Lithuanian news clustering using document embeddings

Lukas Stankevičius
Faculty of Informatics
Kaunas University of Technology
Kaunas, Lithuania
lukas.stankevicius@ktu.edu

Mantas Lukoševičius
Faculty of Informatics
Kaunas University of Technology
Kaunas, Lithuania
mantas.lukosevicius@ktu.lt

***Abstract*—A lot of research of natural language processing is done and applied on English texts but relatively little is tried on less popular languages. In this article document embeddings are compared with traditional bag of words methods for Lithuanian news clustering. The results show that for enough documents the embeddings greatly outperform simple bag of words representations. In addition, optimal lemmatization, embeddings vector size, and number of training epochs were investigated.**

***Keywords*—document clustering; document embedding; lemmatization; Lithuanian news articles.**

I. Introduction

The knowledge and information are inseparable part of our civilization. For thousands of years from news of incoming troops to ordinary know-how could have meant death or life. Knowledge accumulation throughout the centuries led to astonishing improvements of our way of live. Hardly anyone could persist having no news or other kinds of information even throughout the day.

Despite information scarcity centuries ago, nowadays we have the opposite situation. Demand and technology greatly increased the amount of information we can acquire. Now one's goal is to not get lost in it. As an example, the most popular Lithuanian news website each day publishes approximately 80 news articles. Add other news websites not only from Lithuania but the entire world and one would end up overwhelmed to read most of this information.

The field of text data mining emerged to tackle this kind of problems. It goes "beyond information access to further help users analyze and digest information and facilitate decision making" [1]. Text data mining offers several solutions to better characterize text documents: summarization, classification and clustering [1]. However, when evaluated by people, the best summarization results currently are given only 2-4 points out of 5 [2]. Today the best classification accuracies are 50-94% [3] and clustering of about 0.4 F1 score [4]. Although achieved classification results are more accurate, the clustering is perceived more promising as it is universal and can handle unknown categories as it is the case for diverse news data.

After it was shown that artificial neural networks can be successfully trained and used to reduce dimensionality [5], many new successful data mining models had emerged. The aim of this work is to test how one of such models – document to vector (Doc2Vec) can improve clustering of Lithuanian news.

II. Related work on Lithuanian language

Articles on Lithuanian language documents clustering suggest using K-means [4], spherical K-means [6] or Expectation-Maximization (EM) [7] algorithms. It was also observed that K-means is fast and suitable for large corpora [7] and outperforms other popular algorithms [4].

[6] considers Term Frequency / Inverse Document Frequency (TF-IDF) as the best weighting scheme. [4] adds that it must be used together with stemming while [6] advocates to do minimum and maximum document frequency filtering before applying TF-IDF. These works show that TF-IDF is significant weighting scheme and it could be optionally tried with some additional preprocessing steps.

We have not found any research on Lithuanian language regarding document embeddings. However, there are some work on word embeddings. In [8] word embeddings using different models and training algorithms were compared after training on 234 million tokens corpus. It was found that Continuous Bag of Words (CBOW) architecture significantly outperformed skip-gram method while vector dimensionality showed no significant impact on the results. This implies that document embeddings like word embeddings should follow same CBOW architectural pattern. Other work [9] compared traditional and deep learning (with use of word embeddings) approaches for sentiment analysis and found that deep learning demonstrated good results only when applied on the small datasets, otherwise traditional methods were better. As embeddings may be underperforming in sentiment analysis it will be tested if it is a case for news clustering.

III. Text Clustering Process

To improve clustering quality some text preprocessing must be done. Every text analytics process consists „of three consecutive phases: Text Preprocessing, Text Representation and Knowledge Discovery“ [1] (the last being clustering in our case).

A. Text preprocessing

The purpose of text preprocessing is to make the data more concise and facilitate text representation. It mainly involves tokenizing text into features and dropping the ones considered less important. Extracted features can be words, chars or any *n*-gram (contiguous sequence of *n* items from a given sample of text) of both. Tokens can also be accompanied by the structural or placement aspects of document [10].

