Proceedings of the Federated Conference on Computer Science and Information Systems pp. 357–360
DOI: 10.15439/2017F414
ISSN 2300-5963 ACSIS, Vol. 11

Deep Learning methods for Subject Text Classification of Articles

Piotr Semberecki
Wrocław University of Science and Technology
wybrzeże Stanisława Wyspiańskiego 27,
50-370 Wrocław, Poland
Email: piotr.semberecki@pwr.edu.pl

Henryk Maciejewski
Wrocław University of Science and Technology
wybrzeże Stanisława Wyspiańskiego 27,
50-370 Wrocław, Poland
Email: henryk.maciejewski@pwr.edu.pl

***Abstract*—This work presents a method of classification of text documents using deep neural network with LSTM (long short-term memory) units. We have tested different approaches to build feature vectors, which represent documents to be classified: we used feature vectors constructed as sequences of words included in the documents, or, alternatively, we first converted words into vector representations using word2vec tool and used sequences of these vector representations as features of documents. We evaluated feasibility of this approach for the task of subject classification of documents using a collection of Wikipedia articles representing 7 subject categories. Our experiments show that the approach based on an LSTM network with documents represented as sequences of words coded into word2vec vectors outperformed a standard, bag-of-word approach with documents represented as frequency-of-words feature vectors.**

I. Introduction

THE problem of automated classification of text documents is one of important tasks solved by text mining methods. A number of diverse applications of text classification were reported in literature, ranging from subject categorization [3], analysis of sentiment of reviews or opinions, to authorship recognition of documents [4], [8], [9], etc.

Standard methods of text classification consist in representing documents with usually high-dimensional feature vectors and then training classifiers such as SVM, Naive Bayes, k-NN, etc. [7], [5], [6]. Although several ways of representing documents with feature vectors were proposed (see e.g. [1]), a commonly used approach consists in constructing feature vectors which represent (possibly weighted) frequencies of selected words or collections of words (bigrams, n-grams, phrases) that appear in subsequent documents. These approaches can be broadly named as bag-of-words methods.

In this work, we want to investigate feasibility of a conceptually different approach to (i) representing documents with feature vectors, and (ii) training classifiers. The key difference is that rather than using feature vectors which are based on frequencies of words, we use feature vectors that rely on sequences of words in documents. As a consequence of this, we also need to change the type of classifier used: we use in this study a neural network with the long short-term memory (LSTM) [15] element, which allows to learn sequences from the training data. We use two different ways to of representing sequences of words for training the LSTM network: with simple encoding of words, and with word2vec method which represents words in vector space [10].

We provide empirical verification of this approach based on a collection of Wikipedia articles which represent 7 subject categories (arts, history, law, medicine, religion, sports and technology), with 1000 articles per category. As this corpus was also used in our previous study [2], where subject classification was based on bag-of-words approach, we have a chance to compare performance of these two methods.

Technically, the presented approach to classification was implemented using the Keras with Tensorflow library for deep neural networks with the models trained on a GPU. We also provide some technical details regarding implementation of the classifiers, as this may be of use to the interested readers. Keras is a wrapper to Tensorflow that gives ability to construct neural networks layers in just few lines of code, which defines the layers that the model consist of. Figures included in this paper present what network architectures were developed (Fig. 1-4).

The idea behind this paper, was to show how to build effective text classifier using state-of-the-art Deep Learning methods and tools and show what issues can appear during this procedure.

II. Standard (BOW) approach to classification

In traditional Bag-of-Words approach the key-words are filtered from training data. Usually, some Natural Language Processing methods can be involved such as: Segmentation, Tokenization, PoS Tagging, Entity Detection, Relation Detection [12]. Creating such objects from text can give a lot of information about its content. The appearance and frequencies of specific tokens, entities are used as a basis for Bag-of-Words model. However, the number of this kind of object can be very large. Therefore, methods to reduce dimensionality of data are needed, for instance TF-IDF, PCA, LDA, SVD, t-SNE etc. [11], [20] to take only the most important words for classification.

