Application of graphs to textural document categorization

Yu Sun
Marion Neumann
CSE Master Project Final Report, Fall 2017

1 Introduction

Text categorization is important application in the field of natural language processing. The related techniques are used in a broad range of domains, from news filtering to semantic analysis. Traditional methods vectorize a document for further analysis by retrieving words, such as bag-of-words and n-grams. Surprisingly, no textural structure is extracted from the text, which is critical for humans to comprehend a paragraph of text. Throughout the project, we consider a representation of text to preserve the structural information, which is completely neglected in the word-based methods. Hence, we consider the text categorization as graph classification problem.

Graphs are powerful data structures that are used to illustrate complex information about entities and interaction between them, *e.g.* chemical compounds. By abstracting information into simple nodes and edges, graphs represent objects in an analytical way. For instance, researchers predict the class of newly discovered molecules by analyzing its graphical structure, *e.g.*, predicting carcinogenicity in molecules. The same logical can be applied to text categorization.

In [1], a graph representation of text, namely graph-of-words, was presented for feature reduction of text. In this project, we further explored the power of graph-of-words by enriching the graph with semantic and grammatical information. For the classification, an powerful graph kernel, namely propagation kernel, was used for classifying graphs with labels and attributes of nodes. Simulations on the corpus of formal articles and informal reviews gives a relative complete picture of the ability to capture useful structural information in different texts.

2 Related Work

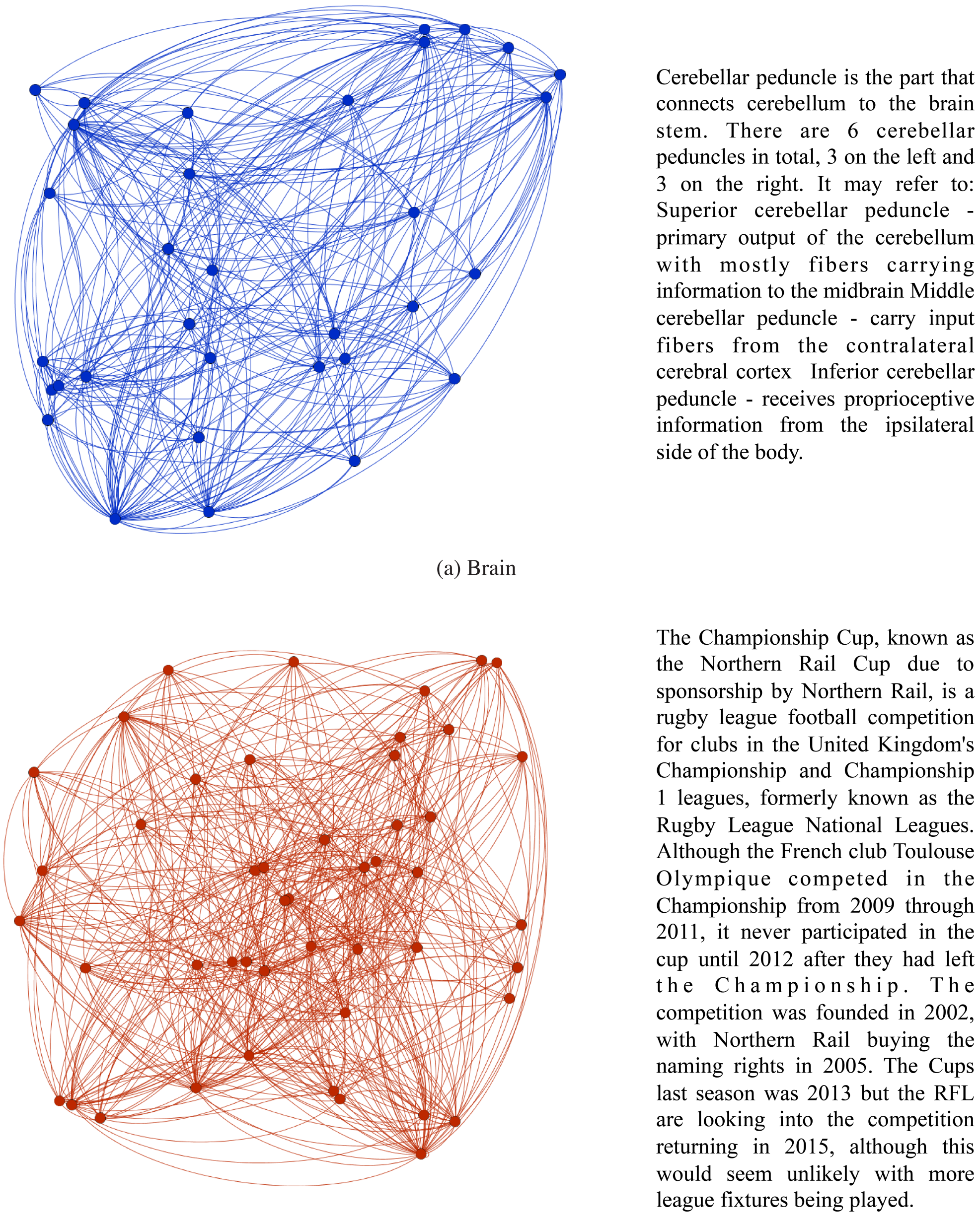
In the section, we present the related work in the field of text categorization, graph classification & kernel and the pioneering attempts in the combination of the two fields.