The most and least frequent items are considered uninformative and dropped. Tokens found on every document are not descriptive and they usually include stop words such as “*and*”, “*to*”. On the other hand, too rare words are insufficient to attribute to any characteristic and due to their resulting sparse vectors only complicate the whole process.

Existing text features can be further concentrated by these methods:

- stemming;
- lemmatization;
- number normalization;
- allowing only maximum number of features;
- maximum document frequency – ignore terms that appear in more than specified documents;
- minimum document frequency – ignore terms that appear in less than specified documents.

It was shown that the use of stemming in Lithuanian news clustering greatly increased clustering performance [4].

B. Text representation

For the computer to make any calculations with the text data it must be represented in numerical vectors. The simplest representation is called “Bag Of Words” (BOW) or “Vector Space Model” (VSM) where each document has counts or other derived weights for each vocabulary word. This structure ignores linguistic text structure. Surprisingly, in [11] it was reviewed that “unordered methods have been found on many tasks to be extremely well performing, better than several of the more advanced techniques”, because “there are only a few likely ways to order any given bag of words”.

The most popular weight for BOW is TF-IDF. Recent study [4] on Lithuanian news clustering have shown that TF-IDF weight produced the best clustering results. TF-IDF is calculated as:

$$tfidf(w,d) = tf(w,d) \cdot log\,\frac{N}{df(w)} \qquad (1)$$

where:

- *tf*(*w*,*d*) is term frequency, the number of word *w* occurrences in a document *d*;
- *df*(*w*) is document frequency, the number of documents containing word *w*;
- *N* is number of documents in the corpus.

One of the newest and widely adopted document representation schemes is Doc2Vec [12]. It is an extension of the word-to-vector (Word2Vec) representation. A word in the Word2Vec representation is regarded as a single vector of real number values. The assumption of Word2Vec is that the element values of a word are affected by those of other words surrounding the target word. This assumption is encoded as a neural network structure and the network weights are adjusted by learning observed examples [13]. Doc2Vec extends Word2Vec from the word level to the document level and each document has its own vector values in the same space as that for words [12].

C. Text clustering

There are tens of clustering algorithms to choose from [14]. One of the simplest and widely used is *k*-means algorithm. During initialization, *k*-means algorithm selects *k* means, which corresponds to *k* clusters. Then algorithm repeats two steps: (1) for every data point choose the nearest mean and assign the point to the corresponding cluster; (2) recalculate means by averaging data points assigned to the corresponding cluster. The algorithm terminates, when assignment of the data points does not change after several iterations. As the clustering depends on initially selected centroids, the algorithm is usually run several times to average over random centroid initializations.

IV. The Data

A. Articles

Article data for this research was scraped from three Lithuanian news websites: the national *lrt.lt* and commercial websites *15min.lt* and *delfi.lt*. Articles URL’s were scraped from sitemaps in *robots.txt* files in websites. Total of 82793 articles (26336 from *lrt.lt*, 31397 from *15min.lt* and 25060 from *delfi.lt*) were retrieved spanning random release dates of 2017 year.

Raw dataset contains 30338937 tokens from which 641697 are unique. Unique token count can be decreased to:

- 641254, dropping stop words;
- 635257, normalizing all numbers to a single feature;
- 441178, applying lemmas and leaving unknown words;
- 41933, applying lemmas and dropping unknown words;
- 434472, dropping stop words, normalizing numbers, applying lemmas and leaving unknown words.

Each article has on average 366 tokens and on average 247 unique tokens. Mean token length is 6.51 characters with standard deviation of 3.

While analyzing articles and their accompanying information, it was noticed that some labelling information can be acquired from article URL. Both websites have categorical information between the domain and article id parts in URL. Total of 116 distinct categorical descriptions were received and normalized to 12 distinct categories as described at [4]. Category distributions are:

- Lithuania news (20162 articles);
- World news (21052 articles);
- Crime (7502 articles);
- Business (7280 articles);
- Cars (1557 articles);
- Sports (5913 articles);
- Technologies (1919 articles);
- Opinions (2553 articles);
- Entertainment (769 articles);
- Life (944 articles);
- Culture (3478 articles);
- Other (9664 articles, which do not fall into previous categories).

