Helpier: An intelligent approach for classification of mental health issues using NLP and neural networks

BY

Desai Dhyani Dhaval- 19BCE1218
Joyeeta Dey – 19BCE1243

A project report submitted to
Dr.BHARADWAJA KUMAR
SCHOOL OF COMPUTING SCIENCES AND ENGINEERING

<u>In partial fulfillment of the requirements for the course of CSE4022 – Natural Language Processing</u>

VIT UNIVERSITY, CHENNAI CAMPUS

Vandalur-Kelambakkam Road
Chennai-600127

BONAFIDE CERTIFICATE

Certified that this project report entitled "Helpier: An intelligent approach for classification of mental health issues using NLP" and neural networks
is a bonafide work of **Desai Dhyani Dhaval(19BCE1218) and Joyeeta Dey(19BCE1243)** who carried out the J-component under my supervision and guidance.

Dr. BHARADWAJA KUMAR

Professor

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING (SCSE)

VIT UNIVERSITY, CHENNAI
CAMPUS CHENNAI-600127

ACKNOWLEDGEMENT

We wish to express our sincere thanks and a deep sense of gratitude to our project guide, **Dr. Bharadwaja Kumar,** Professor, SCSE, for his consistent encouragement and valuable guidance pleasantly offered to us throughout project work.

We are extremely grateful to the Dean of the SCOPE, VIT Chennai, for extending the facilities of the school towards our project and for the unstinting support. We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us throughout the course.

1. ABSTRACT

According to the World Health Organisation (WHO), over 264 million people of all ages suffer from depression due to various mental health issues. (WHO, 2020) As such, depression represents a leading cause of disability worldwide. The early diagnosis of mental health issues is an important preventive measure aimed at tackling this important global concern. NLP is a field at the intersection of linguistics, Artificial Intelligence (AI), and computer science that is concerned with enabling computers to interpret, analyze and approximate the generation of human speech. Woebot, Wysa, Joyable, and Talkspace are a few examples of chatbots that are available as Android/iOS apps or websites that can perform mental health assessments using natural conversation.

Although these systems are good at providing general therapeutic advice, they do not replace the role of a psychiatrist. However, such systems allow preventive measures and early diagnosis of mental health issues to avoid their increasing severity.

In the due course of our project, we aim to apply NLP techniques to process patients' explanations of their mental health issues and use neural networks like LSTM to classify their issues into distinct categories. Such a classification system can help identify problem areas and direct patients to appropriate therapists or support groups for their particular issues.

Users would simply have to explain their issue as they would to a therapist and the system would process that explanation using NLP techniques and classify them into a particular category. Such a system can be used in multiple areas. A counseling therapy provider could choose to have their patient perform an initial assessment online, which would help them automatically direct the patient towards an appropriate therapist. This means that providers could save on costs related to initial assessments. Second, it could also be used in mental health counseling applications where the system tries to consolidate patients by constantly analyzing their natural language descriptions of the issues they are dealing with.

2. Table of Contents

3. KEYWORDS

3.1. Noise Removal

Noise removal is the first step in NLP and includes stripping text of any formatting such as HTML tags, page/paragraph breaks, and punctuation marks. Without this step, the computer may interpret 'the', 'The' and ' The' as entirely different words and cause ambiguity in a trained model. To avoid this, most NLP tasks implement a noise removal technique before breaking up the sentences into individual words.

3.2. Tokenization

Tokenization is defined as the process of identifying the basic units within a language expression that need not be decomposed further in subsequent processing. In English, words delimited by spaces are the tokens and the process of breaking down a sentence into its words is called tokenization.

3.3. Normalization

Text can contain a range of non-standard token types such as digit sequences, words, acronyms, and letter sequences in all capitals, mixed-case words, abbreviations, roman numerals, URLs, and email addresses. Re-writing such text using ordinary words is known as the process of normalization. In NLP, several methods exist for this purpose, which can be language- or task-dependent.

3.4. Stemming

Stemming is a process that involves using a blunt axe to chop off word prefixes and suffixes. Through stemming, words such as 'working' and 'worked' become simply 'work'. However, words like 'ring' become 'r' and 'rung' remain as 'rung'. Due to such anomalies, careful attention must be taken before applying a direct stemming method within NLP.

3.5. Lemmatization

Similarly, lemmatization is the process of reducing the inflectional forms and sometimes derivationally related forms of a word into its common base form. This reduces words down to their root dictionary forms. An example of a lemmatized text is when the words 'am', 'are', and 'is' are changed to 'be'.

3.6. POS Tagging

Part-of-Speech (POS) tagging is a basic step in NLP that checks each word token's functional role within a sentence. Identifying the part of speech of a word can provide useful info on how the word is used within a sentence. This subsequently allows for the effective recognition of intents and entities. POS tagging can also be helpful for stemming and lemmatization since it ensures that the correct root form of the word is identified using its part of speech. WordNet is a large lexical English database where nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms called synsets (Princeton University, 2020). The synsets are interlinked by conceptual semantic and lexical relations so that each synset expresses a distinct concept but is meaningfully interconnected. As such, WordNet provides an easy way to access the part of speech of a certain word and is widely used in the English language NLP for this purpose.

3.7. Stop Word Removal

Stop words are those words within a language that commonly occur within many sentences and are thus filtered out before or after processing natural language data. The words such as ‘do’, ‘did’, ‘is’, ‘am’, ‘are’, ‘has’ and ‘have’ often occur in English sentences and add to the meaning of the entire context. However, in certain NLP tasks such as sentiment analysis and topic modeling, the frequent occurrence of these words may create a mode bias towards the space of these words, thereby resulting in a poor model. To rectify this issue, such words are omitted from natural language text before analysis. 2.8. Word Indexing and Padding The tokenized words from within a training corpus are each given an index. This is done to assign an identity to each unique word token. Machine learning (ML) and deep learning algorithms are mathematical models and can only process numerical data. The nature of language data is such that they are not numerical. Therefore, each word token is assigned a numerical ID. A dictionary of tokens (as keys) and their IDs (as values) can be created to map tokens to their IDs and vice versa. After assigning IDs for all possible tokens within the training data, each data point (i.e. each portion of text) is converted to a sequence of IDs that map to their word tokens. This creates a sequence of numerical IDs instead of tokens. Once all data points are converted to numerical IDs, the maximum sequence length is set by checking the data point with the largest sequence length. Then, a process called padding is performed on all sequences with lengths smaller than the maximum sequence length. Padding, usually at the end of the sequence, adds zeros so that the maximum length for each data point is the same. This is a crucial task for sequence modeling because neural networks accept data points with the same number of features. In this case, the length of a sequence is considered as the number of features.

3.9. Word Embeddings

A word embedding is the representation of a word as a numeric vector that will help to compare how words are used and identify words that occur in similar contexts. Vectors can be used in various fields to define anything with multiple dimensions that can hold a varying amount of data. With NLP, word tokens can be converted into vectors of fixed pre-determined dimensions. The data within a vector can be the length of the word, the number of vowels, the occurrence of a certain letter, a part of speech, etc. Therefore, word embedding vectors provide latent information regarding how word tokens are used within a language. In our project, we have tested with two-word embeddings **FastText and Glove word** embeddings

3.10. Neural Networks

Neural networks are mathematical models of so-called ‘artificial neurons’ that mimic biological neurons by taking in some inputs, processing them, and sending outputs. Each neuron in a neural network is responsible for a distinct calculation of the input signal provided.

Since this project involved analyzing the sequences of words (sentences) in the English language to classify them, RNNs were considered the most effective solution.

3.11. Recurrent Neural Network

Usually, the type of input data used in ML and AI are independent and identically distributed (IID). However, this is not the case with sequences. Sequential data elements depend on previous timesteps and are not independent of each other (Raschka and Mirjalili, 2019). Thus, prediction depends upon the same data recorded in the past. Figure 1 represents the sequences for input and output.

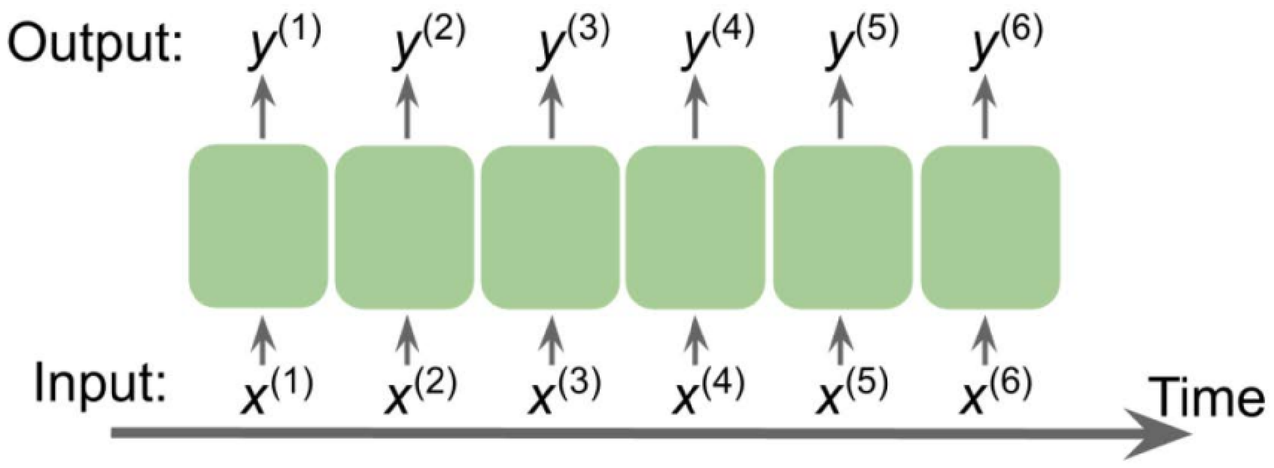

Output:
$y^{(1)}$ $y^{(2)}$ $y^{(3)}$ $y^{(4)}$ $y^{(5)}$ $y^{(6)}$
Input:
$x^{(1)}$ $x^{(2)}$ $x^{(3)}$ $x^{(4)}$ $x^{(5)}$ $x^{(6)}$
Time

Fig 1

Here, X(1), X(2) … X(T) represents the input sequence superscripted with its timesteps. The length of the sequence is T and the outputs are denoted by y(1), y(2)….y(T) for the same time axis. The design of RNNs ensures that the ordering information of sequential data is kept and fed into the network. It can be said that this design helps RNNs keep the memory of previously observed examples. Figure 2 compares standard feed-forward networks and RNNs.

