To develop more scalable methods to detect toxic and misleading content on platforms like Quora.

By
Satyam Priyadarshi
Student ID: 2323146
Supervisor: Dr. Peter Hancox

UNIVERSITY OF BIRMINGHAM

A thesis submitted to the University of Birmingham
For the degree of MSc in Computer Science

School of Computer Science
University of Birmingham
Birmingham, UK

September 2022

Table of Contents

Abstract ..... *iii*

Acknowledgement ..... *v*

CHAPTER I: OVERVIEW *1*

1.1 INTRODUCTION **1**

1.2 Motivation **1**

1.3 Problem Formulation **2**

1.4 Performance metric **2**

1.5 Methodology **3**

CHAPTER II : Literature Review *4*

Introduction **4**

2.1 Lexical-Based Semantic analysis **4**

2.2 Machine Learning based Semantic Analysis **5**

2.2.1 Text Preprocessing 5

2.2.2 Supervised learning using Logistic Regression 6

2.3 Deep learning based semantic analysis **6**

2.3.1 Word Embeddings (GLoVe) 6

2.3.2 Bidirectional LSTM with Attention Mechanism 7

2.4 BERT (Bidirectional Encoder Representations from Transformers) **8**

CHAPTER III: Background *9*

3.1 Pre-Processing **9**

3.2 Vectorization **10**

3.2.1 Bag-of-Words 10

3.2.2 N-grams 10

3.2.3 Stemming and Lemmatization 11

3.2.4 Word2vec 11

3.2.5 TF-IDF(Term Frequency Inverse Document Frequency) 12

3.3 Machine Learning **12**

3.3.1 Machine Learning Steps 13

3.3.2 Logistic Regression 13

3.4 Word Embeddings **15**

3.4.1 GLoVe (Global Vectors for word representation) 16

3.5 Neural Networks **16**

3.5.1 LSTM (Long short-term memory) 17

3.5.2 Bidirectional LSTM with Attention Mechanism 18

3.6 BERT **19**

3.7 Development Environment **20**
3.8 Python Libraries imported **20**
CHAPTER IV: Methodology ***22***
4.1 Dataset Description **22**
4.1.2 Target Distributions 25
4.2 Exploratory Data Analysis **25**
4.2.1 N-gram analysis 25
4.2.2 Meta features 29
4.2.3 Word Clouds 30
4.2.4 TF-IDF Featurization 32
4.3 Logistic Regression **33**
4.3.1 Cost functioning Gradient Descent 33
4.4 Pre-Processing of data **34**
4.3 Use of Word Embeddings **34**
4.4 Bi-directional LSTM with Attention Mechanism **35**
4.5 Pre-trained BERT Implementation **36**
4.5.1 Model Description 36
4.5.2 Training of Pre-trained Bert 37
4.6 Summary **39**
Chapter V: Results and Discussions ***40***
CHAPTER VI: Conclusion ***41***
References and citations ***42***
Appendix ***45***

Figure 1. Bag of Words Example 10
Figure 2. N-gram example 11
Figure 3. Example of Stemming and Lemmatization 11
Figure 4. word2vec skip-gram mode 12
Figure 5. Logistic function 14
Figure 6. Logistic function 15
Figure 7. Top view illustration of Gradient Descent 15
Figure 8. Comparison of results for Word2Vec and Glove Embeddings 16
Figure 9. Representation of a Neural network for text classification problem 17
Figure 10. Architecture of LSTM 18
Figure 11. Bidirectional LSTM with Attention Layer for relation detection 19
Figure 12. BERT Architecture 20
Figure 13. Training Dataset description 24
Figure 14. Target variable distributions 25
Figure 15. Unigram Analysis 26
Figure 16. Bigram Analysis 26
Figure 17. Trigram Analysis 27
Figure 18. Quadgram Analysis 28
Figure 19. Number of characters in each class 29

Figure 20. Number of words in each class 29
Figure 21. Number of punctuations in each class 30
Figure 22. Word cloud for Training Dataset 30
Figure 23.Worcloud for all Sincere Questions 31
Figure 24. Word Cloud for all Insincere Questions 31
Figure 25. Generated tf-idf scores for all words 32
Figure 26. Results obtained from Logistic regression 33
Figure 27. Dataset after Pre-Processing 34
Figure 28. Bidirectional LSTM with Attention Mechanism Architecture 36
Figure 29. Training and validation loss for Experiment-1 38
Figure 30. Training loss vs Validation loss for experiment 2 38
Figure 31. Training loss vs Validation loss Experiment-3 38
Figure 32. Training loss vs Validation loss Experiment-4 39
Figure 33. Training loss vs Validation loss Final Experiment 39

Table 1. Comparison of results obtained by comparing Logistic Regression and Bi-directional LSTM with Attention Mechanism 40
Table 2. Comparison of F1 Scores for different dataset sizes and ratio of distribution for Our target values 40

Abstract

We live in an era when the majority of the world is connected by the Internet. On one hand, we earn money, respect, and knowledge from social media by creating, sharing, and searching for useful content; on the other hand, some people use social media to incite hatred and violence against individuals or groups based on race, religion, sexual orientation, and so on.

Machine Learning and Deep Learning approaches were investigated and a text classification predictive model was developed to detect inappropriate text. By filtering out inappropriate content, users with questions feel happier and more able to ask for and find the information they require, leading to increased activity; experienced people feel happier to share their knowledge because more people will appreciate and benefit from their contribution.

Acknowledgement

I'd like to acknowledge and thank my supervisor, Dr. Peter Hancox, for making this work possible. His guidance and advice got me through every stage of writing my project.

I'd also like to thank my parents and my entire family for their unwavering support and understanding while I was conducting research and writing my project. Your prayers for me have kept me going this far.

Finally, I'd like to thank God for allowing me to go through all of this. Day by day, I've felt your guidance. You are the one who allowed me to complete my degree. I will continue to put my faith in you for my future.

CHAPTER I: OVERVIEW

1.1 INTRODUCTION

Currently, if we have a question, we search online for an answer because it instantly solves any issue within seconds. For this reason, a number of Community Question Answering (CQA) systems have become more popular in recent years, including Quora, Yahoo! Answers, Stack Overflow, and many more websites. Many people use these forums to get answers to their questions because they allow users to post questions online, have experts from around the world respond, and then share their opinions or expertise with other users. This feature encourages more participation, which has contributed to the forums' popularity. While many individuals use these forums to ask sincere inquiries, some others abuse this medium to disseminate misinformation and xenophobia. To enable actual people to access these forums, those questions and individuals must be recognized.

How to deal with harmful and controversial information has become an existential issue for every large website nowadays. In order to maintain their platform as a place where users may feel secure sharing their knowledge with the world, Quora wishes to address this issue head-on. People may use Quora as a platform to learn from one another. People may interact with others who share original thoughts and well-written replies by posting questions on Quora. Eliminating questions with ulterior motives, such as those that are based on erroneous premises or that are intended to make a statement rather than elicit constructive replies, is a crucial task.
We're entering a competition sponsored by Quora on Kaggle where we'll be building algorithms to spot and reject fake queries (Quora Insincere Questions Classification | Kaggle, 2022). The business problem we are trying to solve is to predict whether a question posted on Quora is Sincere (0) or Insincere (1). To solve this issue, Quora has used both manual review and machine learning.
As the field of abusive language analysis and detection develops, it is important to critically evaluate the connections between the many subtasks that have been categorized under this heading. A typology was developed to capture key similarities and differences amongst subtasks and address the implications for data annotation and feature constructions based on studies on hate speech, cyberbullying, and online abuse (Waseem, et al., 2017). Since the goal of the project, I'm working on is to identify serious inquiries, a lot of the work that has previously been done in natural language processing serves as a source of inspiration.

1.2 Motivation

This research will use machine learning and Neural Network (NN) models to predict whether a question posted on Quora is Insincere or Sincere. The development of sophisticated financial polarity-lexicons that may be utilised to examine the relationship between financial feelings and future firm performance has received a lot of attention in recent studies. However, it is generally recognised that the overall semantic orientation of a phrase may differ from the previous polarity of specific words based on experience from other domains, where sentiment analysis is frequently used (Malo, et al., 2013).
To get to know more about the problem we need to know what makes a question Insincere or how can we differentiate between a Sincere or Insincere questions.
A query that has the intent to make a statement rather than elicit useful information is referred to as being disingenuous. Some traits that might indicate an insincere query include:

- Has a tone that isn't neutral
    1. Uses hyperbole to emphasise a point about a group of individuals
    2. Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
    1. Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype

2. Makes disparaging attacks/insults against a specific person or group of people
3. Based on an outlandish premise about a group of people
4. Disparages against a characteristic that is not fixable and not measurable

- Isn't grounded on reality
- Uses sexual content (incest, bestiality, pedophilia) for shock value and not to seek genuine answers

1.3 Problem Formulation

Well, this looks like quite a cumbersome task. In order to arrive to a solution for this problem we need to modify it to a business or real-world problem by defining the problem formulation using certain steps:

- Aim – To predict whether a question posted on Quora is Sincere or Insincere.
- Problem – To build a model that learns the function

$$f(q_n) = 1 \parallel 0$$

q_n = Question posted on Quora

1 = Insincere questions & 0 = Sincere questions

- Real world Objectives and Constraints

1. The cost of Misclassification could be very high.
2. We can say that our model is trying to predict the probability of question for being Sincere or Insincere. Where the distribution of datapoints is highly imbalanced.
3. No strict latency concerns but still our final model should not take lot of time to classify the posted question.
4. Interpretability of the question is very important. Since, Insincere questions are meant to make a statement which could further lead to hate comments on the platform. Therefore a good score is needed to predict the output.

- Type of Machine learning problem .

$$Y_i \in \{0,1\}$$

A binary classification problem for a supervised machine learning process.

1.4 Performance metric

The F1-score combines a classifier's accuracy and recall into a single statistic by calculating their harmonic mean. It mainly used to compare the effectiveness of two classifiers. Assume classifiers A and B have greater recall and precision, respectively. The F1-scores for both classifiers in this situation may be used to assess which one yields superior results (What is the F1-score?, 2022).

The F1-score for a classification model is calculated as

$$\frac{2(P * R)}{P + R}$$

P = precision ; R = recall

Precision - Precision is calculated by dividing the total number of returned results by the number of correct results for a text search on a set of documents. Precision considers all documents that are retrieved, but it may also be measured at a certain cut-off rank, taking only the top results that the system returns into consideration.

Recall - Recall is the ratio of the number of accurate results to the number of results that should have been returned for a text search on a batch of documents.

1.5 Methodology

There are 3 sections to this study.

1. The first looks at how Sincere and Insincere Questions are classified using machine learning. Numerous feature representations and weighting approaches are employed in the machine learning methodology. We also addressed why F1- Score is the appropriate metric for results computation and took a close look at how the dataset was constructed. I came to the conclusion that machine learning models might not provide the best results for the solution after realising the complexity of the issue, therefore I stuck with the Logistic Regression classification model for my baseline model. The Experimental analysis section gives a detailed explanation of this.
2. In the second experiment, we train our dataset using glove embedding to produce an embedding matrix. The Background and Literature review sections provide detailed explanations of how word embedding is used previously and the motivation behind their creation. The training of our Bidirectional LSTM with Attention Mechanism and the determination of the F1-Score come next.
3. To calculate my results for the last experiment, I used the most recent BERT models. The experimental component of this study provides a thorough explanation of the philosophy of the use of Bert and how this model affected my results.

CHAPTER II : Literature Review

Introduction

This section will explore and discuss machine learning and neural network algorithms from previous studies that relate to prediction and sentiment classification. This chapter introduces several sentiment classification techniques discussed in the literature, with a focus on the algorithms, nature of the data used and overall performance of the classifiers. This section gives a detailed research for the results obtained by other computer scientists focusing on the formulation of an approach in order to arrive to acceptable results .

2.1 Lexical-Based Semantic analysis

The lexicon-based method determines the tone of the entire text by looking at the contextual semantic orientation of each word in the text. The text's words and the list's words that have been kept and tagged with their semantic orientation are contrasted. The overall attitude of the text is determined based on the semantic orientation of each word, and it is categorised as either positive, negative, or neutral. The way that the lexicon-based sentiment analysis algorithm aggregates the sentiment values of positive and negative terms inside a message sets it apart from previous models. This method calls for the use of a "bag of words," a lexicon of positive and negative terms is required .

In (Aung, et al.,2017), student feedback is analysed using a lexicon-based methodology to determine the students' attitudes, which are then divided into positive and negative categories. The degree of teaching performance may be predicted by automatically analysing student text input using a lexicon-based technique. The opinion of instructors is therefore observed, defining the amount of favourable or negative opinions, by assessing the sentiment information comprising intensifier words extracted from students' comments. This system displays the instructors' opinions in terms of whether they are very positive, moderately positive, weakly positive, strongly negative, moderately negative, weakly negative, or neutral. Lexicon-based methods establish polarity by matching opinion terms from a sentiment lexicon with the relevant data. To measure how Positive, Negative, and Objective the words in the dictionary are, they assign sentiment ratings to the opinion terms.The orientation of words or phrases is calculated using the lexicon-based approach(Ngoc, et al.,2014). Further a sentiment regarding a student in a range of scores from -3 to +3 is recorded. Using a dictionary-based technique, the writers in (Hu et al.,2004) summarized customer input based on product characteristics. WordNet identified the moods of the retrieved adjectives as either positive or negative, with an accuracy rate of 84%, using information from Amazon and CNET.

Sentiment Analysis uses artificial intelligence to extract human emotion from text, providing extraordinary insight into the data. However, because the model ignores or misinterprets slang phrases or acronyms, the sentiment analysis of the text is significantly impacted by their use. The greatest solution to this issue is to substitute terms with their complete forms. When these short-forms are

swapped out for their long-form, the accuracy of the system increases. The most widely used slang terms are collected for this reason and then put to the word bag. By calculating the score of the contextual orientation of each word in the text, lexicon-based analysis may be used to determine the sentiment of the text.

2.2 Machine Learning based Semantic Analysis

Many academics and programmers from all around the world have employed different machine learning classification models to address a text categorization issue. This section will examine earlier efforts made in the subject and look in detail at text categorization methods.

2.2.1 Text Preprocessing

Preprocessing is one of the crucial components of a typical text classification system. The contribution of preprocessing operations to classification performance at various feature dimensions, their possible interconnections, and their reliance on the respective languages and domains are all thoroughly evaluated in (Uysal and Gunal, 2014). The objective of this study is to conduct a thorough investigation of the impacts of preprocessing on text classification, which examines a number of aspects including classification accuracy, text domain, text language, and dimension reduction. For this, two unique domains—email and news—as well as two different languages—Turkish and English—are utilised to compare all viable configurations of regularly used preprocessing tasks (Uysal and Gunal, 2014).

A conventional text classification framework consists of the preprocessing, feature extraction, feature selection, and classification stages. Common preprocessing procedures include tokenization, stopword removal, lowercase conversion, and stemming. The bag-of-words technique (Salton, Wong and Yang, 1975) is typically used in the feature extraction step of the vector space model (Uysal and Gunal, 2014).

Without a detailed examination of their effects on classification accuracy, text classification research frequently uses stemming, lowercase conversion, stopword elimination, and alphabetic tokenization. There hasn't been any in-depth research on how preprocessing chores impact text classification. For example, the effectiveness of stopword removal and stemming for English news datasets is explored in (Song, Liu and Yang, 2005). The findings indicate that stopword elimination and stemming have little impact. However, stopword removal and stemming are suggested in order to reduce the dimensionality of the feature space and improve the efficacy of the text classification system. The effects of lemmatization, stemming, and stop-word removal are investigated on English and Czech datasets (Kostoff and Geisler, 1999). Most of the time, stopword removal improved classification accuracy. However, word normalisation (also known as stemming or lemmatization) has a negative rather than a positive effect on text categorization. Using stop-word removal and omitting word normalisation may be the best strategy for text classification.

TFIDF works by proportionally raising a term's frequency in the text, but it is counterbalanced by the term's frequency in other publications. As a result, common words that appear often across all of the texts, such as "this," "are," etc., do not receive very high marks. However, if a phrase is used too frequently in a small percentage of the papers, it will receive a higher grade since it may indicate the context of the document. The link between confidence, precision, and recall was covered in (Zhang, Gong and Wang, 2005), which proposes a novel improved term frequency/inverse document frequency (TF-IDF) strategy that employs confidence, support, and distinctive words to increase the memory and precision of text categorization. TFIDF is widely used to denote the weight of the text feature. It does, however, have certain problems. TF-IDF is unable to account for word distribution across the text, as well as word importance levels and categorization disparities. A new feature weighting technique called TFIDFC I is added to the original TFIDF in order to reflect the differences across classes. When TFIDFC I and a certain classification method are combined, the macro F1 value is always higher than what was suggested by TFIDF in (HUANG, YIN and HOU, 2011).

