Text Categorization as a Graph Classification Problem

François Rousseau **Emmanouil Kiagias** **Michalis Vazirgiannis**

LIX, École Polytechnique, France

Abstract

In this paper, we consider the task of text categorization as a graph classification problem. By representing textual documents as graph-of-words instead of historical n-gram bag-of-words, we extract more discriminative features that correspond to long-distance n-grams through frequent subgraph mining. Moreover, by capitalizing on the concept of k-core, we reduce the graph representation to its densest part – its main core – speeding up the feature extraction step for little to no cost in prediction performances. Experiments on four standard text classification datasets show statistically significant higher accuracy and macro-averaged F1-score compared to baseline approaches.

1 Introduction

The task of text categorization finds applications in a wide variety of domains, from news filtering and document organization to opinion mining and spam detection. With the ever-growing quantity of information available online nowadays, it is crucial to provide effective systems capable of classifying text in a timely fashion. Compared to other application domains of classification, its specificity lies in its high number of features, its sparse feature vectors and its skewed multiclass scenario. For instance, when dealing with thousands of news articles, it is not uncommon to have millions of n-gram features, only a few hundreds actually present in each document and tens of class labels – some of them with thousands of articles and some others will only a few hundreds. These particularities have to be taken into account when envisaging a different representation for a document and in our case when considering the task as a graph classification problem.

Graphs are powerful data structures that are used to represent complex information about entities and interaction between them and we think text makes no exception. Historically, following the traditional bag-of-words representation, unigrams have been considered as the natural features and later extended to n-grams to capture some *word dependency* and *word order*. However, n-grams correspond to sequences of words and thus fail to capture *word inversion* and *subset matching* (e. g., "article about news" vs. "news article"). We believe graphs can help solve these issues like they did for instance with chemical compounds where repeating substructure patterns are good indicators of belonging to one particular class, e. g., predicting carcinogenicity in molecules (Helma et al., 2001). Graph classification has received a lot of attention this past decade and various techniques have been developed to deal with the task but rarely applied on textual data and at its scale.

In our work, we explored a graph representation of text, namely graph-of-words, to challenge the traditional bag-of-words representation and help better classify textual documents into categories. We first trained a classifier using frequent subgraphs as features for increased effectiveness. We then reduced each graph-of-words to its main core before mining the features for increased efficiency. Finally, we also used this technique to reduce the total number of n-gram features considered in the baselines for little to no loss in prediction performances.

The rest of the paper is organized as follows. Section 2 provides a review of the related work. Section 3 defines the preliminary concepts upon which our work is built. Section 4 introduces the proposed approaches. Section 5 describes the experimental settings and presents the results we obtained on four standard datasets. Finally, Section 6 concludes our paper and mentions future work directions.

2 Related work

In this section, we present the related work in *text categorization*, *graph classification* and the combination of the two fields like in our case.

2.1 Text categorization

Text categorization, a.k.a. *text classification*, corresponds to the task of automatically predicting the class label of a given textual document. We refer to (Sebastiani, 2002) for an in-depth review of the earliest works in the field and (Aggarwal and Zhai, 2012) for a survey of the more recent works that capitalize on additional meta-information. We note in particular the seminal work of Joachims (1998) who was the first to propose the use of a linear SVM with TF×IDF term features for the task. This approach is one of the standard baselines because of its simplicity yet effectiveness (unsupervised n-gram feature mining followed by standard supervised learning). Another popular approach is the use of Naive Bayes and its multiple variants (McCallum and Nigam, 1998), in particular for the subtask of *spam detection* (Androutsopoulos et al., 2000). Finally, there are a couple of works such as (Hassan et al., 2007) that used the graph-of-words representation to propose alternative weights for the n-gram features but still without considering the task as a graph classification problem.

2.2 Graph classification

Graph classification corresponds to the task of automatically predicting the class label of a given graph. The learning part in itself does not differ from other supervised learning problems and most proposed methods deal with the feature extraction part. They fall into two main categories: approaches that consider *subgraphs as features* and *graph kernels*.

2.2.1 Subgraphs as features

The main idea is to mine frequent subgraphs and use them as features for classification, be it with Adaboost (Kudo et al., 2004) or a linear SVM (Deshpande et al., 2005). Indeed, most datasets that were used in the associated experiments correspond to chemical compounds where repeating substructure patterns are good indicators of belonging to one particular class. Some popular graph pattern mining algorithms are gSpan (Yan and Han, 2002), FFSM (Huan et al., 2003) and Gaston (Nijssen and Kok, 2004). The number of frequent subgraphs can be enormous, especially for large graph collections, and handling such a feature set can be very expensive. To overcome this issue, recent works have proposed to retain or even only mine the discriminative subgraphs, i. e. features that contribute to the classification decision, in particular gBoost (Saigo et al., 2009), CORK (Thoma et al., 2009) and GAIA (Jin et al., 2010). However, when experimenting, gBoost did not converge on our larger datasets while GAIA and CORK consider subgraphs of node size at least 2, which exclude unigrams, resulting in poorer performances. Moreover, all these approaches have been developed for binary classification, which meant mining features as many times as the number of classes instead of just once (*one-vs-all* learning strategy). In this paper, we tackle the scalability issue differently through an unsupervised feature selection approach to reduce the size of the graphs and a fortiori the number of frequent subgraphs.