2.1 Text Categorization

Text categorization (*or text classification, topic retrieval*) is correspondent to assigning a given document to one of the predefined labels. We refer to [2] for a brief history of the problem. It is worth noticing that T. Joachims first used the support vector machines (SVMs) with term-frequency inverse-document-frequency (Tf-idf) features to categorize texts [3], which initialized the usage of machine learning in the problem. Nowadays, the popular approach is the use of neural network (NN), whose strong ability of extracting information from the data fits the arena very well [4]. proposed a neural network for learning a vector in the Euclidean space for each document in the training corpus. However, these approaches are mainly driven by the words.

2.2 Graph classification & kernel

Graph classification has attracted a continuous attention for decades. Generally, classifying graphs does not differ from classifying ordinary data examples. What makes graph classification distinct is defining an appropriate metric, in other word, kernel, to measure the similarity between graphs. Based on how the graph structure is captured, we can distinguish graph kernels into 4 classes: kernels based on walks [5] and paths [6], kernels based on limited-size subgraphs [7], and kernels based on structure propagation [**?**], where the propagation kernel belongs to.



Cerebellar peduncle is the part that connects cerebellum to the brain stem. There are 6 cerebellar peduncles in total, 3 on the left and 3 on the right. It may refer to: Superior cerebellar peduncle - primary output of the cerebellum with mostly fibers carrying information to the midbrain Middle cerebellar peduncle - carry input fibers from the contralateral cerebral cortex Inferior cerebellar peduncle - receives proprioceptive information from the ipsilateral side of the body.

(a) Brain

The Championship Cup, known as the Northern Rail Cup due to sponsorship by Northern Rail, is a rugby league football competition for clubs in the United Kingdom's Championship and Championship 1 leagues, formerly known as the Rugby League National Leagues. Although the French club Toulouse Olympique competed in the Championship from 2009 through 2011, it never participated in the cup until 2012 after they had left the Championship. The competition was founded in 2002, with Northern Rail buying the naming rights in 2005. The Cups last season was 2013 but the RFL are looking into the competition returning in 2015, although this would seem unlikely with more league fixtures being played.

(b) Rugby league

Figure 1: Shows the constructed graph and correspondent paragraph. (a) is from the class of ***brain***. (b) is from the class of ***rugby league***

2.3 Similar Work

In 2007, [8] performed subgraph mining on the graph-of-words representations to retrieve keywords in web-site articles. In 2015, [] proposed the concept of graph-of-words and considered the text categorization as a graph classification problem. In their work, texts are transformed into undirected graphs without node labels and attributes and classified by subgraph kernel method. Without any help of node features, the graph-of-words representation achieved competitive results compared to several baselines (linear SVM wit Tf-idf).

