NLP 641 Report: Prediction Post

`Goal`

The goal is to develop predictive computational models using features of posts on Reddit to develop and evaluate technology that might be useful to assist in identifying and monitoring mental state, particularly the risk of suicidality. In support of this goal, we will analyze archival data collected online in the University of Maryland Reddit Suicidality Dataset.

`Code`

The folder contains the following major files

1. traning_testing_modeling_part_1_of_3.ipynb
2. pre_processing_unseen_data_part_2_of_3.ipynb
3. sentence_encoder_part_3_of_3.py
4. nlp_helper_functions.py [Not to be run; It will be used by other notebooks and scripts]
5. DataPreprocessing_DL.ipynb - Data Preprocessing for Deep Learning Models
6. utils.py - Utils/Helper python file
7. Train_Transformer.ipynb - RoBERTa trainer and fine tuning
8. Train_Numerical.ipynb - Only Numerical Features Neural Network (Maybe we can ignore this?)
9. Train_Hybrid.ipynb - Our Hybrid Architecture Fine Tuned RoBERTa + Numerical Features

`Data`

Challenges in getting the data:

- **IRB approvals, training,**

Description of the Dataset

- **Task A consists of posts on the Reddit r/suicidewatch channel**
- **There were about ~900 posts, 456 unique user ids**
- **The label was mapped to a userid, not the post**
- The original dataset had four labels, namely a, b, c, and d, from no-risk to increasing order of suicide risk
- The goal of this task was reduced to binary classification.
    - a, b, c is marked as No Risk or label 0 for the rest of the report
    - d is marked as High Risk or label 1 for the rest of the report.

Data Preprocessing

- For User Ids, which had multiple posts, the analysis was aggregated—for example, an average number of words from Joy, Sadness. The same was done for sentence polarity, average post length, and average title length.

- All the text from post_body and title was cleaned and then aggregated by the user for the post itself. (it was joined to make it one doc to one user mapping)

Text Cleaning

- Removed stop words.
- Removed punctuations and special symbols
- Converted to lower-case

> I was at party. When everyone were having fun... i went to smoke... I don't remember much, but it fucking hurt's, but it's nothing compare to pain inside. Most of things i do... is just for other people. I don't want them to be sad, beacuse of me. I almost never talk about my problems and even i must... am just saying its fine. That am okey.. People think am cool person, confident etc... but they are so fucking wrong.Scar is something real... Am glad that i did that... am just opening my hand and see it... It's not in my head... Am witness of my depression... One of few things that are real... is this scar. Am 21 old male... but i feel like 80... Just wanted to share my thoughts.Thank you.. that you are here.

Before Cleaning the sentence.

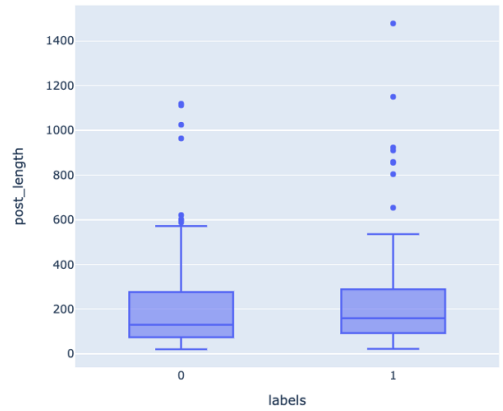
```
'party fun smoke remember fucking hurt compare pain inside things people sad beacuse talk problems fine okey people c
ool person confident fucking wrong scar real glad opening hand head witness depression things real scar male feel wan
ted share thoughts'
```

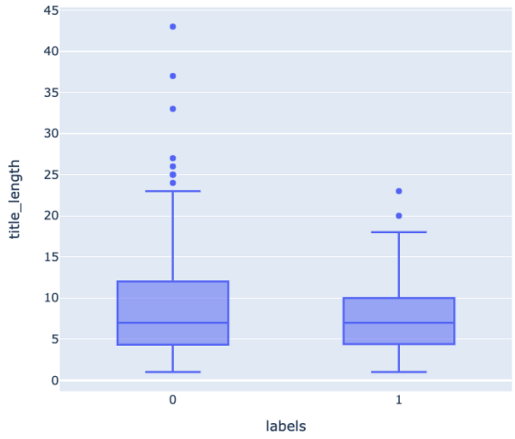
After Cleaning the sentence.

