# The Benefit of Document Embedding in Unsupervised Document Classification

Jaromír Novotný[(✉)] and Pavel Ircing

Faculty of Applied Sciences, Department of Cybernetics, The University of West Bohemia, Plzeň, Czech Republic
{fallout7,ircing}@kky.zcu.cz
http://www.kky.zcu.cz/en

**Abstract.** The aim of this article is to show that the document embedding using the doc2vec algorithm can substantially improve the performance of the standard method for unsupervised document classification – the K-means clustering. We have performed rather extensive set of experiments on one English and two Czech datasets and the results suggest that representing the documents using vectors generated by the doc2vec algorithm brings a consistent improvement across languages and datasets. The English dataset – 20NewsGroups – was processed in a way that allows direct comparison with the results of both supervised and unsupervised algorithms published previously. Such comparison is provided in the paper, together with the results of supervised classification achieved by the state-of-the-art SVM classifier.

**Keywords:** Document embedding · Doc2vec · Classification
K-means · SVM

## 1 Introduction

It is generally accepted that even such a simple unsupervised algorithm as the classic K-means achieves surprisingly good classification results, if it is presented with appropriate feature vectors. Our previous research [8] confirmed that the well-established tf-idf vectors work rather well. The aim of the work presented in this paper was to test whether the recently introduced document embeddings produced by the doc2vec method [2,4,15] can further improve the performance.

## 2 Datasets

As our basic dataset, we have again picked the *20NewsGroups* English corpus[1] which is widely used as a benchmark for document classification [1,3,5,7,8,11,12].

[1] This data set can be found at http://qwone.com/~jason/20Newsgroups/ and it was originally collected by Ken Lang.

It contains 20 000 text documents which are evenly divided into 20 categories that each contain discussion about a specific topic.

The second data set *CNO* and all sub-sets of this data set are in the Czech language. It contains approximately 68 000 articles divided into 31 categories[2]. This corpus was created so that it is at least in size and partially also in topics comparable to the English data set.

Third group of data sets – *TC* and *Large TC* – consists of the transcription of phone calls from the Language Consulting Center (LCC) of the Czech Language Institute of the Academy of Sciences of the Czech Republic, which provides a unique language consultancy service in the matters of the Czech language. The counselors of the LCC are answering questions regarding the Czech language problems on a telephone line open to public calls. The data, gathered from these language queries are unique in several aspects. The Language Consulting Center deals with completely new language material so it is the only source of advice for new language problems. It also records peripheral matters that will never be explained in dictionaries and grammar books as these are focused on the core of the language system.

In order to compare our results with the ones published previously, we have re-created two subdivision of the *20NewsGroup* corpus. The first one is created according to [12] and also used in our previous work [8] where it is described in more details.

The other subdivision is created in order to compare the results with experiments described in [1,3]. *20NG1* data sub-set consists of the 5 new categories (according to [1]) created by joining original ones as follows: `Motorcycle` – Motorcycle and Autos; `Hardware` – Windows and MAC; `Sports` – Baseball and Hockey; `Graphics` – Computer graphics; `Religion` – Christianity, Atheism and misc. Furthermore, they divided this sub-set to three training and testing data sets, where they used *[50, 200, 350]* documents as test data and the rest as training data.

*20NG2* input is whole unchanged *20NewsGroup* corpus divided into training (13 000 documents) and testing (aproximatelly 7 000 documents) data (the same divisions as in [3]).

The results achieved on the *CNO* and *TC* sets and sub-sets cannot be directly compared with the results of other research teams as the data are not (yet) made publicly available. However, these data are important for our own research and we decided to publish the results here to show some important properties of the doc2vec embedding (see the discussion below).

From the first Czech data – *CNO* – we have created the following subsets:

– Set *CNO* consists of all 31 original categories. This results in approximately 68 000 documents in total.

[2] It was created from a database of news articles downloaded from the http://www.ceskenoviny.cz/ at the University of West Bohemia and constitutes only a small fraction of the entire database – the description of the full database can be found in [14].

