# EMOMINER at SemEval-2019 Task 3: A Stacked BiLSTM Architecture for Contextual Emotion Detection in Text

**Nikhil Chakravartula**
Teradata / Hyderabad
nikhil.chakravartula@gmail.com

**Vijayasaradhi Indurthi**
Teradata / Hyderabad
vijayasaradhi.indurthi@teradata.com

## Abstract

This paper describes our participation in the SemEval 2019 Task 3 - Contextual Emotion Detection in Text. This task aims to identify emotions, viz. happiness, anger, sadness in the context of a text conversation. Our system is a stacked Bidirectional LSTM, equipped with attention on top of word embeddings pre-trained on a large collection of Twitter data. In this paper, apart from describing our official submission, we elucidate how different deep learning models respond to this task.

## 1 Introduction

Sentiment analysis is an established field in NLP, but just identifying positive and negative sentiments may not be enough. Applications require systems to go further beyond sentiment analysis and perform emotion analysis, which deals with identifying discrete emotions like anger, joy, sadness, etc. The task is challenging because the importance of context in emotion analysis cannot be overstated (Malik et al., 2017; Vanzo et al., 2014). Also, text in a conversation often contains language slangs, emoticons, emojis and other noisy data that make it difficult to identify the type of feeling expressed.

Many approaches have been put forward to identify emotions in a text. Purver and Battersby (2012); Balabantaray et al. (2012) used SVM classifier on twitter data to carry out emotion analysis. Potapova and Gordeev (2016) deployed a model based on Random Forests to identify aggression in texts. These approaches are often assisted by lexicons and require heavy feature engineering.

In recent years, deep learning approaches have outperformed traditional algorithms in NLP tasks (Bahdanau et al., 2014). Felbo et al. (2017); Gupta et al. (2017) utilized LSTMs to achieve emotion identification in text. Abdul-Mageed and Ungar (2017) showed that GRNNs achieved a very good performance on 24 fine-grained types of emotions. Kratzwald et al. (2018) proposed sent2affect, a tailored form of transfer learning for affective computing, where the network is pre-trained for sentiment analysis task, and subsequently the output layer is tuned to the task of emotion recognition. In this paper, we propose a stacked Bidirectional LSTM architecture to recognize emotions in text.

| happy | sad | angry | others |
|---|---|---|---|
| 14.06% | 18.11% | 18.25% | 49.56% |

Table 1: Train dataset composition.

The rest of the paper is organized as follows. In sec. 2 we give a brief explanation of the shared task. In sec. 3 we describe the process of feature engineering from the text. In sec. 4 we discuss system architecture. It is followed by sec. 5 which contains the various settings used in our experiments. In sec. 6, we analyse the results and conclude our paper in sec. 7 with future ideas and vision.

## 2 Shared Task Description

The SemEval 2019 Task 3 is as follows: Given three turns of a conversation by two users, say turn1 by user1, turn2 by user2 and turn3 by user1, the system must identify the emotion of turn3 based on the conversation. It has to be one of the four values - happy, sad, angry and others. The dataset is provided by the organizers of the task. The composition of the dataset is described in Table 1. More details about the task can be found in the task description paper (Chatterjee et al., 2019).

*Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*, pages 205–209
Minneapolis, Minnesota, USA, June 6–7, 2019. ©2019 Association for Computational Linguistics

# 3 Feature Engineering

## 3.1 Pre Processing

We perform the following pre-processing operations on the text before feature engineering.

- All text is converted to lower case.
- All contractions are replaced with their full form. For example, *don't* will be replaced by *do not* and *can't* will be replaced by *can not*.
- All punctuation marks are removed.
- **Spell correction and Emoji expansion:** Many conversations include words in an elongated form (*nooooo, youuuuu, heyyyyyy etc.,*) and slangs(*wassup, 4u, lolz etc.,*). We perform spell corrections (Jurafsky and Martin, 2018) on these words to reduce the vocabulary size and to account for better results. Text8[1] is utilized to generate unigram and bigram word statistics with ekphrasis (Baziotis et al., 2017) to perform spell correction.

