubtedly increased the scientific value of my work.

I wish to thank the funding bodies responsible for financing me, especially the Slovenian Research Agency for financing my PhD study and the European Commission for funding the research projects BISON and FIRST on which I collaborated.

I thank Igor Mozetič and Ingrid Petrič for their help in the initial phases of this research as well as coworkers Dragi Kocev, Ivica Slavkov and Darko Aleksovski for sharing their specialized data mining knowledge. I am grateful to all the other colleagues of the Department of Knowledge Technologies at the Jožef Stefan Institute. Each of them is very important piece of the puzzle establishing an excellent research and working environment. Mili Bauer and Tina Anžič are the glue holding this puzzle together and I appreciate their personal touch in every administrative issue I had as well as their efforts in minimizing my bureaucracy work.

Special thanks for putting up with me and spending their time in infinite mutual motivational debates go to my office mates. In the chronological order they are Vid Podpečan, Borut Sluban, and Miha Grčar. Besides, I wish to thank Petra Kralj Novak for introducing me to this wonderful community which opened my doors into the exciting world of science.

A large "thank you" goes to all my friends for their moral support and understanding. The two connected to my PhD the most are Darko Čerepnalkoski whose clear-minded vision and tireless persuasion in biking kept me sane all this time and Jerneja Šajn who substituted me in my parental role during the numerous work-intensive periods.

I would like to express my deep gratitude to my parents Marija and Franci, who made my peruse of such high education possible in the first place. I'm particularly grateful also to my parents-in-law Vojka and Lado who kept my family operational in the times of my absence.

Finally, the most sincere thanks go to my close family. I would have not been able to push myself so far without your endless supply of love and happiness. Thank you my dearest Nataša for giving me the needed time and for your patience and confidence in me during this process.

# References


Agrawal, R.; Mannila, H.; Srikant, R.; Toivonen, H.; Verkamo, A. I. Fast discovery of association rules. *Advances in Knowledge Discovery and Data Mining* **12**, 307–328 (AAAI Press, Menlo Park, USA, 1996).

Berthold, M.; Dill, F.; Kötter, T.; Thiel, K. Supporting Creativity: Towards Associative Discovery of New Insights. In: *Advances in Knowledge Discovery and Data Mining*. Lecture Notes in Computer Science **5012**, 14–25 (Springer, Heidelberg, Germany, 2008).

Berthold, M. R., ed. *Bisociative Knowledge Discovery* (Springer, Heidelberg, Germany, 2012).

Berthold, M. R. Towards Bisociative Knowledge Discovery. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 1–10 (Springer, Heidelberg, Germany, 2012).

Boden, M. A. Creativity and Artificial Intelligence. *Artificial Intelligence* **103**, 347–356 (1998).

Boden, M. A. *The Creative Mind - Myths and Mechanisms* (Routledge, London, UK, 2004).

Breiman, L. Bagging Predictors. *Machine Learning* **24**, 123–140 (1996).

Caruana, R.; Niculescu-Mizil, A. Data mining in metric space: an empirical analysis of supervised learning performance criteria. In: *Proceedings of the 10th SigKdd International Conference on Knowledge Discovery and Data Mining*. 69–78 (ACM Press, New York, USA, 2004).

Caruana, R. Multitask Learning. *Machine Learning* **28**, 41–75 (1997).

Chen, C. *Mapping Scientific Frontiers: The Quest for Knowledge Visualization* (Springer, London, UK, 2003).

Cios, K. J.; Swiniarski, R. W.; Pedrycz, W.; Kurgan, L. A. The Knowledge Discovery Process. *Data Mining: A Knowledge Discovery Approach*. 9–24 (Springer, New York, USA, 2007).

Colton, S.; Wiggins, G. A. Computational Creativity: The Final Frontier? In: *Proceedings of the 20th European Conference on Artificial Intelligence* **242**, 21–26 (IOS Press, Fairfax, USA, 2012).

Compton, P.; Jansen, R. Knowledge in context: a strategy for expert system maintenance. In: *Proceedings of the 2nd Australian Joint Conference on Artificial Intelligence*. 283–297 (Springer, New York, USA, 1988).

Compton, P.; Jansen, R. A philosophical basis for knowledge acquisition. *Knowledge Acquisition* **2**, 241–257 (1990).

Dalianis, H.; Jongejan, B. Hand-crafted versus Machine-learned Inflectional Rules: The Euroling-SiteSeeker Stemmer and CST's Lemmatiser. In: *Proceedings of the 5th International Conference on Language Resources and Evaluation*. 663–666 (ELRA, Paris, France, 2006).

Daume, I. H.; Marcu, D. Domain Adaptation for Statistical Classifiers. *Journal of Artficial Intelligence Research* **26**, 101–126 (2006).

Deerwester, S. C.; Dumais, S. T.; Landauer, T. K.; Furnas, G. W.; Harshman, R. A. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science* **41**, 391–407 (1990).

Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research* **7**, 1–30 (2006).

Dietterich, T. G. Ensemble Methods in Machine Learning. *Lecture Notes in Computer Science* **1857**, 1–15 (2000).

Digiacomo, R. A.; Kremer, J. M.; Shah, D. M. Fish-oil dietary supplementation in patients with Raynaud's phenomenon: A double-blind, controlled, prospective study. *The American Journal of Medicine* **86**, 158–164 (1989).

Dubitzky, W.; Kötter, T.; Schmidt, O.; Berthold, M. Towards Creative Information Exploration Based on Koestler's Concept of Bisociation. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 11–32 (Springer, Heidelberg, Germany, 2012).

Dunn, O. J. Multiple Comparisons Among Means. *Journal of the American Statistical Association* **56**, 52–64 (1961).

Dwork, C.; Kumar, R.; Naor, M.; Sivakumar, D. Rank aggregation methods for the Web. In: *Proceedings of the 10th International Conference on World Wide Web.* 613–622 (ACM Press, New York, USA, 2001).

Džeroski, S.; Erjavec, T. Learning to Lemmatise Slovene Words. In: *Learning Language in Logic.* Lecture Notes in Computer Science **1925**, 415–434 (Springer, New York, 2000).

Erjavec, T.; Džeroski, S. Machine Learning of Morphosyntactic Structure: Lemmatizing Unknown Slovene Words. *Applied Artificial Intelligence* **18**, 17–41 (2004).

Erjavec, T.; Sárossy, B. Morphosyntactic Tagging of Slovene Legal Language. *Informatica (Slovenia)* **30**, 483–488 (2006).

Erjavec, T. MULTEXT-East Version 4: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In: *Proceedings of the 7th International Conference on Language Resources and Evaluation.* 2544–2547 (ELRA, Paris, France, 2010).

Fawcett, T. An introduction to ROC analysis. *Pattern Recognition Letters* **27**, 861–874 (2006).

Fayyad, U.; Piatetsky-Shapiro, G.; Smyth, P. From data mining to knowledge discovery in databases. *Ai Magazine* **17**, 37–54 (1996).

Feldman, R.; Sanger, J. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data* (Cambridge University Press, Cambridge, UK, 2007).

Fisher, R. A. *Statistical Methods and Scientific Inference* (Oliver and Boyd, Edinburgh, UK, 1959).

Flach, P. A.; Wu, S. Repairing Concavities in ROC Curves. In: *Proceedings of the International Joint Conferences on Artificial Intelligence.* 702 (AAAI Press, Menlo Park, USA, 2005).

Forth, J.; Wiggins, G. A.; McLean, A. Unifying Conceptual Spaces: Concept Formation in Musical Creative Systems. *Minds and Machines* **20**, 503–532 (2010).

Fortuna, B.; Lavrač, N.; Velardi, P. Advancing Topic Ontology Learning through Term Extraction. In: *Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence*. Lecture Notes in Computer Science **5351**, 626–635 (Springer, Heidelberg, Germany, 2008).

Friedman, M. A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics* **11**, 86–92 (1940).

Gordon, M. D.; Dumais, S. Using Latent Semantic Indexing for literature-based discovery. *Journal of the American Society for Information Science* **49**, 674–685 (1998).

Gordon, M. D.; Lindsay, R. K. Toward Discovery Support Systems: A Replication, Re-examination, and Extension of Swanson's Work on Literature-Based Discovery of a Connection between Raynaud's and Fish Oil. *Journal of the American Society for Information Science* **47**, 116–28 (1996).

Gordon, M.; Lindsay, R. K.; Fan, W. Literature Based Discovery on the World Wide Web. In: *ACM Transactions on Internet Technology*. 261–275 (ACM Press, New York, USA, 2001).

