Emotional Intensity Classification

Srigayatri Rachakonda
grachako@sfu.ca

Oluwaseyi Talabi
otalabi@sfu.ca

Saghar Irandoust
sirandou@sfu.ca

Abstract

Emotional intensity describes the variations in how the magnitude of emotion is expressed in individuals. Having this knowledge about customers' opinions can help enhance the capabilities of a commercial system. We focus on predicting the degree of emotional intensity of tweets in this project, as social media captures a wide range of emotions. This project is the second subtask of Task 1 (Affect in Tweets) of SemEval 2018. We developed pre-trained language models and deep learning models that were used to classify the emotional intensity of the 4 different classes of emotion namely anger, fear, joy and sadness followed by a comparison of the models' results to the model that won the competition.

1 Introduction

We communicate using language to convey not only our thoughts and opinions but also the sentiment, emotions, and intensity of the emotions expressed. Therefore, it has become critical for computer applications to understand the degree of a person's emotions in order to provide personalized experiences. It is often sufficient to understand whether the person is showing intense emotions or not. Therefore, in this project, we aim to predict the emotional intensity category (no emotion, low emotion, moderate emotion, or high emotion) of a tweet given its emotion class that will best represent the state of the tweeter.

2 Approach

Our aim is to compare 2 different approaches - BERT classifiers, LSTM classifiers and compare them with the baseline of the model.

2.1 Baseline

We use SeerNet (Duppada et al., 2018), the model proposed by the winner of the SemEval task in 2018, as our baseline. SeerNet uses several word/sentence embeddings and trains 2models separately - an XGBoost Classifier and a Random Forest Classifier. An ensemble of these 2 models is then generated to produce the final result. The base-line model developed 5 feature vectors representing the tweet at the time of development.

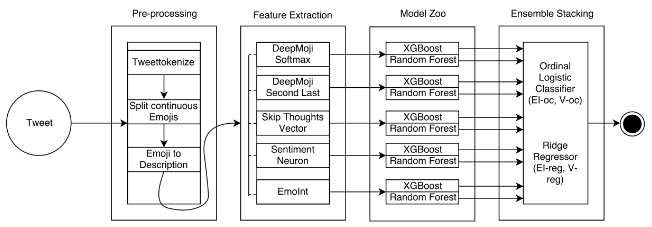


Figure 1: SeerNet System Architecture

2.2 BERT Classifiers

BERT, a bidirectional contextual language model, released at the end of 2018. It has since achieved state-of-the-art results for several NLP tasks. BERT had not been released at the time of the release of SeerNet. As BERT identifies the contextual relationship between words, we use it to see how it impacts the overall results compared to using an ensemble of tree-based methods.

2.3 LSTMs Classifier

In addition, we also trained LSTM-based classifiers to predict emotional intensity.

$$i_t = \sigma(W^i x_t + U^i h_{t-1} + b^i)$$
$$f_t = \sigma(W^f x_t + U^f h_{t-1} + b^f)$$
$$o_t = \sigma(W^o x_t + U^o h_{t-1} + b^o)$$
$$g_t = tanh(W^g x_t + U^g h_{t-1} + b^g)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ g_t$$
$$h_t = o_t \circ tanh(c_t)$$

Long-Short Term Memory network (LSTMs), are capable of remembering information for long periods of time, throughout the sentence. It does so by following a gated mechanism. The forget gate of LSTMs decides if the previous input must be forgotten or not. The cell state contains the previously known context. An input gate checks if any new information is to be added to the

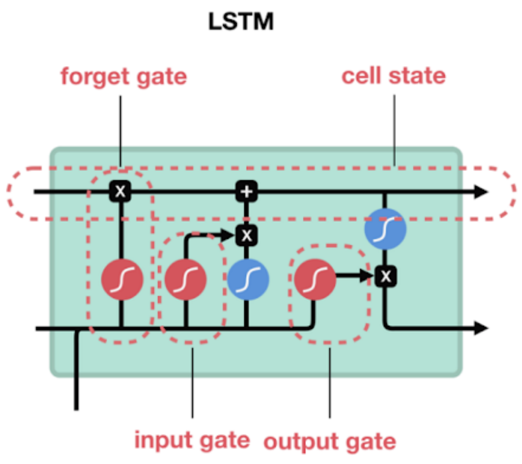


Figure 2: LSTM System Architecture

cell state or not, and the output gate takes the input and the cell state to determine the new output.