  Emojis play a crucial part in identifying the emotion of a conversation. A conversation often contains a high number of emojis that intrinsically determines its nature. Identifying this quintessential importance, we use a python package named emoji[2] to expand the emojis into representative keywords. Eg: 'unamused face'
- **Parts Of Speech**: Part-of-speech (POS) tagging is an important and fundamental step in Natural Language Processing. The Part-of-speech gives a large amount of information about a word and its neighbours, syntactic categories of words and similarities and dissimilarities between them. NLTK (Steven Bird and Loper, 2009) is used to extract the Parts Of Speech tags for each word in the conversation, and then concatenated them with GloVe vectors. As a result, Glove vectors will have syntactic information of words (Rezaeinia et al., 2017).

## 3.2 Feature Extraction

- **Word Embeddings:** Glove840B - common crawl (Pennington et al., 2014) pre-trained word embeddings are used to convert each of the words in the conversation to a 300-dimensional feature vector.
- **One Hot Encoding:** The POS tags generated in the previous step are converted to a constant vector using One-Hot Encoding
- **Lexicon:** We exploit the DepecheMood affective lexicon Deepechemood++ (Araque et al., 2018) that has been built in a completely unsupervised fashion, from affective scores assigned by readers to news articles. DepecheMood++ allows for both high-coverage and high-precision, providing scores for 187k entries on the following affective dimensions: Afraid, Happy, Angry, Sad, Inspired, Don't Care, Inspired, Amused, Annoyed.

# 4 System Architecture

## 4.1 Embedding Layer1 (EL1)

This embedding layer takes as input a fixed sequence of 200 words and converts each word into its corresponding 300 dimensional glove word vector (Pennington et al., 2014).

## 4.2 Embedding Layer2 (EL2)

This embedding layer takes as input a fixed sequence of 200 Parts Of Speech tags and converts each of them into a constant one-hot vector.

## 4.3 Embedding Layer3 (EL3)

This embedding layer takes as input a fixed sequence of 200 words and converts each of them into a vector based on the values in DepecheMood affective lexicon.

## 4.4 BiLSTM

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) architecture that has been designed to address the vanishing and exploding gradient problems of conventional RNNs. Unlike feed-forward neural networks, RNNs have cyclic connections making them powerful for modelling sequences. They have been successfully used for sequence labelling and sequence prediction tasks (Sak et al., 2014). An LSTM has 3 types of gates, the forget gate, the input gate and the output gate. The information flow is governed by the following equations.

[1] http://mattmahoney.net/dc/textdata.html

[2] https://github.com/carpedm20/emoji/

| S.No | Setting | F1$\mu_{\text{avg}}$ |
|---|---|---|
| 1 | EL1 + LSTM(256) + dropout(0.3) | 0.8891 |
| 2 | EL1 + BiLSTM(256) + dropout(0.2) | 0.8903 |
| 3 | EL1 + LSTM(256) + dropout(0.3) + Attention | 0.8950 |
| 4 | EL1 + BiLSTM(256) + dropout(0.3) + Attention | 0.8918 |
| 5 | EL1 + BiLSTM(128) + dropout(0.2) + BILSTM(128) + dropout(0.3) | 0.8951 |
| 6 | EL1 + BiLSTM(128) + dropout(0.2) + BILSTM(128) + dropout(0.3) + Attention | **0.8956** |
| 7 | EL1 + EL2 + BiLSTM(128) + dropout(0.2) + BiLSTM(128) + dropout(0.3) | 0.8965 |
| 8 | EL1 + EL3 + BiLSTM(128) + dropout(0.2) + BiLSTM(128) + dropout(0.3) | 0.8969 |
| 9 | EL1 + EL2 + EL3 + BiLSTM(128) + dropout(0.2) + BiLSTM(128) + dropout(0.3) | 0.8931 |

Table 2: Results of different settings. S.No 1-6 are the variations of the system that are evaluated for the competition. S.No 7-9 provide further analysis of the system after the competition ended. All results shown are obtained with five fold cross validation on the train set.