A. Related work

In this work, we used NLTK (Natural Language Processing Toolkit) algorithms for tokenization. However, we adopted a different approach to feature selection: rather than encoding

the frequencies of key-words, the words from sentences were directly transformed into sequences of encoding vectors and were used for training deep learning methods such as Long Short-Term Memory Network (LSTM). This approach has a common factor with probabilistic models such as n-grams, conditional random fields and other Markov-models, which also use sequences and are based on appearance probability of specific words.

LSTM neural networks are considered as state-of-the-art approaches offering very high accuracy results in several Natural Language Processing tasks such as: Bi-directional LSTM-CRF [18] for Part of Speech Tagging and Tree-LSTMs for sentiment analysis [19]. Also, simpler versions of LSTMs, referred to as Gated Recurrent Units (GRUs) [16] are used as key parts of larger systems like state-of-the-art Dynamic Memory Networks, developing more complicated tasks such as questions-answering systems [17].

B. Sample data for empirical verification of the methods

The dataset used for this work has been introduced in [2]. It is based on English Wikipedia articles. Each article has at least 400 characters. The corpus consists of 7 categories such as arts, history, law, medicine, religion, sports and technology with 1000 articles in each category. The data was split randomly into 800 training and 200 testing articles. Also, later in the tests k-fold validation was performed. The test was performed for 2, 3 and 7 categories independently.

Prior to feature selection and model development, we preprocessed the text data using standard NLP methods (with NLTK library) – the first step was to tokenize the text documents. This approach was similar to previous work [2] and is a part of traditional NLP processing chain. We used English Punkt as sentence tokenizer for segmentation task. Next, the sentences were split to words by RegexpTokenizer over white spaces and punctuation was removed from the text. After that, all letters where changed to lowercase. Finally, all stopwords where removed from data.

The corpus used in this study was very diversified. Each article had on average 520 words with standard deviation of 902. The largest article had 14591 words, however, for training purpose each of the articles was cropped to 500 or 1000 words in length.

C. Results of BOW method for this dataset in previous work

This work is an extension of the previous research [2], where subject classification was done using standard Machine Learning such as Decision Trees, Naive Bayes classifier etc., with the focus on distributed implementation, in order to manage large volumes of data. The best results in the previous work was obtained using Bag-of-Words model with TF-IDF and Naive Bayes, where recognition of three categories: History, Arts and Law was done with ca 75.28% accuracy on the testing corpus.

III. Proposed approach to representation of text docs for classification

The goal of this work was to apply the new Deep Learning approach to problem of subject classification and compare this with the results of the study solved previously. During this research, two approaches to feature selection were tested. In order to use Deep Recurrent Neural Network (i.e. deep networks with the LSTM component), all the data had to be used as vectors of sequences. The first approach to obtaining such vectors was based on a very simple vocabulary encoding. It worked well for binary classification, however its accuracy deteriorated when the number of classes (subject categories) increased. The second approach was based on more sophisticated word2vec method, which was more stable and elastic solution.

A. Method 1 – Simple encoding of word sequences

The first idea presented in this paper came from Kaggle challenge called "Sentiment Analysis on Movie Reviews" [21], in which was provided IMBD Movies reviews dataset was provided for sentiment classification, which is a quite close related problem to subject classification. This dataset is also available in Keras. Based on this approach, also the Wikipedia tokenized corpus used in this work was encoded.

The simplest idea to encode words is to enumerate them. In this approach, the words are encoded using the following mapping: word_from_dictionary: number. The dictionary is ordered from most frequent words in learning set to the least frequent ones, with the conventions: 1 - denotes the start of the sentence, 2 - means word out-of-vocabulary, 0 - is padding if the vector is shorter. This padding is done in front of the vector and the start of the vector is truncated if the sequences are longer than 1000 words. The vocabulary size was 10 000 tokens. Larger number of tokens would provide computational difficulties and this was the reason why this method has limited application. Example vector, which represents sentence can have a form [0, 0, 0, 1, 3565, 2, 3214, ...]. This method worked with 91.52% accuracy on two classes, but on three classes it decreased to 76.53% and 58.93% on seven classes (table I).

