The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

An Ensemble Neural Network for the Emotional Classification of Text

Oscar Youngquist
Department of Computer Science
and Software Engineering
Rose-Hulman Institute of Technology
youngqom@rose-hulman.edu

Abstract

In this work, we propose a novel ensemble neural network design that is capable of classifying the emotional context of short sentences. Our model consists of three distinct branches, each of which is composed of a combination of recurrent, convolutional, and pooling layers to capture the emotional context of text. Our unique combination of convolutional and recurrent layers enables our network to extract more emotionally salient information from text than formerly possible. Using this network, experiments classifying the emotional context of short sequences of texts from five distinct datasets, were conducted. Results show that the novel method outperforms all historical approaches across all datasets by 8.31 percentage points on average. Additionally, the proposed work produces results that are on average as accurate as state of the art methods, while using two orders of magnitude less training data. The contribution of this paper is a novel ensemble recurrent convolutional neural network capable of detecting and classifying the emotional context of short sentences.

Introduction and Related Work

We introduce, implement, and evaluate a novel ensemble recurrent convolutional neural network for the purpose of classifying the emotional context of text. The model uses convolutional features to seed recurrent layers with a prior knowledge of the text's most significant time invariant features in order to better identify the latent emotional semantic context of the text.

The emotional classification of text is a sub-topic of text classification and therefore, has been solved historically by using traditional machine learning techniques and feature engineering (Strapparava and Mihalcea 2008; Creed and Beale 2008; Li and Xu 2014). However, recently proposed models have also seen significant advances in performance using artificial neural networks (Poria et al. 2016b; 2016a). The work done by Muhammad Abdul-Mageed and Lyle Ungar in their 2017 paper "EmoNet" represents the current state of the art work in this area and is based on a gated recurrent neural network (GRNN) (Abdul-Mageed and Ungar 2017).

GRNN's and recurrent neural networks (RNN) in general have proven to be excellent at text classification tasks (Pal, Ghosh, and Nag 2018; Zolkepli 2018; Zhang, Wang, and Liu 2018). Additionally, while convolutional neural networks (CNN) have largely been reserved for image recognition/classification tasks, several successful text classification architectures based on CNN have been proposed (Kim 2014; Severyn and Moschitti 2015; Jaderberg, Vedaldi, and Zisserman 2014). Most recent work in this field attempts to find combinations of CNN's and RNN's that take advantage of the strengths of both network architectures (Sosa 2017; Bouazizi and Ohtsuki 2017; Chen et al. 2017; Yin et al. 2017; Lai et al. 2015).

Network Architecture

One of the key aspects of this architecture is its three distinct parallel branches. The three branches process complimentary inputs, namely the left-shifted text, the right-shifted text and the current text, in order to provide context for emotional classification and was inspired by the work of (Lai et al. 2015). Furthermore, the flow of information through the network can be broken down into four main stages: inputs, word embedding, left and right context convolutional feature extraction, and recurrent/merge layers. Figure 1 illustrates the network architecture of our system.

Inputs

Each of the inputs described in the following passage is a vector of plain text words that have been parsed to remove all non-alphanumeric characters. The input for the first branch of the model - which can be seen as the top branch in the Figure 1 -– is the left-shifted context of the input word vector. Left-shifted means that the input vector to this branch is the same as the unmodified text input vector, but with each word in the sentence being shifted one position to the left: so the second word in the sentence becomes the first, the third the second, and so on. The input to the middle branch is simply the unmodified text input vector. The input to the bottom branch is the right-shifted context vector. The right-shifted input is created the same way as the left-shifted input, but with each word being shifted one position to the right instead of the left.