`Feature Engineering`

Length of Posts, Length of Title, Number of Posts by User_id

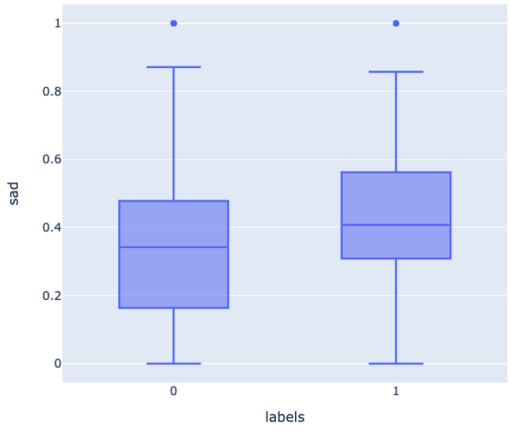
- The average length of text **(not cleaned)** for the post with suicide risk was around 232 words compared to 201 words for posts with no risk.
- We have observed a high standard deviation.
- Interestingly title length was marginally longer for people with no suicide risk. The mean is 9 and 7 words for no-risk and risk, respectively.
- After analyzing, we ***filtered posts that were less than 30 words***





Emotional Lexicon (Joy, Anger, Disgust, Sadness etc.)

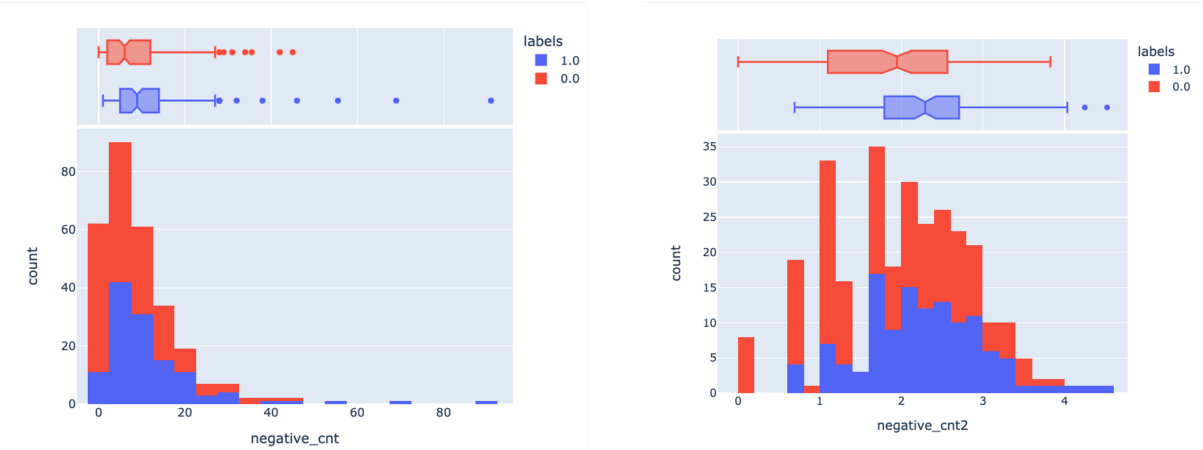
- Every sentence counts the number of lexica based on their category. This word to emotion mapping is taken from Stanford University Lexicon Data
- According to the NHS website, the significant causes of suicide are the following
    - Feeling of hopelessness
    - Anger
    - Depression
    - Sadness
- Individual Emotions were very noisy, so we aggregated Anger, Sadness, Fear, and Disgust as one feature
- We squared the value and used it as the feature
- Performed t-test and got the p-value of 0.012.



