Postgraduate Major Project

(2019 MOD002726 TRI3 F01CAM/TRI3-1 F02CAM)

An LSTM-based approach to the classification of mental health issues using natural language processing

Project Report

SID: 0925739

Acknowledgements

This project has been a result of a yearlong effort to understand the core concepts of AI and its multiple use cases. I would like to first thank my lecturer and supervisor Dr. Lakshmi Babu-Saheer for leading such an excellent course and guiding me throughout my Masters journey. Secondly, I would like to thank the Department of Computing and Technology at Anglia Ruskin University for providing the facilities and materials without which the course would have felt incomplete. Finally, I want to extend my gratitude to my mother, Sharada Kafle Paudyal, who has always been motivating me towards progress in my career.

Abstract

The success of LSTM networks in various natural language processing has been evident in the researches in recent years. This project is an attempt to apply the power of LSTM models to classify the natural language description of a user's mental health issue into pre-defined categories. It is a classification project where the LSTM model will extract the sequential information from the natural language text and use that to classify the samples. The data used for the project contained the text description of mental health issue along with its ground truth category. The text sample itself was subjected to text processing techniques within NLP and LSTM was built on numericized data. The results of the project show that such a system is possible. The model resulted in 73% accuracy before overfitting and also showed promising potential for better result with availability of a larger dataset. Transfer learning using BERT was tried on the dataset too which showed signs of convergence during training but failed to generalise. It was concluded that for a small dataset like the one used for this project simpler models are the go-to option which leaves scope for further research in training BERT on a similar larger dataset.

Table of Contents

List of Figures

1. Introduction

1.1. 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-stigmatisation 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 counselling 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 lay out 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 modelling 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.

1.2. Aims and Objectives

The purpose of this research was to explore sequence modelling 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 modelling 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.

1.3. Proposed Research Questions

- How can NLP analyse the natural language descriptions of patients' mental health to classify them?
- How effective is an LSTM network for analysing natural language text for classification?
- How can transfer learning help in NLP?

1.4. Major Findings

The findings of this project confirm that LSTM networks can indeed provide a framework for classifying natural language text into predefined categories of mental health issues. A bidirectional LSTM model with 100 neurons each was used for sequence modelling on the dataset. The model trained returned 70% accuracy, which is a promising result considering the size of the dataset. However, a larger dataset could result in better convergence. Moreover, transfer learning with a model trained on a large corpus also showed an accuracy of approximately 62%. The Bidirectional

Encoder Representations from Transformers (BERT) model proposed by Devlin et al. (2018) is an effective pre-trained model used to conduct transfer learning in NLP. However, the dataset used in this project seemed too small for the BERT model to fine-tune on, which is why the simpler LSTM model resulted in slightly better performance than transfer learning using BERT.

1.5. Summary of Conclusion

The project concludes by summarising its accomplishments and reflecting on how such systems can be used in mental health analysis. Finally, recommendations are made for further expansion of the present study by using more data samples and advancing use cases of the project in different systems.

2. Literature Review

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 believing 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 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 utilises a few of these tasks, which are described in this section.

2.1. 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 ‘<p>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.

2.2. Tokenization

Webster and Kit (1992) defined tokenization 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 individual words is called tokenization. However, Webster and Kit (1992) argued that some languages may not have delimiters such as spaces, resulting in internal code potentially being used for this purpose. Furthermore, Santos (1990) proposed pragmatic ways to transfer English idioms and fixed expressions for machine translation (MT). Moreover, research by Linden et al. (1990) focused on determining the idiomatic and non-idiomatic meaning of idioms. Additionally, Webster and Kit (1990) argued that taking idioms and fixed expressions as basic units at the same level as words generalised tokenization better, which resulted in more robust NLP and MT systems.

2.3. 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. According to Bigi (2014), 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.

2.4. 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.

2.5. Lemmatization

Similarly, lemmatization is the process of reducing the inflectional forms and sometimes derivationally related forms of a word into its common base form (Manning et al., 2009). 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’.

2.6. POS Tagging

Part-of-Speech (POS) tagging 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 English language NLP for this purpose.

2.7. 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 modelling, 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 modelling 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.

2.9. 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. As popularised by Mikolov et al. (2013), such vectorisation of word tokens results in better performance for trained models when compared to traditional n-gram methods.

Although the vectorisation of word tokens can be performed manually by creating rules based on their features, Python libraries such as Keras, SpaCY and Word2vec facilitate the easy creation of fixed dimensional vectors for word tokens. The word embedding process is based on a theory known as the distributional hypothesis. The hypothesis states that words that co-occur in the same contexts tend to have similar meanings. Therefore, words that exist within the same context are mapped to similar places within the vector space of the word embeddings. As a result, the numerical values within word embedding vectors do not provide information about words; instead, they gather meaning from how similar or dissimilar they are from other words. The cosine distance between similar words will be smaller, while the cosine distance between words with very different contexts will be larger. Therefore, word embedding vectors provide latent information regarding how word tokens are used within a language.

2.10. 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. The idea behind neurons was proposed by McCulloch and Pitts (1943) when they described the nerve cell as a simple logic gate with binary outputs. Moreover, in 1957, Frank Rosenblatt proposed the perceptron learning rule, which would automatically learn the weight coefficients that could be multiplied with the input features, and the product would be used to determine whether neuron fires or not. Neural networks have recently gained considerable popularity in the field of AI and several neural network architectures have emerged for distinct learning tasks. RNNs are a type of neural network that can process inputs of sequences in continuous time-space. Since this project involved analysing the sequences of words (sentences) in the English language to classify them into categories, RNNs were considered the most effective solution (Karpathy, 2015).

2.11. 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.

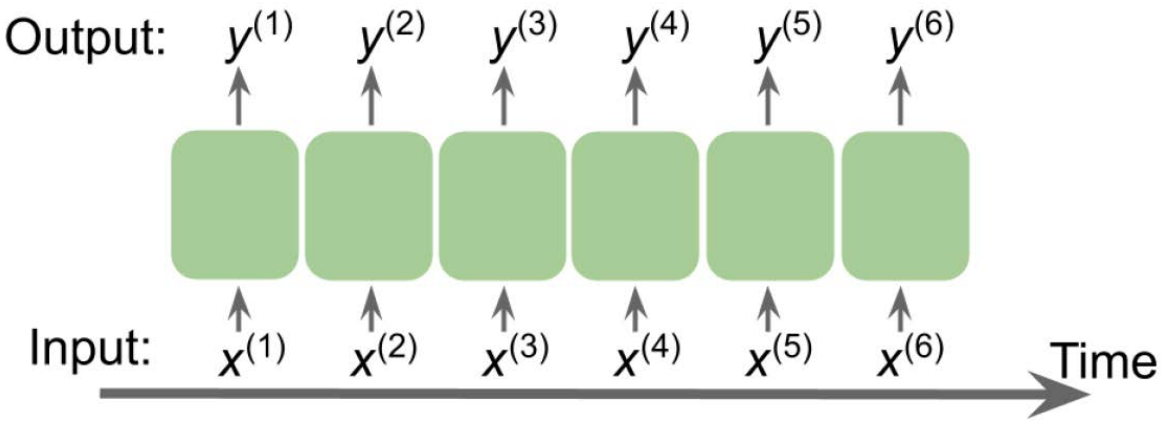

Figure 1: Sequence data (Raschka and Mirjalili, 2019)

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.

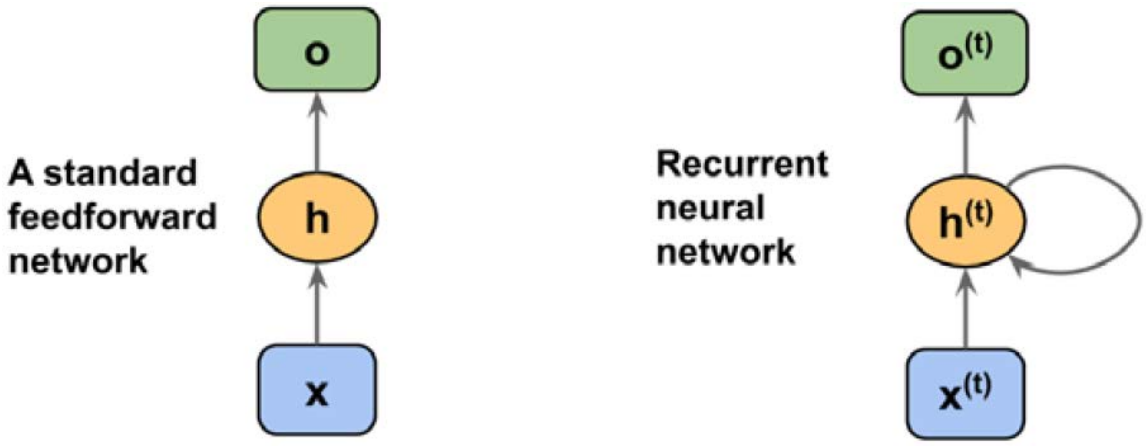

Figure 2: Feed-forward vs recurrent network (Raschka and Mirjalili, 2019)

In Figure 2, x, h and o are the input, hidden and output layers, respectively. Each of these layers are 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.

