Multivariate Gaussian Document Representation from Word Embeddings for Text Categorization

Giannis Nikolentzos
École Polytechnique and AUEB
nikolentzos@aueb.gr

Polykarpos Meladianos
École Polytechnique and AUEB
pmeladianos@aueb.gr

François Rousseau
École Polytechnique
rousseau@lix.polytechnique.fr

Michalis Vazirgiannis
École Polytechnique and AUEB
mvazirg@aueb.gr

Yannis Stavrakas
IMIS / RC ATHENA
yannis@imis.athena-innovation.gr

Abstract

Recently, there has been a lot of activity in learning distributed representations of words in vector spaces. Although there are models capable of learning high-quality distributed representations of words, how to generate vector representations of the same quality for phrases or documents still remains a challenge. In this paper, we propose to model each document as a multivariate Gaussian distribution based on the distributed representations of its words. We then measure the similarity between two documents based on the similarity of their distributions. Experiments on eight standard text categorization datasets demonstrate the effectiveness of the proposed approach in comparison with state-of-the-art methods.

1 Introduction

During the past decade, there has been a significant increase in the availability of textual information mainly due to the exploding popularity of the World Wide Web. This tremendous amount of textual information growth has established the need for the development of effective text-mining approaches.

Traditionally, documents are represented as bag-of-words (BOW) vectors. The BOW representation is very simple and it has proven effective in easy and moderate tasks, however, for more demanding tasks, such as short text modeling, its performance drops significantly.

In order to overcome the weakness of BOW, researchers proposed methods that try to learn a latent low-dimensional representation of documents. Latent Semantic Analysis (Deerwester et al., 1990) and Latent Dirichlet Allocation (Blei et al., 2003) are the main employed methods for this task. However, these methods do not systematically yield improved performance compared to the BOW representation.

Recently, there has been a growing interest in methods for learning distributed representations of words (Bengio et al., 2003; Collobert et al., 2011; Mikolov et al., 2013; Mnih and Kavukcuoglu, 2013; Pennington et al., 2014; Lebret and Collobert, 2014). In the embedding space, semantically similar words are likely to be close to each other. Moreover, simple linear operations on word vectors can produce meaningful results. For example, the closest vector to "Vietnam" + "capital" is found to be "Hanoi" (Mikolov et al., 2013).

Several recent works make use of distributed representations of phrases to tackle various NLP problems (Bahdanau et al., 2015; Lebret et al., 2015). There is therefore a clear need for methods that generate meaningful phrase or document representations based on the representations of their words. The most straightforward approach generates phrase or document representations by simply summing the vector representations of the words appearing in the phrase or document.

In this paper, we propose to model documents as multivariate Gaussian distributions. The mean of each distribution is the average of the vector representations of its words and its covariance matrix measures the variation of the dimensions from the mean with respect to each other. Empirical evaluation proves the superiority of the proposed representation over the standard BOW representation and other baseline approaches in a host of

different datasets.

The rest of this paper is organized as follows. Section 2 provides an overview of the related work. Section 3 provides a description of the proposed approach. Section 4 evaluates the proposed representation. Finally, Section 5 concludes.

2 Related Work

Mitchell and Lapata (2008) proposed a general framework for generating representations of phrases or sentences. They computed vector representations of short phrases as a mixture of the original word vectors, using several different element-wise vector operations. Later, their work was extended to take into account syntactic structure and grammars (Erk and Padó, 2008; Baroni and Zamparelli, 2010; Coecke et al., 2010). Lebret and Collobert (2015) proposed to learn representations for documents by averaging their word representations. Their model learns word representations suitable for summation. Le and Mikolov (2014) presented an algorithm to learn vector representations for paragraphs by inserting an additional memory vector in the input layer. Song and Roth (2015) presented three mechanisms for generating dense representations of short documents by combining Wikipedia-based explicit semantic analysis representations with distributed word representations.

Neural networks with convolutional and pooling layers have also been widely used for generating representations of phrases or documents. These networks allow the model to learn which sequences of words are good indicators of each topic, and then, combine them to produce vector representations for documents. These architectures have been proved effective in many NLP tasks, such as document classifcation (Johnson and Zhang, 2015), short-text categorization (Wang et al., 2015), sentiment classification (Kalchbrenner et al., 2014; Kim, 2014) and paraphrase detection (Yin and Schütze, 2015).