3 Proposed Method

3.1 Preliminary Concepts

Graph-of-words We model a given textual document as a graph-of-words, which corresponds to a graph whose vertices represent unique terms of the document and whose edges represent the co-occurrences between the terms within a fixed-size sliding window. The underlying assumption of graph-of-words is that all terms of the document have undirected relationships to each other, and the relationship of two terms fades as their term distance gets larger. We weighted each of the edges in the graph with the frequency. figure 1 illustrates the graphrized textual document.

Labels & Attributes of Nodes In the graph classification, the discrete labels and continuous features of nodes are commonly introduced into the graph as supplementary information. In the model, we assigned terms of the document labels and attributes respectively. Popular classes of words' labels in the field of natural language processing (NLP) are part-of-speech (POS) and name-entity-recognition (NER) tags. We used both two classes as our nodes' labels in the simulations. For the attributes, we used a neural network to learn each word's embeddings [9], which encodes the semantic meaning of the word into a vector with arbitrary dimension. For instance, the word 'king' can be calculated by 'queen' - 'women' + 'men' in the space of embeddings. Learning word embeddings is another popular field and we refer to [9] for further explanation.

Propagation Kernel The propagation kernel is introduced by M. Neumann in 2015 for solving the challenges in the graph classification problem [10].

- Missing information learning to partially labeled graph.
- Uncertain information arising from aggregating information from multiple sources.
- continuous information derived form complex and possibly noisy measurements.

In the project, the incompleteness of labels, continuous values in the word embeddings and the heterogeneity of nodes' features result the aforementioned challenges exactly. The general formulation of the Propagation kernel is given as follows.

Definition 1. *The Propagation Kernel is a kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ among graph instance $G_i \in \mathcal{X}$.*

$$K(G_i, G_j) = \sum_{u \in G_i} \sum_{v \in G_j} k(u, v), \tag{1}$$

where the u and v indicate node in the the G_i and G_j respectively, and $k(\cdot, \cdot)$ represents the kernel of nodes. With consideration of a sequence of graphs G_i^t with evolving node information based on information propagation, we define the kernel contribution of iteration t by

$$K(G_i^{(t)}, G_j^{(t)}) = \sum_{u \in G_i(t)} \sum_{v \in G_j(t)} k(u, v), \tag{2}$$

The node kernel $k(\cdot, \cdot)$ can be further expressed as

$$k(u, v) = k_l(u, v) k_a(u, v) \tag{3}$$

where the $k_l(\cdot, \cdot)$ represents the kernel of nodes with labels taken into account, and $k_a(\cdot, \cdot)$ represents the kernel with attributes taken into account. Overrall the Propagation Kernel K can be written as

$$K(G_i, G_j) = \sum_{t=1}^{t_{max}} K(G_i^{(t)}, G_j^{(t)}), \tag{4}$$

For detailed formulation we refer to [].

