TCS Research at SemEval-2018 Task 1: Learning Robust Representations using Multi-Attention Architecture

Hardik Meisheri
TCS Research
New Delhi, India
hardik.meisheri@tcs.com

Lipika Dey
TCS Research
New Delhi, India
lipika.dey@tcs.com

Abstract

This paper presents system description of our submission to the SemEval-2018 task-1: Affect in tweets for the English language. We combine three different features generated using deep learning models and traditional methods in support vector machines to create a unified ensemble system. A robust representation of a tweet is learned using a multi-attention based architecture which uses a mixture of different pre-trained embeddings. In addition, analysis of different features is also presented. Our system ranked 2^{nd}, 5^{th}, and 7^{th} in different subtasks among *75* teams.

1 Introduction

In Natural Language processing, Sentiment analysis refers to the degree of positiveness or negativeness of the information presented in the text. Traditionally sentiment analysis is treated as either a binary classification task (positive, negative) or a multi-class classification task (very negative, negative, neutral, positive, very positive). Affect analysis on the other hand refers to detecting discrete sets of emotions present in the text such as anger, joy, sadness etc (Dalgleish and Power, 2000; Plutchik, 2001). Predicting intensities of these emotions to fine granularity can help us better understand the sentiment and emotions of the writer.

Detecting sentiments or affect from text have a number of useful applications. For example, the degree of disgust or anger expressed in customer complaints or reviews can help us decide the priorities of issues to look at, or the joy or optimism expressed in customer feedbacks can be a major factor in deciding the marketing strategy for a company.

Sentiment or affect analysis for social media text is a challenging task due to the extensive use of slang, frequent spelling mistakes, innovative and unpredictable use of hashtags and extensive use of emojis and smileys.

SemEval-2018 Task 1: Affect in tweets, provides data for 3 languages: English, Arabic, and Spanish. For each language, there are 5 subtasks that are presented, 2 Regression tasks, 2 classification tasks and 1 Multi-Label task. Further details of tasks are presented in section 3. This task was similar to the WASSA shared task (Mohammad and Bravo-Marquez, 2017) and dataset presented here is the extension of the data presented for WASSA shared task.

In this paper, we present our approach to solving these tasks for English language tweets. We have proposed a system which uses various pre-trained embeddings to handle out-of-vocabulary words and emoji present in the text along with cleaning of raw text. In addition, to create a better representation of the text, we use two sets of embeddings learnt over two different corpus which results in parallel attention mechanism - one set from the twitter space and another from a common crawl corpus. Finally, we combine features generated from the deep learning model with other features to generate an ensemble system.

Major contributions of this paper are:
1. Generating word vector representation of a tweet from three different set of pre-trained embeddings which can handle emoji/smileys and the out of vocabulary words in the dataset.
2. Deep neural network architecture which generates robust representation of the text with the help of parallel attention mechanism.

Rest of the paper is organized as follows, Section 2 presents the preprocessing step to generate mixed set of embeddings and the model architecture. It also presents the different sets of features that are used for final ensemble system. In section 3, data, training and experimentation setup is described for different subtasks. Section 4 states

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 291–299
New Orleans, Louisiana, June 5–6, 2018.

the results of the proposed system and detailed discussion of the feature over development and test data. Finally, section 5 concludes the paper with summary of the approach presented.

2 Proposed Approach

Figure 1 describes the overall system architecture used for regression and classification subtasks. We have extracted different types of features from the raw text, which fall under three different categories. Deep learning features are the ones which are generated from the model that is trained and proposed in this paper. Lexicon-based features are generated from training sets. In addition, features from pre-trained models are used. These models were trained over large corpus.

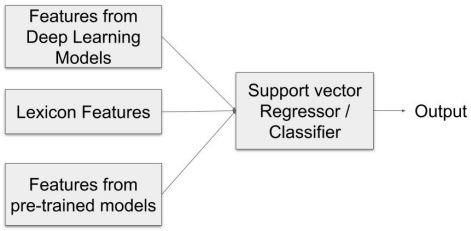

Figure 1: System Diagram.