3 Gaussian Document Representation from Word Embeddings

Let $\mathcal{D} = \{d_1, d_2, \ldots, d_m\}$ be a set of m documents. The documents are pre-processed (tokenization, punctuation and special character removal) and the vocabulary of the corpus $\mathcal{V}$ is extracted. To obtain a distributed representation for each word $w \in \mathcal{V}$, we employed the *word2vec* model (Mikolov et al., 2013). Specifically, for our experiments, we used a publicly available model[1] $\mathcal{M}$ consisting of 300-dimensional vectors trained on a Google News dataset of about 100 billion words. Words contained in the vocabulary $w \in \mathcal{V}$, but not contained in the model $w \notin \mathcal{M}$ were initialized to random vectors.

To generate a representation for each document, we assume that its words were generated by a multivariate Gaussian distribution. Specifically, we regard the embeddings of all words w present in a document as i.i.d. samples drawn from a multivariate Gaussian distribution:

$$\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{1}$$

where $\mathbf{w}$ is the distributed representation of a word w, $\boldsymbol{\mu}$ is the mean vector of the distribution and $\boldsymbol{\Sigma}$ its covariance matrix.

We set $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ to their Maximum Likelihood estimates, given by the sample mean and the empirical covariance matrix respectively. More specifically, the sample mean of a document corresponds to the centroid of its words, i. e. we add the vectors of the words present in the text and normalize the sum by the total number of words. For an input sequence of words d, its mean vector $\boldsymbol{\mu}$ is given by:

$$\boldsymbol{\mu} = \frac{1}{|d|} \sum_{w \in d} \mathbf{w} \tag{2}$$

where $|d|$ is the cardinality of d, i. e. its number of words. The empirical covariance matrix is then defined as:

$$\boldsymbol{\Sigma} = \frac{1}{|d|} \sum_{w \in d} (\mathbf{w} - \boldsymbol{\mu})(\mathbf{w} - \boldsymbol{\mu})^T \tag{3}$$

Hence, each document is represented as a multivariate Gaussian distribution and the problem transforms from classifying textual documents to classifying distributions.

To measure the similarity between pairs of documents, we compare their Gaussian representations. There are several well-known definitions of similarity or distance between distributions. Some examples include the Kullback-Leibler divergence, the Fisher kernel, the χ^2 distance and the Bhattacharyya kernel. However, most of these measures are very time consuming. In our setting where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are very high-dimensional (if n

[1] https://code.google.com/archive/p/word2vec/

is the dimensionality of the distributed representations, then $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$), the complexity of these measures is prohibitive, even for small document collections.

We proceed by defining a more efficient function for measuring the similarity between two distributions. More specifically, the similarity between two documents d_1 and d_2 is set equal to the convex combination of the similarities of their mean vectors $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and their covariance matrices $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$. The similarity between the mean vectors $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ is calculated using cosine similarity:

$$sim(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) = \frac{\boldsymbol{\mu}_1 \cdot \boldsymbol{\mu}_2}{\|\boldsymbol{\mu}_1\| \|\boldsymbol{\mu}_2\|} \tag{4}$$

where $\| \cdot \|$ is the Euclidean norm for vectors. The similarity between the covariance matrices $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ can be computed using the following formula:

$$sim(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2) = \frac{\sum \boldsymbol{\Sigma}_1 \circ \boldsymbol{\Sigma}_2}{\|\boldsymbol{\Sigma}_1\|_F \times \|\boldsymbol{\Sigma}_2\|_F} \tag{5}$$

where $(\cdot \circ \cdot)$ is the Hadamard or element-wise product between matrices (we sum over all its elements) and $\| \cdot \|_F$ is the Frobenius norm for matrices. Hence, the similarity between two documents is equal to:

$$\begin{aligned} sim(d_1, d_2) = \alpha\big(sim(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)\big) \\ + (1 - \alpha)\big(sim(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)\big) \end{aligned} \tag{6}$$

where $\alpha \in [0, 1]$. It is trivial to show that the above similarity measure is also a valid kernel function.

4 Experiments

We evaluate the proposed approach as well as the baselines in the context of text categorization on eight standard datasets.

4.1 Baselines

We next present the baselines against which we compared our approach:

1) BOW (binary) Documents are represented as bag-of-words vectors. If a word is present in the document its entry in the vector is 1, otherwise 0. To perform text categorization, we employed a linear SVM classifier.

2) NBSVM It combines a Naive Bayes classifier with an SVM and achieves remarkable results on several tasks (Wang and Manning, 2012). We used a combination of both unigrams and bigrams as features.

Dataset	# training examples	# test examples	# classes	vocabulary size	*word2vec* size
Reuters	5,485	2,189	8	23,585	15,587
Amazon	8,000	CV	4	39,133	30,526
TREC	5,452	500	6	9,513	9,048
Snippets	10,060	2,280	8	29,276	17,067
BBCSport	348	389	5	14,340	13,390
Polarity	10,662	CV	2	18,777	16,416
Subjectivity	10,000	CV	2	21,335	17,896
Twitter	3,115	CV	3	6,266	4,460

Table 1: Summary of the 8 datasets that were used in our document classification experiments.

