Sentiment Analysis of a document using deep learning approach and decision trees

Arman S. Zharmagambetov
LLC AlemResearch
Almaty, Kazakhstan
armanform@gmail.com

Alexandr A. Pak
Institute of ICT
Almaty, Kazakhstan
aa.pak83@gmail.com

***Abstract* — The given paper describes modern approach to the task of sentiment analysis of movie reviews by using deep learning recurrent neural networks and decision trees. These methods are based on statistical models, which are in a nutshell of machine learning algorithms. The fertile area of research is the application of Google's algorithm Word2Vec presented by Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean in 2013. The main idea of Word2Vec is the representations of words with the help of vectors in such manner that semantic relationships between words preserved as basic linear algebra operations. The extra advantage of the mentioned algorithm above the alternatives is computational efficiency. This paper focuses on using Word2Vec model for text classification by their sentiment type.**

Keywords — NLP, sentiment analysis, deep learning, machine learning, text classification;

I. INTRODUCTION

In recent years, there is an active trend towards using various machine learning techniques for solving problems related to Natural Language Processing (NLP). One of these problems is the automatic detection of emotional coloring (positive, negative, neutral) of the text data, i.e. sentiment analysis. The goal of this task is to determine whether a given text (or movie review in our case) is positive, negative or neutral according to its influence on the reputation of particular movie. The difficulty of sentiment analysis is determined by the emotional language enriched by sleng, polysemy, ambiguity, sarcasm, all this factors are misleading for both humans and computers.

The high interest of business and researchers to the development of sentiment analysis are caused by the quality and performance issues. Apparently the sentiment analysis is one of the most in-demand NLP tasks. For instance, there are several international competitions and contests [1], which tries to identify the best method for sentiment classification. Sentiment analysis had been applied on various levels, starting from the whole text level, then going towards the sentence and\or phrase level.

It is obvious that similar sentimental messages (text, sentence…) can have various thesauruses, styles, and structure of narration. Thus, points corresponding to the similar messages can be located far away from each other that makes the sentiment classification task much harder [2]. The scientific novelty of the paper is in reducing the space of thesaurus into the meaning's space with the help of Word2Vec algorithm [3] in the sentiment classification task thereby decreasing the negative effect of points' divergence. In the section II there are the overview of previous works connected to sentiment classifications in English. The section III contains short description of data structures and objectives pursued. In the section IV there are details of algorithm implementation and the main ideas of data processing. And traditionally the paper concludes section V, which contains the discussion and results.

II. RELATED WORK

The study of sentiment analysis have relatively small history. Reference [4] is generally considered the principal work on using machine learning methods of text classification for sentiment analysis. The previous works related to this field includes approaches based on maximum relative entropy and binary linear classification [5] and unsupervised learning [6].

Most of these methods use well known features as bag-of-words, n-grams, tf-idf, which considered as the simplest one [7]. But as show the experiment results the simple models often works better than complicated ones. Reference [7] use distant learning to acquire sentiment data. Additionally, since they mostly work with movie comments and tweets, they used additional features as ending in positive emoticons like ":)" ":-)" as positive and negative emoticons like ":(" ":-(" as negative. They build models using Naive Bayes, MaxEnt and Support Vector Machines (SVM), and they report SVM outperforms other classifiers. In terms of feature space, they try a Unigram, Bigram model in conjunction with parts-of-speech (POS) features. They note that the unigram model outperforms all other models.

The other feature is syntactic meta information. It's obvious that the recursively enumerable grammar describes the most complete of any natural language. The computational performance of the best syntactic parser of context free grammar is linear, so syntactic information is expensive for the sentiment analysis task. However, those experiments that involved dependency relations showed that syntax contributes significantly to both Recall and Precision of most algorithms. For the task of text classification in general see [8-10] deal with a task sentiment classification based on syntactic relations. In

reference [11] it was shown that POS-tagging and other linguistic features contributes to the classifier accuracy. The experiments were conducted on feedback data from Global Support Services survey.

In this paper we perform extensive feature analysis and show that the use of only 100 abstract linguistic features performs as well as a hard unigram baseline.

III. DATASET

The labeled data set consists of 50,000 IMDB [12] movie reviews, specially selected for sentiment analysis. The sentiment of reviews is binary, meaning the IMDB rating < 5 results in a sentiment score of 0, and rating >=7 have a sentiment score of 1. No individual movie has more than 30 reviews. The 25,000 review labeled training set does not include any of the same movies as the 25,000 review test set. In addition, there are another 50,000 IMDB reviews provided without any rating labels. Each entry on this dataset consists of the following field:

- id - Unique ID of each review.
- sentiment - Sentiment of the review; 1 for positive reviews and 0 for negative reviews.
- review - Text of the review.

The goal is to increase the accuracy(precision and recall) in sentiment classification of test dataset.

