Text classification based on the use of convolutional neural networks

E.S. Popova[1], V.G Spitsyn[1]

[1]Tomsk Polytechnic University, Lenin Ave. 30, Tomsk, Russia, 634050

Abstract. The article is devoted to neural network text classification algorithms. This paper presents the main components of the text classification system, as well as the same problems associated with the use of the architecture of convolutional neural networks. The algorithm for obtaining vector representations of the dictionary is described.

1. Introduction

The tasks of natural language processing (NLP) are becoming increasingly relevant due to the constantly growing amount of information on the Internet and the need to navigate it. One of the widely used tasks in natural language processing and controlled machine learning in various business problems is text classification, an example of a learning task with a teacher, since a labeled data set containing text documents and their labels are used to train the classifier. The purpose of text classification is the automatic classification of text documents into one or more predefined categories. The following are special cases of the task of classifying texts:

- Text sentiment analysis.
- Spam detection.
- Automatic tagging of customer requests.
- Categorizing news articles on predefined topics.

Text classification is widely used in areas and applications like sentimental analysis (IMDB, YELP reviews classification), sentimental analysis of the stock market, a smart answer used by GOOGLE. It a very active area of research in both academia and industry. Today, in addition to classic text mining algorithms, methods based on in-depth training of neural networks, which offer a flexible and versatile approach to representing the world in the form of visual and linguistic information, have become widespread. The following are neural network architectures that can be used to solve the problem of text classification:

- Convolutional Neural Network (CNN).
- Recurrent Neural Network (Recurrent Neural Network, RNN).
- Hierarchical Attention Network (HAN).

In this article, convolutional neural networks will be used to solve the problem, which was first introduced in 1998 by the French researcher Jan Lekun, as a development of the neocognitron model and designed for effective image recognition.

CNN is commonly used in computer vision, but recently they have been actively applied to various tasks of NLP and based on article [2] from a team of authors from Intel and Carnegie-Mellon University, they are even better suited than recurrent neural networks, RNN), who have reigned

supreme in this area over the past years. This article uses the Keras machine-learning framework and the Python programming language to solve the problem.

2. Training set

As datasets for training and neural network testing will use a set data consisting of 50,000 IMDB movie reviews on English specially selected for analysis tonality. The tone in the sample is binary, i.e. IMDB – rated less than 5 was assigned a score of 0, and a rating of at least 7 – rating 1. For each film, no more than 30 reviews. It should also be noted that in the sample there are no reviews rated 5 or 6 because cannot be unambiguously attributed to positive or negative reviews therefore they are not fit into the binary classification model. All surveys in the sample are randomly mixed. Set The data has the following structure:

- Id - unique identifier of each review.
- Sentiment - mood review; 1 for positive feedback and 0 for negative feedback.
- Review - review text.

3. Text preprocessing

Text preprocessing allows to reduce the original feature space, without loss of useful information. This is an algorithm. Below are the main methods of morphological and syntactic preprocessing of the text, which are part of linguistic analysis, which is the basis for many modern approaches to text mining, and includes the following steps [3]:

- Tokenization is the very first step in processing text. It consists in splitting long lines of text in smaller: we divide paragraphs into sentences, sentences for words.
- Normalization – for high-quality text processing should be normalized. All words are given to one register, punctuation marks are removed, the abbreviations are spelled out, the numbers are given to their text writing, etc. Normalization needed for unification of text processing methods. Stamming is the elimination of appendages to the root, then there is a suffix, prefix, ending and lead words to the base.
- Lemmatization is close to stemming. Difference in that lemmatization leads a word to meaning canonical form of a word (infinitive for a verb, nominative singular – for nouns and adjectives). For example, charter – charter, prices – price, best-good.
- Delete stop words. Stop words – words that are not carry no semantic load. They are also called noise words. For example, in English it is articles, in Russian – interjections, unions, mats, etc.

In this paper, some of the methods listed above will be used to achieve a better classification quality. It is necessary that the source texts contain as little as possible data that does not carry useful information, for example, in the sample there are HTML tags such as </ br>, abbreviations and punctuation. To remove HTML tags, use the Python library BeautifulSoup Package. A regular expression package was used to remove punctuation characters. Further, the entire sample was reduced to lower case. Also in the sample there are stop words such as “a”, “and” “is” “the” and others that do not carry a semantic load. To remove them, the Natural Language Toolkit (NLTK) library was used.