- Set *RCNO1* consists of 11 original categories which contain at least 1000 documents.
- Set *RCNO2* consists of 10 original categories containing between 500 and 1500 documents.
- *RCNO3* set is created from 12 categories, each containing randomly chosen 1000 documents from the original categories. This set is created for the purpose to be similar to *20NewsGroup* corpus.

The data *TC* and *Large TC* sets were created from a corpus obtained by LCC. These data sets consist of manually transcribed 607 parts of historical mono phone calls (each call can contain more than one parts, each part with different questions about different topic) and automatically transcribed (by ASR system[3]) 3128 parts of actual stereo phone call, all divided into 20 categories by their topic. These 20 categories were manually assigned by counselors from LCC (for example "semantics" or "lexicology") and corresponds with the higher level of the linguistic topic tree. The division of phone calls into categories is not uniform, some categories contain only a few parts. The setting is based on previous findings. *TC* consists of mentioned 20 categories containing 3713 transcripted text parts of the phone calls. Some of the categories are formed from a small number of texts (for example only 10), we responded to that by creating *Large TC* data consisting of 10 original categories (3343 transcripted text parts) where each contains at least 100 text parts.

## 3 Preprocessing

First processing step is only in case of the *20NewsGroups* data, where we removed all the headers except for the **Subject**. Then all uppercase characters were lowercased and all digits were replaced by one universal symbol.

As the next processing step, we wanted to conflate different morphological forms of the given word into one representation. We opted for lemmatization. The MorphoDiTa [13] tool was picked for the task – it works for both English and Czech and is available as a Python package.[4]

Traditional stop word removal is further preprocessing operation done in this paper by picking only the top $T$ lemmas with highest mutual information (MI).

After applying all these processing steps we can create following vector representations:

### 3.1 Representation by TF-IDF Weights

Common representation in text processing task named *TF-IDF* weights – i.e. combination of Term Frequency (*TF*) and Inverse Document Frequency (*IDF*)

[3] Created by colleagues at University of West Bohemia.
[4] `ufal.morphodita` at https://pypi.python.org/pypi/ufal.morphodita.

weights. The well-known formula to compute *TF-IDF* weights $w_{l,d}$ for the lemmas $l \in L$ and documents $d \in D$:

$$w_{l,d} = tf_{l,d} * idf_l \tag{1}$$

where $tf_{l,d}$ denotes the number of times the lemma $l$ occurs in document $d$ and $idf_l$ is computed using formula:

$$idf_l = \frac{N}{N(l)} \tag{2}$$

where $N$ is a total number of documents and $N(l)$ denotes a number of documents containing the lemma $l$.

In essentially all further experiments we use implemented Python package `sklearn` [9][5] for computing *TF-IDF* weights.

### 3.2 Representation by Doc2vec Weights

According to [4] doc2vec representation is simple extension of word2vec. This is done by embedding word sequences into vectors. Input can be n-grams, sentences, paragraphs or whole documents. This type of representation is considered as state-of-the-art for sentiment analysis, which is essentially also a classification task. There was therefore a good chance that it will help in our task as well.

In this paper we use the doc2vec implementation in Gensim package [10] for Python. Input data are in form of pairs consist of feature vector representation gain from 3 and label of the given document. The output is then vectors of doc2vec weights, where every row corresponds to a specific document.

### 3.3 Use of LSA Reduction on Representations 3.1 and 3.2

We have also tried to further reduce the dimension of the vector representations described in 3.1 and 3.2 by the Latent Semantic Analysis (LSA) and consequently analyze the effect on the classification accuracy. The LSA method is implemented in the Python package `sklearn` – the module `TruncatedSVD`.

## 4 Classification Methods

For our purposes, we picked one simple supervised and one simple unsupervised method. Our goal is to use unsupervised classification and at least get similar results to supervised ones.

[5] More precisely the `TfidfVectorizer` module from that package.

