Biomedical Text Categorization with Concept Graph Representations Using a Controlled Vocabulary

Meenakshi Mishra, Jun Huan
Department of Electrical Engineering and Computer Science, University of Kansas
{mmishra, jhuan}@ittc.ku.edu

Said Bleik
Information Systems Department New Jersey Institute of Technology
sb252@njit.edu

Min Song
Department of Lib. and Information Science
Yonsei University
min.song@yonsei.ac.kr

ABSTRACT

Recent work using graph representations for text categorization has shown promising performance over conventional bag-of-words representation of text documents. In this paper we investigate a graph representation of texts for the task of text categorization. In our representation we identify high level concepts extracted from a database of controlled biomedical terms and build a rich graph structure that contains important concepts and relationships. This procedure ensures that graphs are described with a regular vocabulary, leading to increased ease of comparison. We then classify document graphs by applying a set-based graph kernel that is intuitively sensible and able to deal with the disconnectedness of the constructed concept graphs. We compare this approach to standard approaches using non-graph, text-based features. We also do a comparison amongst different kernels that can be used to see which performs better.

Categories and Subject Descriptors

Dataming Methodologies: Biomedical text mining

Keywords

Text Categorization, Graph Classifier, Biomedical informatics

1. INTRODUCTION

Biomedical electronic document databases are growing exponentially, resulting in huge digital repositories. Organizing and searching these documents manually is increasingly costly and time consuming. With the rapid growth, biomedical literature has been the subject of intensive information retrieval and machine learning investigations throughout past decades. Text categorization is one challenging research area where text documents are categorized using predefined labels based on their content. Applying improved text categorization techniques to the biomedical databases is essential to overcome the information overload problem and to facilitate indexing, filtering and managing the growing number of articles in those databases.

Most of the existing text categorization techniques use a vector representation of documents. In the vector space model, key entities and concepts are identified from text and used as features. The disadvantage of the vector representation is the lack of semantic relationships among key entities and concepts in the text. Recently, graph mining and graph modeling techniques have begun to gain popularity in modeling complex data such as protein sequences and structures and social networks [1]. The advantage of graph modeling is the use of "rich" semantic representation of relationships among key entities and concepts in a text and hence may yield improved results when classifying documents.

In addition, kernel functions for graphs and other structured data have garnered particular interest. In this work we have designed a customized kernel function based on set matching to compute the similarity between document graphs. This kernel decomposes the similarity problem into two components: 1) matching concept terms encoded as nodes in a document graph, and 2) computing the overall similarity of the edges in one document graph to the edges in another. This approach will evaluate two document graphs as similar if they share both a large number of concept terms as well as the same relationships between those terms. The choice of this kernel function was made with computational simplicity in mind, as well as ease of dealing with disconnected graphs.

Since the reliability of the kernel function in measuring document similarity is directly dependent on the methods used to encode the document graphs, in this work we have paid particular attention to representations that provide consistent descriptions across many different documents.

Several approaches to text categorization using graph representations have been explored as outlined in section 2. The novelty of our approach lies primarily in the methods used to generate the nodes and edges for each document graph. While previous works have focused on nodes that encode specific words or sentences, the approach described here focuses on so-called concept graphs that encode specific biomedical concepts as nodes in a document graph. These concept nodes use a regular and controlled vocabulary for describing documents, and avoid issues with vagaries and inconsistencies in the terms used by various papers. By using such a controlled vocabulary we ensure that matches between concept nodes reliably indicate similarities between documents.

The proposed technique is based on the extension of our previous study of representing full-text articles as a graph [22]. Different from our previous work, in this paper, we assign weights

BIOKDD'12, August 12, 2012, Beijing, China.

to edge based on concept relation. Generation of edges between concept nodes follows an approach that encodes the relationships between the concept terms presented in a document graph. These relationships essentially denote containment between different concept terms. For example, if the concept terms "Protein" and "H SP90" are both present in a document, then an edge will be added between them indicating that "HSP90" is a child of "Protein" since the former is a more specific case of the latter.