Grčar, M.; Krek, S. Obeliks: statistični oblikoskladenjski označevalnik in lematizator za slovenski jezik. In: *Proceedings of the 8th Language Technologies Conference* **C**, 89–94 (IJS, Ljubljana, Slovenia, 2012).

Han, J.; Kamber, M. *Data Mining: Concepts and Techniques* (Morgan Kaufmann, Burlington, US, 2006).

Hand, D. J.; Mannila, H.; Smyth, P. *Principles of Data Mining* (The MIT Press, Cambridge, USA, 2001).

Higgins, J. M. *101 Creative Problem Solving Techniques: The Handbook of New Ideas for Business* (New Management Publishing Company, New York, USA, 2005).

Hodge, V. J.; Austin, J. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review* **22**, 85–126 (2004).

Hoi, S. C. H.; Jin, R. Semi-supervised ensemble ranking. In: *Proceedings of the 23rd National Conference on Artificial intelligence* **2**, 634–639 (AAAI Press, Menlo Park, USA, 2008).

Hristovski, D.; Stare, J.; Peterlin, B.; Džeroski, S. Supporting discovery in medicine by association rule mining in Medline and UMLS. *Medinfo* **10**, 1344–8 (2001).

Hristovski, D.; Peterlin, B.; Mitchell, J. A.; Humphrey, S. M. Using literature-based discovery to identify disease candidate genes. *International journal of medical informatics* **74**, 289–298 (2005).

Hristovski, D.; Friedman, C.; Rindflesch, T. C.; Peterlin, B. Exploiting semantic relations for literature-based discovery. In: *AIMA Annual Symposium proceedings*. 349–353 (AIMA, Bethesda, USA, 2006).

Hristovski, D.; Friedman, C.; Rindflesch, T. C.; Peterlin, B. Literature-Based Knowledge Discovery using Natural Language Processing. In: *Literature-based Discovery*. Information Science and Knowledge Management **15**, 133–152 (Springer, Heidelberg, Germany, 2008).

Ide, N.; Véronis, J. MULTEXT: Multilingual Text Tools and Corpora. In: *Proceedings of the 15th Conference on Computational linguistics*. 588–592 (ACL, Morristown, USA, 1994).

Jensen, L. J.; Saric, J.; Bork, P. Literature mining for the biologist: from information retrieval to biological discovery. *Nature Reviews Genetics* **7**, 119–129 (2006).

Jones, K. S. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* **28**, 11–21 (1972).

Jones, K. S. Document Retrieval: Shallow Data, Deep Theories; Historical Reflections, Potential Directions. *Advances in Information Retrieval*. Lecture Notes in Computer Science **2633**, 1–11 (Springer, Heidelberg, Germany, 2003).

Juršič, M.; Mozetič, I.; Lavrač, N. Learning Ripple Down Rules for Efficient Lemmatization. In: *Proceedings of the 10th International Multiconference Information Society*. 206–209 (IJS, Ljubljana, Slovenia, 2007).

Juršič, M.; Mozetič, I.; Erjavec, T.; Lavrač, N. LemmaGen: Multilingual Lemmatisation with Induced Ripple-Down Rules. *Journal of Universal Computer Science* **16**, 1190–1214 (2010).

Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging Concept Identification for Constructing Information Networks from Text Documents. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 66–90 (Springer, Heidelberg, Germany, 2012a).

Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Bisociative Literature Mining by Ensemble Heuristics. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 338–358 (Springer, Heidelberg, Germany, 2012b).

Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Cross-domain literature mining: Finding bridging concepts with CrossBee. In: *Proceedings of the 3rd International Conference on Computational Creativity*. 33–40 (University College Dublin, Dublin, Ireland, 2012c).

Kiparsky, P. "Elsewhere" in phonology. In: *A Festschrift for Morris Halle*. 93–106 (Holt, Rinehart and Wilson, New York, 1973).

Koestler, A. *The Act of Creation* (Penguin Books, New York, USA, 1964).

Kötter, T.; Berthold, M. From Information Networks to Bisociative Information Networks. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 33–50 (Springer, Heidelberg, Germany, 2012).

Kranjc, J.; Podpečan, V.; Lavrač, N. ClowdFlows: A Cloud Based Scientific Workflow Platform. In: *Machine Learning and Knowledge Discovery in Databases*. Lecture Notes in Computer Science **7524**, 816–819 (Springer, Heidelberg, Germany, 2012).

Lavrač, N.; Sluban, B.; Juršič, M. Cross-Domain Literature Mining through Outlier Document and Bridging Concept Detection. In: *Proceedings of the Workshop on Analysis of Complex Networks at ECML PKDD*. 35–47 (ACNE Committee, Barcelona, Spain, 2010).

Lindsay, R.; Gordon, M. Literature-Based Discovery by Lexical Statistics. *Journal of the American Society for Information Science* **50**, 574–587 (1999).

Lovins, J. B. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics* **11**, 22–31 (1968).

Luhn, H. P. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development* **2**, 159–165 (1958).

Macedoni-Lukšič, M.; Petrič, I.; Cestnik, B.; Urbančič, T. Developing a Deeper Understanding of Autism: Connecting Knowledge through Literature Mining. *Autism Research and Treatment* **2011**, 1–8 (2011).

Manandhar, S.; Džeroski, S.; Erjavec, T. Learning Multilingual Morphology with CLOG. In: *Proceedings of the 8th International Workshop on Inductive Logic Programming*. 135–144 (Springer, London, UK, 1998).

Mladenić, D. Combinatorial Optimization in Inductive Concept Learning. In: *Proceedings of the 10th International Conference on Machine Learning*. 205–211 (Morgan Kaufmann, San Francisco, USA, 1993).

Mladenić, D. Learning word normalization using word suffix and context from unlabeled data. In: *Proceedings of the 19th International Conference on Machine Learning*. 427–434 (Morgan Kaufmann, San Francisco, USA, 2002a).

Mladenić, D. Automatic word lemmatisation. In: *Proceedings of the 5th International Multiconference Information Society* **B**, 153–159 (IJS, Ljubljana, Slovenia, 2002b).

Nemenyi, P. *Distribution-free multiple comparisons* (Princeton University, New Jersey, USA, 1963).

Ohsawa, Y.; Benson, N. E.; Yachida, M. KeyGraph: automatic indexing by co-occurrence graph based on building construction metaphor. In: *Proceedings of the 4th Conference on Advances in Digital Libraries Conference*. 12–18 (ACM Press, New York, USA, 1998).

Pan, S. J.; Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* **22**, 1345–1359 (2010).

Perez-Iratxeta, C.; Wjst, M.; Bork, P.; Andrade, M. A. G2D: A Tool for Mining Genes Associated with Disease. *BMC Genet* **6**, 1–9 (2005).

Petrič, I.; Urbančič, T.; Cestnik, B.; Macedoni-Lukšič, M. Literature mining method RaJoLink for uncovering relations between biomedical concepts. *Journal of Biomedical Informatics* **42**, 219–227 (2009).

Petrič, I.; Cestnik, B.; Lavrač, N.; Urbančič, T. Outlier Detection in Cross-Context Link Discovery for Creative Literature Mining. *The Computer Journal* **55**, 47–61 (2012a).

Petrič, I.; Cestnik, B.; Lavrač, N.; Urbančič, T. Bisociative Knowledge Discovery by Literature Outlier Detection. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 313–324 (Springer, Heidelberg, Germany, 2012b).

Petrič, I. *Text mining for discovering implicit relationships in biomedical literature* (Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2009).

Plisson, J.; Lavrač, N.; Mladenić, D.; Erjavec, T. Ripple Down Rule learning for automated word lemmatisation. *AI Commun.* **21**, 15–26 (2008).

Plucker, J. A.; Beghetto, R. A.; Dow, G. T. Why Isn't Creativity More Important to Educational Psychologists? Potentials, Pitfalls, and Future Directions in Creativity Research. *Educational Psychologist* **39**, 83–96 (2004).

Popovič, M.; Willett, P. Processing of Documents and Queries in a Slovene Language Free Text Retrieval System. *Literary and Linguistic Computing* **5**, 182–190 (1990).

Porter, M. F. An algorithm for suffix stripping. *Program* **14**, 130–137 (1980).

Pratt, W.; Yetisgen-Yildiz, M. LitLinker: capturing connections across the biomedical literature. In: *Proceedings of the 2nd international conference on Knowledge capture*. 105–112 (ACM Press, New York, USA, 2003).