2.2.2 Graph kernels

Gärtner et al. (2003) proposed the first kernels *between* graphs (as opposed to previous kernels *on* graphs, i. e. between nodes) based on either random walks or cycles to tackle the problem of classification between graphs. In parallel, the idea of marginalized kernels was extended to graphs by Kashima et al. (2003) and by Mahé et al. (2004). We refer to (Vishwanathan et al., 2010) for an in-depth review of the topic and in particular its limitations in terms of number of unique node labels, which make them unsuitable for our problem as tested in practice (limited to a few tens of unique labels compared to hundreds of thousands for us).

2.3 Similar works

The work of Markov et al. (2007) is perhaps the closest to ours since they also perform subgraph feature mining on graph-of-words representations but with non-standard datasets and baselines. The works of Jiang et al. (2010) and Arora et al. (2010) are also related but their representations are different and closer to parse and dependency trees used as base features for text categorization by Kudo and Matsumoto (2004) and Matsumoto et al. (2005). Moreover, they do not discuss the choice of the support value, which controls the total number of features and can potentially lead to millions of subgraphs on standard datasets.

3 Preliminary concepts

In this section, we introduce the preliminary concepts upon which our work is built.

3.1 Graph-of-words

We model a textual document as a *graph-of-words*, which corresponds to a graph whose vertices represent unique terms of the document and whose edges represent co-occurrences between the terms within a fixed-size sliding window. The underlying assumption is that all the words present in a document have some undirected relationships with the others, modulo a window size outside of which the relationship is not considered. This representation was first used in keyword extraction and summarization (Ohsawa et al., 1998; Mihalcea and Tarau, 2004) and more recently in ad hoc IR (Blanco and Lioma, 2012; Rousseau and Vazirgiannis, 2013). We refer to (Blanco and Lioma, 2012) for an in-depth review of the graph representations of text in NLP.

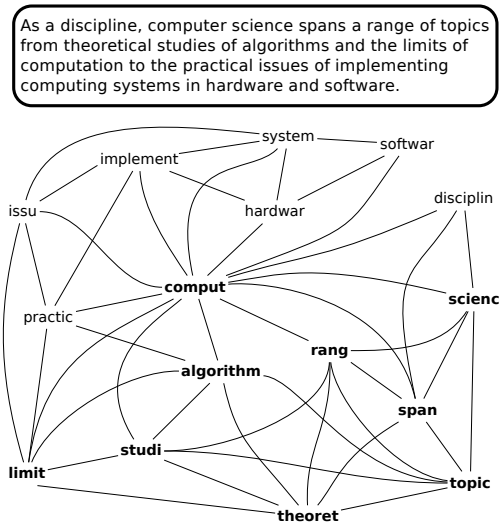

Figure 1: Graph-of-words representation of a textual document – in bold font, its main core.

Figure 1 illustrates the graph-of-words representation of a textual document. The vertices correspond to the remaining terms after standard preprocessing steps have been applied (tokenization, stop word removal and stemming). The undirected edges were drawn between terms co-occurring within a sliding window over the processed text of size 4, value consistently reported as working well in the references aforementioned and validated in our experiments. Edge direction was used by Filippova (2010) so as to extract valid sentences but not here in order to capture some word inversion.

Note that for small-enough window sizes (which is typically the case in practice), we can consider that two terms linked represent a *long-distance bigram* (Bassiou and Kotropoulos, 2010), if not a bigram. Furthermore, by extending the denomination, we can consider that a subgraph of size n is a *long-distance n-gram*, if not an n-gram. Indeed, the nodes belonging to a subgraph do not necessarily appear in a sequence in the document like for a n-gram. Moreover, this enables us to "merge" together n-grams that share the same terms but maybe not in the same order. In the experiments, by abusing the terminology, we will refer to them as n-grams to adopt a common terminology with the baseline approaches.