The techniques described above have been applied to a set of biomedical texts collected from BioMed Central. The documents are open access publications and are categorized by the journal they were published in. The biomedical terms extracted from the text can be mapped to concepts from the Unified Medical Language System (UMLS) database. UMLS is a comprehensive repository of biomedical knowledge and is accessed by many data mining applications. It is provided by the National Library of Medicine [2]. The procedure of building the concept graph is similar to how humans intuitively identify the topic of a certain text by reading. We usually spot keywords in the text, recognize the concepts behind these keywords, and relate them to other identified concepts that we find in the text. Once the overall picture is clear, we categorize the text into a class that is commonly seen in the literature. Our proposed method involves a combination of concept identification and graph mining techniques to classify biomedical documents. The proposed method is evaluated on several text categorization problems and compared to existing document classification methods utilizing text-based features. The results demonstrate an improvement in classification accuracy when using the concept graphs compared to using only the unstructured textual features retrieved from the selected documents. We have compared the affect of making use of different parameters associated with the concept term on the classification performance. We have also provided results comparing the use of concepts with and without relationship edges.

The rest of the paper is organized as follows. In Section 2 we define text categorization and refer to some of the related. In Section 3 we describe the concept graph building and classification algorithm. In Section 4 we describe the experimental data sets, the model construction and evaluation, and the analysis of the results. Section 5 concludes the paper with discussion on the performance of the proposed method.

2. RELATED WORK

Text categorization is the automatic process of sorting documents into classes or groups based on their content. Text categorization has attracted significant research interest in information science [3]. The applications of text categorization include indexing and classifying of scientific publications, email filtering, literature based discovery, and finding relationships among biomedical entities. The success of a text categorization application is based on the efficiency and accuracy of the underlying information retrieval and machine learning techniques used.

Several text categorization techniques have been proposed to automate the manual process of organizing and searching documents. One of the popular techniques is the Naïve Bayesian approach. The Naïve Bayesian probabilistic approach was suggested for automatic indexing of documents and is shown to be straightforward but surprisingly efficient in terms of classification [4]. It is assumed that the extracted feature words are independent and therefore Bayes' theorem can be used in the classification algorithm.

Graphs have also been used to categorize documents based on graph matching [5]. Complex structures such as documents can be represented as graphs where nodes represent textual or other document features, and edges represent relationships between those features. The addition of relationship edges to describe documents can create a much higher-dimensional feature space, thus allowing for more nuanced and potentially useful embeddings of the documents.

The relations used to connect graph nodes can be as diverse as the applications. [6] proposes a graph representation for document summarization tasks. They use a thesaurus and association rules to connect key phrases in the text. [7] also uses graphs to represent documents for summarization. They use 3 graphs to capture word-word, word-sentence, and sentence-sentence relationships in the text. They then compute word and sentence saliency scores to rank their results.

As for text categorization, there have been some attempts that use graph representations and graph mining to enhance feature representation and selection. In [8], 3 different data sets were used for classification experiments each having its own representation of relationships between node objects in a graph. Co-authors were used to link scientific publications, actors to link movies, and page hyperlinks to link Wikipedia documents. Weighted frequent subgraphs were used in [9] to construct effective feature vectors for classification and to overcome the computation overhead that is associated with graph structures. [10] uses exact and inexact graph matching as well as substructure pruning and ranking to optimize classification and compare their result to a Naïve Bayesian classifier. [11] attempts to exploit the linguistic syntactic and semantic characteristics of phrases in text. They encode phrases as graphs and use a substructure and pattern discovery algorithm for classification.

A common preprocessing used for graph classification is projecting the graph onto a kernel space using a kernel function. One possible kernel function can be defined as an inner product between two graphs and must be positive semi-definite and symmetric.

Such a function embeds graphs or any other objects into a Hilbert space, and is termed a Mercer kernel from Mercer's theorem. Kernel functions can enhance classification in two ways: first, by mapping vector objects into higher dimensional spaces; second, by embedding non-vector objects in an implicitly defined space.