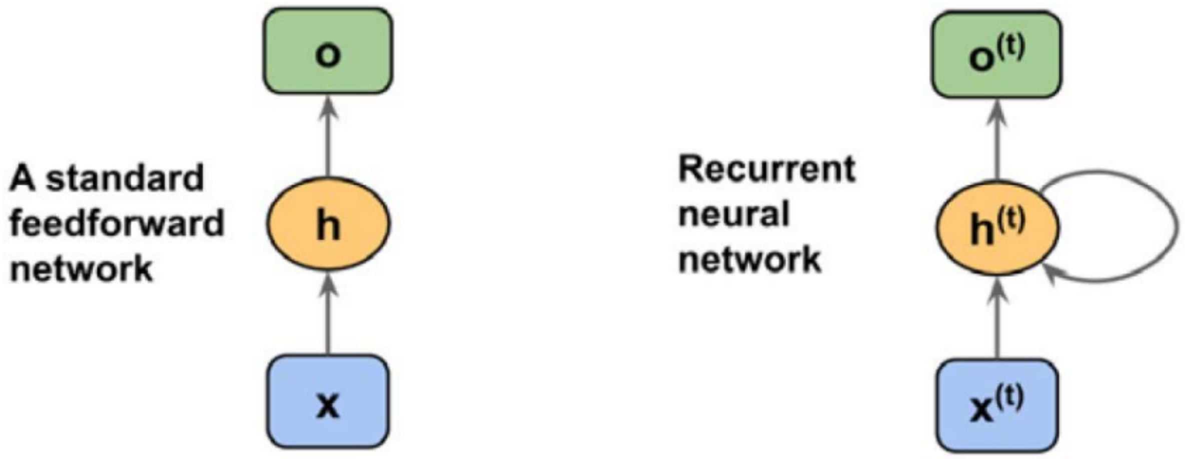


Fig-2

In Figure 2, x, h, and o are the input, hidden and output layers, respectively. Each of these layers is vector finalize vectors of many neuron units. The layers in the RNN are superscripted by (t) to denote the timestamp. In feed-forward networks, the input passes to the hidden layer and the output layer. However, in

RNNs, the hidden layer receives as input both the input layer and the previous timestep's hidden layer, which is denoted by a loop. This accounting of previous timesteps' hidden layers allows RNNs to retain a memory of the past. As every timestep performs this loop, the memory of several timesteps behind can be considered when calculating outputs. This process occurs at each timestep during training and prediction (Raschka and Mirjalili, 2019). Figure 3 is an unfolded diagram of single-layer RNN and multi-layer RNN.

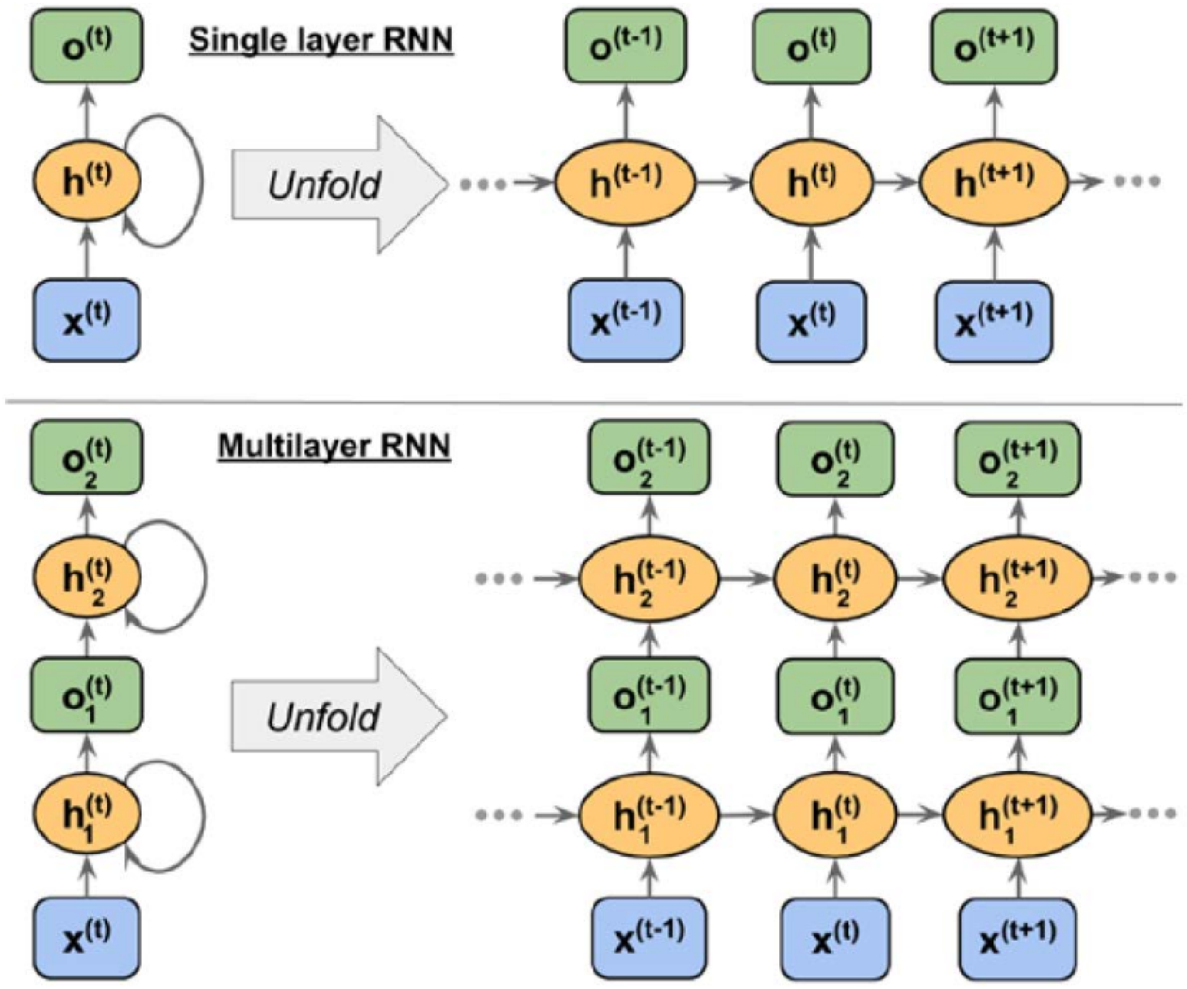

Single layer RNN
$o^{(t)}$
$h^{(t)}$
$x^{(t)}$
Unfold
$o^{(t-1)}$ $o^{(t)}$ $o^{(t+1)}$
$h^{(t-1)}$ $h^{(t)}$ $h^{(t+1)}$
$x^{(t-1)}$ $x^{(t)}$ $x^{(t+1)}$
Multilayer RNN
$o_2^{(t)}$ $h_2^{(t)}$ $o_1^{(t)}$ $h_1^{(t)}$ $x^{(t)}$
Unfold
$o_2^{(t-1)}$ $o_2^{(t)}$ $o_2^{(t+1)}$
$h_2^{(t-1)}$ $h_2^{(t)}$ $h_2^{(t+1)}$
$o_1^{(t-1)}$ $o_1^{(t)}$ $o_1^{(t+1)}$
$h_1^{(t-1)}$ $h_1^{(t)}$ $h_1^{(t+1)}$
$x^{(t-1)}$ $x^{(t)}$ $x^{(t+1)}$

Fig-3

Due to their architecture, RNNs suffer from vanishing and exploding gradients as sequences become longer. Since the text sequences can be very long in NLP, gradients often become much smaller or larger while training the network using the backpropagation algorithm. Backpropagation is a neural network training algorithm that utilizes the gradients of errors during each training step to adjust the weight coefficients of each neuron (Geron, 2019). Therefore, training RNNs for longer sequences (as per NLP) can result in a model with poor performance.

3**.12. Long Short-Term Memory (LSTM) Network**

In 1997, Hochreiter and Schmidhuber proposed LSTM cells to overcome the vanishing and exploding gradients issue faced by RNNs used in longer sequences.

Each neuron within the hidden layer is replaced by an LSTM cell within an LSTM network. Figure 4 presents an LSTM memory cell and its components

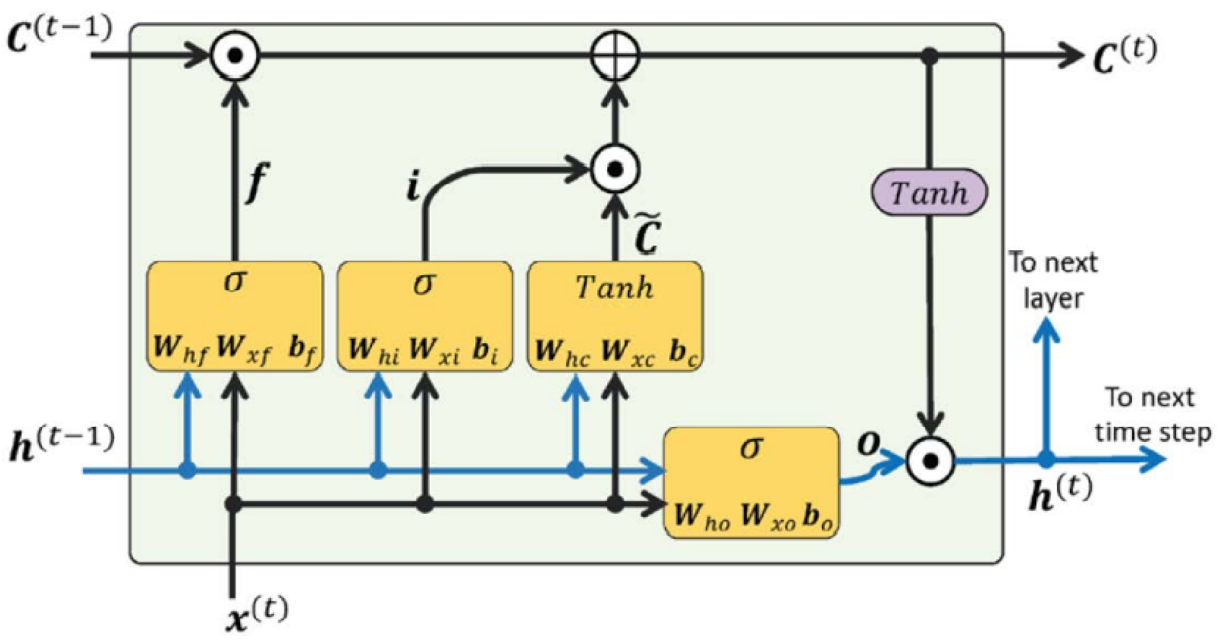


Fig-4

3.13. Activation Functions

Activation functions are used to finalize the output made by each neuron. Some neurons are designed to output only their dot product, which is known as identity function activation (Raschka and Mirjalili, 2019). However, for many ML tasks, the identity function may not be ideal and thus several activations functions exist to provide for this shortcoming. The most common loss functions in use are the rectified linear unit (ReLU), Sigmoid (logistic), Softmax, and Hyperbolic Tangent (tanh). ReLU is used for linear activation with all negative outcomes adjusted to zero, Sigmoid is used for binary classification, Softmax is used for multiclass classification, and tanh is mostly used in hidden layers of a neural network (Geron, 2019). This project will employ ReLU, Softmax, and tanh activation functions

