University of Dublin

Trinity College

Artificial Neural Networks & Languages

Performing Sentiment Analysis using discrete machine learning architectures

Swaraj Shandilya
B.A.I. Computer Engineering

Final Year Project – August 2021

Supervisor Asst. Prof. Yvette Graham

School of Computer Science and Statistics
O'Reilly Institute, Trinity College, Dublin 2, Ireland

Declaration

I hereby declare that this work, except where otherwise stated, is entirely my own work. It has not previously been submitted as an exercise for a degree, either at Trinity College, or in any other university and that the library may lend or use it or any part thereof on request.

_____

Signature

August 2021

Acknowledgements

I would like to express my sincerest gratitude to the following people for their guidance and support towards the accomplishment of this final year project.

My supervisor Asst. Prof. Yvette Graham for her valuable contribution and guidance throughout my dissertation. She has been very patient with me and has given me valuable feedback after every meeting we had regarding the project. Without his continuous support and advice, this project would not have been possible.

Finally, the colleagues who tried my work, for their willingness to contribute, and their helpful feedback is greatly appreciated. I would like to thank my family and friends for their encouragement and continued support throughout these years in the university.

Abstract

Sentiment analysis can be used to better understand human psychology by gaining access to large amounts of data from online platforms. It enables efficient and cost-effective large-scale data processing. Large corporations and industries frequently want to know in real-time what consumers and the general public think of their products and services. Manually reading every post on the website and extracting useful viewpoint information from it, on the other hand, is not feasible. If you do it manually, there is far too much data. This project investigates sentiment analysis's application in the business to understand its strengths and limitations in order to learn more about it. As a data set, we use product users' review comments about products and reviews about retailers from Amazon, and we classify review text based on subjectivity/objectivity and buyer's negative/positive attitude. Using Natural Language techniques, we identified the potential benefits of Sentiment Analysis in this project. To implement it, we created Deep Neural Network architectures such as Convolutional Neural Networks and Recurrent Neural Networks, as well as state-of-the-art Bidirectional Encoder Representations from the Transformers architecture. The final implementation of this pre-trained model is done on a web application that scrapes reviews of any product from the Amazon website, applies sentiment analysis on the top reviews, and visualizes the results using tables, graphs, and word clouds. The results showed that the BERT approach outperforms the other deep learning approaches with an accuracy of 99%. However, other algorithms also reached promising accuracies of at least 85%.

Contents

List of Tables

List of Figures

List of Acronyms

NLP	Natural Language Processing
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
BERT	Bidirectional Encoder Representations from Transformers
NER	Named-entity recognition
NN	Neural Network
SVM	Support Vector Machine
GRU	Gated Recurrent Unit
CSV	Comma-separated values
URL	Uniform Resource Locator
UI/UX	User Interface/User Experience
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
Web Hook*	Also called web call-back or HTTP push API
WebApp	Web Application

Chapter 1

Introduction

In this chapter we introduce the readers to the motivation for this project and objectives to be achieved while its implementation.

1.1. Background

As social media growth continues, organizations and individuals use their content from media such as forum discussions, blogs, reviews, posting and commentaries on social media sites, micro - blogs, Twitter etc. to make decisions. The statistics for social media usage in 2021 is given below in Figure 1. In general, the overall contextual polarity or writer's feeling is determined through sentimental analysis. An Opinion is a personal judgement formed by a number of people about a particular thing that is not based on fact or knowledge. Opinion commonly refers to the subjective belief and the result of emotion or interpretation of facts that a person thinks about something. Opinion Mining (OM), also known as Sentiment Analysis, is a type of natural language processing used to determine public sentiment toward a product or topic.

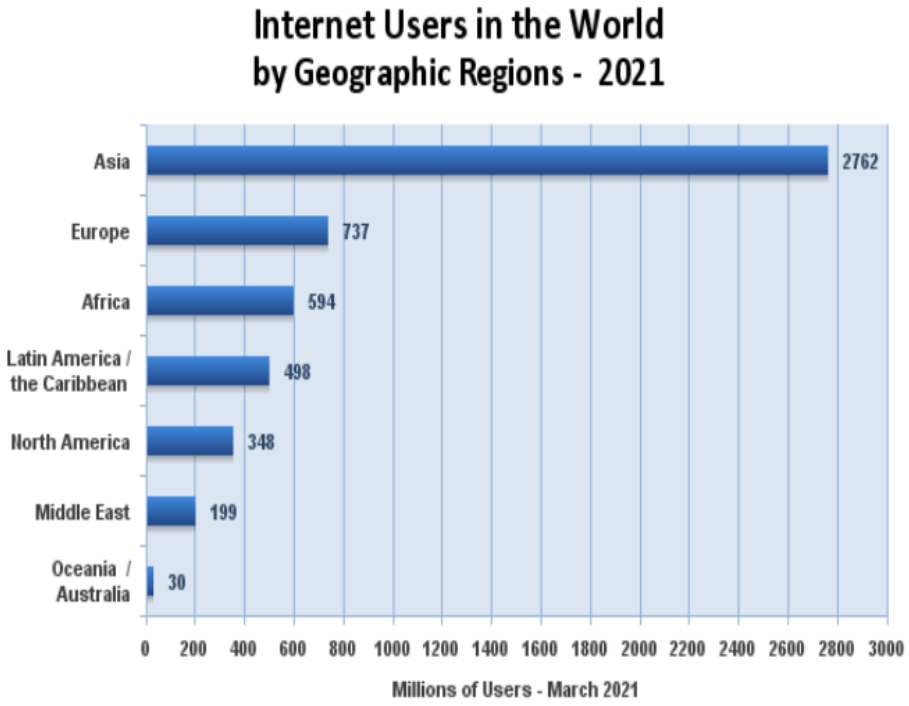

Figure 1. Social media usage statistics

The world that we see today is being digitized more and more. In this digitalised world of e-commerce, the customers don't have to leave their homes to buy their products. As people rely on online products nowadays, the importance of a review is increasing. In the business world, reviews for a specific product are processed, product characteristics (quality, features, etc.) are generated, and opinions are aggregated. It entails gathering and analysing opinions about services or products expressed in blog posts, reviews, tweets and comments. Sentiment analysis is useful for financial and business strategizing, such as judging the success of a new product launch, determining which service or product version is widely known, and identifying demographics like specific features. In the early 2000s, the scope of sentiment analysis study expanded. Author (Haque et al., 2018) In the field of sentiment analysis, different domains and different types of sentiment classification can be done,

which enable a finely grained classification of feelings through the focus on polarity ranging from negative to positive [11]. Below is an example of a customer feedback text and its possible sentiment.

Customer Feedback Text	Sentiment
"This café is great, the staff are really friendly and the coffee is delicious"	Positive
"I would not recommend this café to anyone. Their coffee is terrible and is really expensive"	Negative

Table 1. Example Review

Positive as well as negative reviews play a vital role and influence the company and its products. If the review is positive, it will eventually help other customers interested in such products to purchase from this company, so that these comments serve as an example to potential consumers. On the other hand, in cases of negative comments, they are helpful in enhancing the products and in identifying deficiencies or new needs required for the products of the company.

There is therefore the need to employ various text mining methods like Natural Language Processing (NLP) techniques, and also sentiment analysis and other machine learning techniques, in order to classify and analyse that large amount of text contained in reviews. The NLP techniques help to extract from a given dataset high quality patterns, trends and information. So there is a question: what is the NLP and why we need it?

Natural Language is a human's mother tongue, such as English, Spanish, and so on. Natural Language Processing (NLP) is an artificial intelligence branch which helps computers understand, interpret and manipulate human language. Many organisations use different applications for NLP, such as speech and word to speech conversion, machine translation, context extraction, sentimental analysis, etc . In the current case, NLP is a common domain.