Kernel functions for graphs have received much attention recently. The simplest kernels are defined in terms of set operations between nodes and edges. Some more sophisticated developments include kernels based on comparing simple structures such as paths between two graphs such as the shortest path [12], marginalized [13]and spectrum [14] kernels, as well as cycles [15]. Other kernels rely on more complicated structure comparisons such as between subtrees [16] and subgraphs [17]. Some rely on direct matching of graph substructures [18]. String kernels were used in text classification in [19]. The feature space was generated using all string subsequences and the kernel measured the similarity of documents based on the similarity of those subsequences of strings. [5] used a semantic kernel that incorporates Wikipedia background knowledge to enrich the document representation. They achieved improved accuracy in document classification when compared to traditional bag-of-words representation.

3. THE ALGORITHM

Our algorithm consists of two major components. The first is the graph generation part which is based on a named entity recognition (NER) module and a concept identification module. The second is the application of a graph kernel function to compute the similarity between the generated graphs and a kernel classifier to discriminate between papers given their embedding in the kernel space.

Figure 1 shows the data flow of the procedure of extracting concepts and relations as well as feeding them to a graph kernel function for text categorization. In brief, the process is as follows: first, a set of biomedical articles are selected from BioMed Central; next, biomedical concepts are extracted from the documents and mapped to concepts from the UMLS database; concept relationships are then extracted and graphs are constructed consisting of nodes representing concepts and edges representing concept relationships; finally, the concept graphs are used to compute a kernel matrix.

The overall process consists of two phases: 1) Input Graph Construction and 2) Classifier Learning and Output. Each phase is described in details in sections 3.1 and 3.2 below, beginning with graph construction.

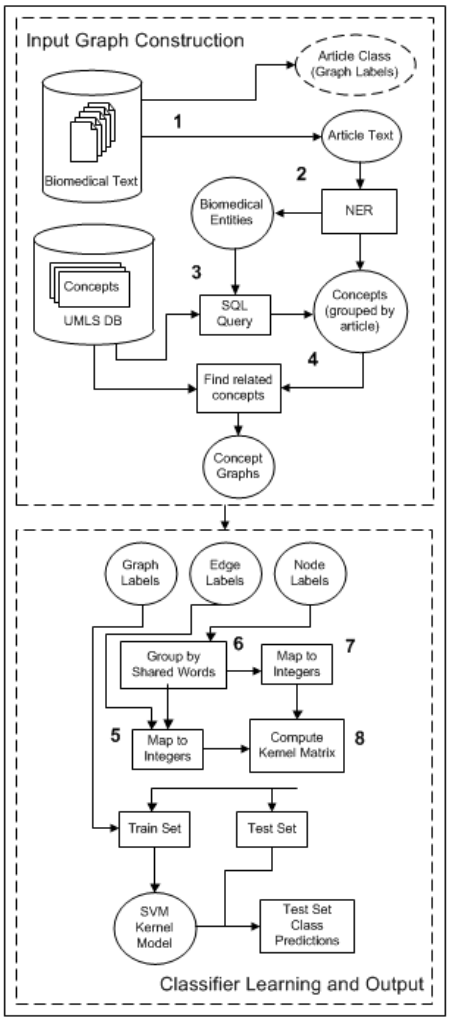

Figure 1. System Overview

3.1. Graph Construction and Processing

The graph construction phase begins by collecting a set of published articles from different journals of BioMed Central. The articles were grouped by the journal in which they were published. The journal name here represents a high level category of biomedical-related disciplines and thus is used as the class label for the different sets of documents. The text content is then used to construct a set of concept graphs, where each document is represented by one graph.

As we mentioned earlier, each concept graph consists of a set of concept nodes and relation edges. To extract the concepts from the text, we use LingPipe's [20] named entity recognition (NER) module which is trained on the Genia corpus [21]. To ensure the concepts correspond to a controlled set of vocabulary, we attempt to map them to the UMLS database. UMLS is a comprehensive biomedical ontology of concepts provided by the National Library of Medicine [2]. Mapping the named entities into UMLS concepts involves comparing all potential substrings of the key phrases extracted by NER since those are sometimes longer than the concepts in UMLS and contain additional adjectives or terms. The named entity "5 and 10 lM parthenolide" for example doesn't exist as a concept but the substring "parthenolide" does.