2.2.2 Supervised learning using Logistic Regression

By defining two classes depending on the dataset given, Logistic Regression (LR) has been found to be successful at solving binary classification issues. According to study in (Occhipinti, Rogers and Angione, 2022), one of logistic regression's benefits is that it may be simpler to understand than other machine learning algorithms. Regularization techniques may be used as a feature selection tool when the dataset contains a lot of variables, which will make it easier to comprehend the findings.

The aim of LR is to find the best parameters $\boldsymbol{\beta}$o and $\boldsymbol{\beta}$1 to determine the best fitting model that describes the relationship between the input features and the predicted class value. The logistic regression model takes the natural logarithm of the odds as a regression function of the predictors. With 1 predictor, X, this takes the form ln[odds(Y=1)]=β0+β1X, where ln stands for the natural logarithm, Y is the outcome and Y=1 when the event happens (versus Y=0 when it does not), β0 is the intercept term, and β1 represents the regression coefficient, the change in the logarithm of the odds of the event with a 1-unit change in the predictor X. The difference in the logarithms of 2 values is equal to the logarithm of the ratio of the 2 values, so by taking the exponential of β1, we obtain the ratio of the odds (the odds ratio) corresponding to a 1-unit change in X (Logistic Regression, 2022).

2.3 Deep learning based semantic analysis

Sentiment analysis is a potent text analysis tool that uses machine learning and deep learning algorithms to automatically mine unstructured data (such as social media, emails, customer care issues, and more) for opinion and emotion. Machine learning has evolved, and deep learning (DL) is one such progression. Artificial neural networks, which link together algorithms to mimic the operation of the human brain, have made it possible to use machine learning in many real-world contexts, including self-driving vehicles and automated customer assistance.
Using text analysis techniques, sentiment analysis classifies emotions (positive, negative, and neutral) within data. Utilizing deep learning, sentiment analysis models may be trained to read for context, sarcasm, etc., grasp the writer's genuine mood and feelings, and comprehend material beyond simple definitions (Deep Learning, 2022).
Deep learning, which utilises many algorithms in a sequential chain of events to solve complicated problems, is hierarchical machine learning that enables you to process enormous volumes of data effectively and with a minimum of human input.
Machine learning and deep learning are sometimes used synonymously. Machine learning is deep learning, but deep learning is more sophisticated. Human intervention is necessary to fix a fundamental machine learning error, adjust the output, and "push" the model to learn. However, using the deep learning algorithm chain, the neural network may learn to rectify itself.

2.3.1 Word Embeddings (GLoVe)

In order to process natural language, word representation is essential. The standard method of expressing words as separate symbols lacks generalizability and is inadequate for many purposes. As a result, we look for a representation that encompasses word similarities in both semantics and syntax. Like this, words are stored as extremely high dimensional yet sparse vectors, with each element measuring how closely a word is related to a certain context (Baroni and Lenci, 2010), (Turney and Pantel, 2010). Word embeddings are real-valued word representations trained on natural language datasets that can capture lexical semantics. Models that suggest these representations have grown in favour recently, although the best technique of evaluation is still up for debate. Since language processing necessitates the recall of ideas from memory in response to a continuous flow of information. This retrieval is made easier if one can determine the main idea in a statement, exchange, or document and utilise that idea to forecast similar ideas and clear up confusion in terms. It is also regarded as one of the earliest publications that led to the usage of embeddings. In (Griffiths, Steyvers and Tenenbaum, 2007), a revolutionary approach to semantic representation, word meanings are represented in terms of a collection of probabilistic themes.

A revolutionary distinct task across content archives was shown in (Kusner et al.,2015), Word Mover's Discard (WMD). The minimal distance that the embedded words of one document must "travel" to reach the embedded words of the second document is how WMD calculates the difference between two text documents. The ideas discussed in (Kusner et al.,2015)are supported by current findings in word embeddings, which derive semantically significant representations for words from local cooccurrences in sentences. As seen by the findings in (Kusner et al.,2015), the distance metric may be viewed as a particular case of the Earth Mover's Distance, a well-researched transportation issue for which several extremely effective solutions have been created. Eight real-world archive classification information sets were used to demonstrate progress in (Kusner et al.,2015), which showed that the WMD measure produces very low KNN text classification error rates when compared to seven state-of-the-art baselines. Compared to the original skip-gram embeddings, the dependency-based embeddings in (Kusner et al.,2015) are less topical and show more functional similarity. Different embeddings were produced because of dependency-based context experiments. Additionally, it is shown in (Kusner et al.,2015) how the coming about embedding model may be questioned for the discriminative settings for a specific word. It is also observed that the learning approach seems to prefer conjunctions and prepositional objects as well as somewhat proximal syntactic contexts.(Schnabel et al., EMNLP 2015) provides a comprehensive review of assessment methods for unsupervised embedding algorithms that extract useful word representations from text. These assessment methodologies lessen bias, boost understanding, and allow us to swiftly and precisely crowdsource data-driven relevance judgements by directly comparing embeddings in connection to specific queries (Schnabel et al., EMNLP 2015). Comparing performance across tasks might help us understand the data that an embedding stores, but we shouldn't expect any one job to stand in for an abstract characteristic.

We look at and explain the model properties necessary for such regularities to occur in word vectors. The Glove model combines the advantages of the two most significant model families in the literature: local context window approaches and global matrix factorization. Glove is a novel global log-bilinear regression model. Our approach successfully uses statistical data by limiting training to the nonzero components of a word-word cooccurrence matrix rather than the complete sparse matrix or particular context windows in a large corpus. The model creates a vector space with considerable substructure, as shown by its performance of 75% on a recent word analogy task. It outperforms comparable models in tests requiring named entity recognition and similarity (Pennington et al., EMNLP 2014).

2.3.2 Bidirectional LSTM with Attention Mechanism

Recent developments in neural network algorithms have shown promise in the field of sentiment classification. Two popular neural network models, the recurrent neural network (RNN) and the convolutional neural network (CNN), have both been used in the past for sentiment categorization. In this section, we'll look at the Bi-directional LSTM with Attention mechanism.
A recurrent neural network (RNN) called long short-term memory (LSTM) was created to solve the issue of disappearing gradients and to handle longer-term dependencies than conventional RNNs. A group of LSTM units with input, output, and forget gates are used to describe long-term dependencies with arbitrary gaps. Two independent recurrent layers are combined in a bidirectional-LSTM, one of which gets the input sequence and the other of which receives a reversed copy. Bidirectional-LSTMs perform better than standard LSTMs in classifying sequences because they analyse both the forward and backward directions simultaneously (Liu and Guo, 2019).
The issue of textual content fake news identification utilising interpretable characteristics and algorithms is examined in (Hosseini et al.,2022). In specifically, a deep probabilistic model is created that combines a dense representation of textual news with semantic topic-related characteristics deduced from a Bayesian admixture model, bi-directional Long Short-Term Memory (LSTM) networks, and variational autoencoders. leading to a model ablation study that was undertaken, which

supports the accuracy and efficacy of integrating topic characteristics and neural embeddings both qualitatively and quantitatively through separability in lower dimensional embeddings. (Yang et al.,2017) uses a bidirectional attention-based LSTM technique to enhance target-dependent sentiment categorization. The alignment between the target entities and the most distinctive attributes is learned using our technique. We conduct extensive experiments on a real-life dataset. The experimental findings demonstrate that our methodology produces cutting-edge outcomes. Several methods are being proposed for target-dependent sentiment classification, according to (Yang et al.,2017), in order to capture the sentiments on target entities. Numerous recent research employ deep neural networks to address this issue since feature engineering requires a lot of manual labour. Their model is among the first to investigate attention-based architecture for target-dependent sentiment classification, and it draws inspiration from the recent success of attention-based neural networks.

The difficulty with the sequence model for the sentiment analysis task—which is explained in [27]—is that it will decode the input file sequences into a specified length vector. The input text information will be lost if the vectors are set to a length that is too short, and the text will ultimately be misinterpreted since sentiment analysis jobs do not fully utilise this special sentiment information. Bidirectional LSTM with an attention mechanism was presented as a solution to this issue, and it was shown that this improved the performance of bidirectional LSTM models for sentiment analysis. The F1 score obtained is an improvement above my baseline Machine Learning model, according to my experimental analysis, and the model employed in (Li et al., 2020) featured a multi-channel feature (SAMF BiLSTM) that functions similarly to Bidirectional LSTM with Attention mechanism.

2.4 BERT (Bidirectional Encoder Representations from Transformers)

Bidirectional Encoder Representations from Transformers, or BERT. BERT is intended to simultaneously condition on both left and right context in all layers in order to pre-train deep bidirectional representations from unlabelled text. (Devlin et al.,2019) claims that without making significant task-specific architectural adjustments, the pre-trained BERT can be improved with just one more output layer to provide state-of-the-art models for a variety of tasks including question answering and language inference. In order to address the post-pandemic mental health problem, BERT models are also employed to identify depression in text, according to (Novika et al.,2022). A DECK (Depression Checklist), a behavioural test paradigm for depression that allows for greater interpretability and increases the generalizability of BERT classifiers in the depression domain, is described in (Novika et al.,2022). Two Twitter-based datasets and one clinical interview-based dataset were used to analyse the results.

1. BERT are resistant to some gender-sensitive text variants.
2. BERT rely on the crucial first-person pronoun use depressed linguistic signal.
3. Some additional depression systems, such as suicidal ideation, are missed by BERT.

Understanding the relationship between the text and the usage of depressed words in it depends on the results from (Novika et al.,2022). In order to make the impact of these words on the suggested reading of this text clear to grasp. Additionally, it was noted in (Yenicelik et al.,2020) that each stage of the standard NLP pipeline is represented by the model in an understandable and localizable manner, and the areas in charge of each step follow the usual order: POS tagging, parsing, NER, semantic roles, then coreference. Qualitative study suggests that the model may and frequently does modify this pipeline dynamically, altering lower-level judgments based on knowledge obtained from higher-level representations that has been used to resolve ambiguities. It can be claimed from this specific research that BERT captures the phases of the typical NLP pipeline in an interpretable and localizable fashion. Whereas studying the Linguistic patterns it appears that simple syntactic information emerges sooner in the network, while high-level semantic information occurs at higher levels.

BERT embedding vectors in (Yenicelik et al.,2020) demonstrate that semantics create seamless structures that are closely correlated with sentiment and syntax rather than emerging as discrete

clusters. We address the question of how contextual word embeddings generated by BERT capture semantic notions with a heavy focus on polysemy.
According to results, BERT produces closed semantic zones that blend into one another without being easily distinguished from one another to get goals. The topic of whether BERT contributes a more flexible idea of semantics in comparison to the hard-coded examples created by linguists is raised by the recurrent constraints of rigid distinctions between senses as provided by WordNet.
BERT, a contextualised embedding, performs better than static embeddings like skip-gram, CBOW, and GloVe as a strong input representation for NLP applications. However, the dynamic nature of these embeddings, which are determined by a sentence-level context, prevents their application in lexical semantics tasks. We solve this issue by making use of dynamic embeddings as word representations in training static embeddings, therefore exploiting their great representation capacity for disambiguating context information. On a variety of lexical semantics tasks, the findings demonstrate that this strategy leads to improvements over conventional static embeddings, attaining the best reported results on training dataset.

CHAPTER III: Background

Natural language processing (NLP) is a broad area of study where linguistics, computer science, and artificial intelligence all come together. It includes a broad spectrum of exciting issues with excellent practical applications, much like named entity recognition, machine translation, or machine question answering. The way that each of these topics manages textual data is different. Although unstructured, text may be a rich source of information, but it can be difficult to make inferences from it. Text classification is one of the most important tasks in supervised machine learning (ML). It is a method of classifying documents into tags or categories that enables us to swiftly and efficiently organise and analyse material. One of the fundamental NLP tasks, sentiment analysis has several uses, such as spam identification, topic labelling, intent detection, etc. The categories can include a wide range of subjects and are determined by the dataset used.
Sentiment analysis aims to isolate the polarity of a text's sentiments from its substance. The sentiment polarity is a number that reflects whether a textual opinion is favourable polarity i.e positive(polarity=1), negative (polarity=0), or neutral. However, the work is special since it involves dealing with textual data, a non-numerical sort of data that required to be changed for a function to use it as an input and make inferences from it.

To make this possible we use these two basic steps:

- Pre-processing – to make text cleaner by removing extra information from it.
- Vectorization – conversion of textual data into numerical data.

3.1 Pre-Processing

For every machine learning or deep learning assignment, cleaning or pre-processing the textual input is equally as important as creating the model itself. Because it deals with human language, text data is one of the most unstructured forms of data currently available. For every machine learning or deep learning assignment, cleaning or pre-processing the textual input is equally as important as creating the model itself. Given that textual data is one of the most unstructured forms of data currently available for interacting with Human language.
Understanding the issue and the dataset we are working with, which have been described in detail in the methodology section, is the first step in solving any machine-learning challenge. Since the data was gathered from online question-and-answer services like Quora, whose database is made up entirely of text, we need to apply a variety of text pre-processing techniques to deal with a dataset like this in order to overcome this issue. The most popular ways include deleting punctuation, fixing misspelt words, cleaning word contractions, lowercasing the whole text corpus, and substituting common acronyms for URLs and arithmetic formulae.
In the Methodology section, each of the strategies was thoroughly detailed.

3.2 Vectorization

The transformation of text into numerical vectors is one of the initial processes after data pre-processing. Some of the methods utilised in this research will be described in the subsections that follow.

3.2.1 Bag-of-Words

Before modelling text data using any machine learning model to obtain results, the bag of words is a method of encoding the data. This method for removing characteristics from data is relatively straightforward and flexible. A textual depiction that shows how words appear in a document is called a "bag of words."
Two elements make up this:
1. An established vocabulary.
2. Building a d-dimensional vector, where each dimension stands in for a word, and counting how many times it appears.
We may infer from bag-of-words analysis that comparable must produce closer vectors. Bag-of-words also have a drawback in that they perform poorly when the terminology we used to create the text corpus differs somewhat, since this might lead to data repetition or duplication.

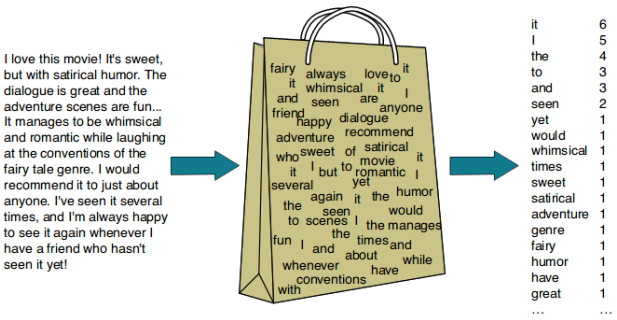

Figure 1. Bag of Words Example

3.2.2 N-grams

In the domains of probability and text analytics, an n-gram is a continuous sequence of n items from a given sample of text or speech. The objects might be syllables, phonemes, characters, or words, depending on the application. The text or speech corpus is where the n grammes are gathered. In N-gram models, upcoming items are predicted using an item sequence. Think about a string of letters that were utilised in a document. Using training data, an N-Gram model will analyse the letter sequence and provide a probability distribution for the likelihood of upcoming values. Each possible value will be given a probability, and the sum of all probabilities will equal 1.

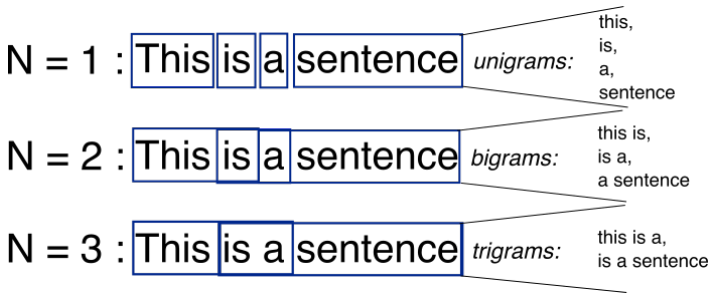

Figure 2. N-gram example

3.2.3 Stemming and Lemmatization

Stemming and lemmatization are techniques used by chatbots and search engines to examine the meaning of a word. While stemming uses the word's stem, lemmatization uses the word's context.

Stemming algorithms operate by deleting the word's start or finish using a list of frequently occurring prefixes and suffixes that may be present in inflected words. We claim that this approach has certain disadvantages since random cutting may occasionally work but not usually. Lemmatization, on the other hand, considers the morphological analysis of the words. It is crucial to have complete dictionaries so that the algorithm may utilise them to link the form to its lemma in order to do this. Using the same example sentences once more, you can see how it works. Another crucial point to stress is that a lemma, as opposed to a stem, is the root form of all of its inflectional forms. So, instead of stems, regular dictionaries are collections of lemmas.

