A Methodology for Mining Document-Enriched Heterogeneous Information Networks

Miha Grčar[1] and Nada Lavrač[1]

[1] Jožef Stefan Institute, Dept. of Knowledge Technologies, Jamova cesta 39, 1000 Ljubljana, Slovenia
{Miha.Grcar, Nada.Lavrac}@ijs.si

Abstract. 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 and 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 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 classification accuracy.

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

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 which takes advantage of both research fields, text mining and mining heterogeneous information networks.

Text mining [3], 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 corporate and governmental environments and especially by the rapid growth of the world's largest source of semi-structured data, the Web. Text mining extracts knowledge from document collections by using data mining, machine learning, natural language processing, and information retrieval techniques. Unlike in typical data mining, 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-NN, k-Means, SVM, and Naive Bayes).

Naturally, not all data comes in the form of documents. A lot of the recent data mining research is done on the data from networked systems where individual agents or components interact with other components, forming large, interconnected, and heterogeneous networks. For short, such networks are called heterogeneous information networks [6]. 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). In general, such networks can be formed from data in relational databases and ontologies where the objects are interlinked with heterogeneous links. In heterogeneous information networks, knowledge discovery is usually performed by resorting to social network analysis [20], link analysis techniques [21], and other dedicated approaches to mining heterogeneous information networks [6].

In many real-life scenarios, documents are available in information networks. This results in heterogeneous information networks in which some objects are associated each with its corresponding set of text 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). The abundance of such document-enriched networks motivates the development of a new methodology that joins the two worlds, text mining and mining 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 [14], 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 applying it to the categorization of video lectures on VideoLectures.net <http://videolectures.net/>, one of the world's largest academic video hosting Web portals.

The paper is structured as follows. In Section 2, we first present the related work. In Section 3, we present the proposed methodology. We present a new classifier that exploits the properties of the presented feature vector construction process in Section 4. Section 5 presents the experimental results and Section 6 concludes the paper with several ideas for future work.

2 Related Work

Text mining employs basic machine learning principles, such as supervised and unsupervised learning [19], to perform higher-level tasks such as text categorization (also known as "text classification"), topic ontology construction, text corpora

visualization [4], and user profiling [10]. Most text mining tasks rely on a bag-of-words vector representation of documents [15].

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 [16]. In [18], the authors present a method for categorizing Web pages into the Yahoo! Taxonomy <http://dir.yahoo.com/>. 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 that category. A similar approach is presented in [5], where Web pages are being categorized into the DMoz taxonomy <http://www.dmoz.org/>. Each category is modeled with the corresponding centroid bag-of-words vector and a document is categorized simply by computing the cosine similarity between the document's bag-of-words vector and each of the computed centroids. Apart from Naive Bayes [19] and centroid-based classifiers [22], SVM [9] is also a popular classifier for text categorization.

In the field of mining heterogeneous information networks, a different family of analysis algorithms was devised to deal with data analysis problems. 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 of activation [2], hubs and authorities (HITS) [11], PageRank and Personalized PageRank [14], SimRank [7], and diffusion kernels [12; 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 [1] employs global PageRank (importance) and Personalized PageRank (relevance) to enhance 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 Personalized PageRank vectors (PPVs) for all possible query words. HubRank [25] is an improvement of ObjectRank in terms of space and time complexity at no expense to accuracy. It examines query logs to compute several *hubs* for which PPVs are precomputed. In addition, instead of precomputing full-blown PPVs, they compute fingerprints [27] which are a set of Monte Carlo random walks associated with a node. 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 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.

For the classification tasks, Zhu and Ghahramani [30] present a method for transductive learning which 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 [31] can also be employed for categorization. However, most of these methodologies are devised for

graphs rather than heterogeneous networks. GNetMine [32] is built on top of the graph regularization framework but takes the heterogeneity of the network into account and consequently yields better results. CrossMine [33] 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.

Even though in this paper, we deal with feature vectors rather than kernels, the kernel-based data fusion approach presented by Lanckriet et al. [13] is closely related to our work. In their 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) [28; 29] which is tightly integrated into the SVM's margin maximization process. In [13], the authors define a quadratically constrained quadratic program (QCQP) 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 6 different kernels (4 of them are diffusion kernels based on graph structures). They show that their data fusion approach outperforms 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 [17] which enables us to directly optimize the target evaluation metric.