Mapping the biomedical entities into predefined concepts also allows us to look for possible relations among them within UMLS. A concept string might refer to multiple concepts with different meanings whereas a concept unique identifier (CUI) refers to only one concept associated with one or more string descriptors that might slightly vary because of the different vocabulary sources merged in UMLS.

For each text document we create a new Concept Graph and add the mapped concepts as nodes. The graph nodes hold the string values of each concept and the corresponding CUIs. The multiple CUIs are implicitly disambiguated by possible relations that might be added to the graph since edge weights are also used in weighting nodes. For each pair of nodes, we attempt to find a relation in UMLS and add it as an edge between the nodes if it exists. The available relations are of semantic nature some of which are synonym, parent-child, and sibling relationships. Figure 2 shows a sample text and the corresponding concept graph with the extracted nodes and edges.

To weight the concepts before being fed to the classifier, we use the following three weight components:

1. *cf*: The concept occurrence frequency in the text document.

2. *idfw*: The inverse document frequency weight of a concept:

$$idfw_i = 1 - \left(\frac{\log(idf_i)}{\log(N)} \right)$$

where *idfi* is the number of documents term i occurs in, and N is the total number of documents indexed. This weight is similar to the traditional inverse document frequency (*IDF*) measure except that the index is built beforehand only once using a fixed dataset of over 20,000 PubMed documents spanning different topics. This weight ensures common biomedical concepts are given lower weights due to their lower discriminatory value. *idfw* is a value between 0 and 1 where lower values indicate that the concept term is a very common one in the biomedical domain.

3. *cw*: The connectivity weight of a concept node. This weight quantifies the importance of a concept in terms of its relationships to other concepts in the text. In other words, it is a measure of the

node connectivity within the graph. It is calculated as the magnitude of the relations weights vector for a certain concept:

$$cw = \sqrt{\sum_{1}^{n} cf_i^2}$$

where n is the number of concepts related to concept i and cfi is the frequency of a related concept i. The value of cw not only captures how much a concept is related to other concepts but also how much it is related to important concepts of high frequencies in a document.

3.2. Classifier Learning with Kernels

After transforming a set of papers into a set of graphs, a graph kernel function is applied to compute the similarity between all pairs of paper graphs, and the resulting kernel matrix is used for classification. A simple set-based kernel is used to measure concept graph similarity based on the number of shared nodes and number of edges with matching endpoints. There are a couple properties of these concept graphs that make a set-based kernel function attractive. The first reason is that the set computations used are easily implemented and understood, leading to a kernel function that is easy to interpret, which results in a greater confidence in producing reliable measures of graph similarity. The second reason is that many of the concept graphs are disconnected or sparse, with many more nodes than edges, which can pose problems for some graph mining algorithms. By decomposing the graphs into sets of nodes and edges this issue is eliminated.

The kernel between two concept graphs (concept graph kernel) is defined as the sum of two components, a kernel between edges (edge kernel K_E), and a kernel between nodes (node kernel K_N):

$$K(x,y) = K_N(x,y) + K_E(x,y)$$

In this paper, we investigate the utility of using the node kernel, the edge kernel, and the kernel between two concept graphs.

The kernel between the nodes of two concept graphs x and y is defined as the weighted ratio of the nodes intersecting between two graphs to the union of the nodes in the two graphs.

$$K_N(x,y) = \sum_i \sum_j \frac{I(N_i = N_j) w(N_i)}{w(N_i) + w(N_j) - I(N_i = N_j) w(N_i)}$$

Here, i and j are indices having the value 1,2…n_x and 1,2….n_y respectively, where n_x and n_y are the number of nodes in graph x and graph y. N_i and N_j are the node concept terms in graph x and y respectively, and $w(N_i)$ or $w(N_j)$ represents the weight allotted to the node concept term N_i and N_j. The weight for each node concept term is either the frequency of the node, the inverse of the inverse document frequency, the connectivity weights or a combination of the above three parameters. I is the indicator function which holds the value 1 if N_i is equal to N_j and zero otherwise.