As LSTMs are very powerful in capturing the sequence of the information, we wanted to observe how using these classifiers would affect the results and whether they would outperform the BERT classifier or not.

2.4 Code

We followed a fairly simple cleaning process for the tweets, by removing the screen names of people, removing URLs, and removing any retweet references. We also converted the emojis descriptions and used them as we discovered that emojis help indicate the intensity of the emotion. The dataloader was adapted from (Himanshu, 2018).

The code for BERT models we built was adapted from (Mccormick and Ryan, 2019). However, we modified the code to suit our purpose. We also built a Hybrid BERT model which was an adaptation from (Handelman, 2019) Github repository that takes not only the text but also the emotion to classify the intensity. The code for LSTM based classifier was adapted from (Vania, 2018).

3 Datasets

The dataset that is used for this project is from the SemEval-2018 Task 1 dataset, specifically, the Emotional Intensity - ordinal classification dataset. The data contains tweets collected in English, Arabic, and Spanish; however, for this task, we will be focusing only on English tweets. The dataset was collected by polling the Twitter API based on certain terms for every emotion (joy, anger, fear, and sadness) and was annotated into 4 classes - no emotion, low emotion, moderate emotion, and high emotion.

Exploratory Analysis: During our exploration of the dataset, we found many tweets that were tagged to fear (more than 2000) while there were around 1700 tweets each for the other emotions. We also discovered that most of the tweets having no intensity in them. These findings are shown in the figures below.