3.2 Node/edge labels and subgraph matching

In graph classification, it is common to introduce a node labeling function μ to map a node id to its label. For instance, consider the case of chemical compounds (e. g., the benzene C_6H_6). Then in its graph representation (its "structural formula"), it is crucial to differentiate between the multiple nodes labeled the same (e. g., C or H). In the case of graph-of-words, node labels are unique inside a graph since they represent unique terms of the document and we can therefore omit these functions since they are injective in our case and we can substitute node ids for node labels. In particular, the general problem of *subgraph matching*, which defines an isomorphism between a graph and a subgraph and is NP-complete (Garey and Johnson, 1990), can be reduced to a polynomial problem when node labels are unique. In our experiments, we used the standard algorithm VF2 developed by Cordella et al. (2001).

3.3 K-core and main core

Seidman (1983) defined the k-core of a graph as the maximal connected subgraph whose vertices are at least of degree k within the subgraph. The non-empty k-core of largest k is called the *main core* and corresponds to the most cohesive set(s) of vertices. The corresponding value of k may differ from one graph to another. Batagelj and Zaveršnik (2003) proposed an algorithm to extract the main core of an unweighted graph in time linear in the number of edges, complexity similar in our case to the other NLP preprocessing steps. Bold font on Figure 1 indicates that a vertex belongs to the main core of the graph.

4 Graph-of-words classification

In this section, we present our work and the several approaches we explored, from unsupervised feature mining using gSpan to propose more discriminative features than standard n-grams to unsupervised feature selection using k-core to reduce the total number of subgraph and n-gram features.

4.1 Unsupervised feature mining using gSpan

We considered the task of text categorization as a graph classification problem by representing textual documents as graph-of-words and then extracting subgraph features to train a graph classifier. Each document is a separate graph-of-words and the collection of documents thus corresponds to a set of graphs. Therefore, for larger datasets, the total number of graphs increases but not the average graph size (the average number of unique terms in a text), assuming homogeneous datasets.

Because the total number of unique node labels corresponds to the number of unique terms in the collection in our case, graph kernels are not suitable for us as verified in practice using the MATLAB code made available by Shervashidze (2009). We therefore only explored the methods that consider subgraphs as features. Repeating substructure patterns between graphs are intuitively good candidates for classification since, at least for chemical compounds, shared subparts of molecules are good indicators of belonging to one particular class. We assumed it would the same for text. Indeed, subgraphs of graph-of-words correspond to sets of words co-occurring together, just not necessarily always as the same sequence like for n-grams – it can be seen as a relaxed definition of a n-gram to capture additional variants.

We used gSpan (graph-based Substructure pattern (Yan and Han, 2002)) as frequent subgraph miner like (Jiang et al., 2010; Arora et al., 2010) mostly because of its fast available C++ implementation from gBoost (Saigo et al., 2009). Briefly, the key idea behind gSpan is that instead of enumerating all the subgraphs and testing for isomorphism throughout the collection, it first builds for each graph a lexicographic order of all the edges using depth-first-search (DFS) traversal and assigns to it a unique minimum DFS code. Based on all these DFS codes, a hierarchical search tree is constructed at the collection-level. By pre-order traversal of this tree, gSpan discovers all frequent subgraphs with required support.

Consider the set of all subgraphs in the collection of graphs, which corresponds to the set of all potential features. Note that there may be overlapping (subgraphs sharing nodes/edges) and redundant (subgraphs included in others) features. Because its size is exponential in the number of edges (just like the number of n-grams is exponential in n), it is common to only retain/mine the most frequent subgraphs (again just like for n-grams with a minimum document frequency (Fürnkranz, 1998; Joachims, 1998)). This is controlled via a parameter known as the *support*, which sets the minimum number of graphs in which a given subgraph has to appear to be considered as a feature, i. e. the number of subgraph matches in the collection. Here, since node labels are unique inside a graph, we do not have to consider multiple occurrences of the same subgraph in a given graph. The lower the support, the more features selected/considered but the more expensive the mining and the training (not only in time spent for the learning but also for the feature vector generation).

4.2 Unsupervised support selection

The optimal value for the support can be learned through cross-validation so as to maximize the prediction accuracy of the subsequent classifier, making the whole feature mining process *supervised*. But if we consider that the classifier can only improve its goodness of fit with more features (the sets of features being nested as the support varies), it is likely that the lowest support will lead to the best test accuracy; assuming subsequent regularization to prevent overfitting. However, this will come at the cost of an exponential number of features as observed in practice. Indeed, as the support decreases, the number of features increases slightly up until a point where it increases exponentially, which makes both the feature vector generation and the learning expensive, especially with multiple classes. Moreover, we observed that the prediction performances did not benefit that much from using all the possible features (support of 1) as opposed to a more manageable number of features corresponding to a higher support. Therefore, we propose to select the support using the so-called *elbow method*. This is an unsupervised empirical method initially developed for selecting the number of clusters in k-means (Thorndike, 1953). Figure 3 (upper plots) in Section 5 illustrates this process.