B. Method 2 – word2vec - based encoding

Word2vec is a method of representation of words in multi-dimensional vector space, recently proposed by Mikolov et al. [10]. Vector representations of words created (trained) on sufficiently large text corpus exhibit interesting linguistic regularities, e.g. distance between vector representations of words is an indicator of semantic similarity between the words. Due to these properties, vector representations of text have been used as features in many tasks related to natural language processing, such as word clustering, machine translation etc. In this work we want to investigate if word2vec representations of sequences of words can yield successful features for subject classification of documents.

In case of this work, the word2vec was calculated on a small number of words from training set with the dimensionality of the vector space equal 100. These first results show that

the training corpus was too small to properly calculate this vector values, because fraction of out-of-vocabulary words was almost 69% in binary classification and decreased to 67% in seven category multinational classification. This lead to 78% and 14% accuracy in classification tasks, respectively.

The problem that emerged here was to gather more data. However, it was not feasible to get the required volume of training data (articles) from Wikipedia. Also, artificial text augmentation wouldn't be easy. The solution in order to overcome these limitations which we chose was to use a technique commonly applied in image classification called transfer learning. In Natural Language Processing this is usually done by using pretrained word vectors. One of the easily accessible vectors where available at the Google News word2vec official site [13]. It is worth to mention, that the Wikipedia is a domain specific set of texts, rather different than Google News. The fraction of out-of-vocabulary words was almost equal both for two and seven categories and was 53% comparing to previously mentioned 69% and 67% training set Word2vec. As a result, the efficiency was 92.25% for binary and 86.21% for seven category multinomial classification (table I). It has shown that even the Google News vectors had less corresponding words in the training data, it is enough general, that the LSTM Network can be effectively trained and it works even better than word vectors created from the training data. However, to fit a model with this type of vectors on a single GPU the length of sequences had to be shortened from 1000 to 500. Using the same size of length for simple vectorizer resulted that, it wasn't possible to fit this model. The reason was that Google News Word2vec had 300 vector length per word.

IV. Architecture of the Proposed Recurrent Deep Network

The idea in this paper was to use a standard LSTM network. This network has an advantage over standard Recurrent Neural Network that it doesn't have problems with vanishing gradients [14]. Simpler version of LSTM is GRU (Gated Recurrent Unit), that is almost as effective as LSTM, but it can be trained faster. However, the corpus used in this work had only 40 MB, therefore this wasn't necessary. The number of cells in LSTM was 500.

For each type of approach and number of categories slightly different network was used. In the simple vectorization approach an Embedding Layer was used. This layer encoded each token into 32 dense vectors. This is needed for proper LSTM training. In Word2vec approach this can be omitted. Other differences are in the last, dense, fully-connected layer. The number of units in this layer depends on the number of categories that each network is supporting (Fig. 1-4).

V. Empirical Verification

The results were obtained using Keras framework with Tensorflow backend (table I). All the experiments were done on NVIDIA 1080 Ti GPU with 11GB of RAM. The maximum number of epochs used in training was 50. The duration of

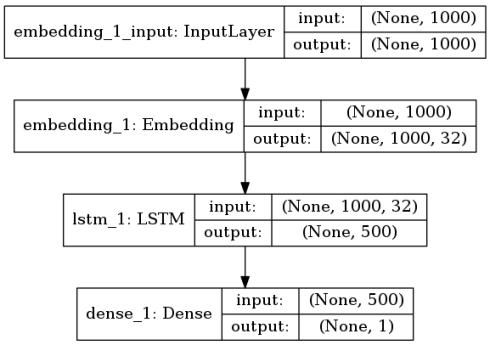

Fig. 1. Neural network used with simple vectorizer for binary classification. The simple encoded vectors with 1000 length are transformed into dense 1000x32 embeddings. LSTM has 500 units and dense layer has 1 unit.

