Finki at SemEval-2016 Task 4: Deep Learning Architecture for Twitter Sentiment Analysis

Dario Stojanovski, Gjorgji Strezoski, Gjorgji Madjarov, Ivica Dimitrovski
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius University
Rugjer Boshkovikj 16 1000 Skopje, Republic of Macedonia
{stojanovski.dario, strezoski.g}@gmail.com
{gjorgji.madjarov, ivica.dimitrovski}@finki.ukim.mk

Abstract

In this paper, we present a novel deep learning architecture for sentiment analysis in Twitter messages. Our system **finki**, employs both convolutional and gated recurrent neural networks to obtain a more diverse tweet representation. The network is trained on top of GloVe word embeddings pre-trained on the Common Crawl dataset. Both neural networks are used to obtain a fixed length representation of variable sized tweets, and the concatenation of these vectors is supplied to a fully connected softmax layer with dropout regularization. The system is evaluated on benchmark datasets from the Sentiment Analysis in Twitter task of the SemEval 2016 challenge where our model achieves best and second highest results on the 2-point and 5-point quantification subtasks respectively. Despite not relying on any hand-crafted features, our system manages the second highest average rank on the considered subtasks.

1 Introduction

Twitter sentiment analysis is an area of Natural Language Processing (NLP) dealing with the classification of sentiment polarity in Twitter messages. Most of the approaches to this problem are generally based on hand crafted features and sentiment lexicons (Mohammad et al., 2013; Pak and Paroubek, 2010). These features are then used as input to classifying algorithms such as, Support Vector Machines (SVM) and naive Bayes classifier. However, such approaches require extensive domain knowledge, are laborious to define, and can lead to incomplete or over-specific features.

Deep learning methods for sentiment analysis, on the other hand, handle the feature extraction automatically which provides for robustness and adaptability. Notably, most popular deep learning methods are convolutional neural networks (CNN), which have been shown to achieve state-of-the-art results (Kim, 2014; dos Santos and Gatti, 2014), though some works propose different models such as Recursive Neural Tensor Network (Socher et al., 2013).

Recurrent neural networks (RNN) are intuitive architectures for NLP as they inherently take into account the ordering of words in the text as opposed to CNNs which take only a small limited context window. However, to our knowledge, these networks have not been applied to sentiment analysis in Twitter messages. Le et al. (Le and Zuidema, 2015) report state-of-the-art results with Long Short Term Memory (LSTM) networks on binary and fine-grained classification on the Stanford Sentiment Treebank dataset.

In this paper, we present a novel deep learning architecture for sentiment classification and quantification in Twitter messages. The model consists of a convolutional and a gated recurrent neural network (GRNN). Both neural networks are used to model a suitable representation of a tweet. The feature representations output from the networks are fused and fed to a standard softmax regression classifier. The system leverages unsupervised pre-training of word embeddings. For this, we utilize the publicly available GloVe[1] word embeddings (Pennington et al., 2014), specifically ones trained on the Common

[1] http://nlp.stanford.edu/projects/glove

Crawl dataset. In previous work (Stojanovski et al., 2015), we have experimented with multiple filters with additional window sizes of 4 and 5 and we leave such system implementation for future work.

We evaluate our deep learning system on four out of five subtasks of the Sentiment Analysis in Twitter task (Task 4) (Nakov et al., 2016) as part of the SemEval 2016 challenge. We competed in the 2-point and 5-point classification and quantification. Our model achieves high results on the quantification subtasks, getting second place on Subtask E and attaining the best score on Subtask D.

2 Deep learning architecture

The proposed model for sentiment analysis in this paper, consists of two neural networks. The first is a convolutional neural network with a single filter with windows size of 3. The second part of the architecture is a gated recurrent neural network. The system architecture is presented in Figure 1. The model is implemented using the Keras[2] library for deep learning on a Theano backend.

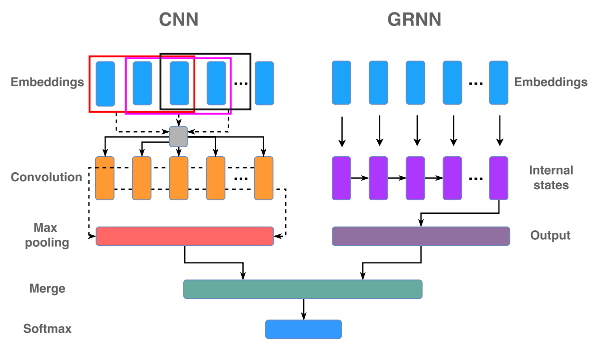

Figure 1: Deep neural network architecture.

2.1 Preprocessing

Twitter constraints tweet length to a maximum of 140 characters. Consequently, users are forced to find new and unpredictable ways of expressing themselves. Determining sentiment in these circumstances is very challenging and, as a result, we apply some preprocessing steps in order to clean tweets from unnecessary information. All URLs and HTML entities are removed from the tweets along with punctuation with the exception of question and exclamation marks. Emoticons and Twitter specifics such as hashtags are kept in their original form, unlike user mentions, which are completely removed. We also lowercase all words. Additionally, each appearance of an elongated word is shortened to a maximum of three character repetitions. Since all tweets are in relation to some topic, and the model has to determine the overall sentiment for the quantification tasks, we decided to replace words matching the tweet topic with generic tokens.