3.14. Loss Functions

Neural networks are trained in a process called backpropagation, where the errors calculated by comparing the output of the network to the ground truth are propagated backward through the layers of the network to adjust the weight coefficients of each neuron. The loss function determines the error and backpropagation method and demands that the loss function be differentiable so the derivatives of loss at each layer can be calculated to adjust the weight parameters for the neurons (Geron, 2019). Loss functions compute the error between the predicted outcome and the ground truth for each sample for which the learning model makes a prediction. The summation of all the errors is called total loss and is used during backpropagation to adjust the weight parameters of the neurons so predictions during the next epoch of training result in a smaller total loss value (Raschka and Mirjalili, 2019). Several loss functions are used according to the learning task the model is performing. In the case of multiclass classification, the categorical cross-entropy loss function is used which is shown below.

$$CCE = -\frac{1}{N}\sum_{i=0}^{N}\sum_{j=0}^{J} y_j \cdot log(\hat{y}_j) + (1 - y_j) \cdot log(1 - \hat{y}_j)$$

3.15. Optimizers

Optimization algorithms are widely used within the ML community since they help to minimize or maximize objective functions. Since neural networks are considered universal function approximators, several optimization algorithms are proposed to help optimize parameters within the neurons of neural networks. Optimizers determine how quickly the loss of a neural network is driven towards its global function minima (Raschka and Mirjalili, 2019). Some examples of optimizers are gradient descent, stochastic gradient descent, Adam and RMSprop.

3.16. Overfitting and Underfitting

During the training phase, the model starts to make predictions on the training data and it may be far from reducing its loss to the optimum level. At this stage, the model has not seen enough data features to train its parameters and will fail to generalize for both training and validation sets. This phenomenon is known as underfitting. As the training progresses to further epochs, a certain optimum level is reached where the validation loss approaches the training loss. If the model is trained for further epochs, the validation loss will start to increase while the training loss will continue to decrease. If this occurs, the model may fit the training set well but will fail to generalize to new examples. This phenomenon is known as overfitting (Raschka and Mirjalili, 2019). For a good-performing model, the training must be stopped at the epoch where the optimum is reached.

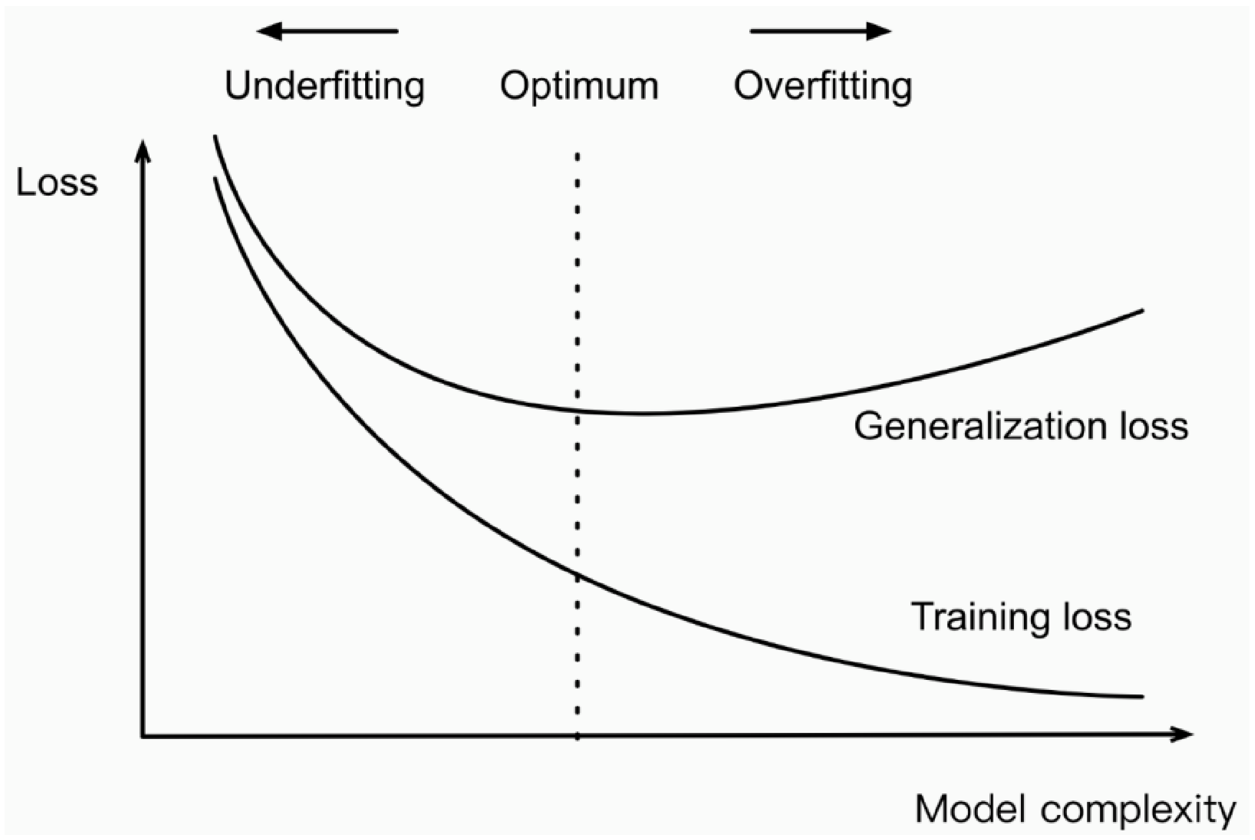


Fig-5

3.17. Performance Measures for Classification

Performance measures are metrics that help researchers understand how effectively a learning model has been trained. They provide insights into whether a model will generalize well on unseen data. Several performance measures are available depending on the learning task. Since this project is a classification task, some of the relevant classification measures commonly used in the ML community will be outlined in the following subsections.

3.17.1. Accuracy

Accuracy is the most commonly used metric for classification. It is the proportion of true results among the total number of samples examined (Geron, 2019). Although accuracy is a good metric for classification evaluation, it may not always be the best metric—especially in cases where the target class is very sparse. An example is a case for cancer detection where the dataset may contain very few samples with positive cancer detection. The model could easily classify all samples as negative and still return a high accuracy score. The formula for accuracy is as follows:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{All Samples}}$$

3.17.2. Precision
Precision is the measure of how many of the total positive outcomes predicted by a learning model are truly positive (Geron, 2019). It is a metric of choice if there is a need to be certain of a model's prediction. An example of this is a system to predict whether a customer's credit rating should be decreased. Not being certain of the prediction may cause customer dissatisfaction, meaning there is a need to be certain about the prediction. The formula for precision is as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

3.17.3. Recall (sensitivity)
Recall or sensitivity is the measure of how effectively a model predicts positive outcomes correctly overall existing positive outcomes (Geron, 2019). It is a good evaluation metric to use when there is a need to capture as many positives as possible. For the cancer detection example, all possible positives must be identified—even if some of them are negative—so that further testing can be performed on all suspected positive cases. The formula for the recall is as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

3.17.4. F1 score
F1 score is an evaluation metric that returns a number between 0 and 1 and represents the harmonic mean of precision and recall (Geron, 2019). It is used when there is a need for a model to have both good precision and recall. It helps maintain a balance between precision and recall since both low precision and low recall would result in the F1 score becoming low. The formula for the F1 score is as follows:

$$\text{F1 Score} = \frac{2 \text{ x } (\text{Precision x Recall})}{\text{Precision} + \text{Recall}}$$

3.18. Transfer Learning
Transfer learning emerges from the problem of the bottleneck caused by trying to train a model on massive datasets. Training a model using a massive dataset is time-consuming, requires data management, and is computationally expensive. Moreover, it is also difficult to tune hyperparameters. According to Zhuang et al. (2020), transfer learning solves this by improving the performance of learning models on target domains by transferring the knowledge contained in different

but related source domains. With transfer learning, the hidden layers within a trained model are transferred with its learned parameters into a new model where only the later layers are trained to solve the problem in the new related domain. In other words, transfer learning is a situation in which what has been learned in one domain is exploited to improve model generalization in another domain.

3.19 BERT model

BERT is a pre-trained model published by Devlin et al. (2019) in Google that is publicly available. It provides dense vector representations for natural language data using a pre-trained deep neural network with Transformer architecture. It is trained on the BooksCorpus with 800M words and also on a version of English Wikipedia that has 2,500M words. In a recent paper by Gonzalex-Carvajan and Garrido-Merchan (2020), a comparison of the BERT model with traditional ML classifiers was provided using four different text classification experiments. In all of their experiments, it was proven that the BERT model resulted in higher accuracy and was also far less complicated to implement than traditional ML models. However, they also acknowledged the limitations of the BERT model and recommended further research in the hyperparameter auto-tuned BERT model for NLP tasks with Bayesian optimization.

4. INTRODUCTION

4.1. Background

According to the World Health Organisation (WHO), over 264 million people of all ages suffer from depression due to various mental health issues. (WHO, 2020) As such, depression represents a leading cause of disability worldwide. The early diagnosis of mental health issues is an important preventive measure aimed at tackling this important global concern. In a telephone survey conducted in Germany, it was revealed that shame and self-stigmatization were the main reasons for not seeking psychiatric help for depression (instead of perceived stigma and negative reactions) (Knesebeck et al., 2018). However, recent advances in technology—especially in natural language processing (NLP)—have presented the world with various application-based and online diagnosis tools where the patient can be diagnosed from the comfort of their own home. NLP is a field at the intersection of linguistics, Artificial Intelligence (AI) and computer science that is concerned with enabling computers to interpret, analyse and approximate the generation of human speech. Woebot, Wysa, Joyable and Talkspace are a few examples of chatbots that are available as Android/iOS apps or websites that can perform mental health assessments using natural conversation. Although these systems are good at providing general therapeutic advice, they do not replace the role of a psychiatrist. However, such systems allow preventive measures and early diagnosis of mental health issues to avoid their increasing severity. The present study applied NLP techniques to process patients' explanations of their mental health issues and used neural networks to classify their issues into distinct categories. Such a classification system can help identify problem areas and direct patients to appropriate therapists for their particular issues. The data used for this research was extracted online from the repository by Nicolas Bertagnolli, who received the data from CounselChat.com (Bertagnolli, 2020). The data consists of counseling conversations in which a mental health issue is posed by a patient and a therapeutic response is provided by a qualified therapist. The paper will first present the aims and objectives of the research and layout an appropriate hypothesis. It will then review relevant existing literature on the posed problem as well as the existing methods in use. The paper will then explain the research method used to address the problem. Thereafter, the paper will delve into the development process of the classification algorithm, where the NLP and sequence modeling techniques used for this research will be described. The paper will then present the findings of the research and discuss their relevance. Finally, the paper will provide a coherent conclusion including the limitations of the research, and make recommendations for further research.