From a high-level perspective, the approaches presented in this section either (1) extract features from text documents for the purpose of document categorization, (2) categorize objects by propagating rank, similarity, or labels with a PageRank-like authority propagation algorithm, or (3) take network heterogeneity into account in a classification setting. In this work, we employ several well-established approaches from these three categories. The main contribution is a general-purpose framework for feature vector construction establishing an analogy between bag-of-words vectors and Personalized PageRank (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. Our methodology is thoroughly discussed in the following sections.

3 Proposed Methodology

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 5.1 for an example, where edge weights represent either video lecture co-authorship counts or the number of

users viewing the same video lecture). We also assume that several objects of interest are associated with text documents, which is not mandatory for the methodology to work. Fig. 1 illustrates the proposed feature vector construction process.

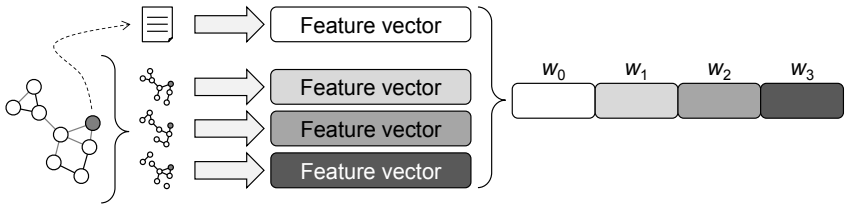

Fig. 1. The proposed methodology for transforming a heterogeneous information network and the corresponding text documents into a feature vector format. Feature vector construction is shown for one particular object.

Text documents are first transformed into feature vectors (i.e., TF-IDF bag-of-words vectors) as briefly explained in Section 3.1. In addition, each graph is transformed into a set of feature vectors. We employ Personalized PageRank for this purpose as explained in Section 3.2. 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 3.3. 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. These feature vectors can then be used as input for solving typical data mining tasks.

3.1 Constructing Feature Vectors from Text Documents

To convert text documents into their bag-of-words representations, we follow a typical text mining approach [3]. The documents are tokenized, stop words are removed, and the word tokens are stemmed (or lemmatized). Bigrams are considered in addition to unigrams. Infrequent words are removed from the vocabulary. Next, TF-IDF 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).

3.2 Constructing Structural-Context Feature Vectors with Personalized PageRank

For computing the structural-context feature vectors, we employ Personalized PageRank (P-PR) [14]. “Personalized” in this context refers to using a predefined set of nodes as the source of rank. In our case, P-PR is run from a single source node representing the object for which we want to compute the feature vector. The process

is equivalent to a random walk that starts in the source node. 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 to 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 $PR_i(j)$, where j is any node in the graph. $PR_i(j)$ is the probability that a random walker starting from node i will be observed at node j at an arbitrary point in time.

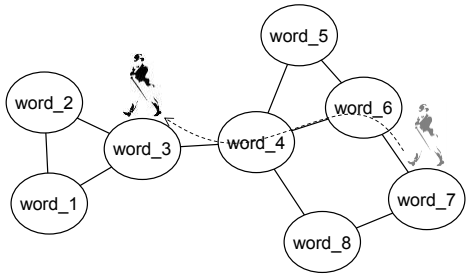

Fig. 2. The random writer principle: the random walker is "writing down" words that it encounters along the way. This is similar to generating random texts with a language model.

Recall that each node is a feature to be used in feature vector construction. For simplicity, consider that each feature is named by a single word, and that the random walker is "writing down" words that it encounters along the way (this principle is illustrated in Fig. 2). 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 bag-of-words 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 bags-of-words and thus nicely fits into the existing text mining frameworks.

In text mining, cosine similarity is normally used to compare bag-of-words vectors. Cosine similarity is equal to computing dot product provided that the two vectors are normalized in the Euclidean sense (i.e., their l^2-norm is equal to 1). Since we use 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 $\mathbf{v}_i = \langle PR_i(1), PR_i(2), \ldots, PR_i(n) \rangle$ for object i, the corresponding structural-context feature vector $\mathbf{v}_i'$ is thus computed as $\mathbf{v}_i' = \| \mathbf{v}_i \|^{-1} \langle PR_i(1), PR_i(2), \ldots, PR_i(n) \rangle$.