It is clearly visible that category distribution is not uniform. The biggest categories are “Lithuanian news” and “World news” taking up to 49 % of all articles.

B. *Words*

Lithuanian word data was scraped from two semantic information databases: *morfologija.lt* and *tekstynas.vdu.lt/~irena/morfema_search.php*. The latter website has more accurate information, including word frequency while the first is very large and was observed having some mistakes. Therefore, these two databases were merged prioritizing words from the second one. Resulting word database contained 2212726 different word forms including 72587 lemmas.

V. Clustering Evaluation

The main evaluation metrics can be acquired by confusion matrix, depicted in Table I. Here for true and predicted conditions we get counts of following types:

- TP (true positives). The true condition is positive and the predicted condition is positive.
- TN (true negatives). The true condition is negative and the predicted condition is negative.
- FP (false positives). The true condition is negative but the predicted condition is positive.
- FN (false negatives). The true condition is positive but the predicted condition is negative.

If it would be a classification task, then we would know real classes and just simply get percentage of them predicted accurately. However, in the clustering process nor we know actual class, nor we have a meaning of returned predicted class. We must rely an additional information - label of our news article category, given by the editor of the news website. This way we make assumption that clusters we want to achieve are similar to categories of articles. There indeed must be a reason, some similarity between articles, why they were put in the same category. The only drawback of our approach is that having high number of documents would require many pair calculations. Based on chosen condition, confusion matrix elements are as following:

- TP – pairs of articles have same category label and are predicted to be in the same cluster.
- TN – pairs of articles belong to different categories and are predicted to be in different clusters.
- FP – pairs of articles belong to different categories but are predicted to be in the same cluster.
- FN – pairs of articles having same category label but are predicted to be in different clusters.

We will use F1, as the one widely used, and MCC, as more robust, evaluation scores:

$$F1 = 2\frac{precision \cdot recall}{precision + recall} \tag{2}$$

$$precision = \frac{TP}{TP+FP} \tag{3}$$

$$recall = \frac{TP}{TP+FN} \tag{4}$$

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{5}$$

MCC score ranges from -1 (total disagreement) to 1 (perfect prediction), while 0 means no better than random prediction. F1 score varies from 0 (the worst) to 1 (perfect).

VI. Experiments

To ensure that experiments are as reproducible as possible, each experiment was repeated 50 times and confidence interval of each resulting clustering scores calculated. In each repetition distinct number of articles were randomly (each time) selected from the dataset. However, for the same number of documents this repeated random pickup would be the same (if we were to have another experiment with same number of documents then these 50 samplings of articles would be the same). This ensures that we evaluate as much data as possible while keeping the same subset for different experiments.

All experiments were carried out using only articles from the 10 biggest categories. For each of them equal number of articles were sampled. Only variables associated with dataset loading, text preprocessing and representation phases were varied. Actual clustering was done using *k*-means algorithm.

In all experiments the following actions and parameters were used if not specified otherwise:

- used 1500 articles;
- vocabulary pruned to maximum of 10000 words;
- 0.95 maximum document frequency (BOW);
- 0.05 minimum document frequency (BOW);
- Distributed Bag of Words (DBOW) architecture of Doc2Vec model used;
- Doc2Vec method trained on same articles to be clustered (not all corpus);
- window size of 5 words (Doc2Vec models);
- 20 training epochs (Doc2Vec models);
- 200 vector size (Doc2Vec models);
- minimum word count of 4 (Doc2Vec models);
- all number normalized to “#NUMBER” feature;
- words with known lemma lemmatized;
- words in stop word list dropped from documents;
- unigrams used (feature as a single word).

A. *Number of articles and preprocessor method experiment*

In this experiment dataset size and preprocessor method were varied to determine how the two are correlated. Tried text representations include BOW and Doc2vec with distributed bag of words variation. It was also examined how well Doc2Vec would perform if trained on all the 82793 articles.

B. *Reducing words to lemmas experiment*

This experiment investigated 3 scenarios:

1) lemmas are not used;
2) words for which lemmas could be found were replaced with them and other words discarded;
3) same as 2 but unknown words remained.

Another parameter, namely maximum number of features, solves similar issues as lemmatization. Due to this reason several values of maximum number of allowed features were tried.

