HHU at SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Data using Machine Learning Methods

Tobias Cabanski, Julia Romberg, Stefan Conrad
Institute of Computer Science
Heinrich Heine University Düsseldorf
D-40225 Düsseldorf, Germany
tobias.cabanski@hhu.de
{romberg,conrad}@cs.uni-duesseldorf.de

Abstract

In this Paper a system for solving SemEval-2017 Task 5 is presented. This task is divided into two tracks where the sentiment of microblog messages and news headlines has to be predicted. Since two submissions were allowed, two different machine learning methods were developed to solve this task, a support vector machine approach and a recurrent neural network approach. To feed in data for these approaches, different feature extraction methods are used, mainly word representations and lexica. The best submissions for both tracks are provided by the recurrent neural network which achieves a score of 0.729 in track 1 and 0.702 in track 2.

1 Introduction

Analysing texts from the finance domain is a task that can help market traders to make important decisions because research has shown that sentiments and opinions can affect market dynamics (Goonatilake and Herath, 2007). In the meantime, the internet contains a huge corpus of finance news from news websites or social media platforms. Natural language processing methods can be used to analyse this data as, for instance, sentiment analysis. To improve the understanding of the special characteristics of the domain, SemEval-2017 provides Task 5.

2 Related Work

In the task of sentiment analysis, machine learning methods are widely used. One approach is shown in (Agarwal et al., 2011) where a sentiment analysis on twitter messages is performed by combining different features. It was found out that the use of multiple features processed by a support vector machine leads to good classification scores.

The work of (Yadav, 2016) shows that recurrent neural networks can provide a good performance for this task. A powerful system can be created even with doing only a little preprocessing on the text data.

3 Task Description

The SemEval-2017 Task 5 is divided into two tracks which each consider a different data basis (Cortis et al., 2017). The objective of both tracks is the prediction of a sentiment score with reference to a company or a stock in a given piece of text. The sentiment score is a number within the interval $[-1, 1]$ with -1 denoting a very bearish sentiment and $+1$ denoting a very bullish sentiment.

Track 1 is focused on microblog messages about stock market events. The data corpus consists of 1710 annotated messages taken from StockTwits[1] and Twitter[2], whereas a cashtag and the related spans are given in each case.

Track 2 refers to financially relevant news headlines. The annotated data given for the system's training comprises 1156 headlines. For every headline-score pair the corresponding company name is specified. Spans are not given.

4 System Description

Below, the system is outlined. In doing so, preprocessing steps, feature selection, and the used machine learning techniques are described.

4.1 Preprocessing

Preceding the feature extraction, the texts are processed by first expanding contractions and remov-

[1] https://stocktwits.com/

[2] https://twitter.com/

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 832–836,
Vancouver, Canada, August 3 - 4, 2017.

ing URLs in the text.

Emotions generally take an important part in the microblog domain and it was shown that the consideration of punctuation and bracket characters can improve the score because they can express special meaning like facial expressions (Agarwal et al., 2011). After the revision of the data, it shows that these characters rarely occur in the data of track 1 and apparently never occur in the track 2 data. Therefore, these characters are removed.

Furthermore, character strings of multiple white spaces are normalized to length 1 and case insensitivity is introduced. Additional tokenization is done using SpaCy.[3]

4.2 Features

Two different types of features are examined for the construction of the system. In the first feature set the preprocessed data is transformed into a numerical word representation. The second feature set consists of multiple sentiment lexicon resources.

4.2.1 Text Representations

Textual input has to be transformed into a machine-readable representation. For this, the ensuing two approaches are selected.

Word2vec (Mikolov et al., 2013) generates a vector space based representation for a word in the text. Each unique word in the corpus is represented by a feature vector having similar words being positioned near each other in the space. A pre-trained Levy and Goldberg dependency-based model (Levy and Goldberg, 2014) from SpaCy is used as well as a self-trained model which is constructed using gensim[4]. In contrast to the pre-trained model, the self-trained model contains the whole input vocabulary, but bases on a smaller corpus.

4.2.2 Lexica

Since there is only a small amount of training data, it was decided to bring additional information into the system by using different lexica.

SentiWordNet (Baccianella et al., 2010) consists of WordNet's synset corpus (Fellbaum, 1998). Every synset term has a positive and a negative score between zero and one named PosScore and NegScore. It is also possible to calculate the objectivity of a word by subtracting one from the sum of PosScore and Negscore. Since a word can have more than one score according to different meanings, the final score for a word is the mean of all found scores in the SentiWordNet.

VADER lexicon (Hutto and Gilbert, 2014) is a gold-standard sentiment lexicon that is geared to social media platforms. Hence, words and symbols that are not included in conventional lexical resources are contained and scored with continuous values.

Opinion Lexicon (Hu and Liu, 2004) is a resource that is constantly being updated since 2004. The words in this lexicon are classified binary as positive or negative. Like in the VADER lexicon, this resource was also trained on social media data.

MaxDiff Twitter Sentiment Lexicon (Kiritchenko et al., 2014) is an additional lexicon that delivers a continuous score for words. This corpus is mainly trained on Twitter data.