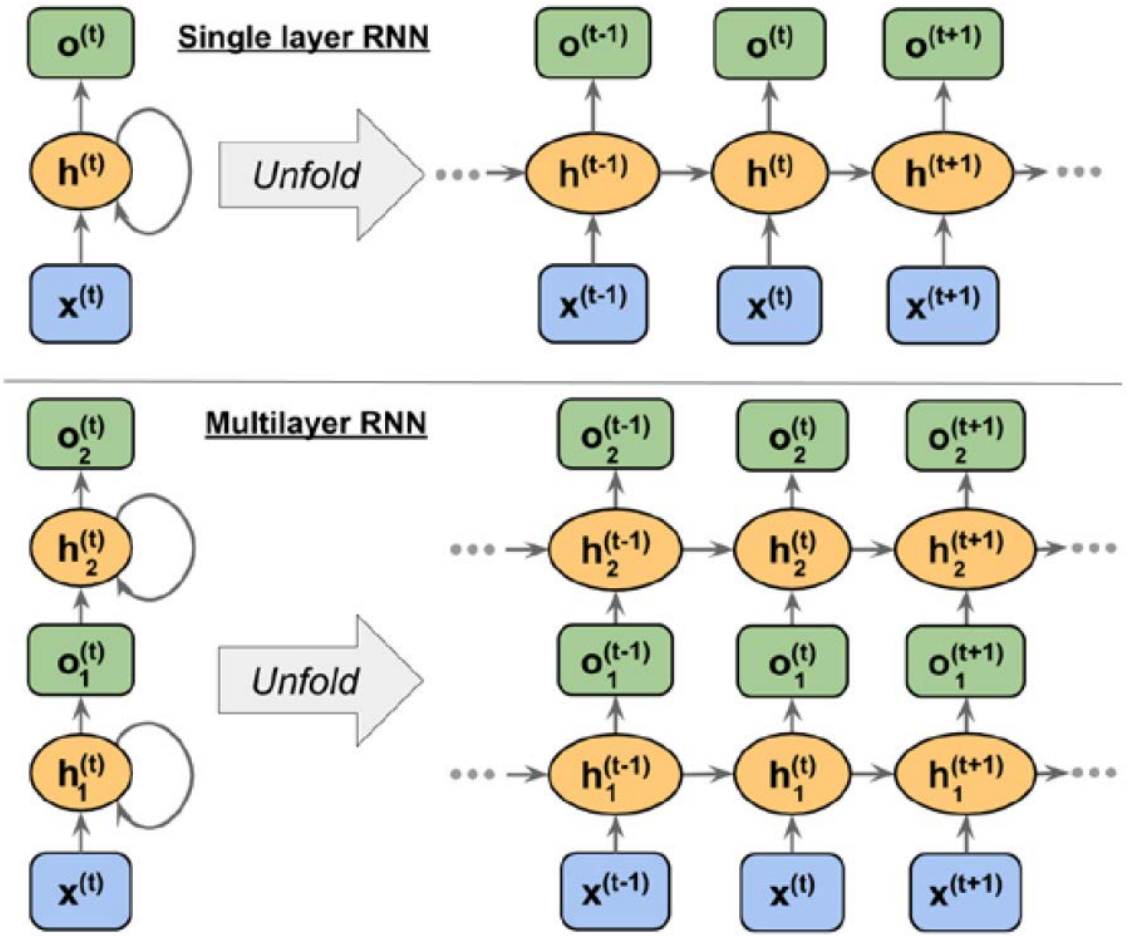

Figure 3: Single-layer and multi-layer RNNs unfolded (Raschka and Mirjalili, 2019)

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 utilises 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.

2.12. 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.

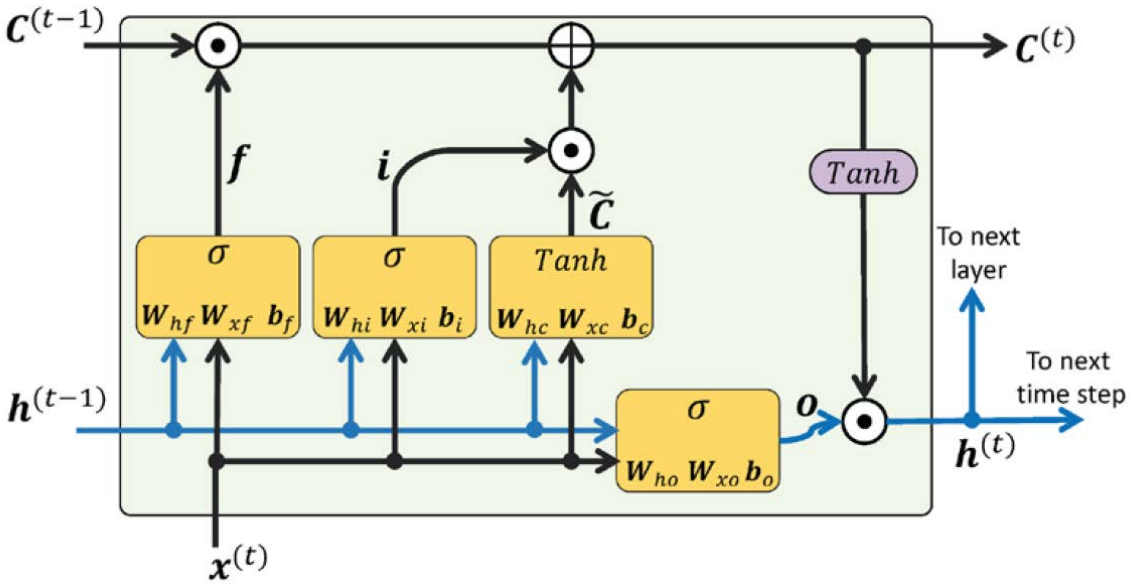

Figure 4: LSTM cell (Raschka and Mirjalili, 2019)

The main working is that the cell state from the previous timestep is modified to obtain the cell state and current timestep. Moreover, the flow of data is not controlled by weight but by computational units known as 'gates'.

The forget gate($\boldsymbol{f_t}$) decides with a sigmoid function which information is allowed to go through, thus stopping it from growing indefinitely.

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hf}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f\right)$$

The input gate ($\boldsymbol{i_t}$) and candidate value ($\tilde{\boldsymbol{C}}_t$) update the cell state($\boldsymbol{C_t}$):

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{xi}\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hi}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i\right)$$

$$\tilde{\boldsymbol{C}}_t = \tanh\left(\boldsymbol{W}_{xc}\,\boldsymbol{x}^{(t)} + \boldsymbol{W}_{hc}\,\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_c\right)$$

$$\boldsymbol{C}^{(t)} = \left(\boldsymbol{C}^{(t-1)} \odot \boldsymbol{f}_t\right) \oplus \left(\boldsymbol{i}_t \odot \tilde{\boldsymbol{C}}_t\right)$$

Finally, the output gate ($\boldsymbol{o_t}$) decides the updates on the hidden units at the current timestep.

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{xo}\,\boldsymbol{x}^{(t)} + \boldsymbol{W}_{ho}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o\right)$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{C}^{(t)}\right)$$

(Raschka and Mirjalili, 2019).

2.13. Activation Functions

Activation functions are used to finalise 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, whose equations are as follows.

Rectified linear unit (ReLU)

$$ReLU(z) = \begin{cases} 0, & z < 0 \\ z, & z \geq 0 \end{cases} = \max(0, z)$$

Hyperbolic tangent (tanh)

$$tanh(z) = \frac{\sinh(z)}{\cosh(z)} = \frac{e^{2x} - 1}{e^{2x} + 1}$$

Softmax

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

2.14. 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 backwards 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.

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

2.15. Optimisers

Optimisation algorithms are widely used within the ML community since they help to minimise or maximise objective functions. Since neural networks are considered universal function approximators, several optimisation algorithms are proposed to help optimise parameters within the neurons of neural networks. Optimisers determine how quickly the loss of a neural network is driven towards its global function minima (Raschka and Mirjalili, 2019). Some examples of optimisers are gradient descent, stochastic gradient descent, Adam and RMSprop.

2.16. 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 generalise 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 generalise 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.