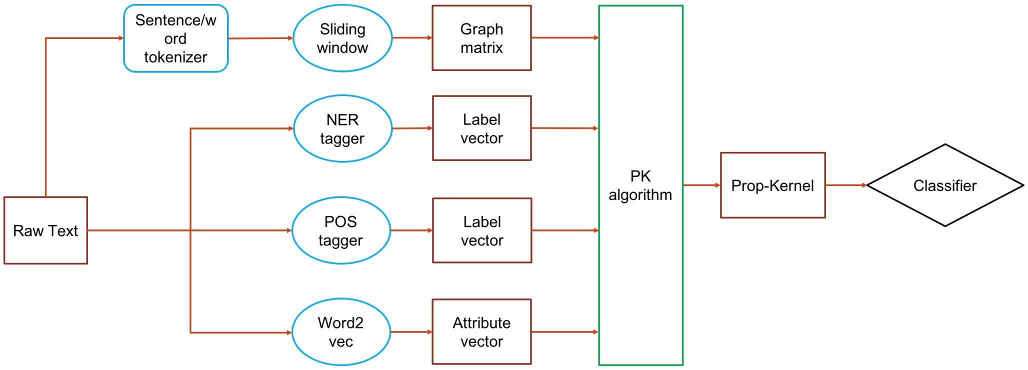


Figure 2: Architecture of the model

3.2 Model Architecture

Figure 2 shows the schema of the model we built for the project. We separate the model into two parallel parts: 1) the graph construction 2) the node labeling and. In the construction of graph, after standard preprocessing operations are performed, the clean text are transformed into a graph-of-words representation via the sliding window. The detailed procedures are: 1) a fixed-size sliding window is placed at the beginning of the text, 2) each time the window will slide in the direction of words' order with one-word step size and 3) all the terms within the sliding are mutually connected to each other. The weight of the edge is set to be the frequency.

In the node labeling, preprocessing operations are not performed. The raw text is directly sent to the PoS tagger & Ner tagger and the word2vec model for getting PoS & Ner labels and word embeddings. Since efficiently and accurately tagging words is not the concern of the project, we utilize open sources to perform the desired functions. In our case, Standford CoreNlp is used for PoS and Ner tagging. We refer to [11] for details. It is worth mentioning that our model is designed in a module-based manner so that it is adaptive to taggers implemented in different ways. Similarly, the neural network can be replaced by other methods without changing in the model design.

Parallelizing the model benefits us in two aspects. Firstly, the interference between node labeling and graph construction can be avoided. The preprocessing operations on the input text, *e.g.* stop-words filtering, is necessary for constructing the graph, but these operations eliminate some useful information for node labeling. For instance, the existence of stop-words results to different term labels in some situation if NER tagging is performed. Secondly, the parallelization enables the model scale for huge dataset. As aforementioned, the attributes of terms are calculated by the neural network, which is known for heavy training time. By parallelizing the model, we reduced the training time. (Maybe too bold to say that?)

Our model is flexible to any off-the-shelf classifier that can be kernelized. For the ease of implementation, we used SVM as our classifier in the experiments.

4 Experiments

In this section, we present the experiments that we conducted to evaluate our methods. We compare graph-of-word containing different nodes' features and discuss the results. We test some common baseline methods on our datasets and present their results as reference.

4.1 Datasets

Two datasets of different types were used in our experiments: one is the dataset consisting of formal texts and the other one is composed of informal texts. We constructed our formal-text dataset from the DBpedia, which is the open library of the Wikipedia articles. We selected the

Class Name	Number of Graphs	Total # of Nodes	Median # of Nodes			
roadtunnel	244	32718	134	10-classes	15-classes	20-classes
sport	243	41504	170			
race course	240	24947	103			
archaea	238	10254	43			
horse trainer	235	39118	166			
vein	233	14622	62			
colour	223	21263	95			
animanga character	219	31397	143			
NCAA athlete	219	23832	108			
brewery	209	19700	94			
cycad	192	14999	78	5-classes		
ligament	187	12242	65			
embryology	185	18188	98			
road junction	181	15868	87			
canadian football team	154	16626	107			
railway tunnel	153	22856	149	20-classes		
guitarist	150	16809	112			
spacecraft	143	16763	117			
beach volleyball player	141	12385	87			
valley	127	14980	117			

Table 1: 20 selected classes from the DBpedia library

Dataset		Properties			
		# Graphs	Median # nodes	Max # nodes	Total # nodes
Wikipedia	binary-class	999	33	208	41002
	5-class	899	42	437	48955
	10-class	2302	56	375	163458
	15-class	3201	52	437	212413
	20-class	3915	54	437	266478
Amazon	Instant_Video	2526	36	809	159382
	Musical_Instrument	2048	44	578	125743

Table 2: Statistics of datasets

articles of 20 classes in the DBpedia library and extracted the long abstract of each article to form the dataset. Table 1 shows the names and statistics of each class. The informal-text dataset that we used was the pre-organized Amazon product reviews. The Amazon dataset is composed of 2 sub-datasets, reviews for instant videos and reviews for musical instrument, in which each of the reviews is attached an overall score from 1 to 5. We treated the score as the indicator of the satisfactoriness reflected by the review, and therefore we divided the reviews into 5 classes in which class 1 means the least satisfactoriness and class 5 means the highest. Table 2 provides the statistics of each sub-dataset.

