ENGR-E 533 - Deep Learning Systems - Project Report
Emotion Recognition using BERT and its variants

Harsh Atha
Indiana University Bloomington
hatha@iu.edu

Rohit Gampa
Indiana University Bloomington
rgampa@iu.edu

Shreyas Vaidya
Indiana University Bloomington
shrevaid@iu.edu

1 Abstract

Emotions play a crucial role in our day-to-day lives, have an impact on our mental and physical health, as well as our ability to make decisions. Thus, identifying emotions from textual data by creating emotion detection models is crucial, as they may be used for a variety of purposes, such as creating emotional-aware virtual assistants or identifying the emotions of social media users to ascertain their mental and/or physical well-being. They also have significant influence in practical applications ranging from author profiling to consumer analysis. The majority of ER techniques today categorize emotions separately without taking into account the possibility of co-existing emotions. Such methods fail to account for situations where many emotions may overlap. Furthermore, emotion recognition has also remained a difficult endeavor because of a dearth of labeled datasets.

In our project, we focussed on multi-label emotion classification by using deep learning techniques and language models. We mainly considered using CNN and existing popular language model BERT and its variants namely, BERT base, BERT large and DistilBERT, with the primal focus being analyzing their performance and not creation of state of the art models from scratch. The efficacy of our methodology is demonstrated by experiments carried out on two different datasets and also with hyper parameter tuning of these models.

2 Introduction

Emotion is described as a "powerful feeling originating from one's circumstances, mood, or interactions with others", this being the literal definition from Oxford English Dictionary. Emotions often involve something innate that is frequently evoked in social encounters and facilitates communication. Humans are unequivocally emotional creatures: Emotions play a crucial role in human existence and have an impact in our day-to-day lives, our mental and physical well-being, as well as our ability to make decisions. Consequently, it is crucial to create emotion detection models since they have a wide range of uses and social narratives. Emotion classification or recognition is task of identifying emotions from natural language data such as reviews, blogs, news articles, and so on[1]. Some of the example applications of emotion recognition are s follows:

- Emotional-aware virtual assistants or chatbots
- Social Media sentiment analysis
- Consumer Analysis
- Author profiling

These numerous applications have triggered many researchers to work in the domain resulting in a myriad of NLP models and approaches. They focus mainly on positive and negative sentiment analysis, single label emotion classification as well as some focussing on multi-label emotion classification. This stated, however, there exist key challenges while classifying emotions, as follows.

First and foremost, the majority of multi-label emotion techniques currently in use do not successfully capture emotion-specific linkages, which can be helpful for prediction and for learning the relationships between words in a phrase and various emotion labels. Standard methods for classifying emotions tend to deal with each emotion separately. However, emotions are not autonomous; a particular emotive expression might be connected to a variety of emotions. In psychological theories of emotions, the idea of mixed and contrastive emotions, the existence of association and correlation among emotions in a single phrase has been thoroughly discussed.

To illustrate this, let's consider following sentences

1. **Text:** iphones are amazing but since last update, my phone's power drains quickly.
   Emotion: disgust, joy, sadness
2. **Text:** What's plan for tomorrow? I am excited!
   Emotion: joy, love, optimism
3. **Text:** I am using Dell Computer.
   Emotion: neutral

Now, if we look at the first sentence, it consists of a combination of good and negative feelings but leans more towards negative. The labels provided to this are a mix of positive and negative emotions. For instance, words like "amazing" that serve as clues are more likely to be connected to positive emotions than terms like "power drains." DL and language models can predict the proper labels by learning such correlations between emotion labels and the words in a phrase. Moreover, the second sentence emphasizes that some emotions are more likely to be connected than others such as joy often comes with love.

Secondly, there is limited availability of labeled data. As we saw, emotions are not only subjective but also fuzzy, with multiple emotions occurring at once. Different emotion models include various numbers and types of emotion categories. As a result, the creation of resources connected to emotions, such as training data, has been constrained to a few manually annotated datasets or lexicons, a labor-intensive and expensive procedure. The recent success of word embeddings has drawn more attention to the design of emotion classification systems, helping to address the issue of insufficient training data. Each word in the vocabulary V is translated into a dense, low-dimensional, continuous-valued vector in word embeddings, which are distributed word representations. The fundamental concept is that words that commonly appear together in comparable situations are assigned to areas of the vector space that are similar to one another.