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study

Form	Morphological information	Lemma
studies	Third person, singular number, present tense of the verb study	study
studying	Gerund of the verb study	study

Figure 3. Example of Stemming and Lemmatization

3.2.4 Word2vec

The word2vec approach uses a neural network model to learn word associations with the aid of a sizable text corpus. Once trained, a model like this may suggest new words to finish a statement or find keywords that are comparable. As the name indicates, word2vec represents each distinct word as a particular collection of numbers called a vector. The vectors are carefully chosen such that the degree of semantic similarity between the words represented by each vector can be calculated using a simple mathematical formula (the cosine similarity between the vectors).
Word embeddings are produced using Word2vec, a group of connected models. These models are two-layer shallow neural networks that have been taught to reconstruct language contexts for words. Word2vec builds a vector space, often with several hundred dimensions, using a large corpus of text as input, assigning each unique word in the corpus a corresponding vector in the space. If two-word vectors in a corpus have the same context, they are placed next to one another in the vector space. Using either the continuous bag-of-words (CBOW) or continuous skip-gram model architectures, Word2vec may provide a distributed representation of words. Using a window of nearby context words, the model predicts the current word in the continuous bag-of-words architecture. The prediction is independent of how the surrounding words are arranged (bag-of-words assumption). The model predicts the context words that will be right after the current word in the continuous skip-gram architecture. The skip-gram design gives near context phrases the advantage over distant ones.

(Mikolov et al.,2013) (Mikolov et al.,2013) The authors point out that while CBOW is speedier, skip-gram performs better for unusual words.

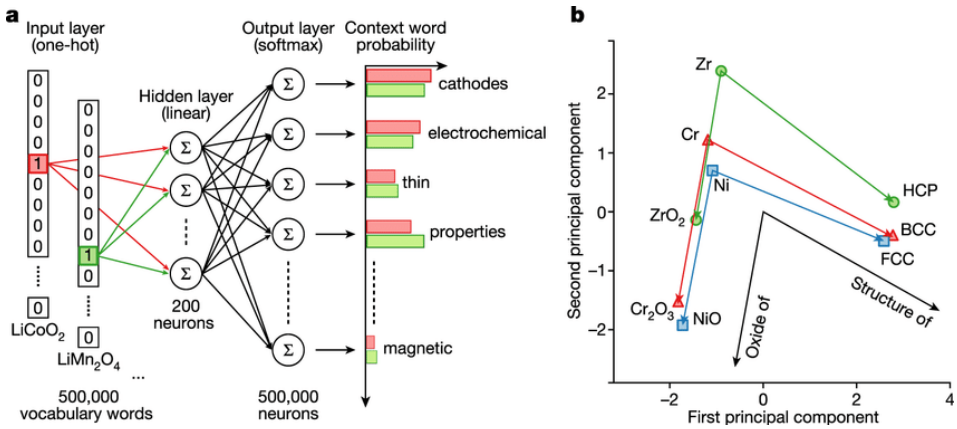

Figure 4. word2vec skip-gram mode

3.2.5 TF-IDF(Term Frequency Inverse Document Frequency)

Inverse document frequency and term frequency are used by TF-IDF to determine which terms are significant as features. Like bag-of-words, TF-IDF identifies frequent terms by using data on phrase frequency in each document. However, overused words that appear in every document, such "the," are meaningless. Inverse document frequency helps to weight and dismiss words that appear in all texts, enabling the identification of significant elements in important phrases.

- Term frequency

Counting the instances of each phrase in each document; the frequency at which a term appears in a document is referred to as term frequency.

$$\mathrm{tf}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

- Inverse document frequency

The inverse document frequency indicates the quantity of information a word communicates or if it is common or unusual across all texts. This number is derived by dividing the total number of documents by the number of documents containing the phrase, then calculating the logarithm of that quotient. The value is then logarithmically scaled.

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Tfidf(t,d,D) = tf(t,d)*idf(t,D)

3.3 Machine Learning

Machine learning is a class of statistical tools and techniques that may be used to train models to anticipate future events using experiences and past occurrences. By identifying and categorising patterns that are similar, machine learning can enhance decision-making. Machine learning approaches are categorised using three types of learning:

- Supervised Machine Learning

In supervised learning, machine learning models are trained using a set of labelled training data. Each piece of information in the database receives a final label and a set of attributes (or class). While the labels in regression problems are continuous, like "height" or "weight," those in classification problems are categorical and discrete, like "healthy" or "diseased." Both classification and regression problems may be handled using supervised learning. Examples of supervised machine learning models include Naive Bayes, Decision Trees, Random Forests, and Support Vector Machines.

- Unsupervised Machine Learning

The training data for this model only includes input attributes for each record since labels are either inaccessible or would be too expensive to obtain. Unsupervised models include things like k-nearest neighbour and K-means clustering.

- Reinforcement Machine Learning

Reinforcement learning (RL), a subfield of machine learning, examines how intelligent agents should act in a certain environment to maximise the idea of cumulative reward. One of the three primary machine learning paradigms is reinforcement learning, along with supervised learning and unsupervised learning.

By not requiring the display of labelled input/output pairs or the explicit correction of undesirable behaviours, reinforcement learning differs from supervised learning. Instead, achieving a balance between exploitation and exploration (of uncharted territory) is the focus (of current knowledge). (Kaelbling, Littman and Moore, 1996)With partially supervised RL algorithms, the advantages of supervised and RL algorithms may be merged.

3.3.1 Machine Learning Steps

There are various crucial procedures which are needed to be followed while constructing a machine learning model (Pennington, 2022).

- Identify the problem: Prior to developing any machine learning models, it is critical to comprehend the problem. It is essential to identify current issues and choose the questions the model needs to answer.
- Data collection and analysis: Data can come from a wide range of sources and take on several different formats (e.g., text, digits, images and sound). The acquired information must be translated into a format that the model can understand (i.e., vector of numbers). It is essential to comprehend the data and seek for recognisable trends.
- Pre-processing the data may have an impact on the performance of the model. It is necessary to scan the data and remove any noisy data points.
- Model development and training: Decide which learning models will be applied. The model will be trained using some of the acquired data, also known as training data.
- Model testing and assessment: Test data must be utilised to evaluate the model after training. This data is used to evaluate model performance and identify the amount of data that was successfully and wrongly classified.
- Model deployment: The model may be used to classify forthcoming, unknown circumstances after being examined and its performance monitored.

Supervised learning may be used to handle both regression and classification issues. The focus of this section will be classification models due to the nature of the study. Several machine learning techniques are included in the classification category.

3.3.2 Logistic Regression

A classification approach is utilised when an answer variable is categorical. The objective of logistic regression is to establish a relationship between features and likelihood of a certain event. The logit functions, which are employed in logistic regression, aid in creating a connection between the dependent variable and independent factors by anticipating the probabilities or possibilities of occurrence. Probabilities are converted into binary values via the logistic functions, also known as sigmoid functions, which may subsequently be used to create predictions.

Types of Logistic Regression

- Binary Logistic regression

There are just two possible outcomes or classes for the dependent variable. Example – Pass or Fail, Head or Tails etc.

- Multinomial Logistic Regression

Only two, three, or more alternative outcomes/classes without ordering are conceivable for the dependent variable. Example – Predicting food quality (Good, Bad or Great).

- Ordinal Logistic Regression

Only two, three, or more different outcomes/classes with ordering are conceivable for the dependent variable. Example- a star rating from 1 to 5.

Even though logistic regression is a sort of linear model, it differs from linear regression models in that it does not make the same assumptions about the correlations between dependent and independent variables, homoscedasticity, and error terms, among others.

However Logistic regression has few assumptions of its own:

- It is presumptive that the independent variables' multicollinearity is either negligible or non-existent.
- It is presumptively true that independent variables are linearly connected to odds logarithm.
- For accurate prediction, a big sample is presumed.
- The observations are thought to be independent of one another.
- No significant values (outliers) exist in the continuous predictors (independent variables).
- When using ordered logistic regression, the dependent variable must be ordered when using logistic regression with two classes when the dependent variable is binary.

3.3.2.1 Logistic function

The output of the linear function is condensed between 0 and 1 by the logistic regression instead of fitting the best fit line.

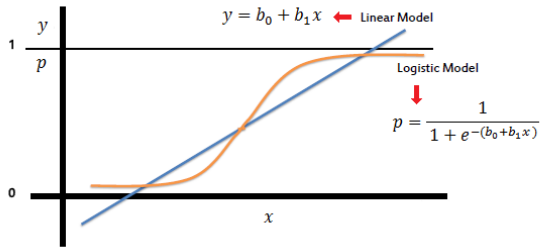

Figure 5. Logistic function

In the logistic regression formula

When b0+b1X == 0, the p will be 0.5.

When b0+b1X > 0, the p will be moving in the direction of 1.

When b0+b1X < 0, the p value will be decreasing toward 0.

Cost Function of logistic Regression

A machine learning model's efficacy is assessed using a cost function given a collection of data. In, essence, Cost Function computes and shows the difference between expected and projected values as a single real number. However, to put it simply, Cost Function is the average error of the n-sample data, while Loss Function is the error for each individual data point. Cost Function and Loss Function are commonly misinterpreted. In contrast to the Loss Function, which only reflects one training example, the Cost Function represents the whole training set.

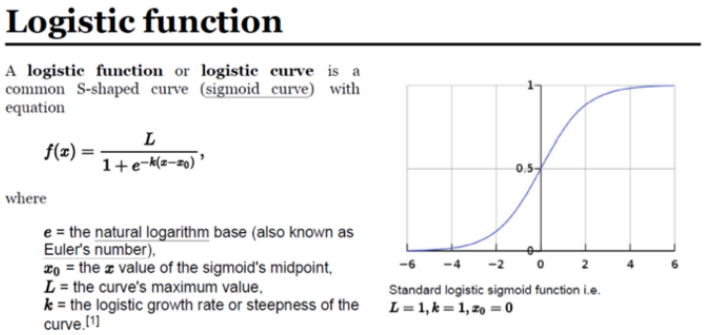

Figure 6. Logistic function

3.3.2.2 Gradient Descent

The coefficients of a function that minimises a cost function are determined using an optimization method known as gradient descent (cost). Gradient descent is the preferred method when parameters need to be identified using an optimization methodology but cannot be computed analytically (for instance, using linear algebra). You may utilise a variant of gradient descent known as stochastic gradient descent when you have a lot of data.

In the initial stage of the procedure, the training dataset must be organised at random. To alter the order in which the coefficient changes are implemented, this is done. Since the coefficients are changed after every training instance, the updates and the associated cost function will both be unpredictable and noisy. By altering the order in which the coefficient updates are performed, it makes advantage of this random walk and keeps it from being lost or stuck.

Except for the cost being calculated for a single training pattern rather than being put together for all training patterns, the update method for the coefficients is the same as the one described above. Stochastic gradient descent may speed up learning for very large training datasets, and usually, just a few iterations over the dataset—between one and ten—are enough to produce an excellent or good-enough set of coefficients.

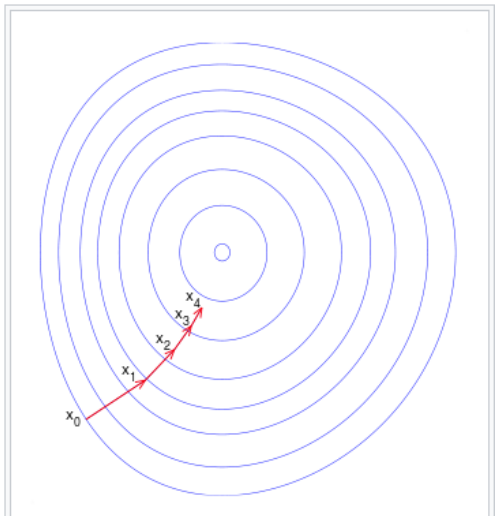

Figure 7. Top view illustration of Gradient Descent

3.4 Word Embeddings

In a word embedding, which is a learned representation for text, words with the same meaning are represented identically. This approach of encoding words and documents may be responsible for one of the significant developments in deep learning for challenging natural language processing problems. Individual words are represented as real-valued vectors in a predetermined vector space in a

process known as a word embedding. Since each word is allocated to a distinct vector and the vector values are learned similarly to a neural network, the technique is frequently referred to as deep learning.
The key to the strategy is the idea of using a dense distributed representation for each word. Each phrase is represented by a real-valued vector, which usually has tens or hundreds of dimensions. The number of dimensions required for sparse word representations, such one-hot encoding, can reach hundreds of millions.
The distributed representation is learned based on word use. This enables words that are often used to naturally have representations that accurately convey their meaning. This may be described as crisp yet brittle when compared to the bag of words paradigm, where distinct words have different representations regardless of how they are used until specifically addressed. Word-embedding techniques develop a real-valued vector representation for a specified fixed sized vocabulary from a corpus of literature. On other tasks, such as document classification, the learning procedure is either coupled with the neural network model or it is an unsupervised procedure utilising document statistics.

3.4.1 GLoVe (Global Vectors for word representation)

In tasks including named entity recognition, word analogies, and word similarity, GloVe outperforms previous models. GloVe is a new global log-bilinear regression model for the unsupervised learning of word representations. (Pennington et al.,2014) at Stanford developed the word2vec method for successfully learning word vectors. The Global Vectors for Word Representation, or GloVe algorithm, is its extension.
Traditional vector space model representations of words were produced using Latent Semantic Analysis (LSA), a matrix factorization technique. While word2vec is better at identifying meaning and displaying it on tasks, LSA is better at utilising global text statistics. GloVe is a methodology that combines the global statistics of LSA-style matrix factorization with the local context-based learning of word2vec. Instead of utilising a window to identify local context, GloVe constructs an explicit word-context or word co-occurrence matrix using statistics across the whole text corpus. The result is a learning model that could ultimately result in word embeddings that are more effective (Pennington, 2022).

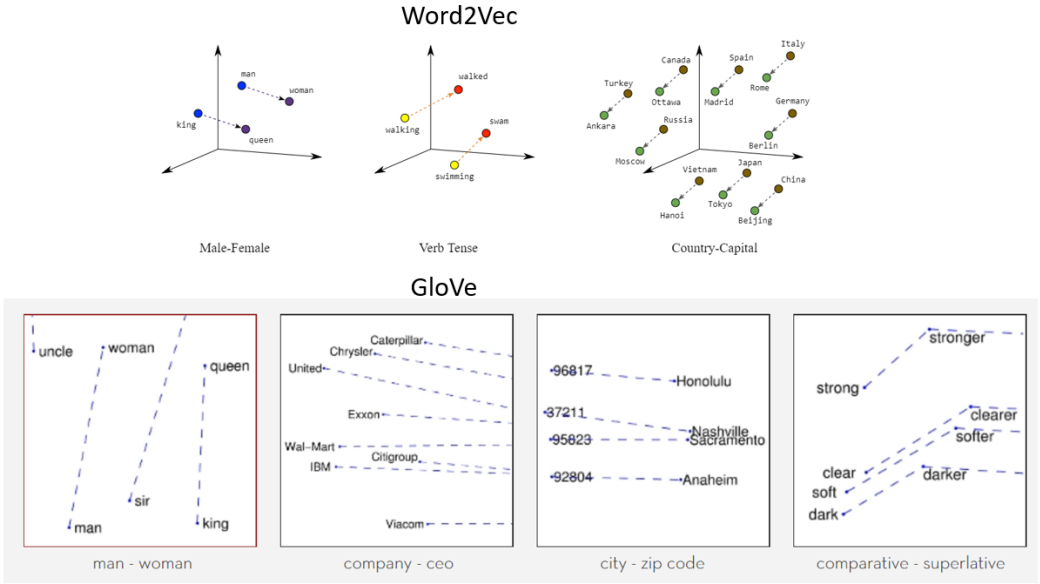

Figure 8. Comparison of results for Word2Vec and Glove Embeddings

3.5 Neural Networks

In more modern use, a neural network is a network or circuit comprised of artificial nodes or neurons rather than actual neurons. (Mikolov,chen et al.,2013)Therefore, a neural network may either be a biological neural network made up of biological neurons or it can be an artificial neural network used to address problems in artificial intelligence (AI). The connections between real neurons are modelled in artificial neural networks as weights between nodes. Positive weights imply excitatory connections, whereas negative values represent inhibitory connections. Each input is given a weight before being added together. This operation is known as a linear combination. The last factor controlling the output's amplitude is an activation function. For example, a common output range is between 0 and 1. This action is referred to as a linear combination. Finally, the output's amplitude is controlled by an activation function. A normal range for acceptable output, for instance, is between 0 and 1, or it may be between 1 and 1.

When a dataset can be used to train them, these artificial networks may be used for adaptive control, predictive modelling, and other activities. Self-learning arising from experience may occur inside networks, which are able to draw conclusions from a complex and seemingly disconnected set of information. (Definition of Neural Network - Gartner Information Technology Glossary, 2022)

The architecture of a text categorization neural network is very dissimilar from the networks previously demonstrated. It still has a dense layer (or layers), a SoftMax output layer for multiclass classification with one neuron per class, or a sigmoid output layer for binary classification with one neuron. But before those layers, there is an embedding layer and a flatten layer. The former transforms word vector arrays—which include data on the relationships between words—into arrays of scalar values, which represent words through word embeddings. The latter "flattens" the 2D arrays generated by the embedding layer into 1D arrays, allowing input to a dense layer.