Financial Sentiment Lexicon As word polarities might depend on a specific domain, a lexicon based on the training data from the respective track is created. The Financial Sentiment Lexicon is built from the training data: First, the score of a short message is assigned to all words of the message. Then a vocabulary of all distinct words is built and the scores of identical words are averaged. In the final pruning step words with occurrence less than 0.1% in the data are removed to prevent that very infrequent words have an impact on the score. Also words with occurrence greater than 10% in the data are removed so that very frequent words like stop words don't influence the score.

4.3 Regression Methods

On the one hand, support vector regression is applied. This regression method bases upon a support vector machine (SVM) which is a widely used machine learning method and which delivers good results for various tasks. The implementation of it is realized by the python package Scikit-Learn.[5]

The other regression method that is used is a recurrent neural network (RNN) using a Long Short-Term Memory Cell (Hochreiter and Schmidhuber, 1997). This is an improvement of a standard RNN in which the network is able to keep information

[3] https://spacy.io/

[4] https://radimrehurek.com/gensim/

[5] http://scikit-learn.org

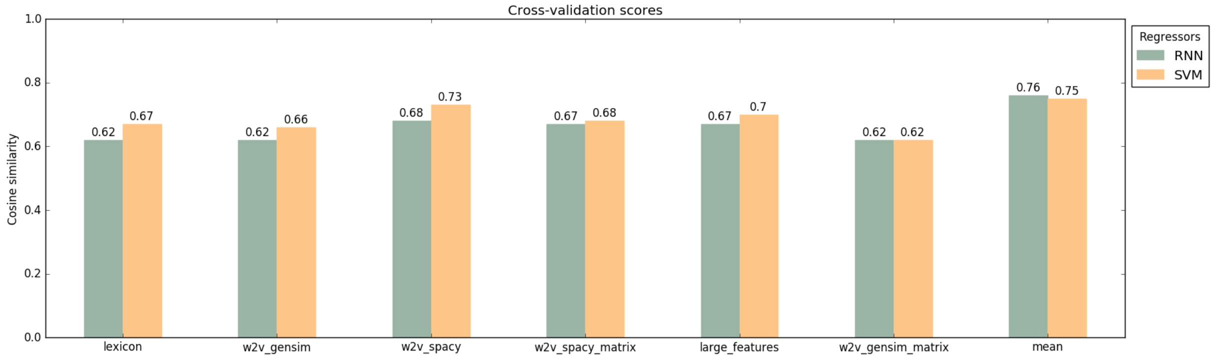

Figure 1: Results of the cross validation of Track 1

over a larger amount of time steps. Unlike the SVM, the RNN can process a message sequentially observing the sentence structure. The implementation is realized by the python package TensorFlow.[6] The model consists of two LSTM cells and a layer count of 10.

5 Results

For every track, the support vector regression and the RNN approach are tested on the testing data using the depicted features. The predicted values are rated by the cosine similarity between the gold standard and the predicted sentiment score. For evaluation of the system, cross-validation is used by creating five subsets out of the training data set.

5.1 Track 1

The results for track 1 are presented in Figure 1. The Figure shows the results of the regression of the features, which are determined by a recurrent neural network and a support vector machine. The presented combination of features has delivered the best result when the average score was formed.

The *lexicon* feature is the sentiment score of every single word in a message, aggregated to the mean for the whole message. *w2v_gensim* relates to the word2vec representation through gensim that has been described in chapter 4.2.1. All word vectors of a short message are averaged to a single feature vector. *w2v_spacy* refers to the second word2vec model named in chapter 4.2.1 which bases on SpaCy. Like described for w2v_gensim, the definite feature vector of a message consists of the average of every word's representation vector. The two matrix features *w2v_spacy_matrix* and *w2v_gensim_matrix* are created in the same way as the word vector features describes before, but they don't average the single word vectors to one vector. Instead, every word vector is represented by a line of a matrix with a fixed count so that every matrix for a message has the same shape. This representation is optimized for the recurrent neural network because it can process one line at a single time step. To use the matrix features in a support vector machine, the matrix will be reshaped to a vector where all word vectors are concatenated to one single vector. The *large_features* are further matrix features, where a word representation is concatenated to the lexicon scores of a single word. For that, a 50-dimensional word vector is created by gensim and then a five-dimensional vector is built by looking up the score for the word in the lexica. This feature has a matrix representation with a fixed line count as in the matrix features described before.

All feature combinations were tested. Figure 1 shows the similarity scores obtained by the individual features and the mean score of all single feature predictions. As the figure shows, this aggregation leads to a cosine similarity which is higher than all single features. The combinations, that are not explicitly presented, did not lead to better results.