Attention mechanism has been successful in sequence to sequence learning problems specifically for neural machine translation (NMT) (Bahdanau et al., 2014). These mechanism helps model to focus more on the task in hand. The proposed architecture uses two parallel processing towers which use attention as a means of focusing on sentiment specific words. Figure 2 presents a snapshot of the random sample for sadness emotion, first row denotes the output of the attention mechanism and row 2 denotes the output of the model. We observe that the attention mechanism helps in focusing on words which are relevant to the sentiment task in hand, such as crying, dying etc. which further helps in improving the performance of the model. The multi-attention mechanism is inspired from (Lin et al., 2017) where, they have used more multi-attention over the same embedding space to focus on more than one word. In contrast, we use limit our attention to at max 2 words as the tweet is much more compact in nature as well as we do it over different embedding space to encapsulate much more information.

For generating features as mentioned above, we employ pre-processing steps to normalize the text with respect to sentiment specific words and its usage.

2.1 Preprocessing

As mentioned earlier, tweets in the raw form are noisy and prone to many distortions in terms of syntactic and semantic structure. These preprocessing steps are common to all the features generated. Deep learning features require additional steps which are explained in respective section.

1. All the characters in the text are converted to lower case.
2. Twitter contains lot of words with more than 2 repeating characters such as happpyyyyyyy, we limited occurrence of each character to maximum of 2 successive times.
3. To handle hashtags, # symbol is removed from all the words.
4. Extra spaces and new line character is deleted from the tweet to ensure the compactness of the tweets.

2.2 Deep Learning Features

Figure 3 shows the model which was used to generate deep learning features. In this model, we have used different embeddings to enhance the representations of raw text. There are two parallel architectures which take the same raw input but generate the representation from a different embedding space. This helps in encapsulating the word and its usage in twitter space as well as keeping a general semantic and syntactic structure of word intact.

For tower one in the figure 3, text is preprocessed using steps mentioned earlier. In addition, following pre-processing steps are performed:

1. *Usernames* in twitter which starts with @ is replaced by **mention** token.
2. Punctuations are removed except *[,], [?], [!], [.]*
3. Words that are most probably used as slangs in twitter are replaced with its corresponding expanded versions such as *"y'all"* is replace by *"you all"*.

Embedding matrix is generated from the preprocessed text using combination of three pretrained embeddings: Glove (Pennington et al.,

Figure 2: Attention Example.

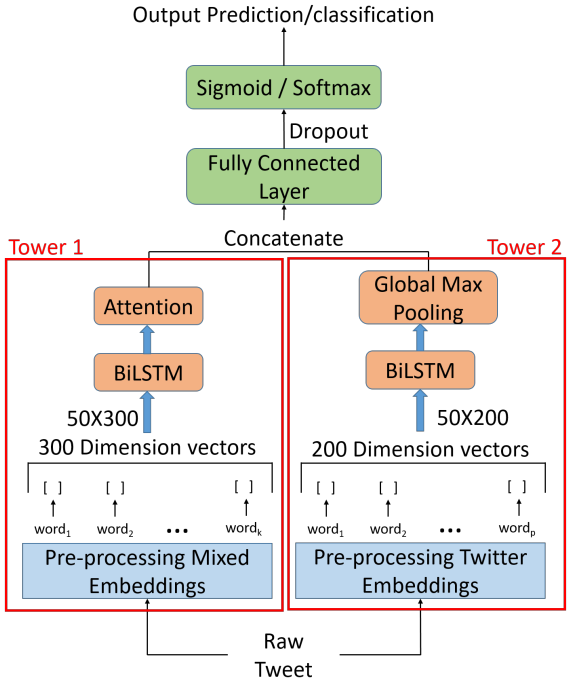

Figure 3: Model Diagram.

word_token = Tokenize tweet
for *each word in word_token* **do**
  if *word is in EmojiEmbb* **then**
    word_vector = get_vector(EmojiEmbb, word_vector)
  else if *word is in Glove* **then**
    word_vector = get_vector(Glove, word_vector)
  else if *word is in CharEmbb* **then**
    word_vector = get_vector(charEmbb, word_vector)
  else
    chars = tokeinze word_token into character
    n = length(chars) word_vector = $\sum_1^n get\_vector(charEmbb, chars)$
  end
end

Algorithm 1: Embedding Matrix generation

2014) trained over common crawl corpus with 300 dimension vector, Character[1] level embeddings trained over common crawl glove corpus providing 300 dimensional vectors for each character and emoji2vec (Eisner et al., 2016) which provides 300 dimension vectors for most commonly used emojis in twitter platform. Procedure to generate a representation of a text using all these embeddings is presented in Algorithm 1, where *get_vector* is a function of token and embedding type and returns the corresponding vector for the token from the pre-trained embedding specified.