There are many reasons why NLP is so needed in the world today, but there are only a few main reasons:

1. Large Volume of textual data

NLP enables computers to communicate with humans in their native language while also automating other language-related tasks. Every day, a large amount of Natural language data is generated in today's world via social media and online services such as e-commerce, etc. A human cannot comprehend such a large volume of unstructured data. When compared to a human, NLP can analyse more natural language-based data without tiring and in a consistent, unbiased manner.

2. Structuring a highly unstructured data source

As we all know, human language is extremely complex and diverse; not only are there hundreds of languages and dialects, but each language has its own set of grammar and syntax rules. NLP aids in the resolution of linguistic ambiguity and adds useful numeric structure to data. Even though NLP has many applications in this project we are going to focus on Sentiment Analysis.

1.2. Motivation

Being extremely interested in everything related to Machine Learning, the project provided me with the opportunity to learn and confirm my interest in this field. The ability to make estimates, predictions, and give machines the ability to learn on their own is both powerful and limitless in terms of application possibilities. Machine Learning can be used in finance, medicine, and almost any other field. That is why I decided to base my project on Machine Learning. This project was motivated by my desire to investigate the sentiment analysis field of machine learning because it allows me to approach natural language processing, which is a very hot topic right now.

Tweets can sometimes express opinions on a variety of topics. Many business decisions and even political sentiments about a candidate rely on these opinions.

- Before making a purchase, consumers can use sentiment analysis to research products or services. As an example, consider the Kindle.
- Marketers can use this to conduct public opinion research on their company and products, as well as to analyse customer satisfaction. Election polls, for example.
- Organizations can also use this to collect critical feedback on issues with newly released products. As an example, consider brand management (Nike, Adidas)

1.3. Problem Statement

This project aims to extract text features and analyse the sentiment of text as positive, negative, or neutral.

- Input: Textual content of a Amazon Review
- Output: Label signifying if the sentiment of the text is positive/negative/neutral

CHALLENGES: Some of the challenges faced in this project are mentioned below:

- Noisy text (Misspellings, lack of grammar)
- Lack of context
- Acronyms e.g.: lol, brb, gr8
- Emoticons e.g.: :), :(
- Negation
- Labelled data was not available.

1.4. Research Objective

The following are our study's objectives:

- Product reviews are scraped from websites that feature a variety of products. We have used amazon.in to scrape such data.
- Semi-supervised sentiment analysis was used to analyse and categorize review data.
- Categorization or classification of opinion sentiment into-
  - (+ve) Positive
  - (-ve) Negative
- BERT model was implemented on scrapped data and compared to two machine learning models, CNN and LSTM-RNN. The BERT model outperformed the other two models, yielding favourable results.
- We evaluated our model’s performance using various performance metrics such as accuracy, recall, precision, and F1-score.

1.5. Report Overview

A brief outline or guideline to the readers. The report is structured as follows:

Chapter 2 - Background, presents its readers with an overview of the background research on the topic, introduction to NLP and Sentiment Analysis, review of literature and state of the art existing solutions that have been considered in the development of this work.

Chapter 3 – Design & Methodology, provides its readers with an outline of the methodology pursued by the author in designing three different neural network architectures, discovering the requirements, aspects considering its user interface and experience.

Chapter 4 - Implementation, provides technical insights into the development of the web application. The author describes key features, research, and documentation undertaken in the implementation.

Chapter 5 - Evaluation, provides an analysis on evaluating the training and validation accuracy of all the models which have been used in this project. Finally, we will take a look at the web application and understand how these pre-trained models are being used. A brief reflection upon the requirements met in the project and the scalability of the implementation.

Chapter 6 – Discussion & Conclusion, provides an outline on the contribution of this project, commercial viability, experience & achievement through the project, with a highlight on possible future work to take it one step further. Finally a summary concluding the project report.

Chapter 2

Background & Literature Review

This chapter provides the necessary background information for understanding the research conducted in this project. This project is about sentiment analysis, which is defined as textual mining that identifies and extracts sentiments and subjective information from text based on its context, assisting businesses in understanding the social sentiment of their brand, product, or service while monitoring online discussions. Figure 1 depicts the sentimental approach's hierarchy. The project consists of four sections that deal with the use of NLP, machine learning and deep learning techniques and the relevant terminology used in sentiment analysis and then examines the literature review and lists the approaches that are used to develop the project.

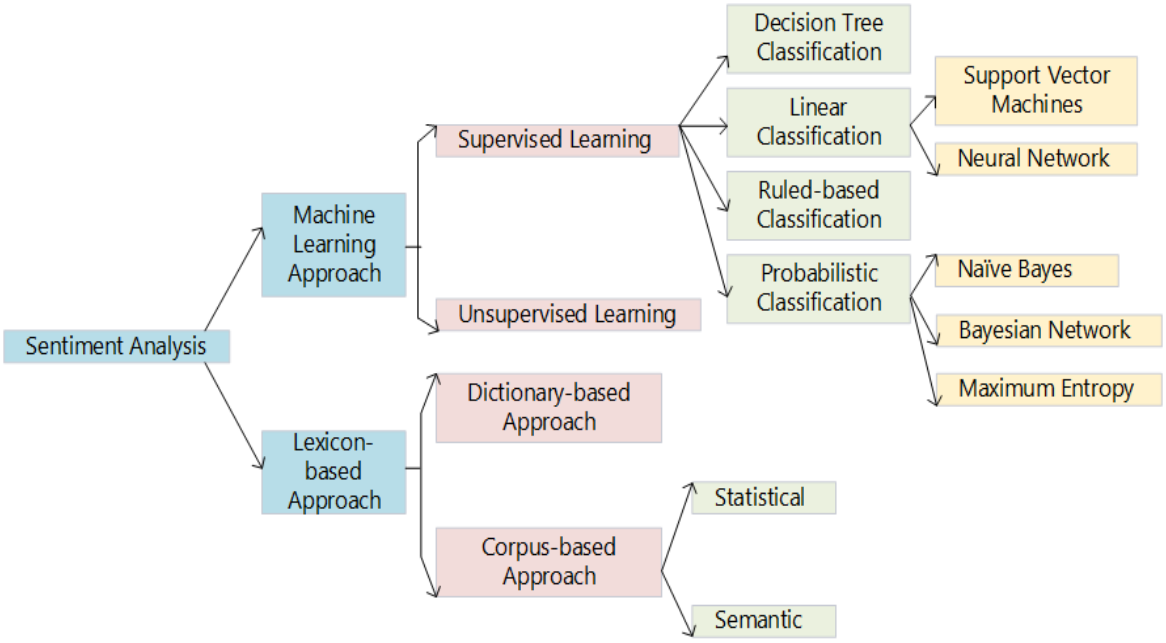

Figure 2. Hierarchy of Sentimental Approach

2.1. Understanding various techniques used in Sentiment analysis

Sentiment analysis is a process of analysis of online pieces of written text to determine whether they have a positive, negative or neutral tone. It is also known as Opinion mining or Emotion AI. In simple terms, the analysis of feelings helps to determine the attitude of the author on a subject which is shown below using an example:

Suppose there is a company which sells gadgets such as phones, laptops, tablets, video games etc. They have created a website to sell their products and customers can now order any item from their website and can also provide feedback, whether they liked or hated the product.

User Review 1: I love this iPhone and its battery backup is good.	Positive review
User Review 2: This laptop has a very low built quality.	Negative review
User Review 3: I ordered this tablet today.	Neutral

Table 2. Example of Sentiment analysis classification.

2.1.1. NLP Techniques

As humans, we communicate with one another using Natural Language, which is simple to understand but becomes much more complicated as we dig deeper. There are billions of people, each with their own way of communicating, which means that a lot of tiny variations and sentiments are added to the language, which is easy for us to interpret but becomes a challenge for machines. This is why we need Natural Language Processing, a process that allows computers to understand Natural Language in the same way that humans do (NLP).