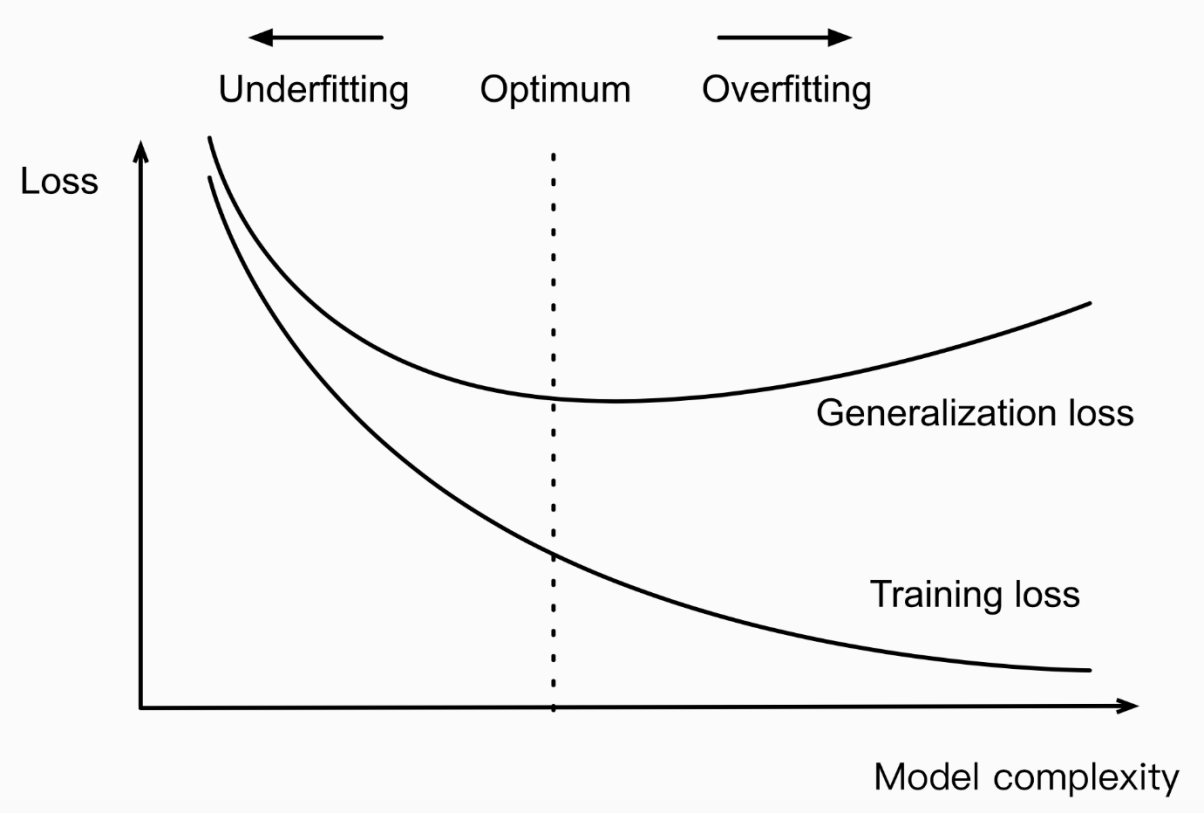

Figure 5: Overfitting, underfitting and optimum

2.17. 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 generalise 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.

2.17.1. 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:

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

2.17.2. 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:

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

2.17.3. Recall (sensitivity)

Recall or sensitivity is the measure of how effectively a model predicts positive outcomes correctly over all 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 recall is as follows:

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

2.17.4. 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 F1 score is as follows:

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

2.18. 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 generalisation in another domain.

2.18.1. 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 optimisation.

2.19. Related Work

The use of LSTM cells has gained considerable popularity in NLP due to the feasibility of processing longer sequences and currently represents the most widely used state-of-the-art solution to many NLP applications. 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.

One of the main applications of LSTM is sentiment analysis (also known as opinion mining), which involves the use of text analysis, computational linguistics and biometrics to systematically identify, extract, quantify and study affective states and subjective information within a block of text written in natural language. Sentiment analysis is a classification task that takes natural language data as input and outputs pre-defined categories of sentiment. It is extensively used in data mining, web mining and social media analytics to understand human behaviour. Sentiments can be positive, negative, neutral or an arithmetic score that measures the effectiveness of the sentiment. Tholusuri et al. (2019) demonstrated that using an LSTM network for sentiment analysis resulted in better performance than any other ML models in the IMDB movie reviews dataset. Similarly, 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 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.

Another paper by Gopalakrishnan and Salem (2020) investigated the use of various parameter-reduced LSTM models, which they called slim-LSTM. They further evaluated the effect of using bidirectional methods within these models. With similar pre-processing, they evaluated the results in all their proposed slim-LSTMs, which appeared better than standard LSTM cells. They stated that the dataset size and balance of distribution affected their dataset and recommend artificially balancing training datasets for improved performance. Moreover, they recommended the use of both RMSprop and Adam optimisers and did not recommend adding multiple dense layers after the LSTM layer within networks.

Similarly, another application of LSTM is emotion recognition, which is also a classification task. Su et al. (2018) noted that the LSTM-based method achieved a recognition accuracy of 70.66%, which was better than the CNN-based methods it was compared to. Their model employed semantic as well as emotional word vectors for each word. Semantic word vectors were derived using a simple word2vec model. For emotional word vectors, they projected each word to all emotional words defined in an affective lexicon and applied an autoencoder on it for dimensionality reduction. Then, the concatenated semantic and emotional vectors were passed on as inputs to the LSTM network for classification. They concluded that the concatenated vector features outperformed the use of individual feature vectors.

Finally, another similar application of the LSTM network involves classifying consumer complaints into their departmental categories within a large company. A Medium article by Li (2019) explored the US Consumer Finance Complaints dataset provided in Kaggle to successfully implement an LSTM network that classified consumer complaints into 18 categories. The dataset was considerably large (361,574 entries) and the trained LSTM network returned 82.4% accuracy on the test set. The model was a simple one-directional LSTM with variational dropout for NLP added after embedding and the LSTM layer had 100 cells with regular and recurrent dropouts to compensate overfitting.

3. Research Methodology and Implementation

This section will explore different types of research methodologies used within academia with a particular focus on the methodologies used for this project. According to Berndtsson et al. (2008, p.10), research is considered the activity of a diligent and

systematic investigation of a particular area to discover or revise facts, theories, applications, etc. There are two main methods of research used in practice: quantitative research and qualitative research. Quantitative methods rely on numerical data upon which a hypothesis is set and attempts to falsify the hypothesis are exercised. The hypothesis is considered a true hypothesis if it withstands the falsification attempts until proven otherwise. This method helps with attaining an understanding of how something is constructed, how it is built or how it works. Dawson (2009) emphasised the importance of ensuring the repeatability of experiments and testing hypotheses so that results are reliable and opportunities for scrutinising findings are provided.

Similarly, Berndtsson et al. (2008) described qualitative methods as those methods that help with 'increasing the understanding of a substantive area rather than producing an explanation for it'. Qualitative methods apply analyses that involve the investigation and interpretation of human or organisational aspects concerning technology. Two examples of such methods are case studies and surveys.

Since this project is an attempt to classify user-given text into different categories, it is mostly quantitative research. However, due to considerations related to language use, syntactic information could also fall under qualitative analysis. However, most of the analysis was performed within a classification algorithm where all information extracted from the user-given text was converted to numerical data to be analysed by the classification model. The analysis involved checking whether the model made predictions correctly, which was performed by numerically comparing the accuracy scores. Therefore, this project is considered a quantitative research project. The project utilised freely available natural language datasets and the neural networks will be trained with random seeds set at the build stage so the results are reproducible.

The approach taken to collect data was an online search of counselling conversations between a mental health patient and a therapist. It was quickly realised during the search process that such a dataset is rare due to the ethical implications it carries. Counselchat.com is an online counselling service that provides users with therapists' advice based on their own explanation of their issue. As such, it collected the natural language descriptions of patients' mental health issues. The dataset used in this project was provided to ML engineer Nicolas Bertagnolli by the founders of

Counselchat.com (Bertagnolli, 2020). Nicholas himself has conducted several analyses on the dataset, which can be seen in his repository. For this project, only two columns within the dataset—namely 'questionText' and 'topics'—were used. The 'questionText' column contains the natural language text provided by patients and the 'topics' column contains the category of mental health issue it belongs to.

3.1. Ethical Considerations

Ethical considerations have become a significant issue in data analysis and AI. Due to the increase in data collected online and the power that data can provide to the companies that collect them, governments worldwide have implemented data protection regulations to protect people's privacy rights from being violated. In the EU, the General Data Protection Regulation (GDPR) governs data protection, which applies to the UK. However, this may change once the UK leaves the EU. The GDPR specifically protects individuals from their personal data being used without their consent. Moreover, if datasets have any information that can eventually lead to personal information (e.g. name, age, address and city), then such information should be anonymised before use in data analysis or ML algorithms (Gruschka et al., 2018). This is to ensure that individuals are not targeted based on the findings from the data.

The dataset for this project was provided by the founders of Counselchat.com to Nicolas Bertagnolli, who is an ML Engineer. The dataset only has an identifier for the patients and not their personal or sensitive information. However, the dataset contains personal information about the therapists who provided counselling advice to those patients. Since this project pertains to analysing patients' descriptions of their issues, the information related to therapists—and even their responses—were considered unnecessary. Therefore, the columns containing that information were removed before conducting any analysis. The only two columns used for the project were the text description of the mental health issue written by the patient (i.e. the input) and the category that the issue falls under, which is the ground truth target that the model attempted to classify the text into.