Lastly, there are a wide range of Deep Learning Systems and Language models used to perform emotion classification, ranging from recurrent and convoluted neural networks, to transformer and attention based language models. These models have a variety of use cases, thus often raising a challenging task of utilizing appropriate technique for a given task. This can be addressed by carrying out exhaustive experiments with a variety of domain specific datasets and combination of techniques.

The major objective of our work is to carry out an experimental research on the performance of language models in classifying emotions on different datasets. We have considered two datasets, Twitter and Wikipedia, annotated with multiple emotion labels. We primarily focus on evaluating how the performance of BERT and its variants, varies with the size and complexity of these datasets along with hyper parameter effects on the accuracies of the same.

The rest of this paper is organized as follows. Section 3 introduces the related work. Section 4 describes the proposed method, followed by the experimental setup and results in Section 5 and Section 6 respectively. Finally, Section 7 concludes the paper.

3 Related Work

There is a lot of NLP material on recognizing emotions. For example, Bravo-Marquez et al. (2016)[4] suggest a way for expanding it for the language used in Twitter. Earlier studies concentrated on lexicon-based approaches, which employ particular words and their accompanying labels to determine emotions in text such as NRC (Mohammad and Turney, 2013)[2] and EmoSenticNet 1581 (Poria et al., 2014)[3]. Other approaches (Bostan and Klinger, 2018[5]; Liew et al., 2016[6]; Wang et al., 2012[7]) treat the emotion recognition task as a supervised learning task in which a learner (e.g., linear classifier based methods) is trained on the features of labelled data to classify inputs into one label.

A number of neural network models have more recently been created for this job, yielding competitive results on various emotion data sets. Some of these models (Islam et al., 2019[8]; Xia and Ding, 2019[9]) generally concentrate on a single-label emotion classification, in which only a single label is assigned to each input. There are further models for multilabel emotion categorization that assign one or more labels to each input.

Our work mainly focuses on the multilabel emotion recognition model proposed in Hassan Alhuzali and Sophia Ananiadou et al. (2021)[1].They propose a new model called "SpanEmo" that casts multi-label emotion categorization as span-prediction. It includes a loss function that models the coexistence of multiple emotions in the input text. The experiment with SemEval2018 multilabel emotion data across three language sets (i.e., English, Arabic, and Spanish). In addition to replicating this, we experimented with two different datasets to analyse their effectiveness and tried another model with hyper parameter tuning.

4 Method

The primary focus of this work is based on three methods as follows:

1. **Convolutional Neural Networks:**

   (a) A CNN is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other.

   (b) The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex.

   (c) A CNN is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters.

   (d) Figure 1 shows a representation of a CNN which comprises of Convolution and Pooling layers.

   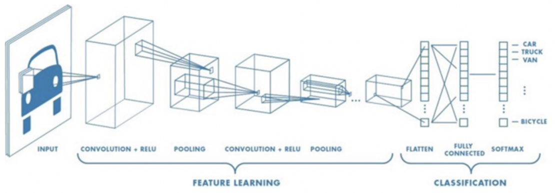

   Fig 1: Convolutional Neural Network

   While this is primarily used for Classification problems in Image datasets, we have converted our sentences to word embeddings and we pass those to the convolution layers to extract features and encode semantic meaning of words and sentences. Like an Image is classified into its respective class, the CNN will identify probability of each label via softmax function and if the probability is above 0.5, it will mark the respective column as 1. We used CNN as basic benchmark deep learning method.

2. **Bidirectional Encoder Representations from Transformers (BERT):**

   (a) BERT is a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. This process is called attention.

   (b) Historically, language models could only read text input sequentially, either left-to-right or right-to-left, but couldn't do both at the same time. BERT is different because it is designed to read in both directions at once. For a set of N examples and C classes, both the label set and input sentence are passed into the encoder BERT. This encoder received 2 segments: classes and sentences and the label set was created as follows:

   $$H_i = Encoder([CLS] + |C| + [SEP] + s_i)$$

   where, [CLS] and [SEP] are special tokens and | C | denotes the size of classes. One advantage we get out of feeding both the classes and text into encoder is that it can learn association between classes and the sentences directly.