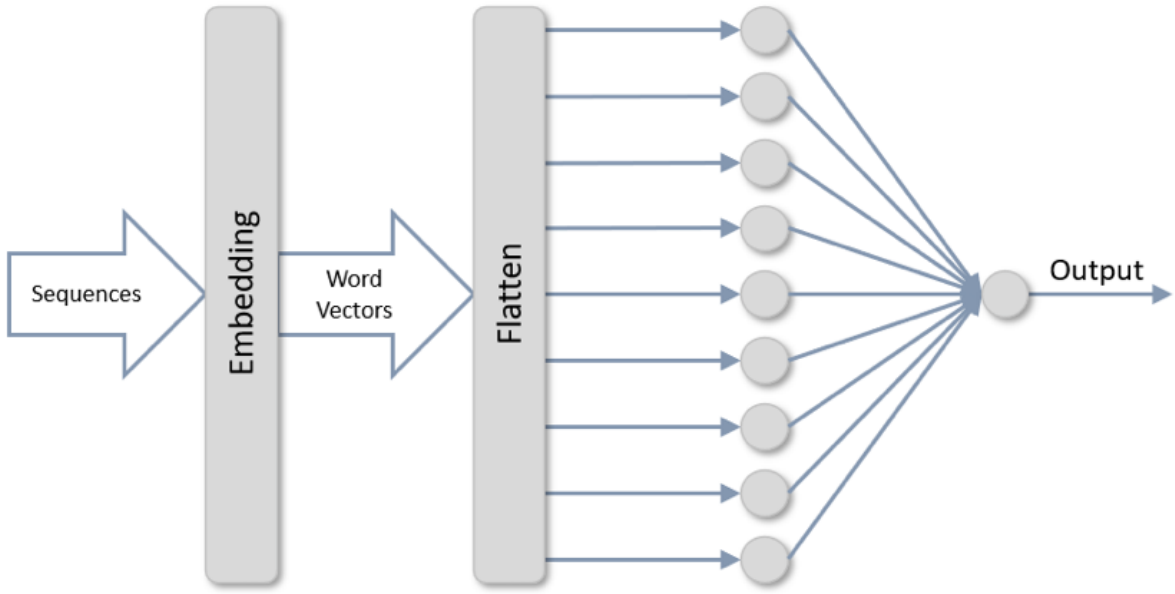

Figure 9. Representation of a Neural network for text classification problem

3.5.1 LSTM (Long short-term memory)

LSTMs, or long short term memory networks, are a special form of RNN that can recognise long-term dependencies. Following its initial presentation by Hochreiter & Schmidhuber (1997), they were further refined and made famous by several authors. They are now widely used and work remarkably effectively when used to solve a variety of problems.

Intentionally, LSTMs are created to prevent the long-term reliance issue. To demonstrate how LSTMs differ from traditional neural networks made up of simple neurons, a few examples may be utilised.

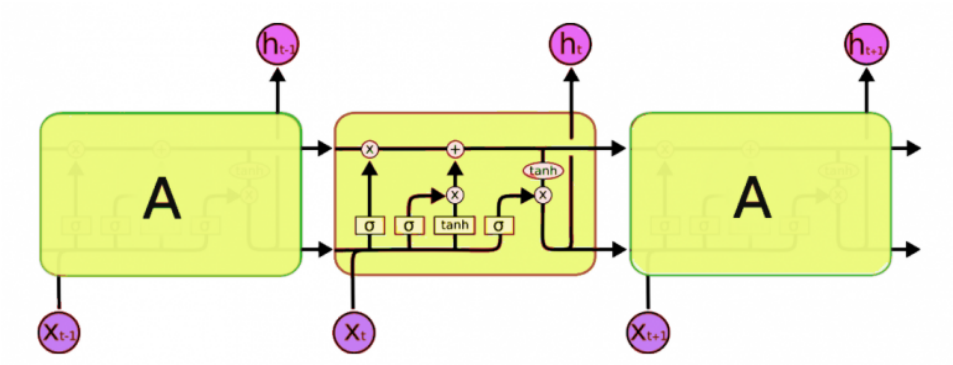

Figure 10. Architecture of LSTM

3.5.2 Bidirectional LSTM with Attention Mechanism

Each training sequence is fed into two distinct recurrent nets that are both connected to the same output layer in a bidirectional recurrent neural network. This suggests that every point in a given sequence, including those that occur before and after it, is fully sequentially known to the BRNN. The net is free to use as much or as little of this context as is necessary, therefore there is no need to provide a (task-dependent) time-window or target delay size.
In neural networks, attention is a technique that mimics cognitive attention. By amplifying some and decreasing others, the impact is meant to persuade the network to pay closer attention to the little but important parts of the input data. An algorithm that identifies which part of the data is more important than another depending on the context is trained using gradient descent. Attention-like processes were initially discussed in the 1990s under the heading's multiplicative modules, sigma pi units, and hypernetworks (Definition of Neural Network - Gartner Information Technology Glossary, 2022). Its flexibility comes from its capacity to change as the programme progresses, as contrast to standard weights, which must remain constant. Attention is employed in perceivers for multisensory data processing, transformers for language processing, neural Turing machines for memory, and differentiable neural computers for reasoning (sound, pictures, video, and text). (Definition of Neural Network - Gartner Information Technology Glossary, 2022) (Vaswani et al., 2022) (Ramachandran et al., 2022)
The addition of the attention mechanism improved the performance of the encoder-decoder paradigm for machine translation. The attention mechanism was created to allow the decoder to use the most important parts of the input sequence in a flexible manner by combining all the encoded input vectors in a weighted manner, with the most pertinent vectors obtaining the biggest weights.

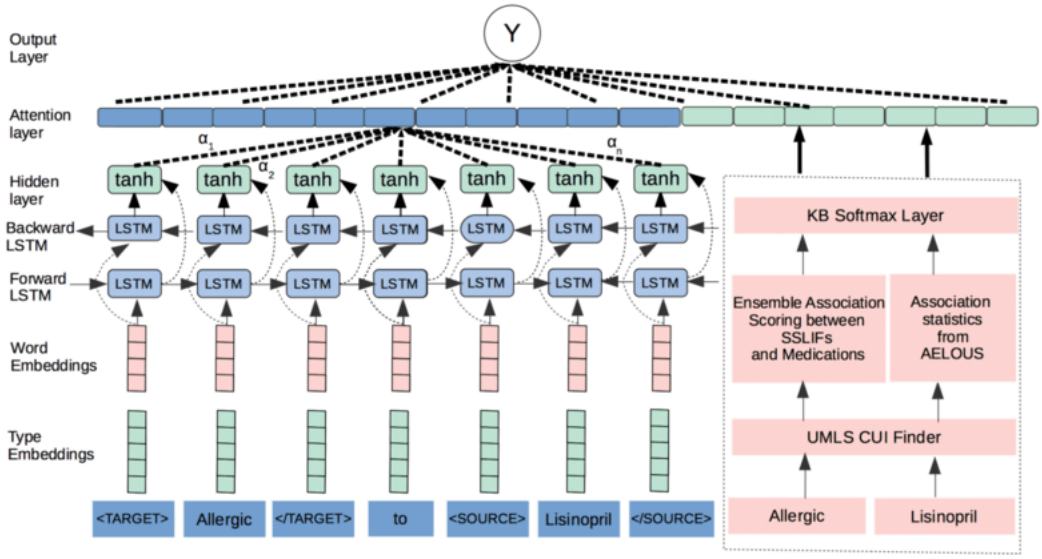

Figure 11. Bidirectional LSTM with Attention Layer for relation detection

3.6 BERT

The absence of sufficient training data is one of the main problems facing NLP. Although there is a vast quantity of text data accessible overall, we must divide it up into the many different areas in order to build task-specific datasets [28]. We only have a few thousand to a few hundred thousand human-labeled training samples after doing this. Unfortunately, deep learning-based NLP models need a lot more data to function properly; they significantly improve when trained on millions or billions of annotated training instances. Researchers have created a variety of methods for training general purpose language representation models utilising the massive amounts of unannotated content on the web to fill up this data gap (this is known as pre-training). When working with issues like question answering and sentiment analysis, for example, these general purpose pre-trained models may subsequently be fine-tuned on smaller task-specific datasets. When compared to starting from zero and training on the smaller task-specific datasets, this method significantly increases accuracy. In the deep learning field, BERT, a relatively new addition to these approaches for NLP pre-training, generated a stir since it demonstrated cutting-edge outcomes in a range of NLP tasks, including question answering (Tenney et al.,2019).

The best thing about BERT is that it is available for free download and use. We can either use the BERT models to extract high-quality language features from our text data or we can fine-tune these models on a specific task, like sentiment analysis and question answering, with our own data to produce cutting-edge predictions.

A language model would have studied this text sequence during training from either left to right or from a combination of left to right and right to left in the pre-BERT era. For creating sentences, this one-directional method works well. We may anticipate the subsequent word, add it to the sequence, and then predict the subsequent word until we have a complete phrase (Yenicelik et al.,2020).

BERT, a bidirectional trained language model, now joins the conversation (this is also its key technical innovation). As opposed to the single-direction language models, we can now see the context and flow of language more deeply.

BERT employs a cutting-edge method called Masked LM (MLM), which randomly masks words in the phrase before attempting to predict them. This method is used in place of traditional word prediction. When a word is "masked," the model considers both left and right surrounds, as well as both directions of the phrase, to anticipate the hidden word. It considers the preceding and following tokens simultaneously, in contrast to earlier language models. This "same-time portion" was absent from the existing combined left-to-right and right-to-left LSTM based models. (BERT may be more accurately described as non-directional.)

BERT is dependent on a Transformer (the attention mechanism that learns contextual relationships between words in a text). An encoder to read the text input and a decoder to provide a prediction for

the job make up a simple Transformer. Since the objective of BERT is to produce a language representation model, just the encoder portion is required. A series of tokens that are first transformed into vectors and then processed by the neural network make up the input to the BERT encoder. However, BERT requires the input to be modified and embellished with additional metadata before processing can begin(Understanding the BERT Model, 2022):

1. Token embeddings: In the first sentence, a [CLS] token is added to the input word tokens, and a [SEP] token is added at the end of each sentence.
2. Segment embeddings: Each token is given a marker designating Sentence A or Sentence B. This means that the encoder can tell which phrases are which.
3. Positional embeddings: Each token is given a positional embedding to show where it belongs in the phrase.

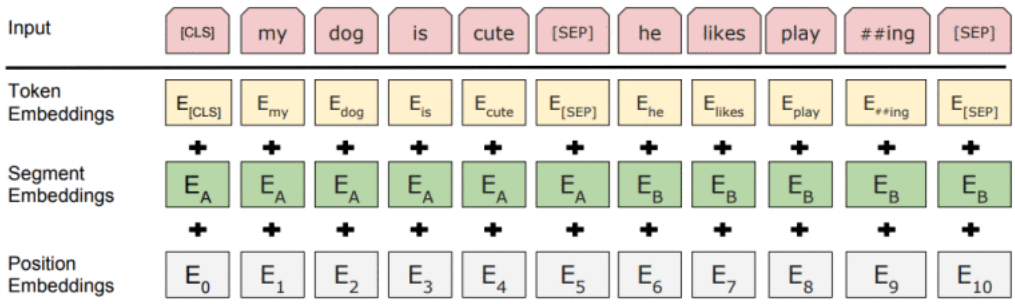

Figure 12. BERT Architecture

In a wide range of general language understanding tasks, including as sentiment analysis, question answering, paraphrase detection, and linguistic acceptability, BERT exceeded the state-of-the-art.

3.7 Development Environment

A development environment or programming environment is a combo of a text editor and code implementation. Here we used kaggle notebooks built uopn google colab, The majority of this project is based on Deep learning technology. This project will utilise kaggle notebooks, a free Jupyter notebook interactive development environment (REPL) hosted in Google's cloud.

Kaggle Notebooks : Based on Google Colab environment it's a document that enables you to write, execute, and share Python code in a web browser. It is a variant of the popular Jupyter Notebook that is part of the Google toolkit. Jupyter Notebooks (and consequently Kaggle notebooks) allow you to produce a document including executable code in addition to text, photos, HTML, LaTeX, etc., which is then kept in your kaggle profile under your work section and can be shared with peers and co-workers for editing, commenting, and viewing.

3.8 Python Libraries imported

Python, like PERL, is an interpreted, object-oriented programming language that has gained popularity due to its clear syntax and readability. Python is said to be relatively simple to learn and portable, which means that its statements can be interpreted in a variety of operating systems such as UNIX-based systems, Mac OS, MS-DOS, OS/2, and various versions of Microsoft Windows 9,8. Guido van Rossum, a former Dutch resident whose favorite comedy group at the time was Monty Python's Flying Circus, created Python.

Python is a scripting language that can be used in Microsoft's Active Server Page (ASP) technology. Python is used to create the scoreboard system at the Melbourne Cricket Ground in Australia. The popular Web application server Z Object Publishing Environment is also written in Python.

PYTHON PLATFORM

Python implementation is available on 21 other platforms in addition to Windows, Linux, and macOS. Iron Python is a.NET framework-based Python implementation that can operate on Windows, Linux, and other platforms that support the.NET framework.

PYTHON LIABRARY

In the past, Machine Learning jobs were performed by manually coding all algorithms and mathematical and statistical formulas. As a result of numerous Python libraries, frameworks, and modules, it is now significantly simpler and more efficient than in the past. Python is currently one of the most popular programming languages for this task, and it has supplanted many other languages in the business, in part due to its extensive library collection. These are the Python libraries utilised in our project:

- Scikit-learn
- NumPy
- PyTorch
- Pandas
- Matplotlib
- Transformers
- Seaborn

NumPy:

NumPy is a popular Python toolbox for performing high-level mathematical operations on enormous multidimensional arrays and matrices. Machine Learning is widely used in essential scientific calculations. It is extremely useful for linear algebra, the Fourier transform, and random number functions. NumPy is used internally by sophisticated frameworks such as TensorFlow to manage Tensors.

Scikit:

Scikit-learn is a popular machine learning tool for classical ML approaches. It is built on two basic Python libraries, NumPy and SciPy. Scikit-learn supports the vast majority of supervised and unsupervised learning techniques. Scikit-learn may also be used for data mining and data analysis, making it an ideal tool for machine learning novices. This is how we separate our dataset into train and test sets.

PyTorch:

PyTorch is a popular open-source Machine Learning library for Python that is built on Torch, a C-based open-source Machine Learning toolkit with a Lua wrapper. It provides a wide range of tools and libraries to aid with Computer Vision, Natural Language Processing (NLP), and a variety of other ML applications. It enables developers to do GPU-accelerated Tensor calculations and assists in the building of computational graphs.

Pandas:

Pandas is a well-known Python data-analysis toolbox. It has nothing to do with Machine Learning. As we all know, before to training, dataset preparation is essential. Pandas is beneficial in this circumstance since it was originally built for data extraction and processing. It has complex data structures as well as a plethora of data analysis capabilities. It includes several methods for capturing, combining, and filtering data.

Matplotlib:

Matplotlib is a well-known Python data visualisation library. It is unrelated to Machine Learning, as is Pandas. It is very handy when a coder want to visualise data patterns. It's a 2D plotting and graphing library. Python plot simplifies charting for programmers by allowing them to customise line styles, font properties, axis formatting, and so forth. It provides a plethora of graphs and plots for data visualisation, such as histograms, error charts, bar charts, and so on.

Transformers:

Transformers is a library that provides many pre-trained models such as BERT and T5, which we may subsequently fine-tune for various applications. It offers APIs for easily downloading and training cutting-edge pre-trained models. Using pre-trained models reduces computer costs, carbon footprint, and training time.

CHAPTER IV: Methodology

In order to determine if a question submitted on Quora is serious or not, this chapter explains the methodology and tools used to train sentiment classifiers. The dataset's origin, characteristics, and use in this study are all covered in Section 4.1.

4.1 Dataset Description

Since the issue I'm attempting to address was the subject of a lucrative competition organised by Quora, the dataset for my research was collected from Kaggle. For this competition we are predicting wether a question asked on quora is sincere or Insincere.
A query that has the intent to make a statement rather than seek out useful responses is referred to as being insincere. Some traits that might indicate an insincere query include:

- Has a non-neutral tone
    - Has an exaggerated tone to underscore a point about a group of people
    - Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
    - Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype

    - Makes disparaging attacks/insults against a specific person or group of people
    - Based on an outlandish premise about a group of people
    - Disparages against a characteristic that is not fixable and not measurable
- Isn't grounded in reality
    - Based on false information, or contains absurd assumptions
- Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers.

The motive behind this competition has been clearly explained in the Introduction section.

The question that was posed and whether it was judged to be untrue are both part of the training data (goal = 1). The ground-truth labels have some noise in them; their accuracy cannot be guaranteed.

Training Data Fields

• qid- or a special question identifier

• question text - the text of a Quora question.