Difference in Distribution in the measure of Agregated Sadness Value between thee two labels

Sentence Polarity (Positive, Negative)

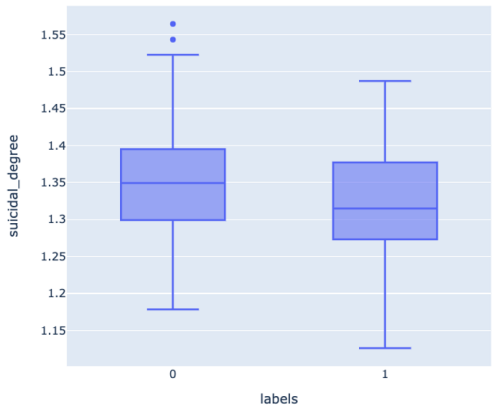
- Every sentence counts the number of lexica based on their category. Positive or Negative. This is taken from Stanford University Lexicon Data.
- Both the features were noisy, so we removed the positive count.
- Trying to measure the negativity in the sentence, After looking at the data, we concluded that negative (normalized) between .42 and .58 makes the model noisy and performed poorly.
- Removed the post from the dataset, which has highly mixed emotions.
- After removing the posts with mixed emotions, we did a log transformation of the values and used them as a feature.
- Performed the t-test and got a p-value below 0.05, indicating the feature is significantly essential to separate the two classes.



Before and After Transformation of Negative Count Feature

Inverse Suicidal Degree (Roberta-Sentence Encoder)

- After analyzing the data and the NHS website for mental health, `we created our custom sentences`.
  - I want to kill myself,
  - I want to die,
  - hopeless,
  - I am depressed
  - I want to commit suicide



- Used Hugging Face Sentence Encoder to encode these sentences.
- Used cosine similarity to match it against each of the titles and assign the maximum value as a feature
- we used the `1/x transformation` for the value computed in the previous step.

Difference in Distribution between the two labels for Inveerse Suicidal Degree Feature

Posts by Hour of the day

- We also tried to explore if the time of the post had any co-relation with the labels but due to lack of information on the timezone and the origin country of the post. Therefore it was not considered as a feature for the analysis.

`Modeling : Traditional Machine Learning`

Benchmark - Random Guess

- Benchmark – 57% labels as no risk and 43% labels as risk
- So random guess of no-risk would be 57% time correct, but it would be horrible for the task at hand, that is, to predict suicide risk.
- Also, If we predict every label as suicide, we would be correct only 42% time which is also not ideal.

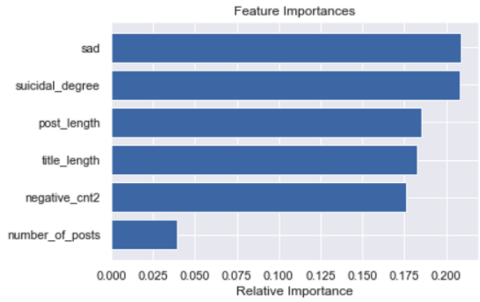
Benchmark - Traditional Machine Learning

- Used the test-train split 30 and 70%, respectively
- The data used here has been carefully filtered to exclude posts with mixed emotion and posts with less than 30 words.
- Used the Standard Scaler to normalize the data
- Use Random Forest Classifier with hyperparameter tuning.
    - bootstrap=True, min_samples_split=5, class_weight={0: 1, 1: 5}

Feature Importance

- Out of the numerical features created, Aggregated Sad Emotion and Suicidal Degree had the most significant impact.

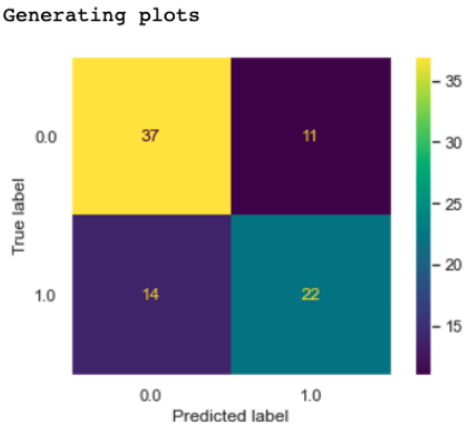
- It indicates that the post's title is also a good indicator of the user's intent.