4.3 Considered classifiers

In text categorization, standard baseline classifiers include k-nearest neighbors (kNN) (Larkey and Croft, 1996), Naive Bayes (NB) (McCallum and Nigam, 1998) and linear Support Vector Machines (SVM) (Joachims, 1998) with the latter performing the best on n-gram features as verified in our experiments. Since our subgraph features correspond to "long-distance n-grams", we used linear SVMs as our classifiers in all our experiments – the goal of our work being to explore and propose better features rather than a different classifier.

4.4 Multiclass scenario

In standard binary graph classification (e. g., predicting chemical compounds' carcinogenicity as either positive or negative (Helma et al., 2001)), feature mining is performed on the whole graph collection as we expect the mined features to be able to discriminate between the two classes (thus producing a good classifier). However, for the task of text categorization, there are usually more than two classes (e. g., 118 categories of news articles for the Reuters-21578 dataset) and with a skewed class distribution (e. g., a lot more news related to "acquisition" than to "grain"). Therefore, a single support value might lead to some classes generating a tremendous number of features (e. g., hundreds of thousands of frequent subgraphs) and some others only a few (e. g., a few hundreds subgraphs) resulting in a skewed and non-discriminative feature set. To include discriminative features for these *minority* classes, we would need an extremely low support resulting in an exponential number of features because of the *majority* classes. For these reasons, we decided to mine frequent subgraphs per class using the same relative support (%) and then aggregating each feature set into a global one at the cost of a *supervised* process (but which still avoids cross-validated parameter tuning). This was not needed for the tasks of spam detection and opinion mining since the corresponding datasets consist of only two balanced classes.

4.5 Main core mining using gSpan

Since the main drawback of mining frequent subgraphs for text categorization rather than chemical compound classification is the very high number of possible subgraphs because of the size of the graphs and the total number of graphs (more than 10x in both cases), we thought of ways to reduce the graphs' sizes while retaining as much classification information as possible.

The graph-of-words representation is designed to capture dependency between words, i. e. dependency between features in the context of machine learning but at the document-level. Initially, we wanted to capture recurring sets of words (i. e. take into account word inversion and subset matching) and not just sequences of words like with n-grams. In terms of subgraphs, this means words that co-occur with each other and form a dense subgraph as opposed to a path like for a n-gram. Therefore, when reducing the graphs, we need to keep their densest part(s) and that is why we considered extracting their main cores. Compared to other density-based algorithms, retaining the main core of a graph has the advantage of being linear in the number of edges, i. e. in the number of unique terms in a document in our case (the number of edges is at most the number of nodes times the fixed size of the sliding window, a small constant in practice).

4.6 Unsupervised n-gram feature selection

Similarly to (Hassan et al., 2007) that used graph-of-words to propose alternative weights for the n-gram features, we can capitalize on main core retention to still extract binary n-gram features for classification but considering only the terms belonging to the main core of each document. Because some terms never belong to any main core of any document, the dimension of the overall feature space decreases. Additionally, since a document is only represented by a subset of its original terms, the number of non-zero feature values per document also decreases, which matters for SVM, even for the linear kernel, when considering the dual formulation or in the primal with more recent optimization techniques (Joachims, 2006).

Compared to most existing feature selection techniques in the field (Yang and Pedersen, 1997), it is *unsupervised* and *corpus-independent* as it does not rely on any labeled data like IG, MI or χ^2 nor any collection-wide statistics like IDF, which can be of interest for large-scale text categorization in order to process documents in parallel, independently of each other. In some sense, it is similar to what Özgür et al. (2005) proposed with corpus-based and class-based keyword selection for text classification except that we use here *document-based* keyword selection following the approach from Rousseau and Vazirgiannis (2015).

5 Experiments

In this section we present the experiments we conducted to validate our approaches.

5.1 Datasets

We used four standard text datasets: two for multi-class document categorization (WebKB and R8), one for spam detection (LingSpam) and one for opinion mining (Amazon) so as to cover all the main subtasks of text categorization:

- `WebKB`: 4 most frequent categories among labeled webpages from various CS departments – split into 2,803 for training and 1,396 for test (Cardoso-Cachopo, 2007, p. 39–41).
- `R8`: 8 most frequent categories of Reuters-21578, a set of labeled news articles from the 1987 Reuters newswire – split into 5,485 for training and 2,189 for test (Debole and Sebastiani, 2005).
- `LingSpam`: 2,893 emails classified as spam or legitimate messages – split into 10 sets for 10-fold cross validation (Androutsopoulos et al., 2000).
- `Amazon`: 8,000 product reviews over four different sub-collections (books, DVDs, electronics and kitchen appliances) classified as positive or negative – split into 1,600 for training and 400 for test each (Blitzer et al., 2007).