• target: a "sincere" query has a value of "1," else "0,"

The data set made available by Kaggle includes 375,806 unlabeled test instances and 1,306,122 labelled training examples in csv format.

Examples of Insincere questions :

- Is there a thing straight women like, do they exist?
- Was Adolf Hitler bad?
- Has the United States become the largest dictatorship in the world?

Examples of Sincere questions:

- Which novels are good for amateur readers?
- What is more important between work and love?
- What are the pros and cons of homeschooling?

In addition to the training and test datasets, Kaggle also offers us 4 alternative pre-trained word embeddings to aid in the model development process. These include the word embeddings in Paragram, Stanford's glove, Facebook's fastText, and Google's word 2 vec.

	qid	question_text	target
0	00002165364db923c7e6	How did Quebec nationalists see their province...	0
1	000032939017120e6e44	Do you have an adopted dog, how would you enco...	0
2	0000412ca6e4628ce2cf	Why does velocity affect time? Does velocity a...	0
3	000042bf85aa498cd78e	How did Otto von Guericke used the Magdeburg h...	0
4	0000455dfa3e01eae3af	Can I convert montra helicon D to a mountain b...	0
5	00004f9a462a357c33be	Is Gaza slowly becoming Auschwitz, Dachau or T...	0
6	00005059a06ee19e11ad	Why does Quora automatically ban conservative ...	0
7	0000559f875832745e2e	Is it crazy if I wash or wipe my groceries off...	0
8	00005bd3426b2d0c8305	Is there such a thing as dressing moderately, ...	0
9	00006e6928c5df60eacb	Is it just me or have you ever been in this ph...	0
10	000075f67dd595c3deb5	What can you say about feminism?	0
11	000076f3b42776c692de	How were the Calgary Flames founded?	0
12	000089792b3fc8026741	What is the dumbest, yet possibly true explana...	0
13	000092a90bcfbfe8cd88	Can we use our external hard disk as a OS as w...	0
14	000095680e41a9a6f6e3	I am 30, living at home and have no boyfriend....	0
15	0000a89942e3143e333a	What do you know about Bram Fischer and the Ri...	0
16	0000b8e1279eaa0a7062	How difficult is it to find a good instructor ...	0
17	0000bc0f62500f55959f	Have you licked the skin of a corpse?	0
18	0000ce6c31f14d3e09ec	Do you think Amazon will adopt an in house app...	0
19	0000d329332845b8a7fa	How many baronies might exist within a county ...	0

Figure 13. Training Dataset description

4.1.2 Target Distributions

Despite the quantity of the training dataset, the distribution of the targets is relatively unbalanced. Only 80,810 questions from the training set (6.2% of all questions) are genuinely earnest, whereas 1,225,312 questions (93.8% of all questions) are. This is one of the dataset's greatest issues that we must take into consideration when we create our machine learning and deep learning models, and it is the key reason why this competition selected the F1 Score as an assessment metric.

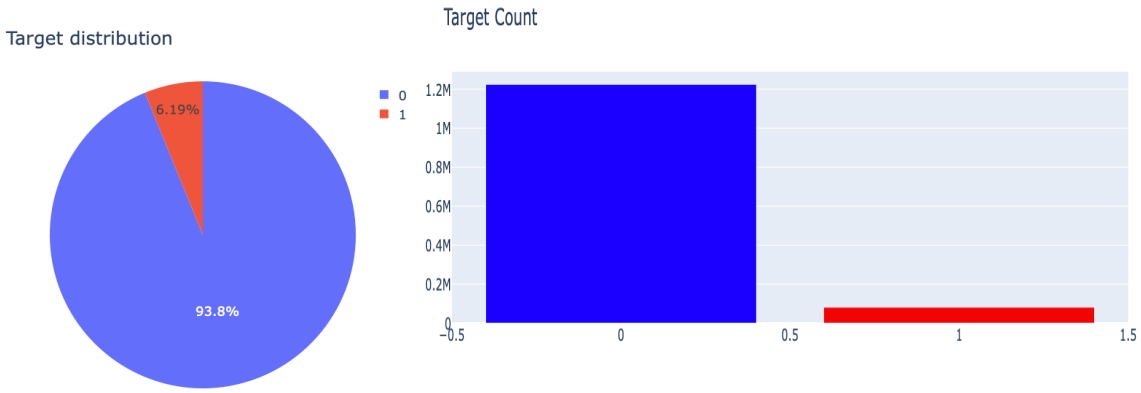

Figure 14. Target variable distributions

4.2 Exploratory Data Analysis

4.2.1 N-gram analysis

An n-gram is a continuous sequence of n elements from a given sample of text or voice in the fields of probability and computational linguistics. Depending on the application, the objects may be syllables, phonemes, letters, or words. The text or speech corpus is where the n grammes are gathered.

For us to properly interpret the text data that has been provided, n-grams are highly helpful. They aid in our comprehension of the words used to make an inquiry. Several preprocessing approaches must be used in order to do an n-gram analysis on our dataset. To begin, we must separate the text into tokens and eliminate all stop words. The next step is to obtain N-grams from NLTK library and to maintain the count of each n-gram in python dictionary.

Stop words are words that are used in a stop-list and are filtered out either before or after text is preprocessed since they are unimportant and do not add to a word's meaning.

NLTK is a well-known framework for creating Python applications that operate on textual data. Along with a collection of text preparation modules for categorization, tokenization, stemming, tagging, parsing, and semantic reasoning, it offers an intuitive interface to tools like Word-net.

On both sorts of questions for this exercise, we used three different n-grams: uni-gram, bigram, and trigram. The outcomes of doing unigram, bigram, and tri-gram on genuine and fake questions. The findings of a unigram analysis are displayed in the figure below.

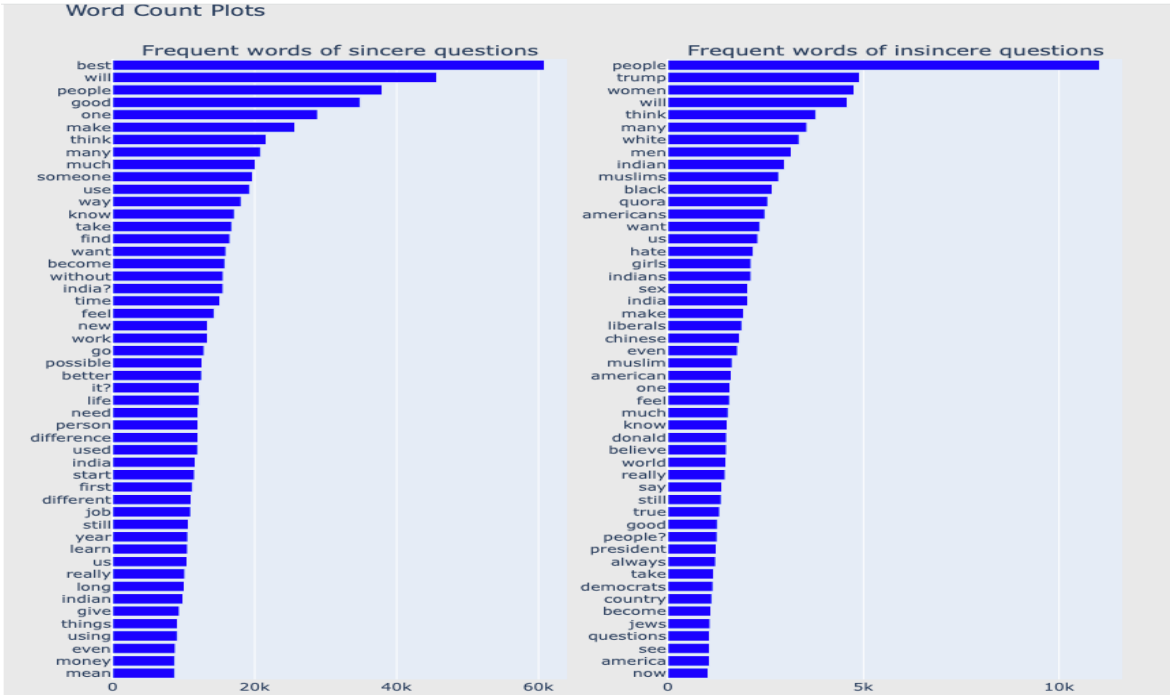

Figure 15. Unigram Analysis

The findings show that many terms have been put into the target value categories that were provided to us. The most often used terms in recorded insincere queries are people, trump, women, white, hatred, girls, etc. By using negative terms, it may be claimed that these words contribute something to the sentence. Sincere queries also frequently use words like "best," "people," "good," "possible," and "better," which give a phrase a positive tone. Additionally, it has been noted that both sorts of queries use a few words like "will," "think," "many," etc.

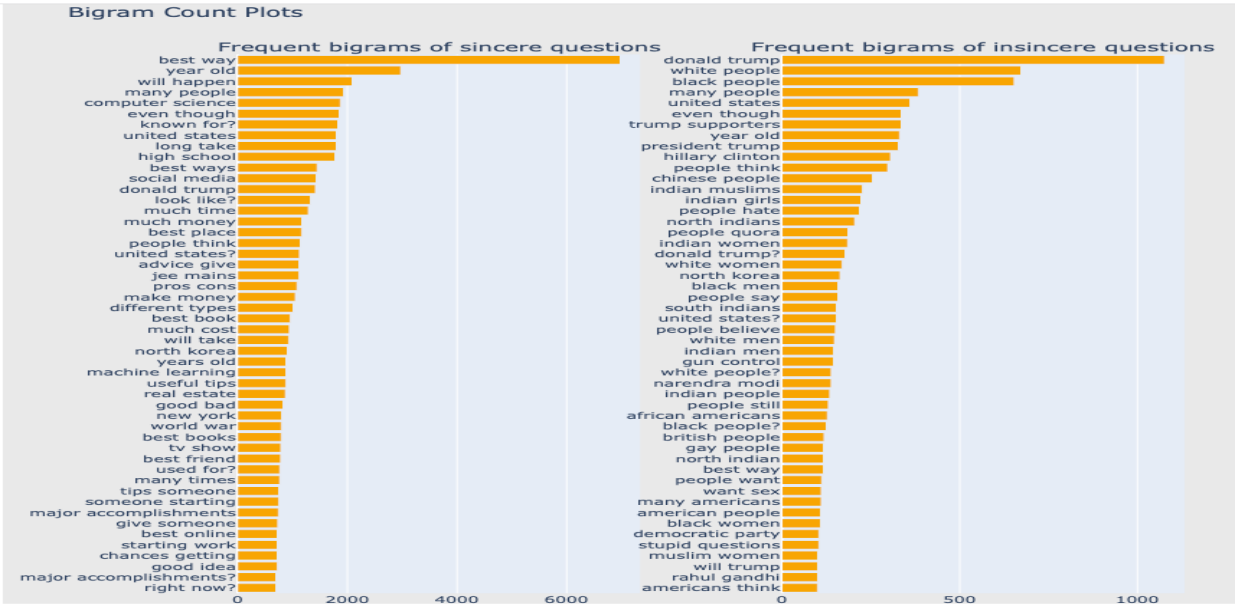

Figure 16. Bigram Analysis

The findings show that many terms have been put into the target value categories that were provided to us. The most often used terms in recorded insincere queries are white people, black people, Donald Trump, Indian ladies, Indian Muslims, etc. By using negative terms, it may be claimed that these words contribute something to the sentence. The previous data shows how the amount of words in our inquiry text has significantly decreased due to the extremely low number of serious questions. Like, sincere queries frequently include phrases like "best way," "many individuals," "computer science," "pros," "con," and "making money," among others, giving the sentence a more positive tone. Since there are a lot of Sincere questions in the dataset, the word frequencies are essentially identical to what was shown in the previous figure. Additionally, it has been noted that both sorts of inquiries contain a few terms like Donald Trump, many people, people think, etc. However, they do not occur with the same frequency.

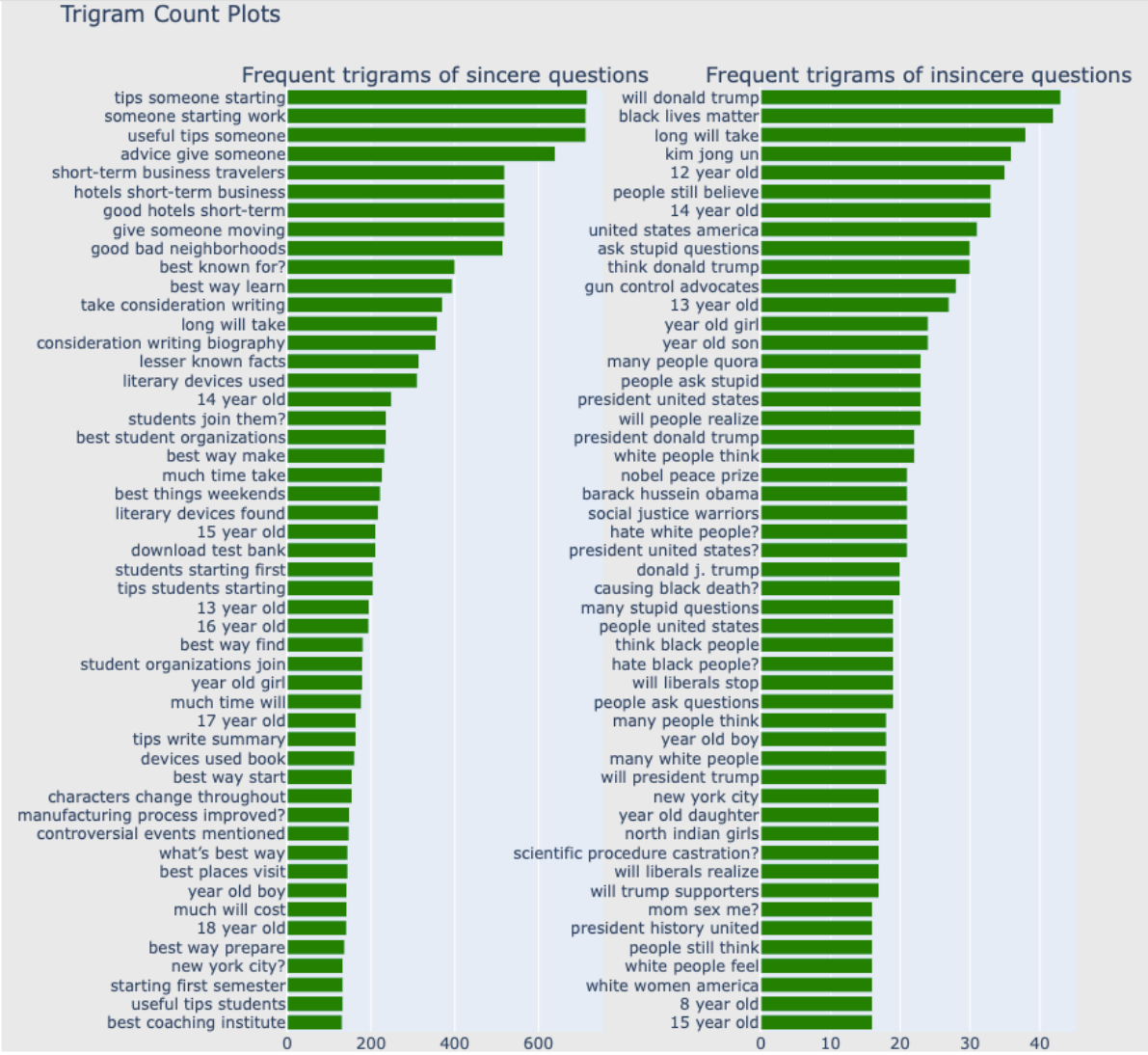

Figure 17. Trigram Analysis

The findings show that many terms have been put into the target value categories that were provided to us. The most often used phrases in honest inquiries were "white people," "black lives matter," "would Donald Trump," "Indian ladies," "Kim Jong Un," and "united States of Americas." It might be argued that these words make a statement and support the use of delicate language or direct personality commentary. The previous data shows how the amount of words in our inquiry text has

significantly decreased due to the extremely low number of serious questions. Similar to this, real queries frequently include phrases with positive connotations, such as "advice provide someone," "excellent hotels short-time best way starter," and "best method starting." When asking a question on Quora, these terms seem to be a crucial component of many statements. Since there are a lot more sincere questions in the dataset, the word frequencies have altered, but it's still remarkable to see the highest word count in the area of 600 to 800, indicating that out of the 1.32 million questions in the dataset, some are similar or indicate a shared interest.
Similar results have been obtained for n =4 N-gram plot.