1) Named Entity Recognition (NER)

This is one of the most widely used Semantic analysis techniques. The text conveys semantic information. This technique's algorithm takes a paragraph or phrase as input and identifies all names or nouns in that input. Daily use cases include news categorization, an efficient search engine, and customer support.

2) Tokenization

Tokenization is the process of dividing a text into a list of tokens, which can include sentences, words, characters, punctuation, numbers, and so on. Tokenization has two primary benefits. The first is that it significantly reduces search, and the second is that it makes efficient use of storage space.

3) Stemming and Lemmatization

The process of reducing derived words to the form of their word, basis or root — usually in writing. For instance, stemming essentially eliminates all suffixes. Therefore, once the word "playing" has been stemmed, it will be "play," "asked" will be "ask."

Lemmatization usually includes steps taken using vocabulary and morphological words to remove only influxive ends and return the word's base or dictionary form, known as the lemma. In short, lemmatization is a lemma of a word which means that, after the comprehension of the part of speech or the context in which the word is written in any document, the word form must be reduced.

4) Bag of Words

This method is used to pre-process text and extract all functionality from a text document that can be used in modelling machine learning. It also clarify the occurrence of words in a corpus in textual terms(document). This is called "bag," i.e. only if the words in the document are known and not where the words are located.

Sentence:	Bag of Words:
“Anand does not love Ram.”	Anand, does, not, Ram, and, love, games
“Anand love games.”	

Table 3. Bag of Words Example

2.1.2. Machine Learning Techniques

Two types of machine learning are available that are commonly used for sentiment analysis: unsupervised and supervised. Unsupervised learning does not have a category, and it does not provide the correct targets at all, resulting in clustering. Supervised learning is based on labelled datasets, which are fed into the model during the process. We have discussed briefly the three supervised techniques, namely naive Bayes, support vector machine, and maximum entropy.

1) Naïve Bayes

It was used both in training and classification phases because of its simplicity. It is a probabilistic classifier that can study a set of categorised documents according to a model. It compares the contents of the documents to a list of words to classify them. Tweet d is assigned the class c*, where f represents a feature and ni(d) represents the number of times feature fi was found in tweet d.

$$C^* = argmac_c P_{NB}(c|d)$$

$$P_{NB}(c|d) = \frac{(P(c)\sum_{i=1}^{m} P(f|c))n_{i(d)})}{P(d)}$$

There are m features in total. Maximum likelihood estimates are used to obtain the parameters $P(c)$ and $P\ (f|c)$,which are then smoothed by one. The pre-processed data, along with the extracted feature, is used to train the classifier using Naive Bayes. When the training is complete, during classification, it gives the polarity of feelings. The review "I'm happy," for instance, results in a positive polarity.

2) Maximum Entropy

The entropy defined on the conditional probability distribution is maximized by maximum entropy. It even handles the overlap feature and is similar to logistic regression in terms of determining distribution over classes. It also adheres to some feature exception constraints.

$$P_{ME}(c|d) = \frac{\exp[\sum_i f_i y_i(c,d)]}{\sum_c \exp[\sum_i f_i y_i(c,d)]}$$

Where c is the class, d is the tweet, and fi is the vector of weight. The weight vectors determine the importance of a feature in classification. The same processes are used to determine the polarity of sentiments as the naïve lines discussed above.

3) Support Vector Machine

The support vector machine analyses the data, defines the decision boundaries and computes in input space using kernels. The input data consists of two sets of m vectors each. Then, each piece of data represented as a vector is assigned to a specific class. The task now is to find a margin between two classes that is not near any document. The distance defines the classifier's margin; maximizing the margin reduces indecisive decisions.

2.1.3. Deep Learning Techniques

The exponential growth in the number of complex data sets every year means that machine learning processes need to be further advanced to ensure a robust and exact classification of information. Deep learning approaches have outperformed past machine learning algorithms on tasks such as image classification, the processing of natural language, face recognition, etc. Following are some of the popular neural networks:

Deep Neural Networks

Deep neural network architectures are designed to learn by connection from multiple layers where every layer is connected only to the previous layer, but only to the next layer in the hidden part. The input layer contains embedded vectors. For classification in several classes i.e.; multi-class problem, neurons in the output layer are equal to the number of classes, and only one neuron is needed for binary class.

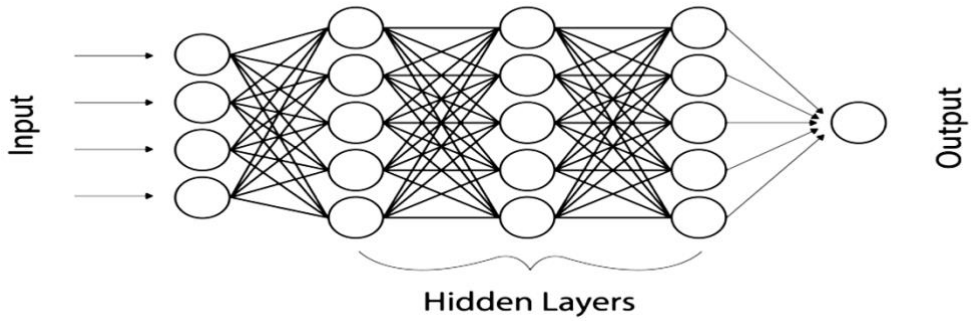

Figure 3. Deep Neural Networks.

1) Recurrent Neural Networks (RNN)

Researchers are also interested in recurrent neural networks (RNN) for text mining and classification. The previous data points in the sequence are given more weights by RNN. As a result, this technique is an effective method for categorizing text, sequential and string data. The neural net in RNN considers previous nodes information in a highly advanced way that allows the structures in the data set to be better analysed semantically.

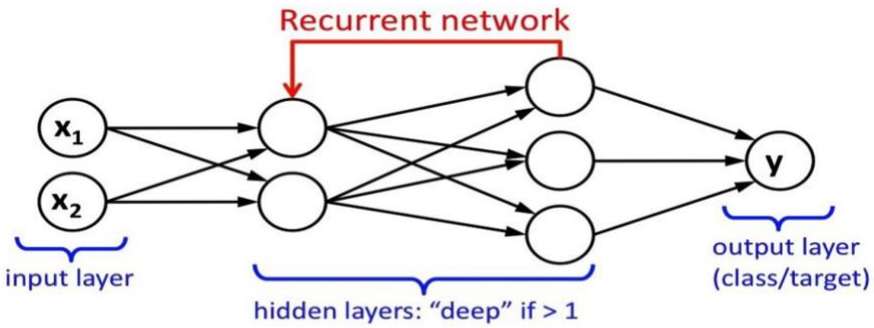

Figure 4. Recurrent Neural Networks

2) Gated Recurrent Unit (GRU)

J. Chung et al. and K. Cho et al. developed the Gated Recurrent Unit (GRU) as an RNN gating mechanism. GRU is a simplified version of the LSTM architecture, but it differs in a few ways: GRU has two gates and no internal memory (as illustrated in Figure; finally, a second non-linearity is not used i.e. tanh in Figure).

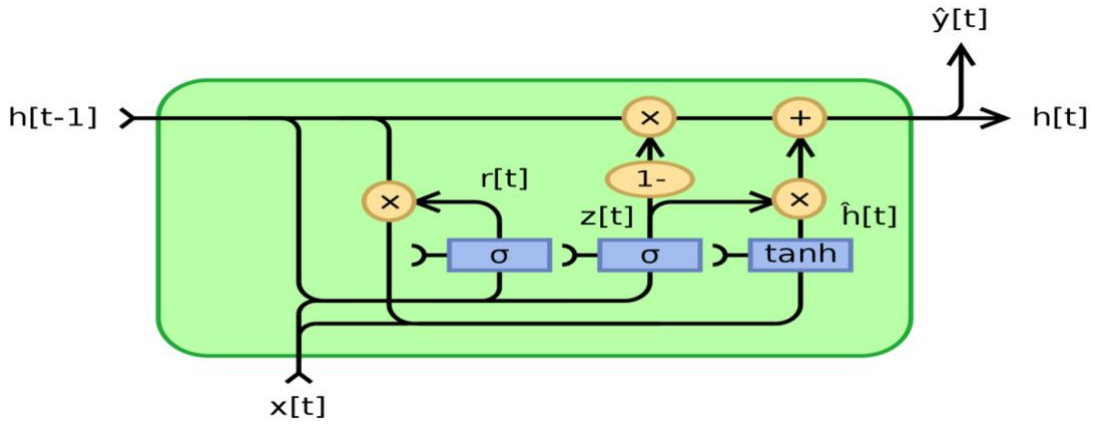

Figure 5. Gated Recurrent Unit.