When comparing the scores of the SVM and the RNN, it is noticeable that the SVM scores for the single features are the same or better than those of the RNN. But after aggregating the results of the features for the final score, the RNN yields a slightly better result than the SVM. Another considerable thing is the comparison of the matrix feature scores. Due to the optimized representation for a RNN, it is expected that the score of the matrix features processed by the RNN is better than processing them by the SVM. Though, the evaluation shows a different result: The matrix features

[6] https://tensorflow.org

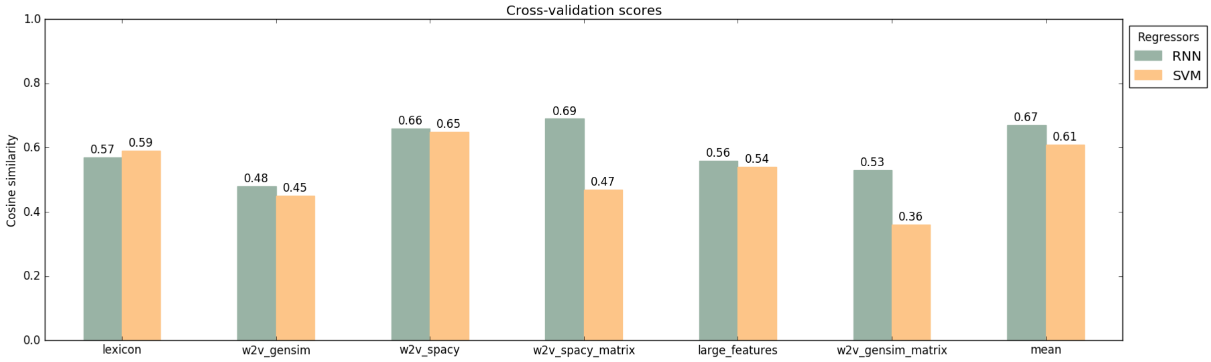

Figure 2: Results of the cross validation of Track 2

perform better when processed by a SVM. This could be caused by the length and structure of the data in track 1. Since the data only contains short snippets of the full message text, which is often only one word, the step-wise data processing of the RNN offers no advantage.

5.2 Track 2

The results of the cross-validation of track 2 are shown in Figure 2. The same features as in track 1 were used to make the results comparable. The mean over all single feature predictions has also been approved as best and been used for the final score.

In comparison to the first track, the cosine similarity is lower for most of the features. This might be due to the fact that, unlike track 1, the full text of a headline is delivered so that the average message length is higher and more words influence the score.

Another fact that is show in figure 2 is that the RNN outputs better results over most of the single features and also over the mean. This can be explained by the structure of the data. Since the full headline text has to be processed, a message consists of more words than in track 1. The three matrix features keep most of the information of the words in a message and the step-wise processing of the feature matrix by the RNN shows much better results than reshaping the matrix to a single vector and processing it by a SVM.

5.3 Official results

In this project, the prediction of the mean-feature-SVM approach and of the mean-feature-RNN approach were each chosen to be submitted.

The official evaluation for track 1 has resulted in a better score for the RNN approach with an overall rank of 6 and a score of 0.729. The SVM approach has a slightly worse result with a score of 0.720. In track 2, also the RNN approach reached a better score than the SVM with a rank of 7 and a cosine similarity of 0.702. The SVM achieved a similarity of 0.655. The official evaluation confirms the cross-validation results by the better performance of the RNN over the SVM.

6 Conclusion and Future Work

It has been shown that the extraction and aggregation of multiple features can lead to a score that performs better than every single feature score. The best single scores are represented by the word vectors where the cosine similarity is near the mean score for track 1 and in one case better than the mean score in track 2. It can also be emphasized that the RNN outputs better scores on longer messages than the SVM, especially when the stepwise procedure is used with the matrix-shaped features.

To further improve the results it is possible to weight the single features before averaging them. To learn the weights, a neural network can be used which takes all single feature scores as input and outputs one value as the sentiment prediction.

Another method to improve the scores is to notice the cashtag or company name. By adding features that take the dependency of the words into account, it is possible to find out which words have an influence on a specific entity. When using a RNN as regression method, it is also possible to pass the entity as an input, for example as a word vector.

References

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment Analysis of Twitter Data. In *Proceedings of the Workshop on Languages in Social Media*. Association for Computational Linguistics, pages 30–38.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association, pages 2200–2204.

Keith Cortis, André Freitas, Tobias Dauert, Manuela Huerlimann, Manel Zarrouk, Siegfried Handschuh, and Brian Davis. 2017. SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, pages 517–533.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Rohitha Goonatilake and Susantha Herath. 2007. The Volatility of the Stock Market and News. *International Research Journal of Finance and Economics* 3(11):53–65.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation* 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pages 168–177.

Clayton Hutto and Eric Gilbert. 2014. VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. In *Eighth International AAAI Conference on Web and Social Media*. Association for the Advancement of Artificial Intelligence, pages 216–225.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014. Sentiment Analysis of Short Informal Texts. *Journal of Artificial Intelligence Research* 50:723–762.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 302–308.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781* .

Vikrant Yadav. 2016. thecerealkiller at SemEval-2016 Task 4: Deep Learning based System for Classifying Sentiment of Tweets on Two Point Scale. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, pages 100–102.