Provost, F. J.; Fawcett, T. Robust Classification for Imprecise Environments. *Machine Learning* **42**, 203–231 (2001).

Quinlan, J. R. Learning with Continuous Classes. In: *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*. 343–348 (World Scientific, Singapore, Singapore, 1992).

Resnick, M.; Myers, B.; Nakakoji, K.; Shneiderman, B.; Pausch, R.; Selker, T.; Eisenberg, M. Design Principles for Tools to Support Creative Thinking. In: *Proceedings of the NSF Workshop on Creativity Support Tools*. 25–36 (CST Organising Committee, Washington, USA, 2005).

Ritchie, G. D. A closer look at creativity as search. In: *Proceedings of the 3rd International Conference on Computational Creativity*. 41–48 (Open University Press, Dublin, Ireland, 2012).

Rivest, R. L. Learning decision lists. *Machine Learning* **2**, 229–246 (1987).

Robertson, S. E.; Jones, K. S. Relevance weighting of search terms. *Journal of the American Society for Information Science* **27**, 129–146 (1976).

Rokach, L. Ensemble-based classifiers. *Artificial Intelligence Review* **33**, 1–39 (2009).

Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management* **24**, 513–523 (1988).

Salton, G.; Wong, A.; Yang, C. S. A vector space model for automatic indexing. *Communications of the ACM* **18**, 613–620 (1975).

Sclano, F.; Velardi, P. TermExtractor: a Web Application to Learn the Shared Terminology of Emergent Web Communities. In: *Enterprise Interoperability II*. 287–290 (Springer London, London, 2007).

Segond, M.; Borgelt, C. BisoNet Generation using Textual Data. In: *Proceedings of the Workshop on Explorative Analytics of Information Networks at ECML PKDD*. 12–22 (ECML/PKDD Organization Committee, Bled, Slovenia, 2009).

Sehgal, A. K.; Qiu, X. Y.; Srinivasan, P. Analyzing LBD Methods using a General Framework. *Literature-based Discovery*. Information Science and Knowledge Management **15**, 75–100 (Springer, Heidelberg, Germany, 2008).

Shearer, C. The CRISP-DM Model: The new blueprint for data mining. *Journal of Data Warehousing* **5**, 13–22 (2000).

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference* **90**, 227–244 (2000).

Shneiderman, B.; Fischer, G.; Czerwinski, M.; Resnick, M.; Myers, B.; Candy, L.; Edmonds, E.; Eisenberg, M.; Giaccardi, E.; Hewett, T.; Jennings, P.; Kules, B.; Nakakoji, K.; Nunamaker, J.; Pausch, R.; Selker, T.; Sylvan, E.; Terry, M. Creativity Support Tools: Report From a U.S. National Science Foundation Sponsored Workshop. *International Journal of Human-Computer Interaction* **20**, 61–77 (2006).

Shneiderman, B. Creativity support tools: accelerating discovery and innovation. *Communications of the ACM* **50**, 20–32 (2007).

Shneiderman, B. Creativity Support Tools: A Grand Challenge for HCI Researchers. In: *Engineering the User Interface*. 1–9 (Springer, London, UK, 2009).

Sluban, B.; Lavrač, N. Supporting the search for cross-context links by outlier detection methods. *BMC Bioinformatics* **11**, 1–2 (2010).

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating Outliers for Cross-Context Link Discovery. In: *Artificial Intelligence in Medicine*. Lecture Notes in Computer Science **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).

Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the Power of Outliers for Cross-Domain Literature Mining. In: *Bisociative Knowledge Discovery*. Lecture Notes in Computer Science **7250**, 325–337 (Springer, Heidelberg, Germany, 2012).

Smalheiser, N. R.; Swanson, D. R. Using ARROWSMITH: a computer-assisted approach to formulating and assessing scientific hypotheses. *Computer Methods and Programs in Biomedicine* **57**, 149–153 (1998).

Smalheiser, N.; Torvik, V. The Place of Literature-Based Discovery in Contemporary Scientific Practice. *Literature-based Discovery*. Information Science and Knowledge Management **15**, 13–22 (Springer, Heidelberg, Germany, 2008).

Smalheiser, N.; Torvik, V.; Bischoff-Grethe, A.; Burhans, L.; Gabriel, M.; Homayouni, R.; Kashef, A.; Martone, M.; Perkins, G.; Price, D.; Talk, A.; West, R. Collaborative development of the Arrowsmith two node search interface designed for laboratory investigators. *Journal of Biomedical Discovery and Collaboration* **1**, 1–8 (2006).

Smalheiser, N. R.; Torvik, V. I.; Zhou, W. Arrowsmith two-node search interface: a tutorial on finding meaningful links between two disparate sets of articles in MEDLINE. *Computer methods and programs in biomedicine* **94**, 190–197 (2009).

Srinivasan, A.; Compton, P.; Malor, R.; Edwards, G.; Sammut, C.; Lazarus, L. Knowledge Acquisition in Context for a Complex Domain. In: *Proceedings of the 5th European Knowledge Acquisition Workshop*. 1–10 (Pergamon, Oxford, UK, 1991).

Swanson, D. R.; Smalheiser, N. R. Undiscovered Public Knowledge: a Ten-Year Update. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining.* 295–298 (AAAI Press, Menlo Park, USA, 1996).

Swanson, D. R.; Smalheiser, N. R. An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artificial Intelligence* **91**, 183–203 (1997).

Swanson, D. R.; Smalheiser, N. R.; Torvik, V. I. Ranking indirect connections in literature-based discovery: The role of medical subject headings. *Journal of the American Society for Information Science and Technology* **57**, 1427–1439 (2006).

Swanson, D. R. Undiscovered Public Knowledge. *The Library Quarterly* **56**, 103–118 (1986a).

Swanson, D. R. Fish oil, Raynaud's syndrome, and undiscovered public knowledge. *Perspectives in Biology and Medicine* **30(1)**, 7–18 (1986b).

Swanson, D. R. Migraine and magnesium: eleven neglected connections. *Perspectives in Biology and Medicine* **31**, 526–557 (1988).

Swanson, D. R. Medical literature as a potential source of new knowledge. *Bulletin of the Medical Library Association* **78**, 29–37 (1990).

Swanson, D. R. Intervening in the Life Cycles of Scientific Knowledge. *Library Trends* **41**, 606–631 (1993).

Swanson, D. R. Literature-Based Discovery? The Very Idea. *Literature-based Discovery.* Information Science and Knowledge Management **15**, 3–11 (Springer, Heidelberg, Germany, 2008).

Tiffin, N.; Kelso, J. F.; Powell, A. R.; Pan, H.; Bajic, V. B.; Hide, W. A. Integration of text- and data-mining using ontologies successfully selects disease gene candidates. *Nucleic acids research* **33**, 1544–1552 (2005).

Urbančič, T.; Petrič, I.; Cestnik, B.; Macedoni-Lukšič, M. Literature Mining: Towards Better Understanding of Autism. In: *Proceedings of the 11th Conference on Artificial Intelligence in Medicine.* Lecture Notes in Computer Science **4594**, 217–226 (Springer, Berlin, Heidelberg, 2007).

Urbančič, T.; Petrič, I.; Cestnik, B. RaJoLink: A Method for Finding Seeds of Future Discoveries in Nowadays Literature. In: *Proceedings of the 18th International Symposium on Foundations of Intelligent Systems.* Lecture Notes in Computer Science **5722**, 129–138 (Springer, Heidelberg, Germany, 2009).

Veale, T.; Gervás, P.; Pease, A. Understanding creativity: A computational perspective. *New Generation Computing* **24**, 203–207 (2006).

Veale, T.; Gervás, P.; Pérez y Pérez, R. Computational Creativity: A Continuing Journey. *Minds and Machines* **20**, 483–487 (2010).

Vintar, Š. Bilingual term recognition revisited: The bag-of-equivalents term alignment approach and its evaluation. *Terminology* **16**, 141–158 (2010).

Wang, Y.; Witten, I. H. Induction of model trees for predicting continuous classes. In: *Proceedings of the 9th European Conference on Machine Learning*. 128–137 (Springer, London, UK, 1997).

Weaver, K. Magnesium and Migraine. *Headache: The Journal of Head and Face Pain* **30**, 168 (1990).

Weeber, M.; Klein, H.; de Jong van den Berg, L. T.; Vos, R. Using concepts in literature-based discovery: Simulating Swanson's Raynaud–fish oil and migraine–magnesium discoveries. *Journal of the American Society for Information Science and Technology* **52**, 548–557 (2001).