3.3 Combining Feature Vectors

The final step in the proposed methodology is to combine the computed feature vectors—either structural-context or bag-of-words vectors—describing a particular object with 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" *m* feature vectors, i.e., *m* – 1 structural feature vectors and a bag-of-words feature vector. For a given set of *n* objects, let us denote the *m* sets of feature vectors with $\mathbf{V}_1, \ldots, \mathbf{V}_m$, where each $\mathbf{V}_k$ is a matrix with *n* rows, in which *i*-th 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 there has been a lot of work done recently on Multiple Kernel Learning (MKL) which can also be employed for data fusion [13]. In MKL, multiple kernels are combined into a weighted convex combination of kernels which yields a combined kernel $\mathbf{K}_\Sigma = \sum_k \alpha_k \mathbf{K}_k$, $\sum_k \alpha_k = 1$, $\alpha_k \geq 0$. In analogy, we derive the following equation which shows how the above weights α_k can be used to combine feature vectors:

$$\mathbf{V}_\Sigma = \sqrt{\alpha_1}\mathbf{V}_1 \oplus \sqrt{\alpha_2}\mathbf{V}_2 \oplus \ldots \oplus \sqrt{\alpha_m}\mathbf{V}_m \ . \qquad (1)$$

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

$$\mathbf{V}_\Sigma\mathbf{V}_\Sigma^{\mathrm{T}} = \left(\sqrt{\alpha_1}\mathbf{V}_1 \oplus \ldots \oplus \sqrt{\alpha_m}\mathbf{V}_m\right)\left(\sqrt{\alpha_1}\mathbf{V}_1 \oplus \ldots \oplus \sqrt{\alpha_m}\mathbf{V}_m\right)^{\mathrm{T}} =$$
$$= \sum_k \alpha_k \mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} = \sum_k \alpha_k \mathbf{K}_k = \mathbf{K}_\Sigma \ .$$

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

In general, 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 5), we employ a stochastic optimizer and directly optimize the target evaluation metric.

4 Efficient Classification with PageRank-based Centroid Classifier

The combined feature vectors are ready to be employed for solving data mining tasks. For classification and clustering, any kernel or distance-based algorithm can be used (e.g., SVM, k-NN, k-Medoids, agglomerative clustering). With some care, the

algorithms that manipulate feature vectors (e.g., Centroid Classifier and k-Means) can also be employed.

We empirically evaluated some of these algorithms (see Section 5) where we applied the methodology for a categorization task. It turned out that Centroid Classifier offers a very good performance and is much more efficient than its competitors. This outcome has motivated the development of a new centroid-based classifier which exploits the flexibility of the proposed feature-vector construction process in order to compute centroids extremely efficiently.

Suppose we have several sets of feature vectors represented as rows in matrices $\mathbf{V}_1, \ldots, \mathbf{V}_m$. 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 i-th 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 ($\sum_k \alpha_k = 1$, $\alpha_k \geq 0$):

$$\mathbf{C} = \sqrt{\alpha_1}\frac{\mathbf{C}_1}{\|\mathbf{C}_1\|} \oplus \sqrt{\alpha_2}\frac{\mathbf{C}_2}{\|\mathbf{C}_2\|} \oplus \ldots \oplus \sqrt{\alpha_m}\frac{\mathbf{C}_m}{\|\mathbf{C}_m\|}\ , \quad \text{where}\ \ \mathbf{C}_k = |\mathbf{R}|^{-1} \sum_{i \in \mathbf{R}} \mathbf{V}_k[i],\ \ 1 \leq k \leq m\ . \tag{2}$$

Let us now focus on one of the "inner" centroids representing one of the structural contexts, $\mathbf{C}_k$ ($1 \leq k \leq m$). 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.

In case of having r classes and n objects, $n >> r$, 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 Section 5.4.

5 VideoLectures.net Categorization Case Study

The task in the VideoLectures.net case study was to develop a method that will assist in 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 the methodology in this use case, confronting it with a typical text mining approach and an approach based on diffusion kernels.