3) Long Short-Term Memory (LSTM)

The concept of Long Short-Term Memory was pioneered by S. Hochreiter and J. Schmidhuber (LSTM). Long Short-Term Memory (LSTM) is a type of RNN that performs better than basic RNNs at preserving long-term dependency. Because LSTM employs multiple gates to carefully regulate the amount of information allowed into each node state, it is especially useful for overcoming the vanishing gradient problem. The figure depicts the basic cell of an LSTM model.

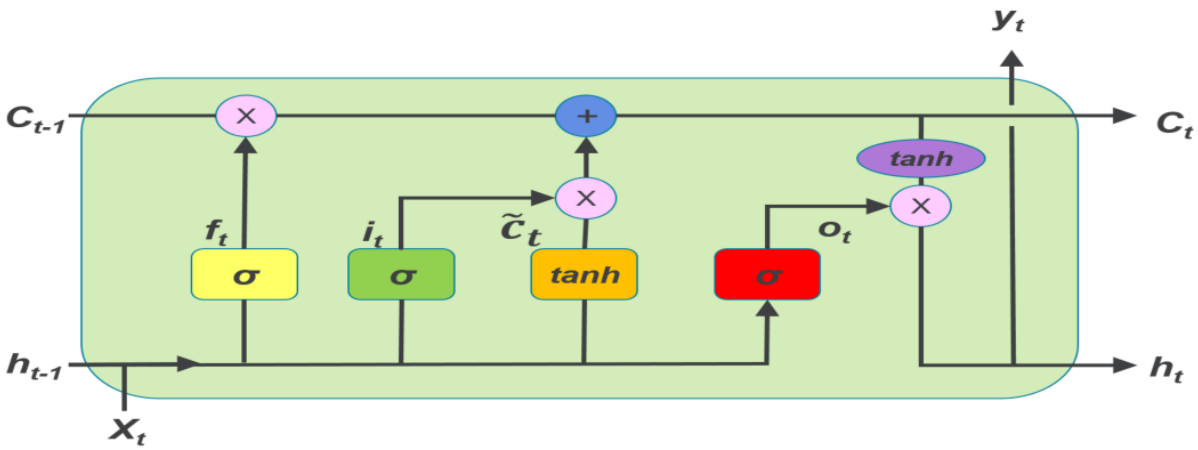

Figure 6. Long Short-Term Memory.

4) Convolution Neural Network (CNN)

A deep learning architecture for the hierarchical classification of documents is also CNN (Convolution neural network). CNNs originally intended for image processing with architecture similar to the visual cortex were also effective in text classification. An image tensor is convoluted with a set of kernels size d * d in a basic CNN for image processing. These convolution layers can be stacked to provide multiple input filters and are called feature maps. CNN uses pooling to reduce computer complexity by decreasing output in the network from one layer to the next. Diverse pooling techniques are used to minimise output while maintaining important features.

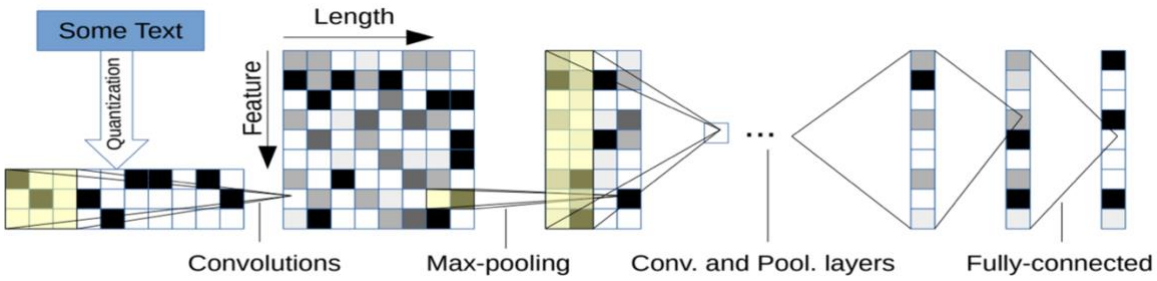

Figure 7. Convolution Neural Network.

2.2. Literature Review

This section includes an overview of the important literature, which contributes for deep research analysis conducted in the sentimental analysis project. The referenced material would assist researchers in understanding the strengths and weakness of the study.

Until now, many researchers have worked on the analysis of product reviews. **(Elli, Maria, and Yi-Fan, 2016) [1]** used sentiment analysis to develop a business model by extracting sentiments from reviews. They claimed that the proposed tools are accurate enough due to their robustness. They improved the accuracy of their decision by using business analytics. Other notable works include emotion detection, gender based on names, and fake review detection. Python and R are both widely used programming languages. Support Vector Machines and Binomial Naive Bayes were used [1].

(G.kaurand A.Singla et al., 2016) [2] used product user reviews comments about products and reviews about retailers from Flipkart(an Indian E-commerce website) as a dataset and labelled review text by objectivity or subjectivity and buyer's negative or positive attitude. Such reviews are profitable in some ways, promising for both customers and product manufacturers. This paper focuses on the semantic meaning-based classification of item reviews. They have analysed the fundamentals of opinion mining, the pros and cons of previous opinion mining systems, and provided some direction for future analysis work in the current study. The authors proposed entirely different approaches, such as spelling correction in review text to make the most sensible comment for knowing the encounter of words using Word Net dictionary, and then classifying comment using a hybrid algorithm combining Decision Trees and Naive Bayes algorithm [2].

(A.Tuzhilin et al., 2017) [3] proposed a recommendation method that not only predicts products of affection to the user, as traditional recommendation methods do, but also a specialized form of consuming the products to increase the user's knowledge of those items. For example, it may persuade the user to visit a specific location (item), such as a restaurant, and order certain foods, such as Italian (an aspect of utilization). The Sentiment Utility Logistic Model is the name of their model (SULM). Based on what the user has said about a specific form of the item, this model predicts the user's opinion about the item and then determines the critical form of the user's experience. They tested the proposed method on actual sentiment reviews from three real-world practical applications and demonstrated that their method performs well in these applications by preparing recommendations about the critical form that improve user wisdom [3].

(Fangand, X. J.Zhan et al., 2015) [4] proposed a model for sentiment polarity categorization, a difficult problem statement in opinion mining. They proposed a new algorithm for identifying negation phases and devised a new mathematical approach for calculating sentiment scores. For sentiment polarity

categorization, a feature vector generation method is proposed. Two experiments on sentiment polarity categorization were conducted; one at the review level and one at the sentence level, and the execution of three classification models was determined and compared based on their experimental results. They made use of online product review data gathered from Amazon. They carried out experiments for sentence-level and review level categorization, and the desired outcomes were obtained [4].

(Zhi Li et al., 2020) [5] creates a Danmaku sentiment dictionary as well as a new method for observing polarity in Danmaku reviews that incorporates a sentiment lexicon as well as Nave Bayes. The proposed approach has a significant impact on sentiment score and polarity detection, according to experimental results. The system can handle a large amount of data with consistent and fast performance, it speeds up training on a specified typical value, and it reduces the number of parameters to learn. The disadvantages are that data sets are too small, diagnostic information cannot be configured with ambiguous prior information, and label relation issues are ignored when three or more classes are present [5].