### 4.1 K-Means

Simple unsupervised classification algorithm – the classic K-means clustering method [6] – is being used here. It is generally accepted that even such a simple method is quite powerful for unsupervised data clustering if it is given an appropriate feature vector. As we have shown in [8], even simple feature vectors consisting of the tf-idf weights appear to capture the content of the document rather well (and the reduced feature vectors obtained from LSA do it even better). However, we expected to obtain even better results from doc2vec weights as they have been shown to be very good for extraction of the semantic information from the documents.

The `sklearn` package implementation is being used as the version of the K-means algorithm. All preprocessed representation created according to 3 are used and this model is applied to all the data sets described in Sect. 2. Results can be found in Sect. 6.

### 4.2 SVM

The supervised classification method being used here is the classic Linear SVM algorithm. This simple but powerful supervised data classification algorithm could be quite sufficient. This algorithm was run only with *TF-IDF* weights representation.

We have used the version of Linear SVM algorithm implemented in our favourite `sklearn` package (to be exact the module `Linear SVM`). Results can be found in Sect. 6.

## 5 Evaluation

Quite a few measures for evaluation of the classification algorithms are widely-used in published papers. In our experiments, we have decided to use accuracy, precision, recall and F1; this choice was guided mostly by the fact that we wanted to compare the performance of our algorithms to the previously published results.

The Accuracy (*Acc*) measure is picked only because of *20NG2* data set. It represents the percentage of correctly classified documents. This percentage is simply a number of the test documents, which are assigned with the correct topic.

The Tables 1 and 3 lists the results with the use of *Precision* and *Recall* measures computed according to [12]. Following equations for computing micro-average type of *Precision* and *Recall* measures are explained in our previous work [8] or in article [12].

$$P(T) = \frac{\sum_c \alpha(c,T)}{\sum_c \alpha(c,T) + \beta(c,T)} \tag{3}$$

$$R(T) = \frac{\sum_c \alpha(c,T)}{\sum_c \alpha(c,T) + \gamma(c,T)} \tag{4}$$

Standard equation for computing F1 measure is [1]:

$$F1 = 2 * \frac{P * R}{P + R} \tag{5}$$

The results reported in Tables 1 and 3 lists only the *Precision* measure, this is caused by usage of uni-labeled data sets (number of original categories in corpus have to be also the same as the number of output clusters from algorithms), the $P(T)$ is necessarily equal to $R(T)$ and to $F1$ and it is sufficient to report only one of those values.

## 6 Results

First sets of results are listed in Table 1; these results were achieved on *20NG*, *10NG*, *Binary[0/1/2]*, *5Multi[0/1/2]*, *10Multi[0/1/2]* data sets. We are reporting only *10Multi Average*, *5Multi Average*, *2Multi Average* result of the smaller data sub-sets and compare it with the values reported in the previously published paper [12]. It were used only results of unsupervised Sequential Information Bottleneck (*sIB*) method created by the autors of the mentioned paper. In our experiments, Linear SVM uses 10-fold cross validation technique and we run K-means algorithms 10 times over each subset (same approach used in [12]). Averaged results from those runs are listed in Table 1. The meaning of the K-means experiment labels is listed in the following table:

- *TF-IDF* uses tf-idf weights as input, every vector has size 5000.
- *TF-IDF (LSA)* uses tf-idf weights reduced by LSA method, every vector has size 200.
- *doc2vec* uses doc2vec weights as input, every vector has size 5000.
- *doc2vec (LSA)* uses doc2vec weights reduced by LSA method, every vector has size 200.
- *TF-IDF + doc2vec* is combination of *TF-IDF (LSA)* with *doc2vec (LSA)* weights, every vector has size 400.

In Table 2 are listed second sets of results. We again compare our results with values reported in the previously published papers [1,3]. The authors of the [1] paper used SVM based 1 (*SVM b. 1*) and SVM based 2 (*SVM b. 2*) methods. Both of these methods are classic SVM algorithms, in case of *SVM b. 1* method uses as input generated training data by use of WordNet, documents of input corpus and preprocessing as: stop-word removal, tokenization, TF-IDF representation, clusters created by Latent Semantic Indexing (LSI), etc. The *SVM b. 2* method is same in preprocessing but uses the corpus of input documents. The results of both their methods and our used algorithms are macro *F1-measures* from three data sub-sets divided into training and testing data according to Sect. 2.