4.2. Aims and Objectives

The purpose of this research was to explore sequence modeling techniques in natural language data to classify user-written text into various categories of mental health issues. Since the text provided by users is a sequence of words, it is considered that sequence2sequence neural network architectures are the best fit for this problem. A recurrent neural network (RNN) is a type of sequence modeling neural network. Moreover, long short-term memory (LSTM) cells are used to improvise results since sequences of words can be too long for RNNs to process. Text pre-processing was used to convert text data into numerical data so that it can be accepted by the neural network. Finally, a simple text input system took the text input from users so that the trained model could classify it into the appropriate categories of mental health issues.

5. <u>Literature Survey</u>

The use of LSTM cells has gained considerable popularity in NLP due to the feasibility of processing longer sequences.
Young et al. (2018) agreed with this point in their exploration of deep learning-based NLP, wherein they mentioned that the forget gates within an LSTM cell allow for error to backpropagate through an unlimited number of timesteps, thereby overcoming both the vanishing and exploding gradients problem faced by simple RNNs.
Wang et al. (2018) experimented with social media data and applied LSTM networks for a sequence of word embedding vectors generated from social media data. They reported that the LSTM model outperformed both the naïve Bayes model and the extreme learning machine (ELM) model. From their findings, it was evident that the deep learning methods using LSTM and word embeddings allowed them to effectively learn word usage in the context of social media provided that there was enough training data. This provides a stronghold for the use of LSTM networks in text classification since they are effective at retaining longer sequence memory, which is vital in NLP. However, they also mentioned that the quantity and quality of training data greatly affected the performance of their model.

This section will review the core theoretical concepts used within this project as well as the latest mental health assessment research using NLP. The history of NLP dates back to 1950 when Alan Turing proposed a test for artificial intelligence evaluating whether a computer can use language to fool humans to believe it is human (Ganegedara, 2018). However, besides approximating human speech, NLP has a wide range of other applications that include sentiment analysis, the detection of spam emails and bias within the text, improving accessibility for people with special needs, question answering, and language translation, among others.
Based on recent research, it is understood that NLP aims to create a machine that can pass the Turing test. To accomplish this, researchers have broken down several intermediate tasks within NLP, including tokenization, word-sense disambiguation, named entity recognition, part-of-speech tagging, sentence classification, language generation, question answering, and machine translation. This project utilizes a few of these tasks, which are described in this section.

6. DATA SETS

Counselchat Dataset

Counselchat.com counseling data was provided by the founders to Nicolas Bertagnoli. Data is available to use from his public GitHub repository. Initially, the data was scraped from **www.counselchat.com**.

There are **31** topics on the forum, with the number of posted responses ranging from **317 for the topic of "depression" to 3 for "military issues"**. There are **307 therapist** contributors on the site, most of whom are located on the West Coast of the US (Washington, Oregon, California). They range in licensing from Ph.D. level psychologists, social workers, and licensed mental health counselors.

The site had some duplication of questions and answers that appeared to be generated by the users themselves (e.g., where a therapist copy-pasted a response from one question to another similar question); we did not clean the data for this kind of duplication.

The dataset is presented as a CSV with 10 columns described below:

- questionID — A unique question identifier which is distinct for every question
- questionTitle — The title of the question on counsel chat
- questionText — The body of the individual's question to counselors
- questionLink — A URL to the last location of that question (might not still be active)
- topic — The topic the question was listed under
- therapistInfo — The summary of each therapist, usually a name and specialty
- therapistURL — a link to the therapist's bio on counselchat
- answerText — The therapist response to the question
- upvotes — The number of upvotes the answerText received
- split — The data split for training, validation, and testing.

In general, most questions have only a few responses with 75% of questions having two or fewer total responses. However, many questions have a lot of therapist engagement.

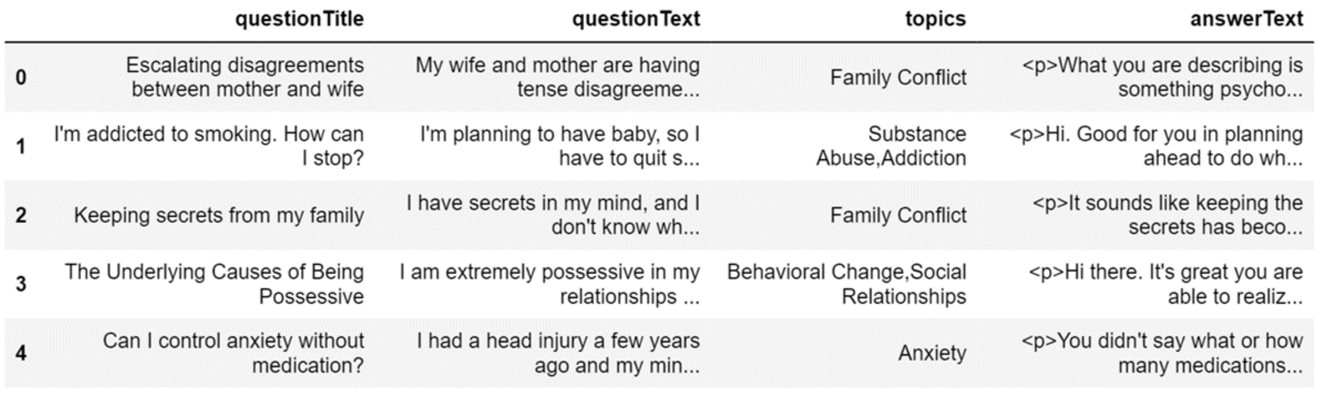
<u>7. METHODOLOGY</u>

The present project was implemented in the Python programming language. Jupyter notebook was used to write the code for the system and display the results. The Tensorflow library by Google Inc and its Keras API were used as the backend of the neural networks. The collected data were analysed, wrangled and cleaned using the Pandas library.

7.1. Initial Data Analysis

The data were loaded into a Pandas DataFrame from a CSV file (available in Nicolas Bertagnolli's public Github repository). Upon loading the dataset, the author noted that it contained some sensitive information such as therapist names and URLs as well as the text and category data required for the project. Moreover, additional columns existed regarding upvotes, answer text and question ID, which were considered unnecessary. All of the sensitive and unnecessary columns were first removed from the dataset before progressing into further processing to comply with the ethical considerations. Figure 6 shows all the column names that were available.

Once the aforementioned columns were removed, the only two columns that remained were 'questionText' and 'topics', for which the first five lines are shown in Figure 6.



	questionTitle	questionText	topics	answerText
0	Escalating disagreements between mother and wife	My wife and mother are having tense disagreeme...	Family Conflict	<p>What you are describing is something psycho...
1	I'm addicted to smoking. How can I stop?	I'm planning to have baby, so I have to quit s...	Substance Abuse,Addiction	<p>Hi. Good for you in planning ahead to do wh...
2	Keeping secrets from my family	I have secrets in my mind, and I don't know wh...	Family Conflict	<p>It sounds like keeping the secrets has beco...
3	The Underlying Causes of Being Possessive	I am extremely possessive in my relationships ...	Behavioral Change,Social Relationships	<p>Hi there. It's great you are able to realiz...
4	Can I control anxiety without medication?	I had a head injury a few years ago and my min...	Anxiety	<p>You didn't say what or how many medications...

Fig 6.

The dataset included 1482 samples. Through data exploration, it was noticed that there were 99 null values in 'questionText' and 10 null values in 'topics'. The rows containing these null values were simply removed from the dataset since there was no alternative method to impute missing values. Thereafter, 1376 samples remained in the dataset.

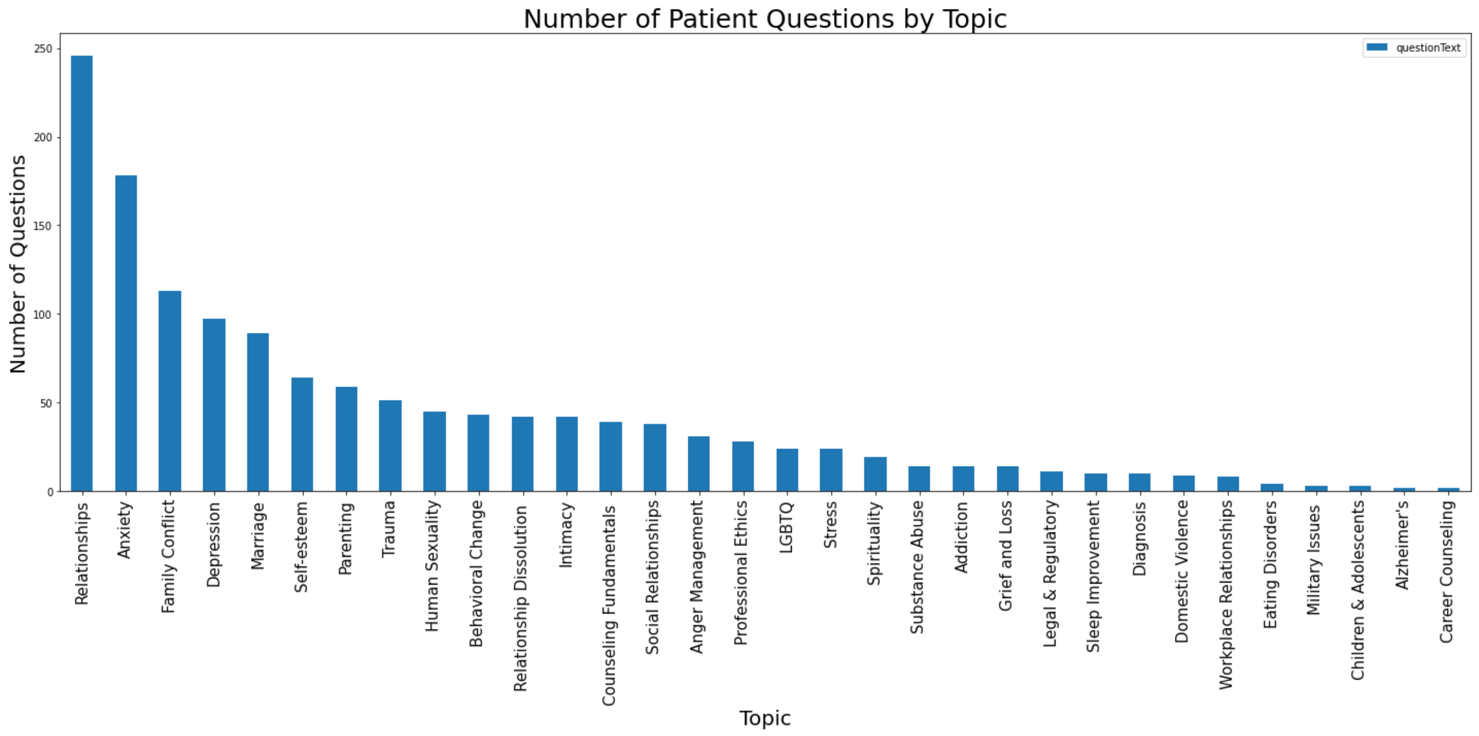


Fig 7.