Weeber, M.; Kors, J. A.; Mons, B. Online tools to support literature-based discovery in the life sciences. *Briefings in Bioinformatics* **6**, 277–286 (2005).

Wiggins, G. A. A Preliminary Framework for Description, Analysis and Comparison of Creative Systems. *Journal of Knowledge Based Systems* **19**, 449–470 (2006).

Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biometrics Bulletin* **1**, 80–83 (1945).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques* (Morgan Kaufmann, Burlington, US, 2005).

Wren, J. D.; Bekeredjian, R.; Stewart, J. A.; Shohet, R. V.; Garner, H. R. Knowledge discovery by automated identification and ranking of implicit relationships. *Bioinformatics* **20**, 389–398 (2004).

Yetisgen-Yildiz, M.; Pratt, W. Using statistical and knowledge-based approaches for literature-based discovery. *Journal of Biomedical Informatics* **39**, 600–611 (2006).

Yetisgen-Yildiz, M.; Pratt, W. Evaluation of Literature-Based Discovery Systems. *Literature-based Discovery*. Information Science and Knowledge Management **15**, 101–113 (Springer, Heidelberg, Germany, 2008).

Yetisgen-Yildiz, M.; Pratt, W. A new evaluation methodology for literature-based discovery systems. *Journal of Biomedical Informatics* **42**, 633–643 (2009).

Zadrozny, B. Learning and evaluating classifiers under sample selection bias. In: *Proceedings of the 21st International Conference on Machine Learning*. 903–910 (ACM Press, New York, USA, 2004).

# Online references

*Note to the reader: please use http://phd.matjaz.jursic.si to retrieve the updated list of online references in the case when a chosen link is broken.*

This section lists all online links that are referenced in the thesis. We defined several groups in order to break the long list into smaller and manageable sets of links, which are used in similar contexts. Every link contains a short description about it, contents and wherever possible lists references to published work about it.

This section is provided due to a relatively large number of online links that we reference and are of extreme importance (sometimes the main contribution) for this thesis. One of the web's basic properties is its constant change and many of the stated links are expected to be unavailable much before the printed version of this thesis will be unavailable. Although we will try to maintain our sites on provided links as long as possible, we cannot give this guarantee for the outside links that are not under our control. This list tries to combat this unstable situation by gathering all links in one place—instead of scattering them all throughout the thesis—and additionally providing a single web page (emphasized at the begging of this section) where the list will be kept updated.

## LemmaGen native resources

[1] Main website entry page: *http://lemmatise.ijs.si*

[2] Online lemmatization service: *http://lemmatise.ijs.si/Services*

[3] Older version of online service suitable for automatic large text and whole file lemmatization: *http://lemmagen.ijs.si/Lemmatisation.aspx*

[4] Software libraries download page: *http://lemmatise.ijs.si/Software*

[5] Software library, the exact version described in this thesis downloadable from: *http://lemmatise.ijs.si/Download/File/Software%23LemmaGen_v2.0.zip*

[6] Lemmatization SOAP web service resource definitions: *http://lemmatise.ijs.si/Services?wsdl*

[7] Example of LemmaGen library usage for lemmatization: *http://lemmatise.ijs.si/Software/ExampleSimpleUsage*

[8] Example of LemmaGen library usage for outputting the lemmatization rules: *http://lemmatise.ijs.si/Software/ExampleDecTreeOut*

[9] List of publications: *http://lemmatise.ijs.si/Documents*

[10] Contact information: *http://lemmatise.ijs.si/Home/Contact*

## Lemmatizers and online lemmatization services

[11] CST's Lemmatiser: *http://www.cst.dk/online/lemmatiser/uk/*

[12] RDR Lemmatizer *http://nl2.ijs.si/analyze/lemRDR.tgz*

[13] Javascript Porter Stemmer Online: *http://qaa.ath.cx/porter_js_demo.html*

[14] CST's online tools: *http://ida.hum.ku.dk/tools/index.php*
[15] Stemming and Lemmatization with Python NLTK: *http://text-processing.com/demo/stem/*
[16] Snowball – Demo: *http://snowball.tartarus.org/demo.php*
[17] SMILE Text Analyzer: *http://smile-stemmer.appspot.com/*

## Multext-East resources

[18] Multext-East main web page: *http://nl.ijs.si/ME/*
[19] Multext-East v3 datasets: *http://nl.ijs.si/ME/Vault/V3/*
[20] Multext-East v4 datasets: *http://nl.ijs.si/ME/V4/*

## LemmaGen dissemination and impact

[21] Obeliks, part-of-speech tagger for Slovene language: *http://www.slovenscina.eu/tehnologije/oznacevalnik*
[22] Orange data mining toolbox: *http://orange.biolab.si/*
[23] Lemmagen-Python-Extension project, Python extension code to expose lemmatization functions of LemmaGen: *https://github.com/sushantkhurana/Lemmagen-Python-Extension*
[24] Lemmagen4J, LemmaGen rewritten from C# to Java: *http://zitnik.si/mediawiki/index.php?title=Software*
[25] Project slovene_lemmatizer, creating Slovene lemmatizer based on C++ version of LemmaGen: *https://bitbucket.org/mavrik/slovene_lemmatizer*
[26] Text-processing-bundle project: A bundle containing code related to processing of unstructured sources: *https://github.com/AlertProject/Text-processing-bundle*
[27] Sparc.TagCloud, A TagCloud library for .NET: *https://github.com/chrisdavies/Sparc.TagCloud*
[28] Taverna 2 workflow based on LemmaGen: *http://www.myexperiment.org/workflows/1738.html*

## CrossBee, supplementary tools and underlying libraries

[29] CrossBee main website entry page: *http://crossbee.ijs.si*
[30] CrossBee SOAP web services resource definitions: *http://crossbee.ijs.si/Services?wsdl*
[31] CrossBee downloadable software library: *http://source.ijs.si/btermdiscovery*
[32] TopicCircle online document clustering platform: *http://crossbee.ijs.si/TopicCircle*
[33] LATINO, Link analysis and text mining toolbox: *http://sourceforge.net/projects/latino/*

## ClowdFlows workflow execution environment

[34] ClowdFlows online workflow execution environment: *http://clowdflows.org* or *http://workflow.ijs.si*

[35] ClowdFlows, static version (not updated with new features) containing the exact workflows presented in this thesis *http://phd.matjaz.jursic.si/crossbeeClowdflows*

## Other related references

[36] BISON project, Bisociation Networks for Creative Information Discovery, 7th Framework Programme FET-Open European Union project 2008–2011: *http://www.BisoNet.eu/*

[37] ELRA, European Language Resources Association: *http://www.elra.info/*

[38] LDC, Linguistic Data Consortium: *http://www.ldc.upenn.edu/*

[39] MEDLINE bibliographic database of life sciences and biomedical information: *http://www.nlm.nih.gov/pubs/factsheets/medline.html*

[40] PUBMED freely accessible database of citations and abstracts on life sciences and biomedical topics: *http://www.ncbi.nlm.nih.gov/pubmed*

[41] MeSH, controlled vocabulary thesaurus: *http://www.nlm.nih.gov/pubs/factsheets/mesh.html*

[42] UMLS, Unified Medical Language System, a collection of knowledge sources and tools to facilitate the development of computer systems: *http://www.nlm.nih.gov/pubs/factsheets/umls.html*

[43] Unicode Standard Annex #29: *http://www.unicode.org/reports/tr29/#Word_Boundaries.*

[44] Snowball: A small string processing language designed for creating stemming algorithms: *http://snowball.tartarus.org*

[45] Google Trends: *http://www.google.com/trends/*

[46] Google AdWords: *http://adwords.google.com/*

[47] Git, a free and open source distributed version control system: *http://git-scm.com/*

# Index of Figures

# Index of Tables

# Appendix

## A TopicCircle

This appendix presents an additional tool developed as part of research presented in this thesis, i.e., a browser-based online web application for hierarchal semi-automatic document clustering named *TopicCircle* (Online references [32]). Figure A.1 shows a basic interactive view of the clusters which also demonstrates why the system is called TopicCircle.