   (c) The need for using a BERT pre-trained model is because it is already pre-trained from Wikipedia data and fine tuned further. The time taken to train the original BERT by Google was around 4 days using 64 TPUs. Figure 2 represents the architecture of 1 encoder.

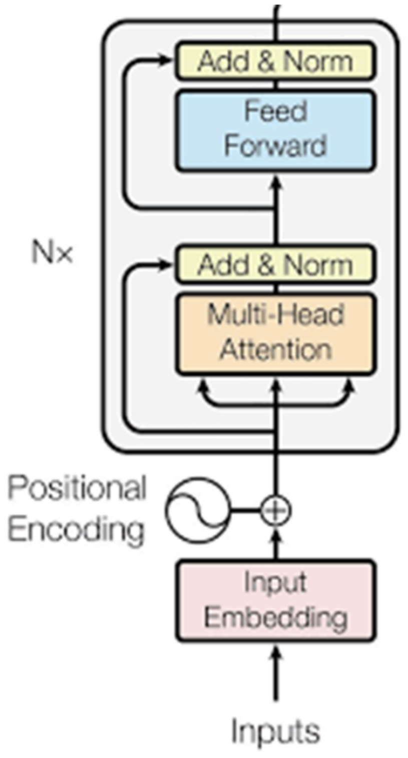

Fig 2: BERT

3. **DistilBERT:**
   (a) One of the challenges in training BERT model is the computational resources and time involved. The number of parameters involved in BERT base are 110 million. In order to cut down this we use DistilBERT, which is using the concept of distillation on the original BERT architecture.

   (b) This model uses 66 million parameters while retaining 97% of the performance of the base BERT model. Reducing the number of parameters has an impact on the performance metrics and the goal was to test how it affected the metrics and time taken to train our model.

   (c) The difference between both our methods is just the pre-trained weights are different in this scenario.

5 Experimental Setup

5.1 Data

1. **Affect in Tweets Dataset:** This dataset consists of user IDs, tweets, and 11 labels which are one hot encoded in form of 0s and 1s, where 1 means emotion is present. This was used in Part 5 of SemEval 2018 Task 1, where the challenge was to identify presence of an emotion in the tweet.

2. **Jigsaw Toxic Comment Classification:** This dataset consists of Wikipedia comments and have been labelled by human users for toxic behavior.

5.2 Data Pre-processing

In order to train the datasets by different kinds of models, they need to be pre-processed to check for any discrepancies and/or if anything needs to be cleaned or imputed. The following pre-process routines were implemented in both the datasets:

1. **Affect in Tweets Dataset:**
   (a) For CNN part of the training, the dataset was cleaned first using regex to remove special characters. We then used GloVe (Global Vectors) 6B 100 embeddings on top of that to create a distributed word representation. GloVe is achieved by mapping words into a meaningful space where the distance between words is related to semantic similarity.

   (b) For the Transformer based approach, we used the TextPreProcessor module of ekphrasis preprocessor, which a tool designed for the specific characteristics of Twitter, i.e., misspellings and abbreviations (Baziotis et al., 2017). The tool offers different functionalities, such as tokenization, normalization, spelling correction, and segmentation. We used the tool to tokenize the text, convert words to lower case, normalize user mentions, URLs and repeated characters. Tokenizers which are part of inbuilt Transformer based libraries were used based on the model chosen.

2. **Jigsaw Toxic Comment Classification:**
   (a) Similar to Tweet dataset, for the CNN approach, the data was cleaned for any special characters and word embeddings were created out of the cleaned data using GloVe 6B 100.

   (b) For the Transformer based approach, there were some challenges tokenizing the data as we are tokenizing the classes as well. In this case the labels such as severe_toxic and identity_hate were treated as separate tokens by the default library, so we tweaked the labels of toxic to hurt, severe_toxic to toxic and identity_hate to hate. After that we were able to use the labels as required. This dataset did not have a validation dataset, so we created one by splitting the train dataset into a 80:20 ratio for train and validation respectively.

   (c) We have sampled 40% of the dataset in order to test the effects of a large dataset. Initially using a P100 GPU it took more than 1.2 hrs to train 50% of the dataset which made training difficult as course credits allocated were not sufficient to finish the training. Hence, we switched to a V100 GPU which is faster and helped us converge using 40% of the dataset.

