The Computer Journal Advance Access published June 7, 2012

doi:10.1093/comjnl/bxs058

A Methodology for Mining Document-Enriched Heterogeneous Information Networks

MIHA GRČAR*, NEJC TRDIN AND NADA LAVRAČ

Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
**Corresponding author: miha.grcar@ijs.si*

The paper presents a new methodology for mining heterogeneous information networks, motivated by the fact that, in many real-life scenarios, documents are available in heterogeneous information networks, such as interlinked multimedia objects containing titles, descriptions and subtitles. The methodology consists of transforming documents into bag-of-words vectors, decomposing the corresponding heterogeneous network into separate graphs, computing structural-context feature vectors with PageRank, and finally, constructing a common feature vector space in which knowledge discovery is performed. We exploit this feature vector construction process to devise an efficient centroid-based classification algorithm. We demonstrate the approach by applying it to the task of categorizing video lectures. We show that our approach exhibits low time and space complexity without compromising the classification accuracy. In addition, we provide a qualitative analysis of the results by employing a data visualization technique.

Keywords: text mining; heterogeneous information networks; data fusion; classification; centroid-based classifier; diffusion kernels; data visualization

Received 27 November 2011; revised 18 March 2012
Handling editor: Einoshin Suzuki

1. INTRODUCTION

In many real-life data mining scenarios involving document analysis, the accompanying data can be represented in the form of heterogeneous information networks. We address this data analysis setting by proposing a methodology that takes advantage of both types of data by employing techniques from text, graph and heterogeneous network mining, and data fusion.

Text mining [1], which aims at extracting useful information from document collections, is a well-developed field of computer science. In the last decade, text mining research was driven by the growth of the size and the number of document collections available in companies and public organizations and especially by the rapid growth of the world's largest source of semi-structured data, the Web. Text mining is used to extract knowledge from document collections by using data mining, machine learning, natural language processing and information retrieval techniques. The data preprocessing step plays a crucial role in text mining. In this step, documents are transformed into feature vectors according to a certain representational model and then processed with the available machine learning algorithms that can handle sparse vector collections with high feature dimensionality and continuous or binary features [such as k-nearest neighbors (k-NN), k-means clustering, support vector machine (SVM) and Naive Bayes).

Naturally, not all data come in the form of documents. A lot of recent research involves analyses of data from networked systems where individual agents or components interact with other components, forming large, interconnected and heterogeneous networks. In short, such networks are called heterogeneous information networks [2]. Some examples of heterogeneous information networks are communication and computer networks, transportation networks, epidemic networks, social networks, e-mail networks, citation networks, biological networks and also the Web (with the emphasis on its structure). Such networks can also be formed from data in relational databases and ontologies. In heterogeneous information networks, knowledge discovery is usually performed by resorting to social network analysis [3], link analysis [4, 5] and other dedicated approaches to mining heterogeneous information networks [2].

In many real-life scenarios, information networks also contain documents. This results in heterogeneous information networks in which (some) objects are associated with corresponding sets of textual documents. Examples of such networks include the Web (interlinked HTML documents), multimedia repositories (interlinked multimedia descriptions, subtitles, slide titles, etc.), social networks of professionals (interlinked CVs), citation networks (interlinked publications) and even software code (heterogeneously interlinked code comments) [6]. The abundance of such document-enriched networks motivates the development of a new methodology that joins the two worlds, text mining and mining of heterogeneous information networks, and handles the two types of data in a common data mining framework.

The methodology presented in this paper is based on decomposing a heterogeneous network into (homogeneous) graphs, computing feature vectors with Personalized PageRank (P-PR) [7], and constructing a common vector space in which knowledge discovery is performed. Heterogeneity is taken into account in this final step of the methodology where all the structural contexts and the text documents are 'fused' together. We demonstrate the methodology by employing it for the task of categorizing video lectures hosted on VideoLectures.net,[1] one of the world's largest academic video hosting Web portals.

This work extends the paper presented at Discovery Science 2011 [8] in that it (i) extends the related work section (most notably with a discussion on data fusion techniques), (ii) discusses network decomposition on a toy example (Section 3.1), (iii) includes a discussion on data mining tasks that can be addressed and tools that can be used when using the proposed methodology (Section 3.5), (iv) presents additional text preprocessing experiments and the results (Section 4.2), (v) provides additional insights into weights computation (Section 4.3) and (vi) provides qualitative explanations of the classification results through vector space visualization (Section 7).

The paper is structured as follows. In Section 2, we present the related work. In Section 3, the proposed methodology is presented. Section 4 presents the experimental results of employing the methodology in a real-life video-lecture categorization task. We present a new centroid classifier that exploits the properties of the presented feature vector construction process in Section 5 and discuss its efficiency in terms of reduced time and space complexity in Section 6. In Section 7, we provide a qualitative analysis of the results by employing a data visualization technique. Section 8 concludes the paper and briefly discusses plans for further work.

2. RELATED WORK

The related work comes from the fields of text mining, graph and heterogeneous network mining and data fusion. In the following subsections, we discuss some of the related approaches from these research areas.

2.1. Text, graph and heterogeneous network mining

Text mining employs basic machine learning principles, such as supervised and unsupervised learning [9], to perform tasks such as text categorization (also known as 'text classification'), topic ontology construction [10], text corpora visualization [11] and user profiling [12, 13]. Most text mining approaches rely on the bag-of-words (BOW) vector representation of documents [14].

Text categorization is a widely researched area due to its value in real-life applications such as indexing of scientific articles, patent categorization, spam filtering and Web page categorization [15]. In [16], the authors present a method for categorizing Web pages into the Yahoo! taxonomy.[2] They employ a set of Naive Bayes classifiers, one for each category in the taxonomy. For each category, the corresponding classifier gives the probability that the document belongs to this category. A similar approach is presented in [17], where Web pages are being categorized into the DMoz taxonomy.[3] Each category is modeled with the corresponding centroid BOW vector and a document is categorized simply by computing the cosine similarity between the document's BOW vector and each of the computed centroids. Apart from Naive Bayes [9] and centroid-based classifiers [18, 19], an SVM [20] is also a popular classifier for text categorization.

In the areas of graph and heterogeneous network mining, a different family of data analysis algorithms was devised. Important building blocks are the techniques that can be used to assess the relevance of an object (with respect to another object or a query) or the similarity between two objects in a network. Some of these techniques are: spreading activation [21], hubs and authorities (HITS) [22], PageRank and P-PR [7], SimRank [23] and diffusion kernels (DK) [24]. These methods are extensively used in information-retrieval systems. The general idea is to propagate 'authority' from 'query nodes' into the rest of the graph or heterogeneous network, assigning higher ranks to more relevant objects.

ObjectRank [25] employs global PageRank (importance) and P-PR (relevance) to enhance the keyword search in databases. Specifically, the authors convert a relational database of scientific papers into a graph by constructing the data graph (interrelated instances) and the schema graph (concepts and relations). To speed up the querying process, they precompute P-PR vectors for all possible query words. Stoyanovich *et al.* [26] present a ranking method called EntityAuthority which defines a graph-based data model that combines Web pages, extracted (named) entities and ontological structure in order to improve the quality of keyword-based retrieval of either pages or entities. The authors evaluate three conceptually different

[1] http://videolectures.net/.

[2] http://dir.yahoo.com/.

[3] http://www.dmoz.org/.

methods for determining relevant pages and/or entities in such graphs. One of the methods is based on mutual reinforcement between pages and entities, while the other two approaches are based on PageRank and HITS, respectively.

In [5], the authors propose a link-based classification algorithm for homogeneously interlinked data. In their approach, they describe each object (represented as a node in the graph) with object properties (i.e. BOW vector) and, in addition, with categorization information observed in its immediate neighbors. They employ logistic regression for the classification task and fuse the two different types of data (i.e. textual and structural) in a probabilistic framework. To get the initial information about categories, they simply perform text classification, discarding the structural information. After that, they perform an iterative refinement process in which they describe each object with the categorization information from its immediate neighborhood (in addition to its BOW representation) and re-categorize the object. The authors show that the classification performance is improved when compared with a simple content-only classification. Their methodology bears some similarity to our work in that it fuses textual and structural data to improve the performance. It also employs an iterative process to make use of distant relationships in the graph (we achieve the same goal with PageRank). The main difference is that our approach is not limited to classification tasks (even though we present a classification case study in Section 4) or to a particular machine learning algorithm. In addition, we explicitly discuss how structural heterogeneity can be taken into account.