As such, this project complies with the GDPR and the ethical considerations necessary for the project were fulfilled.

4. Development / Analysis

4.1. System Design

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.

4.2. 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.

```
print(chatdf.columns)
```

```
Index(['questionID', 'questionTitle', 'questionText', 'questionUrl', 'topics',
       'therapistName', 'therapistUrl', 'answerText', 'upvotes'],
      dtype='object')
```

Figure 6: Dataset columns

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 7.

```
counseldf.head()
```

	questionText	topics
0	My wife and mother are having tense disagreeme...	Family Conflict
1	I'm planning to have baby, so I have to quit s...	Substance Abuse,Addiction
2	I have secrets in my mind, and I don't know wh...	Family Conflict
3	I am extremely possessive in my relationships ...	Behavioral Change,Social Relationships
4	I had a head injury a few years ago and my min...	Anxiety

Figure 7: Data retained for the project (first five lines)

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.

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 (see Figure 8).

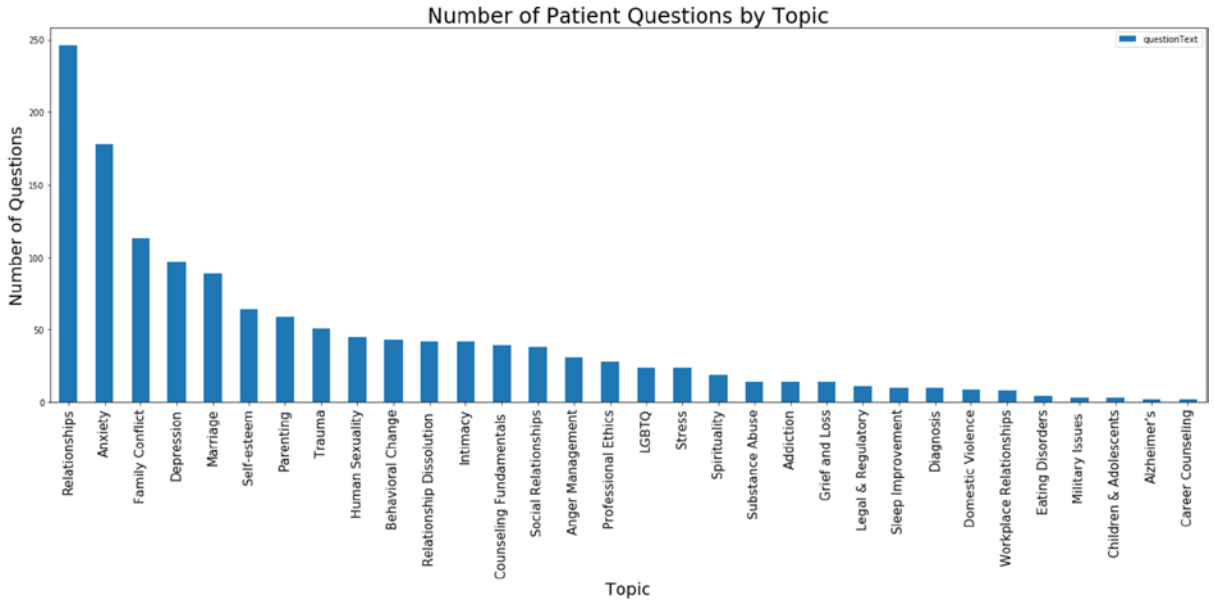

Figure 8: Number of patient questions by topic

As seen in Figure 8, 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.

4.3. Text Processing

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

4.3.1. 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 9 demonstrates how noise removal affects a data sample.

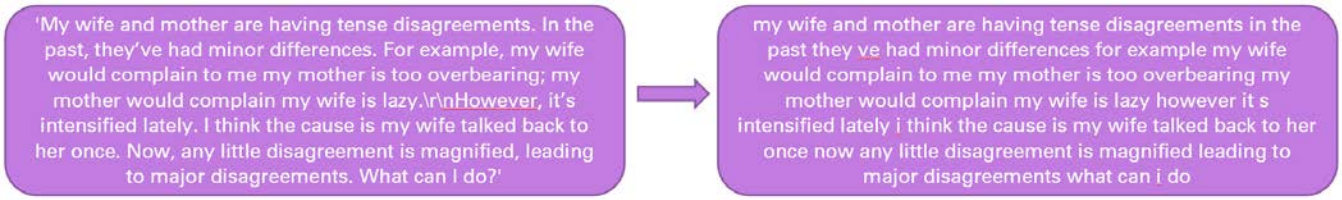

Figure 9: Noise removal in the first sample

4.3.2. Tokenization

Following noise removal, tokenization was performed for each sample. This process simply involved breaking the text into its word tokens. Figure 10 provides a tokenized version of the same example (first sample).

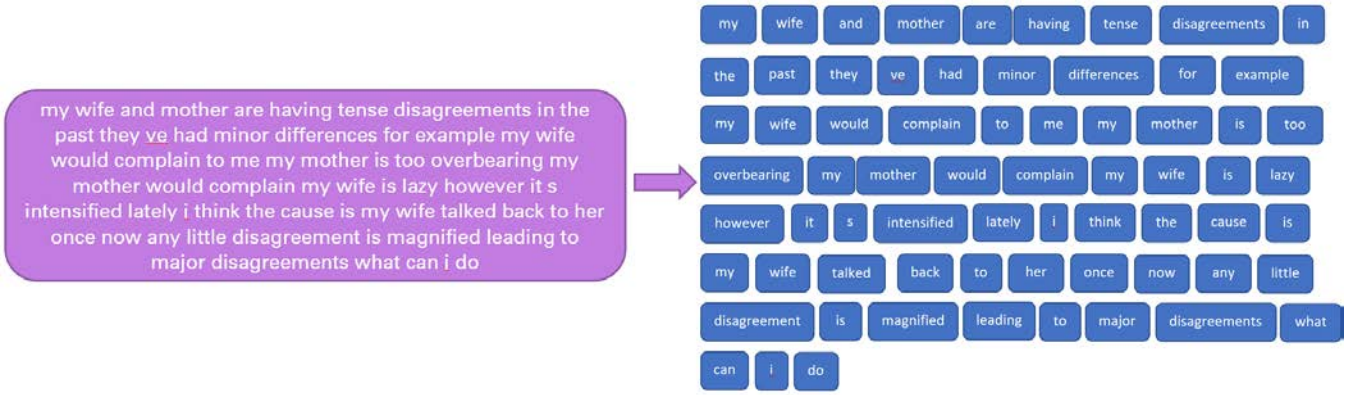

Figure 10: Tokenization in the first sample

4.3.3. 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.

Figure 11: Lemmatization in the first sample

4.3.4. 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 12 shows the stop word removal performed for the first sample.

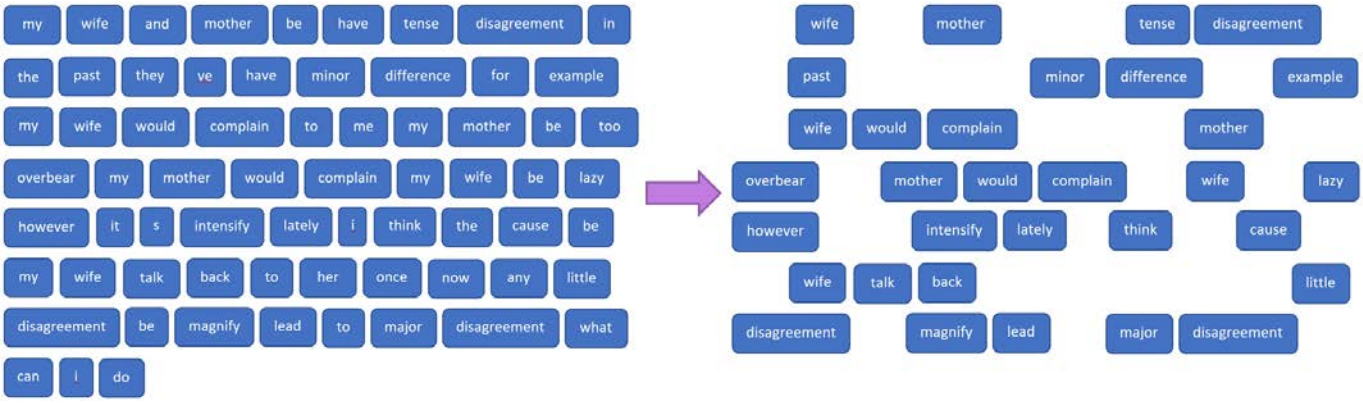

Figure 12: Stop words removed from the first sample

4.3.5. 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 pre-processing library. The text tokenizer within the Keras pre-processing 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 13.

