Text Classification with Deep Neural Networks

Maaz Amajd

MIPT, maazamjad@phystech.edu

Zhanibek Kaimuldenov

MIPT, janibek.kaimuldenov@gmail.com

Ilia Voronkov

NRU HSE, ivoronkov@hse.ru

Abstract. In this paper, we analyze the use of different neural networks for the text classification task. The accuracy of the studied text classifiers can be changed by a small number of previously classified texts. This is important due to the fact that in many applications of text classification a large number of unlabeled texts are easily accessible, while the receipt of marked texts is quite a difficult task. The paper also shows that the convolution neural network can work better at the level of words, and does not require knowledge of the syntactic or semantic structure of the language. On the other hand, a recurrent neural network for the level of data representation in the form of a sequence can effectively classify the text. Experimental results obtained for text corpora from two different sources show that using a vector data representation can also improve the accuracy of the classification.

1 Related Work

Text classification is a classic topic for natural language processing and has many important applications in topics such as parsing, semantic analysis, information extraction and web searching [1]. Therefore, it has become a source of attraction for many researchers.

In natural language processing, nowadays the core task in text processing is how to present features. Most techniques, such as bag-of-words where unigram, bigram and more broadly n-grams or some other developed patterns that are being used for feature extraction. In order to extract more useful and distinct features, many methods have developed like LDA [2], PLSA [4], frequency and MI [5]. In spite of the fact that many researchers have developed some more complex features (such as tree kernel) [6] in order to extract more contextual information and accurate word order, but there exist few issues such as data sparseness which has the great impact on the classification accuracy. In deep machine learning, some of the most successful deep learning methods involve deep neural networks. In the past few years, deep neural networks and expeditiously advancement in the direction of pre-trained word embedding has become the source of new opulent ideas of NLP tasks. Word embedding is a distributed feature learning over sequences of words and heavily assuage the data sparsity issues. It is also worthwhile to mention that some researchers [3] [7] present that pre-trained word embedding has the ability to extract useful syntactic and semantic regularities. In addition, some composition-based methods were proposed to extract the semantic representation of text with the help of word embedding. To construct sentence representation [8] [9] [10] suggested the idea of *Recursive Neural Network* (Recursive NN) that has ended up being effective performance. Recursive NN has the ability to extract the semantic of a sentence by using tree structure technique. Its performance heavily relies on upon the execution of the textual tree development. Nevertheless, the time complexity of contracting such textual tree is at least $O(n^2)$ where n is the length of the text. If a sentence or document is so long, this approach would be too time-taking. Additionally, it can be very difficult to develop a relationship between two sentences by a tree structure. Consequently, Recursive NN is unsatisfactory for molding long sentences or document.

Recurrent Neural Network (Recurrent NN) is another type of model that shows a time complexity $O(n)$. This model investigates a text word to word and saves the semantics of all the past text in a rigid-sized hidden layer [11]. There is no doubt that it has the ability to capture the semantics of a big text but it is a biased model. It means that it focuses on later words than earlier words that cause to minimize the efficiency of capturing the semantics of a whole document as all the words have the same probability to appear in the sequences of words.

Convolutional Neural Network (CNN) is introduced in natural language processing in order to solve the biases issue. Convolutional neural networks initially were used for video recognition tasks, but recent works [12] showed that CNN can also be applied to NLP (natural language processing). Social media and networks become very interesting topic for scientists, since nowadays more and more people are sharing their opinions on different subjects online. CNN has the ability to extract the semantic of texts in a very systematic way as compared to recurrent or recursive neural networks with time complexity $O(n)$. It can capture critical dialogues in a text by using a max-pooling layer. Although previous research on CNN reveals the kernel technique as a rigid window [13] [14]. In addition, it is so hard to find the most suitable size of a kernel (window size). If a size of the kernel is huge, it would be a cause of many parameter spaces (possibly hard to train it) whereas small size kernel may lead inaccurate results by missing discriminative information.