C. *Training epochs and embedding vector size experiment*

In this experiment two parameters for Doc2Vec were optimized: training epochs (from 5 to 100) and vector size (from 5 to 400). Distributed bag of words version of Doc2Vec was used.

D. Clustering articles from a defined release interval

In this experiment the best configurations for BOW and Doc2Vec will be tried on articles released in one week from 2017-04-28 to 2017-05-04 dates, covering total of 1001 articles. Both models with same articles will be run 50 times and the best run selected. Doc2Vec is trained on same articles used for clustering using maximum number of 40000 features and vector size of 52.

The best resulting clusters will be analyzed with the same BOW workflow as documents but reducing features only with 0.8 maximum and 0.1 minimum document frequencies. 10 words with the biggest TF-IDF weights will be selected as representative of each cluster.

VII. Results and Analysis

A. Number of articles and preprocessor method experiment

Experiment results are shown in Fig. 1. The best recorded MCC score is 0.403 (0.464 for F1) for Doc2Vec, distributed bag of words variation trained on all corpus and clustering 3000 articles. It is clearly visible that all text representation models are better with higher number of documents. When clustering a small number of documents we can observe that BOW model outperforms Doc2Vec if the latter is trained only on documents that are later used for clustering. However, starting with 300 documents Doc2vec outperforms BOW model. This shows that Doc2Vec model depends on how many documents it is trained on as the model trained on all corpus has the biggest MCC score of 0.201 when clustering 100 articles. However, advantage of training on all corpus instead of only documents to be clustered quickly diminishes as the number of clustering documents approaches 700.

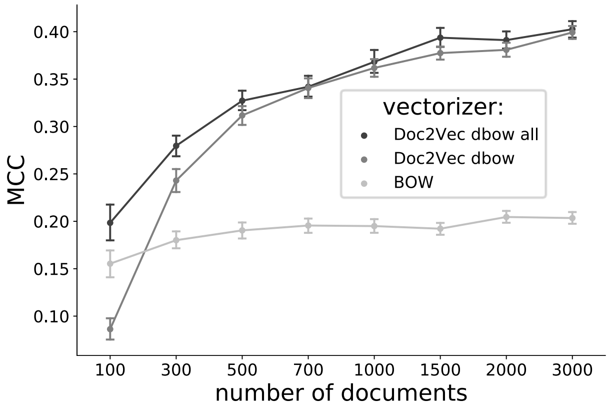

Fig. 1. MCC score dependency on text representation method and number of documents used in clustering.

B. Reducing words to lemmas experiment

Experiment results are depicted in Fig. 2. It was observed that converting known words to lemmas gives MCC score boost both for BOW and Doc2Vec models. The highest increase of MCC score (from 0.122 to 0.221 for 10000 maximum features) for BOW representation is observed then after lemmatization non-lemmatized words are dropped. On the other hand, Doc2Vec representation yields higher MCC score increase then non-lemmatized words are left (from 0.356 to 0.401 for 40000 maximum number of features). It is clearly visible that both vectorization methods benefit from lemmatization.

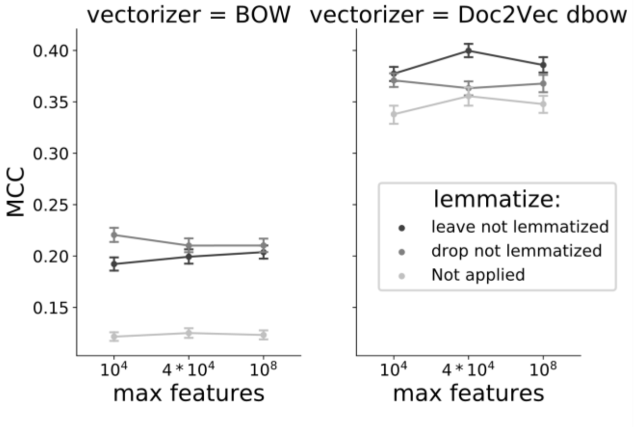

Fig. 2. MCC score dependency on how words are changed to their lemma with or without constrain of maximum features.