3) Centroid Documents are projected in the word embedding space as the centroids of their words. This representation corresponds to the mean vector $\boldsymbol{\mu}$ of the Gaussian representation presented in Section 3. Similarity between documnets is computed using cosine similarity (Equation 4).

4) WMD Distances between documents are computed using the Word Mover's Distance (Kusner et al., 2015). To compute the distances, we used pre-trained vectors from *word2vec*. A k-nn algorithm is then employed to classify the documents based on the distances between them. As in (Kusner et al., 2015), we used values of k ranging from 1 to 19.

5) CNN A convolutional neural network architecture that has recently showed state-of-the-art results on sentence classification (Kim, 2014). We used a model with pre-trained vectors from *word2vec* where all word vectors are kept static during training. As regards the hyperparameters, we used the same settings as in (Kim, 2014): rectified linear units, filter windows of 3, 4, 5 with 100 feature maps each, dropout rate of 0.5, l_2 constraint of 3, mini-batch size of 50, and 25 epochs.

4.2 Datasets

In our experiments, we used several standard datasets: (1) `Reuters`: contains stories collected from the Reuters news agency. (2) `Amazon`: product reviews acquired from Amazon over four different sub-collections (Blitzer et al., 2007). (3) `TREC`: a set of questions classified into 6 different types (Li and Roth, 2002). (4) `Snippets`: consists of snippets that were collected from the results of Web search transactions (Phan et al., 2008). (5) `BBCSport`: consists of sports news articles from the BBC Sport website (Greene and Cunningham, 2006). (6) `Polarity`: consists of positive and negative snippets acquired from Rotten Tomatoes (Pang and Lee, 2005). (7)

Method \ Dataset	Reuters		Amazon		TREC		Snippets	
	Accuracy	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
BOW (binary)	0.9571	0.8860	0.9126	0.9127	0.9660	0.9692	0.6171	0.5953
Centroid	0.9676	0.9171	0.9311	0.9312	0.9540	0.9586	0.8123	0.8170
WMD	0.9502	0.8204	0.9200	0.9201	0.9240	0.9336	0.7417	0.7388
NBSVM	**0.9712**	0.9155	0.9486	0.9486	0.9780	0.9805	0.6474	0.6357
CNN	0.9707	0.9297	0.9448	0.9449	0.9800	0.9800	**0.8478**	**0.8466**
Gaussian	**0.9712**	**0.9388**	**0.9498**	**0.9497**	**0.9820**	**0.9841**	0.8224	0.8244

Method \ Dataset	BBCSport		Polarity		Subjectivity		Twitter	
	Accuracy	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
BOW (binary)	0.9640	0.9690	0.7615	0.7614	0.9004	0.9004	0.7467	0.6205
Centroid	0.9923	0.9915	0.7783	0.7782	0.9100	0.9100	0.7361	0.5727
WMD	0.9871	0.9866	0.6642	0.6639	0.8604	0.8603	0.7031	0.4436
NBSVM	0.9871	0.9892	**0.8698**	**0.8698**	**0.9369**	**0.9368**	**0.7852**	0.6191
CNN	0.9486	0.9461	0.8037	0.8031	0.9315	0.9314	0.7549	0.6137
Gaussian	**0.9974**	**0.9974**	0.8021	0.8020	0.9310	0.9310	0.7534	**0.6443**

Table 2: Performance (accuracy and macro-average F1-score) in text categorization on the 8 datasets.

`Subjectivity`: contains subjective sentences gathered from Rotten Tomatoes and objective sentences gathered from the Internet Movie Database (Pang and Lee, 2004). (8) `Twitter`: contains a set of tweets, each labeled with its sentiment (Sanders, 2011). Table 1 shows statistics of the 8 datasets.

4.3 Text Categorization

To perform text categorization, we employed an SVM classifier (Boser et al., 1992). Since the proposed similarity function (Equation 6) is a kernel, we directly built the kernel matrices[2]. We tuned parameter α of the proposed approach using cross-validation on the training set of TREC and used the same value on all datasets ($\alpha = 0.5$).