The kernel between edges is defined as sum of all pairwise similarities between edges in one concept graph, and each edge in the other:

$$K_E(x,y) = \sum_i \sum_j K_e(E_i, E_j)$$

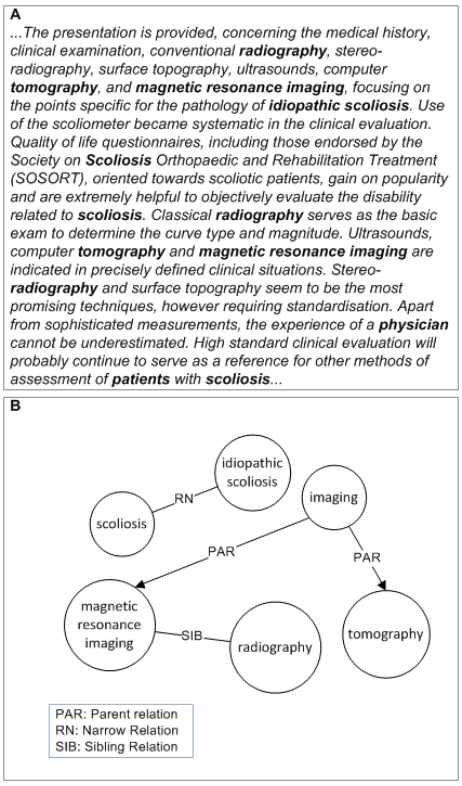

Figure 2. Sample text and corresponding graph

where i and j are indices from 1 to the number of edges in graph x and y respectively. E_i and E_j are the i^{th} and i edges of graph x and y. The set of such similarities forms a bipartite graph, where nodes in one partition correspond to edges in one concept graph, and nodes in the other partition correspond to edges in the other concept graph. The set of bipartite edges correspond to similarities between nodes in each partition.

The similarity between two edges is calculated via yet another kernel similar to the node kernel, defined as:

$$K_e(E_i, E_j) = \sum_{l=1,2} \sum_{m=1,2} \frac{I(N_l = N_m) w(N_l)}{w(N_l) + w(N_m) - I(N_l = N_m) w(N_l)}$$

where: E_i is an edge in graph x and E_j is an edge in graph y. Each edge forms a connection between the nodes N_1 and N_2.

Once the kernel between all graphs is computed, the graphs are entered into a kernel matrix. This matrix can then be used in a kernel-based classifier to make predictions on new data. We have used support vector machines to classify the data once the kernels were obtained.

4. EXPERIMENTAL STUDY

In this paper, we utilize two different sets of data to show three points: 1) the performance of concept graph methods vs. text-based methods, 2) the utility of different weight combinations assigned to the nodes, and 3) the performance difference between using document concepts for classification with and without relationship edges.

4.1. Dataset

The two data sets consist of collections of biomedical publications collected from a set of journals of BioMed Central. The articles are grouped by the journal they were published in and

the journal names are used as the classes for prediction by the trained classification model. Table 1 shows the class labels and number of samples for each data set. In total, the first set contains 81 documents and the second contains 100 documents.

Table 1: Training datasets

Data set 1		
Label	**Class**	**Samples**
A	Aesthetic Plastic Surgery	9
B	Aging Cell	9
C	AIDS and Behavior	6
D	Angiogenesis	7
E	Apoptosis	15
F	Gastrointestinal Surgery	22
G	Hematopathology	13

Data set 2		
Label	**Class**	**Samples**
A	Cardiovasc_Disord	12
B	Gastroenterol	24
C	Genomics	24
D	Musculoskelet_Disord	14
E	Pregnancy_Childbirth	9
F	Psychiatry	17

4.2. Model Construction

Once the graphs corresponding to queried papers are constructed and the edge relationships and node concept labels have been mapped to integer labels, a kernel matrix of similarities between all the graphs can be computed. The kernel used was first normalized before actually being used for the training purposes.

$$KN_{ij} = \frac{K_{ij}}{\sqrt{K_{ii} \times K_{jj}}}$$

Here, KN_{ij} is the single term in the normalized kernel matrix *KN* located in i^{th} row and j^{th} column. *K* is the un-normalized kernel matrix, and K_{ij} is a term in matrix *K* with *i* and *j* specifying the location of the term. *K* can refer to node kernel, edge kernel or concept graph kernel depending on which kernel is being used.