$$f_{\text{t}} = \sigma(W_{\text{f}} \cdot [h_{\text{t-1}}, x_{\text{t}}] + b_{\text{f}}) \tag{1}$$
$$i_{\text{t}} = \sigma(W_{\text{i}} \cdot [h_{\text{t-1}}, x_{\text{t}}] + b_{\text{i}}) \tag{2}$$
$$\tilde{C}_{\text{t}} = \tanh(W_{\text{C}} \cdot [h_{\text{t-1}}, x_{\text{t}}] + b_{\text{C}}) \tag{3}$$
$$C_{\text{t}} = f_{\text{t}} \times C_{\text{t-1}} + i_{\text{t}} * \tilde{C}_{\text{t}} \tag{4}$$
$$o_{\text{t}} = \sigma(W_{\text{o}} \cdot [h_{\text{t-1}}, x_{\text{t}}] + b_{\text{o}}) \tag{5}$$
$$h_{\text{t}} = o_{\text{t}} \times \tanh C_{\text{t}} \tag{6}$$

Where:

- $\text{W}_\text{i}$, $\text{W}_\text{f}$, $\text{W}_\text{o}$, $\text{W}_\text{c}$ : are the trained weights.
- $\text{b}_\text{i}$, $\text{b}_\text{f}$, $\text{b}_\text{o}$, $\text{b}_\text{c}$ : are the trained biases
- $\sigma$: is the sigmoid function.
- $\text{x}_\text{t}$ : is the input at time step t
- $\text{c}_\text{t}$ : is the cell state at time t
- $\text{h}_\text{t}$ : is the output at time step t

Single directional LSTM can only use the contextual information from the past. Bidirectional LSTM can use the contexts of the past as well as the future, generating two independent sequences of LSTM output vectors (Schuster and Paliwal, 1997). The output at each time step is the concatenation of two output vectors from both the directions, i.e.,

$$h_{\text{t}} = \overrightarrow{h_{\text{t}}} \oplus \overleftarrow{h_{\text{t}}}$$

### 4.5 Dropout

Dropout is a regularization technique in which units and their connections are randomly dropped from the neural network during training (Srivastava et al., 2014). This prevents units from co-adapting too much. Dropout of *p* sets *p* fraction of units to 0 at each update during training time. We employ dropout in our system to avoid overfitting.

### 4.6 Attention

Not all words in a sentence contribute to a sentiment. A neural network armed with an attention mechanism can actually understand how to disregard the noise and focus on what's relevant. This is especially effective in sequence tasks as the network can choose to remember only that context that's relevant (Zhang et al., 2018).

## 5 Experiments

### 5.1 Evaluation Metrics

Evaluation will be done by calculating micro averaged F1 score($F1\mu$) for the three emotion classes i.e. *Happy*, *Sad* and *Angry*. The *Others* class is ignored in the evaluation.

- $P\mu = (\sum TPi) \div (\sum (TPi + FPi))$ $\forall i \in Happy, Sad, Angry$
- $R\mu = (\sum TPi) \div (\sum (TPi + FNi))$ $\forall i \in Happy, Sad, Angry$
- $F1\mu = (2 \times P\mu \times R\mu) \div (P\mu + R\mu)$

TPi is the number of samples of class i which are correctly predicted, FNi and FPi are the counts of Type-I and Type-II errors respectively for the samples of class i. Please note that both the precision and recall are micro-averaged.

### 5.2 Methodology

All the experiments are developed using the Scikit-Learn (Pedregosa et al., 2011) machine learning library and keras deep learning library (Chollet et al., 2015) with Tensorflow backend (Abadi et al., 2015). We concatenate turn1, turn2 and turn3 using a separator 'eos', read as 'end of sentence'. Similarly, we concatenate the POS tags of all the turns as well using the same separator. Five-fold cross validation is used to evaluate our models. In all our experiments, the batch size is 200, the learning rate is 0.008 and the number of epochs is 10. The loss is categorical cross-entropy and the optimizer used is rmsprop. In all the settings, the activation for LSTM/BiLSTM is tanh and the last layer is a dense layer of 4 units with sigmoid activation. Attention layer, if employed, is used after the last LSTM/BiLSTM layer. All our code is publicly available in a Github repository.[3]

Table 2 shows the results of different variations of the system.