Zhu and Ghahramani [27] present a method for transductive learning that first constructs a graph from the data and then propagates labels along the edges to label (i.e. classify) the unlabeled portion of the data. The graph regularization framework proposed by Zhou and Schölkopf [28] can also be employed for categorization. However, most of these methodologies are devised for graphs rather than heterogeneous networks. GNetMine [29] is built on top of the graph regularization framework but takes the heterogeneity of the network into account and, consequently, yields better results. CrossMine [30] is another system that exploits heterogeneity in networks. It constructs labeling rules while propagating labels along the edges in a heterogeneous network. These approaches clearly demonstrate the importance of handling different types of relations and/or objects in a network separately.

2.2. Data fusion

Rather than devising a methodology for a specific task, domain or network type, we resort to data fusion to take the heterogeneity into account. Data fusion refers to combining different types of data (media) in order to perform a data analysis task. It is widely studied in the field of multimedia analysis where data is obtained from different modalities such as video, audio, text and motion. An extensive survey was presented by Atrey *et al.* [31]. According to the authors of the survey, data fusion can either be performed on the feature level (early fusion) or on the decision level (late fusion). Feature-level fusion refers to combining features or feature vectors in the data transformation process. Propositionalization [32], an approach well known from inductive logic programming [33, 34] and relational data mining [35], belongs to this category of data fusion techniques. It refers to the process of converting a relational knowledge representation into a propositional feature vector representation. An extensive survey of propositionalization approaches can be found in [32]. Feature-level fusion is advantageous in that the employed training algorithm can study correlations between features, which is not possible with the decision-level approaches. Decision-level fusion refers to solving the task for each modality separately and then combining the results through a fusion model (e.g. [5, 36]). One of the simplest late fusion approaches is majority voting which is often used in ensembles of machine learning models [36]. If the data mining approach is based on the probabilistic framework (e.g. Naive Bayes, logistic regression, maximum entropy model), it is possible to perform fusion by using Bayesian inference (e.g. [5]). The decision-level approaches have the advantages of (i) being more scalable (several smaller models are built instead of one large model), (ii) allowing the use of different models in the inference phase and (iii) providing a uniform representation of data (i.e. a set of decisions) that is further processed with a fusion model. We additionally point out that data fusion can also be performed on the similarity level, which corresponds to combining similarities between objects over different modalities. The kernel-based fusion methods fall into this category (e.g. [37–39]). The most obvious advantage of this type of fusion, similarly to the decision-level approaches, is that the fusion model deals with a uniform data representation (i.e. a set of similarities). One of the disadvantages is that only the kernel or similarity-based data analysis algorithms can be employed after the fusion process.

In our work, we employ a linearly weighted feature-level fusion approach. The term 'linearly weighted fusion' refers to combining data from different modalities in a linear fashion (e.g. linear combination of similarity scores [37–39]). Although the basic idea is simple, the weights need to be adjusted in the fusion process, which is not a trivial task. Note that performing early fusion makes our methodology relatively flexible in terms of which data analysis algorithm can be employed and what kind of data mining task can be addressed. We counter the heterogeneity issue by limiting ourselves to textual data and networks and by devising a method of representing network nodes in a similar fashion to texts (see Section 3.3). Furthermore, for classification tasks, we counter the scalability issue by devising an efficient centroid-based classifier (see Section 5).

Even though in this paper, we deal with feature vectors rather than kernels, the kernel-based data fusion approach presented by Lanckriet *et al.* [39] is closely related to our work. In their

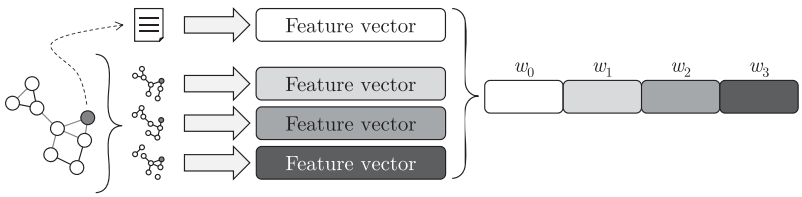


FIGURE 1. The proposed methodology for transforming a heterogeneous information network and the corresponding text documents into a joint feature vector format. Feature vector construction is shown for one particular object.

method, the authors propose a general-purpose methodology for kernel-based data fusion. They represent each type of data with a kernel and then compute a weighted linear combination of kernels (which is again a kernel). The linear-combination weights are computed through an optimization process called Multiple Kernel Learning (MKL) [37, 38], which is tightly integrated into the SVM's margin maximization process. The authors define a quadratically constrained quadratic program in order to compute the support vectors and linear-combination weights that maximize the margin. In the paper, the authors employ their methodology for predicting protein functions in yeast. They fuse together six different kernels (four of them are DK based on graph structures). They show that their data fusion approach outperforms the SVM trained on any single type of data, as well as the previously advertised method based on Markov random fields. In the approach employed in our case study, we do not employ MKL but rather a stochastic optimizer called Differential Evolution (DE) [40], which enables us to directly optimize the target evaluation metric.

3. PROPOSED METHODOLOGY

In this work, we employ several well-established approaches from the fields of text, graph and heterogeneous network mining. The main contribution is a general-purpose framework for feature vector construction, establishing an analogy between BOW vectors and P-PR. In contrast to the approaches that use authority propagation algorithms for label propagation, we employ P-PR for feature vector construction. This allows us to fuse text documents and different types of structural information together and thus take the heterogeneity of the network into account. This section presents the proposed methodology for transforming a heterogeneous information network into a feature vector representation.

We assume that we have a heterogeneous information network that can be decomposed into several homogeneous undirected graphs with weighted edges, each representing a certain type of relationship between objects of interest (see Section 3.1). We also assume that several objects of interest are associated with text documents, which is not mandatory for the methodology to work.

Figure 1 illustrates the proposed feature vector construction process. Text documents are first transformed into feature vectors (i.e. BOW vectors) as briefly explained in Section 3.2. In addition, each graph (resulting from the decomposition of the given heterogeneous network) is transformed into a set of feature vectors. We employ P-PR for this purpose as explained in Section 3.3. As a result, each object is now represented as a set of feature vectors (i.e. one for each graph and one for the corresponding text document). Finally, the feature vectors describing a particular object are combined into a single concatenated feature vector as discussed in Section 3.4. We end up with a typical machine learning setting in which each object, representing either a labeled or unlabeled data instance, is represented as a sparse feature vector with continuous feature values. The concatenated feature vectors are ready to be employed for solving various data mining tasks as briefly discussed in Section 3.5.

3.1. Decomposing heterogeneous networks into graphs

To illustrate how a heterogeneous network can be decomposed into separate undirected graphs, let us consider a toy heterogeneous network of scientific publications shown in Fig. 2.

Let us now assume that the publications are the objects of interest which means that they will be represented as nodes in the resulting graphs. The heterogeneous network shown in Fig. 2 interlinks the publications in several different direct and indirect ways:

(i) Two publications are interlinked (indirectly, through authors) if they were published by at least one common author (resulting in the 'publication-author-publication' or 'P-A-P' graph).
(ii) Two publications are interlinked (indirectly, through proceedings) if they were published in the same proceedings (resulting in the 'publication-proceedings-publication' or 'P-P-P' graph).
(iii) Two publications are interlinked if one cites the other (resulting in the 'publication-cites-publication' or 'P-c-P' graph).

The three resulting graphs are shown in Fig. 3. In this toy example, most of the edges in the graphs are weighted equally (their weight is 1). The only exception is the P-A-P graph where the weights correspond to the number of authors two lectures

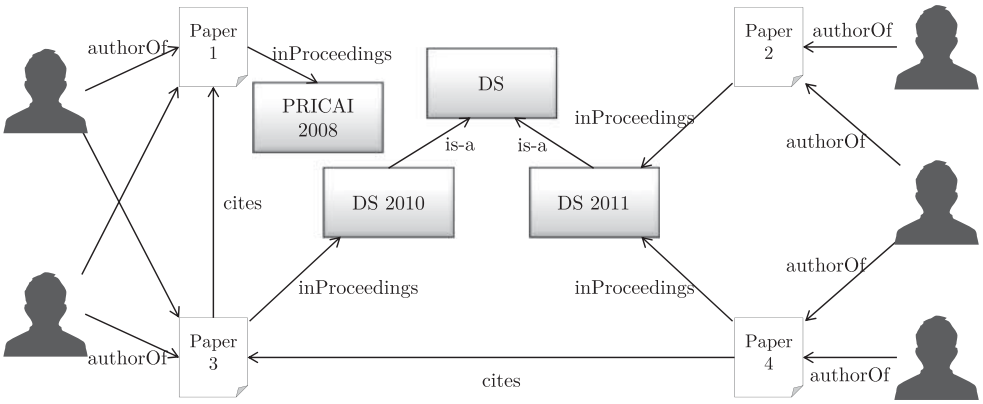


FIGURE 2. Toy heterogeneous network of scientific publications.