Metrics

`Test Data`

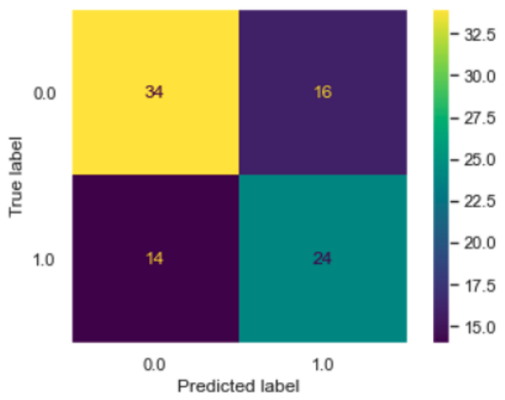
- Recall that label 1 is a critical metric, and out of all the actual suicide risks, the model can identify 67% of them.
- It has also shown good performance in predicting ”no-risk.”
- Although cross-validation mean accuracy is 65% across three splits




```
Accuracy  = 0.7
Precision for label 1 = 0.61
Recall    for label 1 = 0.67
Precision for label 0 = 0.77
Recall    for label 0 = 0.73
```

`Unseen Data` : ***This is tested on the crowd_test data for Task A***

- Recall that label one on Unseen Data is 60%, which is significantly better than the baseline in predicting the risk for suicide.
- Precision is another critical metric 63% of the time, if the model predicted a risk, it was a risk, compared to a baseline of 42%
- It has also shown good performance in predicting ”no-risk.”




```
Accuracy  = 0.66
Precision for label 1 = 0.632
Recall    for label 1 = 0.6
Precision for label 0 = 0.632
Recall    for label 0 = 0.6
```

Cross-Validation

- Created 5-folds , with accuracy going as high as `67%` and out of the 5, all models have results above 60%
- The `mean` accuraccy score from cross-validation is `0.64` and the `standard deviation` is `0.03`

```
Doing cross-validation splitting with stratify=Yes. Showing 10 indexes for items i
n train/test splits in 5 folds.
TRAIN: [0 1 2 3 4 5 6 7 8 9] TEST: [10 15 25 32 33 34 38 61 66 67]
TRAIN: [0 1 2 3 4 5 6 7 8 9] TEST: [14 21 22 24 28 30 37 43 51 63]
TRAIN: [ 1  2  5  6  8 10 13 14 15 16] TEST: [ 0  3  4  7  9 11 12 18 19 26]
TRAIN: [ 0  3  4  5  7  8  9 10 11 12] TEST: [ 1  2  6 13 20 29 35 36 42 47]
TRAIN: [ 0  1  2  3  4  6  7  9 10 11] TEST: [ 5  8 16 17 23 27 39 40 46 53]

Running 5-fold cross-validation on 70.0% of the data, still holding out the rest f
or final testing.
Running Random Forest 5-fold cross-validation on 70.0% of the data, still holding
 out the rest for final testing.
accuracy scores = [0.60714286 0.67857143 0.61818182 0.63636364 0.67272727],
mean = 0.64, stdev = 0.03
```

`Deep Learning Part 1: Glove Embedding Approach (Only Text Features)`

Data Preprocessing

- Aggregated the text by user_id.

```
# configuration
MAX_SEQUENCE_LENGTH = 100
MAX_VOCAB_SIZE = 20000
EMBEDDING_DIM = 300
VALIDATION_SPLIT = 0.2
```

- Performed tokenization and padding to create fixed length vectors.
- Loaded pre-trained vectors from Glove 300 and created the embedding metrix

```
BATCH_SIZE = 32
EPOCHS = 30
```

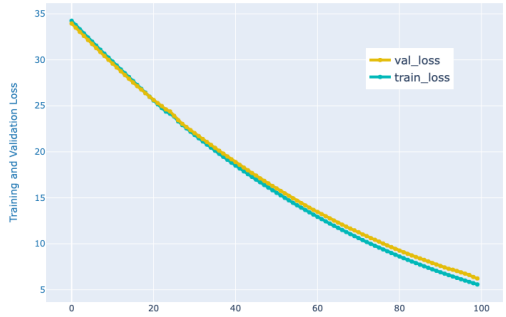
Modeling

- Input going through the embedding layer followed by two BiLSTM Layers and a dense layer.
- dropout of 0.01 and kernal regularization of 0.01 is used to avoud over-fittling.