Embedding vectors that are generated for each tweet are then converted to into matrix, with a number of rows being the size of maximum sequence, rest is zero padded. This matrix then forms the input to the Bidirectional LSTM(BiLSTM) layer (Graves and Schmidhuber, 2005), which helps in generating representations by taking all the words in sequence into account.

[1]https://github.com/minimaxir/char-embeddings

The output of each time-step is then fed to Attention Mechanism (Bahdanau et al., 2014). The core concept behind the attention mechanism forces the model to focus on important words that are related to the task.

Tower 2 in the Figure 3 uses pre-processed mechanism where, all the punctuations are removed, usernames are removed without any replacement with tokens and, special characters including smiley and emojis are removed. Embedding matrix is generated using pre-trained glove embeddings trained over twitter corpus and provides 200-dimensional vectors for each word. These are zero padded as mentioned earlier and is fed into another BiLSTM layer. Maxpooling is applied over the output of BiLSTM to extract the most prominent vector from the rest over the temporal dimension which act as a attention over word sequences. Maximum sequence length for the embedding space is kept at 50, as twitter has a character limit of 140 characters.

The output of tower 1 and 2 are then concatenated and then fed into the fully connected network with 2 layers. Final layer contains a different number of neurons and activation functions de-

pending on the subtasks which are stated in the experiments section 4. To handle overfitting we have used L2 regularization dropout in layers and batch size is kept at 512.

2.3 Traditional Features

We defined features that are used in most of the traditional sentiment analysis techniques are termed as traditional features. As per the baseline system provided in the WASSA Emotion Intensity Task we define baseline features. The knowledge sources that have been used to generate baseline feature are: MPQA subjective lexicon (Wilson et al., 2005), Bing Liu lexicon (Ding et al., 2008), AFINN (Nielsen, 2011), Sentiment140 (Kiritchenko et al., 2014), NRC Hashtag Sentiment Lexicon (Mohammad and Kiritchenko, 2015), NRC Hashtag Emotion Association Lexicon (Mohammad et al., 2013), NRC Word-Emotion Association Lexicon (Mohammad and Turney, 2013), NRC-10 Expanded Lexicon (Bravo-Marquez et al., 2016) and the SentiWordNet (Esuli and Sebastiani, 2007). Two more features are calculated on the basis of emoticons (obtained from AFINN (Nielsen, 2011)) and negations present in the text. This amounts to 45 features for each tweet.

In addition to this, we have used Vader Sentiment Lexicons (Gilbert, 2014), which provides the positive, negative, neutral and compound score for the text. These lexicons are specifically designed for social media texts.

2.4 Features from pre-trained models

We use **SentiNeuron** feature (Radford et al., 2017) from a model which is trained over 82 million Amazon review dataset. The aim of the model was to predict the next word in the review. They have used LSTM with 4096 units. The 2389^{th} neuron was found to be specifically focusing on the sentiment for a given sentence. We use output of this 2389^{th} as a feature. Further more, we have normalized it between *0-1* which helps in performance improvement.

3 Data and Experiments

We participated in all the subtasks of English language, namely: EI-reg (intensity score prediction of 4 emotions), EI-oc (intensity ordinal classification task of 4 emotions), V-reg (intensity score prediction of valence), V-oc (intensity ordinal classification task for valence) and E-c (Multi-label classification task over 11 emotions). Detailed analysis and distribution of the dataset are presented in the task paper (Mohammad et al., 2018).

Table 1: Data Distribution.

	Train	Dev	Test
Anger	1701	388	1002
Fear	2252	389	986
Joy	1616	290	1105
Sadness	1533	397	975
Valence	1181	449	937
Multi-Label	6838	886	3259

For each subtask, deep learning model is same as mentioned earlier, although there is a variation in the feature being used for ensemble approach. Data distribution across train, dev and test dataset is given table 1.