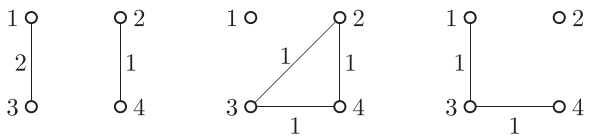


FIGURE 3. The three graphs resulting from the toy network decomposition: the P-A-P, P-P-P and P-c-P graph, respectively.

have in common. Note that in the P-P-P graph, we interlink two publications even if they were not published in the same proceedings but rather in the same family of proceedings (e.g. Discovery Science proceedings regardless of the year). At this point, we could use higher edge weights for pairs of publications that were published in the same proceedings and lower for those that were not published in the same year, but we avoided this in this toy example for the sake of simplicity.

There are no strict rules about which indirect links to take into account when decomposing a network and how to assign the weights to the edges in the resulting graphs. The decomposition procedure should thus follow intuitions and should be evaluated in the context of the data mining task at hand.

3.2. Constructing feature vectors from text documents

To convert text documents into their BOW representations, we follow a typical text mining approach [1, 14]. The documents are tokenized, stop words are removed and the word tokens are stemmed [41]. N-grams (i.e. word sequences of up to a particular length) [42] can be considered in addition to unigrams. Infrequent words are removed from the vocabulary. Next, BOW vectors are computed and normalized in the Euclidean sense. Finally, from each vector, the terms with the lowest weights are removed (i.e. their weights are set to 0).

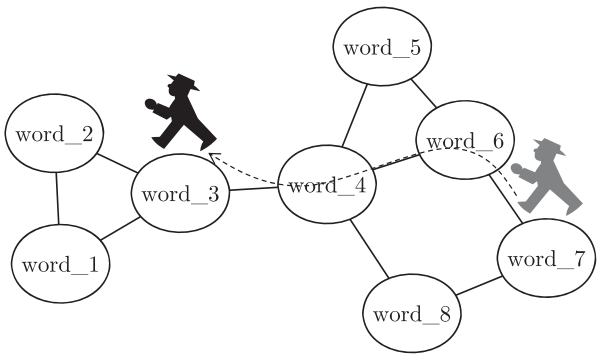
Several concrete settings of BOW vector construction are presented and evaluated in the context of our case study in Section 4.2.

3.3. Using P-PR for computing structural-context feature vectors

For computing the structural-context feature vectors, we employ P-PR [7]. P-PR is essentially a stationary distribution of a Markov chain [43] in which the states are the graph nodes and the transitions encapsulate both the edges between the nodes and the 'teleport behaviour' of a random walker. 'Personalized' in this context refers to using a predefined set of nodes as the starting nodes for random walks. In our case, P-PR is run from a single source node representing the object for which we want to compute the feature vector. At each node, the random walker decides whether to teleport back to the source node (this is done with the probability $1 - d$ where d is the so-called damping factor) or to continue the walk along one of the edges. The probability of choosing a certain edge is proportional to the edge's weight compared with the weights of the other edges connected to the node. In effect, for a selected source node i in a given graph, P-PR computes a vector of probabilities with components $\mathrm{PR}_i(j)$, where j is a node in the graph. $\mathrm{PR}_i(j)$ is the probability that a random walker starting from node i will be observed at node j at an arbitrary point in time.

Let us suppose that each node is assigned a single random word and that the random walker is 'writing down' words that it encounters along the way (this principle is illustrated in Fig. 4). It is not difficult to see that a structural-context feature vector computed with P-PR is in fact the l^1-normalized (i.e. the sum of vector components is equal to 1) term-frequency BOW vector representation of this random text document. This is also one of the main reasons for employing P-PR over other methods for computing structural features: it allows an interpretation that relates P-PR vectors to BOW and thus nicely fits into the existing text mining frameworks.

In text mining, cosine similarity is normally used to compare BOW vectors. Cosine similarity is equal to computing the dot product of two feature vectors, provided that the two vectors are normalized in the Euclidean sense (i.e. their l^2-norm is



"word_7 word_6 word_4 word_3 ..."

$\mathbf{v}_7 = (0.03, 0.03, 0.06, 0.18, 0.09, 0.19, 0.27, 0.15)$

FIGURE 4. The 'random writer' principle: the random walker is 'writing down' words that it encounters along the way. This is similar to generating random texts with a language model.

equal to 1). Since we use the dot product as the similarity measure in the proposed framework, the P-PR vectors need to be normalized in the Euclidean sense in order to conform to the analogy with text mining. Given a P-PR vector $\mathbf{v}_i = \langle \mathrm{PR}_i(1), \mathrm{PR}_i(2), \ldots, \mathrm{PR}_i(n) \rangle$ for object i, the corresponding structural-context feature vector $\mathbf{v}_i'$ is thus computed as

$$\mathbf{v}_i' = \|\mathbf{v}_i\|^{-1} \langle \mathrm{PR}_i(1), \mathrm{PR}_i(2), \ldots, \mathrm{PR}_i(n) \rangle.$$

3.4. Combining feature vectors

The final step of the proposed methodology is to combine the computed BOW and structural-context feature vectors describing a particular object into a single concatenated feature vector. To explain the theoretical background, we first establish a relationship between feature vectors and linear kernels. Suppose that, for a given object i, the concatenated feature vector is obtained by 'gluing together' m feature vectors, i.e. $m-1$ structural feature vectors and one BOW feature vector corresponding to the text document describing the object. For a given set of n objects, let us denote the m sets of feature vectors by $\mathbf{V}_1, \ldots, \mathbf{V}_m$, where each $\mathbf{V}_k$ is a matrix with n rows, in which ith row represents the feature vector corresponding to object i. The corresponding kernels, one for each set of feature vectors, are computed as $\mathbf{K}_k = \mathbf{V}_k \mathbf{V}_k^{\mathrm{T}}$.

This relationship is important because it relates our data fusion approach to MKL which can also be employed for data fusion [39]. In MKL, multiple kernels are combined into a weighted convex combination of kernels which yields a combined kernel $\mathbf{K}_\Sigma = \sum_k \alpha_k \mathbf{K}_k$, $\sum_k \alpha_k = 1$, $\alpha_k \geq 0$. Analogously, we derived the following equation which shows how the above weights α_k can be used to combine feature vectors:

$$\mathbf{V}_\Sigma = \sqrt{\alpha_1}\mathbf{V}_1 \oplus \sqrt{\alpha_2}\mathbf{V}_2 \oplus \cdots \oplus \sqrt{\alpha_m}\mathbf{V}_m.$$

In this equation, $\oplus$ represents the concatenation of matrix rows. To prove that the resulting combined vectors correspond to the kernel $\mathbf{K}_\Sigma$, we have to show that $\mathbf{V}_\Sigma \mathbf{V}_\Sigma^{\mathrm{T}} = \mathbf{K}_\Sigma$:

$$\begin{aligned}\mathbf{V}_\Sigma \mathbf{V}_\Sigma^{\mathrm{T}} &= (\sqrt{\alpha_1}\mathbf{V}_1 \oplus \cdots \oplus \sqrt{\alpha_m}\mathbf{V}_m) \\ &\quad \cdot (\sqrt{\alpha_1}\mathbf{V}_1 \oplus \cdots \oplus \sqrt{\alpha_m}\mathbf{V}_m)^{\mathrm{T}} \\ &= \sum_k \alpha_k \mathbf{V}_k \mathbf{V}_k^{\mathrm{T}} = \sum_k \alpha_k \mathbf{K}_k = \mathbf{K}_\Sigma.\end{aligned}$$

Note that the weights w_k from Fig. 1 directly correspond to the above weights, i.e. $w_k = \sqrt{\alpha_k}$.

In general, the weights α_k can be set in several different ways. We can resort to trial-and-error or a greedy heuristic. We can also consider 'binary weights' and either include or exclude a certain type of vectors. Employing MKL is also an option. In the presented case study (see Section 4), we employ a stochastic optimizer and directly optimize the target evaluation metric.

3.5. Employing machine learning algorithms

The constructed feature vectors can be employed in both supervised (e.g. classification and ranking) and unsupervised (e.g. clustering) machine learning scenarios [9].

Note that from the feature vectors, we can easily compute a kernel (see Section 3.4), and so it is possible to employ any kernel or distance-based machine learning algorithm. Such algorithms include (but are not limited to) the SVM and its variants, k-NN, k-means clustering, agglomerative clustering and multi-dimensional scaling (MDS).

On the other hand, one of the key advantages of our methodology is that we compute feature vectors rather than kernels. It is thus easy to employ available machine learning tools that normally take a set of feature vectors as input. For example, SVM-Light [20], a well-known implementation of the SVM, and its variants (e.g. SVM-Struct for structured outputs, SVM-Multiclass for multi-class predictions, SVM-Rank for ranking)[4] can be used directly. Another advantage of the feature vector representation is the ability to employ feature weighting and feature selection techniques [44].