C. Training epochs and embedding vector size experiment

Clustering results for several epochs and vector sizes are depicted in Fig. 3. The highest average MCC score was recorder for vector size of 150 and 20 epochs at 0.381. It is interesting to note that increasing number of training epochs to 100 reduces MCC to 0.316. This reduction is observer for all vector sizes and could be explained as overfitting. On the other hand, only 5 epochs give poor results with maximum MCC of 0.133 for vector size of 10 and it should be regarded as underfitting. With optimal number of training epochs being 20, there are many vector sizes (from 20 to 400) yielding very similar MCC results. This shows that small vector sizes such as 20 are enough to train 1500 articles dataset for 20 epochs for good text representation.

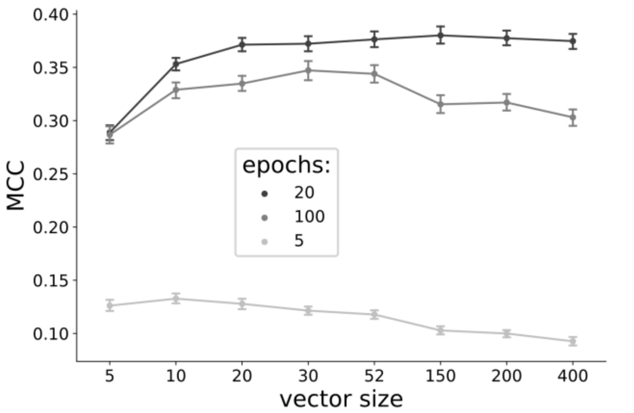

Fig. 3. MCC score dependency on vector size and number of training epochs in Doc2Vec distributed bag of words representation clustering

D. Clustering articles from defined release interval

The best Doc2Vec model trained on a small corpus outperformed the best BOW model (MCC 0.318 and 0.145, F1 0.415 and 0.282). Cluster features and statistics of Doc2vec model are depicted in Table I. It shows that model performs reasonably well and can distinguish:

- very small (1.9 % of all articles) distinct weather forecast category (cluster Nr. 5);
- classical categories as culture, sports, and crime (clusters Nr. 3, 8 and 10);
- hot topics as university reform, Brexit and current political scandals (clusters Nr. 1, 4 and 8).

TABLE I. CLUSTERS STATISTICS

		Category label										
Cluster Nr.	***Number of articles in cluster***	***Other***	***Crime***	***Culture***	***Lithuania news***	***Technologies***	***Opinions***	***World news***	***Entertainment***	***Sports***	***Business***	***Most descriptive features and their translation to English***
1.	40	11	0	0	**24**	0	3	0	0	0	2	universitetas, mokslas, eur, mokykla, studija, pertvarka, akademija, rektorius, vu, kokybė // university, science, eur, school, study, transformation, academy, rector, vu (Vilnius University), quality
2.	87	27	0	2	**35**	3	**15**	3	0	0	2	muzika, alkoholis, kultūra, ntv, filmas, visuomenė, maistas, namas, liga, lelkaitis // music, alcohol, culture, ntv, film, society, food, house, illness, lelkaitis (surname of a person)
3.	118	**29**	1	**40**	18	4	1	4	**16**	2	3	koncertas, teatras, muzika, rež, biblioteka, festivalis, džiazas, kultūra, paroda, muziejus // concert, theater, music, dir, library, festival, jazz, culture, exhibition, museum
4.	106	8	0	0	16	0	1	**80**	0	0	1	es, brexit, derybos, le, pen, may, macronas, partija, th, politinis // es, brexit, talks, le, pen, may, macron, party, th, political
5.	19	0	0	0	16	0	0	2	0	0	1	laipsnis, šiluma, temperatūra, naktis, debesis, debesuotumas, lietus, įdienojus, pūs, termometrai // degree, heat, temperature, night, cloud, clouds, rain, be broad daylight, will blow, thermometers
6.	184	1	0	0	16	5	0	**160**	0	0	2	jav, korėtis, raketa, korėja, branduolinis, putinas, jungtinis, pajėgos, karinis, sirijos // usa, korėtis, rocket, korea, nuclear, putin, united, forces, military, syrian
7.	120	11	1	0	37	4	9	10	0	0	**48**	įmonė, seimas, įstatymas, mokestis, savivaldybė, kaina, šiluma, asmuo, projektas, pajamos // company, parlament, law, tax, municipality, price, heat, person, project, income
8.	79	4	1	1	**67**	0	1	0	0	2	3	seimas, pūkas, partija, teismas, komisija, konstitucija, pirmininkas, įstatymas, apkalti, taryba // parlament, pūkas (surname of a person), party, court, commission, constitution, chairman, law, impeachment, board
9.	64	0	0	0	0	0	0	0	0	**64**	0	rungtynės, taškas, žaidėjas, čempionatas, ekipa, rinktinė, įvartis, pelnyti, pergalė, raptors // match, point, player, championship, team, team, goal, win, victory, raptors (name of basketball club)
10.	184	13	**67**	2	27	3	0	68	0	0	4	policija, automobilis, vyras, vairuotojas, pranešti, įtariamas, sulaikyti, žūti, teismas, asmuo // police, car, man, driver, report, suspected, detained, die, court, person