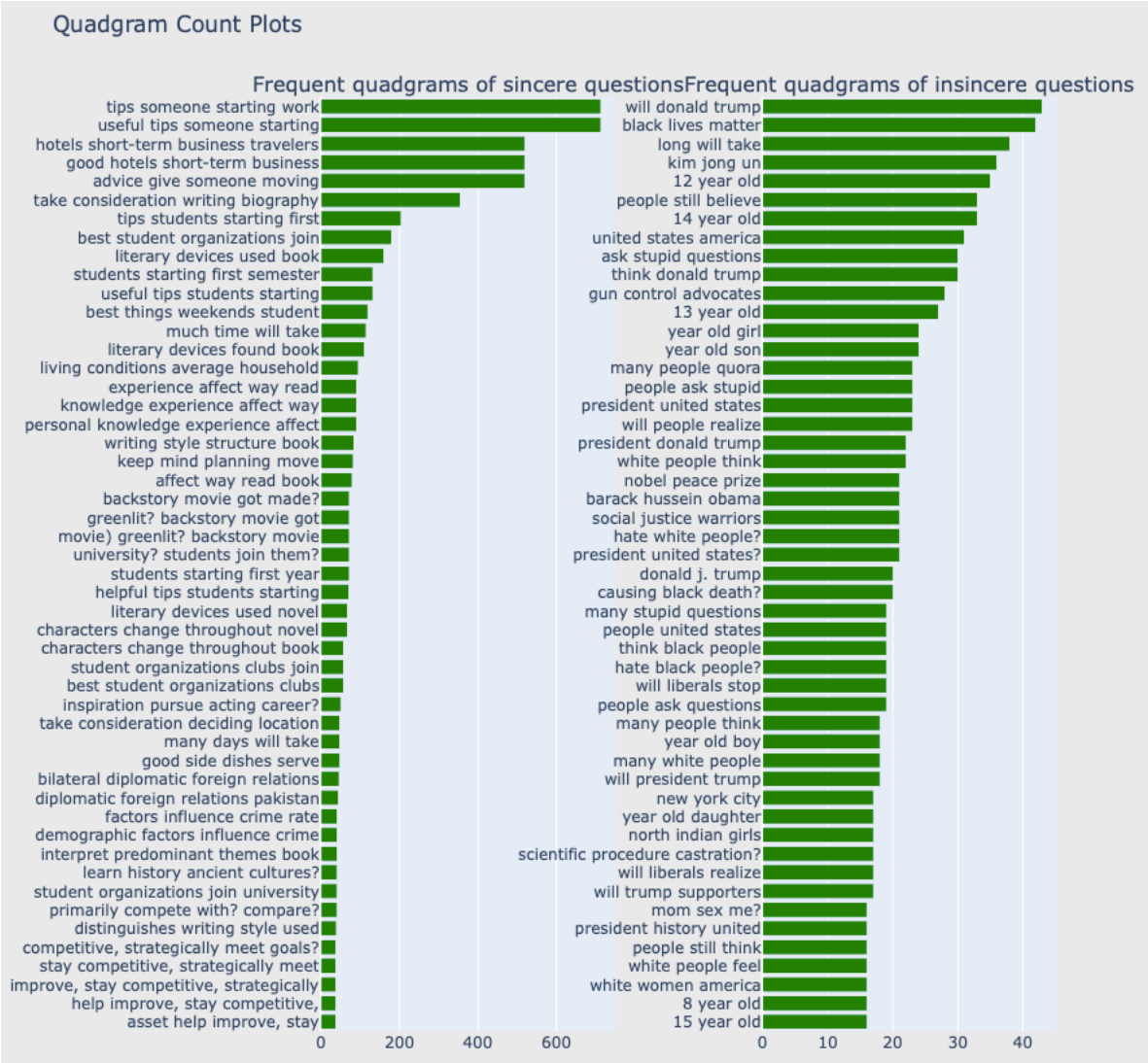

Figure 18. Quadgram Analysis

4.2.2 Meta features

Now let us create some meta features and then look at how they are distributed between the classes. The ones that we will create are

1. Number of words in the text - calculating the length of each words given in the equation text.
2. Number of unique words in the text - to calculate the number of words which are rare in a given sentence.
3. Number of characters in the text - to calculate the numbers of characters used in the making of a question text.
4. Number of stopwords - To calculate the number of stop words in a text.
5. Number of punctuations - To calculate the number of punctuations used in a text.
6. Number of upper case words - To calculate the number of words following the rules of upper cased word representation in a text.
7. Number of title case words - To calculate the number of words used to make a title in a text.
8. Average length of the words - To calculate the average length of the words used in a text.

After executing our code and looking at the dataset, it is apparent that there are more columns, each of which provides data for our model to operate on. To illustrate the significance of the characteristics we developed, I prepared a few box plots with data on the number of words, characters, and punctuation in each class as well as the mean number of words and a few examples that tend to have a lot of words and may be regarded as outliers.

However, unexpectedly, it may be deduced from the findings that inquiries that aren't honest have more character than questions that are. So, this may be a feature in our model that is valuable.

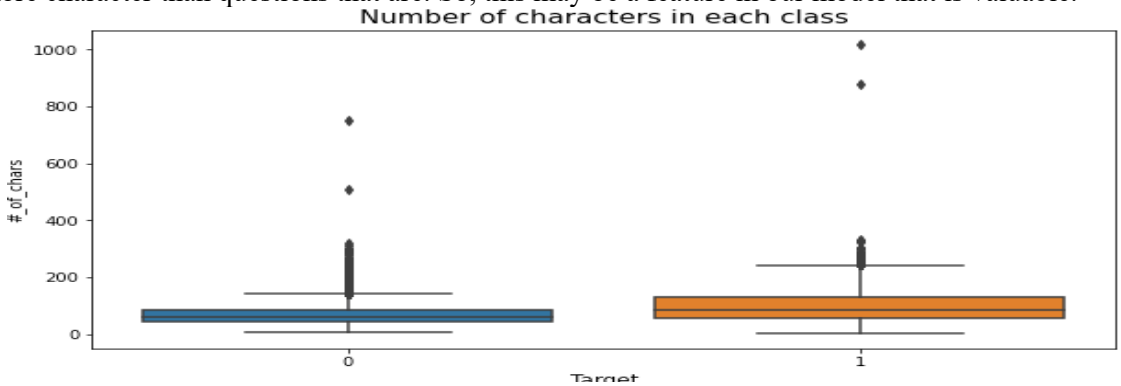

Figure 19. Number of characters in each class

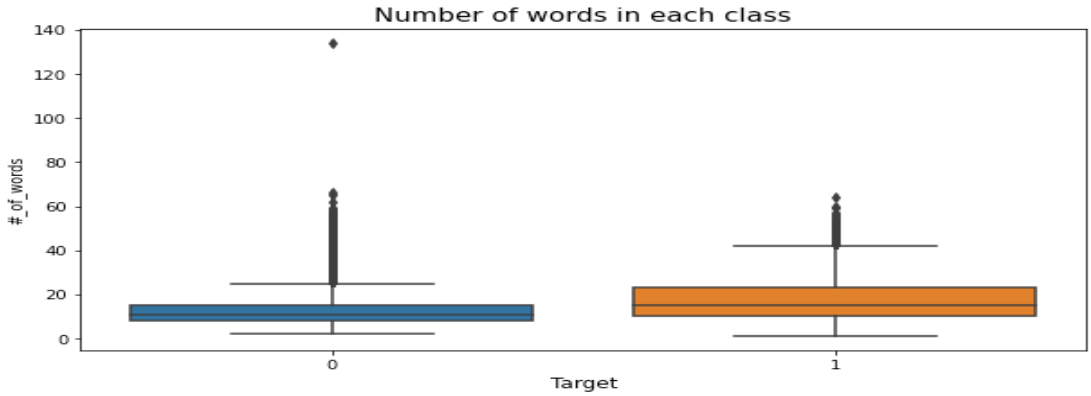

Figure 20. Number of words in each class

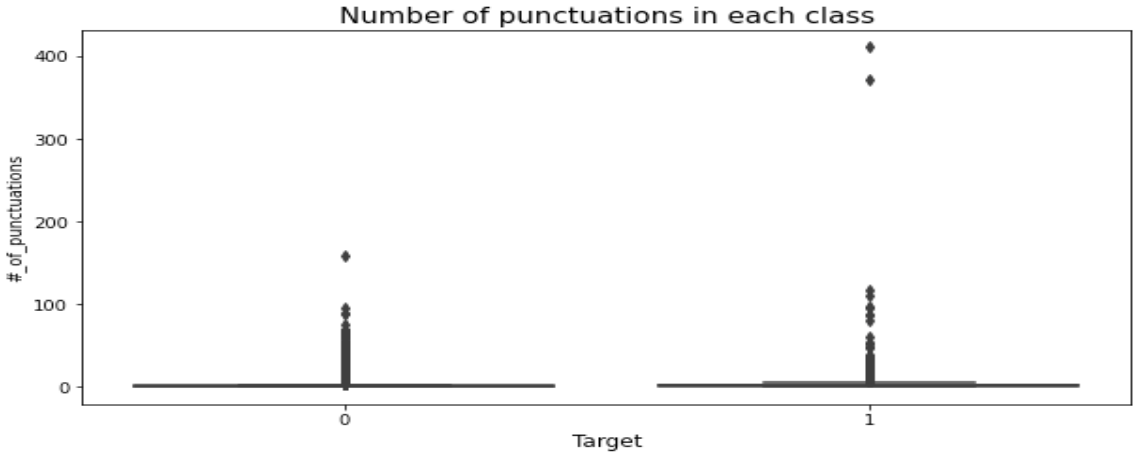

Figure 21. Number of punctuations in each class

4.2.3 Word Clouds

Word clouds are a text data visualisation approach where each word is shown with its frequency or relevance in the context. When used to grasp the main points of the day's news or the content of any YouTube channel, etc., this tool is quite helpful. It comes from the NLTK toolbox. I created distinct word clouds for serious and insincere questions in order to better comprehend the dataset.

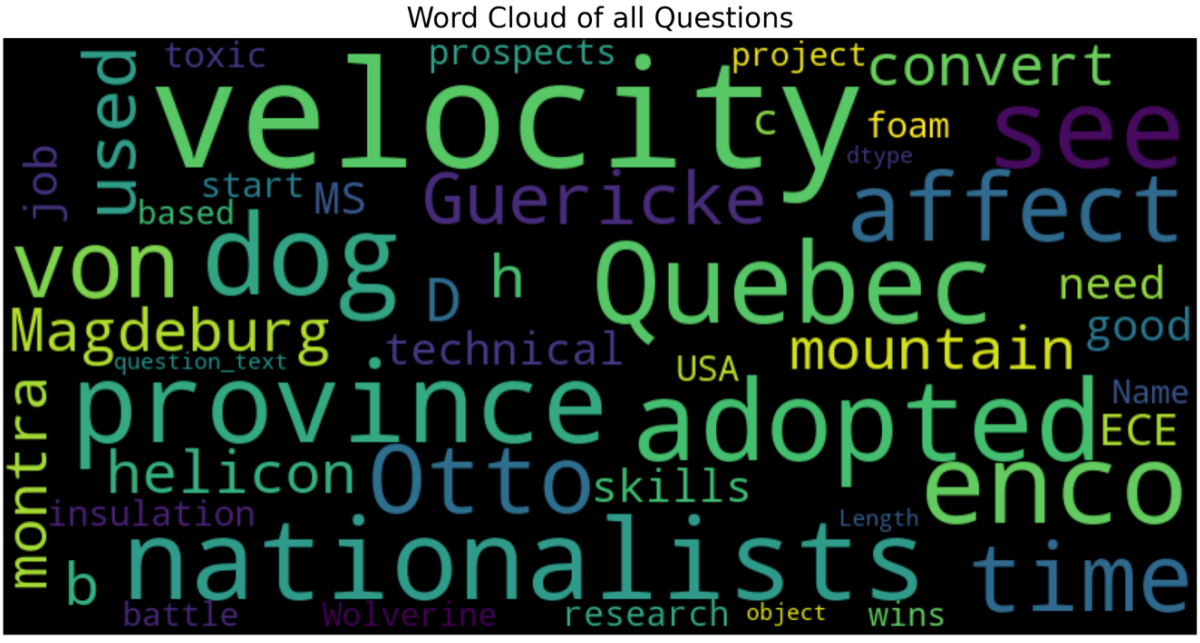

Figure 22. Word cloud for Training Dataset

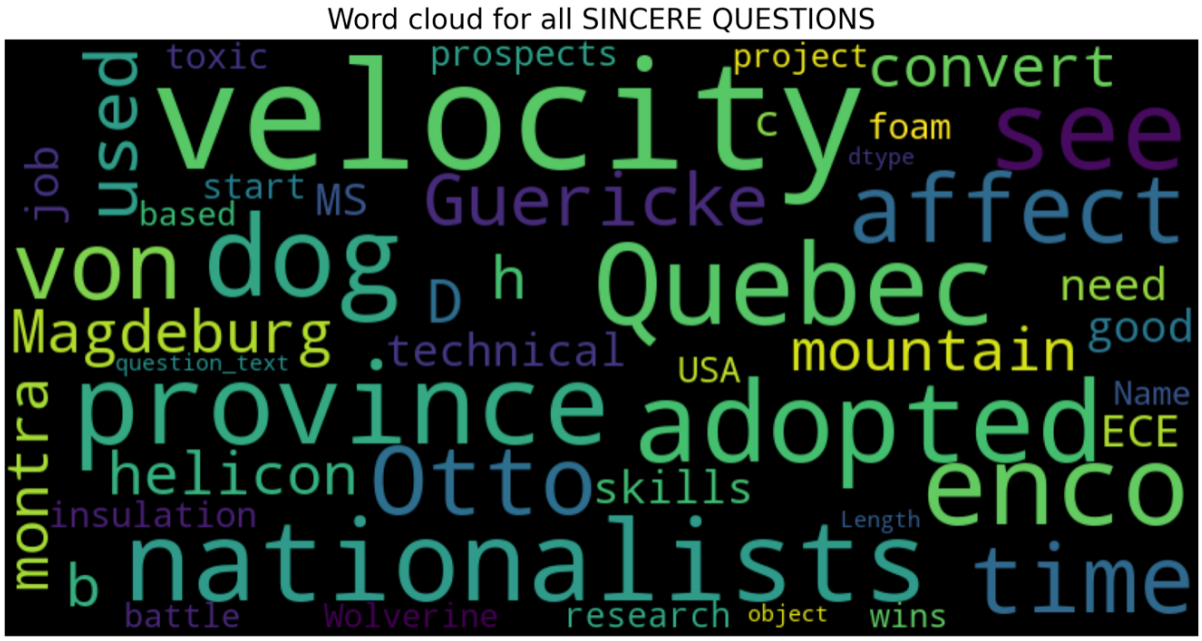

Figure 23.Worcloud for all Sincere Questions

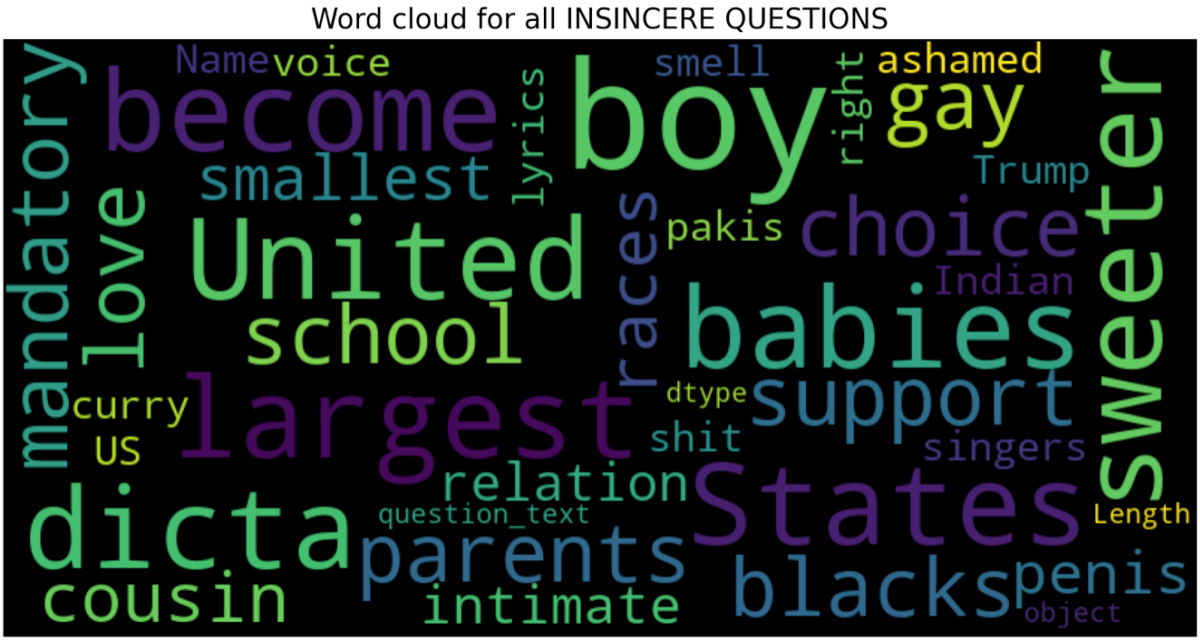

Figure 24. Word Cloud for all Insincere Questions

For an Insincere question Following unigram analysis, I discovered that several suggestive terms were utilised, which supports my hypothesis. Sincere inquiries also produced similar outcomes.