4. Vector representation of words

The vector representation is considered the starting point for most NLP tasks and makes deep learning effective on small data sets. It also underlies many natural language processing systems such as Amazon Alexa, Google translate, etc. This method matches a text word with a certain numerical vector of a fixed dimension. Vectors are constructed in such a way that words found in similar contexts have similar vector representations.Vector representations of words have various useful properties and can be used as follows:

- To search for synonyms or typos in search engines queries.
- Reflections of semantic proximity between words.

- Used as attributes for solving a variety of tasks: identify named entities, tagging parts of speech, machine translate, clustering documents, document ranking, pitch analysis.

Below are ways to get vector representations:

- One-hot encoding.
- SVD.
- Topic modelling.
- Word2vec, GloVe, FastText, StarSpace.

Let's take a closer look at the GloVe vector representation technique from Stanford University, which is popular and is often used for NLP tasks. GloVe - designed for statistical processing of large arrays of textual information. GloVe collects statistics on the cooccurrence of words in phrases, after which the neural network methods solves the problem of reducing the dimension and outputs the compact vector representations of words to the maximum extent reflecting the relationships of these words in the processed texts. For more convenient work with vectors by representations in this work, a previously trained model GloVe was used, which is a file containing tokens and associated word vectors. In particular, a 100-dimensional version of the GloVe model consisting of 400 thousand words will be used, calculated on the data of the English Wikipedia in 2014 with 6 billion tokens.

5. CNN for text classification

There are several approaches using convolutional neural networks for the task of text classification, in this paper the approach based on the encoding of words described in the article[3] was applied. With this approach, each word in the text is mapped to a vector of fixed length, then a matrix is drawn from the vectors obtained for each sample object, which, similarly to images, is fed to the input of a convolutional neural network.

Figure 1 shows an example of a convolutional neural network using word coding.

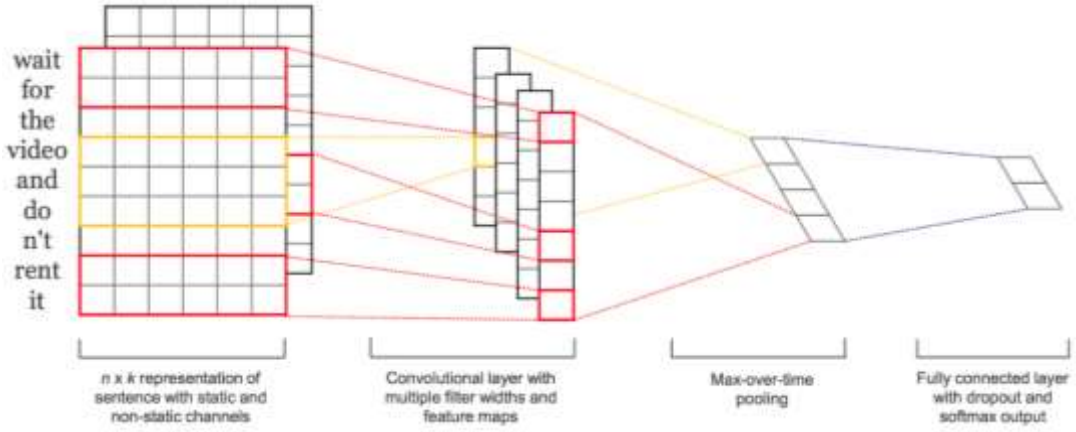

Figure 1. CNN example using word coding.

The following is the convolutional neural network configuration using the GloVe algorithm, described in Section 4, for data classification. The resulting configuration consists of 3 convolutional layers, each of which contains 128 feature maps and has a 5x5 convolution window and 3 subsampling layers with a subsample window size of 5 × 5 and 35x35 on the last layer. The network also includes a reshaping layer and 2 fully connected layers. The activation function on all layers except the last is Relu, on the last one is Softmax.

6. Test results

The During the experiment, the following results were obtained. Figure 2 shows a graph of changes in the accuracy of text recognition during the training of a convolutional neural network using the GloVe algorithm, for 10 learning epochs and a mini sample size of 128. The figure shows that the maximum recognition accuracy is reached at the 7th epoch and reaches 88.9%.

Based on the results obtained, it can be concluded that the selected number of learning epochs is redundant and a further increase in recognition accuracy can be achieved due to more precise adjustment of network hyperparameters, such as small sample size, convolution window size, number of feature maps. It can also be concluded that the network has coped with the task of determining the emotional tonality of the proposed texts.