Dictionary: {'feel': 1, 'get': 2, 'want': 3, 'like': 4, 'know': 5, 'time': 6, 'go': 7, 'year': 8, 'think': 9, 'say': 10, 'make': 11, 'tell': 12, 'relationship': 13, 'love': 14, 'never': 15, 'really': 16, 'friend': 17, 'talk': 18, 'thing': 19, 'always': 20, 'even': 21, 'work': 22, 'help': 23, 'life': 24, 'boyfriend': 25, 'try': 26, 'find': 27, 'still': 28, 'see': 29, 'one': 30, 'day': 31, 'people': 32, 'need': 33, 'start': 34, 'sex': 35, 'anxiety': 36, 'take': 37, 'family': 38, 'lot': 39, 'month': 40, 'back': 41, 'child': 42, 'much': 43, 'every': 44, 'husband': 45, 'anything': 46, 'stop': 47, 'together': 48, 'past': 49, 'live': 50, 'something': 51, 'would': 52, 'someone': 53, 'leave': 54, 'move': 55, 'problem': 56, 'sometimes': 57, 'two': 58, 'bad': 59, 'school': 60, 'ago': 61, 'lose': 62, 'parent': 63, 'ask': 64, 'also': 65, 'everything': 66, 'girl': 67, 'wife': 68, 'come': 69, 'well': 70, 'keep': 71, 'issue': 72, 'seem': 73, 'break': 74, 'depression': 75, 'cheat': 76, 'dad': 77, 'use': 78, 'happen': 79, 'way': 80, 'wrong': 81, 'stress': 82, 'give': 83, 'away': 84, 'u': 85, 'marry': 86, 'last': 87, 'ex': 88, 'since': 89, 'call': 90, 'guy': 91, 'hurt': 92, 'right': 93, 'normal': 94, 'anymore': 95, 'recently': 96, 'good': 97, 'drink': 98, 'mom': 99, 'night': 100, 'mother': 101, 'end': 102, 'stay': 103, 'date': 104, 'girlfriend': 105, 'home': 106, 'trust': 107, 'woman': 108, 'lie': 109, 'first': 110, 'job': 111, 'cry': 112, 'let': 113, 'deal': 114, 'long': 115, 'believe': 116, 'else': 117, 'week': 118, 'care': 119, 'counseling': 120, 'depress': 121, 'late': 122, 'alone': 123, 'around': 124, 'change': 125, 'happy': 126, 'fight': 127, 'could': 128, 'thought': 129, 'person': 130, 'daughter': 131, 'nothing': 132, 'ever': 133, 'kid': 134, 'afraid': 135, 'angry': 136, 'decide': 137, 'listen': 138, 'without': 139, '20': 140, 'hard': 141, 'meet': 142, 'young': 143, 'couple': 144, 'almost': 145, 'though': 146, 'three': 147, 'felt': 148, 'enough': 149, 'lately': 150, 'win': 151, 'disorder': 152, 'therapy': 153, 'become': 154, 'self': 155, 'men': 156, 'everyone': 157, 'great': 158, 'worry': 159, 'scar': 160, 'therapist': 161, 'attack': 162, 'house': 163, 'understand': 164, 'however': 165, 'little': 166, 'act': 167, 'far': 168, 'new': 169, 'look': 170, 'upset': 171, 'anger':

Figure 13: Word index dictionary (first 172 tokens and their indices)

As displayed in Figure 13, 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. Figure 14 presents the conversion of the sample.

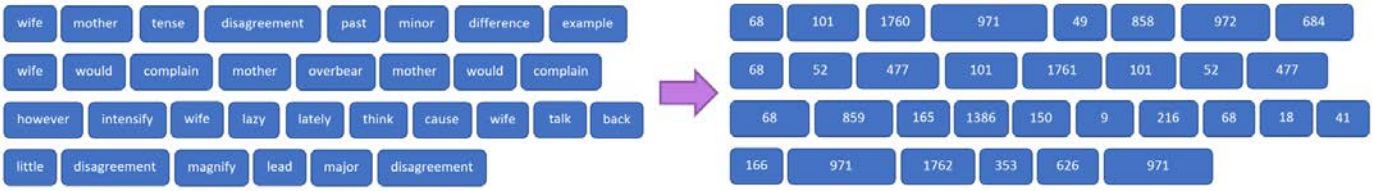

Figure 14: Tokens converted to numerical data

4.3.6. 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 15 presents the first sample padded with zeros.

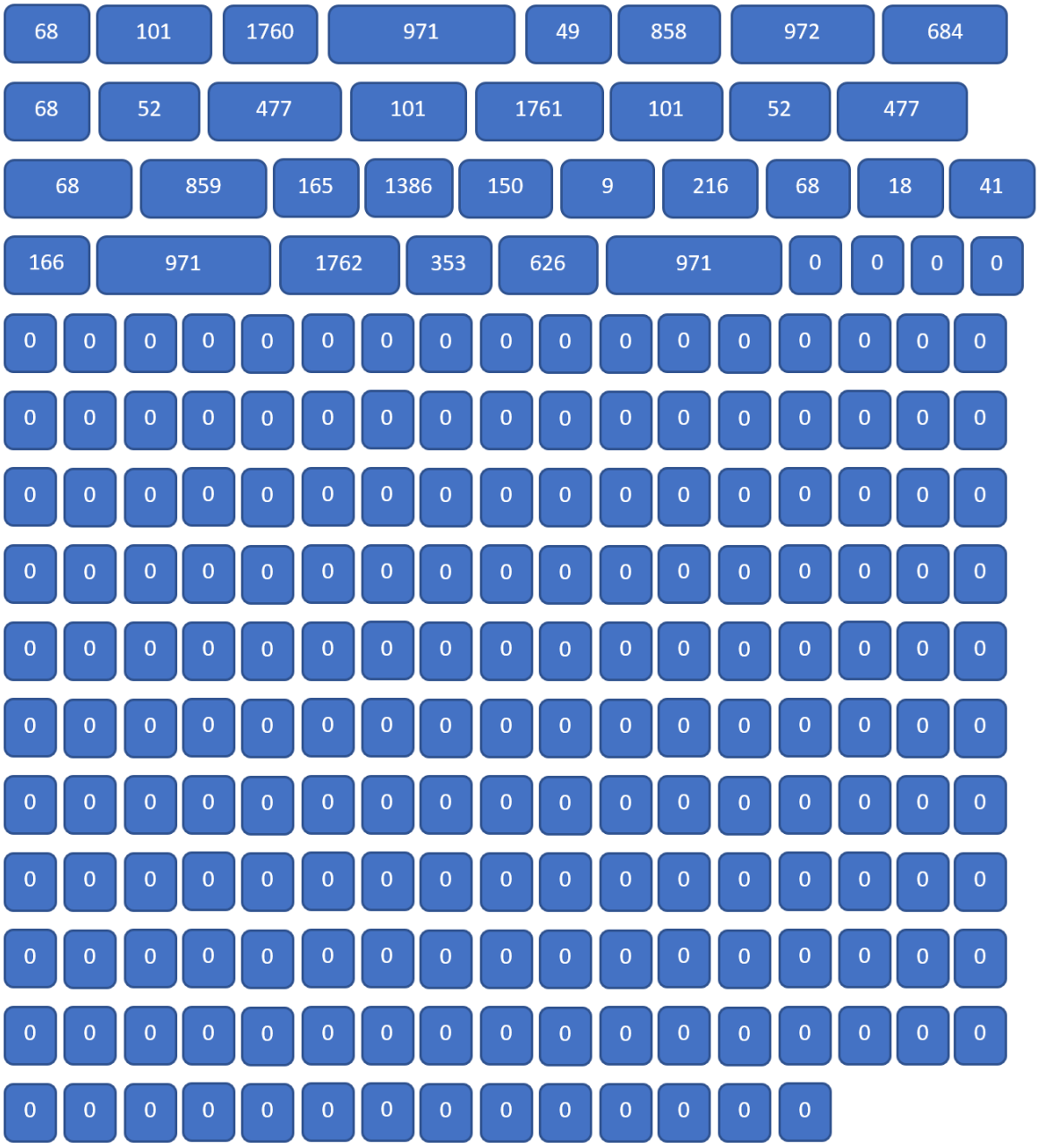

Figure 15: Padding with zeros performed for the first sample

4.3.7. 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. Figure 16 provides an example of embedding applied to the first sample.