4.2.4 TF-IDF Featurization

To decide which terms are important as a feature, TF-IDF uses inverse document frequency and term frequency. In a manner similar to bag-of-words, TF-IDF uses data on term frequency in each document to identify common terms. However, overused terms like "the," which exist in every paper, are useless. Therefore, inverse document frequency aids in the weighting and discounting of words that are present in all texts, allowing for the discovery of relevant features in key phrases. To learn more about the significance of unique terms in our text corpus, we are introducing TF-idf scores. The more points received, the more our model is able to determine whether a question is sincere or not. Since there are far fewer data points for Insincere questions than for Sincere questions, it is necessary for our model to learn about the importance of words in terms of their frequency in order to compare our results using the F1-Score metric. Important words in Sincere or Insincere questions receive positive scores, while words that are frequently used but add no meaning to the text receive negative scores. In the figure below, the findings are displayed.

Weight	Feature	Weight	Feature
+19.705	castrated	+9.710	republicans
+17.520	muslims	+9.676	tamilians
+17.263	democrats	+9.523	bullshit
+17.076	liberals	+9.471	moron
+16.708	castrate	+9.470	losers
+15.554	indians	+9.457	terrorists
+14.486	trump	+9.439	raping
+14.411	americans	+9.427	fucking
+14.167	women	+9.423	moderators
+14.089	blacks	+9.398	shithole
+13.704	jews	+9.372	dick
+13.236	feminists	+9.330	palestinians
+12.927	atheists	+9.268	pakistanis
+12.569	castration	+9.229	europeans
+12.429	obama	+9.216	bhakts
+12.185	homosexuals	+9.191	liberal
+11.815	hillary	+9.127	jewish
+11.768	hindus	+9.077	penis
+11.765	rape	+9.062	homosexual
+11.724	fuck	+9.023	turks
+11.495	shit	+8.960	africans
+11.469	idiots	+8.952	nonsense
+11.318	muslim	+8.901	tennessee
+11.263	girls	+8.832	asshole
+11.208	christians	+8.818	mexicans
+11.085	gay	+8.815	hypocrisy
+10.990	whites	+8.702	cousin
+10.893	holocaust	+8.652	israelis
+10.784	asians	+8.638	assholes
+10.770	stupid	+8.622	realize
+10.763	tamils	+8.592	clinton
+10.729	ass	+8.562	canadians
+10.719	gays	+8.536	israel
+10.640	jew	+8.511	indian
+10.574	chinese	+8.508	bengalis
+10.468	incest	+8.508	uneducated
+10.444	leftists	+8.493	brits
+10.381	black	+8.472	alabamians
+10.358	crap	+8.455	transgender
+10.291	men	+8.370	bitch
+10.185	homosexuality	+8.344	morons
+10.153	white	+8.342	leftist
+9.995	conservatives	+8.312	sister
+9.967	idiot	+8.276	supporters
+9.773	brahmins	+8.269	democrat
+9.712	modi	+8.244	aunty
		+8.213	females

Figure 25. Generated tf-idf scores for all words

4.3 Logistic Regression

A method of supervised machine learning where we typically start with an input X and use our prediction function to obtain an output (y^). Then, we may contrast the output we produced (y^) with the real result in a specific paragraph. The cost that we utilised to update the parameters is provided by this.

Given a question text, we may express it as a vector of dimension V, where V stands for our vocabulary size. In that vector space, we can then add an outcome from our model. As the model continues to run, additional outputs are added to the vector space, increasing the size of V and making the vector space increasingly sparse due to the extremely unbalanced nature of the data.

Logistic regression makes use of sigmoid function which outputs a probability between 0 and 1. The sigmoid function with some weight parameter θ and some input x^{(i)}$x(i)$ is defined as follows:- h(x^(i), θ) = $1/(1 + e^{(-\theta^T*x^(i))}$.Because the sigmoid algorithm returns values between -1 and 1, we may categorise the predictions based on a certain cutoff. (say : 0.5).

The sigmoid function's denominator increases when (θ^T)x(i) approaches zero, which causes the sigmoid to approach zero as well. On the other hand, when (θ^T)x(i) approaches 1, the sigmoid function's denominator approaches 1, which causes the sigmoid to approach 1 as well.

4.3.1 Cost functioning Gradient Descent

The logistic regression cost function is defined as
$J(\theta)=(-1/m)* \sum i=1$ to m $[y(i)\log(h(x(i),\theta)+(1-y(i))\log(1-h(x(i),\theta))]$

We aim to reduce cost by improving the theta using the following equation:
$\theta j:=\theta j-\alpha*\partial J(\theta)/\theta j$

Here, α is called the learning rate. The above process of making hypothesis (h) using the sigmoid function and changing the weights (θ) using the derivative of cost function and a specific learning rate is called the Gradient Descent Algorithm.

Using the k-fold cross validation approach, I divided my data in order to perform my logistic regression. where 5 folds are the specified number. finding the predicted f1-score with a threshold in the range of 0.1 - 0.25 and a step size of 0.01 and providing us with more findings. The graphic below displays the outcomes of the logistic regression. The best F1 Score, according to the data, is obtained with a threshold of 0.17, giving us a score of 0.5969. The findings may be used to draw conclusions about other models that will be implemented later and can be used as our baseline model, proving that linear regression is not a good model to calculate the f1-scores for this dataset.

```
F1 score at threshold 0.1 is 0.5686754495282179
F1 score at threshold 0.11 is 0.5766786085750607
F1 score at threshold 0.12 is 0.5837343484402308
F1 score at threshold 0.13 is 0.5897296495823655
F1 score at threshold 0.14 is 0.5930953833638397
F1 score at threshold 0.15 is 0.595783588912981
F1 score at threshold 0.16 is 0.596408595819841
F1 score at threshold 0.17 is 0.596942968279187
F1 score at threshold 0.18 is 0.5959782669579342
F1 score at threshold 0.19 is 0.5941465645364126
F1 score at threshold 0.2 is 0.5927026869499329
F1 score at threshold 0.21 is 0.5898798041252412
F1 score at threshold 0.22 is 0.5871947354122015
F1 score at threshold 0.23 is 0.5841088485518597
F1 score at threshold 0.24 is 0.5807921595555416
```

Figure 26. Results obtained from Logistic regression

4.4 Pre-Processing of data

Before developing any Deep Learning Model it's important to clean the text using conventional NLP text preprocessing techniques like :

- Replacing url and math equations with common abbreviations.
- Cleaning punctuations.
- Correcting misspelled words.
- Removing stop-words from text corpus.
- Cleaning word-contractions.
- Using word-lemmatization techniques.

All of these methods were adapted from popular methods for cleaning up textual data; to use them, include the nltk library and create additional functions for each of the aforementioned jobs. Then, a single function is created by concatenating all of these functions. The pre-processed text is created using word lemmatization techniques, which is a linguistics procedure for combining a word's inflected forms so they can be analysed as a single item. We see this function to pass our input dataset.

Additionally, our function accepts our input training data as an input, and following the cleaning of our input dataset, a pre-processed question text is produced. The graphic below makes it possible to see the outcome.

	qid	question_text	target	preprocessed_question_text
0	00002165364db923c7e6	How did Quebec nationalists see their province...	0	How Quebec nationalist see province nation 1960s?
1	000032939017120e6e44	Do you have an adopted dog, how would you enco...	0	Do adopted dog encourage people adopt shop?
2	0000412ca6e4628ce2cf	Why does velocity affect time? Does velocity a...	0	Why velocity affect time? Does velocity affect...
3	000042bf85aa498cd78e	How did Otto von Guericke used the Magdeburg h...	0	How Otto von Guericke used Magdeburg hemispheres?
4	0000455dfa3e01eae3af	Can I convert montra helicon D to a mountain b...	0	Can I convert montra helicon D mountain bike c...
5	00004f9a462a357c33be	Is Gaza slowly becoming Auschwitz, Dachau or T...	0	Is Gaza slowly becoming Auschwitz Dachau Trebl...
6	00005059a06ee19e11ad	Why does Quora automatically ban conservative ...	0	Why Quora automatically ban conservative opini...
7	0000559f875832745e2e	Is it crazy if I wash or wipe my groceries off...	0	Is crazy I wash wipe grocery off? Germs everyw...
8	00005bd3426b2d0c8305	Is there such a thing as dressing moderately, ...	0	Is thing dressing moderately different dressin...
9	00006e6928c5df60eacb	Is it just me or have you ever been in this ph...	0	Is phase wherein became ignorant people loved ...
10	000075f67dd595c3deb5	What can you say about feminism?	0	What say feminism?

Figure 27. Dataset after Pre-Processing

4.3 Use of Word Embeddings

With the use of word embeddings, words may be represented in numerical vectors that can reflect the context, semantics, and syntactic characteristics of the languages used in documents. Words having comparable contexts or meanings would be considered "similar" in the vector space (by computing the cosine similarity between the word embeddings of two words). Along with the dataset that may be

utilised in the models, Kaggle gives the contestants four different types of word embeddings. These are listed below:

- GoogleNews-vectors-negative300
- glove.840B.300d
- paragram_300_sl999
- wiki-news-300d-1M

Pre-trained glove word embedding (300 dimensions for 2.2 million vocabulary) was employed for our research. Global vectors is what Glove stands for.
From Stanford's official glove page, we downloaded a glove.
When we unzipped the file, we discovered that it contains a text file with vector space trained to associate words that are closer together in the vector space, and the outcome is an embedding matrix.
GloVe is a technique for producing vector representations of words using unsupervised learning. Training is done using corpus-based global word-word co-occurrence statistics, and the resultant representations show off some of the word vector space's fascinating linear substructures. The global word-word co-occurrence matrix, which tracks how frequently words appear alongside one another in a particular corpus, contains non-zero entries that are used to train the GloVe model. It takes a single trip over the full corpus to get the statistics needed to populate this matrix. This step can be computationally expensive for big corpora, but it only has a one-time upfront cost. Because there are often fewer non-zero matrix entries than there are words in the corpus, subsequent training iterations are completed considerably faster.
The co-occurrence statistics collection and preparation for entry into the model are automated by the tools included in this package. These preparatory processes are segregated from the main training code, which may be run separately. GloVe is essentially a log-bilinear model with a weighted least-squares objective. The main intuition underlying the model is the simple observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning.
GloVe's training goal is to acquire word vectors whose dot product equals the logarithm of the likelihood of the words occurring together. This objective links (the logarithm of) ratios of co-occurrence probability with vector differences in the word vector space since the logarithm of a ratio equals the difference of logarithms. This information is also recorded as vector disparities since these ratios are capable of encoding some kind of significance. As a result, the word vectors produced excel at word analogy tasks like those investigated by the word2vec package.
Conclusion: The co-occurrence matrix and words can be related semantically.

4.4 Bi-directional LSTM with Attention Mechanism

The LSTM types of RNN architecture are summarized. Language translation, driver identification, and human activity recognition are just a few of the time series prediction tasks that LSTM have proven effective at in recent years (Definition of Neural Network - Gartner Information Technology Glossary, 2022) (Ramachandran et al., 2022). Because LSTMs' internal artificial memory is capable of quickly capturing the temporal connections present in a sequential dataset, modelling sequential tasks with LSTMs is successful. The three "gates"—forget gate, input gate, and output gate—that control the internal states of LSTM are referred to as "memory blocks." Together, these gates build a memory within the network that stores the temporal relationships in the data.
Information is exclusively transmitted from the past to the future inputs in a unidirectional LSTM (Vaswani et al., 2022). The extension of unidirectional LSTM known as bidirectional LSTM allows information to travel to and from both the past and future time steps (Ramachandran et al., 2022). Bidirectional LSTM performs better than unidirectional LSTM as a consequence.
Additionally, standard LSTM memory blocks make an effort to learn a single vector representation, but attention mechanisms enable LSTM to develop the ability to attend to (focus on) the most crucial input sequence vectors. In order to extract the most significant information from our textual data for this project, as illustrated in Figure, we merged a bidirectional LSTM with an attention mechanism to develop a "bidirectional" LSTM with an attention model (Ramachandran et al., 2022) .

Prior to being placed in the embedding layer of the bidirectional LSTM, the original query text is tokenized and embedded using the GloVe embedding. Then, two directions are selected with a hidden size of 128 for the bidirectional LSTM design. By tokenizing the data, the original text is changed into a string of numbers. Once the vocabulary has been established, the encode technique is used to obtain the token for each word in the corpus. To ensure that every sequence is the same length, we must pad the sentences with zeros at the end. We won't be able to train the model on batches if that happens. We know that particular terms carry greater weight when determining whether the queries are honest or not, thus the attention mechanism would allow the model to focus on those words rather than the entire phrase. The architecture of this model is implemented as follow:

1. The weight in the embedding layer is the pre-trained GloVe embedding given by Kaggle.
2. Two layers of bidirectional LSTMs are fed word embedding features.
3. Each LSTM layer's output (hidden states) is subjected to an attention method.
4. The output of the second layer of the LSTM is subjected to maximum pooling and average pooling.
5. The fully connected layer receives concatenated features from stages ((3) and (4)) for classification.

The general architecture of this model is depicted in the diagram below. Adam is used to improve the training process with binary cross entropy as the loss function. We also used early halting and dropout with p=0.1 to stop the model from overfitting. I chose to train the model for 10 epochs because we noticed that the validation loss started to rise after 5 epochs while the training loss kept falling. k-fold cross-validation is used to adjust the learning rate. The entire training and inference procedure took around 12 minutes using Kaggle's GPU. On the test set, we were able to get a f1 score of 0.651 after determining the appropriate decision threshold which in itself is an improvement on our baseline model Logistic Regression.

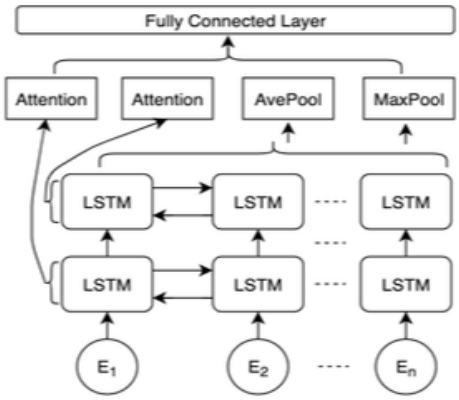

Figure 28. Bidirectional LSTM with Attention Mechanism Architecture

4.5 Pre-trained BERT Implementation

Although pre-trained Bert models are not permitted in this competition, I was curious to know the outcomes if they were, given they represent the state-of-the-art model for NLP applications now. Bert models are built on transformers that remove the recurrent units and substitute a self-attention mechanism and positional encoding in their place. The goal of my final step is to train my dataset using the Hugging Face Bert-base-cased model.

4.5.1 Model Description

BERT is a transformers model that was self-supervisedly pretrained on a sizable corpus of English data. This indicates that an automatic method was used to produce inputs and labels from those texts after it had been pretrained on just the raw texts without any human labelling (which explains why it may use a ton of data that is readily available to the public). It was pretrained with two goals in mind, specifically:

- Masked language modelling (MLM): given a text, the model randomly selects 15% of the words as input, runs the full sentence through the model, and then it must predict the words that were selected as input. This contrasts with conventional recurrent neural networks (RNNs), which typically see the words sequentially, and autoregressive models like GPT, which internally conceal the next tokens. This makes it possible for the model to learn a two-way representation of the statement.
- Next sentence prediction (NSP): During pretraining, the models combine two masked sentences as inputs. They occasionally match sentences that were adjacent to one another in the original text, and sometimes they don't. The model must then forecast whether or not the two phrases were in order.

A standard classifier can be trained using the features produced by the BERT model as inputs if you have a dataset of labelled sentences, for example. In this way, the model learns an internal representation of the English language that can then be used to extract features useful for downstream tasks. This model is particularly intended for use in problems where the entire sentence (perhaps masked) is used to make judgments, such as sequence classification, token classification, or question answering.
Sentences A and B have a chance of 0.5 of being two consecutive sentences in the original corpus, but in the other circumstances, a random sentence from the corpus is used. Keep in mind that a sentence in this context refers to a block of material that is sequential and typically larger than a single sentence. The sole restriction is that the total length of the two "sentences" in the result must be less than 512 tokens. The specifics of how each sentence was masked are as follows:

- 15% of the tokens are masked.
- In 80% of the cases, the masked tokens are replaced by [MASK].
- In 10% of the cases, the masked tokens are replaced by a random token (different) from the one they replace.
- In the 10% remaining cases, the masked tokens are left as is.