VIII. CONCLUSIONS

In this work BOW and Doc2Vec text representation methods were compared. Our research shows that Doc2Vec greatly outperforms BOW model. Clustering weeks' worth of data the highest MCC scores are 0.318 versus 0.145. However, for Doc2Vec method to outperform BOW when clustering less than 300 articles, it must be trained on a much larger dataset. We estimated optimal embedding vector size large enough starting with 20 and optimal number of training epochs around 20. Analysis of words conversion to their lemmas showed that lemmatization of words is beneficial for both BOW and Doc2Vec representations.

REFERENCES

[1] Aggarwal CC, Zhai C, editors. Mining text data. Springer Science & Business Media; 2012 Feb 3.

[2] Liu L, Lu Y, Yang M, Qu Q, Zhu J, Li H. Generative adversarial network for abstractive text summarization. In Thirty-Second AAAI Conference on Artificial Intelligence 2018 Apr 29.

[3] Liu G, Guo J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Neurocomputing. 2019 Feb 1.

[4] V. Pranckaitis and M. Lukoševičius, Clustering of Lithuanian news articles. Proceedings of the IVUS 2017, pp. 27-32.

[5] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. science. 2006 Jul 28;313(5786):504-7.

[6] Mackutė-varoneckienė, Aušra; Krilavičius, Tomas. Empirical study on unsupervised feature selection for document clustering. In Human Language Technologies – The Baltic Perspective 2014. p. 107-110.

[7] Ciganaitė, Greta, Aušra Mackutė-Varoneckienė, and Tomas Krilavičius. Text documents clustering. Informacinės technologijos. XIX tarpuniversitetinė magistrantų ir doktorantų konferencija" Informacinė visuomenė ir universitetinės studijos"(IVUS 2014): konferencijos pranešimų medžiaga, 2014, p. 90-93. 2014.

[8] Kapočiūtė-Dzikienė, Jurgita, and Robertas Damaševičius. Intrinsic evaluation of Lithuanian word embeddings using WordNet. Computer Science On-line Conference. Springer, Cham, 2018.

[9] Kapočiūtė-Dzikienė, Jurgita, Robertas Damaševičius, and Marcin Woźniak. Sentiment analysis of Lithuanian texts using traditional and deep learning approaches. Computers 8.1 (2019): 4.

[10] Aker A, Paramita M, Kurtic E, Funk A, Barker E, Hepple M, Gaizauskas R. Automatic label generation for news comment clusters. In Proceedings of the 9th International Natural Language Generation Conference 2016 (pp. 61-69).

[11] White L, Togneri R, Liu W, Bennamoun M. Sentence Representations and Beyond. In Neural Representations of Natural Language 2019 (pp. 93-114). Springer, Singapore.

[12] LE, Quoc; MIKOLOV, Tomas. Distributed representations of sentences and documents. In: International conference on machine learning. 2014. p. 1188-1196.

[13] MIKOLOV, Tomas, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[14] Charu C. Aggarwal , Chandan K. Reddy, Data Clustering: Algorithms and Applications, Chapman & Hall/CRC, 20