3.1 EI-reg and V-reg: Regression

In this task, given a tweet and its corresponding emotion, we need to predict the intensity of the given emotion in *0-1* range. For this task, Deep learning models with *sigmoid* as activation function and number of hidden unit in last layer as one is used. Official evaluation metric for this is a pearson correlation, so we define a new loss function to train deep learning models.

$$Loss = 0.7 \times (1 - pearson) + 0.3 \times MSE \quad (1)$$

This is a slightly modified version than used by (Meisheri et al., 2017; Goel et al., 2017) for WASSA dataset, where they use the negative of pearson correlation as the loss function. We observe that using weighted sum of negation of pearson correlation and mean square error improved the performance.

Training data was split into 10 different folds, by using stratified splits which were achieved by generating 10 bins over the continuous bins. Ten different models were generated with permutations of 9 out of 10 folds as training and remaining 1 as validation dataset. Finally, dev dataset is passed over all the models and mean of all the models were considered as output. This can be seen as a variation of weak learners concept in decision trees.

For the testing dataset as mentioned earlier, we combine training and development set and then

generate 10 folds to create 10 models with *80-20* split for validation. Parameters used for models are stated in table 2. In addition, we have used Adam optimizer with *0.0001* as learning rate.

Table 2: Parameters for Regression Task.

Layers	Units	Activation	Regularization	Dropout
BiLSTM - Tower 1	70	Tanh	L2 - 0.05	0.35
BiLSTM - Tower 2	70	Tanh	L2 - 0.05	0.35
Attention	-	-	L2 - 0.01	-
Max Pooling	-	-	-	-
Fully Connected Layer 1	100	Selu	L2 - 0.001	0.5
Fully Connected Layer 2	50	Selu	L2 - 0.001	0.3
Output Layer	1	Sigmoid	-	-

The output of deep learning models is considered as a feature for our ensemble method, where we combine other features as mentioned in section 2.3 and section 2.4. In addition to this, the output of other emotion is also used as a feature for the ensemble model which provides an additional context for the prediction task. So, for each emotion in a task, we get additional features from deep learning model which we define as a cross emotion features.

All this features are then passed on to the support vector regression, whose parameters *C* and *Kernel* are tuned using 10 fold cross validation over training set.

3.2 EI-oc and V-oc: Classification

Objective of this task was to classify tweet into one of the ordinal classes, given a tweet and its corresponding emotion. Number of classes for EI-oc were four and for V-oc it was seven. Official evaluation metric for this task was provided as pearson correlation. Output layer in the deep learning model contained four and seven neurons for EI-oc and V-oc respectively with softmax as the activation function. Loss function used for classification task was *categorical crossentropy*. Similar settings of 10-fold as mentioned in regression task earlier was carried out resulting in 10 different models for each emotion and valence. Layer parameters for this task are summarized in table 3. Stochastic gradient descent with nesterov momentum and learning rate *0.01* was used as optimizer for this task.

Similar to regression task, for classification we create ensemble model by combining output of deep learning models with other features. In addition, we also consider output of regression models as additional features for classification. Support vector classifier is used as a final classifier, with *C* and *Kernel* being tuned using 10-fold cross validation. Final submission is done with model being trained by combining training and development dataset and then taking 80-20 split for training and validation.

Table 3: Parameters for Classification Task.

Layers	Units	Activation	Regularization	Dropout
BiLSTM - Tower 1	50	Tanh	L2 - 0.05	0.4
BiLSTM - Tower 2	50	Tanh	L2 - 0.05	0.4
Attention	-	-	L2 - 0.001	-
Max Pooling	-	-	-	-
Fully Connected Layer 1	50	Selu	L2 - 0.01	0.4
Fully Connected Layer 2	20	Selu	L2 - 0.01	0.4
Output Layer	5/7	Softmax	-	-

Table 4: Parameters for Multi-Label Classification Task.

Layers	Units	Activation	Regularization	Dropout
BiLSTM - Tower 1	120	Tanh	0	0.3
BiLSTM - Tower 2	120	Tanh	0	0.3
Attention	-	-	0	-
Max Pooling	-	-	-	-
Fully Connected Layer 1	100	relu	L2 - 0.01	0.3
Fully Connected Layer 2	50	relu	L2 - 0.01	0.2
Output Layer	11	Softmax	-	-

3.3 E-c: Multilabel Classification

In this task, we were provided with tweet and its corresponding labels among 11 emotion: *anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise and trust*. For this task, output layer of our deep learning model contains 11 neurons and sigmoid as a activation function. *Binary cross entropy* is used as a loss function with *Stochastic gradient descent* with Nesterov momentum, *0.01* learning rate and 10^{-6} learning rate decay as optimizer. Parameters for other layers are presented in table 4.