In text classification, the crux of this direction of NLP mainly predominantly concentrates on three parts: how to design techniques to capture nice features, how to capture appropriate features by using designed techniques and last one is how to design distinctive sorts algorithms for machine learning. In text analysis, Bag-of-words is the most commonly used feature capturing tool. Furthermore, there are some others features selection tools for complex features selection, such as noun phrases [15], part-of-speech tags and tree kernels [18]. The central goal of feature selection tools is to eliminate inutility and noisy features in the text in order to get good performance of classification tasks. The most well-known component selection strategy is expelling the stop words (e.g., "the", "a", "an"). In order to capture valuable features, some modern techniques, such as L1 regularization [17], mutual information [5], or information gain are being used. In Machine learning process, Machine learning algorithms frequently utilize classifiers, for example, logistic regression (LR), support vector machine (SVM) and Naïve Bayes (NB) classifiers etc. Nonetheless, these techniques contain the issues of data sparsity problem.

Artificial intelligence revolutionizes the field of deep learning especially *Deep Neural Networks* [19], word representation learning [20] and came up with to deal with data sparsity issues. In the meanwhile, there are many others neural networks models have been suggested for word representation learning [3] [7] [16] [13] [21] [22]. In the text analysis, word embedding is the neural representation of a word is that is a real-valued vector. The word embedding technique makes us capable of assessing word relevance by just utilizing the distance between two embedding vectors.

Word embeddings with the pre-trained technique play an important role in getting the best performance results of neural networks in many NLP tasks. To foresee the sentiment of a sentence [10] employ semi-supervised recursive autoencoders. In the same way, by also using a recurrent neural network [9] suggested a method for paraphrase detection. However, to examine the sentiment of phrases and sentences [23] introduced a new method of a recursive neural tensor network. In [21] employs the recurrent neural network to construct the language models. For dialogue act classification, [14] suggested a novel recurrent network. For semantic role labeling [13] presents convolutional neural network.

2 Experimental Setup for Comparison

Multi-Layer Perceptron (MLP) with a single hidden layer is used. Word Embedding's is fed as input to the neural network. Word Embedding's layer is the first layer in a model. A 32-dimension vector is used to represent each word. For an experiment, only top 5,000 most frequent words in the dataset are used to set the vocabulary. A movie review is bounded at 500 words, truncating longer reviews and padding shorter reviews with zero value so that they are all the same length for modeling.

Finally, the MLP model is defined by creating a word embedding layer as the first layer. The word vector size to 32 dimensions and the input length to 500. The output of this first layer would be a matrix with the size 32 x 500. Embedded layers output will be flattened to one dimension then use one dense hidden layer of 250 units with a rectifier activation function. The output layer has one neuron and will use a sigmoid activation to get the output values between 0 and 1 (probability-like values) as predictions. A batch size of 128 is used for training the model. Adam algorithm is used in training because it gives best solutions by controlling the learning rate [24]. Basically, it uses moving averages of the parameters (momentum) that allow it to use a larger effective step size, and the algorithm will converge to this step size without fine tuning. After the model is trained, evaluation is done to check its accuracy on the test dataset. The proposed model achieves a score of 87.16% accuracy. A greater accuracy can be achieved by training this network using a larger embedding and with the addition of more hidden layers.

Convolutional Neural Networks (CNN), in biological terms, are inspired variants of MLP. They were designed to honor the spatial structure in image data while being vigorous to the position and orientation of learned objects in the picture. This same idea can be used on sequences, such as the one-dimensional sequence of words in a movie review. This is how the same properties that make the CNN model more useful for learning to identify objects in pictures can assist to learn pattern (structure) in paragraphs of words, namely the methods invariance to the specific position of features.

Word Embedding's is fed as input to the convolutional neural network. Word Embedding's layer is the first layer in the architecture of a model. A 32-dimension vector is used to represent each word.

Eventually, define a convolutional neural network model for the experiment. This time, after the Embedding input layer, a Conv1D layer is inserted. This Conv1D layer has 32 feature maps and reads 3 (kernel size) vector elements of the word embedding at a time. The convolutional layer is followed by a 1D max pooling layer with a length and stride of 2 that halves the size of the feature maps from the convolutional layer. The rest of the network is the same as the MLP. It is important to note that Conv1D layer conserves the dimensionality of Embedding input layer of 32-dimensional input of 500 words. The pooling layer compresses this representation by halving it. One main feature of CNNs is that units share weights, which greatly minimize the amount of computation needed for the model training. It is also seen that CNNs are better at capturing the spatial relationships between words. After training and testing the neural network, the model achieves an accuracy of 87.71%.