Furthermore, various ready-to-use data mining platforms can be used as well. Data mining platforms such as Orange [45], Knime [46], Weka [47] and RapidMiner [48] provide the user with a collection of data mining and machine learning components that can be integrated into an executable workflow with a visual workflow editor. These systems have been recently enhanced with the capability of handling sparse datasets and extended with text mining components that can process BOW-like vectors provided by our methodology.

In the case study presented in the following section, we demonstrate the use of the methodology in a classification scenario. In the evaluation process, we employ three different classification algorithms: SVM-Multiclass, k-NN and centroid

[4] http://svmlight.joachims.org/.

classifier (see Section 4.3). Furthermore, we take advantage of our feature vector construction process to devise an efficient centroid-based classification algorithm (see Section 5). Later on, in Section 7, we present another case study in which we employ a visualization technique based on k-means clustering and MDS to provide qualitative explanations of the classification results.

4. VIDEOLECTURES.NET CATEGORIZATION CASE STUDY

The task in the VideoLectures.net case study was to develop a method that can be used to assist the categorization of video lectures hosted by one of the largest video lecture repositories, VideoLectures.net. This functionality was required due to the rapid growth of the number of hosted lectures (150–200 lectures are added each month) as well as due to the fact that the categorization taxonomy is rather fine-grained (129 categories in the provided database snapshot). We evaluated our methodology in this case study, confronting it with a typical text mining approach and an approach based on DKs.

4.1. Dataset

The VideoLectures.net team provided us with a set of 3520 English lectures, 1156 of which were manually categorized. Each lecture is described by a title, while 2537 lectures also have a short description. The lectures are categorized into 129 categories. Each lecture can be assigned to more than one category (on average, a categorized lecture is categorized into 1.26 categories). There are 2706 authors in the dataset, 219 events at which the lectures were recorded and 62 070 portal users' click streams.

From this data, it is possible to represent lectures, authors, events and portal users in a heterogeneous information network. In this network, authors are linked to lectures, lectures are linked to events and portal users are linked to lectures that they viewed. From the available data, we derived the following textual and structural information about video lectures:

(i) Each lecture is assigned a *textual document* formed of the title and, if available, extended with the corresponding lecture description (abstract).
(ii) The *structural information* of this heterogeneous network is represented in the form of three homogeneous graphs in which nodes represent individual video lectures:
   (a) *Same-event graph*. Two nodes are linked if the two corresponding lectures were recorded at the same event. The weight of a link is always 1.
   (b) *Same-author graph*. Two nodes are linked if the two corresponding lectures were presented by the same author or authors. The weight of a link is proportional to the number of authors the two lectures have in common.
   (c) *Viewed-together graph*. Two nodes are linked if the two corresponding lectures were viewed together by the same portal user or users. The weight of a link is proportional to the number of users that viewed both lectures.

4.2. Results of text mining and DKs

We first performed a set of experiments on textual data only, by following a typical text mining approach. In addition, we employed DKs [24] for classifying lectures according to their structural contexts.

In the text mining experiments, each lecture was assigned a text document formed of the title and, if available, extended with the corresponding description. We represented the documents as normalized BOW vectors. In the first set of experiments, we tested several different BOW construction settings. We varied the type of weights (TF or TF-IDF), maximum n-gram length (n), minimum required term frequency (min-freq) and cut-off percentage (cut-off). We employed a Centroid Classifier (for details, see Section 5.1) in the first set of experiments and performed 10-fold cross-validation on the manually categorized lectures. We performed flat classification as suggested in [17]. We measured the classification accuracy on the top 1, 3, 5 and 10 categories predicted by the classifier.

The results are given in Table 1. We can see that the TF-IDF weighting scheme outperforms the TF weighting scheme, that taking bigrams into account in addition to unigrams improves the performance, and that it is beneficial to process only those terms that occur in the document collection at least twice. We therefore use Setting 5 in all our subsequent experiments involving BOW.

In the next set of experiments, we employed two additional classifiers for the text categorization task: SVM and k-NN. In the case of the SVM, we applied SVM-Multiclass [20] for which we set ε (the termination criterion) to 0.1 and C (the trade-off between error and margin) to 5000. In the case of k-NN, we set k (the number of neighbors) to 20. We used the dot product (i.e. the cosine similarity) to compute the similarity between feature vectors.

In addition to the text mining experiments (using only the textual information), we also computed DK of the three graphs (we set the diffusion coefficient β to 0.0001). For each kernel separately, we employed the SVM and k-NN in a 10-fold cross-validation setting. The two classifiers were configured in the same way as before in the text mining setting.

The results are shown in Table 2. The results show that the text mining approach performs relatively well. It achieves 59.51% accuracy on the topmost item and 85.46% on top 10 items (centroid classifier). The same author graph contains the least relevant information for the categorization task. The most relevant information is contained in the viewed-together graph.

TABLE 1. The performance of centroid classifier for text categorization by using different BOW construction settings.

No.	Setting	Accuracy			
		Top 1 (%)	Top 3 (%)	Top 5 (%)	Top 10 (%)
1	TF, $n = 1$, min-freq $= 1$, cut-off $= 0$	53.97	69.46	74.48	81.74
2	TF-IDF, $n = 1$, min-freq $= 1$, cut-off $= 0$	58.99	75.34	79.50	**85.55**
3	TF-IDF, $n = 2$, min-freq $= 1$, cut-off $= 0$	**59.60**	75.34	80.27	85.20
4	TF-IDF, $n = 3$, min-freq $= 1$, cut-off $= 0$	59.42	75.77	80.10	85.20
5	TF-IDF, $n = 2$, min-freq $= 2$, cut-off $= 0$	59.51	**76.21**	**80.79**	85.46
6	TF-IDF, $n = 2$, min-freq $= 3$, cut-off $= 0$	58.13	75.86	80.62	85.20
7	TF-IDF, $n = 2$, min-freq $= 2$, cut-off $= 0.1$	58.99	75.34	79.15	84.25

The bolded values represent the best achieved result for each accuracy measure.

TABLE 2. The results of the selected text categorization algorithms and DKs.

No.	Setting	Accuracy			
		Top 1 (%)	Top 3 (%)	Top 5 (%)	Top 10 (%)
1	Text mining, SVM	59.16	73.09	78.28	82.96
2	Text mining, *k*-NN	58.47	72.74	78.28	83.91
3	Text mining, CEN	**59.51**	**76.21**	**80.79**	**85.46**
4	DK viewed-together, SVM	70.75	86.93	**90.92**	93.68
5	DK viewed-together, *k*-NN	**72.74**	**87.80**	90.83	**93.94**
6	DK same-event, SVM	**32.00**	**49.04**	**54.67**	58.65
7	DK same-event, *k*-NN	31.92	47.66	53.37	**61.07**
8	DK same-author, SVM	18.94	27.51	31.22	36.24
9	DK same-author, *k*-NN	**19.81**	**31.74**	**36.24**	**43.59**

The bolded values represent the best achieved result for each accuracy measure in each group of experiments.

A *k*-NN applied to the viewed-together graph achieves 72.74% accuracy on the topmost item and 93.94% on the top 10 items. It is noteworthy that the choice of the classification algorithm is not as important as the selection of the data from which the similarities between objects are inferred.

4.3. Results of the proposed methodology

In the next set of experiments, we applied the proposed methodology. The results are shown in Table 3.

The first nine experiments summarized in Table 3 were performed by employing the proposed methodology on each graph separately. As before, we performed 10-fold cross-validation on the manually categorized lectures and employed centroid classifier, SVM-Multiclass and *k*-NN for the categorization task (we used the same parameter values as before). We set the PageRank damping factor to 0.4 when computing the structural-context feature vectors.

In the last three experiments summarized in Table 3, we employed the data fusion method explained in Section 3.4. In Experiment 10, we weighted all types of data (i.e. BOW, viewed-together, same-event and same-author) equally. We only show the results for Centroid Classifier (SVM and *k*-NN demonstrated comparable results). In Experiment 11, we employed DE to directly optimize the target evaluation metrics. The objective function was computed in an inner 10-fold cross-validation loop for each evaluation metric separately. We only employed a centroid classifier in this setting as it is fast enough to allow for numerous iterations required for the stochastic optimizer to find a good solution. The weights, determined by DE, averaged over the 10-folds for each evaluation metric separately, are given in Table 4.

In the last experiment (Experiment 12), we removed the viewed-together information from the evaluation process. The reason is that in real life, new lectures are not connected to other lectures in the viewed-together graph because they were not yet viewed by any user. Again, we employed DE in an inner 10-fold cross-validation loop for each evaluation metric separately. The resulting weights are given in Table 5.