Official evaluation metric for this task was Jaccard similarity score. The output of the deep learning model gives values between *0-1* for each emotion. Since the task was to predict the presence or absence of any emotion continuous value must be converted to binary number. Threshold was applied which was learned from training and development set. For training set we found the threshold to be *0.35*, whereas for development set it was found to be *0.30*. For testing set, we take the mean of both the values as threshold.

4 Results and Discussion

In total *75* teams participated in the task, our system was ranked 2^{nd} for V-reg task, 5^{th} for EI-reg task and 7^{th} for both V-oc and EI-oc task. For

Table 5: Comparison of proposed deep Learning model and ensemble model over train and development set for regression task.

	Development Set				Test Set			
	Original Split		80-20 Split		Original Split		80-20 Split	
	Ensemble System	Deep Learning Model	Ensemble System	Deep Learning Model	Ensemble System	Deep Learning Model	Ensemble System	Deep Learning Model
Fear	0.751	0.707	0.791	0.791	0.745	0.725	0.736	0.74
Anger	0.79	0.718	0.747	0.744	0.775	0.721	0.776	0.749
Sadness	0.735	0.689	0.77	0.767	0.764	0.723	0.776	0.741
Joy	0.723	0.675	0.812	0.775	0.767	0.724	0.77	0.731
Average	0.75	0.697	0.78	0.769	0.763	0.723	0.764	0.74
Valence	0.857	0.804	0.85	0.788	0.858	0.832	0.861	0.84

Table 6: Results of Regression and Classification task over test set.

	Reg		OC	
	Orginal Split	80-20 Split	Orginal Split	80-20 Split
Fear	0.745	0.735	0.595	0.561
Anger	0.775	0.775	0.626	0.641
Sadness	0.764	0.776	0.618	0.621
joy	0.767	0.77	0.65	0.655
Average	0.76275	0.764	0.62225	0.6195
Valence	0.858	0.861	0.727	0.777

Multi-Label classification, our system achieved 2^{nd} rank among the teams, with Jaccard similarity score of *0.582*.

Table 6 shows the result for EI-reg, V-reg, EI-oc and V-oc task on official evaluation metric i.e. pearson correlation. We also compare the results over the original split and *80-20* split generated after combining training and development dataset. It can be seen that both of these gives similar results while, for classification original split is better, for regression it is other way. Table 5 shows comparison of Ensemble model and Deep learning model for EI-reg and V-reg. We observe improvement in ensemble model over development dataset in both sets of splits. On the contrary, there is relatively less difference in the test set.

Table 7 contains results of different deep learning architecture for EI-reg and V-reg task. For both of these task, we can observe what is impact of attention over both the towers. We also present the results for each single tower which helps in understanding the need for two towers. Although adding attention over Tower-1 gives little improvement for EL-reg task it provides significant improvement for V-reg task. It is worthwhile to note that *sadness* emotion shows no improvement by adding attention over tower-2.

Table 7: Regression Task Results over model different architectures over test set:80-20 Split.

	Anger	Fear	Sadness	Joy	Average	Valence
Proposed Model	0.749	0.74	0.741	0.731	0.74	0.84
Tower-1	0.727	0.727	0.704	0.709	0.717	0.825
Tower-2	0.714	0.719	0.673	0.70	0.705	0.792
Tower-1 without Attention	0.721	0.709	0.69	0.704	0.711	0.783
Tower-2 without Attention	0.693	0.692	0.673	0.682	0.685	0.766

Table 8: Results of Individual Features in combination with Deep learning features over development set.

Features	Anger	Fear	Sadness	Joy	Valence
dl	0.744	0.791	0.767	0.775	0.788
dl+baselines	0.747	0.792	0.773	0.778	0.789
dl+vader	0.747	0.792	0.772	0.775	0.785
dl+sentineuron	0.748	0.79	0.773	0.778	0.79
dl+valence	0.751	0.792	0.77	0.785	-
dl+cross emotion	0.75	0.794	0.773	0.778	0.809

In table 8 and table 9, results on regression task for *80-20* split for each feature when combined with deep learning feature over development and test set respectively. We observe that adding lexicon features marginally increases the performance of the system. We can conclude from this that deep learning model that we presented can encapsulate most of the information regarding the sentiment which was present in traditional features. Including cross-emotion feature shows considerable increase in the performance.