5.3 Model Training

We used PyTorch and a Tesla P4 GPU with 4 cores of 16GB each on Google Cloud Vertex AI Jupyter Notebook to run our experiments.

1. **CNN:**
   - To find a baseline value to understand the performance of complex models, a simple CNN was used.

- For the CNN based model we set a Max Sequence length of 128 in Twitter data and 750 in Wikipedia data. This max sequence length was used as input shape for the CNN.
- Followed by this were 5 layers of Convolution, Dropout and Max Pooling Layers. The Convolution layers were passed with filter size equal to Max sequence length and kernel size of 5. The dropout ratio was 0.2 and maxpooling of 35.
- After these layers, we flatten the respective output, add a dense layer of num_classes and a softmax activation function.
- For the other layers the activation function used was relu. The optimizer used here was adadelta and a categorical cross entropy loss to compile the model.
- We used a batch size of 128 and learning rate of $1 * e^{-4}$ to fit the model.
- As the training and testing metrics of this approach showed poor results and the data/code had to be tweaked for the other BERT based approaches, we have only added results to the table and removed the code aspect of the training for CNN.

2. **BERT:**
   - For this approach, we include a feature dimension of 786, a batch size of 32, a dropout rate of 0.1, an early stop patience of 4 and 20 epochs.
   - For BERT part of training we use a learning rate of $2 * e^{-5}$ and for the feedforward network a learning rate of $1 * e^{-4}$ with Adam optimizer.
   - The loss function used was cross-entropy loss.
   - BERT in itself has 2 models, base and large. The difference here is the number of encoder layers. Base model has 12 layers while large has 24 layers stacked on top of each other. We are training our model with both of these types.
   - One point to be noted here is that of transfer learning. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. As we already have pre-trained weights of BERT, using that as a starting point is a great advantage and cuts down on training time.
   - Once we have BERT encodings we further feed this into a feed-forward network consisting of a non-linear hidden layer with a Tanh function f and a position vector p, which was used to compute a dot product between f and p.
   - We also add sigmoid activation as our task is predicting multiple classes. We use p in a way similar to that of transformer models using start and end vectors.
3. **DistilBERT:**
   - In order to ensure the results can be compared with BERT we used the same parameters as above for this model as well as the other variants of BERT.

6 Results

We are using 4 major metrics to evaluate our models as follows

- **F1 Micro:** Micro averaging computes a global average F1 score by counting the sums of the True Positives (TP), False Negatives (FN), and False Positives (FP). Micro-averaging essentially computes the proportion of correctly classified observations out of all observations.
- **F1 Macro:** Macro averaging is perhaps the most straightforward among the numerous averaging methods. The macro-averaged F1 score (or macro F1 score) is computed using the arithmetic mean (aka unweighted mean) of all the per-class F1 scores.
- **Jaccard Similarity:** The Jaccard Index, also known as the Jaccard similarity coefficient, is a statistic used in understanding the similarities between sample sets. We apply the Jaccard Index measurements as a way of conceptualizing accuracy of classification.
- **Time to Train:** The resources consumed too are an important parameter in the training. If the training time for each model is largely varying but the F1 scores and JS are matching, it is better to consider the smaller model as we can train our model using lower resources.

As a result of the successful implementation of all variations of the above-mentioned metrics, we were able to generate the results shown in Table 1 and Table 2.

Sr. No.	Dataset	Model	F1 Micro	F1 Macro	Jaccard Similarity	Time to train/epoch (seconds)	No. of Epochs	Total Time (hh:mm:ss)
1	Twitter 2018	CNN	0.4892	0.3716	0.5106	30	29	00:14:30
2	Twitter 2018	Bert base	0.6999	0.5897	0.581	176	15	00:44:00
3	Twitter 2018	BERT large	0.7054	0.5719	0.5806	159	8	00:21:12
4	Twitter 2018	DistilBERT	0.7054	0.5636	0.5848	90	10	00:15:00
5	Wikipedia	BERT base	0.7749	0.6562	0.0605	1691	9	04:13:39
6	Wikipedia	DistilBERT	0.7822	0.6576	0.0652	811	9	02:01:39