Some samples were labelled with multiple topics. A Keras LSTM model was implemented with multi-labels, however the model failed to converge even with multiple attempts. Thus, it was decided that only the first label of the multi-label samples would be kept. A plot was then created to observe the distribution of questionText by topics defined within the dataset.

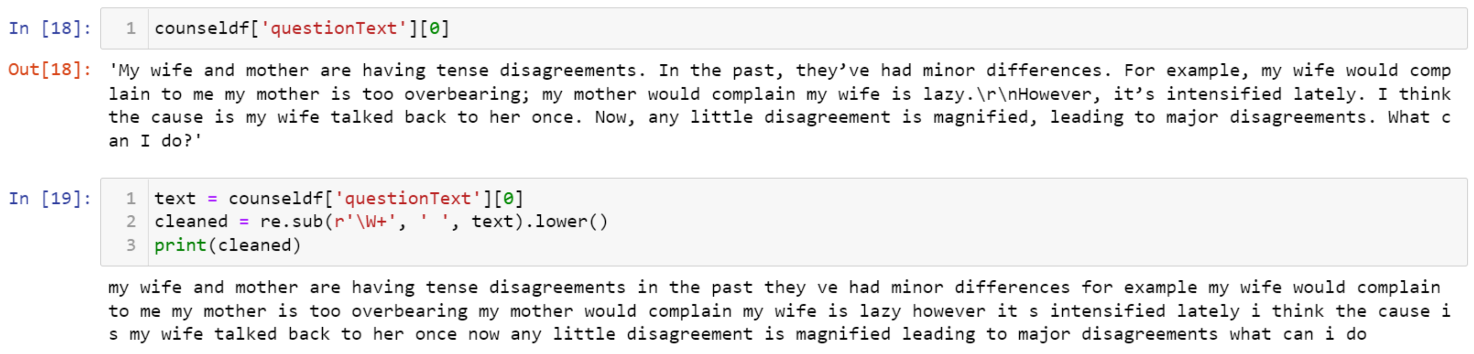
As seen in Figure, the **data were not equally distributed across all categories**. This implies that the **model may be biased toward some categories** within the dataset. To check for model bias, various classification metrics were employed on the trained models.

7.2. Text Processing

In this section, the step-by-step process of text data processing used in this project will be shown.

7.2.1. Noise removal

The data samples were checked individually and it was noted that the samples contained capitalisations, punctuation marks and some HTML tags. These elements are considered noise within a natural language dataset since they do not contain actual meaning. Noise removal was performed using regular expressions (Regex). Figure 8 demonstrates how noise removal affects a data sample.

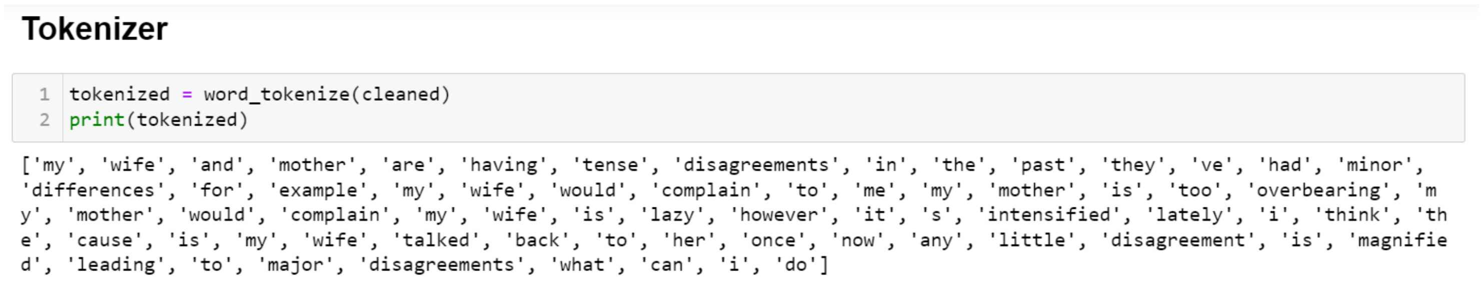


```
In [18]:  counseldf['questionText'][0]

Out[18]: 'My wife and mother are having tense disagreements. In the past, they’ve had minor differences. For example, my wife would comp
         lain to me my mother is too overbearing; my mother would complain my wife is lazy.\r\nHowever, it’s intensified lately. I think
         the cause is my wife talked back to her once. Now, any little disagreement is magnified, leading to major disagreements. What c
         an I do?'

In [19]:  text = counseldf['questionText'][0]
          cleaned = re.sub(r'\W+', ' ', text).lower()
          print(cleaned)

         my wife and mother are having tense disagreements in the past they ve had minor differences for example my wife would complain
         to me my mother is too overbearing my mother would complain my wife is lazy however it s intensified lately i think the cause i
         s my wife talked back to her once now any little disagreement is magnified leading to major disagreements what can i do
```

Fig 8.

7.2.2. Tokenization

Following noise removal, tokenization was performed for each sample. This process simply involved breaking the text into its word tokens. Figure provides a tokenized version of the same example (first sample).



Tokenizer

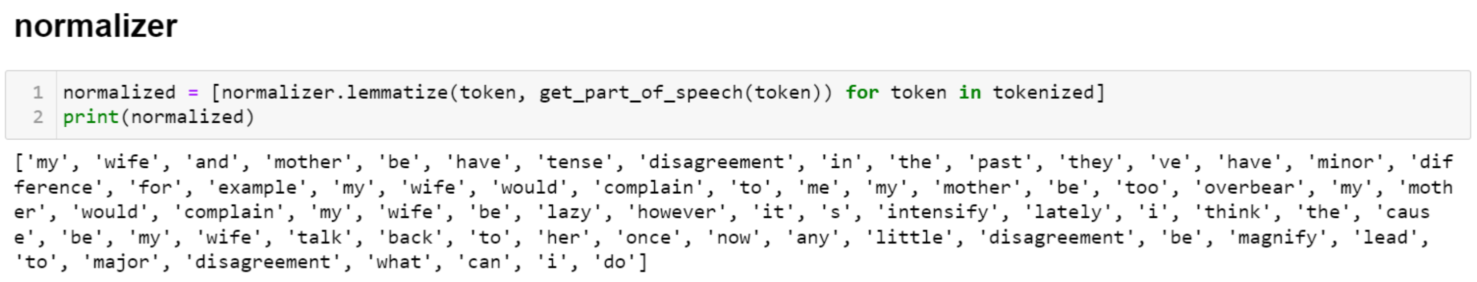
```
tokenized = word_tokenize(cleaned)
print(tokenized)
```

```
['my', 'wife', 'and', 'mother', 'are', 'having', 'tense', 'disagreements', 'in', 'the', 'past', 'they', 've', 'had', 'minor',
'differences', 'for', 'example', 'my', 'wife', 'would', 'complain', 'to', 'me', 'my', 'mother', 'is', 'too', 'overbearing', 'm
y', 'mother', 'would', 'complain', 'my', 'wife', 'is', 'lazy', 'however', 'it', 's', 'intensified', 'lately', 'i', 'think', 'th
e', 'cause', 'is', 'my', 'wife', 'talked', 'back', 'to', 'her', 'once', 'now', 'any', 'little', 'disagreement', 'is', 'magnifie
d', 'leading', 'to', 'major', 'disagreements', 'what', 'can', 'i', 'do']
```

Fig 9.

7.2.3. Lemmatization

The tokenization provided individual word tokens for each sample. However, in the English language, some word tokens take different forms according to their usage within a sentence. Therefore, it was necessary to convert those tokens into their root forms so that the learning model did not treat them as different tokens. Lemmatization was performed by first checking each word’s part of speech using WordNet synsets and changing the words into their root forms according to the parts of speech found.



normalizer

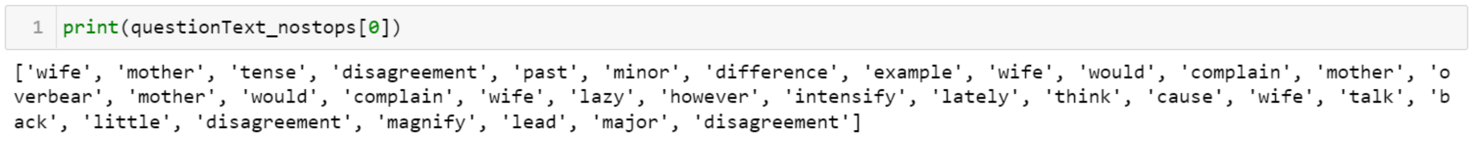
```
normalized = [normalizer.lemmatize(token, get_part_of_speech(token)) for token in tokenized]
print(normalized)
```

```
['my', 'wife', 'and', 'mother', 'be', 'have', 'tense', 'disagreement', 'in', 'the', 'past', 'they', 've', 'have', 'minor', 'dif
ference', 'for', 'example', 'my', 'wife', 'would', 'complain', 'to', 'me', 'my', 'mother', 'be', 'too', 'overbear', 'my', 'moth
er', 'would', 'complain', 'my', 'wife', 'be', 'lazy', 'however', 'it', 's', 'intensify', 'lately', 'i', 'think', 'the', 'caus
e', 'be', 'my', 'wife', 'talk', 'back', 'to', 'her', 'once', 'now', 'any', 'little', 'disagreement', 'be', 'magnify', 'lead',
'to', 'major', 'disagreement', 'what', 'can', 'i', 'do']
```

Fig 10.

7.2.4. Stop word removal

The samples within the dataset had several commonly occurring words that were necessary to remove so that the learning model would not focus on their presence within each sample. Words such as ‘is’, ‘am’, ‘are’, ‘has’, ‘had’, ‘have’, ‘my’, ‘their’, ‘what’, ‘that’, ‘it’ and ‘they’ were commonly observed. Figure 11 shows the stop word removal performed for the first sample.