Inter-feature correlation is presented in figure 4, where we can observe that apart from anger and valence baseline features are weak negatively correlated with other features. Furthermore, vader and sentineuron are less correlated except for valence and yet they provide similar improvement when combined individually with DL features. Although, when both these features are combined together they provide a significant boost.

Table 9: Results of Individual Features in combination with Deep learning features over test set.

Features	Anger	Fear	Sadness	Joy	Valence
dl	0.749	0.74	0.741	0.731	0.84
dl+baselines	0.755	0.739	0.742	0.731	0.84
dl+vader	0.753	0.744	0.746	0.731	0.841
dl+sentineuron	0.753	0.739	0.752	0.734	0.842
dl+valence	0.756	0.743	0.745	0.74	-
dl+cross emotion	0.759	0.744	0.753	0.738	0.849

For classification task across four emotion and valence we observed that, using threshold values obtained by comparing continuous values from regression task provides a better result in pearson correlation. Possible reason for this might be the loss function that we trained for classification model was categorical cross entropy.

4.1 Error analysis

We observe that high difference between the predicted value/class and truth value/class are present at the extreme end of the spectrum. One of the possible reason might be that the distribution of the data shows a Gaussian distribution and there are few samples at the extreme end as described in (Mohammad and Kiritchenko, 2018). In addition, we manually inspect some cases where our model failed, for example for sadness *You are MINE, my baby, my headache, my love, my smile, my frown, my wrong, my right, my pain, my happiness, my everything.* has truth value of *0.140* and our system predicted *0.568* which is way higher than what the writer is trying to convey. The model is predicting slightly above neutral sentiment. Possible reasons include the presence of both positive and negative words present in the alternate sequence. This kind of discourse and irony detection can help in better prediction if incorporated into the models.

In joy emotion, *when will i ever be happy with myself?* has a truth value of *0.109* and predicted value is *0.491*. These kind of rhetorical questions is hard to understand even for humans, for model to understand we need to put in some explicit context. By observing more such samples, we find that adding more context about the different physiological and linguistic phenomenon into the model with appropriate bias can greatly increase the accuracies of the models present.

Table 10 shows the error across different emotions in multi-label task. We observe that there is a high error rate in *anticipation, pessimism, surprise and trust*, possible reasons might be that there are already fewer samples available and the ratio of percentage votes received to the percentage of tweets labeled is also high for this emotion as compared to other emotions (Mohammad and Kiritchenko, 2018). In addition, we observe that there are around *2%* of the tweets contained no emotion in test set, where our model predicted atleast one emotion.

Table 10: Multi-label Error across Emotions.

Emotion	Error	Presence total	Ratio
Anger	521	1101	0.473
Anticipation	469	425	1.103
Disgust	646	1099	0.588
Fear	251	485	0.518
Joy	477	1442	0.331
Love	405	516	0.785
Optimism	704	1143	0.616
Pessimism	398	375	1.061
Sadness	539	960	0.561
Surprise	167	170	0.982
Trust	161	153	1.052

5 Conclusion

In this paper, we describe our approach to SemEval-2018 Task-1 for English tweet. We present ensemble system which is capable of handling noisy sentiment dataset over regression, classification as well as multi-label dataset. Use of the mixture of embedding in parallel makes this system unique in terms of generating better representations with respect to sentiment. Our system achieved 2^{nd}, 5^{th} and 7^{th} in different subtasks. Analyzing different feature combinations from individual results and inter-feature correlation over test data reveals that our deep learning model is able to capture most of the information that is provided by lexicon feature. Multi-label classification has proved to be a challenging task among all the subtask that has been provided as the evaluation score of all the team participating has been low.

We have also presented some examples where our model has performed poorly and conclude that including context feature for sarcasm, irony and rhetoric question can improve the performance further over all the subtasks presented in SemEval for English language.