IV. ALGORITHM

The core of our sentiment analyzer algorithm is the word2vec model of word presentation and random forest for message classification task. The entire method which described in this paper was implemented in python language. The preprocessing includes the following: (1) The all HTML tags, punctuations, were removed by “Beautiful Soup” python library. There are HTML tags such as "
", abbreviations, punctuation - all common issues when processing text from online. (2) Moreover, numbers and links were replaced by tags NUM and LINK, respectively. (3) Removing stop words. Conveniently, there is Python package Natural Language Toolkit (NLTK) [13] that remove stop words with built in lists.

After preprocessing step, the main features can be extracted. There is no need for lemmatization of each word, because word2vec model carry out the process of dealing with each word.

Word2vec, published by Google in 2013, is a neural network implementation that learns distributed representations for words. Distributed word vectors are powerful and can be used for many applications, particularly word prediction and translation. Here, we will try to apply them to sentiment analysis. It accepts each sentence of large un-annotated corpus. There are two different architectures that solves different tasks. At Fig. 1 continuous bag of words (CBOW) architecture presented, the purpose of such network topology assumes mapping from context to particular term and vice versa at Fig. 2 skip-gram architecture that maps particular term to its context.

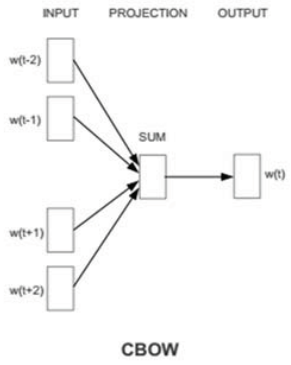

Fig. 1. The architecture of Word2Vec model. The CBOW architecture predicts the current word based on the context. Taken form [3].

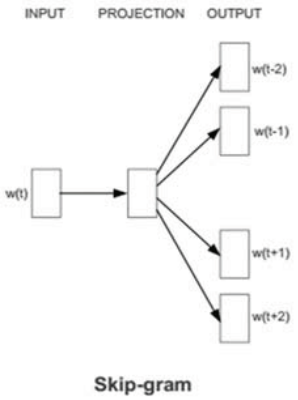

Fig. 2. The architecture of Word2Vec model. The skip-gram predicts surrounding words given the current word. Taken from [3].

Next, the model trains by using deep recurrent neural nets. It uses classical Elman [14] type RNN. In an Elman-type network (Fig. 3), the hidden layer activations h (t) at time step t are computed by transformation of the current input layer x (t) and the previous hidden layer h (t − 1). Output y (t) is computed from the hidden layer h (t).

$$h(t) = f(Wx(t) + Vh(t-1) + b) \quad (1)$$

$$y(t) = g(Uh(t) + c) \quad (2)$$

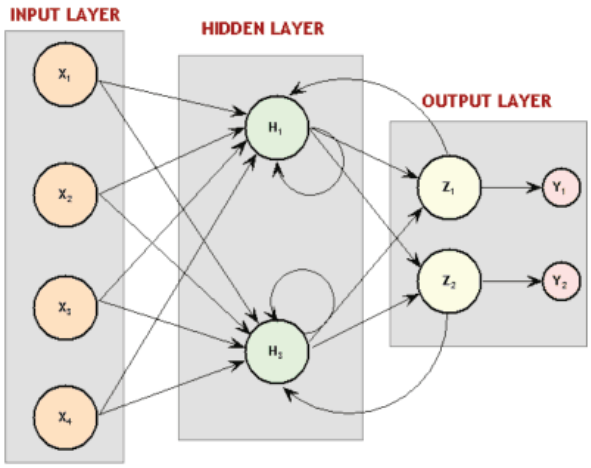

Fig. 3. The classical representation of Deep Recurrent Neural Networks. Taken from [15]

We used “gensim” [16] python library with build in Word2Vec model. It accepts large dataset for training. Therefore we used the whole database of reviews texts with additional 50,000 unlabeled reviews. The following features were used while training for Word2Vec: 300 dimensional space, 40 minimum words and 10 words in context. The vector representation of word has a lot of advantageous. It raises the notion of space, and we can find distance between words and finding semantic similar words. The simple result that can be obtained for such vector presented at TABLE I.

TABLE I. Semantic similar words to ‘man’

Words	Measures
woman	0,6056
guy	0,4935
boy	0,4893
men	0,4632
person	0,4574
lady	0,4487
himself	0,4288
girl	0,4166
his	0,3853
he	0,3829

The next step is clustering such semantic similar words. Word2Vec creates clusters of semantically related words, so another possible approach is to exploit the similarity of words within a cluster. Grouping vectors in this way is known as "vector quantization." The “cython” package [18] with built in word2vec implementation was used in this task. The classical K-means clustering algorithm was used for this purposes. Trial and error suggested that small clusters, with an average of only 5 words or so per cluster, gave better results than large clusters with many words. Then feature for all document can be extracted as it can be done for bag-of-words model. Instead of words they replaced by id of cluster assigned to it and obtain feature vector of document. Generally, there were generated ~3000 clusters, and after filtering and removing unnecessary clusters remained only 2000 clusters of semantic similar words. Consequently, feature vector of each movie review have size 2000.