5.1 Dataset

The VideoLectures.net team provided us with a set of 3,520 English lectures, 1,156 of which were manually categorized. Each lecture is described with a title, while 2,537 lectures also have a short description and/or come with slide titles. 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 2,706 authors in the dataset, 219 events at which the lectures were recorded, and 3,274 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. Data preprocessing was performed by employing the proposed methodology, using as input the following textual and structural information about video lectures.

• Each lecture is assigned a **textual document** formed out of the title and, if available, extended with the corresponding description and slide titles.

• The **structural information** of this heterogeneous network is represented in the form of three homogeneous graphs in which nodes represent video lectures:

- **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.
- **Same-author graph**. Two nodes are linked if the two corresponding lectures were given by the same author or authors. The weight of a link is proportional to the number of authors the two lectures have in common.
- **Viewed-together graph**. Two nodes are linked if the two corresponding lectures were viewed together by a group of portal user. The weight of a link is proportional to the number of users that viewed both lectures.

5.2 Results of Text Mining and Diffusion Kernels

We first performed a set of experiments on textual data only, by following a typical text mining approach. In addition, we employed diffusion kernels (DK) [12] for classifying lectures according to their structural contexts.

In the text mining experiments, each lecture was assigned a text document formed out of the title and, if available, extended with the corresponding description and slide titles. We represented the documents as normalized TF-IDF bag-of-words vectors. We performed 10-fold cross validation on the manually categorized lectures. We performed flat classification as suggested in [5]. We employed several classifiers for the task: Centroid Classifier, SVM, and k-Nearest Neighbors (k-NN). In the case of SVM, we applied $SVM^{multiclass}$ [9] for which we set ε (termination criterion) to 0.1 and C (tradeoff between error and margin) to 5,000. In the case of k-NN, we set k (number of neighbors) to 20. We used dot product (i.e., cosine similarity) to compute the similarity between feature vectors.

In addition to the text mining experiments, we computed DK for the three graphs (we set the diffusion coefficient β to 0.0001). For each kernel separately, we

employed 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.

We measured classification accuracy on 1, 3, 5, and 10 top categories predicted by the classifiers. The results are shown in Fig. 3, where different evaluation metrics are stacked on top of each other thus forming one column for each of the performed experiments. Standard errors are shown next to the accuracy values given in the chart.

The results show that text mining approaches perform relatively well. They achieve 55.10% accuracy on the topmost item (k-NN) and 84.78% 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. k-NN applied to the textualized viewed-together graph achieves 72.74% accuracy on the topmost item and 93.94% on top 10 items. Noteworthy, the choice of the classification algorithm is not as important as the selection of the data from which the similarities between objects are inferred.

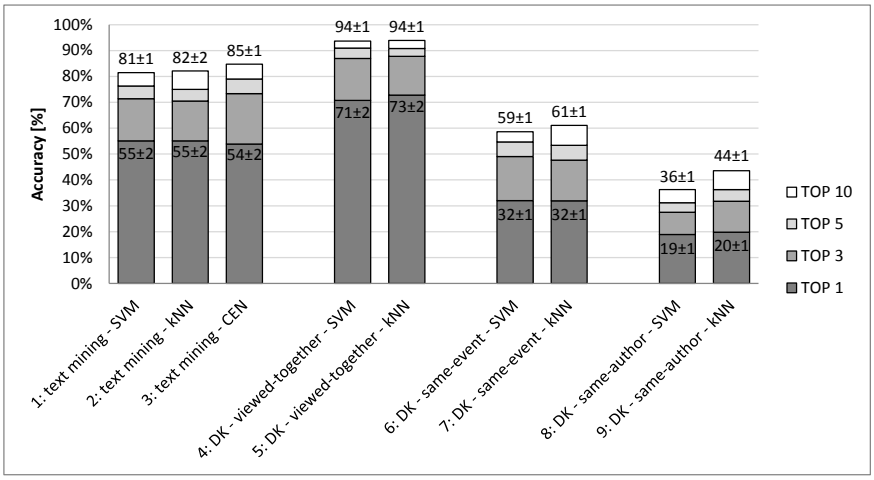

Fig. 3. Results of the selected text categorization algorithms and diffusion kernels.

5.3 Results of the Proposed Methodology

In the next set of experiments, we applied the proposed methodology. The results are shown in Fig. 4.