- **Wikipedia dataset** consists of 3915 articles which are distributed across 20 different classes. The median number of words in each article is 54. We divided the dataset into 5 sub-datasets for simulation of binary and multi-class categorization.
- **Amazon dataset** consists of 4574 articles in total. The median number of words in each article is roughly 40. The amazon dataset is roughly larger than the Wikipedia dataset.

4.2 Implementation

The construction of graphs and node labeling were implemented in *python*, and an interface was built for *matlab*, in which we implemented the Propagation Kernel algorithm. Several open libraries of machine learning and natural language processing were used in our implementation, such as NL-toolkit, Scikit learn and word2vec. For the ease of reproducing the results in our experiments, we uploaded our implementations on Github for evaluation: https://github.com/sunyumark/masterProject. It is worth mentioning that we constructed the Wikipedia datasets ourselves and we are very happy to share them.

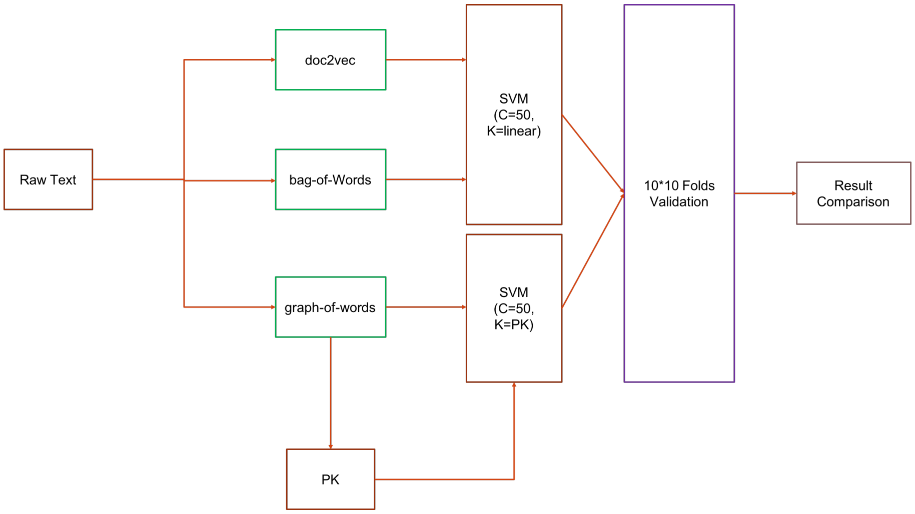


Figure 3: Experiment design

Datasets	Methods						
	doc2vec(linear)	graph-of-words(PK)				baselines	
		attr(10)	attr(10)+ner	attr(10)+pos	pos+ner+attr(10)	count(linear)	tf-idf(linear)
binary-class	0.942	0.992	0.995	**1.000**	0.996	0.999	1.000
5-class	0.886	0.971	0.989	0.990	**0.992**	0.998	0.998
10-class	0.787	0.911	0.961	0.959	**0.973**	0.983	0.993
15-class	0.776	0.823	0.894	0.894	**0.920**	0.974	0.989
20-class	0.753	0.719	0.872	0.842	**0.893**	0.965	0.979
Instant_Video	0.559	0.528	0.558	**0.563**	0.557	0.529	0.571
Musical_Instrument	**0.680**	0.635	0.653	0.661	0.654	0.606	0.679

Table 3: This shows the test results on all datasets. The very left column represents doc2vec, the middle represents the graph-of-words methods, and right shaded columns represent the baseline methods. ***attr(10)*** indicates that words' attributes are used and its dimension is 10. ***+pos*** indicates that the words' label are used. ***(linear)*** indicates linear kernel is used in SVM.