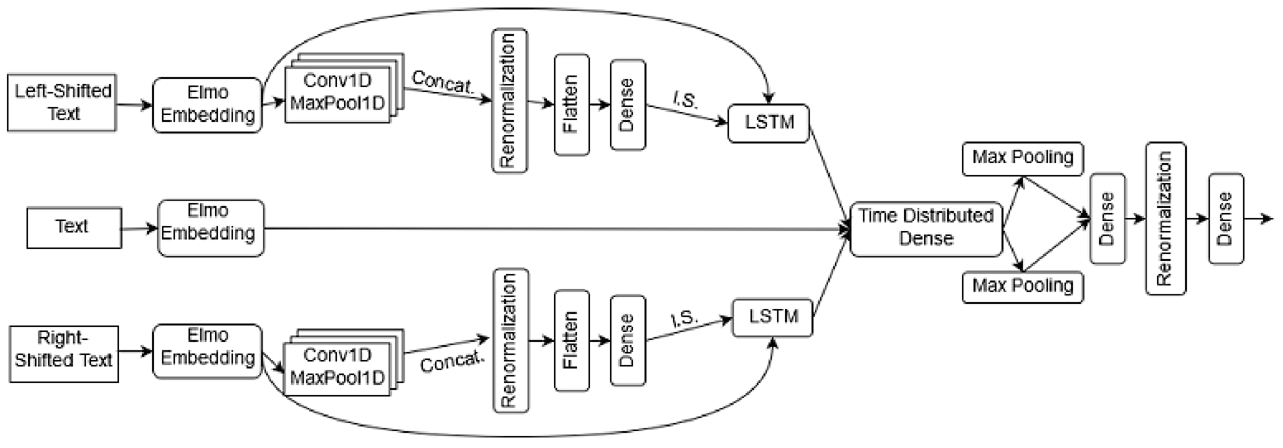

Figure 1: The architecture of our network. The arrows represent the flow of data from one layer to another. Concat. is short for concatenate and I.S. are the initials for Initial State. The convolutional layers have filter sizes of 2, 3, 4, 5, 10.

Both of the shifted inputs are kept the same length as the unmodified input by placing a ''__PAD__' token in the last and first position of the left and right shifted arrays, respectively. By splitting the input in this way, and then generating contextualized representations of the left and right-shifted text via recurrent layers, we are in effect able to take into consideration the full left and right context for each word in the sentence that is being processed. This allows us to take advantage of the spatial-temporal relationships of the semantics of the input text forwards (right context), backwards (left context), and centered (non-shifted text) in time.

Splitting the input text in this manner enables the network to learn the more distributed and complex aspects of the semantics of the written word and was inspired by the work of (Lai et al. 2015). The intent of this design is to use the three inputs, recurrent layers, and time distributed dense layer to replicate the functionality of a sliding-window operation over the text. However, unlike the sliding-window operations found in convolutional layers, this arrangement allows the full context of the utterance to the left and right of any given word to be taken into consideration. This is because the data being used in the sliding-window operation in this model consists of the word embeddings of the non-shifted text input and the context vectors (as generated by the left and right-shifted branches respective LSTM layers) of the words that precede and follow each word in the input. By replicating a sliding-window operation in this way, using this data, the network is able to better capture the "long-distance" contextual relationships of an utterance than a traditional convolutional layer which only operates on partial information from the text (Lai et al. 2015).

Word Embedding

Each branch begins by taking the text as input and producing a contextualized word-vector representation of the input using an ELMo embedding layer; an embedding layer proposed by (Peters et al. 2018). The ELMo layers calculate word vectors through a pre-trained deep bidirectional language model which aims to capture both the complex characteristics of word use and how these characteristics vary across linguistic contexts. This embedding strategy was selected in order to further facilitate the identification and incorporation of the emotional semantic context of the input in the classification process.

Left and Right Context Convolutional Feature Extraction

The word embeddings for the left and right context vectors are fed to five parallel convolutional layers as can be seen in the top and bottom branches in Figure 1. Each of these layers has a different window size: 2, 3, 4, 5, and 10 but are otherwise identical with 128 filters a piece and using the RELU activation function. Additionally, each convolutional layer is followed by a max pooling layer with a window size of two. The five convolutional layer's are separately concatenated together as can also be seen in the top and bottom branches in Figure 1. The purpose of these convolutional layers is the extraction and composition of temporally invariant features that might otherwise be missed or underemphasized by a recurrent network alone and was inspired by the work of (Chen et al. 2017).

While the convolutional features are not themselves fed to the subsequent recurrent layers of this model, they are used to initialize the state or "prior knowledge" of those layers. This is done by first normalizing the concatenated output of the pooling layers via batch renormalization, a technique proposed by (Ioffe 2017), and then flattening the result and sending it through a fully connected layer. The fully connected layer uses the softmax activation function and produces a vector which is used to initialize the recurrent layers state as in the work (Chen et al. 2017). Using this vector to initialize the recurrent state provides the subsequent recurrent layers with a prior knowledge of the most significant temporally-invariant features from the text; allowing the recurrent layers to develop a more comprehensive representation of the text.