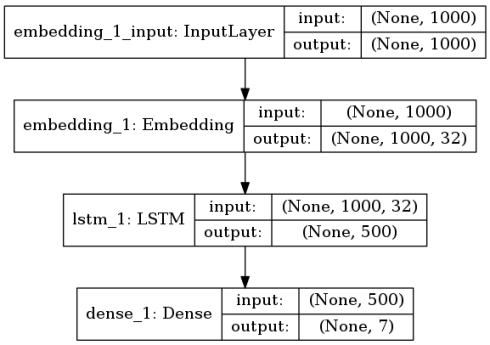

Fig. 2. Neural network used with simple vectorizer for multinomial classification. The simple encoded vectors with 1000 length are transformed into dense 1000x32 embeddings. LSTM has 500 units and dense layer used for classification has 7 units.

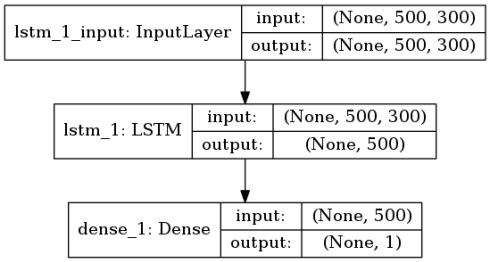

Fig. 3. Neural network used with Word2vec vectorizer for binary classification. The 500 length sequences with 300-length word vectors are put into 500 units LSTM. The dense layer used for classification has 1 unit.

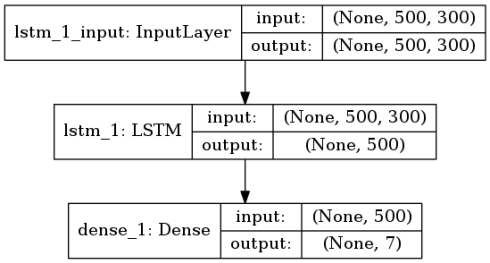

Fig. 4. Neural network used with Word2vec vectorizer for multinomial classification. The 500 length sequences with 300-length word vectors are put into 500 units LSTM. The dense layer used for classification has 7 units.

TABLE I
DEEP NEURAL NETWORKS ACCURACCY

Nr of classes	Simple Vectorizer	Word2vec
2	91.52 %	92.25 %
3	76.53 %	89.52 %
7	58.93 %	86.21 %

Class	Arts	History	Law	Medicine	Religion	Sports	Technology	Avg.
LSTM Prec.	89.5	79.5	92.5	87.5	82.0	90.5	82.0	86.21
CNN Prec.	82.5	62.0	88.0	85.5	83.0	89.5	83.5	82.07

each epoch was from 5 seconds with Google News word2vec to 30 seconds for simple vectorizer multinomial classification. Batch size was 200. For binary classification binary crosstentropy was used, and for multinomial - sparse categorial crossentropy as a loss function. The optimizer was Adam with default parameters, for instance learning rate was 0.001. The activation function was traditional sigmoid.

Also, the result for seven category classification with Google News word2vec for LSTM was compared with Convolutional Neural Network with architecture similar to [22], but without dropout regularization and it achieved ca 82% accuracy (table I). In this table we can also see that LSTM has better precision then CNN in most cases. Therefore CNN was less effective, but it could be trained an order of magnitude faster.

VI. CONCLUSIONS

In this work we demonstrated a method of subject classification of text documents with the documents represented by sequences of words, which were used for training an LSTM neural network. We tried several ways of coding words appearing in the sequences, and found that the most promising results of classification were obtained with words represented in the word2vec vector space. We used word2vec models trained on a large Google news corpus and found that application of models trained on small corpora does not yield successful features for classification. We evaluated the method based on a sample corpus of Wikipedia articles, and obtained accuracy of ca 86% (accuracy of model trained to recognize one out of 7 subject categories). This best performance was realized with LSTM models trained with word2vec-coded sequences of words; these models outperfomed a standard bag-of-words approach reported in our previous work. Although training deep neural networks is commonly regarded as resource-demanding, we found that training deep LSTM neural models as presented in this case study is now feasible using robust libraries such as Keras with Tensorflow and using mid-range GPU devices (system with 11GB of RAM was used). We also provide some technical hints regarding how such tasks can be solved with deep-learning approach.