5.2 Implementation

We developed our approaches mostly in Python using the `igraph` library (Csardi and Nepusz, 2006) for the graph representation and main core extraction. For unsupervised subgraph feature mining, we used the C++ implementation of gSpan from gBoost (Saigo et al., 2009). Finally for classification and standard n-gram text categorization we used `scikit` (Pedregosa et al., 2011), a standard Python machine learning library.

5.3 Evaluation metrics

To evaluate the performance of our proposed approaches over standard baselines, we computed on the test set both the micro- and macro-average F1-score. Because we are dealing with single-label classification, the micro-average F1-score corresponds to the accuracy and is a measure of the overall prediction effectiveness (Manning et al., 2008, p. 281). Conversely, the macro-average F1-score takes into account the skewed class label distributions by weighting each class uniformly. The statistical significance of improvement in accuracy over the n-gram SVM baseline was assessed using the micro sign test ($p < 0.05$) (Yang and Liu, 1999). For the Amazon dataset, we report the average of each metric over the four sub-collections.

Dataset	# subgraphs before	# subgraphs after	reduction
WebKB	30,868	10,113	67 %
R8	39,428	11,373	71 %
LingSpam	54,779	15,514	72 %
Amazon	16,415	8,745	47 %

Dataset	# n-grams before	# n-grams after	reduction
WebKB	1,849,848	735,447	60 %
R8	1,604,280	788,465	51 %
LingSpam	2,733,043	1,016,061	63 %
Amazon	583,457	376,664	35 %

Table 1: Total number of features (n-grams or subgraphs) vs. number of features present only in main cores along with the reduction of the dimension of the feature space on all four datasets.

5.4 Results

Table 2 shows the results on the four considered datasets. The first three rows correspond to the baselines: unsupervised n-gram feature extraction and then supervised learning using kNN, NB (Multinomial but Bernoulli yields similar results) and linear SVM. The last three rows correspond to our approaches.

In our first approach, denoted as "gSpan + SVM", we mine frequent subgraphs (gSpan) as features and then train a linear SVM. These features correspond to long-distance n-grams. This leads to the best results in text categorization on almost all datasets (all if we compare to baseline methods), in particular on multiclass document categorization (R8 and WebKB).

In our second approach, denoted as "MC + gSpan + SVM", we repeat the same procedure except that we mine frequent subgraphs (gSpan) from the main core (MC) of each graph-of-words and then train an SVM on the resulting features. Main cores can vary from 1-core to 12-core depending on the graph structure, 5-core and 6-core being the most frequent (more than 60%). This yields results similar to the SVM baseline for a faster mining and training compared to gSpan + SVM. Table 1 (upper table) shows the reduction in the dimension of the feature space and we see

Table 2: Test accuracy and macro-average F1-score on four standard datasets. Bold font marks the best performance in a column. * indicates statistical significance at $p < 0.05$ using micro sign test with regards to the SVM baseline of the same column. MC corresponds to unsupervised feature selection using the main core of each graph-of-words to extract n-gram and subgraph features. gSpan mining support values are 1.6% (WebKB), 7% (R8), 4% (LingSpam) and 0.5% (Amazon).

Method \ Dataset	WebKB		R8		LingSpam		Amazon	
	Accuracy	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
kNN (k=5)	0.679	0.617	0.894	0.705	0.910	0.774	0.512	0.644
NB (Multinomial)	0.866	0.861	0.934	0.839	0.990	0.971	0.768	0.767
linear SVM	0.889	0.871	0.947	0.858	**0.991**	**0.973**	0.792	0.790
gSpan + SVM	**0.912***	**0.882**	**0.955***	**0.864**	**0.991**	0.972	0.798*	0.795
MC + gSpan + SVM	0.901*	0.871	0.949*	0.858	0.990	**0.973**	**0.800***	**0.798**
MC + SVM	0.872	0.863	0.937	0.849	0.990	0.972	0.786	0.774

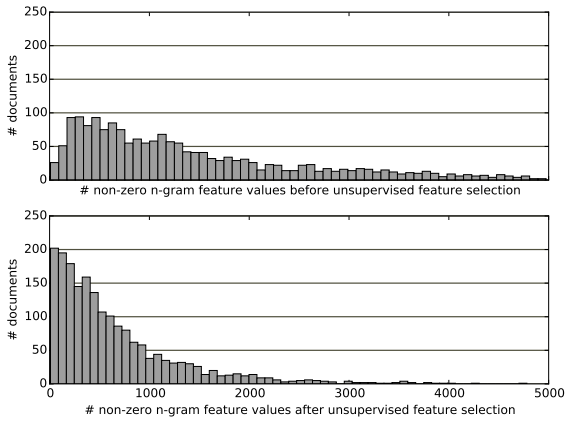

Figure 2: Distribution of non-zero n-gram feature values before and after unsupervised feature selection (main core retention) on R8 dataset.