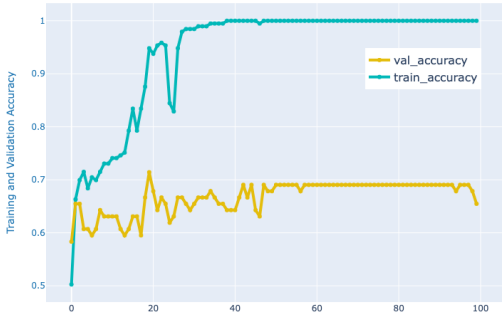
```
#model architecture
input_ = Input(shape=(MAX_SEQUENCE_LENGTH,))
embeddings = embedding_layer(input_)
bilstm_1 = Bidirectional(LSTM(64, return_sequences
=True, return_state=False))(embeddings)
dropout = Dropout(0.1)(bilstm_1)
bilstm_2 = Bidirectional(LSTM(128, return_sequence
s=False, return_state=False, dropout=0.1))
bilstm_layer = bilstm_2(dropout)
dense = Dense(256,kernel_initializer=tf.keras.init
ializers.glorot_normal(seed=None),
    kernel_regularizer=tf.keras.regularizers.l1(0.
01),
    activity_regularizer=tf.keras.regularizers.l2
(0.01), activation='relu')(bilstm_layer)
dense_1 = Dense(1, activation='sigmoid')
output = dense_1(dense)
model_w2v = Model(input_, output)
```

Evaluation

- The model reached a maximum validation accuracy of 71%, but prediction results were not as good as we obtained from the traditional ML Random Forest on the confusion Matrix. Relatively to the Random Forest Model, the High number of “Suicide Risks” was classified as “No Risk,” which is the primary evaluation criteria for this project.

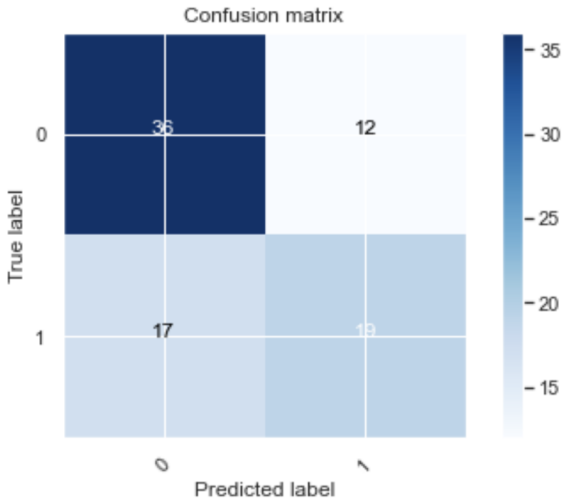


Train and Validation Loss



Train and Validation Accuracy

```
Confusion matrix, without normalization
[[36 12]
 [17 19]]
```



Confusion Matrix for the Glove Embdedding NN model

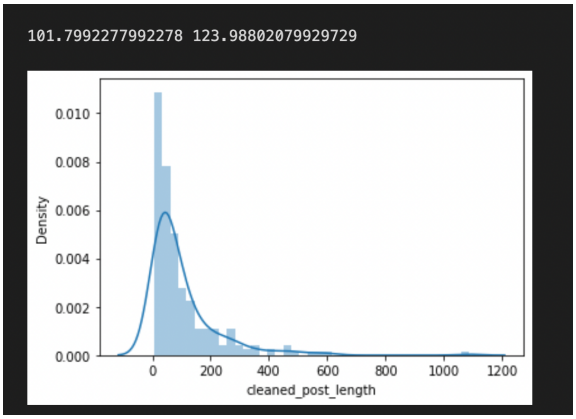
`Deep Learning Approach: Part 2 Transformers Model`

With the advances in transformer models and their capabilities to gain insights from text has made them an almost default go-to model in contemporary times.

With our dataset, we performed multiple experiments to understand the power, usage and limitations of the RoBerta Model.

