Logo, company name

Description automatically generated

Emotion Detection From Text

# INTRODUCTION

Natural Language Processing is now thought of as a technique for computers to process human language naturally because of technological advancements [1]. Language to language translation, sentiment analysis, and text-emotion recognition are just a few of the areas where NLP is used [2]. However, since recognizing people's emotions could be a challenging task for people, computers are much more useful in this regard. We'll be able to demonstrate one of the many available approaches to sentiment analysis via the course of this project. Use the enter key to start a new paragraph. The appropriate spacing and indent are automatically applied.

# PREPROCESSING

The dataset is the go emotions dataset provided freely by Google which contains 58k Reddit comments. The dataset was divided into three parts; training set, validation set and testing set. They were found to be free from null values. The tweets were all converted to lowercases thereby facilitating the processing, post to which punctuations were removed as they are prone to induce the models to errors. Stopping words removal was also done prior to checking the most frequent words and applying a spelling correction. The goal of using a spelling corrector was to help speed up the model phases as they will spend less time trying to first correct them before the processing is done.

Tokenization was done prior to label encoding which helps convert these sentences to binary representations that will be made easier for the model to understand. Stemming helped reduce the tokenized words to their respective base forms. For example running will be changed to run. The word2vec was pretrained to help vectorize all the encoded words. The vocabulary table will help our model interpret the emotions behind the texts.

# MODELS AND RESULTS

1. CNN-LSTM:

In the CNN-LSTM, Adam optimizer was used which consumes the most efficient computer resources, the model was evaluated using loss and accuracy matrices, and the validation loss and validation accuracy were also measured. In addition, the loss was measured using binary cross entropy. Moreover, the callback have patience of 5 which means that the model will stop training if the validation loss didn't change for 5 epochs. The model is Conv1D that consists of 6 dense layers, with Relu activation function for the convolutional layer and sigmoid function for the output layer. Also, the number of epochs was set to 10(for storage purposes) and the batch size was 156.

1. BERT:

We optimized the model using Adam optimizer, with learning rate equals 5e-5, we used Categorical Cross entropy as the loss function. The model layer:

1. Input layer: input\_ids and attention\_mask are the input tensors to the model, representing tokenized input text and mask for padding.
2. BERT layer: bert\_model is a pre-trained BERT model used to encode the input text into contextualized embeddings.
3. GlobalMaxPool1D layer: Performs global max pooling on the output of BERT embeddings, reducing dimensionality.
4. Dense layer (units=128, activation='relu'): Fully connected layer with 128 units and ReLU activation function.
5. Dropout layer (rate=0.1): Regularization layer that randomly sets 10% of input units to 0 during training to prevent overfitting.
6. Dense layer (units=64, activation='relu'): Fully connected layer with 64 units and ReLU activation function.
7. Dense layer (units=32, activation='relu'): Fully connected layer with 32 units and ReLU activation function.
8. Dense layer (units=6, activation='softmax'): Output layer with 6 units and softmax activation function, providing the predicted class probabilities.
9. Model layer: Defines the input and output of the model and specifies that the BERT layer is trainable during fine-tuning.

We used CNN-LSTM and Bert models whose results can be seen below.

Chart, bar chart

Description automatically generated

Figure 1: Comparison in models’ accuracies

As it can be noticed, CNN-LSTM is the successful model in this case because it was able to identify a lot of the testing set data with approximately 0.925 accuracy. This could be because of CNN’s speed in training time [3]. The code can be accessed from :

<https://colab.research.google.com/drive/1JVedrf-iuv-wpa8gcvLXg9DKppHD9Vif?usp=sharing>

# critical analysis

Model Complexity: The BERT model has 340 million or more trainable parameters, making it one of the largest and most complicated models ever created. In contrast to BERT, CNN-LSTM is a much easier model to understand. Due to overfitting on the training set caused by the high complexity of BERT, training set accuracy may suffer, while test set generalization may improve.

##### References

1. R. Gruetzemacher, “The power of Natural Language Processing,” Harvard Business Review, 19-Apr-2022. [Online]. Available: <https://hbr.org/2022/04/the-power-of-natural-language-processing>. [Accessed: 23-Feb-2023].
2. “Natural language processing (NLP): The history and applications,” Genzai, 23-Dec-2022. [Online]. Available: <https://www.genzai.nl/algemeen/natural-language-processing-nlp-the-history-and-applications/> . [Accessed: 23-Feb-2023].
3. “A Quick Dive into Deep Learning: From Neural Cells to BERT,” *Alibaba Cloud Community*. [https://www.alibabacloud.com/blog/a-quick-dive-into-deep-learning-from-neural-cells-to-bert\_595508](https://www.alibabacloud.com/blog/a-quick-dive-into-deep-learning-from-neural-cells-to-bert_595508%20%20)  (accessed Apr. 21, 2023).

**Make sure to remove all placeholder and explanatory text from the template when you add your own text. This text should not be here in the final version!**