In this paper, we first investigate the utility of using concept graphs over the text based approach. Different values of $w(N_i)$ that were tried for the calculation of the node and the edge kernel were concept occurrence frequency (*cf*), reciprocal of inverse document frequency weight (1/idfw), concept weights (*cw*), *cf*×(1/*idfw*), *cf*×*cw*, (1/*idfw*)×*cw* and *cf*×(1/*idfw*)×*cw*. We also explored the case where all $w(N_i)$ was set to 1. We study which of the values of $w(N_i)$ mentioned above gives us the best performance. We also study if inclusion of edges gives any boost in performance or not.

Since dataset 1 had seven different classes, and dataset 2 had six different classes, and support vector machine is a binary classifier, we binarized the problem by attempting to classify just one class versus all others.

We obtained our training and testing data sets using 10-fold cross-validation. We used another 10-fold cross-validation in the training data set to select model parameters. The only parameter that we optimized was the weight ratio *wr* of the positive and negative samples. We used grid search with the range of $[10^{-3}, 10^{1}]$ and cross validation to optimize *wr*. Once we had the optimal value of *wr*, we used all training samples to build a single model and applied the model to the testing data set.

4.3. Model Evaluation

Each trained model was evaluated in the testing data set. We collected accuracy, precision, recall and the *F1* score for each case. We report the average value of the metrics over the 10 cross-validation trials. Accuracy is defined as *(TP+TN)/S* where *TP* stands for number of true positives, *TN* stands for number of true negatives and *S* is the total number of testing samples. Precision is defined as the ratio of true positives to the total number of positives predicted by the classifier (*Precision=TP/(TP+FP)* where *FP* is the number of false positives). Recall is defined as the ratio of the number of true positives to the total number of positives present in the test dataset *(Recall= TP/(TP+FN)* where *FN* is the number of false negatives). The *F1* score is defined as the inverse of the arithmetic mean of the reciprocal values of precision and recall.

$$F1score = \frac{2 \times precision \times recall}{precision + recall}$$

We used F1 score because it is designed for imbalanced data sets as we are studying here. In our experimental study, we report the F1 score only.

4.4. Analysis of Results

A set of concept graphs were first extracted from the journal articles which were intended to be classified according to their journal. Weighted kernels were built for the concept graphs using different values used for weights, and the kernels were used to classify the two datasets.

We first used abstracts of the papers used in the dataset to classify the data. To this end, we used Naïve Bayes classifier since Naïve Bayes is commonly used for text classification. Also, Naïve Bayes assumes all features are conditionally independent given the class, which is a safe assumption to make when the dimensionality of the data is large. In text classification, the dimensionality of data is often found to be large. We observed that for both dataset 1 and dataset 2, when we used Naïve Bayes classifier, the classifier classified every entry as one class for all the labels to be classified. Thus, the precision did not exist. However, when we used concept graphs, the classifier did some work of classifying, and hence we got a value for the *F1* score. Clearly, the use of concept graph outperformed the conventional text based classification using Naïve Bayes.

We tried three different parameters for the value of weights in the kernels *w(Ni)*. The three parameters were concept term frequency *cf*, inverse document frequency *idfw*, and the connectivity weight cw. We also evaluated the case where we did not use any of above parameters, or assigned the value 1 to *w(Ni)*. We refer to this as plain kernel. Table 2 shows the results when using the normalized node kernels weighted by the three

parameters. The values reported are the *F1* scores for the test results. It is interesting to see that the plain kernel outperforms the other kernels in 7 out of the 13 cases. This result is quite interesting and may be due to the fact that the number of concept terms is quite large, and hence, their occurrence in the papers is quite sparse. Thus, the concept frequencies, the inverse document frequencies and the connectivity weights have little effect on the kernels.