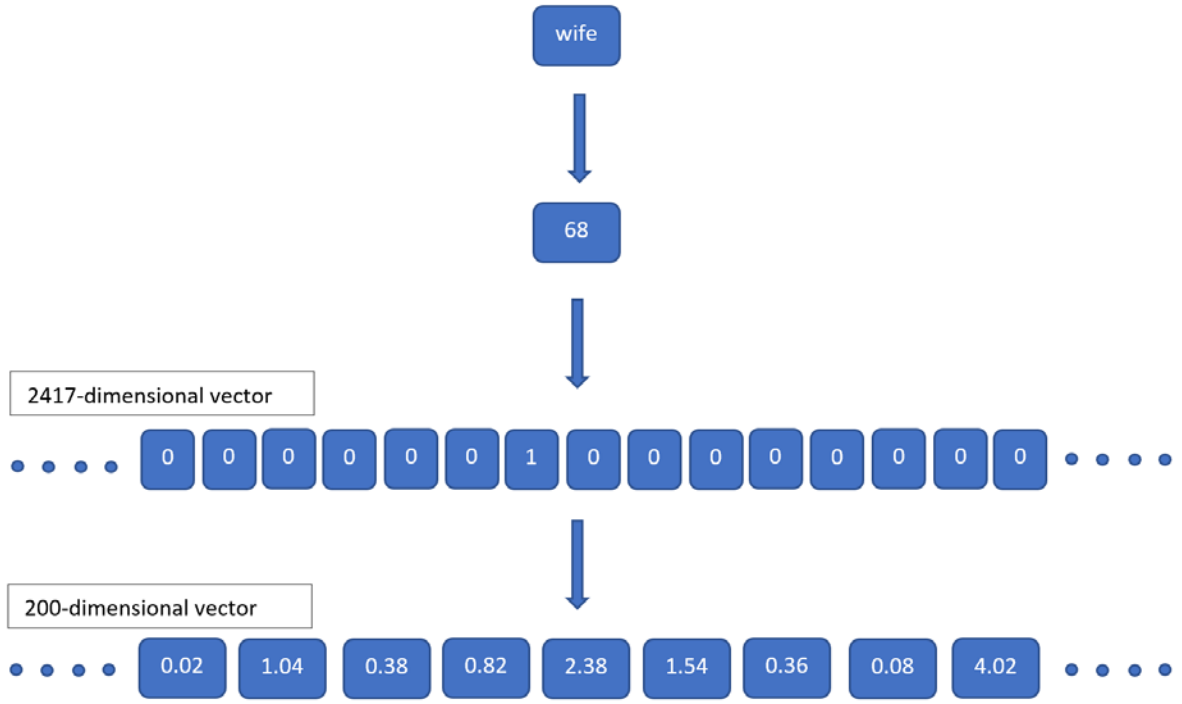

Figure 16: Word embedding vector shown for the first sample

4.4. 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 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, 840,732 total weight parameters were optimised during training. Figure 17 presents the code and model summary for the neural network design.

```
inputs = tf.keras.layers.Input(shape=(None,))
embedding = tf.keras.layers.Embedding(num_question_tokens, 200, mask_zero=True)(inputs)
lstm_outputs = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(100, return_state=False), name='bidir') (
embedding)
dense_outputs = tf.keras.layers.Dense(500, activation=tf.keras.activations.relu) (lstm_outputs)
outputs = tf.keras.layers.Dense(32, activation=tf.keras.activations.softmax) (dense_outputs)
model = tf.keras.models.Model(inputs, outputs)
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, None)]            0
_________________________________________________________________
embedding_1 (Embedding)      (None, None, 200)         483400
_________________________________________________________________
bidir (Bidirectional)        (None, 200)               240800
_________________________________________________________________
dense_2 (Dense)              (None, 500)               100500
_________________________________________________________________
dense_3 (Dense)              (None, 32)                16032
=================================================================
Total params: 840,732
Trainable params: 840,732
Non-trainable params: 0
_________________________________________________________________
```

Figure 17: LSTM model summary

4.5. Results

4.5.1. LSTM training

The model was trained for 50 epochs. The training accuracy continuously increased and ultimately reached 100%. However, the test accuracy stabilised at approximately 70%. 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.

The graph for the loss plot is presented in Figure 18. As shown in this figure, the model starts to overfit to the training data just before 20th epoch. The lowest point for validation loss is considered the optimum point.

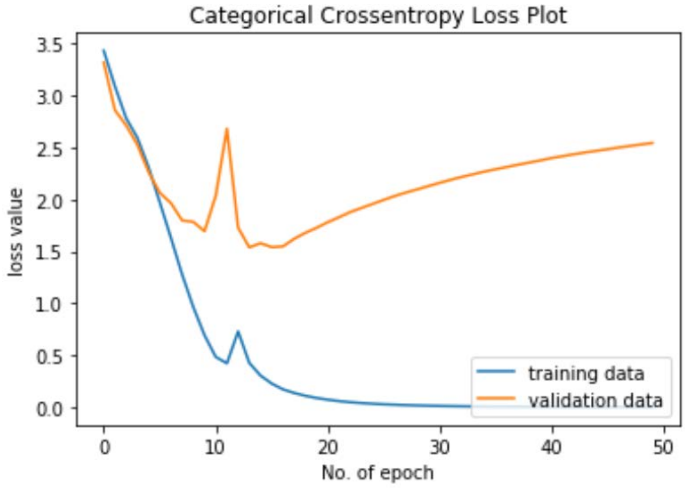

Figure 18: Categorical cross-entropy loss plot for LSTM model training

Several classification metrics were applied to the results and the graphs are presented in Figure 19.

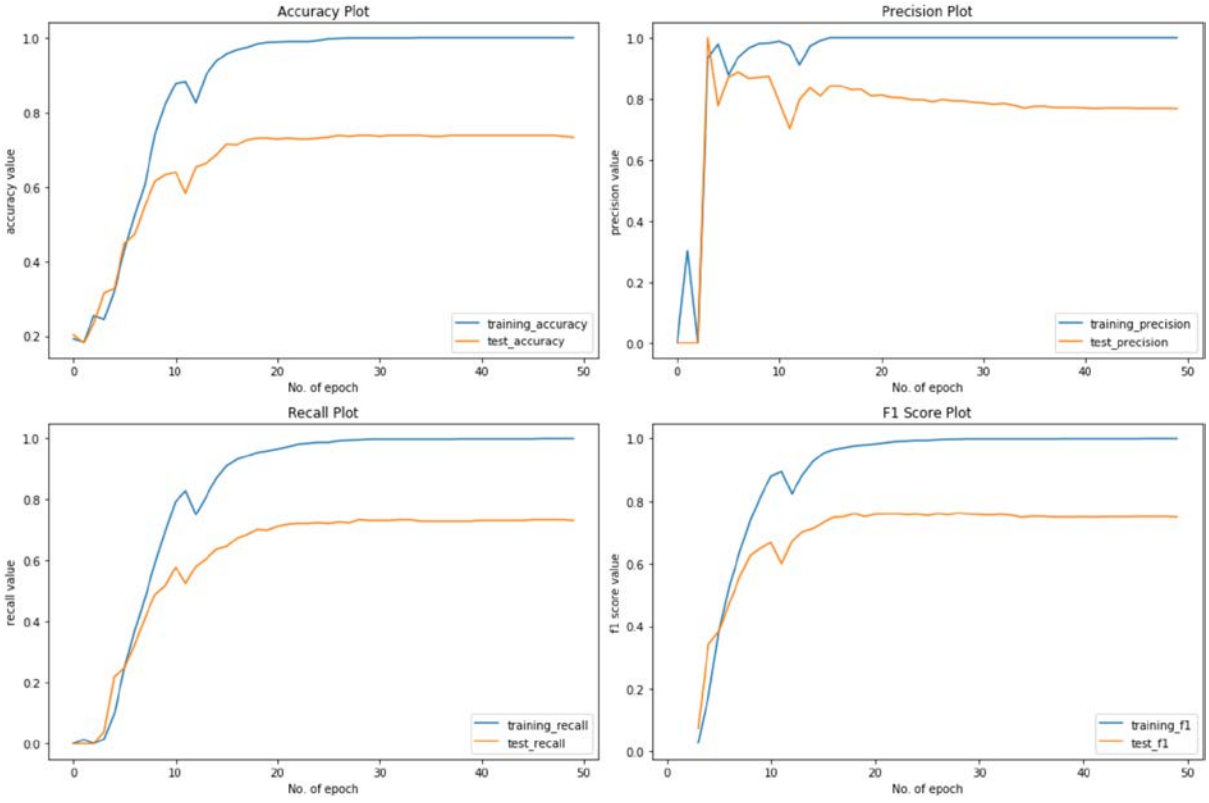

Figure 19: Classification metrics for LSTM model training

The plots shown in Figure 19 indicate that the model adopts a smooth convergence around 20th epoch. This is further supported by the precision, recall and F1 score plots, which indicate that the model can truly identify the classes.

4.5.2. Further training using BERT

The BERT model has its own pre-processing pipeline for which the training data must be processed in an acceptable form. The BERT model requires each sentence to be bound by a 'CLS' token at the beginning and a 'SEP' token at the end of the sentence. Moreover, the model itself requires three inputs: tokens, masks and segment flags. Tokens contain the post-padded tokens of each sample. Masks contain the values '1' for non-zero tokens and '0' for zeroed tokens added during padding. Segment flags contain markers for the number of sentences within a text such as '1' for the first sentence, '2' for the second sentence and so on (Sun et al., 2020).