The first nine experiments in Fig. 4 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 SVM and k-NN for the categorization task (we used the same parameter values as before). In addition, we employed the PageRank-based Centroid Classifier (PRCC) discussed in Section 4. We set the PageRank damping factor to 0.4 when computing structural-context feature vectors.

In the last three experiments in Fig. 4, we employed the data fusion method explained in Section 3.3. In Experiment 10, we weighted all types of data (i.e., bags-of-words, viewed-together, same-event, and same-author) equally. We only show the

results for PRCC (SVM and k-NN demonstrated comparable results). In Experiment 11, we employed Differential Evolution (DE) to directly optimize the target evaluation metric. The objective function to be maximized was computed in an inner 10-fold cross validation loop and was defined as $\sum_{c=1,3,5,10} acc_c$ where acc_c stands for accuracy of the categorization algorithm on top c predicted categories. We only employed PRCC in this setting as it is fast enough to allow for numerous iterations required for the stochastic optimizer to find a good solution. DE computed the following weights: 0.9651, 0.0175, 0.0045, and 0.0130 for the bag-of-words, viewed-together, same-event, and same-author data, respectively. In the last experiment, we removed the viewed-together information from the test set. 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.

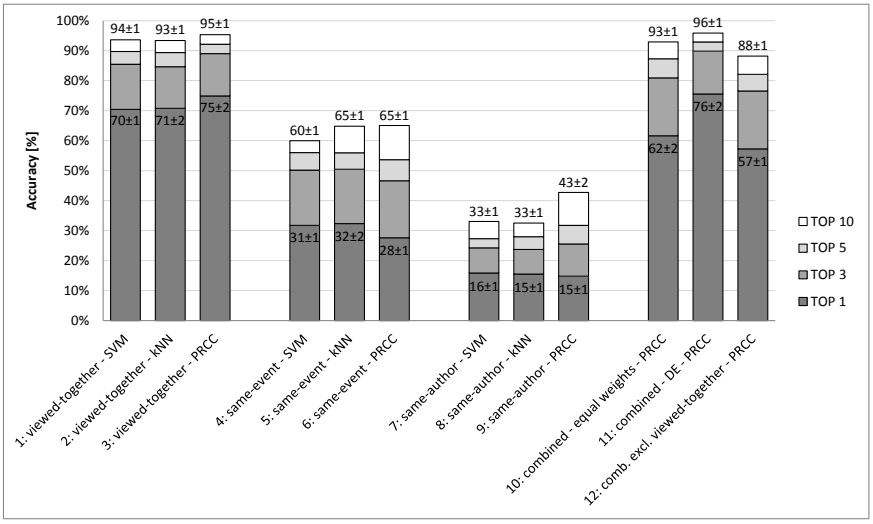

Fig. 4. Results of employing the proposed methodology.

From the results of the first 9 experiments, we can confirm that the most relevant information is contained in the viewed-together graph. PRCC applied to the textualized viewed-together graph exhibits 74.91% accuracy on the topmost item and 95.33% on 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 are inferred. Even so, PRCC does outperform SVM and k-NN on top 10 items and in the case of the viewed-together graph, also on the topmost item. PRCC is outperformed by the other two classifiers on the topmost item in the case of the same-event graph.

When comparing approaches based on our methodology to DK-based approaches, we can see that PRCC applied to textualized viewed-together graph outperforms SVM and k-NN applied to the viewed-together diffusion kernel. On the other hand, with respect to the same-event and same-author graphs, PRCC 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). Since most of the relevant information is contained in the viewed-together graph, the accuracy achieved through combining feature vectors is not much higher than that demonstrated by exploiting the viewed-together graph alone. However, as clearly demonstrated by the last experiment, the combined feature vectors excel when the viewed-together information is not present in the test set. 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 (88.15% versus 84.78% accuracy on top 10 items). A classifier based on combined feature vectors is thus not only more accurate but also robust to missing a certain type of data in test examples.

5.4 Notes on Time and Space Complexity

Whenever a set of new lectures enters the categorization system—whether we use the proposed methodology (termed "bags-of-features" in Fig. 5) or the DK approach—the following procedure is applied: (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 3 steps in each of the 10 folds, computing average values in the end. The results are given in Fig. 5.