Recurrent/Merge Layers

The left and right-shifted text word embeddings are each fed to a bidirectional LSTM layer which has been initialized using their respective left and right convolutional features, as described above. The output of the LSTM layers are concatenated together along with the non-shifted embedded word vector. The concatenated vector is then fed to a time distributed fully connected layer which uses the tanh activation function and has 250 outputs. This output is then processed by two parallel, Siamese style max pooling layers. The output of each pooling layer is then concatenated together and sent through a fully-connected layer, the output of which is sent through a batch renormalization layer. While this particular arrangement of pooling layers might seem counter-intuitive, it was found experimentally that this configuration consistently improved the overall accuracy of the model. Lastly, the renormalized output is fed to the softmax activated fully connected output layer. The design of these last few layers were inspired by the work of (Lai et al. 2015). However, by initializing the state of the recurrent layers with context independent features from convolutional layers, as in the work of (Chen et al. 2017), the recurrent layers in the RCNN architecture can better identify and emphasize the most emotionally salient features.

Experimental Setup

We now detail the experimental design used to evaluate the performance of our network.

Datasets

The validation experiments performed in this work are conducted using the following five datasets:

- **Twitter Emotion Corpus (TEC)**: published by Saif Mohammad, is a dataset containing 21,051 tweets; each labeled by a single emotion (2012b). The emotion labels present in this set are the six basic Ekman emotions: anger, disgust, fear, happiness, sadness, and surprise. Labeling the tweets was accomplished through the application of distant supervision. Tweets which contained a single emotional hashtag, such as #happy, at the end of the tweet were labeled with the emotion that is conveyed by the hashtag. The hashtag was then removed from the tweet before it is used for training or testing purposes. This distant supervision labeling technique does have a potential drawback in the presence of sarcasm in tweets. Labeling tweets through the use of a single hashtag may make the labeling process prone to the introduction of sarcasm into the dataset. Despite this possibility, curating emotional text data in this way has become common practice in works such as (Mohammad 2012a; Mintz et al. 2009; Wang et al. 2012; Abdul-Mageed and Ungar 2017; González-Ibánez, Muresan, and Wacholder 2011). Therefore, this concern will not be explicitly addressed in this work.
- **CrowdFlower (CF)**: this is the dataset from "Sentiment Analysis: Emotion in Text" published by CrowdFlower and used in Microsoft's Cortana Intelligence Gallery (CrowdFlower 2016b; 2016a). This dataset consists of 40,000 tweets labeled with one out of a possible 13 emotions. The dataset was also labeled using the semi-supervised approach described for the TEC dataset.
- **DailyDialogs (DD)**: contains 13,118 conversations, split into 102,980 sentences each hand labeled via expert annotation (Li et al. 2017). Each sentence has a single label from one of seven emotional categories: anger, fear, disgust, happiness, sadness, surprise, and noemo (no emotion).
- **Aggregate Dataset (AD)**: presented by Laura Bostan and Roman Kilnger, is a dataset which is comprised of an aggregation of other emotion classification in text datasets; including the three listed above. This dataset contains 202,062 sentences each labeled with one of 12 emotion labels. The other datasets used in the making of this set, as well as the procedures used to combine them, can be found in (Bostan and Klinger 2018).
- **Zolkepli Emotion Data (ZM)**: is a dataset that was curated by Husein Zolkepli for his Emotion Classification Comparison project (Zolkepli 2018). This dataset contains tweets that have been parsed to remove all non-alphanumeric characters, and labeled as one of six basic Ekman emotions: joy, sadness, love, anger, fear, and surprise. Each emotion label has the same number of tweet examples, resulting in a perfectly balanced dataset. The tweets where labeled using the same semi-supervised approach used for both the TEC and CF datasets.

Dataset	Total Size	Train / Val / Test
ZM	69120	46656 / 8640 / 13824
DD	22407	14720 / 3200 / 4480
TEC	21051	10816 / 3200 / 7008
AG	107724	64608 / 21504 / 21504
CF	40000	28000 / 8000 / 8000

Table 1: A summary of the datasets including the total size of the set and the training/validation/testing split.