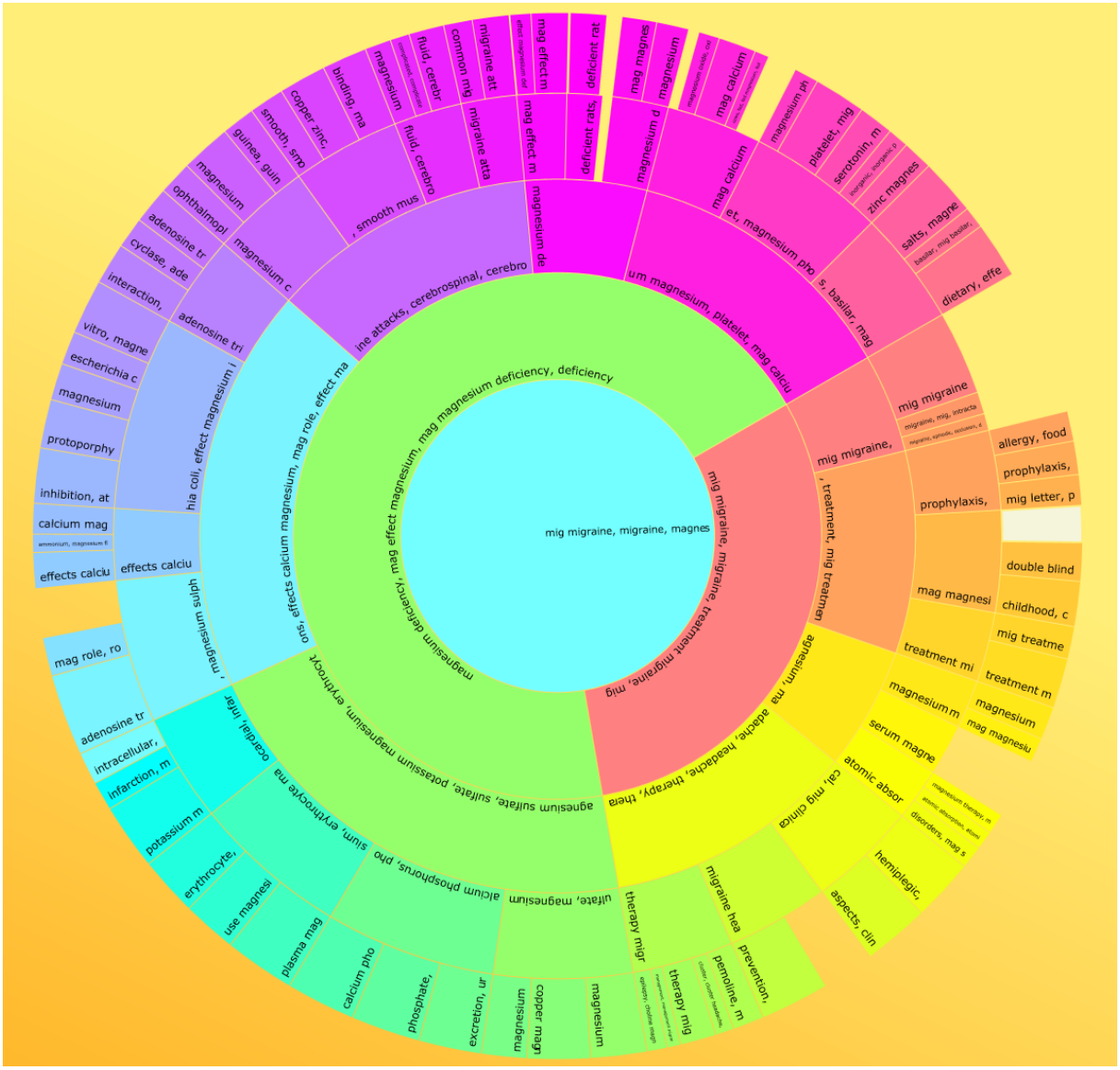



Figure A.1: *The basic TopicCircle's hierarchical cluster visualization.*

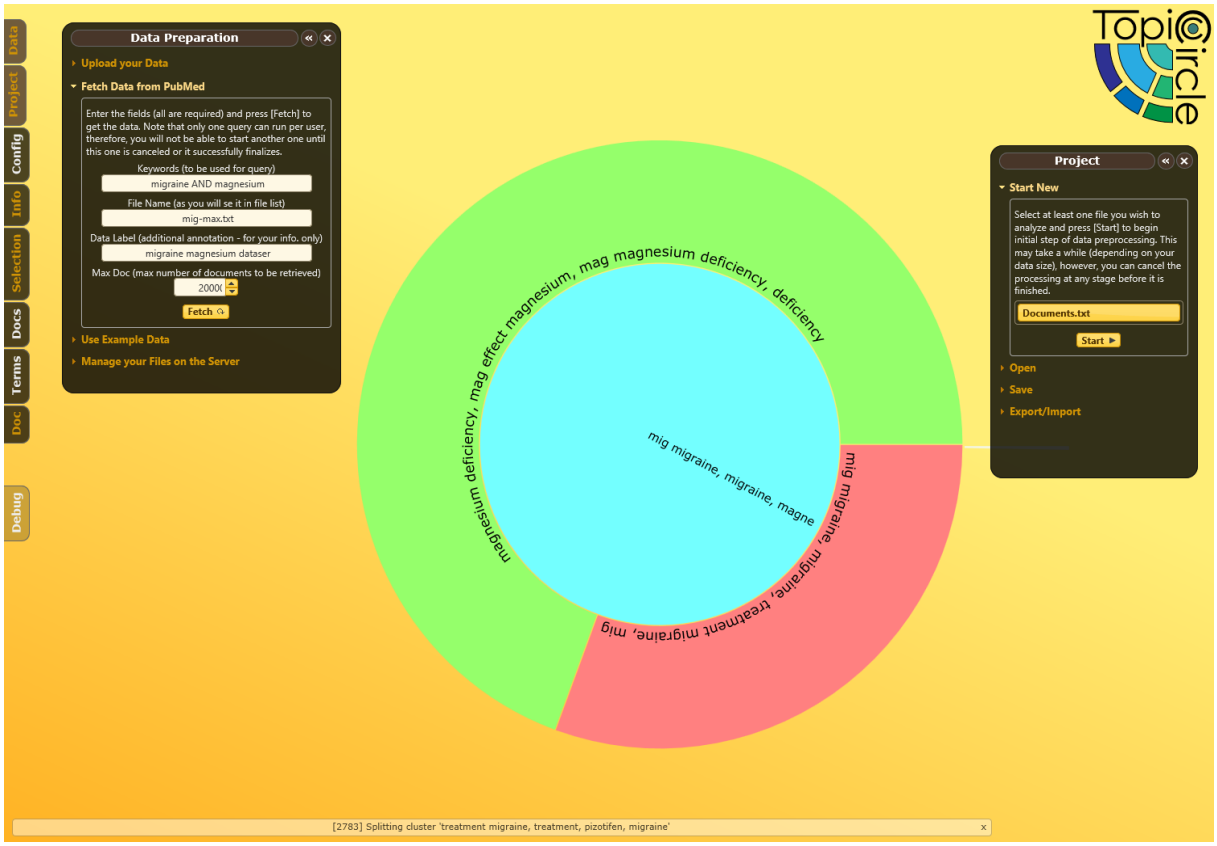



Figure A.2: *Screenshot of the initial phase of topic exploration.* Data preprocessing option menu is opened on the left hand side, classical project option menu on the right hand side while the circle visualizes the first step of splitting the documents into two sets.

Our system has the facility of clustering documents according to their similarity. Similarity between documents can be determined by calculating the cosine of the angle between two documents represented as Bag-of-Words (BoW) vectors (see Section 3.3.1), where BoW approach is used for representing a collection of words from text documents disregarding grammar and word order. Content similarity is measured using the standard TF-IDF (term frequency inverse document frequency) weighting method (see Section 3.3.2). The standard BoW approach is used together with the TF-IDF weighting. BoW representation of text documents is employed for extracting words with similar meaning.

The cosine similarity measure, commonly used in information retrieval and text mining to determine the semantic closeness of two documents where document features are represented using the BoW vector space model, is used to cluster the documents. Cosine similarity values fall within the [0, 1] interval. Value 0 represents extreme dissimilarity, where two documents (a given document and the centroid vector of its cluster) share no common words, while 1 represents the similarity between two exactly identical documents in the BoW representation. For clustering, the standard $k$-means clustering algorithm is used.