Table 1. Convolutional neural network configuration.

Layer type	Activation function	Number of customizable parameters
Input layer	–	0
Embedding layer	–	8420100
Convolution layer, number of feature cards: 128, convolution kernel: 5x5	ReLU	64128
Samples layer, number of feature cards: 128, subsample window: 5x5	–	0
Convolution layer, number of feature cards: 128, convolution kernel: 5x5	ReLU	82048
Samples layer, number of feature cards: 128, subsample window: 5x5	–	0
Convolution layer, number of feature cards: 128, convolution kernel: 5x5	ReLU	82048
Samples layer, number of attribute cards: 128, subsample window: 35x35	–	0
Reshape layer	–	–
Full connected layer, number of neurons: 128	ReLU	16512
layer, number of neurons: 2	Softmax	258
Total number of adjustable parameters		8 665 094

7. Conclusion

As a result of the study, the main groups of NLP tasks were identified, and methods for the preprocessing and vectorization of texts were considered. Also in the course of the study the possibility of using convolutional neural networks for the task of text classification was studied. Based on the results obtained, it can be concluded that to achieve greater network accuracy, you can:

- Configure hyperparameters.
- Additionally improve text preprocessing.
- Use dropout layers.

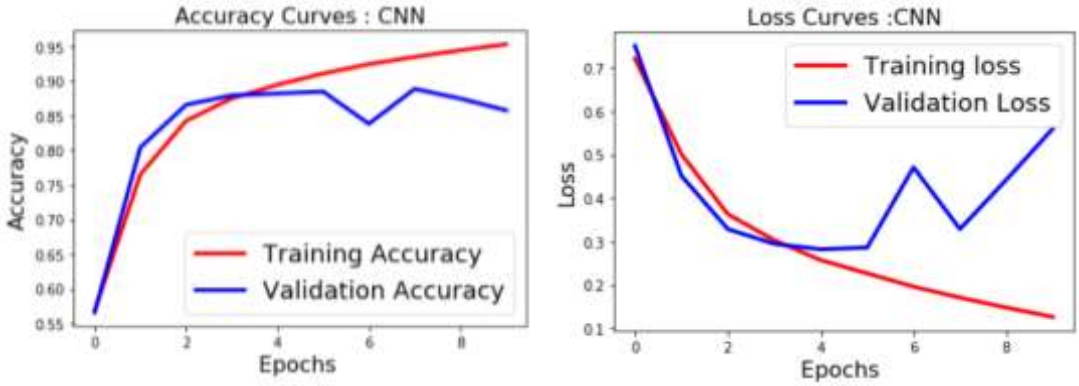

Figure 2. Change the accuracy and loss of recognition of key text.

8. Acknowledgments

The work was performed in the framework of the Program for improving the competitiveness of TPU with the financial support of the Russian Foundation for Basic Research in the framework of the research project No. 18-08-00977 A.

9. References

[1] Fedyushkin, N.A. Concept, problems and types of text mining - Problems and advances in science and technology / N.A. Fedyushkin, S.A. Fedosin // Collection of scientific papers on the basis of the international scientific-practical conference. – 2016. – Vol. 3. – P. 206.
[2] Bai, S. Antivirus Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling / S. Bai, J.Z. Kolter, V. Koltun // arxiv.org/abs/1803.01271, 2018.
[3] Smirnov, I.V. Introduction to the analysis of natural languages: study guide, 2014.
[4] Kim, Y. Convolutional Neural Networks for Sentence Classification // Conference on Empirical Methods of Natural Language Processing (EMNLP), 2014. – P. 1746-1751.
[5] Zhang, X. Character-level Convolutional Networks for Text Classification / X. Zhang, J. Zhao, Y. LeCun // Advances in Neural Information Processing Systems, 2015. – P. 649-657.
[6] Spitsyn, V.G. Intellectual systems: study guide / V.G. Spitsyn, Yu.R. Choi. – Tomsk: Publishing house of Tomsk Polytechnic University, 2012. – 176 p.
[7] Khaikin, S. Neural networks: a full course – M.: Williams, 2006. – 1104 p.
[8] LeCun, Y. Efficient BackProp in Neural Networks: Tricks of the Trade / Y. LeCun, L. Bottou, G. Orr, K. Muller – Springer, 1998.
[9] LeCun, Y. Scaling learning algorithms towards AI / Y. LeCun, Y. Bengio – MIT Press, 2007.