Chapter 1

Experiment Design

This chapter details the experimental setup for carrying out the task of classifying emotion from text. In particular, the chapter outlines the text pre-processing steps, the model architecture, evaluation metrics and technologies that are made use of.

1.1 Data

This study makes use of both the *ISEAR* (International Survey on Emotion Antecedents And Reactions) and *SemEval-2018 Affect In Tweets* datasets. The ISEAR dataset consists of 3000 responses from people around the world who report situations where they experienced each of the emotions from (joy, fear, anger, sadness, disgust, shame, and guilt). The dataset is split such that 80% is assigned to the training set, 10% to the validation set and the last 10% to the testing set.

1.1.1 Pre-trained Word Embeddings & Language Models

Due to the minimal amount of training data, we make use of the pre-trained GloVe embeddings trained on the Wikipedia and Gigaword corpus *(6B tokens, 400k vocab, uncased, 300d)*, the pre-trained Word2Vec embeddings trained on the Google News corpus *(100B words, 3M vocab, 300d)* as well as the ELMo embeddings trained on the 1 Billion word benchmark available from Tensorflow-Hub. Furthermore, the Huggingface Transformers library for Tensorflow is utilized in order to retrieve the pre-trained BERT model and to compute BERT embeddings.

1.1.2 Pre-processing

Text in which people recall experiences likely consists of various informal words and words which don't have any significant correlation towards the emotions being conveyed. Thus, keeping these words in the text increases the overall feature dimensionality. As

a means to counter this issue, this study makes use of text pre-processing techniques such as stop-word removal, removing punctuations and accent marks, removing white spaces, and likewise to (Howard & Ruder 2018), adding special tokens for upper-case words and repetition as this may allow the model to capture characteristics which are important to classification.

1.2 Models

As shown by the enumerated list below, the overall model architecture is designed such that it contains the following layers:

1. Pre-trained embeddings layer
2. Model layer
3. Dropout layer
4. Classification layer

The embedding layer is initialized using pre-trained Word2Vec, ELMo and BERT embeddings such that the pre-processed input text from the dataset is converted into an embedding representation. Furthermore, the model layer makes use of an RNN, CNN and RCNN. With regards to the RNN, the experiment is carried out using both a unidirectional LSTM as well as a bidirectional LSTM. The RCNN architecture proposed by (Lai et al. 2015) is used in order to take advantage of both the RNN and CNN as a means to further improve the emotion classification. Next, the dropout layer is implemented in order to ensure that the model does not over-fit the data. Lastly, the classification layer makes use of Cross-Entropy loss in order to compute the probabilities of the input text belonging to each of the class labels.

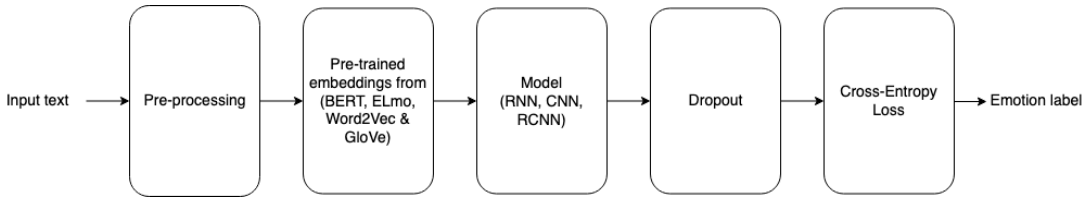

Figure 1.1: Proposed architecture.

Furthermore, we compare the result of the above mentioned Deep Learning models to that of machine learning algorithms such as Logistic Regression. We apply the machine learning algorithms using Term Frequency Inverse Document Frequency (TF-IDF) as a means to convert text into vector form.

1.3 Performance Evaluation

This study evaluates the performance of all models using accuracy and the micro and macro F1 scores. We specifically optimize each model with the micro and macro F1 scores in mind. This is because the dataset consists of labels with more instances than others. Furthermore, the micro and macro F1 scores have been used in various other papers that tackle the task of emotion classification from text (*Papers With Code Emotion Classification* 2019). Therefore, these metrics provide a good comparison for the overall performance of the models implemented in this study.

$$PREmicro = \frac{TP1 + TP2}{TP1 + TP2 + FP1 + FP2} \tag{1.1}$$

$$RECmicro = \frac{TP1 + TP2}{TP1 + TP2 + FN1 + FN2} \tag{1.2}$$

The micro F1 score is equal to the harmonic mean of these two figures (Shams 1970).

$$PREmacro = \frac{PRE1 + PRE2}{2} \tag{1.3}$$

$$RECmacro = \frac{REC1 + REC2}{2} \tag{1.4}$$

The macro F1 score is equal to the harmonic mean of these two figures (Shams 1970).

1.4 Technologies

The experiment makes use of Python along with Tensorflow and Keras for constructing the architecture of the models. Both the Natural Language Toolkit (NLTK), and Keras Tokenizer libraries are used for text pre-processing. The Scikit learn library is used for the implementation of machine learning models.

Bibliography

Howard, J. & Ruder, S. (2018), 'Fine-tuned language models for text classification', *CoRR* **abs/1801.06146**.
URL: *http://arxiv.org/abs/1801.06146*

Lai, S., Xu, L., Liu, K. & Zhao, J. (2015), 'Recurrent convolutional neural networks for text classification'.
URL: *https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9745*

Papers With Code Emotion Classification (2019).
URL: *https://paperswithcode.com/task/emotion-classification*

Shams, R. (1970), 'Micro- and macro-average of precision, recall and f-score'.
URL: *https://rushdishams.blogspot.com/2011/08/micro-and-macro-average-of-precision.html*