The research can be continued in future work in order to increase the accuracy. The idea here would be to create better word vectors on larger corpora then Google News or using other word vector representation such as GloVe (Global Vectors) [23]. Also, some updates of network architectures with regularization method can be considered. However, the results show, that generalization of vector representation is a fundamental part in creating effective Deep Neural Network models for NLP and this vectors can be effectively used for texts that was not their main target.

REFERENCES

[1] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *The Journal of machine learning research*, vol. 3, 2003, pp. 1289–1305.

[2] P. Semberecki and H. Maciejewski, "Distributed classification of text documents of Apache Spark platform," *in Artificial Intelligence and Soft Computing Conference, I* Zakopane, 2016, pp. 621–630.

[3] F. Sebastiani. "Machine learning in automated text categorization." *ACM Computing Surveys*, 34(1):1-47, 2002.

[4] S. Wang and C.D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," *in Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Association for Computational Linguistics*, 2012, pp. 90-94.

[5] K.A. Vidhya, G. Aghila, "A Survey of Naive Bayes Machine Learning approach in Text Document Classification," *International Journal of Computer Science and Information Security*, vol. 7, 2010, no. 2, pp. 206–211.

[6] L. Wang, X. Zhao, "Improved k-nn Classification Algorithm Research in Text Categorization," *in Proceedings of the 2nd International Conference on Communications and Networks (CECNet)*, 2012, pp. 1848–1852.

[7] W. Zi-Qiang, S. Xia, Z. De-Xian, L. Xin, "An Optimal SVM-Based Text Classification Algorithm," *in Fifth International Conference on Machine Learning and Cybernetics*, Dalian, 2006, pp. 13–16.

[8] M. Koppel, J. Schler, S. Argamon, "Authorship attribution in the wild," *Language Resources and Evaluation*, vol. 45(1), 2011, pp. 83–94.

[9] M. Koppel and Y. Winter, "Determining if two documents are written by the same author," *Journal of the Association for Information Science and Technology*, vol. 65(1), 2014, pp. 178–187.

[10] T. Mikolov, K. Chen, G. Corrado, J. Dean, "Efficient Estimation of Word Representations in Vector Space," *in Proceedings of Workshop at ICLR*, 2013.

[11] J. Li, X. Chen, E. Hovy, D. Jurasky, "Visualizing and Understanding Neural Models in NLP ", *CoRR* 2015.

[12] Bird, S., Klein, E., Loper, E.: "Natural Language Processing with Python - Analyzing Text with the Natural Language Toolkit" O'Reilly 2009

[13] "Google News word2vec dataset" https://code.google.com/archive/p/word2vec/

[14] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies". *A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press,* 2001.

[15] S. Hochreiter, J. Schmidhuber "Long Short-term Memory" *Neural Computing* 1997. vol. 9 pp. 1735–1780

[16] J. Chung, C. Gulcehre , K. Cho , Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling", *CoRR* 2014

[17] A. Kumar, O, IRsoy, J. Su, J. Bradbury, R. English, B. Pierce, P. Ondruska, I. Gulrajani, R. Socher, "Ask Me Anything: Dynamic Memory Networks for Natural Language Processing", *CoRR* 2015

[18] Z. Huange, W. Xu, Kai You, "Bidirectional LSTM-CRF Models for Sequence Tagging" *CoRR* 2015

[19] K. Tai, R. Socher, C. Manning "Improved Semantic Representations From Tree-Structured Long Short-Term" *CoRR* 2015

[20] M. Lamar, Y. Maron, M. Johnson, E. Bienenstock, "SVD and clustering for unsupervised POS tagging", *In ACL* 2010, pp. 215–219, July 11–16.

[21] Kaggle "Sentiment Analysis on Movie Reviews" https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews

[22] "How to implement Sentiment Analysis using word embedding and Convolutional Neural Networks on Keras." https://medium.com/@thoszymkowiak/how-to-implement-sentiment-analysis-using-word-embedding-and-convolutional-neural-networks-on-keras-163197aef623

[23] Pennington, J., Socher, R., Manning, C. D.: "GloVe: Global Vectors for Word Representation" *In EMNLP* 2014, pp. 1532–1543