Table 2: The performance of weighted node kernels. The weights used are (plain), concept frequencies cf, inverse document frequencies idf, and connectivity weights cw. The values reported are the F1 scores.

	Label	Plain	cf	1/idfw	cw
Data set 1	**A**	NaN	**0.095**	0.407	NaN
	B	0.191	0.162	**0.258**	NaN
	C	0.090	0.103	0.083	**0.200**
	D	**0.111**	0.109	0.053	0.053
	E	0.258	**0.262**	NaN	0.095
	F	**0.462**	0.304	0.203	0.296
	G	**0.234**	NaN	0.054	NaN
Data set 2	**A**	**0.298**	0.093	0.156	0.188
	B	**0.340**	0.286	0.312	0.238
	C	0.315	0.261	**0.333**	0.234
	D	**0.316**	0.095	0.197	0.103
	E	NaN	0.174	**0.208**	0.173
	F	**0.273**	0.215	0.066	0.209

It seems that the edges of the graphs should also play an important role in the classification as the edges describe the relationships between the concept nodes. In this study, we do not make use of the type of relationship that exists between the concept terms. Rather, we just use the fact that if a relationship exists or not. The results are presented in Table 3, which shows the node kernel, edge kernel and the concept graph kernel only evaluated for the value of *w(Ni)* equal to one. We did evaluate these kernels for other value of *w(Ni)* too, but only show the results for *w(Ni)*=1 because this seemed to work the best for the node kernels. As can be seen in Table 3, using just the node kernel did perform better in eight of the 13 cases. However, this was not the case when using other values of *w(Ni)*.

When using *w(Ni)* equal to *cf* or *cf×(1/idfw)*, the edge kernel outperforms the node and concept weight kernel majority of the time (6 out of 13 and 7 out of 13 times respectively). When *w(Ni)* is set to *cf×(1/idfw)* or *cf×cw*, both the nodes and the concept graph kernel show a better performance than others 5 out of 13 times. Setting *w(Ni)* to be *cf×cw×(1/idfw)* produces a tie between node kernels performance and edge kernel performance. The node kernels solely outperformed others when *w(Ni)* was either set to 1, *cf* or *1/idfw*. Hence, we can say that using just the node kernel outperformed using just the edge kernel or the concept graph kernel in most cases.

Table 3: The performance of Node kernel, Edge kernel and Concept graph kernel. The value of w(Ni) is one. The values reported are F1 scores.

	Labels	Node kernel	Edge kernel	Concept graph kernel
Data set 1	A	NaN	0.094	**0.120**
	B	**0.191**	0.140	0.177
	C	0.090	0.118	**0.167**
	D	0.111	0.111	**0.146**
	E	**0.258**	NaN	0.143
	F	**0.462**	0.171	0.362
	G	**0.234**	0.227	0.206
Data set 2	**A**	**0.298**	0.094	NaN
	B	**0.340**	0.140	0.267
	C	0.315	0.118	**0.333**
	D	**0.316**	0.111	0.078
	E	NaN	NaN	**0.148**
	F	**0.273**	0.171	0.152

5. CONCLUSION

Categorizing biomedical text is a challenging problem due to the huge number of articles published every year. In this study, we propose a promising approach to text categorization based on building concept graphs to represent documents and classifying them using an SVM classifier. The results show that the rich representation of documents in form of graphs does significantly improve the classification performance when compared to traditional Naïve Bayes method. It was also interesting to note that in some cases addition of relationships (edges) to the concepts did improve the classification performance but in most cases, using just the concept terms were sufficient. However, we did not utilize the type of relationship that occurred between the concept terms, which might have contributed in deteriorating the performance.