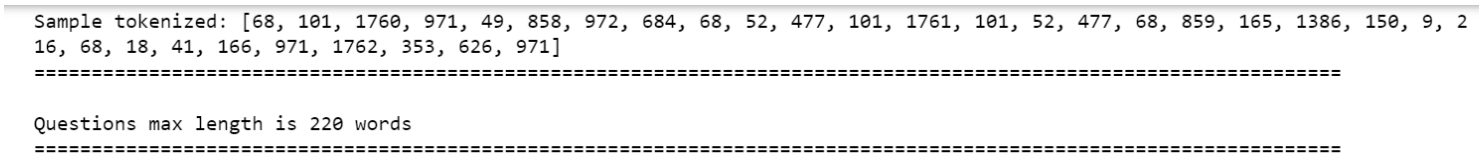
```
print(questionText_nostops[0])
```

```
['wife', 'mother', 'tense', 'disagreement', 'past', 'minor', 'difference', 'example', 'wife', 'would', 'complain', 'mother', 'o
verbear', 'mother', 'would', 'complain', 'wife', 'lazy', 'however', 'intensify', 'lately', 'think', 'cause', 'wife', 'talk', 'b
ack', 'little', 'disagreement', 'magnify', 'lead', 'major', 'disagreement']
```

Fig 11.

7.2.5. Conversion to numerical data

Once the samples were stripped of their stop words, they were converted to numerical data. This was performed using the Keras preprocessing library. The text tokenizer within the Keras preprocessing library takes the whole dataset of a text corpus as input and assigns numerical indices to all unique word tokens. This index map is saved as a dictionary that can be easily accessed using the word_index function of the tokenizer class. A section of the dictionary created using the dataset for this project is presented in Figure 12.



```
Sample tokenized: [68, 101, 1760, 971, 49, 858, 972, 684, 68, 52, 477, 101, 1761, 101, 52, 477, 68, 859, 165, 1386, 150, 9, 2
16, 68, 18, 41, 166, 971, 1762, 353, 626, 971]
===========================================================================================

Questions max length is 220 words
===========================================================================================
```

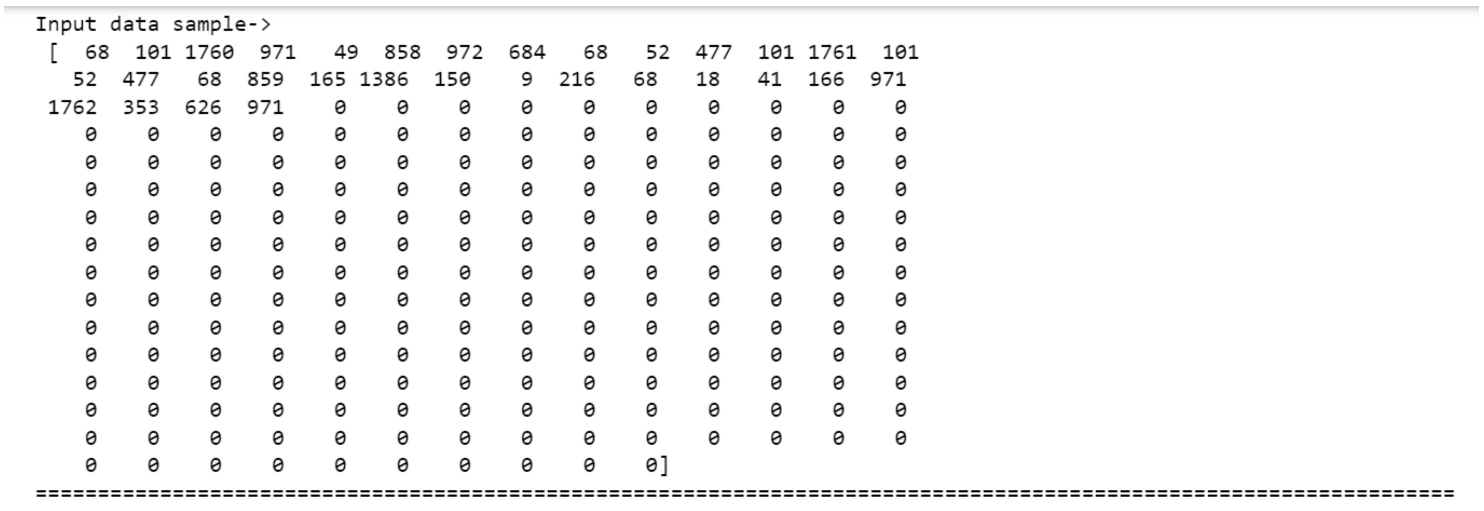
Fig 12.

As displayed in Figure 12, the tokenizer class assigns numerical indices serially without any particular preference to specific word tokens. This dictionary acts as an identifier of the word tokens within the overall token feature space of the whole corpus. The trained learning model will be able to identify the location of the tokens within the feature space using the indices.

After creating this dictionary, the next step was to take in the samples individually and map them to their token indices based on the dictionary created.

7.2.6. Padding

The token lengths for each sample were different, which is the case in most NLP projects. Therefore, the maximum sequence length was determined by checking the length of each sample. The longest sample included 220 tokens. Therefore, all other sequences shorter than 220 tokens were post-padded with zeros to make up for the shortcoming. Figure 13 presents the first sample padded with zeros.



```
Input data sample->
 [  68  101 1760  971   49  858  972  684   68   52  477  101 1761  101
   52  477   68  859  165 1386  150    9  216   68   18   41  166  971
 1762  353  626  971    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0]
==============================================================================================================
```

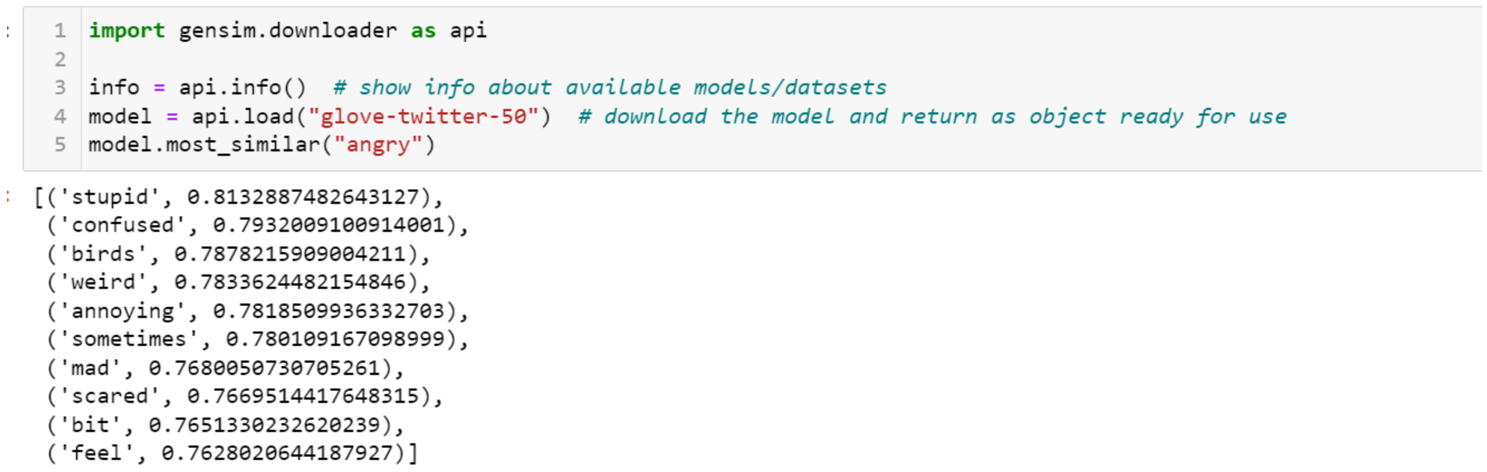
Fig 13.

7.2.7. Word embeddings

To create word embeddings of tokens for each sample within the dataset, a Keras embedding layer was used. In this case, the length of the word index dictionary is 2417. This means that the tokens converted as one-hot vectors would be 2417 dimensions long for each token. This would increase the processing time for each sample and the learning model would take a long time to train and predict. The implemented Keras embedding layer was parameterised to 200-dimensional vectors, meaning each token had only 200 dimensions instead of 2417.

7.2.7. Word2Vec - Glove embeddings

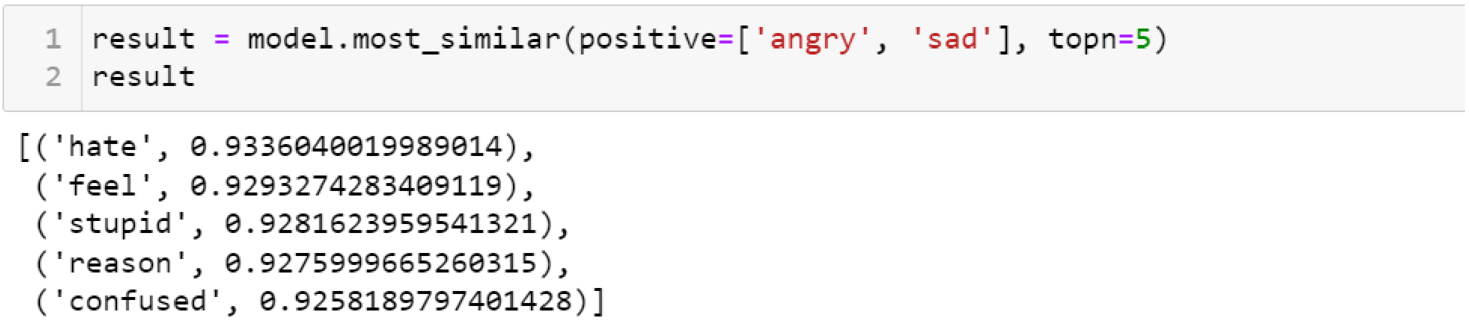
GloVe is a word vector technique. The advantage of GloVe is that, unlike Word2vec, GloVe does not rely just on local statistics (local context information of words), but incorporates global statistics (word co-occurrence) to obtain word vectors. But keep in mind that there's quite a bit of synergy between the GloVe and Word2vec.



```python
import gensim.downloader as api

info = api.info()  # show info about available models/datasets
model = api.load("glove-twitter-50")  # download the model and return as object ready for use
model.most_similar("angry")
```

```
[('stupid', 0.8132887482643127),
 ('confused', 0.7932009100914001),
 ('birds', 0.7878215909004211),
 ('weird', 0.7833624482154846),
 ('annoying', 0.7818509936332703),
 ('sometimes', 0.780109167098999),
 ('mad', 0.7680050730705261),
 ('scared', 0.7669514417648315),
 ('bit', 0.7651330232620239),
 ('feel', 0.7628020644187927)]
```

Fig 14.