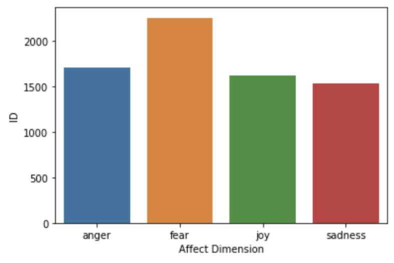


Figure 3: Number of tweets for each emotion

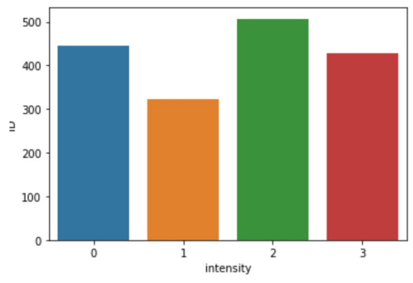


Figure 4: Distribution of intensity for Anger

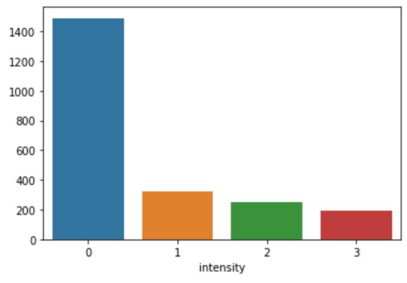


Figure 5: Distribution of intensity for Fear

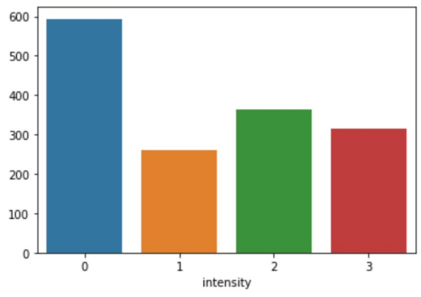


Figure 6: Distribution of intensity for Sadness

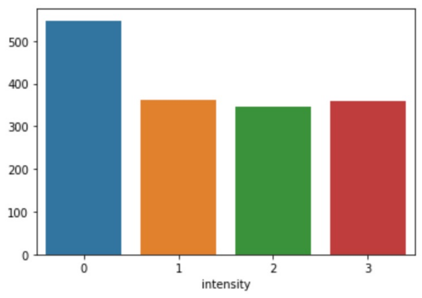


Figure 7: Distribution of intensity for Joy

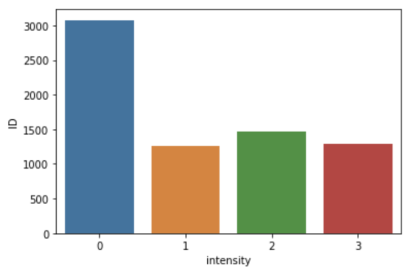


Figure 8: Classes Distribution for all emotions

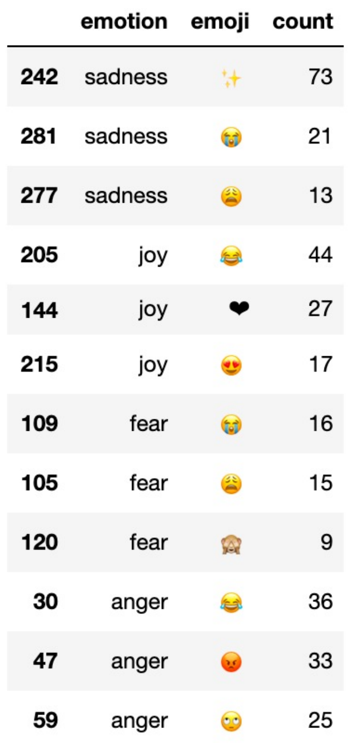
We also found 192 tweets that belonged to more than 1 emotion class. However, among them, 130 of them, they showed emotional intensity only for 1 main class, while for the others, they showed no emotion. As there are only 62 tweets having multiple classes, we are assuming that a single tweet contains only one emotion, and for any other emotion, the tweet has a 0 emotional intensity.

We also found that certain emojis were specifically used with certain emotions. The top 3 emojis used for each emotion is shown below.

4 Evaluation

The task uses a Pearson Correlation coefficient as seen in (Mohammad et al., 2018) between the actual labels and the predicted labels for each emotion (joy, anger, fear, and sadness). The final result is macro-averaged to compare with the baseline model. In addition, we are also comparing our models to the baseline using several secondary evaluation techniques:

- Pearson correlation for a subset of the test set that includes only those tweets with intensity classes low X, moderate X, or high X (where X is an emotion). The organizers refer to this set of tweets as the some-emotion subset (SE)
- Weighted quadratic kappa
- Weighted quadratic kappa on the some-emotion subset of the test set



	emotion	emoji	count
242	sadness	✨	73
281	sadness	😭	21
277	sadness	😩	13
205	joy	😂	44
144	joy	❤	27
215	joy	😍	17
109	fear	😭	16
105	fear	😩	15
120	fear	🙈	9
30	anger	😂	36
47	anger	😡	33
59	anger	🙄	25

Figure 9: Top 3 emojis used with each emotion

5 Experiments

5.1 BERT Classification Models

We conducted several experiments to find out the best performing BERT model.

5.1.1 Simple BERT Classifiers - Uncased

In order to solve the emotional intensity problem, we built a classifier for each emotion. The architecture for the BERT classifier is shown below.

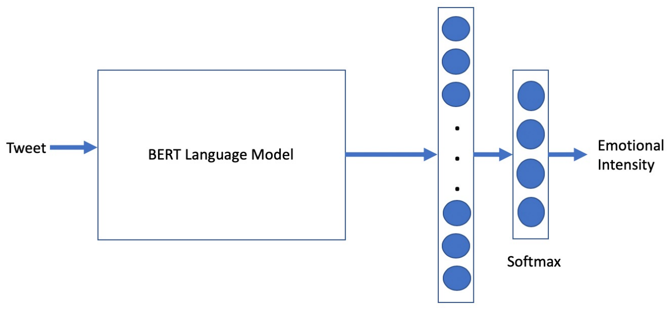


Figure 10: BERT Architecture

We passed the inputs through the BERT language model and built a dropout and linear layer on top of it. We then ran it through a softmax layer to get the log probabilities.