The result of interactive top-down document clustering of the migraine-magnesium documents are presented in Figure A.1. At the first level, all the documents are split into one of the two domains (in this case migraine and magnesium) as shown in Figure A.2. At level two, guided by the user, each of the two domains is further split into $k$ sub-clusters, according to the user-selected $k$ parameter. Each of the clusters is described by its most meaningful keywords

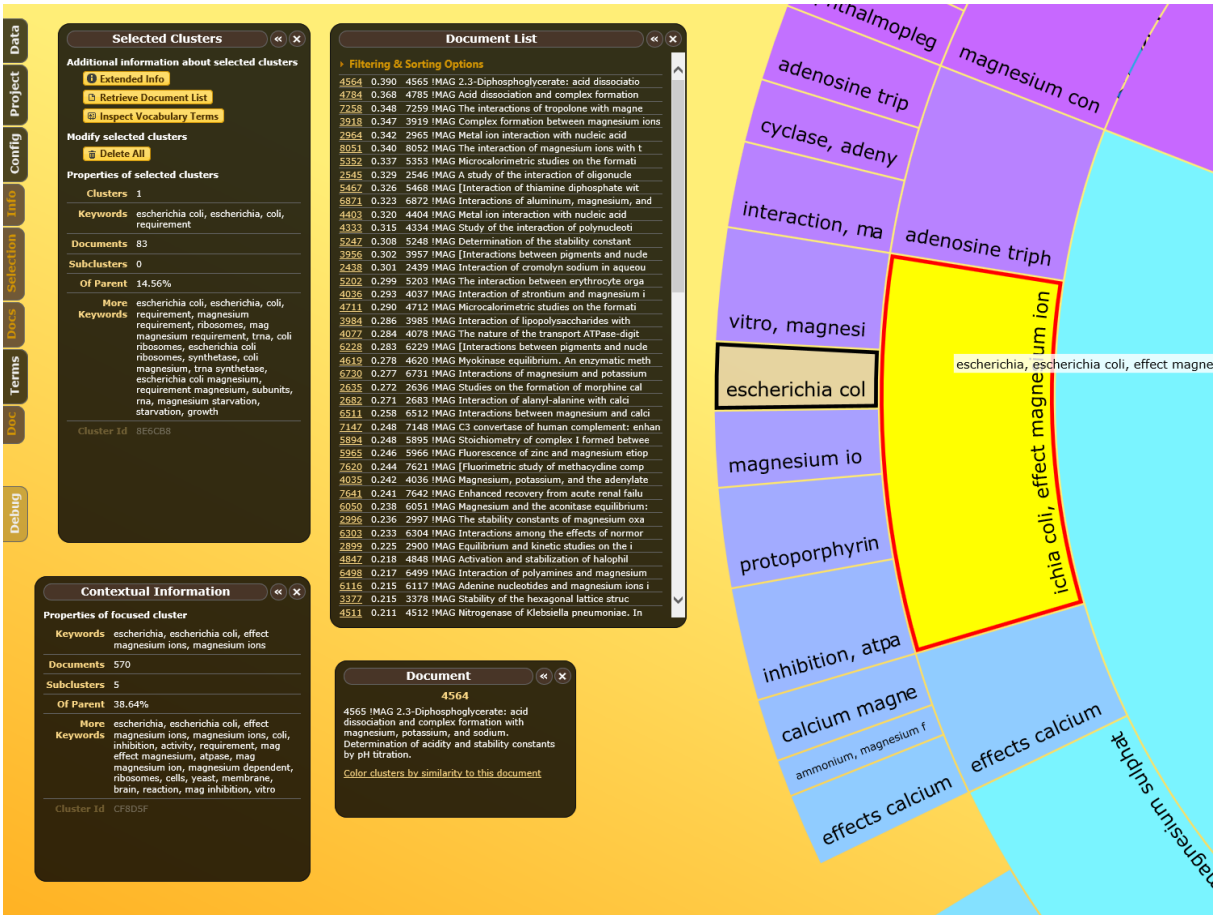



Figure A.3: *An example of TopicCircle's drill-down functionalities and the associated detail information.*

(written inside each cluster and displayed in detail when user moves the mouse over it). Figure A.3 shows some examples of topic drill-down and associated detailed information which can be displayed and which aligns well to the creativity supporting exploration with low threshold and wide walls as described in Section 7.3.3.

The advantages of our new document cluster visualization, e.g., if compared to a well-established semi-automated cluster construction and visualization tool OntoGen (Fortuna et al., 2006), are that (a) the tool is not needed to download, (b) providing much more user friendly environment with especially low threshold for novice users to start exploring their data, and (c) providing wide walls with many different perspectives to the data—e.g., size of the cluster may be based on the number of sub clusters, included documents, or some other calculated property like similarity of the cluster to some query. Similarly is true for the color which may be used in a number of ways to help the user getting a better overview of the data.

Figure A.4 presents an approach of using these properties (in this example we indeed use color) to better visualize cross-domain links which may be present in the data. When the user concentrates on a document in one domain he gets a suggestion of the similar clusters in both domains since all the similar clusters are emphasized with darker color. However, this is only one among many usages of the presented visualization for displaying additional rich cross-context aware information.

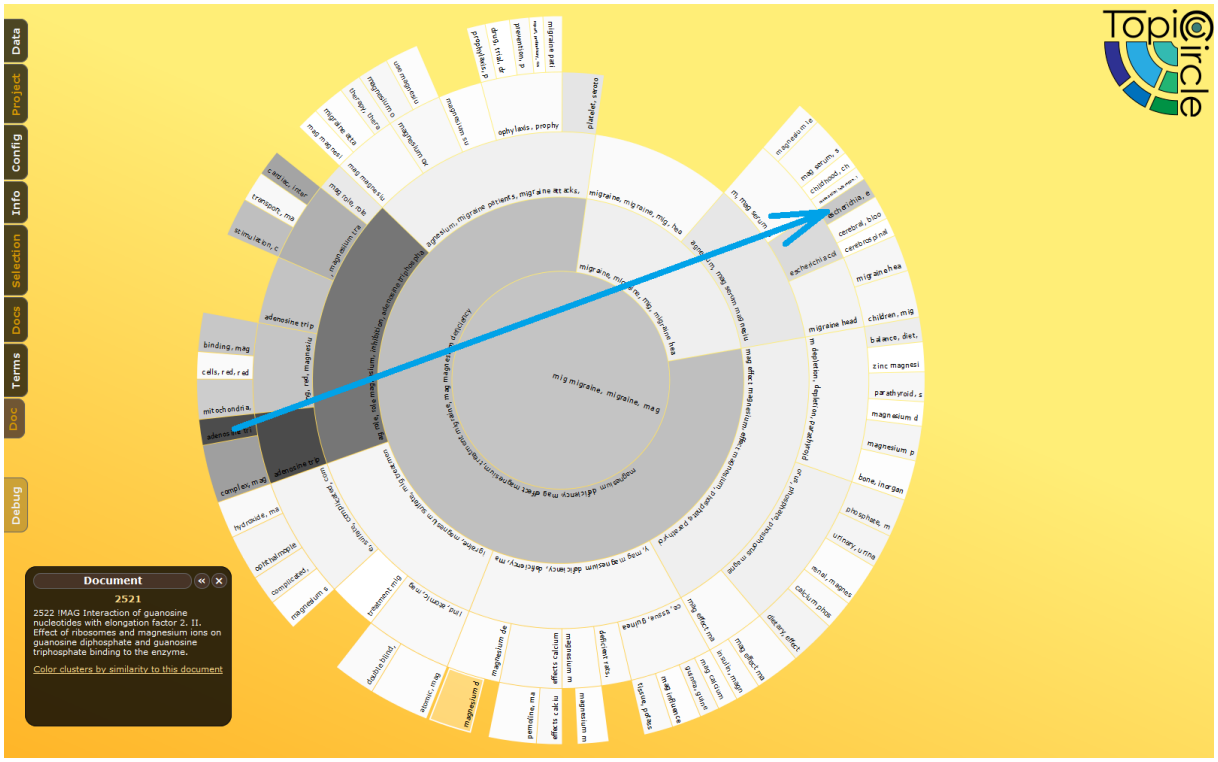



Figure A.4: *An example of using cluster colors to show various information.* In this case cluster color represents the cluster's similarity to a single selected document. The arrow shows similar clusters in two different domains, which can potentially indicate to a novel bisociative link between the two domains.

In terms of cross-context knowledge discovery, the top-down clustering approach enables the user to discover similar document sets within each domain, thus identifying potentially interesting domain subsets for further cross-domain link discovery using our system. Note that in the example presented in Figure A.4, clustering has been performed for each domain separately, therefore not fully demonstrating the potential for cross-domain knowledge discovery. In our past work (Petrič et al., 2012a), however, we have shown that when using clustering on a document set joining documents from both domains, the differences between the cluster identified using similarity measures ($k$-means clustering) and the initial document cluster based its domain can fruitfully serve to identify outlier documents which include an increased number of B-terms and thus a high potential for B-term identification.

