Paper 10 | BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs by Mathieu Cliché – published in 2017

Abstract

1. This paper attempt to produce a state-of-the-art Twitter classifier using CNNs and LSTMs networks
2. The system leverages a large amount of unlabelled data to pre-train word embeddings and using a subset of unlabelled data to fine tune the embeddings using distant supervision
3. Final CNNs and LSTMs are trained on the SemEval-2017 Twitter dataset (where the embeddings are fined tune again)
4. **To boost performances, we ensemble several CNNs and LSTMs together and this approach achieved first rank on all of the five English subtasks amongst the 40 teams**

Introduction

1. The SemEval-2017 Twitter competition is divided into five subtasks which involve standard classification, ordinal classification and distributional estimation
2. Two of the most popular deep learning techniques for sentiment analysis are CNNs and LSTMs

Models

1. CNN
    - The CNN used in this paper is similar to the CNN of Kim (2014)
    - Zero-padding such that all tweets have the same matrix dimension

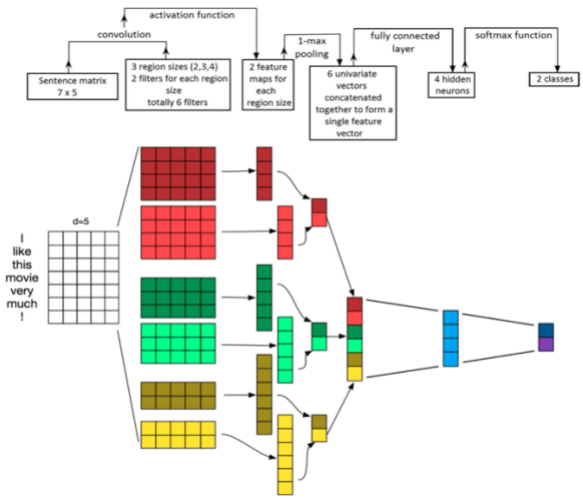

Figure 1: Architecture of a smaller version of the CNN used. Picture is taken from (Zhang and Wallace, 2015) with minor modifications.

    - In practice, the paper used three filter sizes (either [1,2,3], [3,4,5] or [5,6,7]) and we used a total of 200 filtering matrices for each filter size
    - The max-pooling layer extracts the most important feature for each convolution, independently of where in the tweet this feature is located. In other words, the CNN's structure effectively extracts the most important n-grams in the embedding space

- To reduce overfitting, the paper added a dropout layer after the max-pooling layer and after the fully-connected hidden layer (50% dropout probability)

2. LSTM

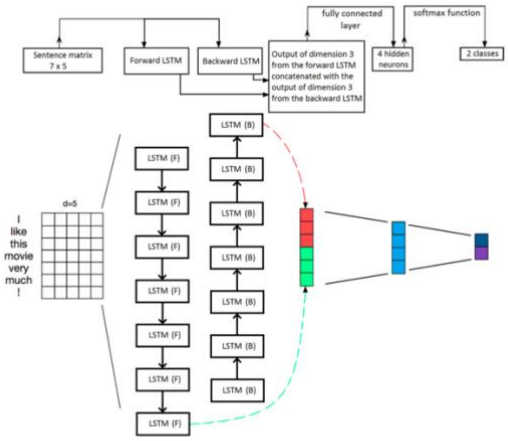

Figure 2: Architecture of a smaller version of the bi-directional LSTM used. Picture is inspired by Figure 1 of (Zhang and Wallace, 2015).

- The main building blocks are two LSTM units
- The initial hidden state is chosen to be a vector of zeros
- The simple RNN suffers from the exploding and vanishing gradient problem during the backpropagation training stage
- LSTMs solves this problem by having a more complex internal structure which allows LSTMs to remember information for either long or short terms

3. One drawback from LSTM is that it does not sufficiently consider post word information because the sentence is read only in one direction (forward). To solve this, the paper used a **bidirectional LSTM**, which is two LSTMs whose outputs are stacked together (One reads the sentence forward, the other reads it backward)
4. Again, the paper applied a dropout layer before and after LSTMs and after the fully-connected hidden layer (50%)

Training strategy

1. To train the models, the paper have access to the following data:
    - **Human-labelled tweets for different subtasks**
    - 100 million unique unlabelled English tweets (from Twitter API)
        i. Extracted a distant dataset of **5 million positive and 5 million negative** tweets (simply by looking for ":)" for positive or ":(" for negative)
        ii. The rest remains **unlabelled data**
2. **Unsupervised training** - Word embeddings
    - 100 million unlabelled tweets to pre-train the word embeddings using 3 unsupervised learning algorithms
        i. Google's Word2Vec
        ii. Facebook's FastText
        iii. Stanford's GloVe
3. **Distant training**
    - The embeddings learned in the unsupervised phase contain very little information about the sentiment polarity of the words since the context for a

positive word (e.g. good) tends to be very similar to the context of a negative word (e.g. bad)

- Therefore, to add polarity information to the embeddings, we fine tune the embeddings via a distant training phase. To do so, the authors use the CNN and initialised the embeddings with the ones learned in the unsupervised phase. We then use the distant dataset to train the CNN to classify noisy positive tweets vs noisy negative tweets