From the results of the first nine experiments, we can confirm that the most relevant information is contained in the viewed-together graph. The centroid classifier applied to the viewed-together graph exhibits 74.91% accuracy on the topmost item and 95.33% on the top 10 items. We can also confirm that the choice of the classification algorithm is not as important as the selection of the data from which the similarities between objects

TABLE 3. The results of employing the proposed methodology.

No.	Setting	Accuracy			
		Top 1 (%)	Top 3 (%)	Top 5 (%)	Top 10 (%)
1	Viewed-together, SVM	70.41	85.46	89.71	93.60
2	Viewed-together, *k*-NN	70.75	84.60	89.36	93.34
3	Viewed-together, CEN	**74.91**	**89.01**	**92.13**	**95.33**
4	Same-event, SVM	31.74	50.17	**55.97**	59.95
5	Same-event, *k*-NN	**32.34**	**50.43**	55.96	64.79
6	Same-event, CEN	27.59	46.62	53.63	**65.05**
7	Same-author, SVM	**15.83**	24.22	27.33	33.04
8	Same-author, *k*-NN	15.48	23.70	27.94	32.52
9	Same-author, CEN	14.79	**25.52**	**31.74**	**42.73**
10	Combined, equal weights, CEN	65.73	83.21	87.97	93.42
11	Combined, DE, CEN	**78.11**	**91.43**	**94.03**	**95.85**
12	Without viewed-together, CEN	**62.28**	**79.84**	**84.08**	**89.27**

The bolded values represent the best achieved result for each accuracy measure in each group of experiments.

TABLE 4. The weights computed in the optimization process in Experiment 11.

Accuracy measure	Average weights			
	Viewed together	Same event	Same author	BOW
Top 1	0.9301	0.0080	0.0065	0.0554
Top 3	0.8712	0.0350	0.0184	0.0754
Top 5	0.7835	0.0699	0.0593	0.0873
Top 10	0.8012	0.0385	0.0692	0.0911

TABLE 5. The weights computed in the optimization process in Experiment 12.

Accuracy measure	Average weights		
	Same event	Same author	BOW
Top 1	0.3414	0.2440	0.4147
Top 3	0.3347	0.2236	0.4417
Top 5	0.3426	0.2416	0.4158
Top 10	0.3283	0.3654	0.3063

are inferred. Even so, the centroid classifier does outperform the SVM and the *k*-NN on the top 10 items and in the case of the viewed-together graph, also on the topmost item. The centroid classifier is outperformed by the other two classifiers on the topmost item in the case of the same-event and same-author graphs.

When comparing the approaches based on our methodology to the DK-based approaches, we can see that the centroid classifier applied to the viewed-together graph outperforms the SVM and the *k*-NN applied to the viewed-together diffusion kernel. On the other hand, with respect to the same-event and same-author graphs, the centroid classifier is outperformed by the DK-based approaches on the topmost predicted category.

The results of Experiment 10 show that weighting all types of data equally does not produce the best results. The accuracy falls in comparison with exploiting the viewed-together graph alone. The optimized weights indeed yield the best results (Experiment 11) and significantly improve the categorization performance (compared with exploiting the view-together graph alone: 78.11 vs. 74.91% on the topmost item, 95.85 vs. 95.33% on the top 10 items). This is also the case when the viewed-together information is not present in the test set (Experiment 12). The classifier is able to exploit the remaining data and exhibit accuracies that are significantly higher than those achieved by resorting to text mining alone (62.28 vs. 59.51% on the topmost item, 89.27 vs. 85.46% on the top 10 items). A classifier based on combined feature vectors is not only more accurate but is also robust to missing a certain type of data in the test examples.

5. EFFICIENT CLASSIFICATION WITH THE PAGERANK-BASED CENTROID CLASSIFIER

From the results presented in Section 4 it is evident that the centroid classifier offers very good performance and is much more efficient than its competitors. This outcome has motivated the development of a new centroid-based classifier that exploits the flexibility of the proposed feature-vector construction process in order to compute the centroids extremely efficiently.

5.1. The standard centroid classifier

In text mining, the centroid vector is a vector representing an artificial prototype document of a document set which 'summarizes' the documents in the set. Given a set of TF-IDF vectors, there are several ways of computing the corresponding centroid vector. Some of the well-known methods are the Rocchio formula, the average of vector components and

the (normalized) sum of vector components. Of these three methods, the normalized sum of vector components is shown to perform best in text classification scenarios [19]. In this case, given a set of document vectors represented as rows in matrix $\mathbf{D}$ (let $\mathbf{D}[i]$ denote the ith row in matrix $\mathbf{D}$) and a set of row indices $\mathbf{R}$, identifying documents that we want to group into a centroid, the normalized centroid vector $\mathbf{C}$ is computed as follows:

$$\mathbf{C}' = \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{D}[i]; \quad \mathbf{C} = \frac{\mathbf{C}'}{\|\mathbf{C}'\|}.$$

Let us now consider a multi-context setting introduced in Section 3. Suppose we have m contexts and thus m sets of feature vectors represented as rows in matrices $\mathbf{V}_1, \ldots, \mathbf{V}_m$. Again, let $\mathbf{R}$ be the set of row indices identifying objects that we want to group into a centroid. Finally, let $\mathbf{V}[i]$ denote the ith row in matrix $\mathbf{V}$. In the proposed framework, in order not to invalidate the intuitions provided in Sections 3.2 and 3.3, the centroid needs to be computed as follows ($\sum_k \alpha_k = 1, \alpha_k \geq 0$):

$$\mathbf{C} = \sqrt{\alpha_1}\frac{\mathbf{C}_1}{\|\mathbf{C}_1\|} \oplus \sqrt{\alpha_2}\frac{\mathbf{C}_2}{\|\mathbf{C}_2\|} \oplus \cdots \oplus \sqrt{\alpha_m}\frac{\mathbf{C}_m}{\|\mathbf{C}_m\|},$$
$$\text{where } \mathbf{C}_k = \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}_k[i]; \quad 1 \leq k \leq m. \tag{1}$$

Note that $\|\mathbf{C}\| = 1$ in this case. Following the above equations, we train a centroid classifier by computing a centroid vector for each class of objects (i.e. for each class label). In the classification phase, we simply compare the feature vector of a new (unlabelled) object with every centroid in the model. The centroid that yields the highest similarity score implies the label to be assigned to the object. More formally, we denote this as follows:

$$\mathbf{C}^* = \text{argmax}_{\mathbf{C}\in\mathbf{M}}\{\text{cossim}(\mathbf{C}, \mathbf{v})\}.$$

In this equation, $\mathbf{M} = \{\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_r\}$ is the centroid model, represented simply as a set of centroid vectors. Note that each centroid vector in the set corresponds exactly to one of the r classes. Furthermore, $\mathbf{v}$ is the (unlabeled) object's feature vector and $\mathbf{C}^*$ is the centroid vector that represents the predicted class.

5.2. PageRank-based centroid classifier on graphs

In this section, we show that each structural context of a centroid vector as given by Equation (1) can be very efficiently computed. Let us focus on one of the 'partial' centroids representing one of the structural contexts in Equation (1), $\mathbf{C}_k$ ($1 \leq k \leq m$). The methodology suggests that, in order to compute $\mathbf{C}_k$, we should construct $|\mathbf{R}|$ P-PR vectors and compute their average. However, it is possible to do this computation a lot more efficiently by computing just one P-PR vector. Instead of running P-PR from a single source node, we set $\mathbf{R}$ to be the set of source nodes (when the random walker teleports, it teleports to any of the nodes in $\mathbf{R}$ with equal probability). It turns out that a centroid computed in this way is exactly the same as if it was computed in the 'slow way' by strictly following the methodology. We show this equivalence in the following paragraphs.