4.3 Evaluating Metric

We evaluate the performance with respect to the classification accuracy, which is calculated in the following formula.

$$acc = \frac{\sum_{x \in D} \delta(correct)}{\sum_{x \in D} 1}, \tag{5}$$

where the numerator represents the number of correctly classified texts and the denominator represents the total number of texts in the dataset. Instead of separating training and testing data from the dataset, we evaluated the test accuracy by conducting 10 × 10 cross-validation.

4.4 Results

As mentioned in previous section, we fixed the classifier to be the SVM. We applied random searching to identify the optimal combination of values in the parameter space, which is composed of window size, maximum iterations of propagation and the cost parameter of SVM. The learned results were 5, 4, 50 for the aforementioned parameters respectively.

Table 3 shows the results on the desired datasets. The best results are in highlighted in bold font. As shown in the table, the attr(10) outperformed the doc2vec on every Wikipedia datasets except for the 20-class one, while the attr(10) was outperformed by the doc2vec on the Amazon datesets. As label information was added into the graph-of-words, the accuracy increased. On some datasets (binary, 5-class and instant video), attr(10)+pos outperformed attr(10)+ner, while attr(10)+ner was better on the other datasets (10-class, 15-class, 20-class and musical instruments). We also tested the graph-of-words containing the combination of the three nodes' features, denoted as attr(10)+pos+ner. The model outperformed others on most of datasets. Common baselines' results are shown in the right shaded columns in table 3. We present these results as reference.

Figure 4 shows the accuracy curves of doc2vec, attr and attr+pos+ner as the dimension of word embeddings changes on 10-class dataset. Vertically, the attr and attr+ner+pos were better than doc2vec at every dimensional level. When dimension was 2, att(2) improved the accuracy by over 60 percent. Horizontally, as the dimension increased, the accuracy curve of attr and attr+ner+pos steady rise slightly while the accuracy curve of doc2vec rise up greatly but was still below the attr and attr+ner+pos. Moreover, we surprisingly observed that attr(2) reached the same accuracy of doc2vec(100).

4.5 Discussion

Structure Extraction The results on two types of datasets reflected that graph-of-words model is capable of capturing the structural information of texts. The accuracy of categorization was largely increased by taking the textual structure into count. In figure 4, over 60 percent improvement was achieved by the graph-of-words, which only contained word embeddings as doc2vec. The improvement can only be caused by the structural information that the graph-of-words abstracted. In the contrary, the graph-of-words faced trouble in discriminating the textual structure from the informal texts, which are usually bad-organized and sometimes difficult for human to read. Overall, the graph-of-words model can absorb textual structure into itself as a type of information.

Information Additivity Although graph-of-words did not performed well merely with the help of nodes attributes, it could utilize other information as bootstrap. As illustrated in the last section, the accuracy of graph-of-words increased as nodes' label were used. Specially, the attr(10)+pos outperformed back doc2vec(10) on the instant video dataset after the nodes' PoS tags were added into the model. The same fact can be observed in figure 4 as well. The accuracy curve of attr+ner+pos was always above attr at every dimensional level. These facts revealed that heterogeneous information is additive under the mechanism of graph-of-words. This property makes graph-of-words as a platform that allows different types of information component to act independently.

Linear Dependency The graph-of-words not only allows different information act independently but also transmits the changes. As shown in figure 4, the accuracy curve of attr rise as that of doc2vec did when the dimension of word embedding increased. We denote the fact as linear dependency of the graph-of-words model. The performance of the graph-of-words linearly depends on the performance of every sub-component of the model.