To assess the effectiveness of the different approaches, we employed two well-known evaluation metrics: accuracy and macro-average F1-score. Table 2 shows the performance of the considered approaches on the eight text categorization datasets. On all datasets except three (Snippets, Polarity, Subjectivity), the proposed approach outperforms the other methods. Furthermore, on two of the remaining three datasets (Snippets, Subjectivity), it achieves performance comparable to the best-performing methods. WMD is the worst-performing method on most datasets. This may be due to the k-nn algorithm that is employed to classify the documents. NBSVM achieves impressive results on all datasets, considering that it does not utilize word embeddings. It is also important to note that the approaches that use word embeddings (Centroid, WMD, CNN, Gaussian) achieve an immense increase in performance on the Snippets dataset. One possible explanation is that these snippets belong to domains that are highly related to these of the articles on which the *word2vec* model was trained. Overall, our results demonstrate the effectiveness of the proposed method and the benefit of using word embeddings for measuring the similarity between pairs of documents.

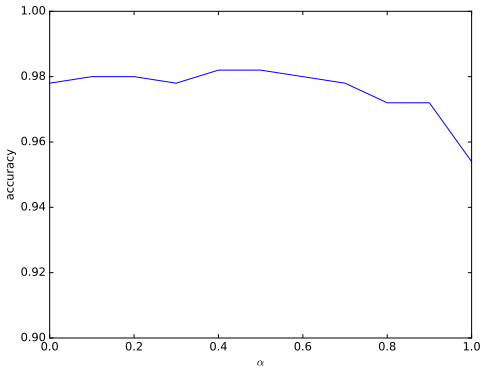

Figure 1: Classification accuracy of the proposed method with respect to parameter α on the TREC dataset.

As regards the proposed method, we also computed the sensitivity of the classification to the value of parameter α. Specifically, Figure 1 shows how the classification accuracy changes with respect to parameter α on the TREC dataset. As you can see, the highest accuracy is achieved for val-

[2]Our code is available at: `http://www.db-net.aueb.gr/nikolentzos/code/gaussian.zip`

ues of α close to 0.5. Furthermore, when dropping the second term of Equation 6 ($\alpha = 1$), the method is equivalent to the Centroid baseline and the performance drops significantly.

5 Conclusion

We proposed an approach that models each document as a Gaussian distribution based on the embeddings of its words. We then defined a function that measures the similarity between two documents based on the similarity of their distributions. Empirical evaluation demonstrated the effectiveness of the approach across a range of datasets. We attribute this performance gain of the proposed approach to the high quality of the embeddings and its ability to effectively utilize these embeddings.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empiricial Methods in Natural Language Processing*, pages 1183–1193.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022.

John Blitzer, Mark Dredze, Fernando Pereira, et al. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 440–447.

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. 1992. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 144–152.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical Foundations for a Compositional Distributional Model of Meaning. *Linguistic Analysis*, 36:345–384.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (almost) from Scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Katrin Erk and Sebastian Padó. 2008. A Structured Vector Space Model for Word Meaning in Context. In *Proceedings of the 2008 Conference on Empiricial Methods in Natural Language Processing*, pages 897–906.

Derek Greene and Pádraig Cunningham. 2006. Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 377–384.

Rie Johnson and Tong Zhang. 2015. Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 655–665.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, page 1746–1751.

Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Weinberger Kilian Q. 2015. From Word Embeddings To Document Distances. In *Proceedings of the 32th International Conference on Machine Learning*, pages 957–966.

Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196.

Rémi Lebret and Ronan Collobert. 2014. Word Embeddings through Hellinger PCA. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490.

Rémi Lebret and Ronan Collobert. 2015. "The Sum of Its Parts": Joint Learning of Word and Phrase Representations with Autoencoders. *arXiv preprint arXiv:1506.05703*.

Remi Lebret, Pedro Pinheiro, and Ronan Collobert. 2015. Phrase-based Image Captioning. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 2085–2094.

Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based Models of Semantic Composition. In *Proceedings of the 46th Annual Meeting on Association for Computational Linguistics*, pages 236–244.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*, pages 2265–2273.

Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis using Subjectivity Summarization based on Minimum Cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empiricial Methods in Natural Language Processing*, pages 1532–1543.

Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. 2008. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100.

Niek J Sanders. 2011. Twitter Sentiment Corpus. *Sanders Analytics*.

Yangqiu Song and Dan Roth. 2015. Unsupervised Sparse Vector Densification for Short Text Similarity. In *Proceeding of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics and Human Language Technologies*, pages 1275–1280.

Sida Wang and Christopher D Manning. 2012. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 90–94.

Peng Wang, Jiaming Xu, Bo Xu, Cheng-Lin Liu, Heng Zhang, Fangyuan Wang, and Hongwei Hao. 2015. Semantic Clustering and Convolutional Neural Network for Short Text Categorization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 352–357.

Wenpeng Yin and Hinrich Schütze. 2015. Convolutional Neural Network for Paraphrase Identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911.