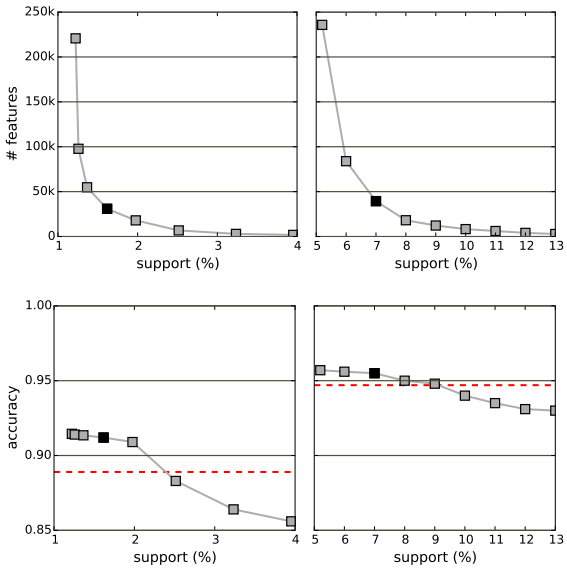

Figure 3: Number of subgraph features/accuracy in test per support (%) on WebKB (left) and R8 (right) datasets: in black, the selected support value chosen via the elbow method and in red, the accuracy in test for the SVM baseline.

that on average less than 60% of the subgraphs are kept for little to no cost in prediction effectiveness.

In our final approach, denoted as "MC + SVM", we performed unsupervised feature selection by keeping the terms appearing in the main core (MC) of each document's graph-of-words representation and then extracted standard n-gram features. Table 1 (lower table) shows the reduction in the dimension of the feature space and we see that on average less than half the n-grams remain. Figure 2 shows the distribution of non-zero features before and after the feature selection on the R8 dataset. Similar changes in distribution can be observed on the other datasets, from a right-tail Gaussian to a power law distribution as expected from the main core retention. Table 2 shows that the main core retention has little to no cost in accuracy and F1-score but can reduce drastically the feature space and the number of non-zero values per document.

5.5 Unsupervised support selection

Figure 3 above illustrates the unsupervised heuristic (elbow method) we used to select the support value, which corresponds to the minimum number of graphs in which a subgraph has to appear to be considered frequent. We noticed that as the support decreases, the number of features increases slightly up until a point where it increases exponentially. This support value, highlighted in black on the figure and chosen before taking into account the class label, is the value we used in our experiments and for which we report the results in Table 1 and 2. The lower plots provide evidence

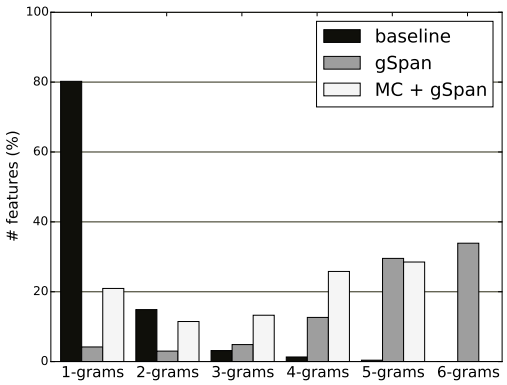

Figure 4: Distribution of n-grams (standard and long-distance ones) among all the features on WebKB dataset.

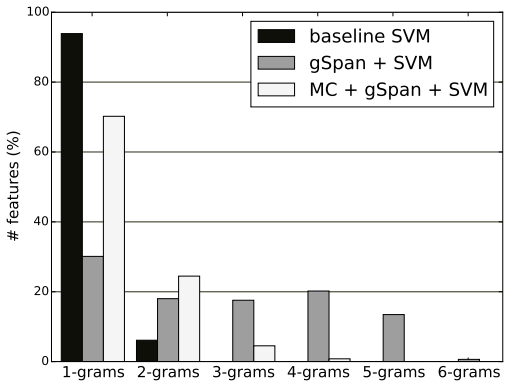

Figure 5: Distribution of n-grams (standard and long-distance ones) among the top 5% most discriminative features for SVM on WebKB dataset.

that the elbow method helps selecting in an unsupervised manner a support that leads to the best or close to the best accuracy.

5.6 Distribution of mined n-grams

In order to gain more insights on why the long-distance n-grams mined with gSpan result in better classification performances than the baseline n-grams, we computed the distribution of the number of unigrams, bigrams, etc. up to 6-grams in the traditional feature set and ours (Figure 4) as well as in the top 5% features that contribute the most to the classification decision of the trained SVM (Figure 5). Again, a long-distance n-gram corresponds to a subgraph of size n in a graph-of-words and can be seen as a relaxed definition of the traditional n-gram, one that takes into account word inversion for instance. To obtain comparable results, we considered for the baseline n-grams with a minimum document frequency equal to the support. Otherwise, by definition, there are at least as many bigrams as there are unigrams and so forth.