5 Conclusion

In the master project, we investigated the graph-of-words representation of textual documents and tested its ability to mix and utilize different types of information. We discussed several properties that we discovered of the graph-of-words model: structure extraction, information additivity and linear dependency. However, we can not compete several baselines in our experiments. We explain the failure in the following reasons:

- **limited parameters searching range:** Although we conduct parameter learning in the experiment test, the searching range is limited due to the limit computational power. The searching range may not cover the optimal solution.
- **Window design:** Sliding window favors ease implementation but introduces cyclic structures into the graph, which increases the difficulty to calculate the graph kernel.
- **Inappropriate attribute value setting:** The inside mechanism, attribute hashing, of the propagation kernel limits the choice of attribute value. In the project, words are projected into a hypercube unit, and thus, the difference between words' embeddings is small. It is

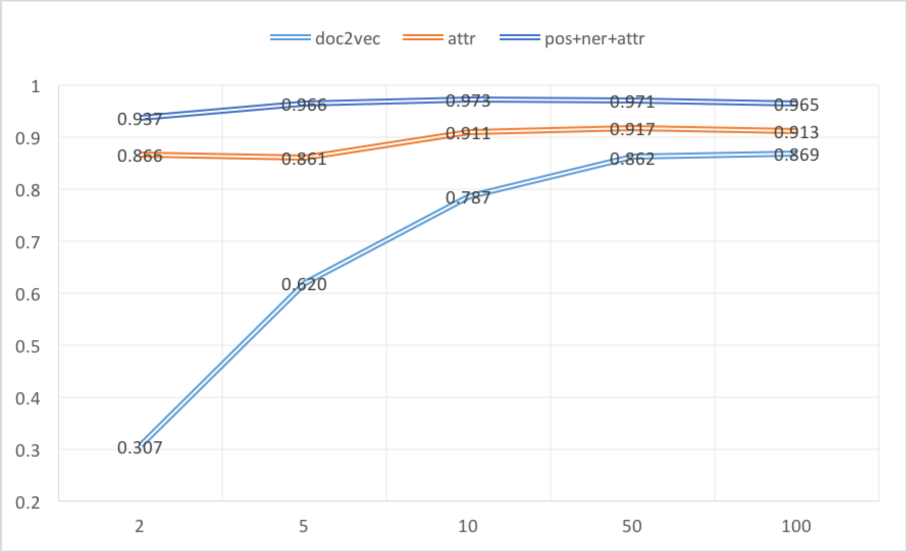


Figure 4: Accuracy curve of doc2vec, attr and pos+ner+attr as the dimension increases.

possible to hash words with different attributes into the same hashing bucket, resulting big differences in the kernel.

We recommend future works to solve these questions.

References

[1] Shumeet Baluja, Deepak Ravichandran, and D. Sivakumar, "Text classification through time: Efficient label propagation in Time-Based graphs," in *Proceeding of the International Conference on Knowledge Discovery and Information Retrieval (KDIR 2009)*. INSTICC, Oct. 2009.

[2] Fabrizio Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, Mar. 2002.

[3] Thorsten Joachims, *Text categorization with Support Vector Machines: Learning with many relevant features*, pp. 137–142, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[4] Quoc V. Le and Tomas Mikolov, "Distributed representations of sentences and documents," *CoRR*, vol. abs/1405.4053, 2014.

[5] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi, "Marginalized kernels between labeled graphs," in *ICML*, 2003.

[6] Aasa Feragen, Niklas Kasenburg, Jens Petersen, Marleen de Bruijne, and Karsten Borgwardt, "Scalable kernels for graphs with continuous attributes," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., pp. 216–224. Curran Associates, Inc., 2013.

[7] Nils Kriege and Petra Mutzel, "Subgraph Matching Kernels for Attributed Graphs," in *International Conference on Machine Learning (ICML)*, 2012, to appear.

[8] Alex Markov, Mark Last, and Abraham Kandel, *Fast Categorization of Web Documents Represented by Graphs*, pp. 56–71, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013.

[10] Marion Neumann, Roman Garnett, Christian Bauckhage, and Kristian Kersting, "Propagation kernels: efficient graph kernels from propagated information," *Machine Learning*, vol. 102, no. 2, pp. 209–245, Feb 2016.

[11] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60.