LSTM for Sequence Classification: Sequence classification is a predictive modeling task. If a sequence of words is given to the model as input, the task is to predict a category (class) for given the sequence. It is not a trivial case to perform because the issue is that the sequences can vary in length, be comprised of a very large vocabulary of input symbols. Different sequences may have different length of words. When these sequences fed to a neural network model, it may require the model to learn the long-term context or dependencies between patterns (symbols) in the input sequence. LSTM recurrent neural network models are used for sequence classification in a movie review dataset.

A movie review is a variable sequence of words. It means that each movie review contains different amount of words and the sentiment of each movie review must be classified. The words have been substituted by integers that exhibit the ordered frequency (how many times a word occurred) of each word in the dataset. In each review, consequently the sentences are made up of a sequence of integers.

Word Embedding's layer is the first layer in a model. It uses 32 length vectors to indicate each word. The next layer is the LSTM layer that contains 100 memory units (called smart neurons). A dense output layer with a single neuron and a sigmoid activation are used because this is a binary classification task. A sigmoid activation function is used to make 0 or 1 (probability-like values) predictions for the two classes (good and bad). A huge batch size of 64 reviews is utilized to space out weight updates. In the end, a model is created for an experiment. The model is fit for a small epochs only. The reason is that it quickly over fits the task. The model with LSTM performing little tuning achieves an accuracy of 88.03%.

Recurrent neural networks (RNN) like LSTM generally have the problem of overfitting. Dropout is a powerful technique for combating overfitting in LSTM models. Dropout can be applied between layers of the neural network. Dropout is applied by adding new Dropout layers between the Embedding and LSTM layers and the LSTM and Dense output layers in different experiments. In first experiment, the model gets the accuracy of 87.17% that indicate a slightly slower trend in convergence compared to the simple LSTM case. Another technique is adapted to compare the results, add dropout to the input and recurrent connections of the memory units with the LSTM precisely and separately. In the second experiment, the model gets the accuracy of 86.44%. It can be seen that LSTM specific dropout has a more pronounced effect on the convergence of the network than the layer-wise dropout.

CNN and LSTM for Sequence Classification: Convolutional neural networks (CNN) proficient at learning the spatial pattern (structure) in input data but LSTM needs to be larger and trained for longer to achieve the same skill. The IMDB review data does have a one-dimensional spatial pattern (structure) in the sequence of words in movie reviews. CNN may be able to select invariant features for positive and negative sentiment. This learned spatial features may then be learned as patterns (sequences) by an LSTM layer. It is easily possible to add a one-dim CNN and max-pooling layers after the Embedding layer. Subsequently, this Embedding layer feeds the consolidated features to the LSTM. A fairly small set consist of 32 features with a small filter length of 3 (kernel size) is used. In the next step, the pooling layer utilized the standard length size of 2 to reduce (halve) the size of the feature map. The rest architecture is same as explained above. The model gets the accuracy of 86.74%. It can be seen that the model achieves similar results to the LSTM for Sequence Classification with Dropout" although with fewer weights and faster training time.

3 CNN for Twitter sentiment analysis

The model architecture is a modification of the CNN architecture in [2]. Input is a tweet itself, which is representing by a matrix of real numbers, each column is a word of the tweet. The quantity of rows corresponds to a dimensionality of the used word embedding. There are different word embedding models have been used in this work. Almost all of them have dimensionality in 300 figures. CNN has one convolutional layer, one max-pool layer and a full-connected layer with a non-linear function. This network's goal is a binary classification task, predict for a given tweet if it is positive or negative.

More concisely, tweet is a vector of words $x = [c_{1,}c_{2,}\dots c_n]x = [c_{1,}c_{2,}\dots c_n]$, where $c_i \in R^l(l = 300)$. $x = [c_{1,}c_{2,}\dots c_n]$ For a given x model should give an output 1 or 0. Where 1 is a positive class, 0 is a negative one. The convolutional layer has 3 different filter lengths: 3, 4, 5, there are 100 different filters producing unique feature map by each length. Since there is no padding in convolutional layer, output of each filter has different length, hence one max-pool layer just takes one the most illustrious feature from a feature map. The output of the max-pool layer then flows to full-connected layer with non-linearity, the last step is generating an output label by using the softmax function. The models were tested with different non-linear activation function. The Sigmoid function was chosen for next steps of the experiment, since it produces almost the same performance in a shorter time (Table 1):

Table 1.