Figure 4 shows that our approaches mine way more n-grams than unigrams compared to the baseline. This happens because with graph-of-words a subgraph of size n corresponds to a set of n terms while with bag-of-words a n-gram corresponds to a sequence of n terms. Note that even when restricting the subgraphs to the main cores, there are still more higher order n-grams mined.

Figure 5 shows that the higher order n-grams still contribute indeed to the classification decision and in higher proportion than with the baseline, even when restricting to the main cores. For instance, on the R8 dataset, {bank, base, rate} was a discriminative (top 5% SVM features) long-distance 3-gram for the category "interest" and occurred in documents in the form of "barclays bank cut its base lending rate", "midland bank matches its base rate" and "base rate of natwest bank dropped", pattern that would be hard to capture with traditional n-gram bag-of-words.

5.7 Timing

With an Intel Core i5-3317U clocking at 2.6GHz and 8GB of RAM, mining the subgraph features with gSpan takes on average 30s for the selected support. It can take several hours with lower support and goes down to 5s using the main cores.

6 Conclusion

In this paper, we tackled the task of text categorization by representing documents as graph-of-words and then considering the problem as a graph classification one. We were able to extract more discriminative features that correspond to long-distance n-grams through frequent subgraph mining. Experiments on four standard datasets show statistically significant higher accuracy and macro-averaged F1-score compared to baselines.

To the best of our knowledge, graph classification has never been tested at that scale – thousands of graphs and tens of thousands of unique node labels – and also in the multiclass scenario. For these reasons, we could not capitalize on all standard methods. In particular, we believe new kernels that support a very high number of unique node labels could yield even better performances.

References

Charu C. Aggarwal and ChengXiang Zhai. 2012. A Survey of Text Classification Algorithms. In *Mining Text Data*, pages 163–222.

Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras, and Constantine D. Spyropoulos. 2000. An Evaluation of Naive Bayesian Anti-Spam Filtering. In *Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning*, pages 9–17.

Shilpa Arora, Elijah Mayfield, Carolyn Penstein-Rosé, and Eric Nyberg. 2010. Sentiment Classification Using Automatically Extracted Subgraph Features. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, CAAGET '10, pages 131–139.

Nikoletta Bassiou and Constantine Kotropoulos. 2010. Word Clustering Using PLSA Enhanced with Long Distance Bigrams. In *Proceedings of the 20th International Conference on Pattern Recognition*, ICPR '10, pages 4226–4229.

Vladimir Batagelj and Matjaž Zaversnik. 2003. An O(m) Algorithm for Cores Decomposition of Networks. *The Computing Research Repository (CoRR)*, cs.DS/0310049.

Roi Blanco and Christina Lioma. 2012. Graph-based term weighting for information retrieval. *Information Retrieval*, 15(1):54–92.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07, pages 440–447.

Ana Cardoso-Cachopo. 2007. *Improving Methods for Single-label Text Categorization*. Ph.D. thesis, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal.

Luigi Pietro Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. 2001. An improved algorithm for matching large graphs. In *Proceedings of the 3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 149–159.

Gabor Csardi and Tamas Nepusz. 2006. The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9.

Franca Debole and Fabrizio Sebastiani. 2005. An Analysis of the Relative Hardness of Reuters-21578 Subsets: Research Articles. *Journal of the American Society for Information Science and Technology*, 56(6):584–596.

Mukund Deshpande, Michihiro Kuramochi, Nikil Wale, and George Karypis. 2005. Frequent Substructure-Based Approaches for Classifying Chemical Compounds. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1036–1050.

Katja Filippova. 2010. Multi-sentence Compression: Finding Shortest Paths in Word Graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 322–330.

Johannes Fürnkranz. 1998. A study using n-gram features for text categorization. Technical Report OEFAI-TR-98-30, Austrian Research Institute for Artificial Intelligence.

Michael R. Garey and David S. Johnson. 1990. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co.

Thomas Gärtner, Peter Flach, and Stefan Wrobel. 2003. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the Annual Conference on Computational Learning Theory*, COLT '03, pages 129–143.

Samer Hassan, Rada Mihalcea, and Carmen Banea. 2007. Random-Walk Term Weighting for Improved Text Classification. In *Proceedings of the International Conference on Semantic Computing*, ICSC '07, pages 242–249.

Christoph Helma, Ross D. King, Stefan Kramer, and Ashwin Srinivasan. 2001. The predictive toxicology challenge 2000–2001. *Bioinformatics*, 17(1):107–108.

Jun Huan, Wei Wang, and Jan Prins. 2003. Efficient Mining of Frequent Subgraphs in the Presence of Isomorphism. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, ICDM '03, pages 549–552.