[2]http://keras.io

2.2 Pre-trained word embeddings

Each word or token that is a part of a tweet is first mapped to an appropriate distributional feature representation, also known as word embedding. Before training, we define the so called lookup table, where each word is associated with the corresponding feature representation. For the purposes of this work, we utilize the publicly available GloVe embeddings, pre-trained on the Common Crawl dataset with a dimensionality of 300. We choose these over the GloVe embeddings trained on Twitter data because of the higher dimensionality, considerably larger training corpus and vocabulary of unique words.

For words in the dataset not present in the lookup table, we use random initialization of word embeddings. However, despite their effectiveness in encoding syntactic and semantic regularities of words, they are oblivious to the words' sentiment characteristics. To counteract this, word embeddings are continuously updated during network training by back-propagating the classification errors. Therefore, sentiment regularities are being encoded in the feature representation.

2.3 Convolutional neural network

One component of our architecture is a convolutional neural network for feature extraction of Twitter messages. Dealing with variable sized text is inherently built into CNNs. Additionally, these networks, to some extent, take into account the ordering of the words and the context each word appears in. Unlike applications of CNNs in image processing, we only employ one convolutional and max pooling layer. The convolutional layer is used to extract local features around each word window, while the max pooling layer is used to extract the most important features in the feature map.

Let's consider a tweet t with length of n tokens. Because of the sliding window manner in which the filters are applied, we apply appropriate padding at the beginning and at the end of the tweet. Padding length is defined as $h/2$ where h is the window size of the filter. Before we apply the convolutional operation, each word is mapped to its corresponding word embedding. A tweet is represented as a concatenation of these word embeddings, $t = [w_1, w_2, \ldots, w_n]$, where w_i is the word embedding for the i-th word in the tweet and $w_i \in R^{300}$.

In this work, we only use a single filter with window size of 3. As tweets are limited in length, smaller window sizes are more favorable in contrast to larger ones. The network learns a filter W_c and a bias term for the filter. The convolutional operation is applied to every possible window of words and as a result a feature x_i is produced. We can formally express the operation as:

$$x_i = f(W_c \cdot t_{i:i+h-1} + b_c), \tag{1}$$

where $t_{i:i+h-1}$ is the concatenation of word vectors from position i to position $i + h - 1$, while $f(\cdot)$ is an activation function. In this work, we choose the hard rectified linear activation function. Each of the produced features are used to generate a feature map

$$x = [x_1, x_2 \ldots x_{n-h+1}]. \tag{2}$$

Then, the max-over-time pooling operation is applied over the feature map, which takes the maximum value $\hat{x} = max\{x\}$. The max pooling layer outputs a fixed sized vector with a predefined dimensionality.

2.4 Gated recurrent neural network

The accompanying part of the CNN in our deep learning architecture is a gated recurrent neural network. RNNs make use of sequential data. They perform the same task for every element in a sequence with the output being dependent on previous computations. These networks compute hidden states and each hidden state depends on its predecessor. They can also be seen as having a memory component, enabling them to look back arbitrarily in the sequence of words.

RNNs suffer from the exploding and vanishing gradient problem. There are two proposed methods for overcoming this issue: the LSTM networks (Hochreiter and Schmidhuber, 1997) and the Gated Recurrent Unit (Chung et al., 2014). We decided to use GRU because of the fewer model parameters, potentially needing less data to generalize and enabling faster training. GRU has gating units that modulate the flow of information inside the unit. The activation s_t^j of the GRU at time t is a linear interpolation between the previous activation s_{t-1}^j and the candidate activation $\hat{s}_t^j$:

$$s_t^j = (1 - z_t^j)s_{t-1}^j + z_t^j \hat{s}_t^j, \tag{3}$$

where an update gate z_t^j decides how much the unit updates its activation or content. The update gate is computed as:

$$z_t^j = \sigma(W_z x_t + U_z s_{t-1})^j. \tag{4}$$

where σ is a logistic sigmoid function. The GRU unlike LSTM has no mechanism to control the degree to which it exposes its state and exposes the whole state each time. The candidate activation is computed as:

$$\hat{s}_t^j = tanh(W x_t + U(r_t \odot s_{t-1}))^j, \tag{5}$$

where r_t is a set of reset gates and $\odot$ is an element-wise multiplication. The reset gate is computed as:

$$r_t^j = \sigma(W_r x_t + U_r s_{t-1})^j. \tag{6}$$

This network also produces a fixed vector which is necessary in our model.

2.5 Network fusion

The outputs from both networks are concatenated to form a single feature vector. This vector is then fed to a fully connected softmax layer. The softmax regression classifier gives probability distribution over the labels in the output space. The label having the highest probability is chosen as the final prediction.

	Positive	Negative	Total
Train	7374	2542	9916
Dev	1009	234	1243
Test	8202	2331	10535

Table 1: Dataset label distribution for Subtasks B and D.

