Suicide is a worldwide problem that results in almost 800,000 deaths each year. In 2016, 1.4 % of deaths worldwide were caused by suicide (World Health Organisation, n.d.). Suicides also have negative flow on effects for people bereaving the victim as it increases the chance of developing a mental illness such as depression and anxiety, additionally marital breakup is more likely to occur after their child commits suicide (Bolton et al., 2013). Therefore, suicide can cause destruction even after the victim’s death and prevention strategies to detect people at suicide risk is critical. The aim of this study was to explore if natural language processing (NLP) can distinguish between suicidal ideation and non-suicidal ideation text. It is hypothesised that NLP can classify with high accuracy whether text consists of suicide ideation.

**Method**

**Dataset**

The suicide ideation dataset was downloaded from Kaggle and the data consists of suicide posts made on subreddits (Komati, 2021). A text example from the suicide class was “ I need help, just help me I am crying so hard” while an example from the non-suicide class was “ Am I weird I don't get affected by compliments if it's coming from someone, I know but I feel really good when internet strangers do it”. The classes were evenly balanced with 116037 samples in the suicide class and 116037 samples in the non-suicide class. There were also no missing values. Python 3 was used to run the analysis and the main packages used were Pandas, NumPy, Sklearn and Keras.

**Target preparation**

Sklearn label encoder class was used to transform the target variable from text to integer format. The transformation resulted in non-suicide equalled 0 and suicide equalled 1.

**Text preparation**

Keras Tokenizer method was used to transform all words to integers (tokens). The parameters used were to select the 60,000 most common words, convert all words to lower case and set out of vocabulary tokens to <OOV>. The token object texts\_to\_sequences method was then used to convert each text sample into a list of integers that corresponded to words. Post padding was then performed so each sample would be equal to the maximum sample’s length.

**Training and test split**

Sklearn train\_test\_split function was used to split the data: 174055 samples were in the training set while 58019 were in the test set. 87027 (½) samples from the training set were moved to a validation set.

**Neural network model**

Keras Sequential API was used to build the model. The first block’s first layer was an embedding layer with an input size of 60,000 and output size of 30. Batch normalization was then performed followed by a 1D global average pooling layer. The second block’s first layer was a dense layer consisting of 30 nodes and ReLU activation function was used. Batch normalization was then used followed by a dropout layer of 0.5. The third block’s first layer was a dense layer consisting of 15 nodes and ReLU activation function was used. Batch normalization was also used and then followed by a dropout layer of 0.3. The final output layer was a dense layer consisting of 1 node and sigmoid function was used.

**Compiling and fitting the model**

The model was compiled using Adam optimizer with a learning rate of 0.0001, the loss function was binary cross entropy and accuracy was the metric assessed. The model was fitted with the training and validation sets, consisted of 50 epochs and a batch size of 32. Call backs were used while training to reduce learning rate by a factor of 0.9 if validation loss did not improve over 3 epochs and training would stop if validation accuracy did not decrease over 10 epochs.

**Results**

*Table 1*. Natural language processing models training and test set accuracy

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Epochs | Training accuracy | Test accuracy |
| Model 1 | 23 | 95 % | 76% |
| Model 2 | 3 | 91 % | 66 % |

A post-hoc model was also tested with the exact same neural network architecture and same compiling parameters. The only difference when fitting was 3 epochs were used considering model 1 showed overfitting continued to occur after 3 epochs. Model 1 was eventually terminated on epoch 23 due to overfitting. Therefore, 3 epochs were deemed the best value that had a good balance of training and validation accuracy while validation accuracy steadily increased up until epoch 3 and this was seen through multiple trials. As can be seen from table 1, model 1 had training accuracy of 95 % and test accuracy of 76 % while model 2 had training accuracy of 91 % and test accuracy of 66 %.

**Discussion**

The hypothesis that natural language processing can distinguish between suicide ideation and non-suicide ideation text with high accuracy was partially supported. Although model 1 may have not been able to predict with very high accuracy, it still did reasonably well with predicting suicide ideation text considering test set accuracy was 76 %. Model 2 trained with 3 epochs was able to do a modest job at predicting suicide ideation text since it scored 66 % on the test set. Although the limitation of model 1 was it took approximately 5 hours longer to train the model compared to model 2. An increase in 10 % classification accuracy was worth the extra computation time considering the implications could potentially be preventing suicide. Following this thought, the implication of this research shows that natural language processing can be utilised to detect suicide ideation and perhaps lead to quicker interventions to prevent suicide from occurring.

References

Bolton, J. M., Au, W., Leslie, W. D., Martens, P. J., Enns, M. W., Roos, L. L., ... & Sareen, J.

(2013).

Komati, N. (2021). Suicide and depression detection. Retrieved from

<https://www.kaggle.com/nikhileswarkomati/suicide-watch/metadata>

Parents bereaved by offspring suicide: a population-based longitudinal case-control

study. *JAMA psychiatry*, *70*(2), 158-167.

World Health Organization. (n.d.). *Mental health and substance data.* Retrieved from

<https://www.who.int/teams/mental-health-and-substance-use/suicide-data>