Let $\mathbf{A}$ be the adjacency matrix of the graph representing one of the structural contexts, normalized so that each column sums up to 1. Let $\mathbf{V}$ be the matrix in which rows represent the corresponding structural-context feature vectors. Let $\mathbf{V}[i]$ denote the ith row in matrix $\mathbf{V}$ (i.e. the P-PR feature vector of the ith object). Let $\mathbf{R}$ be the set of row indices identifying nodes (objects) that we want to group into a centroid. Furthermore, let $\mathbf{t}_i$ be the 'teleport' vector defining the ith node as the source node, having the ith element set to 1 and all others to 0, $\mathbf{t}_i = [0, \ldots, 0, 1, 0, \ldots 0]^{\mathrm{T}}$. The size of this vector is equal to the number of rows in $\mathbf{V}$. Finally, let d be the PageRank damping factor. Then, each row in matrix $\mathbf{V}$ is computed by solving the P-PR equation:

$$\mathbf{V}[i] = (1-d)\mathbf{t}_i + d\mathbf{A}\mathbf{V}[i]. \tag{2}$$

If we now compute the average over the matrix rows (i.e. P-PR vectors) defined by $\mathbf{R}$, we get the following equation:

$$\frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}[i] = \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}((1-d)\mathbf{t}_i + d\mathbf{A}\mathbf{V}[i]),$$
$$\frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}[i] = \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}(1-d)\mathbf{t}_i + \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}d\mathbf{A}\mathbf{V}[i],$$
$$\frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}[i] = (1-d)\sum_{i\in\mathbf{R}}\frac{1}{|\mathbf{R}|}\mathbf{t}_i + d\mathbf{A}\frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}[i],$$
$$\mathbf{C}' = (1-d)\mathbf{t}' + d\mathbf{A}\mathbf{C}',$$
$$\text{where } \mathbf{C}' = \frac{1}{|\mathbf{R}|}\sum_{i\in\mathbf{R}}\mathbf{V}[i]; \quad \mathbf{t}' = \sum_{i\in\mathbf{R}}\frac{1}{|\mathbf{R}|}\mathbf{t}_i.$$

It is easy to see that this equation resembles the single-source P-PR equation (Equation 2). The main difference is the modified teleport vector $\mathbf{t}'$ which contains values $1/|\mathbf{R}|$ at locations that denote the nodes (objects) that we want to group into a centroid. This is exactly the P-PR equation with multiple source nodes where $1/|\mathbf{R}|$ is the probability of choosing a particular source node when teleporting. Therefore, instead of computing the average over several single-source P-PR vectors, we can compute just one multiple-source P-PR vector.

In case of having r classes and n objects, $n \gg r$, this not only speeds up the process by factor n/r but also reduces the time complexity from computing $O(n)$ P-PR vectors to computing $O(1)$ P-PR vectors. Practical implications are outlined in the following section.

6. TIME AND SPACE COMPLEXITY ANALYSIS

Whenever a set of new lectures enters the categorization system—regardless of whether we use the proposed methodology or the DK approach—the following procedure is applied:

TABLE 6. The time, in seconds, spent for feature vector or kernel computation, training and prediction.

	Times [s]		
Setting	Preprocessing	Training	Predicting
DK, *k*-NN	1193	0	1
P-PR, *k*-NN	286	0	85
P-PR, PRCC	0	35	34

(1) kernel or feature vectors are recomputed, (2) a model is trained on manually categorized lectures and (3) new lectures are categorized. Each fold in the 10-fold cross-validation roughly corresponds to this setting. We focused on the viewed-together graph only and measured the times required to perform each of these three steps in each of the 10-folds, computing average values in the end. The results are given in Table 6.

The results show that the DK-based approach (first row) is more demanding than the proposed methodology represented by the second row (1193 vs. 371 s). Roughly speaking, this is mostly due to the fact that in our case study, the diffusion kernel is computed over 3520 objects (resulting in a 3520 by 3520 kernel matrix), whereas, by using the proposed methodology, 'only' 1156 P-PR vectors of length 3520 need to be computed, where 1156 is the number of manually categorized lectures. Note also that computing a series of P-PR vectors is trivially parallelizable as one vector is computed entirely independently of the others (the so-called 'embarrassingly parallel' problem). On a quad-core machine, for example, the time required to compute the P-PR vectors in our case would be ∼80 s. Even greater efficiency is demonstrated by the PageRank-based centroid classifier (PRCC) (the last row). When the PRCC is used, the feature vectors are not pre-computed. Instead, in the training phase, approximately 130 P-PR vectors are computed, one for each category in the training set. In addition, in the prediction phase, ∼115 additional P-PR vectors are computed (115 objects is roughly the size of the test set). The PRCC thus requires only 70 s for the entire process. Needless to say, the PRCC-based approach is also trivially parallelizable, which makes it even more suitable for large-scale scenarios. Let us also point out that this efficiency is not achieved at the cost of decreased accuracy. In fact, the accuracy of the PRCC is exactly the same as that of the centroid classifier (see Section 5). Of all our experiments involving the viewed-together graph, the one employing the centroid classifier (which is equivalent to employing the more efficient PRCC) demonstrates the best accuracy.

Considering the space complexity, let us point out that the PRCC computes and stores only around 130 P-PR vectors of length 3520 (i.e. the PRCC model), which makes it by far the most efficient approach in terms of memory requirements. In comparison, the DK-based approach stores a 3520 by 3520 kernel matrix and the *k*-NN employed by the proposed methodology stores around 1040 P-PR vectors of length 3520 (roughly 1040 objects constitute the training set in each fold). For simplicity, we assumed that these vectors are not sparse, which is actually not the case. Due to the sparseness of the vectors, the amount of space consumed by using our methodology is in reality even lower.

7. QUALITATIVE ANALYSIS THROUGH VECTOR SPACE VISUALIZATION

In this section, we present another case study of our methodology: visualization of vector spaces. In machine learning and data mining, visualization techniques are often used for gaining insights into data and thus guiding the knowledge discovery process. In text mining, document space visualization techniques are used to provide overviews and insights into relatively large document collections [49, 50]. A document space is essentially a high-dimensional BOW vector space. To visualize a document space, feature vectors need to be projected onto a two-dimensional canvas so that the neighborhoods of points in the planar projection reflect the neighborhoods of vectors in the original high-dimensional space. Since the proposed methodology enables us to convert graphs into BOW-like vectors, we can visualize these graphs by using one of the available document space visualization techniques. Even more, we can visualize any 'fusion' of feature vectors obtained by following the proposed methodology. We will employ the document space visualization technique based on least-square meshes [49, 51]—more specifically, the implementation thoroughly presented in [52]—to demonstrate how visualized vector spaces can provide valuable qualitative explanations. Specifically, we will explain why the same-author graph, even though based on the solid intuition that 'a scientist normally sticks to his field of science', demonstrates such poor performance when used for categorization. From this same perspective, we will examine the same-event graph and look for the key difference between the same-author and same-event graphs on one hand and the viewed-together graph on the other.

Figure 5 shows the visualization of the same-author vector space with the edges adopted from the same-author graph. We can clearly see that we are dealing with many disconnected components. Each component corresponds to a group of lectures presented by the same author or several authors of which each collaborated with at least one other author from the group on at least one paper (lecture). The black dots in the visualization represent the lectures that were manually categorized (ground truth) and the white dots represent the uncategorized lectures. Note that (1) only the categorized lectures (black dots) participate in the 10-fold cross-validation process and (2) that, given a categorized lecture from a particular component, only the lectures from the same component participate as features in the feature vector of this categorized lecture. Let us now consider a component with one single categorized lecture (black dot). When such a categorized lecture is part of the test set

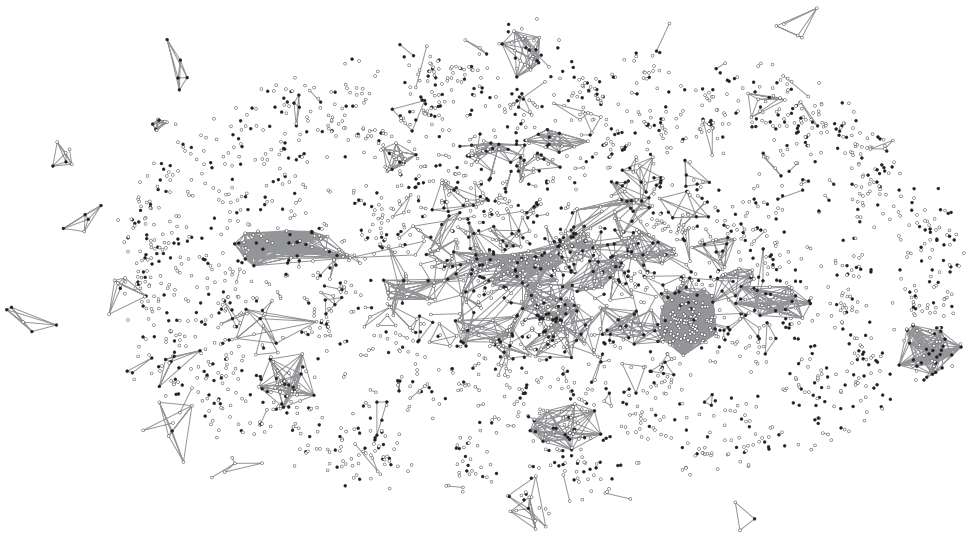


FIGURE 5. Visualization of the same-author vector space with the edges adopt from the corresponding graph.