Then Random Forest classifier was used to identify positive, negative or neutral sentiment of particular document. The Random Forest algorithm is included in scikit-learn (Random Forest uses many tree-based classifiers to make predictions, hence the "forest") [17]. Below, we set the number of trees to 100 as a reasonable default value. More trees may (or may not) perform better, but will certainly take longer to run. Likewise, the more features you include for each review, the longer this will take.

V. Results and Discussions

TABLE II summarize the results on sentiment classification. As mentioned above there are 25,000 reviews were chosen for train and another 25,000 reviews for test data. Also classical “Bag-Of-Words” model with random forest classifier was chosen for comparing results. Despite this simple model, deep learning approach was efficient in terms of words semantics. By grouping together semantic similar words we dig a little bit deeper into the structure of text. So our research in that term can be considered as deep analysis, whereas “Bag-Of-Words” is surface analysis of a text. Therefore hidden meanings can be extracted from reviews.

TABLE II. Semantic similar words to ‘man’

Method	precision	recall	F-measure	accuracy
bag-of-words	85.2%	83.7%	84.4%	84.5%
deep learning	90.3%	87.2%	88.7%	89.8%

Summarizing all mentioned above we can conclude that despite the fact that deep learning approach complicated rather than simple methods as “bag-of-words”, it shows only slightly better results. It may be caused by error level in words’ clustering and noises that appeared in preprocessing step. Also we considered only K-Means clustering of words as the simplest one with approximate number of clusters (5 words in a cluster), therefore clustering method should be improved.

TABLE III. FEATURE VECTORS

	BOW	Deep learning
Features	lemmatized words (5000 most common words), links and smiles were replaced by tags: LINK and ANGRY_SMILE, HAPPY_SMILE. Stopwords are deleted.	2000 semantic similar words clusters, links and smiles were replaced by tags: LINK and ANGRY_SMILE, HAPPY_SMILE. Stopwords are deleted.

The given work shows that surface analysis of a text is not enough for sentiment analysis. Of course, simple methods can be used for systems where high speed and resources are important. But for results oriented services it would be not enough. Even semantic features that extracted from text require additional information. Therefore future works will be related to improvement of given method by working with homonymy and researching syntax structure of the document.

REFERENCES

[1] I. Chetviorkin, P. Braslavskiy, N. Loukachevich, "Sentiment Analysis Track at ROMIP 2011," In Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog 2012", Bekasovo, 2012, pp. 1–14.

[2] A.A. Pak, S.S. Narynov, A.S. Zharmagambetov, S.N. Sagyndykova, Z.E. Kenzhebayeva, I. Turemuratovich, "The method of synonyms extraction from unannotated corpus," In proc. of DINWC2015, Moscow, 2015, pp. 1-5

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

[4] P. Bo and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," In Proceedings of the ACL, 2004

[5] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," In European Conference on Machine Learning (ECML), Springer Berlin/Heidelberg, 1998, pp. 137-142

[6] P.D. Turney, "Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews," Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia, Pennsylvania, 2002, pp. 417-424.

[7] A. Go, R. Bhayani, L. Huang, "Twitter Sentiment Classification Using Distant Supervision," Technical report, Stanford. 2009.

[8] J. Furnkranz, T. Mitchell, and E. Riloff, "A Case Study in Using Linguistic Phrases for Text Categorization on the WWW," In AAAI/ICML Workshop on Learning for Text Categorization, 1998, pp. 5-12.

[9] M.F. Caropreso, S. Matwin, F. Sebastiani, "A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization," Text databases and document management: Theory and practice, 2001, pp. 78-102.

[10] V. Nastase, J.S. Shirabad, M.F. Caropreso, "Using Dependency Relations for Text Classification," In Proceedings of the 19th Canadian Conference on Artificial Intelligence, Quebec City, 2006, pp. 12–25.

[11] M. Gamon, "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," In proc. of COLING-04, Geneva, CH, 2004, pp. 841-847.

[12] Kaggle: Bag of Words Meets Bags of Popcorn https://www.kaggle.com/c/word2vec-nlp-tutorial/data

[13] Natural Language Toolkit http://www.nltk.org/

[14] J.L. Elman, "Finding Structure in Time," Cognitive Science 14 (2), 1990, pp. 179–211.

[15] Timeline of Systematic Data and the Development of Computable Knowledge http://www.wolframalpha.com/docs/timeline/computable-knowledge-history-6.html

[16] Gensim: Topic modeling for humans. https://radimrehurek.com/gensim/

[17] Sci-kit: Machine learning in python. http://scikit-learn.org/stable/

[18] Cython C-Extensions for Python http://cython.org/