The dataset within this project was converted accordingly and a BERT layer downloaded via Tensorflow was incorporated within the neural network design. The trainable parameter of the BERT layer was set to 'false'. Training using the BERT layer took longer than the normal LSTM and the results indicated that the BERT-incorporated neural network performed similarly to the LSTM model with an optimal validation accuracy of 68.95%.

Figure 20 presents the loss plot using BERT. Although the model seems to converge smoothly, the validation loss differed greatly from the training loss. This could be due to limited data samples being available. Training with larger datasets may result in better convergence.

Figure 20: Categorical cross-entropy loss plot for BERT model training

Similarly, Figure 21 presents the graphs for classification metrics attempted on the BERT model. As shown in the graphs, the model's accuracy score is also not high. As

such, it cannot correctly identify the categories for both training and validation data. This is further evidenced by the jagged plots for precision, recall and F1 score.

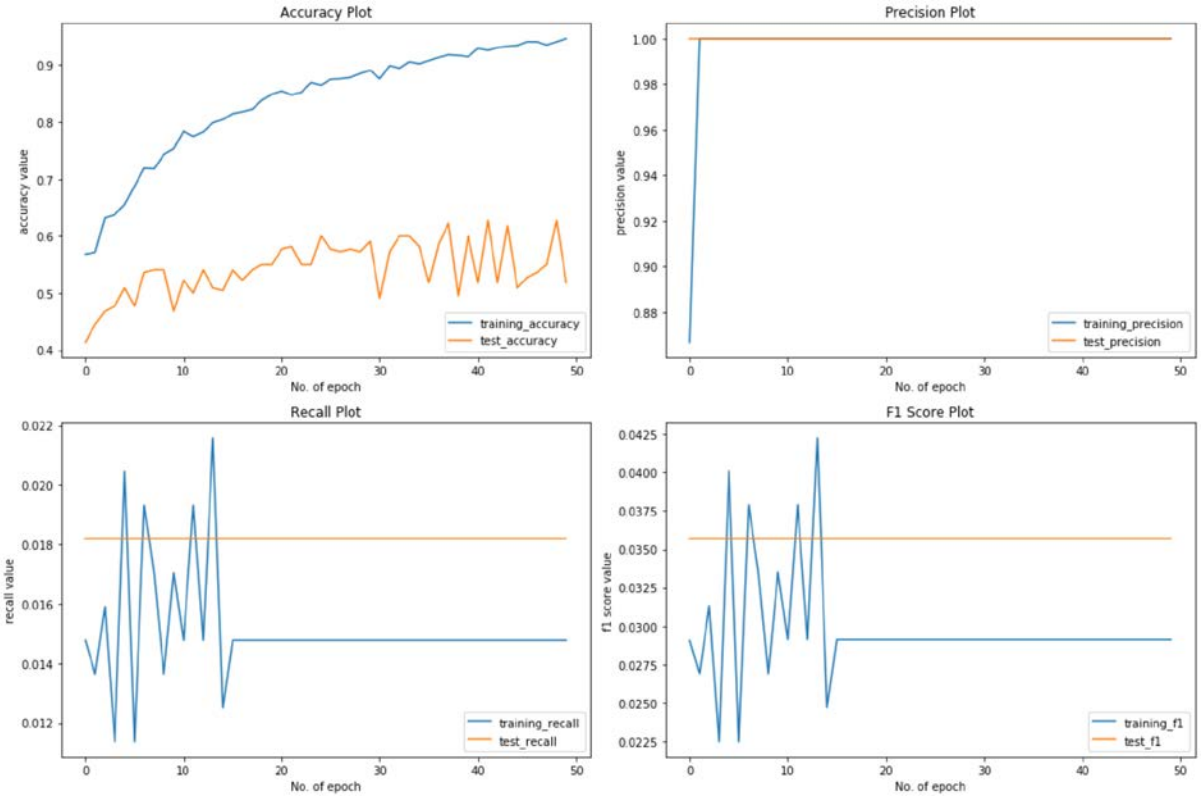

Figure 21: Classification metrics for BERT model training

Although the LSTM model performed slightly better in terms of validation accuracy, the way the neural network converged appears slightly different. The validation data result for the BERT model appears to have a smoother curve than that of the normal LSTM model. This indicates that with a larger dataset, the BERT-incorporated model could potentially achieve better performance since this is model is designed for large natural language datasets.

4.6. Final Model

For this project, the simpler bidirectional LSTM model performed better than the BERT layer-incorporated model. Therefore, the bidirectional model was compiled again as the final model and trained for up to 20 epochs using the entire dataset without making a validation split. After training with the entire dataset, the model was ready to be used for the classification of new unseen data. A function was created that would take the natural language text given by the user as input and go through all of the text-processing steps performed for the training set. Once this was complete, the numerical

padded token version of the input text was sent to the model for prediction and the argmax function of NumPy was used to locate the predicted category.

Testing with new unseen data was performed using a sample for which a correct prediction was made (see Figure 22).

```
predict('''I fight with my husband everyday. We cannot seem to agree on anything.
        The marriage is failing and I need to now do a divorce. This has affected my mentally.
        I cannot stop but think about all the memories we had and I was about to loose everything that mattered.''')
```

```
Relationships
```

Figure 22: Correct prediction made for a new sample

5. Discussion

NLP is a field within AI that has a wide spectrum of applications. Text classification within NLP provides a wide variety of use cases similar to this project. It could be used for any form of text data that can be divided into categories. This project was an initiative to determine whether user-described text can be classified into categories of mental health issues. Such categorisation allows for mental health practitioners to direct the patient directly towards appropriate therapy.

To process natural language descriptions of mental health issues, the project employed text processing. Text processing allowed the researcher to clean noise from within the samples, break them apart into tokens, lemmatize the tokens into root forms and finally convert the lemmatized tokens to numerical data. Converting to numerical data helped create a feature space for all of the word tokens within the dataset. Once a feature space exists, a learning model can be trained to recognise similarities and differences between tokens. Specifically, word embeddings allowed the model to observe how similar or different words are from each other. Using word embedding vectors, it is also possible to find the cosine distance between two words. Finally, an LSTM model was applied to extract sequential information from within the samples of word embedding vector sequences. To add to the network's ability to analyse sequences more comprehensively, a bidirectional LSTM was implemented that allowed the network to analyse sequences from start to end and vice versa. The output from the bidirectional network was then passed on to a 500-neuron dense layer to further subdivide feature information. Finally, the output of this dense layer was sent to a 32-neuron dense softmax classification layer that facilitated the classification of the samples into the 32 pre-defined categories.

The model was created using the Keras functional API. It allowed the researcher to define how each layer affected the inputs they received. The model was compiled with categorical cross-entropy loss, which is the standard loss function used for multiclass classification. The model was trained for up to 50 epochs to assess how it converged. It was realised that after approximately 20 epochs, the model started to overfit. This was visually observed in the loss plot since validation loss started to increase after 20 epochs, while training loss continued its path downwards. This implies that the model started to overfit to the training set after 20 epochs.

The loss plot and classification metrics plot for the LSTM model suggest that the model has converged and can make correct predictions. The best accuracy recorded before the model started to overfit was approximately 73%. The precision plot shows approximately 80% precision, which implies that 80% of the data was correctly predicted. Similarly, 60% recall was shown. Therefore, the F1 score was approximately 70%, which is a good result for this classification project.

The two models trained in this project both showed promising results. Although the BERT model had a layer trained on a larger corpus of natural language data, it showed lower accuracy than the simpler LSTM model. However, this may not be the case if there were more data to train on since BERT is a transfer learning model designed to be used on large NLP datasets. However, for this project, it was more appropriate for the higher-performing and simpler bidirectional LSTM model to serve as the final model.

The results obtained by the LSTM model in the present study are comparable to those of Su et al. (2018), who achieved 70% accuracy in an LSTM model for emotion detection. Regarding the baseline for the dataset, Nicolas Bertagnolli's analysis showed approximately 68% accuracy for the Support Vector Machine (SVM) model. The LSTM model implemented in this project performed better than what was proposed in his analysis.

Testing performed with new sample data shows that the model can generalise. However, this could be tested with more new samples and additional data could help to improve generalisation.

6. 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.

6.1. 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.

References

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.

Bertagnolli, N., 2020. *Counsel Chat: Bootstrapping High-Quality Therapy Data*, [online] Available at: <https://towardsdatascience.com/counsel-chat-bootstrapping-high-quality-therapy-data-971b419f33da> [Accessed 07/07/2020].

Bigi, B., 2014. A multilingual text normalization approach. *Human Language and Technology Challenges for Computer Science and Linguistics*, vol. LNAI-8387, pp. 515-526.

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

Devlin, J., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *ArXiv*, [PDF] Available at: <https://arxiv.org/pdf/1810.04805.pdf> [Accessed 03/10/2020].

Ganegedara, T., 2018. *Natural Language Processing with Tensorflow*. Birmingham: Packt Publishing Ltd.