We initially used an Adam optimizer for each BERT classifier and trained it on each emotion subset. However, we noticed that most of the results remained the same due to its quick convergence. Therefore, we decided to linearly decrease the learning rate at every step. This improved the scores.

We also tuned several hyper-parameters in order to improve our models. We used different learning rates - $2e-5, 3e-5, 5e-5$, dropout probabilities - $0.1, 0.2, 0.3$, epsilon values- $1e-6, 1e-7, 1e-8$, and to assign lower learning rates to the lower layers we used weight decay values- $0.8, 0.85, 0.9, 0.95$. We initially trained for 10 epochs, however, we quickly realized that keeping the number of epochs to 6 worked best. For each task, we selected the best fine-tuning parameters. Using a learning rate of $2e-5$ gave the best performing model for all emotions.

The hyperparameters used for training this model can be seen in Table 1.

5.1.2 Simple BERT Classifiers - Cased

As tweets contain a lot of emotional content, the case of the alphabet (uppercase/lowercase) may be used to determine the emotional intensity. For instance, the use of more capital letters signifies extreme emotion. We trained 4 BERT classifiers, similar to the previous approach, this time preserving the case of the tweet. The results are shown in Table 5

5.1.3 BERT Ordinal Classification Models

As emotional intensities are ordered, We can take advantage of this attribute and transform a 4-class classification problem into a 3-binary classification problem (Frank and Hall, 2001). The i-th binary classifier tries to distinguish between the first i classes versus the others.

We used 3 binary classifiers for each emotion separately, with the same structure as section 5.1.1 and trained them with the same approach. Only this time we gave the binary classes weights due to the extreme data imbalance caused by the nature of ordinal classifiers. The models are shown below and the results are shown in Table 6.

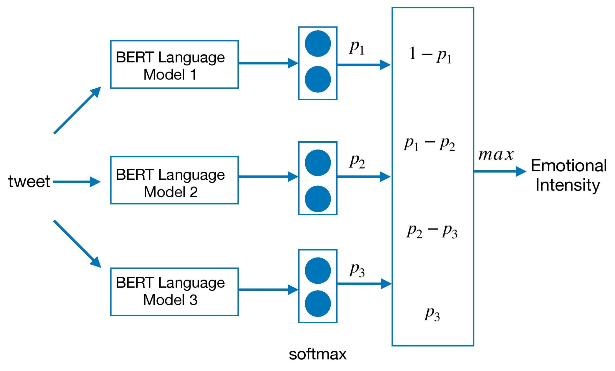


Figure 11: Ordinal Classifier used for each emotion

P_i refers to the probability that emotion intensity is equal to or higher than i. Then the probability of the intensities 0-3 are computed as above, and the maximum determines the intensity.

5.1.4 BERT Hybrid Model

Instead of building a different classifier for each emotion, we decided to try building 1 classifier used for all emotions to see if it resulted in an improvement in the evaluation metrics.

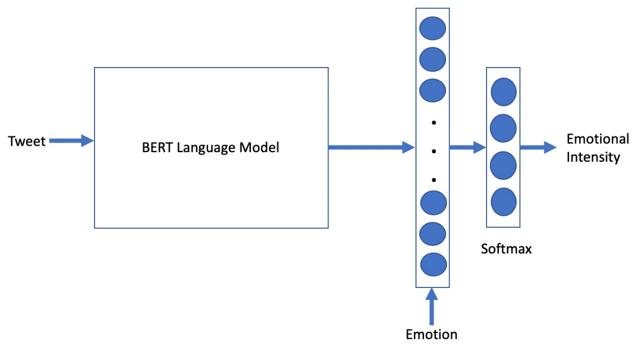


Figure 12: Hybrid Architecture

We modified the BERT classifier structure mentioned above to pass a one-hot encoded vector of emotions as an input to the last layer of the BERT classifier. This ensures that the language modelling part of the classifier remains the same, while also avoiding training multiple models. The architecture is shown below:

Similar to the previous models, we trained this model over multiple hyper-parameters. The final list of hyper-parameters can be seen in below.