According to **(Li Wang et al., 2019) [6]**, SLCABG is a novel sentiment analysis model focused on the language of feelings that incorporates Convolutional Neural Networks (CNN) as well as attention-based Bidirectional Gated Recurrent Units (Bi-GRU). SLCABG incorporates the benefits of polarity, which benefits lexicons as well as deep learning methods. This paper demonstrated its efficacy in large amounts of highly distorted data, created a robust feature vector, and outperformed previous schemes in terms of recall, precision, and time complexity. The drawbacks are that samples can be unrelated or incorrect, resulting in unknown prior knowledge, that the original cross-entropy loss is limited, and that labels cannot fully determine samples [6].

(Sosa and P. M., 2017) [7] proposed the integration of two NNs (Neural networks) such as CNN (Convolution Neural Network")-LSTM (Long short term memory) and LSTM-CNN in the context of Twitter data sentiment analysis. 10,000 tweets consisting of positive and negative tweets were tested for training. The average accuracy of LSTM-CNN is approximately 75.2%. [7].

(Wehrmann et al., 2017) [8] illustrated the "Twitter Sentiment analysis" approach with CNN to use as a "language-agnostic translation-free" approach. The polarity of tweets, written in different languages, has been examined. The structure of the deep neural network has been analysed to require fewer student constraints. The classification algorithm was trained by the latent features. Four different languages, for example, Spanish, Portuguese, German, and English were tested. [8].

(Patil et al., 2015) [9] has developed a classification technique for the analysis of the sentiments by labelling positive or negative users' comments. Speech is engineered and modelled to classify the roles to explore the parts of speech. The voice model parts are an important indicator of the expression of feeling, which works on the detection of subjectivity that shows the close connection between adjective presence plus sentence subjectivity. The performance evaluation is done by finding precision and recall. The findings show that the SVM (Support Vector Machine), more than the artificial neural network (ANN) [9], worked on a text categorization basis.

Approaches	Classification	Advantages	Disadvantages
Machine Learning	Supervised/Unsupervised	-No need of Dictionary -High classification accuracy	Classifier once trained on the text of a specific domain cannot work for other domains
Rule-Based	Supervised/Unsupervised	-At review level 91% performance accuracy & at sentence level 86% -Sentence level classification is better than word level	Accuracy/Efficiency depends on defined rules
Lexicon-Based	Unsupervised	-Learning procedure and labelled data is not required	Powerful linguistic resources are required

Table 4. Weakness and strength in sentimental analysis.

2.3. Attention is all you need [10]

We used the relevant artefacts from the paper "Attention is all you need" (Vaswani et al., 2017) as the basic logic for implementing our hybrid sentimental analysis model [10]. This paper was a significant advancement in that it described the operation of the basic attention mechanism, serving as the primary improvement for a model known as the Transformer.

The initial proposals for sequence-to-sequence problems, such as neural machine translation, were based on the use of RNNs in encoder-decoder architecture. When working with long sequences, these architectures have a significant limitation in that their ability to retain information from the first elements is lost when new elements are incorporated into the sequence. The hidden state in each step of the encoder is associated with a specific word in the input sentence, usually one of the most recent. As a result, if the decoder only accesses the decoder's last hidden state, it will lose important information about the sequence's first elements. To address this limitation, a new concept, the attention mechanism, was introduced.

Attention is all you need" paper [10] proposes that:

"In this work we propose the Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The Transformer allows for significantly more parallelization. The Transformer is the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution".

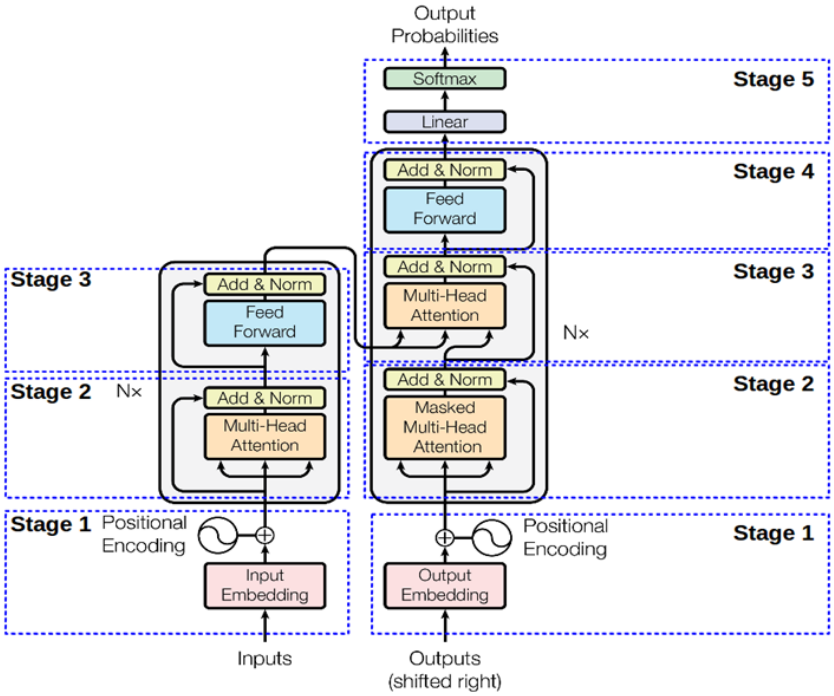

Figure 8. Encoder & Decoder representation of Transformer.

We can see an encoder model on the left and a decoder model on the right. Both have a core block of "an attention and a feed-forward network" that is repeated N times. Self-attention is a sequence-to-sequence operation, in which a sequence of vectors enters and a sequence of vectors exits. In the self-attention mechanism, each input vector is used in three different ways: the Query, the Key, and the Value. These three matrices are commonly referred to as K, Q, and V, and they are three learnable weight layers that are applied to the same encoded input. As a result, because each of these three matrices is derived from the same input vector, we can use the attention mechanism of the input vector itself, a "self-attention" mechanism.

Positional Encoding

We briefly mentioned that the order of the words in the sentence is a problem to solve in this model because the network and the self-attention mechanism are permutations invariant. We get the same solutions if we shuffle the words in the input sentence. We must create a representation of the word's position in the sentence and add it to the word embedding.

$$PE_{(pos,2i)} = \sin\left(pos/1000^{2i/d_{model}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(pos/1000^{2i/d_{model}}\right)$$

To that end, "positional encodings" are added to the input embeddings at the encoder and decoder stack bottoms. The positional encodings and embeddings have the same dimension, so the two can be added. There are numerous positional encodings to choose from.

2.4. Research Gaps

After studying existing research literature in the sentimental analysis research field, the following points are highlighted as inferences drawn and verdict from the above state-of-art as illustrated below:

- The data used in the previous work are standard and the researchers have not worked on user defined data. So, all existing works are non-real-time models for sentiment analysis.
- In the existing work, due to unpredictable phrases and sentence meaning changes, the pre-processing of data is affected, thereby the variation in feature of data is more and the chances of classification accuracy reduction is also high.
- The existing pre-processing cannot provide normalized data and the chances of an irrelevant feature set are more due to the appearance of un-normalized data, data with punctuation and stop words.
- To remove the above problem, several optimization algorithms have been used but the researchers have not obtained high accuracy to optimize the text data.

We used a hybrid classification approach in our work to overcome previous shortcomings and achieve better results.

Chapter 3

Design & Methodology

In this chapter we discuss the methodology and design process undertaken in the development of the project. Also, addressing the understandings from SotA which helped in realizing the correct set of requirements and further implementation. A foundation is laid for the readers in further understanding of the technical implementation.