The recurrent units are removed from Bert since it is built on the transformers architecture, and they are replaced with a self-attention mechanism and positional encoding. The benefit of the transformers model is that unlike recurrent neural networks, which analyse sentences word by word, the transformers model processes sentences as a whole. Since the hidden state of a standard RNN's unit depends on the prior input sequence, it is hypothesised that self attention and positional encoding will allow the transformers to capture long-range dependency better than traditional RNNs. Since there is no connection between each input sequence, the transformer may also be learned concurrently. As a result, BERT models may be trained using a massive unlabeled dataset and hundreds of millions of parameters. Therefore, the language skills acquired on a huge dataset might be useful for jobs down the road, such text categorization.
On the GPU server of Kaggle , we adjusted the model using a batch size of 16 and an Adam optimizer with a 1.5e-6 learning rate. One epoch of training requires around 7 hours because to the large amount of parameters, which tends to surpass our runtime limit on Kaggle notebooks. I chose 2,000 insincere and 28000 sincere question texts at random to retain the dataset's initial imbalance to address this issue. So that the f1 score continues to be the ideal metric for calculating results.

4.5.2 Training of Pre-trained Bert

- transformers are installed
- Bert base cased model and it’s corresponding tokeniser is initialised.
- Inputs ids and attention masks are created for the training data
- Splitting our original dataset into training and validation set with validation ratio set at 0.2.
- Data loader is preparedly randomly sampling the train set and validation set.
- The performance metric(F1-Score) is defined.
- Finally the model is trained with a scaled dataset of around 30,000 datapoints for a total of 8 epochs, taking a total time of 1.5 hours for result evaluation.

The aforementioned Bert Model is trained using the newly created training dataset, and at the fourth epoch, a maximum f1-score of 0.7134 is produced, outperforming our previous deep learning model,

the Bi-directional LSTM with Attention Mechanism. A comparison plot of our model's training loss and validation loss later revealed that while our training loss continued to reduce, our validation loss first fell until the third epoch before finally increasing until the entire training process was complete. It is depicted in the diagram below.
We depict loss on the y axis of the figure and the number of epochs on the x axis.
Validation loss is depicted in orange, whereas training loss is depicted in blue.

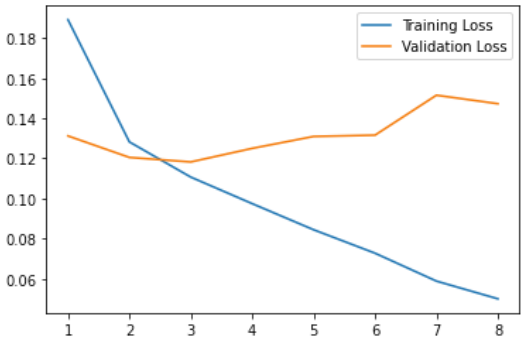

Figure 29. Training and validation loss for Experiment-1

4.5.2.1 Extended experimentation on BERT

Since the dataset given to us highly imbalanced but BERT still managed to deliver a high F1 Score of 0.8134. This made me interested in finding out how BERT will perform with the same textual dataset but with different imbalances .

- For my second experimentation I divided my dataset in the ratio of 1:1 for a comparably smaller size of 4000 points included.

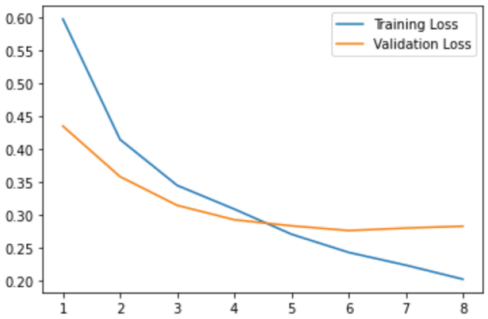

Figure 30. Training loss vs Validation loss for experiment 2

- For third experimentation training dataset comprised of 4000 Insincere and 6000 sincere questions achieving a F1 score of 0.7741 .

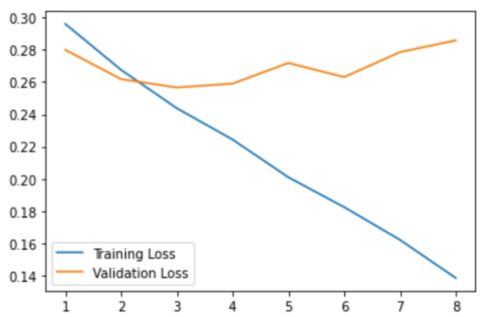

Figure 31. Training loss vs Validation loss Experiment-3

- The fourth experiment had an input comprising of 15000 datapoints in total having 5000 Insincere and 10000 Sincere questions.

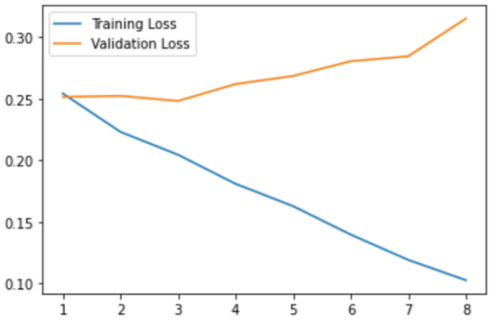

Figure 32. Training loss vs Validation loss Experiment-4

- For our final experimentation dataset was arranged according the original imbalance in the dataset and a F1 Score of 0.7233 was achieved.

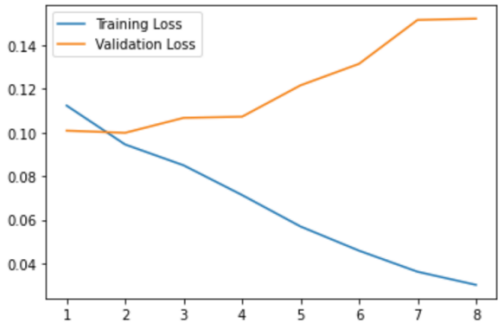

Figure 33. Training loss vs Validation loss Final Experiment

4.6 Summary

In this chapter we trained Logistic Regression , Bidirectional LSTM with Attention Mechanism and used state of art model BERT to record the maximum F1-Score achieved by any models. We observed that BERT performed best giving us a maximum score of 0.7134 out performing Logistic regression and Bi-LSTM with Attention from a huge margin.

We also observed the behaviour of Insincere Questions the way they are formulated and why they are a danger to any online media sites. Many text pre-processing techniques were used to clean our dataset so our model could be trained efficiently . Also certain optimization techniques like Gradient Descent and optimizer like Adam were optimized to enhace our model performance . Our training dataset was used for cross validation using K-cross validation technique and results were obtained.

Chapter V: Results and Discussions

The comparison of two models, Logistic Regression (Baseline model) and Bi-directional LSTM with Attention Mechanism, is summarised in Table 1. Logistic Regression was utilised as my baseline model, and it was applied with TF-IDF featurization, however the results were quite bad. To improve on the baseline model score, I utilised another deep learning model called Bi-directional LSTM with Attention Mechanism, which achieved a maximum F1 Score of 0.6517, an improvement over our baseline model score.

Table 1. Comparison of results obtained by comparing Logistic Regression and Bi-directional LSTM with Attention Mechanism

Models	F1 Score
Logistic Regression	0.596
Bidirectional LSTM with Attention	0.6517

Table 2 illustrates the results of Pre-trained BERT with different data sizes obtaining varying outcomes, although the time required to compute results is relatively high, which is offset by the gain in attained F1 Score. BERT took around 11 minutes to complete 8 epochs on a dataset of size 4000, achieving a maximum F1 Score of 0.7956 during the 5th epoch. The plot of training loss versus validation loss is shown below (Figure 30). The results are intriguing since the training and validation losses decline gradually, in contrast to the curve produced in the preceding image (Figure 29). Since our dataset is balanced.

The curve produced in (Figure 31) for the second experiment demonstrates that the training loss declined gradually, while the validation curve initially decreased and then rose due to the erratic behaviour. Furthermore, for 15000 datapoints, BERT performed well, yielding an F1 Score of 0.7621. The behaviour is identical to that seen in the preceding figure, except both losses crossed at the first epoch. Our most recent experiment with 40000 datapoints yielded a score of 0.7233, which is an improvement above our initial trial with roughly 30000 datapoints. The validation loss continued to rise, whereas the training loss continued to behave normally.

As a result, BERT may be described as a "state-of-the-art-model" for text categorization. Furthermore, with the exception of experimental -2, where we had an identical number of questions in our dataset, the recall scores for all of the best epochs were consistently lower than the corresponding accuracy scores. The imbalance in the dataset is impeding the performance of our models, thus we need resample our data and consider alternative techniques to finding a solution.

Table 2. Comparison of F1 Scores for different dataset sizes and ratio of distribution for Our target values

Model	Dataset Size	Insincere Questions	Sincere Questions	Time taken to train BERT	F1 Score
Pre-trained BERT	~30000	~ 2000	~28000	1.5 hours	0.7134
Pre-trained BERT	~4000	~2000	~2000	11 minutes	0.7956
Pre-trained BERT	~10000	~4000	~6000	29 minutes	0.7741
Pre-trained BERT	~15000	~5000	~10000	43 minutes	0.7621
Pre-trained BERT	~40000	~2500	~37500	2 hours	0.7233

CHAPTER VI: Conclusion

For this research I used Logistic Regression, Bi-directional LSTM with Attention, and BERT to classify insincere questions on Quora. We used simple model results as a baseline and then achieved higher F1 scores on deep models with the word embedding required by Kaggle. Among them, the fine-tuned BERT model performed best in both simple and deep models. However, I have learned a lot more about how to preprocess text data, build word models, and load pretrained word embedding, how model architectures look, how to implement those models using sklearn and PyTorch, and how to tune the hyperparameters in each model accordingly.

The performance metric F1 Score suggested by the competition is very useful for dealing with problems which have imbalanced datasets. Logistic Regression is one of the most popular classification models and simple to implement but it doesn't perform very well for a complex problem. Word Embeddings help us to save a lot of semantic information in the text and helping our models to understand the structure of a text in a better way than compared to the text pre-processing techniques. NLTK text pre-processing techniques have performed well for a long run but it's main focus is to reduce the dimensionality of the embedding matrix. This would have been beneficial if it was used for solving a different problem. For example, To find whether 2 sentences are duplicate of each other.

Bi-directional LSTM with Attention Mechanism performs well when compared to the time taken i.e. 12 minutes for a dataset this huge. Since it's uses GLoVe embedding in the embedding layer, trains our Bi-directional LSTM model. Further, the Attention mechanism enables the model to focus on important word in the sentences rather than whole sentence. As we know some words carry more weights and can be a factor in deciding whether a question is Sincere or Insincere.

Because of GPU limitations on the Kaggle platform, I was unable to train my entire dataset (1.3 million data points) on Bert. To solve this problem, I experimented with different number of Insincere and Sincere questions. Training with smaller and balanced datasets took less time in comparison to imbalanced datasets which took a long time for training and F1 Score was not improving. As a result, the f1-score remains the ideal metric for calculating results for an imbalanced sequential dataset .

References and citations

[1] Kaggle.com. 2022. Quora Insincere Questions Classification | Kaggle. [online] Available at: <https://www.kaggle.com/c/quora-insincere-questions-classification> [Accessed 19 September 2022].
[2] Waseem, Z., Davidon, T., Warmsley, D. & Weber, I., 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. *In Proceedings of the First Workshop on Abusive Language Online,* pp. 78-84.
[3] Educative: Interactive Courses for Software Developers. 2022. What is the F1-score?. [online] Available at: <https://www.educative.io/answers/what-is-the-f1-score> [Accessed 19 September 2022].
[4] Malo, P. et al., 2013. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology,* 65(4), pp. 782-796.
[5] K. Z. Aung and N. N. Myo, "Sentiment analysis of students' comment using lexicon based approach," 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), 2017, pp. 149-154, doi: 10.1109/ICIS.2017.7959985.
[6] Phan Trong Ngoc and Myungsik Yoo, "The lexicon-based sentiment analysis for fan page ranking in Facebook," *The International Conference on Information Networking 2014 (ICOIN2014)*, 2014, pp. 444-448, doi: 10.1109/ICOIN.2014.6799721.
[7] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '04). Association for Computing Machinery, New York, NY, USA, 168–177. https://doi.org/10.1145/1014052.1014073
[8] Uysal, A. and Gunal, S., 2014. The impact of preprocessing on text classification. *Information Processing & Management*, 50(1), pp.104-112.
[9] Salton, G., Wong, A. and Yang, C., 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11), pp.613-620.
[10] Uysal, A. and Gunal, S., 2014. The impact of preprocessing on text classification. *Information Processing & Management*, 50(1), pp.104-112.

[11] Song, F., Liu, S. and Yang, J., 2005. A comparative study on text representation schemes in text categorization. *Pattern Analysis and Applications*, 8(1-2), pp.199-209.

[12] Kostoff, R. and Geisler, E., 1999. Strategic Management and Implementation of Textual Data Mining in Government Organizations. *Technology Analysis & Strategic Management*, 11(4), pp.493-525.

[13] Zhang, Y., Gong, L. and Wang, Y., 2005. An improved TF-IDF approach for text classification. *Journal of Zhejiang University SCIENCE*, 6(1), pp.49-55.

[14] HUANG, C., YIN, J. and HOU, F., 2011. A Text Similarity Measurement Combining Word Semantic Information with TF-IDF Method. *Chinese Journal of Computers*, 34(5), pp.856-864.

[15] Occhipinti, A., Rogers, L. and Angione, C., 2022. A pipeline and comparative study of 12 machine learning models for text classification. *Expert Systems with Applications*, 201, p.117193.

[18] Baroni, M. and Lenci, A., 2010. Distributional Memory: A General Framework for Corpus-Based Semantics. *Computational Linguistics*, 36(4), pp.673-721.

[19] Turney, P. and Pantel, P., 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37, pp.141-188.

[20] Griffiths, T., Steyvers, M. and Tenenbaum, J., 2007. Topics in semantic representation. *Psychological Review*, 114(2), pp.211-244.

[21] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15). JMLR.org, 957–966.

[22] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, Lisbon, Portugal. Association for Computational Linguistics.

[23] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[24] Liu, G. and Guo, J., 2019. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337, pp.325-338.

[25] Hosseini, Marjan and Sabet, Alireza Javadian and He, Suining and Aguiar, Derek," Interpretable Fake News Detection with Topic and Deep Variational Models" [online] arXiv.org. Available at: <https://arxiv.org/abs/2209.01536>

[26] M. Yang, W. Tu, J. Wang, F. Xu, and X. Chen, "Attention Based LSTM for Target Dependent Sentiment Classification", *AAAI*, vol. 31, no. 1, Feb. 2017

[27] Li, W., Qi, F., Tang, M. and Yu, Z. (2020). Bidirectional LSTM with self-attention mechanism and multi-channel features for sentiment classification. *Neurocomputing*, 387, pp.63–77.

[28] Devlin, Jacob et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *ArXiv* abs/1810.04805 (2019): n. pag.

[29] Novikova, Jekaterina & Shkaruta, Ksenia. (2022). DECK: Behavioral Tests to Improve Interpretability and Generalizability of BERT Models Detecting Depression from Text. 10.48550/arXiv.2209.05286.

[30] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

[31] David Yenicelik, Florian Schmidt, and Yannic Kilcher. 2020. How does BERT capture semantics? A closer look at polysemous words. In Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 156–162, Online. Association for Computational Linguistics.

[32] Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of Workshop at ICLR. 2013.

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13). Curran Associates Inc., Red Hook, NY, USA, 3111–3119.

[34] Kaelbling, L., Littman, M. and Moore, A., 1996. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4, pp.237-285.

[35] Pennington, J., 2022. *GloVe: Global Vectors for Word Representation*. [online] Nlp.stanford.edu. Available at: <https://nlp.stanford.edu/projects/glove/> [Accessed 19 September 2022].

[36] Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of Workshop at ICLR. 2013.

[37] Gartner. 2022. Definition of Neural Network - Gartner Information Technology Glossary. [online] Available at: <https://www.gartner.com/en/information-technology/glossary/neural-net-or-neural-network> [Accessed 19 September 2022].

[38] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L. and Polosukhin, I., 2022. *Attention Is All You Need*. [online] arXiv.org. Available at: <https://arxiv.org/abs/1706.03762>

[39] Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A. and Shlens, J., 2022. *Stand-Alone Self-Attention in Vision Models*. [online] arXiv.org. Available at: <https://arxiv.org/abs/1906.05909>

[40] Medium. 2022. *Understanding the BERT Model*. [online] Available at: <https://medium.com/analytics-vidhya/understanding-the-bert-model-a04e1c7933a9>

[41] Circulation. 2022. *Logistic Regression*. [online] Available at: <https://www.ahajournals.org/doi/10.1161/CIRCULATIONAHA.106.682658> [Accessed 19 September 2022].
[42] 2022. *Deep Learning*. [online] Available at: <https://monkeylearn.com/blog/sentiment-analysis-deep-learning/> [Accessed 19 September 2022].

Appendix

Git Repository link provided by University of Birmingham:

https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2021/sxp215/-/tree/main

In order to run this code we need to follow the chronological sequence through which files have been arranged. All the required libraries needed to get the results have already been in .ipynb notebooks. The related datset .csv files (train and test) has been uploaded in the repository for reference.