	Learning Rate	Dropout	Weight Decay	Epsilon
Anger Uncased	2e-5	0.3	0.9	1e-8
Anger Cased	2e-5	0.2	0.95	1e-8
Fear Uncased	2e-5	0.1	0.95	1e-8
Fear Cased	2e-5	0.2	0.85	1e-6
Sadness Uncased	2e-5	0.1	0.9	1e-8
Sadness Cased	2e-5	0.1	0.95	1e-8
Joy Uncased	2e-5	0.2	-	1e-8
Joy Cased	2e-5	0.2	0.85	1e-8
Ordinal BERT	2e-5	-	-	1e-8
Hybrid BERT	3e-5	0.1	0.9	1e-8

Table 1: Hyper-parameters used for BERT

5.2 LSTM-based Classifiers

5.2.1 LSTM

We built an LSTM based classifier using pre-trained GloVe word vectors for Twitter with 200 dimensions.

The initial layer converts the text into embeddings and then passes these through an LSTM layer. The output from these is passed through a dropout layer to prevent overfitting and then passed through a linear layer. Finally, the results are passed through a softmax layer to get the log probabilities.

We trained a deep-learning model for each emotion using an Adam optimizer over several hidden state sizes- from 25 to 200 at increments of 25, learning rates- $1e-1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6$,and dropout probabilities - $0.1, 0.2, 0.3, 0.4, 0.5$.

We realized that a learning rate of $1e-3$ gave the best results for all emotions.

5.2.2 Bidirectional LSTM

The LSTM-based model is biased towards the end part of the sentence more than the starting of a sentence. In order to prevent this bias, we developed bidirectional LSTM models for each emotion.

The training of this model remains similar to the previously defined section except we changed the hidden state sizes to 32, 64, 96, 128.

	Learning Rate	Dropout	Hidden state size
Anger LSTM	1e-3	0.1	75
Fear LSTM	1e-3	0.2	50
Sadness LSTM	1e-3	0.3	100
Joy LSTM	1e-3	0.5	50
Anger BiLSTM	1e-3	0.3	96
Fear BiLSTM	1e-3	0.3	96
Sadness BiLSTM	1e-3	0.2	128
Joy BiLSTM	1e-3	0.1	128

Table 2: Hyper-parameters used for LSTMs

5.3 Results

The results of our models compared with the baseline (SeerNet) is shown below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
SeerNet (Baseline)	0.695	0.547	0.669	0.503
Uncased BERT	**0.676**	0.493	**0.660**	0.435
Cased BERT	0.634	0.486	0.620	0.434
Ordinal BERT	0.638	**0.505**	0.633	**0.438**
Hybrid BERT	0.493	0.382	0.470	0.313
LSTM	0.537	0.397	0.530	0.339
Bidirectional LSTM	0.518	0.357	0.495	0.303

Table 3: Overall macro-averages

The results we achieved for each emotion while training 4 uncased BERT models are shown below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.697	0.494	0.663	0.465
Fear	0.644	0.414	0.637	0.321
Sadness	0.673	0.505	0.669	0.444
Joy	0.696	0.558	0.672	0.510

Table 4: Uncased BERT classifier

The results when we trained case sensitive models are seen below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.655	0.459	0.624	0.444
Fear	0.603	0.485	0.602	0.380
Sadness	0.625	0.445	0.610	0.396
Joy	0.655	0.555	0.645	0.514

Table 5: Cased BERT classifier

The results when we trained ordinal classifier models are seen below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.573	0.404	0.562	0.268
Fear	0.644	0.467	0.640	0.364
Sadness	0.665	0.482	0.665	0.456
Joy	0.669	0.669	0.665	0.665

Table 6: Ordinal BERT classifier

The results when we trained hybrid model are seen below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.546	0.342	0.544	0.0.297
Fear	0.372	0.241	0.345	0.210
Sadness	0.551	0.411	0.549	0.364
Joy	0.501	0.533	0.442	0.380

Table 7: BERT Hybrid Model

The results when we trained LSTM models are seen below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.614	0.368	0.605	0.348
Fear	0.455	0.345	0.453	0.241
Sadness	0.528	0.422	0.520	0.349
Joy	0.552	0.453	0.542	0.417

Table 8: LSTM classifier

The results when we trained Bidirectional LSTM models are seen below.