3.1. Design

As we have discussed so far neural networks or the inception of Artificial Intelligence to the profound idea of classifying or regressifying any given sequence of data is what we are looking at. Our problem is to classify exactly how the sentiment could be related to the given set of words that the user prefers to use, so in our data set we are using an Amazon-based recommended system with multiple label data into three categories – positive, negative, and neutral. First, we must comprehend the operation of neural network architecture.

a. Neurons and the layers
b. Activation function
c. Optimizer
d. Back propagation algorithm

3.1.1. Neurons

A neural network is nothing more than a segmented form of various neurons. Each layer consists of a different number of neurons, and the final layer pertains to a number of desired outputs. In our case, there are two options: positive, negative, and neutral. After doing a deep dive into neural networks, we realize there are four fundamental ideas with neural networks.

3.1.2. Activation layer

The genesis and architecture of a neural network are represented by the activation layer. It not only determines how the neural network performs at each step and epoch, but it also determines which features the neural network can extract into the kernel density function and the higher dimension matrix. This may seem perplexing at first, but once we give our neural network a set of integers, we can give it a set of matrices. Assume that 10 x 758 refers to 10 rows and 758 columns. The first neuron layer, which could have as many as 512 neurons, will then convert the given output into 10 x 512 dimensions. That means that all 758 dimensions have been extracted to 512 dimensions, and the subsequent layer will convert to 256, then 128 and so on until we reach the dimension of 3.

Now we must figure out what is going on in this inter-cohesion of one dimension to another. Our features are extracted into high-level conflicting boundaries as you move from one dimension to another. These opposing boundaries are simply a distinction between our three perceived classes, which are neutral, negative, and positive. To draw these classes, a model may require 10, 20, or even 2 layers of architecture, and the neurons and layers are both dynamic hyper-parameters that the modifier or creator of the neural network must consider, work on, and finally implement. Since it is quite clear what we are looking at, let us proceed to the extinction of the idea that is the activation function. An activation function is a straight forward concept. It is rather enigmatic as to how an input transformation can occur. As in our previous example, let us try to focus again, so we have an input of 10 x 768 with 10 rows and 768 feature columns, but we have 512 neural networks with an integral

function. The function first takes 768 columns from each neural network and maps them to a corresponding output, which can range from – infinity to + infinity. That appears to be correct, but there is an underlying issue here. So, if we have 512 outputs, we can have many outputs with values of 0, 1, 2, 4, 5, and a few outputs with values of 100k and 10000k.

As a result, the significance of the values becomes much greater than that of their counterparts as 0, 1, and 2 ranges. To address this, we implement the activation function.If the output is negative, the negative value will correspond to a lower softmax, which will break the weights and embedding custom functions, resulting in a lower output value. Negative values cause a vanishing or diminishing gradient during back propagation. This diminishing or vanishing gradient is the cause of neural network stagnation. Stagnation can be defined as a neural network's inability to learn or unlearn. Finally, we will comprehend the activation function. As a result, the final result of the activation function determines its normalized and standardized output, which is passed from one layer to the next, removing the weight and biases of large numbers.

3.1.3. Optimizers

At each stage, we attempt to calculate the loss we incur as a result of learning in a neural network. A loss is always a positive number greater than zero and less than infinity. There are many different types of loss functions, such as categorical cross entropy loss, hinge loss, mean square error, mean absolute error, and so on. The full documentation for the losses function can be found at tensorflow.org. These losses, however, can only be calculated. We require an agent who can comprehend the impending loss and how we can precisely reduce or deduct the weight. This is a critical parameter to consider. As a result, we come up with the concept of optimizers. Optimizer is a rather unique agent that works on losses in such a way that it attempts to deduce why the losses are occurring and create an entire derivative. This entire derivative is forwarded from the beginning of the network to the end of the network in reverse cyclic order to reduce the weights of the function and the biases of the function that are related to the given loss and in each tiny step or as we can denote from here epoch. We try to deduce the loss and subtract it from weight and biases. So that’s how an optimizer works.

As it began with Stochastic Gradient Descent, it belongs to the optimizer family. Etal Hilton proposed the first optimizer in 1985. Following that, there are numerous versions of optimizers that are evolving and becoming available. Mini-batch Stochastic Gradient, Batch decent, Adagrad, Rmsprop, Adaboost, and Adam are a few examples (Adaptive Gradient Method). In our case, we will attempt to evaluate all of the different losses and prepare a comparative study to examine and confirm how Adam outperforms the other optimizers.

3.1.4. Back-Propagation

Back propagation is nothing more than the extension of the optimizer concept. The concept itself reflects on how we can deduce a way to increase or decrease a parameter from the weight and bias function in such a way that these parameters can produce a lower loss and thus a higher accuracy, which is indeed desirable. So, in this hyperdimensional plane where the model has transformed the data, the loss appears as categorical cross-entropy or cross-entropy where the loss is to be calculated, creating a derivative. That derivative in turn is transformed into two major rates:

1) The learning rate
2) The decaying rate

Learning rate: The learning rate determines how quickly the gradient moves from one lower minima to another lower minima in the graphical plane, where there may be many different lower minima and one global minima. To get to the global minima, we must first move from the lower minima to the global minima, or other minima in simpler terms.

Decaying rate: We don't know whether one is a global minima or not, there must be a decaying rate that, if we skip the global minima, will bring us back to it in such a way that the loss is minimized and the accuracy is optimized.

So these functions, the learning rate and the decaying rate are an essential and integral part of what you call the loss function or the back propagation function used by the Adam optimizer to optimize the neural network over multiple epochs.

3.2. Convolutional Neural Network (CNN)

As the name suggests, neural networks tend to utilize the idea of convolution and very well define the boundaries which will emit separable data. So it might be unique to understand why we are discussing an image of separable data into a field or geo-field of sentimental analysis. It is an experimental deduction of CNN into the sentimental analysis data in such a way that we can perceive exactly how CNN works for language data.

3.2.1. Background

Let us again understand biases by definition of an image pertaining to 28 x 28 x 3 which is your width, height and channels. When the initial limit is fed into 128 Kernel convolution layers pertaining to 3 x 3 filter and no pairing and image is formed into 56 x 56 x 128 channel images. So let us understand exactly what has happened. Initially, we had 3 channel images of width and height 128 but once we fed into convolution it has broken it to smaller images of 56 x 56 and a greater number of channels of dimensions of 128. That means it's again transforming the same image into multi-dimensional space to break it in such a way that we can see through the exact correlations and functions in it. Further on we should also understand exactly what Max-Pooling is.

3.2.2. Max-Pooling

Max-Pooling or polling is an ability to keep the most essential features in a given set or locality of a matrix into the next dimension. It’s like choosing the best pixel out of all. So, suppose we have a matrix of 2 x 2 with 10, 11, 14 and 17 values. Of these, in a general perception, one could not identify exactly which pixel is better than which and of course, it’s not a comparable idea but when we reach around the feature spaces into the inter-dimensional correlation of features, we generally try to understand the importance of feature could be directly linked to its direct value which is 17 in our case. But we can also go for minimum and average pooling which are also good options but not widely used. It is a known fact to know that max pooling was deduced first after which people recognized the idea of average and minimum pooling but they were eradicated shortly because the shortcomings of both the methods prevailed over the benefits of them.

So once you have understood the max pooling we should understand what a kernel space means.

3.2.3. Kernel Space

A kernel-space is a locality or region of area where the higher dimension features are mapped. These spaces not only are unique in the separation of boundaries of features but are also very well declarable in a complex space or a z-space as we call it. The kernel space majorly pertains to the utility of a kernel. As one might be aware, SVM's or traditional machine learning ideas of using kernels are of fixed ideas. However, deep learning has completely changed the idea of fixed kernels to dynamic and regressive kernels. These regressive kernels pertain to a conclusive idea of creating a kernel extremely out of nowhere and try to fit the kernel into the data in such a way you can extract every point into the given 3 or 4 classes you have. So this kernel space in our previous example was 3 x 3 which means a 3 x 3 window will slide over our volumetric data 128 x 128 x 3 in such a way it can not only collapse into the exact feature dimension spaces it wants but it can also convert the kernels into kernel space in such unique and refined way that you can exactly distinguish between the kernel, the space and the feature made for all feature dimension available to us. It might seem a bit emulative but it is the real emphasis of how a kernel space works and why a kernel space works. That summarizes our utility for Convolution Neural Networks. In the upcoming sections, we will revoke and understand how Recurrent Neural Networks are used.