## 6 Results and Analysis

The results show that attention based models outperform their corresponding equivalents. It is interesting to see from S.No 1-4 that BiLSTM outperforms LSTM when no attention is used but in the presence of attention, LSTM performs better than BiLSTM. The two layer BiLSTM with attention in S.No 6 surmounted all the other variations during the competition. Hence, we submitted this model and achieved an F1$\mu$ score of 0.6939.

S.No 7-9 show our further analysis of the system when different embedding layers are merged. We see that concatenating any one of POS and DepecheMood to the word vectors increased the performance in S.No 7-8, but not by much. However, concatenation of word vectors, POS and DepecheMood decreased performance, as shown in S.No 9.

## 7 Conclusion and Future Work

In this paper, we described a stacked BiLSTM deep learning model to detect emotion in context. We used glove pre-trained embeddings to convert each word into its corresponding word vector and then passed it on to two layers of BiLSTM, applied attention mechanism and finally, passed the intermediate inputs on to a dense layer of 4 units with sigmoid activations. We also depicted the results of adding different features to the pre-trained word vectors. Inspired by the work of Rezaeinia et al. (2017), in the future, we would like to examine more lexicon combinations to analyze the performance of the system. We would also like to make the system deeper to scrutinize how it responds.

[3] `https://git.io/fhFG4`

## References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Muhammad Abdul-Mageed and Lyle Ungar. 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 718–728. Association for Computational Linguistics.

Oscar Araque, Lorenzo Gatti, Jacopo Staiano, and Marco Guerini. 2018. Depechemood++: a bilingual emotion lexicon built through simple yet powerful techniques. *CoRR*, abs/1810.03660.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

R C Balabantaray, Iiit Bhubaneswar, Mudasir Mohammad, and Nibha Sharma. 2012. N.: Multi-class twitter emotion classification: A new approach. *International Journal of Applied Information Systems*, pages 48–53.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. 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. Association for Computational Linguistics.

Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. 2019. Semeval-2019 task 3: Emocontext: Contextual emotion detection in text. In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval-2019)*, Minneapolis, Minnesota.

François Chollet et al. 2015. Keras. `https://keras.io`.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625. Association for Computational Linguistics.

Umang Gupta, Ankush Chatterjee, Radhakrishnan Srikanth, and Puneet Agrawal. 2017. A sentiment-and-semantics-based approach for emotion detection in textual conversations. *CoRR*, abs/1707.06996.

Daniel Jurafsky and James H. Martin. 2018. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall.

Bernhard Kratzwald, Suzana Ilic, Mathias Kraus, Stefan Feuerriegel, and Helmut Prendinger. 2018. Decision support with text-based emotion recognition: Deep learning for affective computing. *CoRR*, abs/1803.06397.

Mubasher H. Malik, Syed Ali Raza, and H.M. Shehzad Asif. 2017. Context based emotion analyzer for interactive agent. *International Journal of Advanced Computer Science and Applications*, 8(1).

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. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

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.

Rodmonga Potapova and Denis Gordeev. 2016. Detecting state of aggression in sentences using CNN. *CoRR*, abs/1604.06650.

Matthew Purver and Stuart Adam Battersby. 2012. Experimenting with distant supervision for emotion classification. In *EACL 2012*.

Seyed Mahdi Rezaeinia, Ali Ghodsi, and Rouhollah Rahmani. 2017. Improving the accuracy of pre-trained word embeddings for sentiment analysis. *CoRR*, abs/1711.08609.

Hasim Sak, Andrew W. Senior, and Françoise Beaufays. 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *CoRR*, abs/1402.1128.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45:2673–2681.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ewan Klein Steven Bird and Edward Loper. 2009. *Natural Language Processing with Python– Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc.

Andrea Vanzo, Danilo Croce, and Roberto Basili. 2014. A context-based model for sentiment analysis in twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2345–2354. Dublin City University and Association for Computational Linguistics.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis : A survey. *CoRR*, abs/1801.07883.