	Pearson	Pearson(SE)	Kappa	Kappa(SE)
Anger	0.533	0.298	0.477	0.266
Fear	0.450	0.353	0.449	0.250
Sadness	0.536	0.323	0.513	0.277
Joy	0.553	0.453	0.542	0.417

Table 9: Bidirectional LSTM

The model which gives the highest Pearson correlation coefficient are the uncased BERT models trained.

5.4 Comparative Analyses

Based on the results shown in section 5.3 we were able to identify some factors that contributed to how each model performed:

5.4.1 BERT Cased VS Uncased models

The BERT un-cased model achieved a higher Pearson Correlation coefficient (approximately 12%)than the BERT cased model for all emotions. The BERT Cased classifier contains more tokens compared to BERT Uncased classifiers. In addition, BERT was trained on cleaner data than tweets. Tweets are messier and the use of capitalization is different from the books dataset and Wikipedia dataset that BERT was trained on. For example, the word "bOy" in a tweet would result in an $< UNK >$ token in BERT. This will result in a lot of data loss.

In the diagram below, the attention between an emotion and a sample tweet is shown below

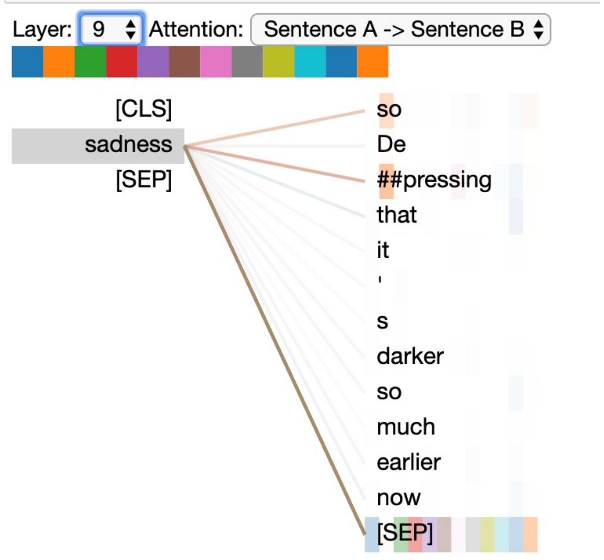


Figure 13: Attention in Cased Model for Sadness

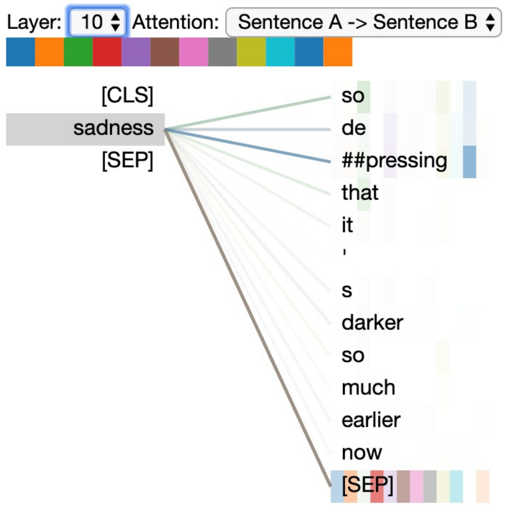


Figure 14: Attention in Uncased Model for Sadness

The diagram above shows the attention patterns between the word anger and a sample tweet collected. This diagram was visualized using (Vig, 2019)

5.4.2 Classification BERT vs Ordinal BERT

We can see that all 4 metrics in ordinal BERT are higher than cased BERT. This could be due to the same issues as the tweet data that we mentioned in the previous part. Compared to the uncased BERT.

5.4.3 BERT Uncased vs Hybrid-BERT

The BERT Hybrid model has all the 4 emotions in a single model. Therefore, the attention it provides to each emotion is weaker compared to a model that is trained on a model just for that emotion.