The method listed as *HM* stated in [3] is semi-supervised classification and uses the hybrid model of deep belief network and soft regression. The unlabeled data are used to train deep belief network model and labelled data are used to

**Table 1.** Comparison of our results with results achieved in [12].

| 20NewsGroups sub-sets | *Precision* of methods [%] | | | | | | |
|---|---|---|---|---|---|---|---|
| | *sIB* | *Linear SVM (TF-IDF)* | K-means method with input representations | | | | |
| | | | *TF-IDF* | *TF-IDF (LSA)* | *doc2vec* | *doc2vec (LSA)* | *TF-IDF + doc2vec* |
| *20NG* | 57.50 | 96.38 | 51.75 | 51.68 | 70.91 | 70.76 | 73.14 |
| *10NG* | 79.50 | 95.61 | 41.43 | 42.42 | 62.80 | 67.81 | 62.67 |
| Average "large" | 68.50 | 95.99 | 46.59 | 47.05 | 66.86 | 69.29 | 67.91 |
| *10Multi Average* | 67.00 | 91.63 | 40.26 | 40.79 | 47.15 | 49.90 | 52.18 |
| *5Multi Average* | 91.67 | 96.85 | 63.65 | 63.25 | 72.45 | 77.76 | 80.95 |
| *2Multi Average* | 91.20 | 99.25 | 93.49 | 93.57 | 96.81 | 96.91 | 96.08 |
| Average "small" | 83.30 | 95.91 | 65.80 | 65.87 | 72.13 | 74.86 | 76.40 |

train softmax regression model and fine-tune the coherent whole system. The results stated as *HM* are only one of the few results in [3], they use different division of the data set to training and testing data, for these results they used 7 500 as the test set, 11 000 as unlabeled training set and 3000 as the labelled training set. For gaining our results we used similar division used in [3]. We gained training (we concatenated their unlabeled and labelled data – approximately 13 000 labelled documents for Linear SVM and without labels for K-means) and test data (approximately 7000 documents).

**Table 2.** Comparison of our results with results achieved in [1,3].

| 20News Group Sets | Methods | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SVM b. 1[a] | SVM b. 2[b] | HM | Our approach | | | | | |
| | | | | *Lin. SVM (TF-IDF)* | K-means method with input representations | | | | |
| | | | | | *TF-IDF* | *TF-IDF (LSA)* | *doc2vec* | *doc2vec (LSA)* | *TF-IDF + doc2vec* |
| *20NG1*[c] F1 [%] | 73.00 | 64.00 | – | 80.00 | 54.00 | 54.00 | 69.01 | 48.00 | 52.00 |
| *20NG2*[d] Acc [%] | – | – | 82.63 | 95.21 | 52.74 | 25.72 | 66.06 | 27.15 | 29.47 |

[a] Training done by using 20News Group and Web Features
[b] Training done by using only 20News Group
[c] Data set prepared according to [1] and describe in Sect. 2
[d] Data set prepared according to [3] and describe in Sect. 2

Results on Czech data sets are listed in Table 3. We state these only for the purpose of testing our approach on the data in the different language than English. The results on the language rather distant from English shows that our approach of the preparation of the data can be also applied in this case.

**Table 3.** Results on *Czech* data sets.

| Czech data sets | *Precision* of methods [%] | | | | | |
|---|---|---|---|---|---|---|
| | *Linear SVM (TF-IDF)* | K-means method with input representations | | | | |
| | | *TF-IDF* | *TF-IDF (LSA)* | *doc2vec* | *doc2vec (LSA)* | *TF-IDF + doc2vec* |
| *CNO* | 76.79 | 28.79 | 28.91 | 30.87 | 29.97 | 29.45 |
| *RCNO1* | 93.94 | 46.13 | 47.06 | 53.71 | 52.79 | 54.60 |
| *RCNO2* | 96.30 | 42.20 | 42.85 | 49.24 | 49.46 | 53.04 |
| *RCNO3* | 93.54 | 51.11 | 51.86 | 61.00 | 61.00 | 61.29 |
| *TC* | 77.92 | 31.29 | 32.12 | 31.51 | 28.65 | 32.53 |
| *Large TC* | 78.89 | 40.34 | 38.79 | 38.68 | 38.54 | 42.08 |