Chuntao Jiang, Frans Coenen, Robert Sanderson, and Michele Zito. 2010. Text classification using graph mining-based feature extraction. *Knowledge-Based Systems*, 23(4):302–308.

Ning Jin, Calvin Young, and Wei Wang. 2010. GAIA: graph classification using evolutionary computation. In *Proceedings of the 2010 ACM SIGMOD international conference on Management of data*, SIGMOD '10, pages 879–890.

Thorsten Joachims. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 137–142.

Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data mining*, KDD '06, pages 217–226.

Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. 2003. Marginalized kernels between labeled graphs. In *Proceedings of the 20th International Conference on Machine Learning*, volume 3 of *ICML '03*, pages 321–328.

Taku Kudo and Yuji Matsumoto. 2004. A Boosting Algorithm for Classification of Semi-Structured Text. In *Proceedings of the 9th Conference on Empirical Methods in Natural Language Processing*, volume 4 of *EMNLP '04*, pages 301–308.

Taku Kudo, Eisaku Maeda, and Yuji Matsumoto. 2004. An application of boosting to graph classification. In *Advances in Neural Information Processing Systems 17*, NIPS '04, pages 729–736.

Leah S. Larkey and W. Bruce Croft. 1996. Combining Classifiers in Text Categorization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '96, pages 289–297.

Pierre Mahé, Nobuhisa Ueda, Tatsuya Akutsu, Jean-Luc Perret, and Jean-Philippe Vert. 2004. Extensions of marginalized graph kernels. In *Proceedings of the 21st International Conference on Machine Learning*, ICML '04, pages 70–78.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Alex Markov, Mark Last, and Abraham Kandel. 2007. Fast Categorization of Web Documents Represented by Graphs. In *Advances in Web Mining and Web Usage Analysis*, number 4811 in Lecture Notes in Artificial Intelligence, pages 56–71.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees. In *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD '05, pages 301–311.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for Naive Bayes text classification. In *Proceedings of the AAAI workshop on learning for text categorization*, AAAI '98, pages 41–48.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Texts. In *Proceedings of the 9th Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 404–411.

Siegfried Nijssen and Joost N. Kok. 2004. A Quickstart in Frequent Structure Mining Can Make a Difference. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge Discovery and Data mining*, KDD '04, pages 647–652.

Yukio Ohsawa, Nels E. Benson, and Masahiko Yachida. 1998. KeyGraph: Automatic Indexing by Co-occurrence Graph Based on Building Construction Metaphor. In *Proceedings of the Advances in Digital Libraries Conference*, ADL '98, pages 12–18.

Arzucan Özgür, Levent Özgür, and Tunga Güngör. 2005. Text Categorization with Class-based and Corpus-based Keyword Selection. In *Proceedings of the 20th International Conference on Computer and Information Sciences*, ISCIS '05, pages 606–615.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830.

François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and TW-IDF: New Approach to Ad Hoc IR. In *Proceedings of the 22nd ACM international conference on Information and knowledge management*, CIKM '13, pages 59–68.

François Rousseau and Michalis Vazirgiannis. 2015. Main Core Retention on Graph-of-words for Single-Document Keyword Extraction. In *Proceedings of the 37th European Conference on Information Retrieval*, ECIR '15, pages 382–393.

Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda. 2009. gBoost: a mathematical programming approach to graph classification and regression. *Machine Learning*, 75(1):69–89.

Fabrizio Sebastiani. 2002. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34(1):1–47.

Stephen B. Seidman. 1983. Network structure and minimum degree. *Social Networks*, 5:269–287.

Nino Shervashidze. Visited on 30/05/2015. Graph kernels. http://www.di.ens.fr/~shervashidze/code.html.

Marisa Thoma, Hong Cheng, Arthur Gretton, Jiawei Han, Hans-Peter Kriegel, Alexander J. Smola, Le Song, Philip S. Yu, Xifeng Yan, and Karsten M. Borgwardt. 2009. Near-optimal Supervised Feature Selection among Frequent Subgraphs. In *Proceedings of the SIAM International Conference on Data Mining*, SDM '09, pages 1076–1087.

Robert Thorndike. 1953. Who belongs in the family? *Psychometrika*, 18(4):267–276.

S. V. N. Vishwanathan, Nicol N. Schraudolph, Risi Kondor, and Karsten M. Borgwardt. 2010. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242.

Xifeng Yan and Jiawei Han. 2002. gspan: Graph-based substructure pattern mining. In *Proceedings of the 2nd IEEE International Conference on Data Mining*, ICDM '02, pages 721–724.

Yiming Yang and Xin Liu. 1999. A Re-examination of Text Categorization Methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 42–49.

Yiming Yang and J. O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the 14th International Conference on Machine Learning*, ICML '97, pages 412–420.