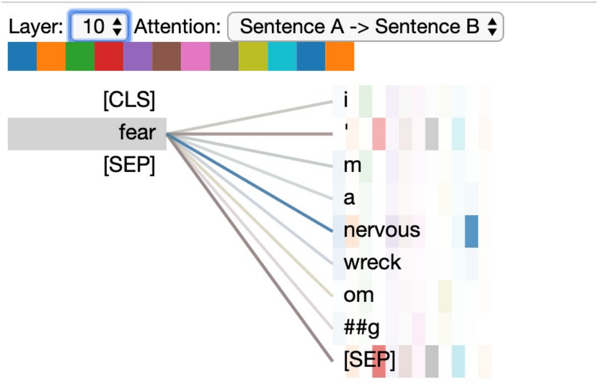


Figure 15: Attention in Uncased Model for Fear

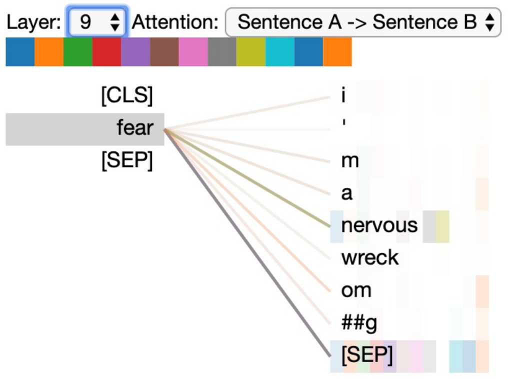


Figure 16: Attention in Hybrid Model for Fear

Therefore, while it prevents training multiple models, it suffers in the performance.

5.4.4 LSTMs vs BiLSTMs

Most of the tweets contain the emojis and hashtags towards the end. These words seem to have a higher impact on the overall performance compared to other words. As LSTMs are biased towards the words at the end of a sentence, the emojis and hashtags are given more importance, thereby giving a slightly better performing model.

5.4.5 Comparison to the baseline

SeerNet (Baseline) made use of many feature vectors such as DeepMoji and EmoInt which is robust to noise in emojis while extracting embeddings and various lexical features in the preprocessing step which gave significant improvements to the model. The ensemble models they used helped improve the overall performance by minimizing errors due to noise, bias, and variance all of which are responsible for the model performance in machine learning.

BERT was able to give a comparable result to the baseline. When looking at the BERT model, the initial attention layers were predicting the relationship between previous and next words, and the later layers were trying to identify similar words. The BERT model provided us multiple features which when passed through the classification layer, gave a comparable result to the baseline.

However, it did not achieve better results than the baseline. The main drawback in our model was the fact that BERT was trained on Books and Wikipedia corpus. The data is very different to what we see in tweets, and there are many words our model was unable to recognize and replaced with ¡UNK¿ token.

The LSTM-based models, however, were not able to produce adequate results. Using GloVe embeddings limited us to only extract the contextual information. The LSTMs and GloVe were unable to extract the various semantic, lexical and sentiment information which was extracted from SeerNet.

6 Future Work

We can clearly see that BERT models perform much better than LSTM-based deep learning models. However, while BERT performs well with messy tweets, if further cleaning steps are added to the tweets, it may improve the performance of the model. For instance, by adding a spelling correction to the tweets before passing them to BERT, we can reduce the number of $< UNK >$ tokens. By using a larger BERT model than the base model we used, the performance may improve.

References

Venkatesh Duppada, Royal Jain, and Sushant Hiray. 2018. Seernet at semeval-2018 task 1: Domain adaptation for affect in tweets. *CoRR*, abs/1804.06137.

Eibe Frank and Mark Hall. 2001. A simple approach to ordinal classification. In *Machine Learning: ECML 2001*, pages 145–156, Berlin, Heidelberg. Springer Berlin Heidelberg.

Michael Handelman. 2019. Text classification with bert, tpus, and engineered features on google colab.

Himanshu. 2018. Sentiment analysis — torchtext.

Chris Mccormick and Nick Ryan. 2019. Bert fine-tuning tutorial with pytorch.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Clara Vania. 2018. Lstm classification using pytorch.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*.