3.3. Recurrent Neural Network (RNN)

Recurrent neural networks developed after the issue's growth of the convolution neural network provides a rather innovative idea in summarisation or locality transmission in textual data sets. Recurrent neural networks are not only unique in their behaviour of creating a global attention summary by which the Corpus of data can be conceptualised as a huge potential vector transmitting through space whose feature dimensions are extracted at every node such that the global summary is for changed in HT and the local summary is transmitted to the next node.

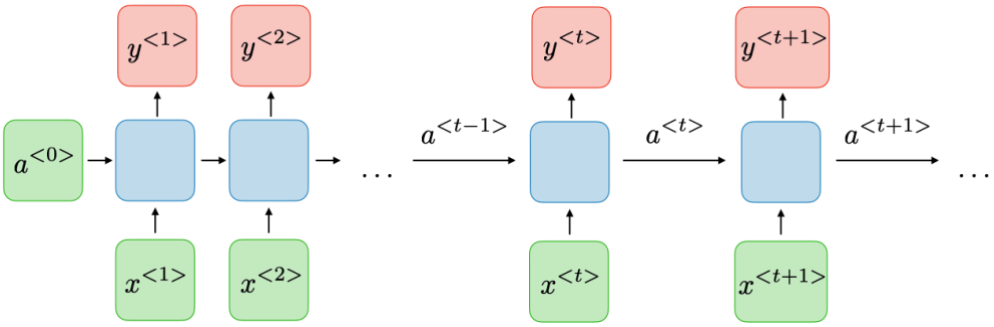

Figure 9. RNN working Diagram.

Recurrent networks work is first discussed in context with attention and global summarisation.

3.3.1. Feature Space Abstraction

Inundated above the utility of RNN creates a summary or potentially fetches the meaning from the data. Recurrence networks because of this multidimensional use case were not only very popular in slicing this feature space which could not only retain time information through frames and references but also create a globular Corpus which on further iterations could create historical evidence of summary through time. However, RNN has its drawbacks and problems: the recurrent network has a great use case about maximum sentences of 7 or 8 words that are also used on a huge Corpus like the Wikipedia data to sentiment analyse it. Recurrence networks are enabled to create local peripheral

features of attention that could attend to a group of some notion text then the global notion of the context. However, they are pretty strong in understanding the global context and are extremely useful in dealing with sentiment analysis of various comment tweets and other data.

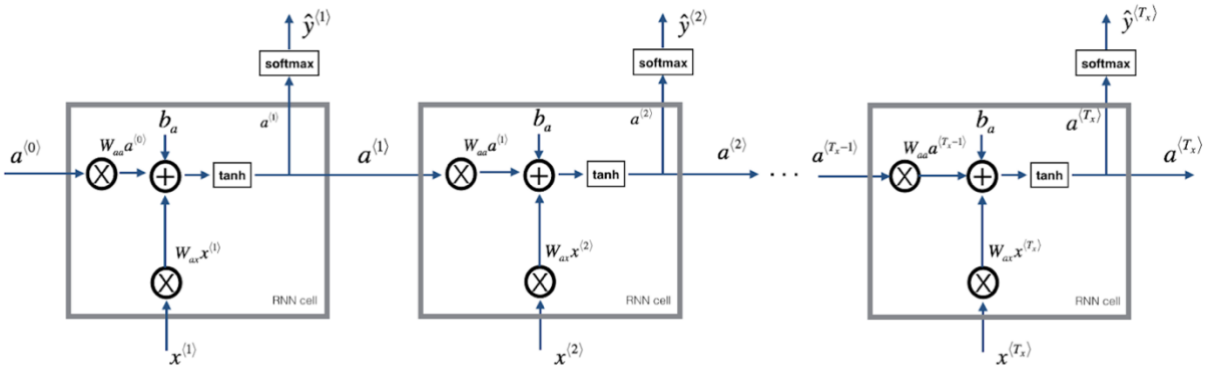

Figure 10. RNN Network.

3.3.2. Comparison between CNN and Attention RNN

As we have discussed so far recurrent networks is a improved version over the convolutional neural networks, which could not only fetch data from the higher dimensional feature kernel spaces of image and temporal and spatial analytics but could have a notion of the conception of time or flowing sequence reality in it. however with the forthcomings of dimension in the same problem could be addressed and elaborated in the sense of creating a global temporal attention throughout the Corpus of the network. Due to the availability of local attention and processing power during the years of 2004 to 2008, the utility of RNN was at its peak and it was used widely by big tech forms like Facebook Yahoo Google and many others. Today all the utility of recurrent networks have been much reduced because of the forthcoming of Transformers and by LSTMs. But their utility as a subsidiary global attention mechanism is still a huge benefit when added and used with Transformers like BERT, Robert and this disbert.

Recently a study by Google analytics showcases the utility of iron has become constant in the years of 2017-2021. However, it is only worth noting that the transformer Era started in the year 2017. We will further notice in the upcoming sections the utility of long short term memory networks and the transformer for the encoder-decoder networks. This does not only show how shallow in local attention context is our recurrent idea of neural networks but also shows how from understanding 8 to 10 words in a Corpus who have gone through understanding more than 500 to 700 words which is the current capacity of the biggest network available.

3.4. Long Short Term Memory (LSTM)

3.4.1. Background

Humans don't stop thinking and start acting only based on the current situation. That's the ability the nature and the neural architecture of human brain has provided with. However, the ability to think and to save data, or the same experiences could not be granted to the human-designed formulated neural networks. This is because humans have the ability to create a long short time memory in brain that is visualised as a step of procedure, which follows in order to keep a certain amount of memory data prevalent over time with or with experience.

3.4.2. Changes from RNN to LSTM

Long short term memory networks are an advancement over the prevalent recurrent neural that are not only able to store data in a global basis or create a global attention mechanism into the data but also formulate a local sub structure classified notion of contextual text. Because of this nature, it is compatible in adapting the situation where the subject differentiates from the original on the global discussion topic validate into a various form of discussion and then again comes back to the contextual global topic of discussion this is very general and happens very regularly in human conversations one example of utilising the long short term memory could be encountered with the Google assistant of 2012 to 2015 during this period Google have utilised over 0.35 billion data set is to utilise into the long short term memory. Due to this Google could not only be able to validate through different sub structuring of streams of data of human to human and human to machine translations but also create a huge Corpus for its upcoming malls of transformers, another architecture to be used.

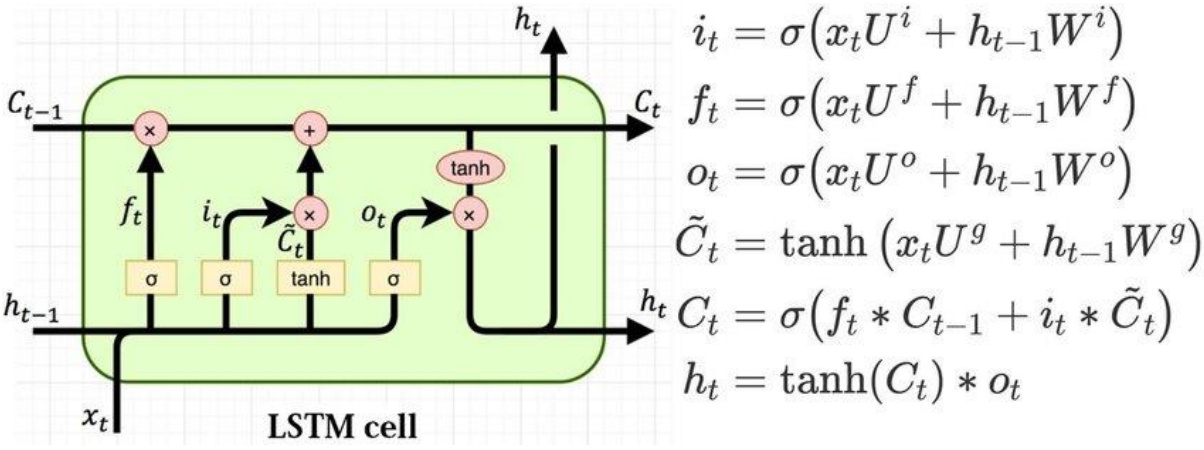

Figure 11. LSTM working Diagram.