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

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> [Accessed 01/10/2020].

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> [Accessed 02/10/2020].

Gruschka, N., Mavroeidis, V., Vishi, K. and Jensen, M., 2018. Privacy Issues and Data Protection in Big Data: A Case Study Analysis under GDPR, *ArXiv*, [PDF]. Available at: <https://arxiv.org/pdf/1811.08531.pdf> [Accessed on: 11 December 2019].

Hochreiter, S. and Schmidhuber, J., 1997. Long Short-Term Memory. *Neural Computation*, 9(8), pp. 1735-1780.

Karpathy, A., 2015. The Unreasonable Effectiveness of Recurrent Neural Networks. *Andrej Karpathy Blog*, [PDF] Available at: <https://web.stanford.edu/class/cs379c/archive/2018/class_messages_listing/content/Artificial_Neural_Network_Technology_Tutorials/KarparthyUNREASONABLY-EFFECTIVE-RNN-15.pdf> [Accessed 08/08/2020].

Knesebeck, O., Lehmann, M., Lowe, B and Makowski, A.C., 2018. Public stigma towards individuals with somatic symptom disorders – Survey results from Germany, *Journal of Psychosomatic Research*, vol. 115, pp. 71-75.

Li, S., 2019. *Multi-Class Text Classification with LSTM*. [online] Available at: <https://towardsdatascience.com/multi-class-text-classification-with-lstm-1590bee1bd17> [Accessed 02/10/2020].

Linden, E. van der, and Kraaji, W., 1990. Ambiguity resolution and the retrieval of idioms: two approaches. *Proceedings of COLING-90*, Helsinki, Finland.

Manning, C.D., Raghavan, P. and Schutze, H., 2009. *Introduction to Information Retrieval*. Cambridge: Cambridge University Press.

McCulloch, W.S. and Pitts, W., 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity, *Bulletin of Mathematical Biophysics*, 5(4), pp. 115-133

Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013. Efficient Estimation of Word Representations in Vector Space, *ArXiv*, [PDF] Available at: <https://arxiv.org/pdf/1301.3781.pdf> [Accessed 01/10/2020].

Princeton University, 2020. *WordNet: A Lexical Database for English*, [online] Available at: <https://wordnet.princeton.edu/> [Accessed 02/10/2020].

Raschka, S. and Mirjalili, V., 2019. *Python Machine Learning*. Birmingham: Packt Publishing Ltd.

Rosenblatt, F. 1957. The Perceptron: A Perceiving and Recognizing Automation, *Cornell Aeronautical Laboratory*, Report 85-60-1, Buffalo, NY.

Santos, D., 1990. Lexical gaps and idioms in machine translation, *Proceedings of COLING-90*, Helsinki, Finland.

Su, M., Wu, C., Huang, K and Hong, Q., 2018. LSTM-based Text Emotion Recognition using Semantic and Emotional Word Vectors, *First Asian Conference on Affective Computing and Intelligence Interaction*, [PDF] Available at: <https://www.researchgate.net/publication/327852974_LSTM-based_Text_Emotion_Recognition_Using_Semantic_and_Emotional_Word_Vectors> [Accessed on 02/10/2020].

Sun, C., Qiu, X., Xu, Y. and Huang, X., 2020. How to Fine-Tune BERT for Text Classification? *ArXiv*, [PDF] Available at: https://arxiv.org/pdf/1905.05583.pdf [Accessed 03/10/2020].

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.

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.

Webster, J.J. and Kit, C., 1992. Tokenization as the initial phase in NLP, *Proceedings of the 14th conference on Computational Linguistics*, August, pp, 1106-1110.

WHO, 2020. *Depression-Fact Sheets*, [online] Available at: <https://www.who.int/news-room/fact-sheets/detail/depression> [Accessed 28/09/2020].

Young, T., Hazarika, D., Poria, S. and Cambria, E., 2018. Recent Trends in Deep Learning Based Natural Language Processing, *ArXiv*, [PDF] Available at: <https://arxiv.org/pdf/1708.02709.pdf> [Accessed 04/10/2020].

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H and He, Q., 2020. A Comprehensive Survey on Transfer Learning, *ArXiv*, [PDF] Available at: <https://arxiv.org/pdf/1911.02685.pdf> [Accessed 03/10/2020].

Appendices

Appendix 1: Github Link for Project Code

The project code can be accessed at the public repository following the link below.

https://github.com/kaflesaurav/PMP_LSTM_NLP_Classifier

Appendix 2: Research Ethics Checklist

You must provide a response to ALL questions. Please refer to the Question Specific Advice for completing the Stage 1 Research Ethics Application Form for guidance.

	Will your research (delete as appropriate):			
1	Involve human participants?		YES	**NO**
2	Utilise data that is not publically available?		YES	**NO**
3	Create a risk that individuals and/or organisations could be identified in the		YES	**NO**
4	Involve participants whose responses could be influenced by your relationship with them or by any perceived, or real, conflicts of interest?		YES	**NO**
5	Involve the co-operation of a 'gatekeeper' to gain access to participants?		YES	**NO**
6	Offer financial or other forms of incentives to participants?		YES	**NO**
7	Involve the possibility that any incidental health issues relating to participants be identified?		YES	**NO**
8	Involve the discussion of topics that participants may find distressing?		YES	**NO**
9	Take place outside of the country where you work and/or are enrolled to study?		YES	**NO**
10	Cause a negative impact on the environment (over and above that of normal daily activity)?		YES	**NO**
11	Involve genetic modification of human tissue, or use of genetically modified organisms classified as Class One activities?[1].		YES	**NO**
12	Involve genetic modification of human tissue, or use of genetically modified organisms above Class One activities?[2].		YES	**NO**
13	Collect, use or store any human tissue or DNA (including but not limited to, serum, plasma, organs, saliva, urine, hairs and nails)?[3]		YES	**NO**
14	Involve medical research with humans, including clinical trials or medical devices?		YES	**NO**
15	Involve the administration of drugs, placebos or other substances (e.g. food, vitamins) to humans?		YES	**NO**
16	Cause (or have the potential to cause) pain, physical or psychological harm or negative consequences to humans?		YES	**NO**
17	Involve the collection of data without the consent of participants, or other forms of deception?		YES	**NO**

[1] Email **FST-Biologicalsafety.GMO@anglia.ac.uk** for further information.

[2] As above.

[3] For any research involving human material you must contact Matt Bristow (matt.bristow@anglia.ac.uk) for further guidance on how to proceed

18	Involve interventions with people aged 16 years of age and under?		YES	**NO**
19	Relate to military sites, personnel, equipment, or the defence industry?		YES	**NO**
20	Risk damage/disturbance to culturally, spiritually or historically significant artefacts/places, or human remains?		YES	**NO**
21	Contain research methodologies you, or members of your team, require training to carry out?		YES	**NO**
22	Involve access to, or use (including internet use) of, material covered by the Counter Terrorism and Security Act (2015), or the Terrorism Act (2006), or which could be classified as security sensitive?[4]		YES	**NO**
23	Involve you or participants in a) activities which may be illegal and/or b) the observation, handling or storage (including export) of information or material which may be regarded as illegal?		YES	**NO**
24	Does your research involve the NHS (require Health Research Authority and/or NHS REC and NHS R&D Office cost and capacity checks)?		YES	**NO**
25	Require ethical approval from any recognised external agencies (Social Care, Ministry of Justice, Ministry of Defence)?		YES	**NO**
26	Involve individuals aged 16 years of age and over who lack 'capacity to consent' and therefore fall under the Mental Capacity Act (2005)?		YES	**NO**
27	Pose any ethical issue not covered elsewhere in this checklist (excluding issues relating to animals and significant habitats which are dealt with in a separate form)?		YES	**NO**

Please note that the Faculty Research Ethics Panel (FREP) will refer to the Office of the Secretary and Clerk any application where, in the view of the Chair, the proposed research poses a risk of a legal or security related nature to Anglia Ruskin University. The Chair will seek guidance from the Secretary and Clerk before the FREP decides if the proposed research can be granted ethical approval and/or the nature of any special arrangements which need to be put in place.

[4] The Counter Terrorism and Security Act (2015) and Terrorism Act (2006) outlaws web posting of material that encourages or endorses terrorist acts, even terrorist acts that have occurred in the past. Sections of the Terrorism Act also create a risk of prosecution for those who transmit material of this nature, including transmitting the material electronically. The storage of such material on a computer can, if discovered, prompt a police investigation. Visits to websites related to terrorism and the downloading of material issued by terrorist groups (even from open-access sites) may be subject to monitoring by the police. Storage of this material for research purposes may also be subject to monitoring by the police. Therefore, research relating to terrorism, or any other research that could be classified as security-sensitive (for example, Ministry of Defence-commissioned work on military equipment, IT encryption design for public bodies or businesses) needs special treatment. If you have any doubts about whether your research could be classified as security-sensitive, please speak to your FREP Chair.

Appendix 3: Research Ethics Training Screenshot