FIGURE 6. Visualization of the same-event vector space with the edges adopted from the corresponding graph.

in the cross-validation process, the corresponding feature vector is orthogonal to every feature vector in the training set (note that only the categorized lectures constitute the training set). This means that it is not possible to categorize it due to the lack of information caused by the sparseness of the same-author graph. In general, the smaller the number of categorized lectures in a component, the bigger the chance that they will all constitute the same fold in the cross-validation setting, which results in the inability to classify any of them when the corresponding fold forms the test set. From this, we can conclude that having many disconnected components containing low numbers of categorized lectures leads to a poor categorization performance.

Figures 6 and 7 show the visualization of the same-event and viewed-together vector space, respectively. We can see (1) that the viewed-together graph contains less disconnected components than the same-event graph, which contains less disconnected components than the same-author graph (note that each single disconnected dot also represents a disconnected component), (2) that the viewed-together graph contains one large component containing nearly all the categorized lectures and (3) that the components in the same-event graph are larger than those in the same-author graph and thus each of them has the potential of containing a larger number of categorized lectures.

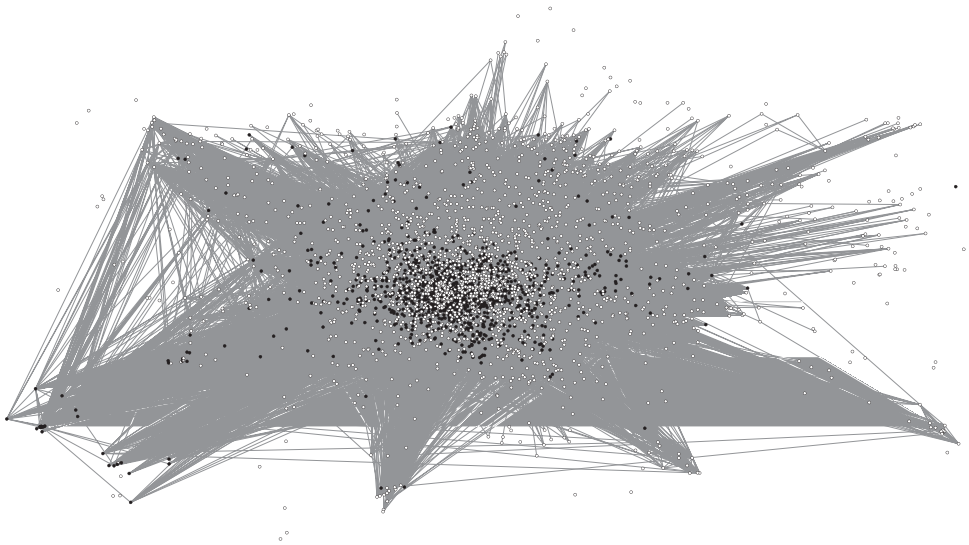


FIGURE 7. Visualization of the viewed-together vector space with the edges adopted from the corresponding graph.

To make sure that these observations are not merely 'visual artifacts', we computed the number of disconnected components and the number of components containing a certain number of categorized lectures in each of the three graphs. The results for the same-author and same-event graphs are shown in Fig. 8, whereas the viewed-together graph consists of 99 disconnected components of which 1 contains 1155 categorized lectures, 1 contains 1 categorized lecture and 97 contain no categorized lectures.

The charts in Fig. 8 clearly support our claims. The same-author graph contains the largest number of components (i.e. 2519) and a relatively large number of components that contain low numbers of categorized lectures. The same-event graph contains roughly 10 times less components and also the number of components containing low numbers of categorized lectures is much lower. If we look at the statistics of the viewed-together graph, we see that it contains only one disconnected categorized lecture that is orthogonal to the training set in the cross-validation process. From this perspective, the viewed-together graph exhibits the most appropriate structure, followed by the same-event and same-author graphs. This is also clearly reflected in our empirical studies presented in Section 4.3.

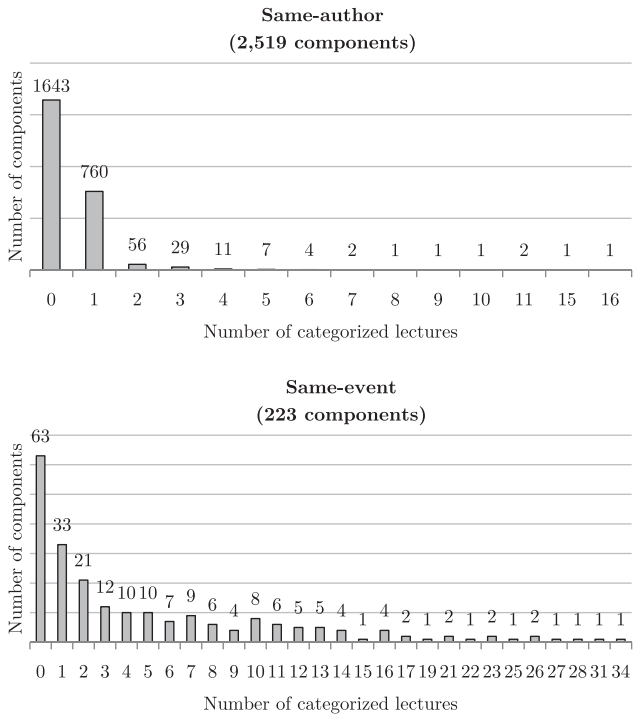


FIGURE 8. The number of disconnected components and the number of components containing a certain number of categorized lectures for the same-author and same-event graphs, respectively.

8. CONCLUSIONS AND FUTURE WORK

We presented a new methodology for mining heterogeneous information networks. The methodology is based on building a common vector space from textual and structural information, for which we use P-PR to compute structural-context features. We also devised and presented an extremely efficient PRCC. We applied the proposed methodology and the devised classifier in a video-lecture categorization case study and showed that the proposed methodology is fast and memory-efficient, and that the devised classifier is accurate and robust.

In future work, we will further develop the analogy between text mining and the proposed methodology, considering stop nodes (analogous to stop words). We will also look for a more efficient way to compute weights when combining feature

vectors. We will apply the methodology to larger problem domains to fully utilize the efficiency demonstrated by the developed PRCC.

FUNDING

This work has been partially funded by the European Commission in the context of the FP7 project FIRST (Large scale information extraction and integration infrastructure for supporting financial decision making) under the Grant Agreement No. 257928. The authors are grateful to the Center for Knowledge Transfer at Jožef Stefan Institute and to Viidea Ltd. for providing the VideoLectures dataset and the case study presented in this paper. The authors are also grateful to Anže Vavpetič for his help and discussions.

REFERENCES

[1] Feldman, R. and Sanger, J. (2006) *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, Cambridge, England and New York, USA.

[2] Han, J. (2009) Mining Heterogeneous Information Networks by Exploring the Power of Links. *Proc. 12th Int. Conf. on Discovery Science*, Porto, Portugal, October 3–5, pp. 13–30. Springer, Berlin, Germany.

[3] Nooy, W.D., Mrvar, A. and Batagelj, V. (2005) *Exploratory Social Network Analysis with Pajek*. Cambridge University Press, Cambridge, UK and New York, USA.

[4] Getoor, L. and Diehl, C. P. (2005) Link mining: a survey. *SIGKDD Explorations Special Issue on Link Mining*, **7**, 3–12.

[5] Lu, Q. and Getoor, L. (2003) Link-Based Text Classification. *Workshop Text-Mining and Link-Analysis at 18th Int. Joint Conf. on Artificial Intelligence*, Acapulco, Mexico, August 11, pp. 496–503. Morgan Kaufmann, San Francisco, USA.

[6] Grčar, M., Grobelnik, M. and Mladenić, D. (2007) Using Text Mining and Link Analysis for Software Mining. *Proc. 3rd Int. Workshop on Mining Complex Data*, Warsaw, Poland, September 17, pp. 1–12. Springer, Heidelberg, Germany.

[7] Page, L., Brin, S., Motwani, R. and Winograd, T. (1999) The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66. Stanford InfoLab, Stanford, USA.

[8] Grčar, M. and Lavrač, N. (2011) A Methodology for Mining Document-Enriched Heterogeneous Information Networks. *Proc. 14th Int. Conf. on Discovery Science (LNCS 6926)*, Espoo, Finland, October 5–7, pp. 107–121. Springer, Berlin, Heidelberg, New York.

[9] Mitchell, T.M. (1997) *Machine Learning*. McGraw-Hill, New York, USA.

[10] Fortuna, B., Grobelnik, M. and Mladenić, D. (2007) OntoGen: Semi-automatic Ontology Editor. *Proc. 2007 Conf. on Human Interface: Part II*, Beijing, China, July 22–27, pp. 309–318. Springer, Berlin, Germany.

[11] Fortuna, B., Mladenić, D. and Grobelnik, M. (2006) Visualization of text document corpus. *Informatica*, **29**, 497–502.