Word Encodings

- The first step was to obtain encoding for the text, we used the RobertaTokenizer from the transformers module by hugginface on python to extract these. The tokenizer function takes in the parameters: text, …., max_len.
- The max_len represents the length of the embedding to obtain. This parameter was selected after analyzing the cleaned data and deciding to pick the max length to be the mean + 1 standard deviation. This ensures that the encodings are not overpowered by longer texts.
- Max_len = 101+123 = 224.
- The tokenizer can also be given a pre-trained model, we picked ‘roberta-base’.



Roberta Model

Robustly optimized BERT model was used to perform the classification task. The huggingface based transformers library provides a function to use RobertaModel().

Pre Trained Model

- As mentioned with the tokenizer, ‘RoBerta-base’ pre-trained weights were initialized onto the model. These weights are generic and do not specifically cater to the task, i.e., suicide detection.
- The RobertaModel provided by hugging face had a pooled layer output of 768 features at the end. A Dense additional layer resulting in 2 output nodes followed by a Softmax activation was added on top of the Roberta Models extracted pooling layer for classification tasks.

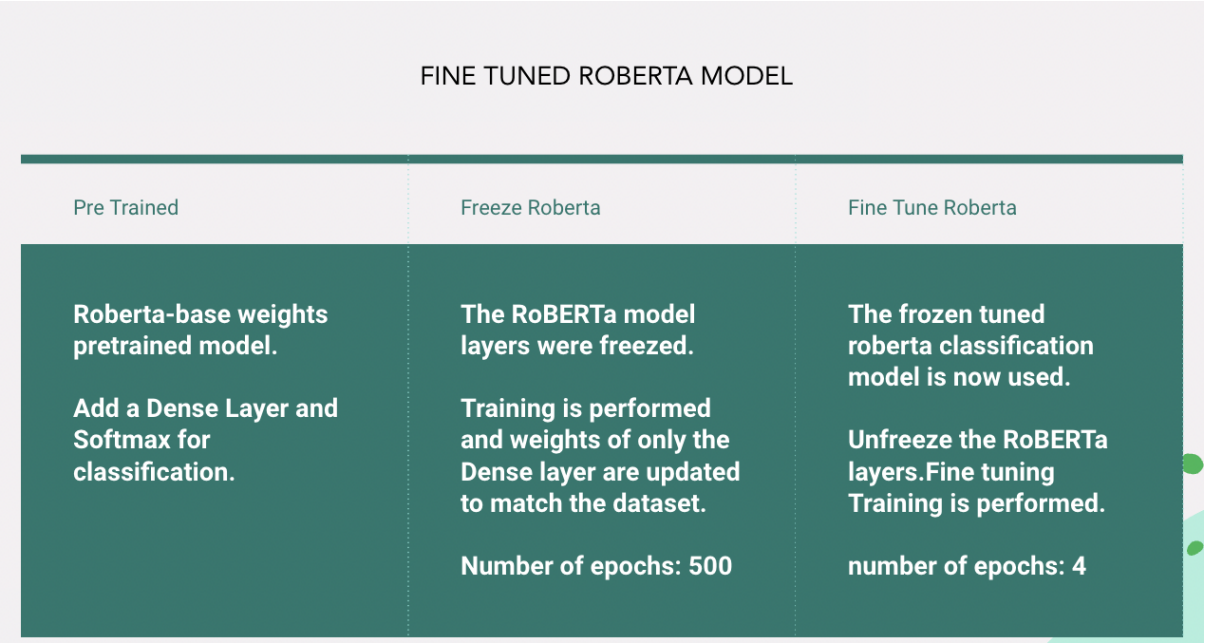
Fine Tuning

- The process of slightly tweaking the model's weights to match the features of the dataset of interest is called fine-tuning. The pre-trained Roberta model was fine-tuned with the

dataset. Unfortunately, due to the small size of the dataset, this mechanism was not very effective. This led to overfitting very quickly but had mediocre accuracy/recall scores. More relevant data would be collected and used to fine-tune the model in the future.

Freezing

- Another interesting approach that was tried is to freeze only the Bert layers from updating weights. However the additional Dense Layer with softmax added above was not frozen. The limited dataset was now used to fine tune only these weights.