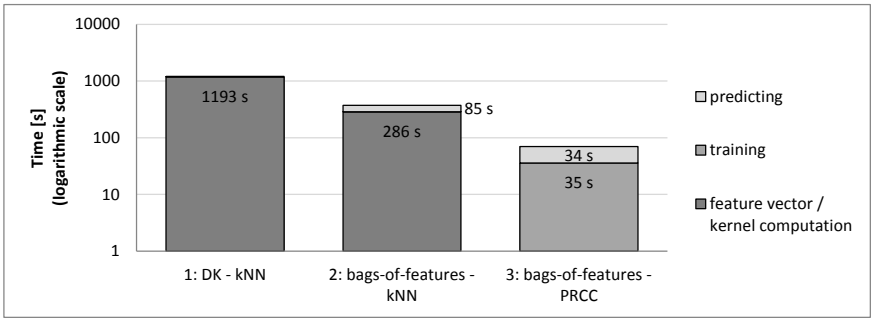

Fig. 5. The time spent for feature vector or kernel computation, training, and prediction. Note that the chart is plotted on a logarithmic scale.

The results show that the DK-based approach (column 1) is more demanding than the proposed methodology represented by column 2 (1,193 seconds vs. 371 seconds). Roughly speaking, this is mostly due to the fact that in our use case, the diffusion kernel is computed over 3,520 objects (resulting in a 3,520 by 3,520 kernel matrix) while by using the proposed methodology, "only" 1,156 P-PR vectors of length 3,520 need to be computed, where 1,156 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 approximately 80 seconds. Even greater efficiency is demonstrated by PRCC (the last column). When PRCC is used, the feature vectors are not precomputed. 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, approximately 115 additional P-PR vectors are computed (115 objects is roughly the size of the test set). PRCC thus requires only 70 seconds 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, of all our experiments involving the viewed-together graph, the one employing PRCC demonstrates the best accuracy.

The thorough analysis of the space complexity is beyond the scope of this paper. Let us just point out that PRCC computes and stores only around 130 P-PR vectors of length 3,520 (i.e., the PRCC model) which makes it by far the most efficient approach in terms of required memory. In comparison, the DK-based approach stores a 3,520 by 3,520 kernel matrix and k-NN employed by the proposed methodology stores around 1,040 P-PR vectors of length 3,520 (roughly 1,040 objects constitute the training set in each fold). For simplicity, we assumed that these vectors are not sparse, which is actually not the case and would speak even more in favor of the proposed methodology.

6 Conclusions and Future Work

We presented a new methodology for mining heterogeneous information networks. The methodology is based on building a common vector space for textual and structural information. We use Personalized PageRank (P-PR) to compute structural-context features. We also devised and presented an extremely efficient PageRank-based centroid classifier. We applied the proposed methodology and the devised classifier in a video lecture categorization use case 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 develop the analogy between text mining and the proposed methodology further, 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 devised PageRank-based Centroid Classifier.

Acknowledgements

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 n. 257928. The authors would also like to thank Center for Knowledge Transfer at Jožef

Stefan Institute and Viidea Ltd. for providing the dataset and use case presented in the paper.