Additional preprocessing was applied to the Aggregate and Daily Dialog datasets. The DD dataset was found to be extremely unbalanced; with "noemo" dominating the other emotional categories. Through data exploration, the Daily Dialog dataset was found to have 85,572 sentences labeled noemo, with the next largest emotional category - joy - containing only 12,885 records. To create a more balanced dataset, so as to counter the negative training effects inherent to heavily biased data, 5,000 noemo labeled sentences were randomly selected from the 85,572 present to be used during training and evaluation. The other noemo labeled sentences where discarded. This resulted in the Daily Dialogs dataset being reduced to a size of 22,407 total records. Furthermore, the same noemo bias was found in the Aggregate dataset which was found to contain 104,338 sentences labeled noemo, and only 37,237 examples for the next largest category: joy. Similar to the DD dataset, 10,000 random noemo labeled sentences were selected to be used in training and evaluation, reducing the size of the Aggregate dataset to

Model	CF		AG		TEC		DD		ZM	
	F1	A	F1	A	F1	A	F1	A	F1	A
Emotion RCNN-EN	**0.380**	**0.384**	**0.520**	**0.523**	**0.600**	**0.603**	**0.727**	**0.724**	**0.950**	**0.951**
(Lai et al.) RCNN	0.360	0.357	0.450	0.451	0.570	0.574	0.675	0.674	0.940	0.939
(Kim) CNN	0.340	0.340	0.450	0.448	0.537	0.538	0.668	0.668	0.910	0.914
Bidirectional LSTM	0.210	0.209	0.340	0.342	0.320	0.321	0.575	0.577	0.910	0.912
BoW + Naïve Bayes	0.300	0.302	0.470	0.468	0.520	0.524	0.658	0.656	0.850	0.859
BoW + SVM	0.340	0.341	0.510	0.513	0.570	0.569	0.690	0.689	0.890	0.899
TFIDF + Naïve Bayes	0.280	0.280	0.380	0.383	0.450	0.452	0.595	0.600	0.680	0.735
TFIDF + SVM	0.350	0.347	0.460	0.456	0.553	0.552	0.670	0.673	0.830	0.851

Table 2: Summary of micro F1 score (F1) and accuracy (A) on the all the test sets for all evaluated models. All scores where calculated using Scikit-learn (Pedregosa et al. 2011).

107,724 records. The total size and training, validation, and testing split for each dataset can be seen in Table 1.

Experimental Settings

Before the experiments were conducted the datasets were processed as follows. Each set was split into training, validation, and testing sets and then pre and post padding was applied to ensure a uniform length for all the input vectors. The length of each input vector was dependent upon the average sentence length in each of the datasets being processed. The TEC, AG, and CF datasets were each padded to a length of 100 words, DD to 80, and ZM was padded to 60 words. Categorical crossentropy was used as the network's loss function, and stochastic gradient descent with clipping was used as the model's training optimizer. Recurrent layers employ L2 regularization and dropout to prevent overfitting. For the learning rate, the training technique: Stochastic Gradient Descent with Restarts (SGDR), as proposed by (Loshchilov and Hutter 2016), was used. A maximum learning rate of 0.2 was selected with a minimum of 0.001, a cycle length of 10 epochs, and a decay rate of 35% per cycle.

Furthermore, due to the small size of the DD and TEC datasets, cross-validation was employed during training and evaluation. 3-Fold cross-validation was utilized for the TEC dataset and 5-Fold cross-validation for the DD dataset.

Evaluation Metrics

The following metrics were used for evaluating the performance of the tested networks: Categorical Accuracy, and micro F1 score (F1).

Comparison of Methods

The following is a list of the models we used to evaluate our network architecture. Each model in the list below was implemented with Tensorflow, Keras, Scikit-learn and using standard machine learning practices or as detailed in the associated paper.

- **Baselines**: The baseline models are a Naïve Bayes Bayes (NB) classifier and a Support-Vector Machine (SVM) classifier. For the NB and SVM models, two word vectorization techniques were used during evaluation: Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TFIDF).
- **Bidirectional LSTM**: A simple bidirectional LSTM network.
- **Yoon Kim CNN (2014)**: Yoon Kim's seminal work demonstrating the effectiveness of CNN for text classification tasks.
- **Lai et al. RCNN (2015)**: The recurrent convolutional neural network that this work is heavily based upon.
- **Emotion RCNN-EN**: Our Network.

Experimental Results

Our network architecture outperforms every model tested against it, see Table 2, including the state of the art RCNN, achieving results that are on average 8.31 percentage points (pps) more accurate than any other evaluated model across all datasets. Furthermore, our model outperforms the other evaluated models in terms of F1 score across all datasets.