Acknowledgments

This work is partially supported by the KU Specialized Chemistry Center (NIH award U54 HG005031). In addition, partial support for this research was provided by the National Science Foundation under grants DUE-0434581 and DUE-0434998, by the Institute for Museum and Library Services under grant LG-02-04-0002-04.

References

[1] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SDM'06: Workshop on Link Analysis, Counter-terrorism and Security*, 2006.

[2] D. A. Lindberg, B. L. Humphreys, and A. T. McCray, "The Unified Medical Language System.," *Methods of information in Medicine*, vol. 32, no. 4, p. 281, 1993.

[3] F. Sebastiani, “Machine learning in automated text categorization,” *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[4] M. E. Maron, “Automatic indexing: an experimental inquiry,” *Journal of the ACM (JACM)*, vol. 8, no. 3, pp. 404–417, 1961.

[5] P. Wang and C. Domeniconi, “Building semantic kernels for text classification using wikipedia,” in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 713–721.

[6] Y. M. Chen, X. L. Wang, and B. Q. Liu, “Multi-document summarization based on lexical chains,” in *Proceedings of the 2005 international conference on machine learning and cybernetics*, 2005, pp. 1937–1942.

[7] X. Wan, J. Yang, and J. Xiao, “Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction,” in *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, 2007, vol. 45, p. 552.

[8] R. Angelova and G. Weikum, “Graph-based text classification: learn from your neighbors,” in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006, p. 492.

[9] C. Jiang, F. Coenen, R. Sanderson, and M. Zito, “Text Classification using Graph Mining-based Feature Extraction,” *Research and Development in Intelligent Systems XXVI*, pp. 21–34.

[10] M. Arey and S. Chakravarthy, “InfoSift: Adapting Graph Mining Techniques for Text Classification,” in *Proceedings of the Eighteenth International FLAIRS Conference*, 2005.

[11] K. R. Gee and D. J. Cook, “Text Classification Using Graph-Encoded Linguistic Elements,” in *Proc. of the 18th Intl. FLAIRS Conf*, 2005.

[12] K. M. Borgwardt and H. P. Kriegel, “Shortest-path kernels on graphs,” 2005.

[13] H. Kashima, K. Tsuda, and A. Inokuchi, “Marginalized kernels between labeled graphs,” in *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, 2003, vol. 20, p. 321.

[14] C. Leslie, E. Eskin, and W. S. Noble, “The spectrum kernel: A string kernel for SVM protein classification,” in *Proceedings of the Pacific Symposium on Biocomputing*, 2002, vol. 7, pp. 566–575.

[15] T. Horváth, T. G\ärtner, and S. Wrobel, “Cyclic pattern kernels for predictive graph mining,” in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 158–167.

[16] P. Mahé and J. P. Vert, “Graph kernels based on tree patterns for molecules,” *Machine learning*, vol. 75, no. 1, pp. 3–35, 2009.

[17] J. Huan, D. Bandyopadhyay, J. Prins, J. Snoeyink, A. Tropsha, and W. Wang, “Distance-based identification of structure motifs in proteins using constrained frequent subgraph mining,” in *Computational systems bioinformatics: CSB2006 conference proceedings, Stanford CA, 14-18 August 2006*, 2006, vol. 4, p. 227.

[18] H. Fr\öhlich, J. K. Wegner, F. Sieker, and A. Zell, “Optimal assignment kernels for attributed molecular graphs,” in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 225–232.

[19] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text classification using string kernels,” *The Journal of Machine Learning Research*, vol. 2, pp. 419–444, 2002.

[20] “LingPipe: Named Entity Tutorial.” [Online]. Available:http://alias-i.com/lingpipe/demos/tutorial/ne/read-me.html. [Accessed: 31-Jul-2011].

[21] J. D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, “GENIA corpus-a semantically annotated corpus for bio-textmining,” *Bioinformatics-Oxford*, vol. 19, no. 1, pp. 180–182, 2003.

[22] S. Bleik, M. Song, A Smalter, J. Huan, G. Lushington "CGM: A biomedical text categorization approach using concept graph mining", *Bioinformatics and Biomedicine Workshop, 2009. BIBMW 2009*, page 38-43.