# B Bisociative network construction

This section describes the extension of the methodology presented in this thesis onto a new type of problem addressing bisociative network construction. The initial goal is straightforward: to construct an information network from text documents. The input to the procedure consists of text documents (e.g., titles and abstract of scientific documents) from two disparate domains. The output of the procedure is an information network, which could, for example, look like the graph presented in Figure B.1. However, the strong bias towards bisociations leads us to using advanced bridging term identification techniques for detecting important network nodes and relations. The following paragraphs define in detail the input, the output, open issues and sketch the proposed solution.

This thesis focuses—similarly as related work from the literature-mining field—on text documents as the primary data source. Texts are in general considered to be one of the most unstructured data sources available, thus, constructing a meaningful graph of data and knowledge (also named an information network) is even more of a challenge.

We are solving the closed discovery problem, which is the topic of research of this thesis and one of the basic assumptions of our methodology. The selected source text documents are originating from at least two dissimilar domains (M1 and M2 contexts by Koestler's naming or A and C domains according to Swanson and his followers). In this thesis, we always describe the methodology using exactly two domains even though it could be generalized to three or more domains.

In this work, the selected knowledge representation formalism is the so-called *bisociative information network*, called *BisoNet*. The BisoNet representation, as investigated in the BISON project (Online reference [36]) and discussed by Kötter and Berthold (2012) is a graph representation, consisting of labeled nodes and edges (see Figure B.1). The original idea underlying the BISON project was to have a node for every relevant concept of an application domain, captured by terms denoting these concepts, that is, by *named entities*. For example, if the application domain is drug discovery, the relevant (named) entities are diseases, genes, proteins, hormones, chemical compounds etc. The nodes representing these entities are connected if there is evidence that they are related in some way. Reasons for connecting two terms/concepts can be linguistic, logical, causal, empirical, a conjecture by a human expert, or a co-occurrence observed in documents dealing with considered domains. E.g., an edge between two nodes may refer to a document (for example, a research paper) that includes the represented entities. Unlike semantic nets and ontologies, a BisoNet carries little semantics and to a large extend encodes just circumstantial evidence that concepts are somehow related through edges with some probability.

Open issues in BisoNet creation are how to identify entities and relationships in data, especially from unstructured data like text documents; i.e., which nodes should be created from text documents, what edges should be created, what are the attributes with which they are endowed and how should element weights be computed. Among a variety of solutions, this thesis presents the one that answers such questions by optimizing the main criterion of generated BisoNets: maximizing their bisociation potential. Bisociation potential is a feature of a network that informally states the probability that the network contains a bisociation. Thus, we want to be able to generate such BisoNets that contain as many bisociations as possible using the given data sources. In other words, maximizing the bisociation potential of the gener-

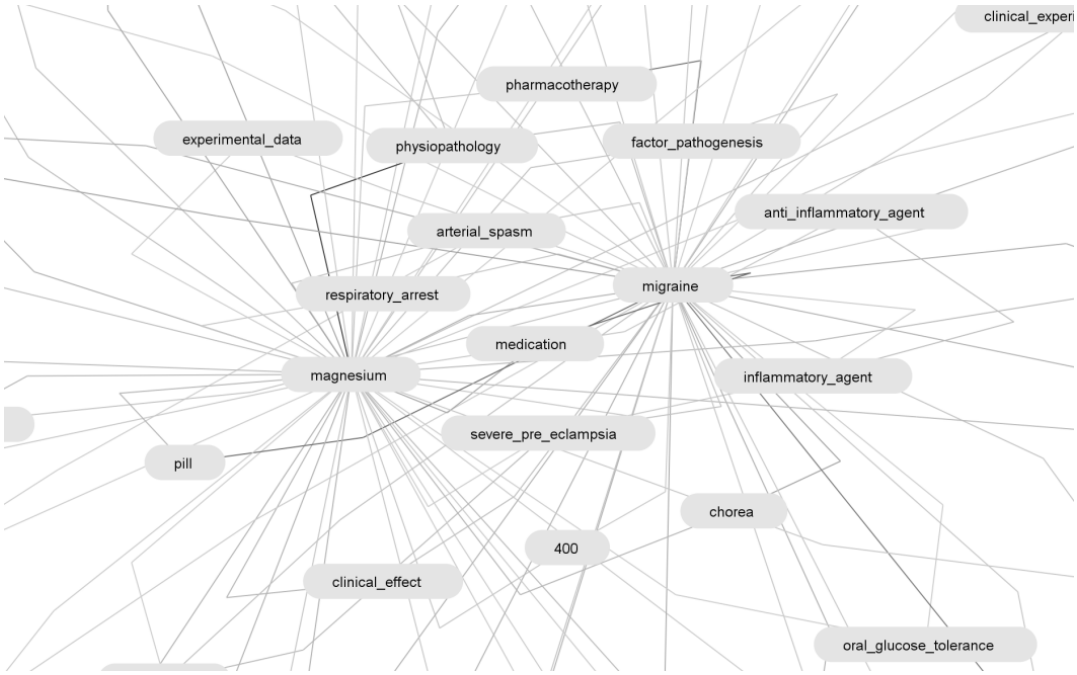



Figure B.1: *Migraine-magnesium example network.* Part of a network created from PubMed articles on migraine and magnesium.

ated BisoNet is our main guidance in developing the methodology for creating BisoNets from text documents.

When creating large BisoNets from texts, we have to address the same two issues as in network creation from any other source: define a procedure for identifying key nodes, and define a procedure for discovering relations among the nodes. However, in practice, a workflow for converting a set of documents into a BisoNet is much more complex than just identifying entities and relations. We have to be able to preprocess text and filter out noise, to generate a large number of entities, evaluate their bisociation potential and effectively calculate various distance measures between the entities. As these tasks are not just conceptually difficult, but also computationally very intensive, great care is needed when designing and implementing algorithms for BisoNet construction.

Our approach to confront the network construction problem is based on developing the following ingredients:

1. Provide basic procedures for automatic text acquisition from different sources of interest on the Web.
2. Employ the state of the art approaches for text preprocessing to extract as much information as available in raw text for the needs of succeeding procedures.
3. Incorporate as much as possible available background knowledge into the stages of text preprocessing and candidate concept detection.
4. Define a candidate concept detection method.
5. Develop a method for relevant bisociative concept extraction from identified concept candidates and perform its evaluation.
6. Select a set of relevant extracted bisociative concepts to form the nodes of a BisoNet.
7. Construct relations between nodes and set their weights according to the Bisociation Index measure published and evaluated by Segond and Borgelt (2009).

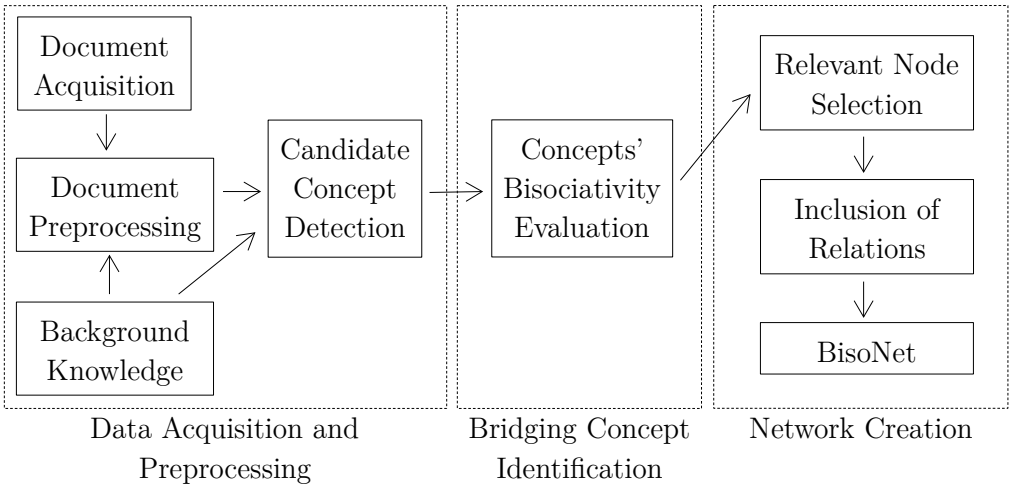



Figure B.2: *Conceptual workflow of the proposed solution for BisoNet creation.*

Figure B.2 illustrates the steps of the methodology proposed by our work. This thesis concentrates mostly on the part of the new methodology for bridging concept evaluation (the middle frame Figure B.2).

