Sentiment Analysis in Twitter

Sourish
Undergraduate
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
sourish@iitk.ac.in

Mentor: Vincent Ng
Associate Professor
Computer Science Department
The University of Texas at Dallas
vince@hlt.utdallas.edu

***Abstract*—The work focuses on message polarity classification: Given a message, classify whether the message is of positive, negative, or neutral sentiment.The task is picked from SemEval-2016 Task 4 Subtask A. Following the work of winner teams of 2016 and 2017 , SwissCheese and DataStories respectively, we have created an ensemble of two systems one employing Convolutional Neural Networks and other bidirectional LSTMs with Attention.**

I. System Description

A. System 1: Convolutional Neural Networks

This system is made by following the work of SwissCheese [1].

1) Sentence Matrix: Each word is represented as a vector and thus a tweet as a sentence matrix.

2) Convolutional layer: A number of filters, which are learned during training, are applied to the sentence matrix to get feature vectors.

3) Max Pooling: The vector elements of feature layer are aggregated by taking the maximum over a fixed set of non-overlapping intervals.A pooled feature map is obtained.

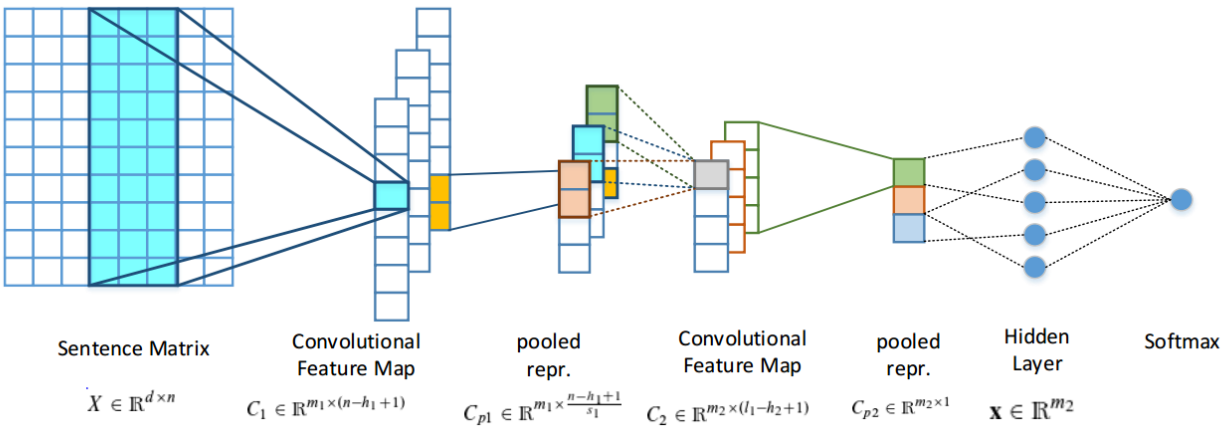

Fig. 1. System 1, Image Courtesy: SwissCheese

The sentence matrix is passed through a convolutional layer followed by max pooling then again by another convolutional layer and max pooling.This is followed by a fully connected layer and then softmax is applied which outputs the probability of each class.

B. System 2: Bidirectional LSTMs with Attention

By following the work of DataStories [2] a two layer bidirectional LSTM with attention is created. The sentence matrix is fed to biLSTM to get word annotations that summarises the information from both directions followed by another biLSTM. To find the relative contribution of each word an attention mechanism is employed which assigns weight a_i to each word annotation.

$$e_i = tanh(W_h \times h_i + b_h)$$

$$a_i = exp(e_i) / \sum_{t=1}^{T} exp(e_t), \sum_{i=1}^{T} a_i = 1$$

$$r = \sum_{i=1}^{T} a_i \times h_i, r \in R^{2 \times L}$$

W_h and b_h are weights and bias respectively learned during training.h_i as shown in image is the output of second biLSMT. T is the number of words in the tweet. L is the length of vector output by each direction of LSTM. Together forward and backward LSTMs output a vector h_i of length $2L$. x_i represent the ith word in tweet.