## 7 Conclusion

A reasonably effective pipeline for unsupervised text documents classification according to their topic is introduced in this paper. Preprocessing of the raw input text[6] and extracted feature vectors[7] are key factors in our approach. Simple supervised Linear SVM and unsupervised classification K-means algorithms were used and as was predicted, the supervised one is superior to the unsupervised. Our main goal is to at least have similar results with unsupervised algorithm to supervised one. The performance of this unsupervised method (stated in Table 2) was almost on par with semi-supervised algorithm and even better against supervised algorithms used in [1]. Also as you can see from all Tables 1, 2 and 3 representation with use of doc2vec model increases performance of our unsupervised method around 10%. This is an important finding of our research, since the benchmark training data – which are necessary for supervised learning – are often not available. Also our approach of preprocessing input data texts is suitable even for simple supervised Linear SVM algorithm whose performance is comparable with more complex one (Table 2).

**Acknowledgments.** This research was supported by the Ministry of Education, Youth and Sports of the Czech Republic project No. LO1506.

## References

1. Chinniyan, K., Gangadharan, S., Sabanaikam, K.: Semantic similarity based web document classification using support vector machine. Int. Arab J. Inf. Technol. (IAJIT) **14**(3), 285–292 (2017)
2. Hamdi, A., Voerman, J., Coustaty, M., Joseph, A., d'Andecy, V.P., Ogier, J.M.: Machine learning vs deterministic rule-based system for document stream segmentation. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 5, pp. 77–82. IEEE (2017)

[6] Applying lemmatization and data-driven stop-word removal.
[7] Use of LSA method.

3. Jiang, M., et al.: Text classification based on deep belief network and softmax regression. Neural Comput. Appl. **29**(1), 61–70 (2018)
4. Lau, J.H., Baldwin, T.: An empirical evaluation of doc2vec with practical insights into document embedding generation. arXiv preprint arXiv:1607.05368 (2016)
5. Liu, Y., Liu, Z., Chua, T.S., Sun, M.: Topical word embeddings. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2418–2424 (2015)
6. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
7. Nguyen, D.Q., Billingsley, R., Du, L., Johnson, M.: Improving topic models with latent feature word representations. Trans. Assoc. Comput. Linguist. **3**, 299–313 (2015)
8. Novotný, J., Ircing, P.: Unsupervised document classification and topic detection. In: Karpov, A., Potapova, R., Mporas, I. (eds.) SPECOM 2017. LNCS (LNAI), vol. 10458, pp. 748–756. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66429-3_75
9. Pedregosa, F.: Scikit-learn: machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011). http://scikit-learn.org
10. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50 (2010). https://radimrehurek.com/gensim/
11. Siolas, G., d'Alche Buc, F.: Support vector machines based on a semantic kernel for text categorization. In: IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN), vol. 5, pp. 205–209 (2000)
12. Slonim, N., Friedman, N., Tishby, N.: Unsupervised document classification using sequential information maximization. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 129–136 (2002)
13. Straková, J., Straka, M., Hajič, J.: Open-source tools for morphology, lemmatization, POS tagging and named entity recognition. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 13–18 (2014)
14. Švec, J., et al.: General framework for mining, processing and storing large amounts of electronic texts for language modeling purposes. Lang. Resour. Eval. **48**(2), 227–248 (2014). https://doi.org/10.1007/s10579-013-9246-z
15. Trieu, L.Q., Tran, H.Q., Tran, M.T.: News classification from social media using twitter-based doc2vec model and automatic query expansion. In: Proceedings of the Eighth International Symposium on Information and Communication Technology, pp. 460–467. ACM (2017)