Baseline Models

When comparing the results of our architecture to the baseline models, it is clear that our network far exceeds the accuracy performance of the baselines. On average our model is 7.975 pps more accurate than the baselines across all the datasets. Furthermore, our network also performs higher in terms of F1 score across all datasets. However, as can be seen in the results for the TEC, AG, and DD datasets, the baseline methods are among the top performing models. With respect to the DD and TEC datasets, it is likely that the SVM model, with BoW text embeddings, outperformed it's deep learning counterparts due to the limited amount of training data available. It is well established that the SVM models perform exceptionally well on small amounts of data, when compared to other machine learning models, and that deep learning models traditionally require massive amounts of training data to be viable. Therefore, the SVM has an advantage in these particular cases. Nonetheless, our proposed network still outperforms the SVM model, across all the datasets, regardless of the amount of available training data. Our network is specifically designed to extract all available emotionally salient information throughout the text. Thus, it is able to outperform the SVM models, as well as all the other models, through gaining access to and learning from more emotionally rich information.

Emotion	F1 score	
	AB	Our Model
joy	0.95	0.94
sadness	0.95	0.97
anger	0.97	0.92
fear	0.94	0.92
surprise	0.93	0.95
avg	0.948	0.946

Table 3: Comparison of micro F1 scores for the common emotions between this work and the state of the art work in (Abdul-Mageed and Ungar 2017).

Non-baseline models

Our model outperforms the traditional CNN by 5.56 pps in terms of accuracy across all the datasets. Furthermore, our model also consistently outperforms the CNN with respect to F1 score. While a CNN is undeniably good at extracting key features, the semantics of emotions in text can be subtle and difficult to capture completely without taking the rest of the sentence's context into consideration. Therefore our model, which combines the key feature extraction capabilities of CNN's with the long-term "memory" of RNN's, is able to outperform the CNN due to having access to the sentence's emotional context during the classification process.

When comparing Emotion RCNN-EN to the RCNN model that inspired it, it can be seen that the new architecture outperforms the RCNN architecture both in term of F1 score and accuracy. Our model outperforms the RCNN model in terms of accuracy by 3.820 pps on average across all the datasets. Most notable are the results for the evaluations completed using the AG and the DD datasets, with the Emotion RCNN-EN outperforming the RCNN by 7.241 and 4.977 pps respectively. In addition to the combination of key feature extraction and the long-term memory capabilities of our network, we believe this improvement can be attributed to the convolutional, temporally-invariant, and emotionally charged semantic features that are used to initialize the Emotion RCNN-EN's recurrent states. This allows the network to better capture the "long-distance" semantic patterns and relationships that would otherwise be missed by traditional convolutional or recurrent structures. Therefore, our model is able to gain access to more relevant information already in the text which is missed by other models; including the RCNN.

Efficiency of Architecture

In addition to classifying emotions with more precision, our model is more efficient than existing models. We evaluated the efficiency of our system against that of the state of the art work of Abdul-Mageed and Ungar by comparing the F1 scores of the shared emotional categories from the dataset used in their work and the ZM dataset (2017). While the ZM dataset is not a subset of the data used by Abdul-Mageed and Ungar, or vice versa, both datasets are comprised of tweets that where labeled and validated using the same methodology. While this is an indirect comparison of these models, as the authors of this work where unable to attain either a trained model or the dataset used by Abdul-Mageed and Ungar, it can nonetheless be used to illustrate the efficiency of our work. The results of this comparison can be seen in Table 3. The accuracy of this work - in terms of F1 score - is nearly the same as the state of the art work, with the average of our model only 0.2 pps lower than theirs.

The neural network employed by Abdul-Mageed and Ungar was trained and evaluated on 1.608 million examples, whereas our work was only trained and evaluated on 69,120 examples. Despite training with two orders of magnitude less data, our model performs within a reasonable margin of error of the current state of the art system. These results suggest that our model is capable of extracting more relevant information from the data and using it to better classify emotion. This ability is a direct result of the design choices explained in the Network Architecture section.

Conclusion

Experimental results from five separate emotion-in-text datasets demonstrate that our model outperforms traditional machine learning and deep learning text classification techniques alike and produces results that are on average as accurate as the current state of the art system or better despite using two orders of magnitude less training data.

The success of our model can be attributed to two key factors. First, the recurrent convolutional network architecture which enables the model to capture the text's log-term context and most emotionally salient features. Second is seeding the recurrent layers of the model with a prior knowledge of the most emotionally significant information in the text. Doing this gives the recurrent layers in our model a "hindsight" as to the emotional context of the text; a hindsight which enables our model to extract more contextual information from the text.