	VN	N	Neu	P	VP	Total
Train	107	871	2083	3654	419	7134
Dev	28	200	520	835	191	1774
Test	138	2201	10081	7830	382	20632

Table 2: Dataset label distribution for Subtasks C and E. (N - negative, VN - very negative, Neu - neutral, VP - very positive, P - positive)

2.6 Regularization and model parameters

Due to the high number of parameters being learned, deep learning methods suffer from overfitting. To counteract this issue, we utilize dropout regularization (Srivastava et al., 2014), which randomly drops a proportion of hidden units in each iteration of network training. The dropout parameter is set to 0.25. The output size of the convolutional network and the GRU network is set to 100. The network is trained using stochastic gradient descent over shuffled mini-batches using the RMSprop (Tieleman and Hinton, 2012) update rule.

3 Experiments and results

3.1 Dataset

We train our model on the benchmark datasets provided by the SemEval challenge. However, due to deletion or changed privacy settings, we were not able to retrieve all tweets. For the 2-point classification and quantification we used the datasets from SemEval 2016 and we apply the topic preprocessing step previously mentioned. Moreover, we use positive and negative tweets from previous editions of the challenge to additionally refine our model in spite of the fact that these tweets are not labeled with the related topic.

For the 5-point classification and quantification, we only used the dataset from this year's edition of SemEval. The model is trained on the provided training and development sets while as validation set we use the provided devtest set. The testing sets are also provided by the SemEval challenge without the need to download the specific tweets. The distribution of the sentiment labels in both datasets are provided in Table 1 and Table 2.

3.2 Results

The performance our model achieves and the official ranking are provided in Table 3. The systems are ranked by the macroaveraged recall for the Subtask B where higher scores are better. On the other subtasks, systems are ranked by the error functions where lower scores are better. From the obtained results, we can see that our system notably performs best on the quantification subtasks.

The merging of the networks provides better performance over their distinctive versions for the quantification tasks. Separately, the CNN and GRNN achieve KLD scores of 0.045 and 0.035 on Subtask D respectively, while only managing 0.761 and 0.632 for the EMD score on Subtask E. On Subtask C, the model surpasses the CNN, which attains a MAE^{M} score of 0.92, but fails in comparison to the GRNN which gets 0.812. On Subtask B, both networks achieve comparable accuracy and F1 score in comparison to our proposed model, but gain better results on the recall measure, improving the performance by $\sim$ 5 points.

For Subtask B, our model performs best according to the accuracy measure, being ranked 4th. According to the average recall and F1 score, the model does not achieve notable performance although it produces significant improvement over baseline scores, especially for the AvgF1 measure. For the 5-point classification, our model again obtains average performances when compared against other teams.

Concerning Subtask D, our deep learning system produces best KLD score and also a considerable improvement over baseline scores on all three measures. Furthermore, the system gains high results on the 5-point quantification subtask as well, being

Measure	Baseline	Score	Rank
Acc	0.778	0.848	4
AvgF1	0.438	0.748	7
AvgR	0.5	0.72	10
MAE$^\mu$	0.537	0.672	6
MAE^M	1.2	0.869	5
AE	0.184	0.074	1
RAE	2.11	0.707	3
KLD	0.175	0.034	1
EMD	0.474	0.316	2
AvgRank			4.5

Table 3: Results and ranks for Subtask B, C, D and E respectively

ranked second. Our model averages a score of 4.3 on the all scores for each subtask, while averaging 4.5 on the main scores. The proposed method of our team is one of the most robust out of all other teams, as it manages second highest average rank on the considered subtasks.

4 Conclusion

In this paper, we presented a novel deep learning model for sentiment classification of Twitter messages. We proposed a fusion of CNN and GRNN for extracting features from Twitter messages and a softmax layer for generating class predictions. The deep neural network is trained on top of GloVe word embeddings pre-trained on the Common Crawl dataset. The model effectiveness is evaluated on the Sentiment Analysis in Twitter task from SemEval 2016 where our system achieved second best average rank on the 2-point and 5-point classification and quantification subtasks, testifying for its robustness.

Although our model achieved high results, there is room for improvement. For future work, we would like to pre-train word embeddings on a large set of distantly labeled tweets. Additionally, it would be interesting to see the effects of using bidirectional GRNN.

Acknowledgments

We would like to acknowledge the support of the European Commission through the project MAESTRA Learning from Massive, Incompletely annotated, and Structured Data (Grant number ICT-2013-612944).

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Cícero Nogueira dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78.

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

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Phong Le and Willem Zuidema. 2015. Compositional distributional semantics with long short term memory. *arXiv preprint arXiv:1503.02510*.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242*.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Fabrizio Sebastiani. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California, June. Association for Computational Linguistics.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Dario Stojanovski, Gjorgji Strezoski, Gjorgji Madjarov, and Ivica Dimitrovski. 2015. Twitter sentiment analysis using deep convolutional neural network. In *Hybrid Artificial Intelligent Systems*, pages 726–737. Springer.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4:2.