References

1. A. Balmin, V. Hristidis, Y. Papakonstantinou: ObjectRank: Authority-based Keyword Search in Databases. Proceedings of VLDB ‘04, pp. 564–575 (2004)
2. F. Crestani: Application of Spreading Activation Techniques in Information Retrieval. Artificial Intelligence Review, vol. 11, pp. 453–482 (1997)
3. R. Feldman, J. Sanger: The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press (2006)
4. B. Fortuna, M. Grobelnik, D. Mladenic: OntoGen: Semi-Automatic Ontology Editor. HCI International ’07, Beijing (2007)
5. M. Grobelnik, D. Mladenic: Simple Classification into Large Topic Ontology of Web Documents. Journal of Computing and Information Technology, vol. 13(4), pp. 279–285 (2005)
6. J. Han: Mining Heterogeneous Information Networks by Exploring the Power of Links. Proceedings of Discovery Science ‘09, pp. 13–30 (2009)
7. G. Jeh, J. Widom: SimRank: A Measure of Structural Context Similarity. Proceedings of KDD ‘02, pp. 538–543 (2002)
8. M. Ji, Y. Sun, M. Danilevsky, J. Han, J. Gao: Graph Regularized Transductive Classification on Heterogeneous Information Networks, 2010. Machine Learning and Knowledge Discovery in Databases ‘10, pp. 570–586 (2010)
9. T. Joachims, T. Finley, C.-N. J. Yu: Cutting-Plane Training of Structural SVMs. Journal of Machine Learning, vol. 77(1) (2009).
10. H. R. Kim, P. K. Chan: Learning Implicit User Interest Hierarchy for Context in Personalization. Journal of Applied Intelligence, vol. 28(2) (2008)
11. J. M. Kleinberg: Authoritative Sources in a Hyperlinked Environment. Journal of the Association for Computing Machinery, vol. 46, pp. 604–632 (1999)
12. R. I. Kondor, J. Lafferty: Diffusion Kernels on Graphs and Other Discrete Structures. Proceedings of ICML ‘02, pp. 315–322 (2002)
13. G. R. G. Lanckriet, M. Deng, N. Cristianini, M. I. Jordan, W. S. Noble: Kernel-based Data Fusion and Its Application to Protein Function Prediction in Yeast. Proceedings of the Pacific Symposium on Biocomputing, pp. 300–311 (2004)
14. L. Page, S. Brin, R. Motwani, T. Winograd: The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford InfoLab (1999)
15. G. Salton: Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley (1989)
16. F. Sebastiani: Machine Learning in Automated Text Categorization. ACM Computing Surveys, vol. 34(1), pp. 1–47 (2002)
17. R. Storn, K. Price: Differential Evolution: A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, vol. 11, pp. 341–359, Kluwer Academic Publishers (1997)
18. D. Mladenic: Machine Learning on Non-Homogeneous, Distributed Text Data. PhD thesis (1998)

19. T. Mitchell: Machine Learning. McGraw Hill (1997)
20. W. de Nooy, A. Mrvar, V. Batagelj: Exploratory Social Network Analysis with Pajek, Cambridge University Press (2005)
21. L. Getoor, C. P. Diehl: Link Mining: A Survey. SIGKDD Explorations, vol. 7(2), pp. 3–12 (2005)
22. S. Tan: An Improved Centroid Classifier for Text Categorization. Expert Systems with Applications, vol. 35(1–2) (2008)
23. T. Gärtner: A Survey of Kernels for Structured Data. ACM SIGKDD Explorations Newsletter, vol. 5(1), pp. 49–58, ACM, NY (2003)
24. S. Chakrabarti: Dynamic Personalized PageRank in Entity-Relation Graphs. In Proceedings of WWW 2007, pp. 571–580 (2007)
25. J. Stoyanovich, S. Bedathur, K. Berberich, G. Weikum: EntityAuthority: Semantically Enriched Graph-based Authority Propagation. In Proceedings of the 10th International Workshop on Web and Databases (2007)
26. D. Fogaras, B. Rácz: Towards Scaling Fully Personalized PageRank. In Proceedings of the Workshop on Algorithms and Models for the Web-graph (WAW 2004), pp. 105–117 (2004)
27. A. Rakotomamonjy, F. Bach, Y. Grandvalet, S. Canu: SimpleMKL. Journal of Machine Learning Research, vol. 9, pp. 2491–2521 (2008)
28. S. V. N. Vishwanathan, Z. Sun, N. Theera-Ampornpunt, and M. Varma: Multiple Kernel Learning and the SMO Algorithm. Advances in Neural Information Processing Systems 23 (2010)
29. X. Zhu, Z. Ghahramani: Learning from Labeled and Unlabeled Data with Label Propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University (2002)
30. D. Zhou, B. Schölkopf: A Regularization Framework for Learning from Graph Data. ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields (2004)
31. M. Ji, Y. Sun, M. Danilevsky, J. Han, J. Gao: Graph Regularized Transductive Classification on Heterogeneous Information Networks. In Proceedings of PKDD, pp. 570–586 (2010)
32. X. Yin, J. Han, J. Yang, P. S. Yu: CrossMine: Efficient Classification Across Multiple Database Relations. In Proceedings of Constraint-Based Mining and Inductive Databases, pp. 172–195 (2004)