4. **Supervised training**
    - Initialise the CNN and LSTM models with the fine-tuned embeddings of the distant training phase, and freeze them for the first ~5 epochs. Then train for another ~5 epochs with unfrozen embeddings and a learning rate reduced by a factor of 10
    - To reduce variance and boost accuracy, the authors ensemble 10 CNNs and 10 LSTMs together through soft voting
    - The models ensembled have different random weight initialisations, different number of epochs (from 4 to 20), different set of filter sizes and different embedding pre-training algorithms

Results

1. To assess the performance of each model, we run the models on the historical Twitter set of 2013, 20914, 2015 and 2016 without using any of those sets in the training dataset. The historical metric is the average F_1 score of the positive and negative class
2. In 2017, the metric of interest is the macro-average recall

System	2013	2014	2015	2016
Logistic regression on 1-3 grams baseline	0.627	0.629	0.586	0.558
CNN (word2vec, convolution size=[3,4,5])	0.715	0.723	**0.688**	0.643
CNN (fasttext, convolution size=[3,4,5])	0.720	0.733	0.665	0.640
CNN (glove, convolution size=[3,4,5])	0.709	0.714	0.660	0.637
CNN (word2vec, convolution size=[1,2,3])	0.712	0.735	0.673	0.642
CNN (word2vec, convolution size=[5,6,7])	0.710	0.732	0.676	0.646
CNN (word2vec, convolution size=[3,4,5], no class weights)	0.682	0.679	0.659	0.640
CNN (word2vec, convolution size=[3,4,5], no distant training)	0.698	0.716	0.660	0.636
CNN (word2vec, convolution size=[3,4,5], no fully connected layer)	0.715	0.724	0.683	0.641
LSTM (word2vec)	0.720	0.733	0.677	0.636
LSTM (fasttext)	0.712	0.730	0.666	0.633
LSTM (glove)	0.710	0.730	0.658	0.630
LSTM (word2vec, no class weights)	0.689	0.661	0.652	0.643
LSTM (word2vec, no distant training)	0.698	0.719	0.647	0.629
LSTM (word2vec, no fully connected layer)	0.719	0.725	0.675	0.634
Ensemble model	0.725	**0.748**	0.679	**0.648**
Previous best historical scores	**0.728**	0.744	0.671	0.633

Table 1: Validation results on the historical test sets of subtask A. Bold values represent the best score for a given test set. The 2013 test set contains 3,813 tweets, the 2014 test set contains 1,853 tweets, the 2015 test set contains 2,392 tweets and the 2016 test set contains 20,632 tweets. Word2vec, fasttext and glove refer to the choice of algorithm in the unsupervised phase. No class weights means no weights were used in the cost function to counteract the imbalanced classes. No distant training means that we used the embeddings from the unsupervised phase without distant training. No fully connected layer means we removed the fully connected hidden layer from the network. Ensemble model refers to the ensemble model described in Sec. 3.4. The previous best historical scores were collected from (Nakov et al., 2016). They do not come from a single system or from a single team; they are the best previous scores obtained for each test set over the years.

- GloVe gives a lower score than both FastText and Word2Vec, therefore the authors **exclude the GloVe variation** in the ensemble model
- The absence of class weights or distant training stage lowers the scores significantly, demonstrating these are great additions

- Even though these individual models give similar scores, their outputs are sufficiently uncorrelated such that ensembling them gives the score a small boost. To assess how correlated these models are, the authors compute the **Pearson correlation coefficient between the output probabilities of any pairs of models**

System/System	System 1	System 2	System 3	System 4	System 5	System 6
System 1	1.0	0.95	0.97	0.97	0.93	0.91
System 2	0.95	1.0	0.95	0.95	0.91	0.92
System 3	0.97	0.95	1.0	0.96	0.92	0.91
System 4	0.97	0.95	0.96	1.0	0.92	0.91
System 5	0.93	0.91	0.92	0.92	1.0	0.95
System 6	0.91	0.92	0.91	0.91	0.95	1.0

Table 2: Correlation matrix for the most important models. System 1: CNN (word2vec, convolution size=[3,4,5]), System 2: CNN (fasttext, convolution size=[3,4,5]), System 3: CNN (word2vec, convolution size=[1,2,3]), System 4: CNN (word2vec, convolution size=[5,6,7]), System 5: LSTM (word2vec), System 6: LSTM (fasttext).

- The most uncorrelated models come from different supervised learning models (CNN vs LSTM) and from different unsupervised learning algorithms
- Results of the ensembled model on the 2017 test set

Subtask	Metric	Rank	BB_twtr submission	Next best submission
A	Macroaveraged recall	1/38	0.681	0.681
B	Macroaveraged recall	1/23	0.882	0.856
C	Macroaveraged mean absolute error	1/15	0.481	0.555
D	Kullback-Leibler divergence	1/15	0.036	0.048
E	Earth movers distance	1/12	0.245	0.269

Table 3: Results on the 2017 test set. The 2017 test set contains 12,379 tweets. For a description of the subtasks and metrics used, see (Rosenthal et al., 2017). For subtask A and B, higher is better, while for subtask C, D and E, lower is better.

Conclusion

1. The objective was to experiment with deep learning models along with modern training strategies in an attempt to build the best possible sentiment classifier for tweets
2. **The final model was an ensemble of 10 CNNs and 10 LSTMs with different hyper-parameters and different pre-training strategies.** This model participated in all of the English subtasks and obtained first rank in all of them
3. For future work, it would be interesting to explore systems that combine a CNN and an LSTM more organically than through an ensemble model