## Link detection: Bisociation Index

In this thesis we have developed a term score measure—bisociative score—which is used in the context of network creation for assigning the weights to the nodes. What remains open is the choice for the measure to establish links among nodes and assign the weights to those links. We present two options discussed in more detail by Segond and Borgelt (2009):

- Jaccard index: this similarity coefficient measures the similarity between sample sets. It is defined as the cardinality of the intersection of the sample sets:

$$\text{JaccInx}(vec_x, vec_y) = \frac{|vec_x \cap vec_y|}{|vec_x \cup vec_y|} = \frac{\text{DotProd}(vec_x, vec_y)}{|vec_x|+|vec_y|-\text{DotProd}(vec_x, vec_y)}, \tag{7}$$

  where lengths $|vec_x|$ and $|vec_y|$ are Manhattan lengths of these vectors.
- Bisociation index: it is the similarity measure defined for the purpose of bisociation discovery in the BISON project. It was introduced and explained in more detail by. This measure cannot be expressed by the dot product. Therefore, the following definition uses the notation from Figure 3.1:

$$\text{BisInx}(vec_x, vec_y) = \sum_{i=0}^{M} \left( \sqrt[k]{w_{x:i} w_{y:i}} \left( 1 - \frac{|\tan^{-1}(w_{x:i}) - \tan^{-1}(w_{y:i})|}{\tan^{-1}(1)} \right) \right), \tag{8}$$

  where $M$ is the number of all the entities.

Weighting models that could be used with the presented two indexes:

- *Binary weighting and Jaccard index distance*: Jaccard index was primary defined on sets, therefore the most suitable weighting model to use with it is the binary weighting model (since every vector then represents a set of features).

- *Term frequency weighting and the Bisociation Index distance*: the Bisociation Index was designed with the term frequency weighting in mind, thus it is reasonable to use this combination when determining a weighting model for the Bisociation index.

For the purpose of BisoNet creation, we used term frequency weighting and the Bisociation Index distance as described above. The detailed explanation of a network construction (e.g., the example in Figure B.1) and evaluation is described by Juršič et al. (2012a).

# Author's bibliography

## Publications related to this thesis

The methodology for building and using the LemmaGen lemmatization algorithms (presented in Chapter 5 of this thesis) was published in a conference paper (Juršič et al., 2007) and a journal paper (Juršič et al., 2010).

The methodology of using heuristics for bridging term scoring and ranking (presented in Chapter 6 of this thesis) was published in the book Bisociative Knowledge Discovery (Berthold, ed., 2012). Sections 6.1 and 6.2 along with Appendix B were essential part of the book chapter about constructing information networks from text documents (Juršič et al., 2012a), while Sections 6.3 and 6.4 were the basis for the book chapter about bisociative literature mining by ensemble heuristics (Juršič et al., 2012b).

The following sections list the publications directly related to this thesis.

### Journal paper

- Juršič, M.; Mozetič, I.; Erjavec, T.; Lavrač, N. LemmaGen: Multilingual Lemmatization with Induced Ripple-Down Rules. *Journal of Universal Computer Science* **16**, 1190–1214 (2010).

### Published scientific conference contributions

- Juršič, M.; Mozetič, I.; Lavrač, N. Learning ripple down rules for efficient lemmatization. In: *Proceedings of the 10th International Conference Information Society* **1**, 206–209 (IJS, Ljubljana, Slovenia, 2007).
- Juršič, M.; Lavrač, N.; Mozetič, I.; Podpečan, V.; Toivonen, H. Constructing information networks from text documents. In: *Proceedings of the Workshop on Explorative Analytics of Information Networks at ECML PKDD.* 23–26 (ECML/PKDD Organization Committee, Bled, Slovenia, 2009).
- Juršič, M.; Mozetič, I.; Grčar, M.; Cestnik, B.; Lavrač, N. Identification of concepts bridging diverse biomedical domains. In: *BMC bioinformatics supplements (short paper)* **11**, 4.1–4.2 (BMC Bioinformatics, London, UK, 2010).
- Lavrač, N.; Sluban, B.; Juršič, M. Cross-Domain Literature Mining through Outlier Document and Bridging Concept Detection. In: *Proceedings of the Workshop on Analysis of Complex Networks at ECML PKDD.* 35–47 (ACNE Committee, Barcelona, Spain, 2010).
- Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Evaluating outliers for cross-context link discovery. In: *Artificial intelligence in medicine: proceedings.* Lecture Notes in Computer Science **6747**, 343–347 (Springer, Heidelberg, Germany, 2011).

- Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Cross-domain literature mining: Finding bridging concepts with CrossBee. In: *Proceedings of the 3rd International Conference on Computational Creativity.* 33–40 (University College Dublin, Dublin, Ireland, 2012).
- Juršič, M.; Sluban, B.; Cestnik, B.; Grčar, M.; Lavrač, N. Bridging Concept Identification for Constructing Information Networks from Text Documents. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 66–90 (Springer, Heidelberg, Germany, 2012).
- Sluban, B.; Juršič, M.; Cestnik, B.; Lavrač, N. Exploring the Power of Outliers for Cross-Domain Literature Mining. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 325–337 (Springer, Heidelberg, Germany, 2012).
- Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. Bisociative Literature Mining by Ensemble Heuristics. In: *Bisociative Knowledge Discovery.* Lecture Notes in Computer Science **7250**, 338–358 (Springer, Heidelberg, Germany, 2012).
- Juršič, M.; Cestnik, B.; Urbančič, T.; Lavrač, N. HCI Empowered Literature Mining for Cross-Domain Knowledge Discovery. In: *Proceedings of the International Conference on Human Factors in Computing & Informatics.* Lecture Notes in Computer Science **7947**, 124–135 (SouthCHI Committee, Maribor, Slovenia, 2013).

## Publications not related to this thesis

### Journal paper

- Grčar, M.; Podpečan, V.; Juršič, M.; Lavrač, N. Efficient visualization of document streams. *Discovery Science.* Lecture Notes in Computer Science **6332**, 174–188 (2010).

### Published scientific conference contributions

- Juršič, M.; Lavrač, N.; Fuzzy clustering of documents. In: *Proceedings of the 11th International Conference Information Society* **1**, 178–181 (IJS, Ljubljana, Slovenia, 2008).
- Podpečan, V.; Juršič, M.; Žáková, M.; Lavrač, N. Towards a service-oriented knowledge discovery platform. In: *Proceedings of the Workshop on Third generation data mining: towards service-oriented knowledge discovery at ECML PKDD.* 25–38 (SoKD'10 Organization Committee, Barcelona, Spain, 2009).

# Biography

Matjaž Juršič was born on July 19, 1980 in Novo mesto, Slovenia. He attended primary and secondary school in Novo mesto where he finished gymnasium focusing in natural sciences and mathematics. During his primary and secondary education, he attended many national competitions in the area of mathematics, physics and logics for which he won several bronze, silver and gold awards. In 1999, he started his studies at the *Faculty of Computer and Information Science*, University of Ljubljana, Slovenia. He was enrolled in a 9-semester BSc program in the area of *Software Engineering*. During his studies, he received two Dean's praises for *exemplary study performance* and held the government scholarship for talented students called *Zoisova štipendija*. In this period, he was part-time employed by companies working on software projects for state budget management, stock brokerage support, advanced game analysis and modeling for the gaming industry, and others, where he gained valuable experience in advanced technologies and large-scale software engineering projects. He finished his studies and defended the BSc thesis in June 2007 under the supervision of Prof. Blaž Zupan and co-supervision of Prof. Nada Lavrač. He received the *University of Ljubljana Prešeren Award*, which is the highest university award for the best BSc theses in Slovenia.

In the Fall of 2007 he started his graduate studies at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia. He is enrolled in the PhD program entitled *New Media and e-Science* under the supervision of Prof. Nada Lavrač and Prof. Bojan Cestnik. During his PhD studies, he was research assistant at the Department of Knowledge Technologies at the Jožef Stefan Institute, Ljubljana, Slovenia and was actively involved in several European projects with the major focus on the European, $7^{\text{th}}$ Framework Programme project BISON (BISOciation Networks for Creative Information Discovery). His research is in the field of data mining in particular text mining and includes the study, development and application of different data mining algorithms. His current research—presented in this thesis—aims at developing a methodology for assisting scientists to get a better insight into their domain by inspecting and overlaying some other, previously uninspected, domains, which may hold some new answers to scientists' questions.