Table 1: Training Results

Sr. No.	Dataset	Model	F1 Micro	F1 Macro	Jaccard Similarity	Total Time (seconds)
1	Twitter 2018	CNN	0.4762	0.3923	0.5009	3
2	Twitter 2018	Bert base	0.7071	0.5428	0.5884	7
3	Twitter 2018	BERT large	0.703	0.5297	0.5774	23
4	Twitter 2018	DistilBERT	0.6932	0.523	0.5657	4
5	Wikipedia	BERT base	0.6203	0.5577	0.0691	4
6	Wikipedia	DistilBERT	0.6659	0.5453	0.0656	2

Table 2: Test Results

- Based on results, we can see that the scores for CNN are the least in all categories and it performs inadequately when using multi-label classification in this dataset. Typically, BERT models have comparable scores in all 3 cases for Twitter data.

- The training time for BERT large is smaller than that of the base model and this can be due to the good starting point provided by already highly trained BERT large.

- DistilBERT, which is the smallest of them takes the least amount of time to converge. Even the results generated by them are in range of 1-2% of each other.

- Firthermore, We can see from the Wikipedia dataset results that the F-1 macro and micro scores are much better compared to the Twitter dataset, but jaccard is performing poorly.

- The training for Wikipedia dataset on Bert-large is not running due to CUDA out of memory issue. We have tried it after changing the GPU from Nvidia Tesla P4 to V100, even then we have the same error. We have also tried to decrease the batch size from 32 to 16,8 and train it, still the error won't go away. Thus, we have skipped the results for it.

7 Conclusion

- Based on our experiments, we observe that CNN is not an appropriate model for training the multi-label classification dataset when it comes to word embeddings for a small dataset. BERT based models show an increment of almost 20-30 % in the all categories and are recommended for the usage.

- There is a tradeoff between training time and the metrics generated. Comparing overall results, if training time and resources are the parameter to be considered DistilBERT is the best model to be chosen as it gives comparable metrics to the other two.

- However, in case of test data BERT base outshines the other two models and gives the best results. That is, If we only focus on the metrics such as accuracy, precision and recall (F1 score micro, macro) BERT base is the model to be chosen.

- It can also be concluded that the size of the model in this case does not have a major impact on the results. Since this is the case, the model with smaller size(DistillBERT) can be used in a edge setting where there are not lot of storage space, and when we need to learn quickly and have fast convergence we can use

BERT-large, but in most scenarios BERT base will be used.

- From the low Jaccard score and high F-1 scores in Wikipedia dataset, we can conclude that, the model is able to identify individual emotions very well, but not able to detect multiple emotions at the same time. In future to solve this we can try to train the model with the dataset for a longer time and also try to modify the loss function to include Jaccard.

References

[1] Hassan Alhuzali Sophia Ananiadou, 2017. SpanEmo: Casting Multi-label Emotion Classification as Span-prediction.

[2] Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. 29(3):436–465.

[3] Soujanya Poria, Alexander Gelbukh, Erik Cambria, Amir Hussain, and Guang-Bin Huang. 2014. Emosenticspace: A novel framework for affective common-sense reasoning. Knowledge-Based Systems, 69:108–123.

[4] Felipe Bravo-Marquez, Eibe Frank, Saif M Mohammad, and Bernhard Pfahringer. 2016. Determining word-emotion associations from tweets by multilabel classification. In 2016 IEEE WIC/ACM International Conference on Web Intelligence (WI), pages 536–539. IEEE.

[5] Laura-Ana-Maria Bostan and Roman Klinger. 2018. An Analysis of Annotated Corpora for Emotion Classification in Text. In Proceedings of the 27th International Conference on Computational Linguistics, pages 2104–2119.

[6] Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In Empirical Methods in Natural Language Processing, pages 3687–3697.

[7] Wenbo Wang, Lu Chen, Krishnaprasad Thirunarayan, and Amit P Sheth. 2012. Harnessing twitter" big data" for automatic emotion identification. In Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom), pages 587–592. IEEE.

[8] Jumayel Islam, Robert E. Mercer, and Lu Xiao. 2019. Multi-channel convolutional neural network for twitter emotion and sentiment recognition. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 1583 (Long and Short Papers), pages 1355–1365, Minneapolis, Minnesota. Association for Computational Linguistics.

[9] Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1003–1012, Florence, Italy. Association for Computational Linguistics.