Activation function	The train time of one epoch, secs	Accuracy (F-score), %
Relu	T≃190 seconds	88.3 %
Sigmoid	T≃130 seconds	89,3 %
Tanh	T≃210 seconds	89.4%

The model was tested with SemEval-2016 [25] dataset using 2 classes of tweets, positive and negative. The positive labeled tweets quantity is 8306, the negative ones are 3190. For word embeddings were used 4 different pre-trained models: word2vec [26], glove-twitter, glove-wikipedia, glove-common. Glove refers to Global Vector Representation, models were trained on twitter, wikipedia or common internet respectively. Glove models are accessible online from [27].

The words from tweet that weren't found in a pre-trained word embeddings are initialized with random numbers and are corrected during the train stage. The dataset was tested by using cross-validation method with 10 parts (10-fold CV used).

4 Results and conclusion

Since the aim of the first experiment is to classify texts of IMDB movie review dataset. The results will be judged separately so that the scores of different techniques can be compared. The results are obtained using the commonly applied method of tenfold cross-validation which calculates average classification accuracy, with accuracy simply defined as the number of correctly predicted labels. For evaluation purposes, accuracy scores are often compared to the majority Baseline, which is the accuracy score obtained when the label from the largest class (i.e. the label with the highest prior probability) is assigned to each document.

Naïve Bayes (multiple entry on text) was applied to the IMDB data on three different stop words list; once with unigrams, once with combinations of unigrams and bigrams, and once with combinations of unigrams, bigrams and trigrams as features for classification. The lists of stop words were selected from different sources (NLTK (Natural Language Toolkit) and Google stop words). NLTK stop words file indicated pretty good results for sentiment classification.

Multilayer neural network and Convolutional neural network was applied to the IMDB dataset. The data distribution was 50% for both training and testing. The model achieves a score of 87.16% and 87.71% accuracy after two epochs respectively. These are good predicting models as compared to the traditional machine learning techniques. A fairly good results can also be achieved after several epochs. CNN show better performance because it has good architecture to highlight the text patterns (sequences) in the training.

The LSTM layer of 100 memory units (called smart neurons) model is used to the IMDB dataset for sequence classification. The data distribution was 50% for both training and testing and the model achieves an accuracy of 88.03% after three epochs. LSTM show better performance because its architecture contain memory cells that can memorize and forget text patterns (sequences). Convolutional neural networks and LSTM were applied to the IMDB dataset for sequence classification with the same data distribution. The model achieves an accuracy of 86.74% after three epochs.

If we compare all the results for IMDB dataset, we can see that CNN show best performance for classification task. On the other hand, the LSTM model achieves good performance for sequence classification. Subsequently, we show their training time and compare their performances (Table 2).

Table 2.

Model	*Accuracy*	*Time(sec)*
Naïve-Bayes	83.29%	1120
Multilayer NN	87.16%	72
Conv NN	87.71%	88
LSTM	88.03%	2230
LSTM(D)	86.44%	1999
CNN+LSTM	86.74%	869

As for comparison for second experiment for short messages (twitter) the results with different word embeddings models should be analyzed with knowing that the number of random initialized words was quite big, from 46% (for word2vec) to 37% (for glove-common). Also glove models from Twitter are 200 dimensions vector, whereas others have 300 dimensionality. As it can be seen on results (Table 3), the relevance of information and its vastness are crucial points for a choice of the word embeddings.

Table 3.

Method	*Accuracy*
Word2vec	89.4 %
Glove-Twitter	89.17 %
Glove-Wikipedia	88.6 %
Glove-Common	89.4 %
SVM with hand-made features [28]	68.46 %
SVM with emoticons and etc. [31]	80.2 %
Hand-made features(no machine learning at all) [30]	60.5 %
Naïve-Bayes algorithm [29]	63 %

Results show that convolutional neural networks are more efficient with sentiment analysis then other machine learning algorithms, and all machine learning algorithms are much more efficient then hand-made features according analysis big datasets. Moreover, convolutional neural networks showed a very high learning rate for different text corpora (IMDB and SemEval-2016).