Fine Tuned Roberta + Numerical Features Hybrid Model

- The true power of the deep learning models don't arise from randomly training with more data. Domain knowledge and methods to specify important features help the models avoid biases and to generalize better to the task at hand. With this view, a model architecture that can combine the previously generated numerical features (from the lexicons) and the RobertaModel was designed.

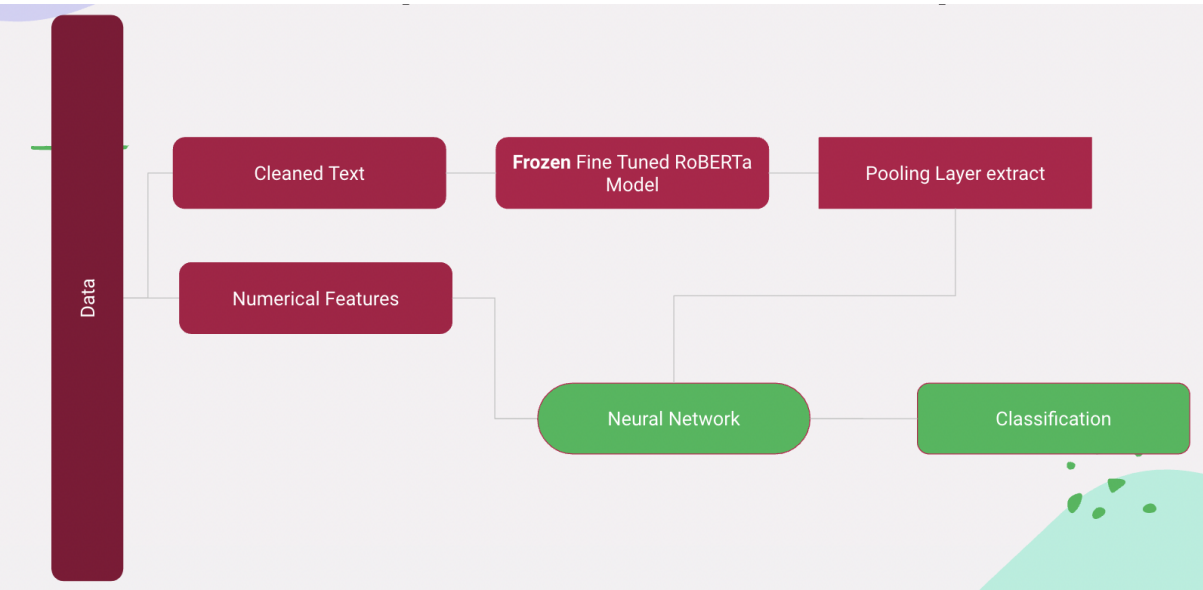
Hybrid Model 1

- Initially, the Fine Tuned RobertaModel was added to the clean text, and the 768 features it gave were concatenated to the numerical features. A Dense Layer and softmax were used to result in it a classification.
- However, this model gave a very erratic result and did not find the proper convergence. Upon some analysis, it was found that the RobertaModel required very few epochs (~4-8), but the Dense layer with the numerical features required many epochs (~500) to learn

the optimal features. This resulted in a problem of underfitting one model and overfitting the other simultaneously.

Hybrid Model 2

- To solve the above problem, the architecture was tweaked a little. The Roberta Model was initially independently trained and fine-tuned (with a small number of epochs). The model layers were frozen, and the resulting features were concatenated with the numerical features. This new model was now trained for a more significant number of epochs. This mechanism ensures that the RobertaModel does not overfit and the Numerical features do not underfit.



Future Improvements

- Due to the complex nature of the Roberta Model, although these did give us some improvements, they were not conclusive due to the non-provable robustness of this model limited by the small dataset used. In the future, more data, numerical features, and a deeper understanding of RobertaModel layers would be the focus of obtaining the right architecture.

References

https://huggingface.co/sentence-transformers

https://www.linkedin.com/pulse/suicide-ideation-nlp-analysis-sergey-sundukovskiy/

https://wandb.ai/ayush-thakur/huggingface/reports/How-to-Fine-Tune-HuggingFace-Transformers-on-a-Custom-Dataset--Vmlldzo0MzQ2MDc

Class notes and codes.