[12] Kim, H.-R. and Chan, P. K. (2008) Learning implicit user interest hierarchy for context in personalization. *J. Appl. Intell.*, **28**, 153–166.

[13] Grčar, M., Mladenić, D. and Grobelnik, M. (2005) User Profiling for Interest-Focused Browsing History. *Proc. 8th Multiconference Information Society IS 2005*, Ljubljana, Slovenia, October 11–17, pp. 182–185. Slovensko društvo Informatika, Ljubljana.

[14] Salton, G. (1989) *Automatic Text Processing—the Transformation, Analysis, and Retrieval of Information by Computer*, reprinted with corr. edition. Addison-Wesley Longman Publishing Co., Inc., Boston, USA.

[15] Sebastiani, F. (2002) Machine learning in automated text categorization. *ACM Comput. Surv.*, **34**, 1–47.

[16] Mladenić, D. (1998) Machine Learning on non-homogeneous, distributed text data. PhD Thesis, Faculty of Computer and Information Science, University of Ljubljana, Slovenia.

[17] Grobelnik, M. and Mladenić, D. (2005) Simple classification into large topic ontology of web documents. *J. Comput. Inf. Technol.*, **13**, 279–285.

[18] Tan, S. (2008) An improved centroid classifier for text categorization. *Expert Syst. Appl.: Int. J.*, **35**, 279–285.

[19] Cardoso-Cachopo, A. and Oliveira, A. L. (2006) Empirical Evaluation of Centroid-Based Models for Single-Label Text Categorization. Technical Report 7/2006. Instituto Superior Técnico, Lisbon, Portugal.

[20] Joachims, T., Finley, T. and Yu, C.-N.J. (2009) Cutting-plane training of structural SVMs. *Mach. Learn.*, **77**, 27–59.

[21] Crestani, F. (1997) Application of spreading activation techniques in information retrieval. *Artif. Intell. Rev.*, **11**, 453–482.

[22] Kleinberg, J. M. (1999) Authoritative sources in a hyperlinked environment. *J. Assoc. Comput. Mach.*, **46**, 604–632.

[23] Jeh, G. and Widom, J. (2002) SimRank: A Measure of Structural-Context Similarity. *Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 23–26, pp. 538–543. ACM, New York, USA.

[24] Kondor, R.I. and Lafferty, J. (2002) Diffusion Kernels on Graphs and other Discrete Structures. *Proc. 19th Int. Conf. on Machine Learning: ICML*, Sydney, Australia, July 8–12, pp. 315–322. Morgan Kaufmann, San Francisco, USA.

[25] Balmin, A., Hristidis, V. and Papakonstantinou, Y. (2004) ObjectRank: Authority-Based Keyword Search in Databases. *Proc. 30th Int. Conf. on Very Large Data Bases*, VLDB'04, Toronto, Canada, August 29–September 3, Vol. 30, pp. 564–575. VLDB Endowment, USA.

[26] Stoyanovich, J., Bedathur, S., Berberich, K. and Weikum, G. (2007) EntityAuthority: Semantically Enriched Graph-based Authority Propagation. *Proc. 10th Int. Workshop on the Web and Databases (WebDB 2007)*, Beijing, China, June 15, pp. 1–10. ACM, New York, USA.

[27] Zhu, X. and Ghahramani, Z. (2002) Learning from Labeled and Unlabeled Data with Label Propagation. Technical Report CMU-CALD-02-107. Carnegie Mellon University, Pittsburgh, USA.

[28] Zhou, D. and Schölkopf, B. (2004) A Regularization Framework for Learning from Graph Data. *ICML Workshop on Statistical Relational Learning*, Banff, Alberta, Canada, July 4–8, pp. 132–137. ACM, New York, USA.

[29] Ji, M., Sun, Y., Danilevsky, M., Han, J. and Gao, J. (2010) Graph Regularized Transductive Classification on Heterogeneous Information Networks. *Proc. 2010 European Conf. on Machine Learning and Knowledge Discovery in Databases: Part I*, Barcelona, Spain, September 20–24, pp. 570–586. Springer, Berlin, Germany.

[30] Yin, X., Han, J., Yang, J. and Yu, P. S. (2004) CrossMine: Efficient Classification Across Multiple Database Relations.*Constraint-Based Mining and Inductive Databases*, Hinterzarten, Germany, March 11–13, pp. 172–195. Springer, Berlin, Germany.

[31] Atrey, P.K., Hossain, M.A., El Saddik, A. and Kankanhalli, M. S. (2010) Multimodal fusion for multimedia analysis: a survey. *Multimedia Syst.*, **16**, 345–379.

[32] Kramer, S., Lavrač, N. and Flach, P. (2001) Propositionalization Approaches to Relational Data Mining. In Džeroski, S. and Lavrač, N. (eds) *Relational Data Mining*. Springer, New York, USA.

[33] Muggleton, S.H. (1992) *Inductive Logic Programming*. Academic Press Ltd., London.

[34] Lavrač, N. and Džeroski, S. (1994) *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, UK.

[35] Džeroski, S. and Lavrač, N. (eds) (2001) *Relational Data Mining*. Springer, Berlin.

[36] Caruana, R., Munson, A. and Niculescu-Mizil, A. (2006) Getting the Most Out of Ensemble Selection. *Proc. 6th Int. Conf. on Data Mining (ICDM'06)*, Hong Kong, China, December 18–22, pp. 828–833. IEEE Computer Society, USA.

[37] Rakotomamonjy, A., Bach, F.R., Canu, S. and Grandvalet, Y. (2008) SimpleMKL. *J. Mach. Learn. Res.*, **9**, 2491–2521.

[38] Vishwanathan, S., sun, Z., Ampornpunt, N. and Varma, M. (2010) Multiple Kernel Learning and the Smo Algorithm. *Advances in Neural Information Processing Systems 23*, Vancouver, Canada, December 6–9, pp. 2361–2369. Curran Associates, Norwich, England.

[39] Lanckriet, G.R.G., Deng, M., Cristianini, N., Jordan, M.I. and Noble, W.S. (2004) Kernel-Based Data Fusion and its Application to Protein Function Prediction in Yeast. *Pacific Symp. on Biocomputing*, Hawaii, USA, January 6–10, pp. 300–311. World Scientific, New Jersey, USA.

[40] Storn, R. and Price, K. (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.*, **11**, 341–359.

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

[42] Mladenić, D. (1998) Feature Subset Selection in Text-Learning. *Proc. 10th European Conf. on Machine Learning (ECML-98)*, Chemnitz, Germany, April 21–23, pp. 95–100. Springer, Berlin.

[43] Nummelin, E. (2004) *General Irreducible Markov Chains and Non-negative Operators*. Cambridge University Press, Cambridge, UK and New York, USA.

[44] MSR-TR-2005-09 (2005) *A Framework for Characterizing Feature Weighting and Selection Methods in Text Classification*. Microsoft Research. Cambridge, UK.

[45] Demšar, J., Zupan, B., Leban, G. and Curk, T. (2004) Orange: From Experimental Machine Learning to Interactive Data Mining. *Proc. 8th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, Pisa, Italy, September 20–24, pp. 537–539. Springer, Berlin, Heidelberg, New York.

[46] Berthold, M.R., Cebron, N., Dill, F., Gabriel, T.R., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K. and Wiswedel, B. (2007) KNIME: The Konstanz Information Miner. *Data Analysis, Machine Learning and Applications–Proc. 31st Annual Conf. of the Gesellschaft für Klassifikation*, Albert-Ludwigs-Universität, Freiburg, March 7–9, pp. 319–326. Springer, Berlin, Heidelberg, New York.

[47] Witten, I.H., Frank, E. and Hall, M.A. (2011) *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Amsterdam.

[48] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. and Euler, T. (2006) YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, August 20–23, pp. 935–940. ACM New York, NY, USA.

[49] Vieira Paulovich, F., Nonato, L.G. and Minghim, R. (2006) Visual Mapping of Text Collections through a Fast High Precision Projection Technique. *Proc. Conf. on Inf. Visualization*, Baltimore, USA, October 29–November 3, pp. 282–290. IEEE Computer Society, Washington, USA.

[50] Fortuna, B., Grobelnik, M. and Mladenić, D. (2005) Visualization of text document corpus. *Inform. J.*, **29**, 497–502.

[51] Sorkine, O. and Cohen-Or, D. (2004) Least-Squares Meshes. *Proc. Shape Modeling International 2004*, Genova, Italy, June 7–9, pp. 191–199. IEEE Computer Society, Washington, USA.

[52] Grčar, M., Podpečan, V., Juršič, M. and Lavrač, N. (2010) Efficient Visualization of Document Streams. *Proc. 13th Int. Conf. on Discovery Science*, Canberra, Australia, October 6–8, pp. 174–188. Springer, Berlin, Germany.