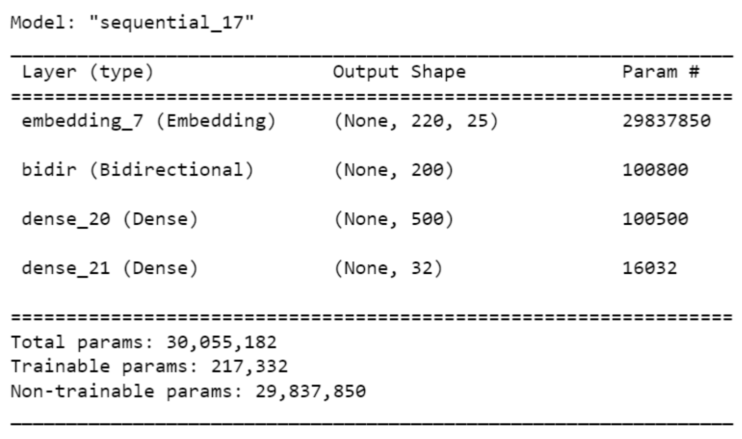
```
result = model.most_similar(positive=['angry', 'sad'], topn=5)
result
```

```
[('hate', 0.9336040019989014),
 ('feel', 0.9293274283409119),
 ('stupid', 0.928162395954132l),
 ('reason', 0.9275999665260315),
 ('confused', 0.9258189797401428)]
```

Fig 15.

7.3. Neural Network Design

As previously stated, this is a sequence classification project. Therefore, an LSTM network was chosen as the core neural network structure. Moreover, a **bidirectional LSTM with 100 neurons and dropout of 0.1** was chosen for sequence modelling so that the learning model would go through the sequence from start to end and vice versa. A **dense layer with 500 neurons and ReLU activation was added after the LSTM network** to allow for further featurisation. Finally, **a 32-neuron dense layer with softmax activation was chosen as the final layer for classifying the samples into 32 categories.** Overall, **30,055,1282 total weight parameters** were optimised during training.



```
Model: "sequential_17"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_7 (Embedding)     (None, 220, 25)           29837850

 bidir (Bidirectional)       (None, 200)               100800

 dense_20 (Dense)            (None, 500)               100500

 dense_21 (Dense)            (None, 32)                16032

=================================================================
Total params: 30,055,182
Trainable params: 217,332
Non-trainable params: 29,837,850
_________________________________________________________________
```

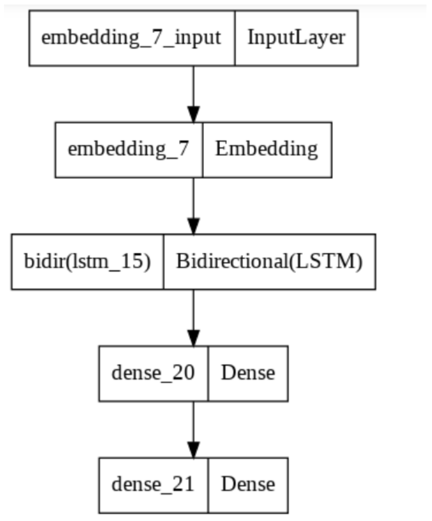

embedding_7_input | InputLayer
embedding_7 | Embedding
bidir(lstm_15) | Bidirectional(LSTM)
dense_20 | Dense
dense_21 | Dense

Fig 16.

<u>8. Results</u>

8.4.1. LSTM training

The model was trained for **30 epochs.** The t**raining accuracy** continuously increased and ultimately reached **84%**. However, the **test accuracy** stabilised at approximately **65%**. This is a great result considering the size of the dataset. It is possible that with more data, the test accuracy could be increased further since there will be more word tokens available for the network to rely on.

8.4.2 BERT Model

The model was trained for **20 epochs** but it took a very long time in training and only **59% accuracy** was achieved. Hence, we didn't consider that model.

8.4.3. Graphs

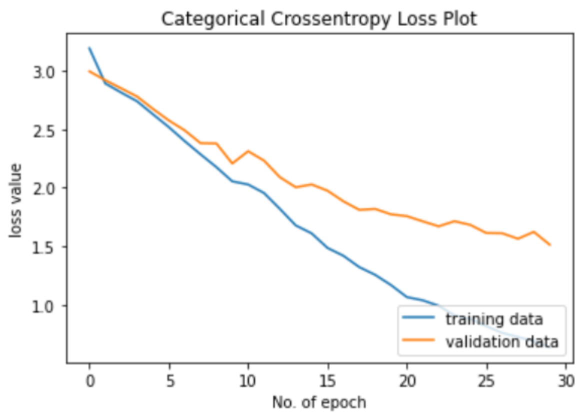
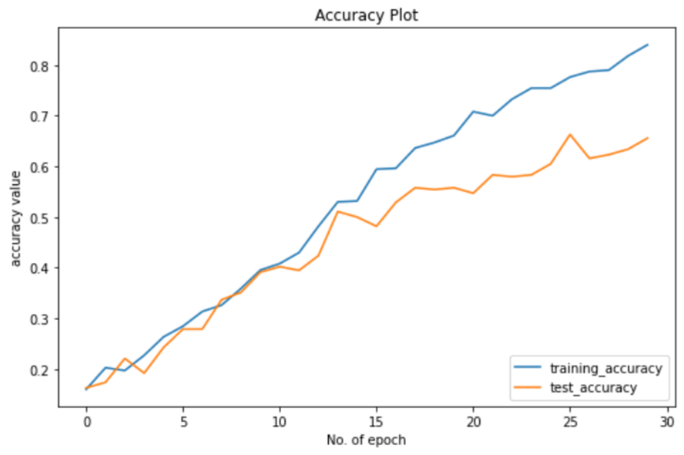


Fig 17.



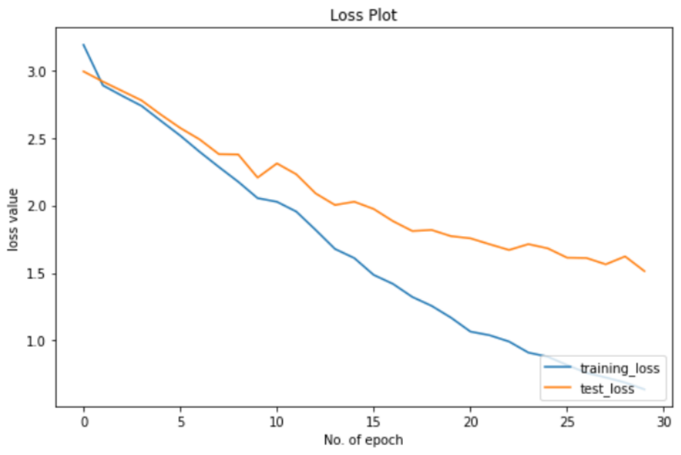
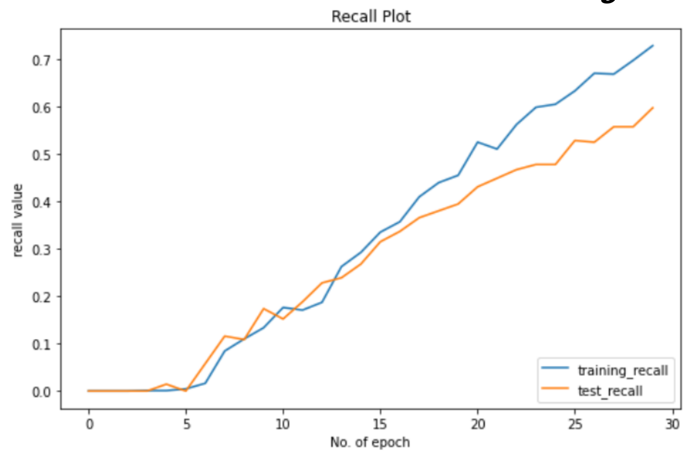


Fig 18 and Fig 19.



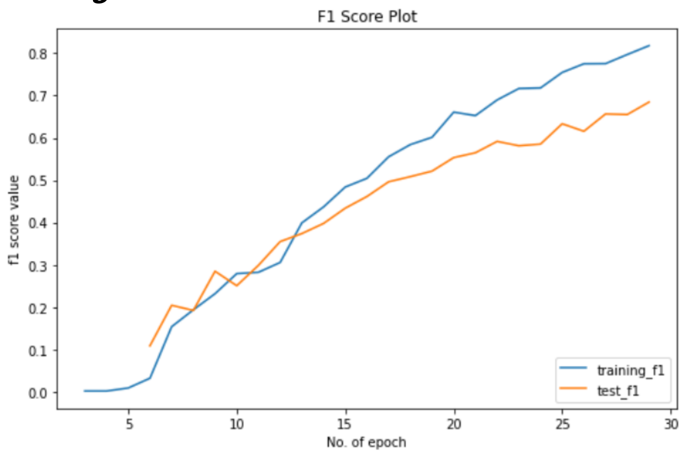
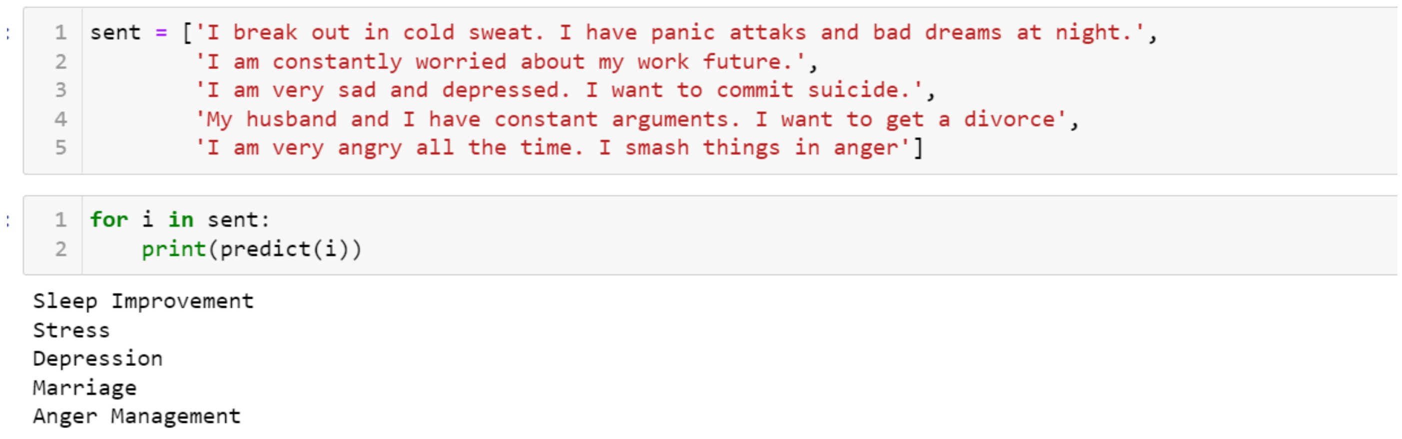


Fig 20 and Fig 21.