The combination of both the recurrent convolutional structure and the convolutional layers seeding the recurrent layers results this model processing input twice. First, in the convolutional seeding layers, and second, in the dual-purpose recurrent and convolutional layers. This process extracts far more emotionally significant information from the text than the approaches taken by other works and enables our model to outperform those models; even with limited training data. We conclude that the new emotional classification technique presented here displays promise and needs to be further evaluated on a much larger and emotionally comprehensive dataset.

Acknowledgments

The author of this work would like to thank and recognize Dr. Michael Wollowski of the Rose-Hulman Institute of Technology, Computer Science and Software Engineering department, for his support in writing and editing this work.

References

Abdul-Mageed, M., and Ungar, L. 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)*, 718–728. Vancouver, Canada: Association for Computational Linguistics.

Bostan, L. A. M., and Klinger, R. 2018. An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, 2104–2119.

Bouazizi, M., and Ohtsuki, T. 2017. A pattern-based approach for multi-class sentiment analysis in twitter. *IEEE Access* 5:20617–20639.

Chen, G.; Ye, D.; Xing, Z.; Chen, J.; and Cambria, E. 2017. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In *2017 International Joint Conference on Neural Networks (IJCNN)*, 2377–2383. IEEE.

Creed, C., and Beale, R. 2008. Emotional intelligence: giving computers effective emotional skills to aid interaction. In *Computational Intelligence: A Compendium*. Springer. 185–230.

CrowdFlower. 2016a. Logistic Regression for Text Classification (Sentiment Analysis).

CrowdFlower. 2016b. Sentiment Analysis: Emotion in Text.

González-Ibánez, R.; Muresan, S.; and Wacholder, N. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*, 581–586. Association for Computational Linguistics.

Ioffe, S. 2017. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *Advances in Neural Information Processing Systems*, 1945–1953.

Jaderberg, M.; Vedaldi, A.; and Zisserman, A. 2014. Deep features for text spotting. In *European conference on computer vision*, 512–528. Springer.

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

Lai, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Li, W., and Xu, H. 2014. Text-based emotion classification using emotion cause extraction. *Expert Systems with Applications* 41(4):1742–1749.

Li, Y.; Su, H.; Shen, X.; Li, W.; Cao, Z.; and Niu, S. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Loshchilov, I., and Hutter, F. 2016. SGDR: stochastic gradient descent with restarts. *CoRR* abs/1608.03983.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 1003–1011. Suntec, Singapore: Association for Computational Linguistics.

Mohammad, S. 2012a. #emotional tweets. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, 246–255. Montréal, Canada: Association for Computational Linguistics.

Mohammad, S. M. 2012b. # emotional tweets. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, 246–255. Association for Computational Linguistics.

Pal, S.; Ghosh, S.; and Nag, A. 2018. Sentiment analysis in the light of lstm recurrent neural networks. *International Journal of Synthetic Emotions (IJSE)* 9(1):33–39.

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

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Poria, S.; Cambria, E.; Hazarika, D.; and Vij, P. 2016a. A deeper look into sarcastic tweets using deep convolutional neural networks. *arXiv preprint arXiv:1610.08815*.

Poria, S.; Chaturvedi, I.; Cambria, E.; and Hussain, A. 2016b. Convolutional mkl based multimodal emotion recognition and sentiment analysis. In *2016 IEEE 16th international conference on data mining (ICDM)*, 439–448. IEEE.

Severyn, A., and Moschitti, A. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 959–962. ACM.

Sosa, P. M. 2017. Twitter sentiment analysis using combined lstm-cnn models.

Strapparava, C., and Mihalcea, R. 2008. Learning to identify emotions in text. In *Proceedings of the 2008 ACM symposium on Applied computing*, 1556–1560. ACM.

Wang, W.; Chen, L.; Thirunarayan, K.; and Sheth, A. P. 2012. Harnessing twitter" big data" for automatic emotion identification. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, 587–592. IEEE.

Yin, W.; Kann, K.; Yu, M.; and Schütze, H. 2017. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*.

Zhang, L.; Wang, S.; and Liu, B. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8(4):e1253.

Zolkepli, H. 2018. NLP Emotion Classification Comparison. https://github.com/huseinzol05/NLP-Models-Tensorflow/tree/master/ClassificationComparison.