References

1. Aggarwal, C. C., and Zhai, C. (2012). A survey of text classification algorithms. In Mining text data. Springer. 163-222.
2. Hingmire, S.; Chougule, S.; Palshikar, G. K.; and Chakraborti, S. (2013). Document classification by topic labeling. In SIGIR, 877-880.
3. Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. (2003). A Neural Probabilistic Language Model. JMLR 3:1137-1155.
4. Cai, L., and Hofmann, T. (2003). Text categorization by boosting automatically extracted concepts. In SIGIR, 182-189.
5. Cover, T. M., and Thomas, J. A. (2012). Elements of information theory. John Wiley Sons.
6. Post, M., and Bergsma, S. (2013). Explicit and implicit syntactic features for text classification. In ACL, 866-872.
7. Mikolov, T.; Yih, W.T. and Zweig, G. (2013) Linguistic regularities in continuous space word representations. In hlt-Naacl, 746-751.
8. Sahami, M., Dumais, S., Heckerman, D., Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. . Learning for Text Categorization: Papers from the AAAI Workshop, pp. 55-62. Tech. rep. WS-98-05, AAAI Press.
9. Socher, R.; Huang, E. H.; Pennington, J.; Ng, A. Y.; and Manning, C. D. (2011a). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In NIPS, volume 24, 801-809.
10. Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; and Manning, C. D. (2011b). Semi-supervised recursive autoencoders for predicting sentiment distributions. In EMNLP, 151-161.
11. Elman, J. L. (1990). Finding structure in time. Cognitive science 14(2):179-211.
12. Kim Y. Convolutional neural networks for sentence classification arXiv preprint arXiv:1408.5882. – 2014.
13. Collobert R. et al. Natural language processing (almost) from scratch Journal of Machine Learning Research. – 2011. – Т. 12. – №. Aug. – С. 2493-2537.
14. Kalchbrenner, N., and Blunsom, P. (2013). Recurrent convolutional neural networks for discourse compositionality. In Workshop on CVSC, 119-126.
15. Lewis, D. D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In SIGIR, 37-50.
16. Mnih, A., and Hinton, G. (2007). Three new graphical models for statistical language modelling. In ICML, 641-648.
17. Ng, A. Y. (2004). Feature selection, l1 vs. l2 regularization, and rotational invariance. In ICML, 78.
18. Post, M., and Bergsma, S. (2013). Explicit and implicit syntactic features for text classification.

19. Hinton, G. E., and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. Science 313(5786):504-507.
20. Bengio, Y.; Courville, A.; and Vincent, P. (2013). Representation learning: A review and new perspectives. IEEE TPAMI 35(8):1798-1828.
21. Huang, E. H.; Socher, R.; Manning, C. D.; and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. . In ACL, 873-882.
22. Mikolov, T. (2012). Statistical language models based on neural networks. . Ph.D. Dissertation, Brno University of Technology.
23. Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In EMNLP, 1631-1642.
24. Kingma,P.D; Ba,L.J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
25. SemEval-2016 URL: http://alt.qcri.org/semeval2016/ (accessed date : 31.05.2017)
26. Mikolov T. et al. Distributed representations of words and phrases and their compositionality //Advances in neural information processing systems. – 2013. – C. 3111-3119.
27. Glove URL: https://nlp.stanford.edu/projects/glove/ (accessed date: 31.05.2017)
28. Mohammad S. M., Kiritchenko S., Zhu X. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets //arXiv preprint arXiv:1308.6242. – 2013.
29. Gamallo P., Garcia M. Citius: A Naive-bayes strategy for sentiment analysis on english tweets //Proceedings of SemEval. – 2014. – C. 171-175.
30. Agarwal A. et al. Sentiment analysis of twitter data //Proceedings of the workshop on languages in social media. – Association for Computational Linguistics, 2011. – C. 30-38.
31. Go A., Bhayani R., Huang L. Twitter sentiment classification using distant supervision //CS224N Project Report, Stanford. – 2009. – T. 1. – No. 12.