1.4.3. Sample Sentence Prediction



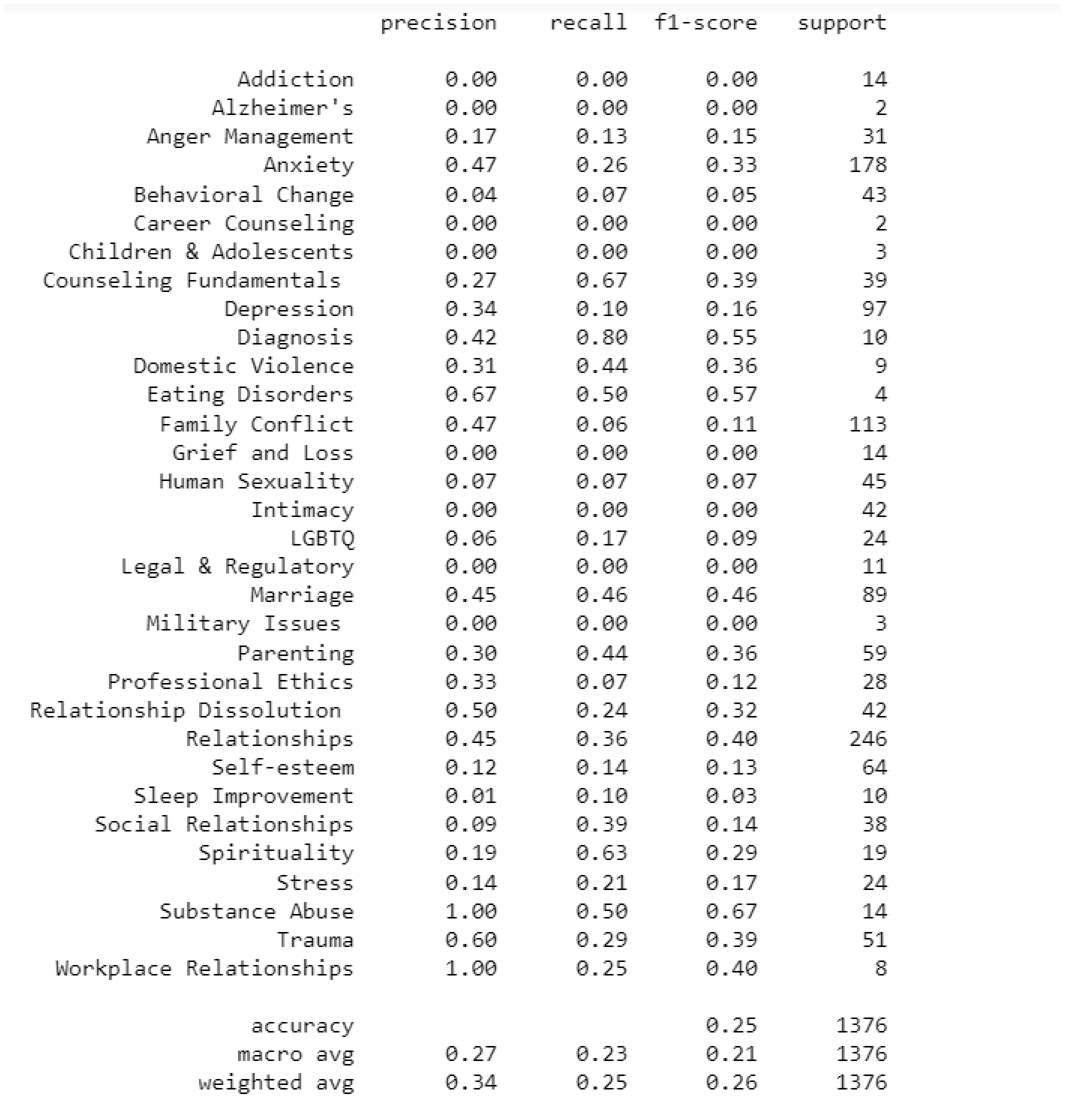
```python
sent = ['I break out in cold sweat. I have panic attaks and bad dreams at night.',
        'I am constantly worried about my work future.',
        'I am very sad and depressed. I want to commit suicide.',
        'My husband and I have constant arguments. I want to get a divorce',
        'I am very angry all the time. I smash things in anger']
```

```python
for i in sent:
    print(predict(i))
```

```
Sleep Improvement
Stress
Depression
Marriage
Anger Management
```

Fig 22.

Classification report



```
                           precision    recall  f1-score   support

               Addiction       0.00      0.00      0.00        14
             Alzheimer's       0.00      0.00      0.00         2
        Anger Management       0.17      0.13      0.15        31
                 Anxiety       0.47      0.26      0.33       178
       Behavioral Change       0.04      0.07      0.05        43
       Career Counseling       0.00      0.00      0.00         2
  Children & Adolescents       0.00      0.00      0.00         3
 Counseling Fundamentals       0.27      0.67      0.39        39
              Depression       0.34      0.10      0.16        97
               Diagnosis       0.42      0.80      0.55        10
       Domestic Violence       0.31      0.44      0.36         9
        Eating Disorders       0.67      0.50      0.57         4
         Family Conflict       0.47      0.06      0.11       113
          Grief and Loss       0.00      0.00      0.00        14
          Human Sexuality       0.07      0.07      0.07        45
                Intimacy       0.00      0.00      0.00        42
                   LGBTQ       0.06      0.17      0.09        24
      Legal & Regulatory       0.00      0.00      0.00        11
                Marriage       0.45      0.46      0.46        89
         Military Issues       0.00      0.00      0.00         3
               Parenting       0.30      0.44      0.36        59
     Professional Ethics       0.33      0.07      0.12        28
 Relationship Dissolution       0.50      0.24      0.32        42
           Relationships       0.45      0.36      0.40       246
             Self-esteem       0.12      0.14      0.13        64
       Sleep Improvement       0.01      0.10      0.03        10
    Social Relationships       0.09      0.39      0.14        38
            Spirituality       0.19      0.63      0.29        19
                  Stress       0.14      0.21      0.17        24
         Substance Abuse       1.00      0.50      0.67        14
                  Trauma       0.60      0.29      0.39        51
  Workplace Relationships       1.00      0.25      0.40         8

                accuracy                           0.25      1376
               macro avg       0.27      0.23      0.21      1376
            weighted avg       0.34      0.25      0.26      1376
```

Fig 22.

9. Conclusion

In conclusion, this project has provided an algorithm that can be used in systems that analyse natural language data to understand patients' mental health issues. Users would simply have to explain their issue as they would to a therapist and the system would process that explanation using NLP techniques and classify them into a particular category. Such a system can be used in multiple areas. A counselling therapy provider could choose to have their patient perform an initial assessment online, which would help them automatically direct the patient towards an appropriate therapist. This means that providers could save on costs related to initial assessments. Second, it could also be used in mental health counselling applications where the system tries to consolidate patients by constantly analysing their natural language descriptions of the issues they are dealing with. Classifying each conversation that patients make within the app would further help the application to maintain overall records of patients' issues. The project highlights the development of a functional AI system for classifying natural language descriptions of mental health issues. The **LSTM model gave 84% accuracy while BERT model gave 59% accuracy.**

8.1. Limitations of the Research and Recommendations

This project provides a baseline for categorising natural language descriptions of mental health issues into categories. As the model presented in this study converged, it became evident that such a system can be very useful in mental health assessment systems.

However, since the **dataset was limited**, the model seems constrained within the trained dataset. Most natural language datasets tend to have 5,000+ samples, with some even including 50,000+ samples. Therefore, the results of this project are considered a starting point to developing a more robust system that would require a larger counselling dataset. Moreover, the categories of mental health issues used within this project were limited since many more categories exist. With a larger dataset, these additional categories could be recognised, which would make the classification model more generalised to new unseen data. Finally, the model could also be tested on new unseen data on a larger scale, which would ensure model validity.

Furthermore, the presented system can be further developed to a more sophisticated system such as a counselling chatbot. In such a system, users could continuously explain their issues within the chat. With each response, the chatbot would identify the issues that the patient is struggling with. Such a system could keep a track of users' issues, which would subsequently be passed on to therapists during counselling.

10. References

[1] Berndtsson, M., Hansson, J., Olsson, B and Lundell, B., 2008. Thesis Projects: A Guide for students in Computer Science and Infromation Systems, London: Springer-Verlag.

[2] Bertagnolli, N., 2020. Counsel Chat: Bootstrapping High-Quality Therapy Data, [online] Available at:

<https://towardsdatascience.com/counsel-chatbootstrapping-high-quality-therapy-data-971b419f33da>

[3] Bigi, B., 2014. A multilingual text normalization approach. Human Language and Technology Challenges for Computer Science and Linguistics, vol. LNAI8387, pp. 515-526.

[4] Dawson, C.W., 2009. Projects in Computing and Information Systems, Harlow: Pearson Education Ltd.

[5] Devlin, J., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, ArXiv, [PDF] Available at: <https://arxiv.org/pdf/1810.04805.pdf>

[6] Ganegedara, T., 2018. Natural Language Processing with Tensorflow. Birmingham: Packt Publishing Ltd.

[7] Geron, A., 2019. Hands-on Machine Learning with Scikit-Learn, Keras and Tensorflow. Sebastopol: O'Reilly Media Inc.

[8] Gonzalez-Carvajal, S. and Garrido-Merchan, E.C., 2020. Comparing BERT against traditional machine learning text classification, ArXiv, [PDF] Available at: <https://arxiv.org/pdf/2005.13012.pdf>

[9] Gopalakrishnan, K. and Salem, F.M., 2020. Sentiment Analysis using Simplified Long Short-term Memory Recurrent Neural Networks, Department of
Electrical and Computer Engineering - Michigan State University, ArXiv,

[PDF] Available at:
<https://arxiv.org/ftp/arxiv/papers/2005/2005.03993.pdf>

[9] Tholusuri, A., Anumala, M., Malapolu, B. and Lakshmi, G.J., 2019. Sentiment Analysis using LSTM, International Journal of Engineering and Advanced Technology, 8(6S3), pp. 1338-1340.

[10] Wang, J., Liu, T., Luo, X and Wang, L., 2018. An LSTM Approach to Short Text Sentiment Classification with Word Embeddings, The ROCLING Conference on Computational Linguistics and Speech Processing, pp. 214-223.