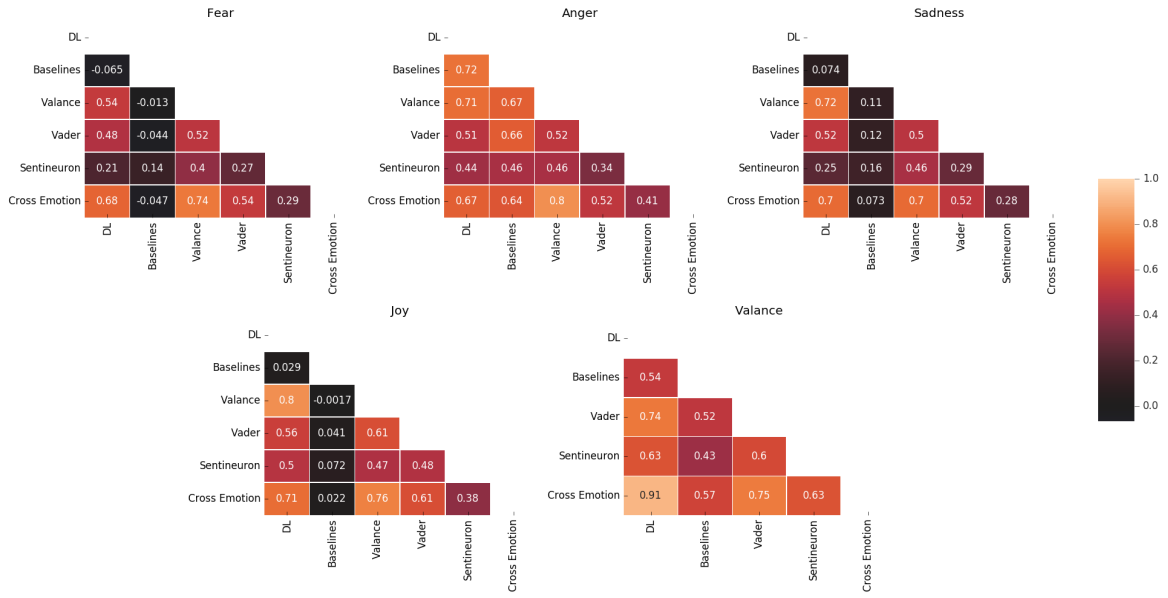

Figure 4: Correlation among various features for test set.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Felipe Bravo-Marquez, Eibe Frank, Saif M Mohammad, and Bernhard Pfahringer. 2016. Determining word–emotion associations from tweets by multi-label classification. In *WI'16*, pages 536–539. IEEE Computer Society.

Tim Dalgleish and Mick Power. 2000. *Handbook of cognition and emotion*. John Wiley & Sons.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining*, pages 231–240. ACM.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bosnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description.

Andrea Esuli and Fabrizio Sebastiani. 2007. Sentiwordnet: A high-coverage lexical resource for opinion mining. *Evaluation*, pages 1–26.

CJ Hutto Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf*.

Pranav Goel, Devang Kulshreshtha, Prayas Jain, and Kaushal Kumar Shukla. 2017. Prayas at emoint 2017: An ensemble of deep neural architectures for emotion intensity prediction in tweets. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 58–65.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Hardik Meisheri, Kunal Ranjan, and Lipika Dey. 2017. Sentiment extraction from consumer-generated noisy short texts. In *Proceedings of IEEE International Conference on Data Mining Workshops (ICDMW)*, New Orleans, USA.

Saif M. Mohammad and Felipe Bravo-Marquez. 2017. Wassa-2017 shared task on emotion intensity. In *Proceedings of the EMNLP 2017 Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media (WASSA), September 2017, Copenhagen, Denmark*.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Saif M Mohammad and Svetlana Kiritchenko. 2015. Using hashtags to capture fine emotion categories from tweets. *Computational Intelligence*, 31(2):301–326.

Saif M. Mohammad and Svetlana Kiritchenko. 2018. Understanding emotions: A dataset of tweets to study interactions between affect categories. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference*, Miyazaki, Japan.

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*.

Saif M Mohammad and Peter D Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Finn Årup Nielsen. 2011. A new anew: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*.

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.

Robert Plutchik. 2001. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American scientist*, 89(4):344–350.

A. Radford, R. Jozefowicz, and I. Sutskever. 2017. Learning to Generate Reviews and Discovering Sentiment. *ArXiv e-prints*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.