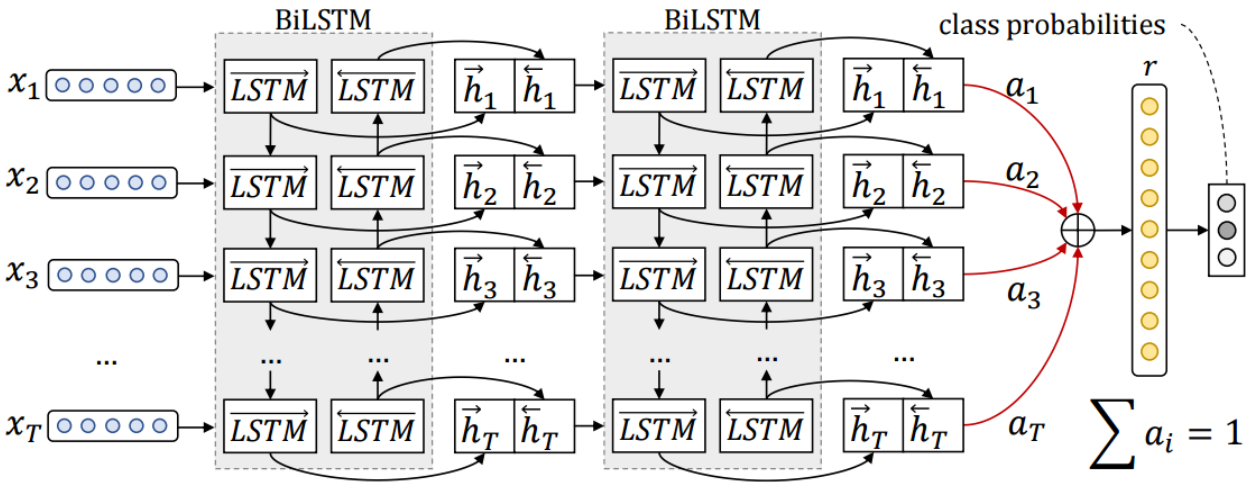

Fig. 2. System 2, Image Courtesy: DataStories

C. Ensemble

Using Scikit-learn [3] for the random forest, the two systems are ensembled by concatinating the values of nodes of fully connected layer of System 1 and vector r of System 2 and feeding it to a Random Forest Classifier, trained using training data, to predict the class.

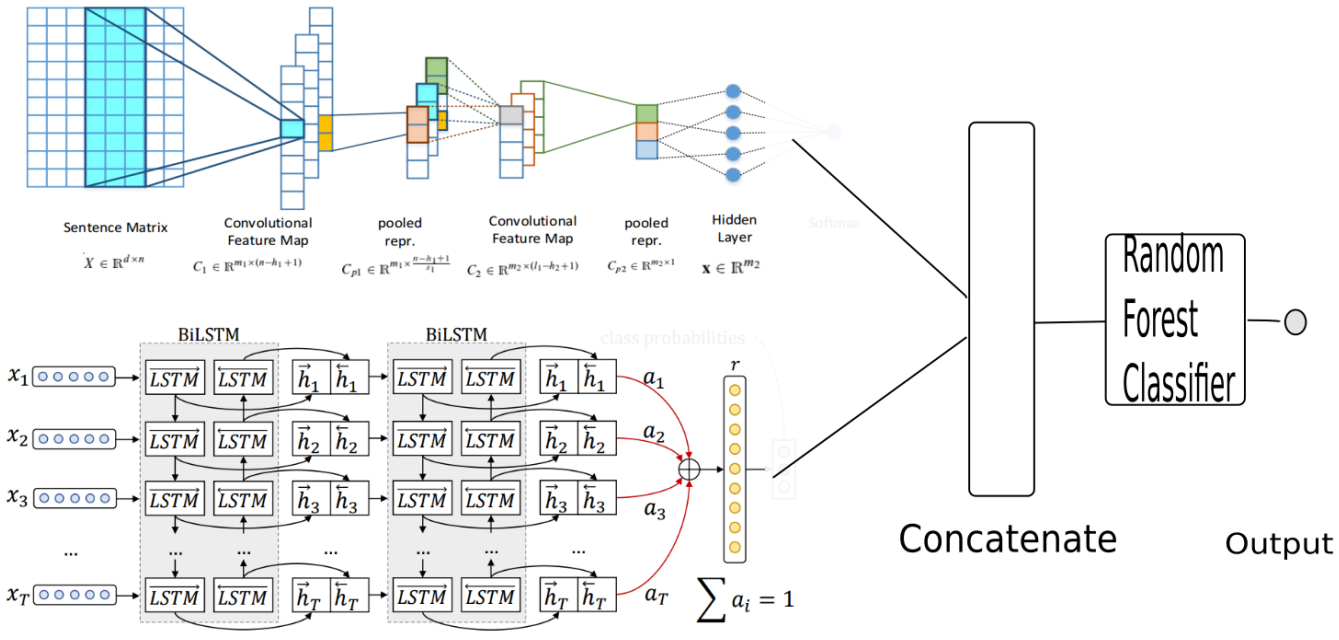

Fig. 3. Ensemble

II. Experimental Setup

The systems are coded using Python libraries TensorFlow and TFLearn.Numpy, Scikit-learn and NLTK are also used.

A. Feature Vectors

Tweets are tokenized using NLTK word tokenizer. We use GloVe [4] pre-trained word vectors on tweeter data,say of dimension d. Let n be maximum number of words found in the word vectors for a tweet in training data. Given a tweet, each word is looked up in GloVe word vectors and if found, is added to the sentence matrix.Rest of the matrix is padded by zero vectors.Thus a sentence matrix of dimension $n \times d$ is obtained.

B. Training the Systems

Each above mentioned system is trained using SemEval 2016 training data of 6000 tweets.The $n \times d$ sentence matrix obtained above is fed to each of these systems for training.

C. Training the Random Forest

The values of nodes of fully connected layer of System 1 is concatenated with the r vector of System 2 for each training tweet and is fed to a Random Forest Trainer as feature vector.

D. Testing

For each test tweet, r vector of system 2 and last layer of fully connected layer of system 1 are concatenated and fed to the trained Random Forest Classifier to get the predicted class.

III. Results

Above system gets an F1 score of 0.54 on SemEval 2016 test data.The code is available on github. [1]

IV. Experiments

Combination	F1 score
CNN(GloVe) + CNN(word2vec)	0.52
CNN(GloVe) + CNN(word2vec) + biLSTM(GloVe)	0.53
CNN(GloVe) + biLSTM(GloVe)	0.54

V. Further Improvements

1) Distant Supervised Training: The embeddings can be made to catch sentiment by training the system on a large corpus of tweets using distant supervised training.

2) Pre Processing: One can use DataStories tool Ekphrasis for better pre-processing of tweets.

[1] https://github.com/sourishk/SentimentAnalysis

References

[1] Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1124–1128, San Diego, California, June 2016. Association for Computational Linguistics.

[2] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

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