3.4.3. LSTM Architecture

The architecture of long short term memory networks: The network utilizes a global attention feature vector Ht which demonstrates the historical data at time period t and another feature vector VT which demonstrates the local visual attention at time period t. Both these variables are filled into every neuron of the layer along with the input creating a two-dimensional feature space vector as an input from where we could achieve a linear transformation of attribute flow where there is a high correlation between local contextual data. On a simpler basis we could understand this for example if we're talking over a sentence which has a reference to the initial subtext of a boy and is utilising the context of the very end of the sentence then the numerical value of attention assigned to the boy would be very close a closely correlated with a numeric attention assigned to the preferential part of the boy for example he him or they. This was earlier with the recurrent network setup was not only not possible but rather edge case left behind, direction of flow in the current networks did not keep any local attention prevalent for this global local attention and thus the feature vector as the output of the network could not demonstrate relations in local context but only had the global context of the sub context in its output does when fitted with long sentences these were not only useless but also did not get the global attention which was lost into various sub local context in the sentence.

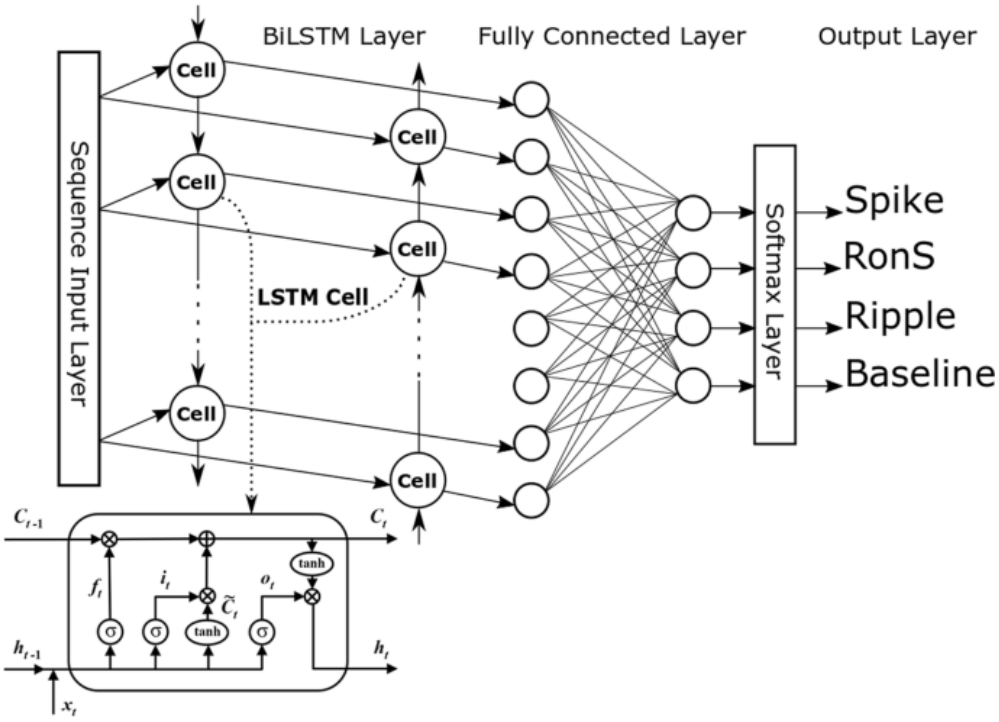

Figure 12. LSTM - RNN Network

Although it seems that the long short term memory networks are pretty good in analysing and understanding long sequences of but their ability to observe and construct local attention is limited to one or two sub contextual statements. Which means for a long sentence let's assume 100 word sentence which can have 325 local sub contextual attentions the long short term memory network would be incapable of understanding all the local divergence in the statement. That's why in the next section we would be dealing with the utility of transformer.

3.5. Transformer

3.5.1. Introduction to Transformer

Transformer architecture is a rather notable development into the field of deep learning and artificial intelligence. The change in the context of artificial intelligence came about understanding the minimise and maximize regarding the attention mechanism was pretty simple that to grass each and every local sab contextual text texture which was prevalent in the Corpus but absent in noticing by the previous architectures. After close understanding and dedicated efforts in 2017 the paper attention is all you need to come up with the idea of Transformers. A transformer is an encoder decoder architecture where the encoder consists of 6 layers and the decor on the other hand consists of another 6 layers. The notion is to create a feature Matrix which not only did use the features of local but global attention and then decode the same in order to translate a given sentence Corpus into another given sentence Corpus. Considering such an exam is important to understand the inner workings of a transformer before we can dive into architecture like Bert, Robert and disbert.

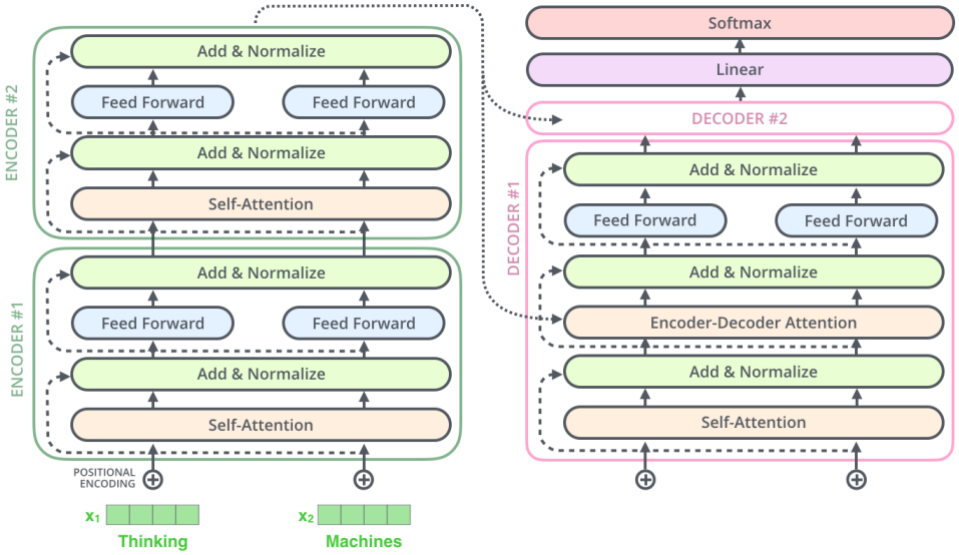

Figure 13. BERT Architecture.

3.5.2. Working of Transformer

When the initial statement for example is fed into the network it is first created a word embedding using the prevalent word2vec, doc2vec or other architecture using this we are now able to reduce our statement into a more relatable contextually map towards or widely known as word embeddings. Once these map embeddings are formulated the one dimensional vector is fed into the transformer encoder version the first encoder version consists of a multi head attention. Attention mechanism deals in creating three neural networks. Initially it is fed into the query where a single vector or a single word relation as per the word embeddings is given as a query which is then SoftMax averaged out over different weights and biases who is composite texture is again SoftMax with the value feeling into another neural flattening layer which outputs a 1D vector now this vector is again fed into a simple artificial neural network in order to maintain the spatial and temporal dimensions of the data. It's important to note here that the attention mechanism does not have any positional features or embeddings associated with it. That's why before feeding the word embeddings we also need to create a sinusoidal positional distribution function closely related to the Fourier series to show that the data positioning is not lost.

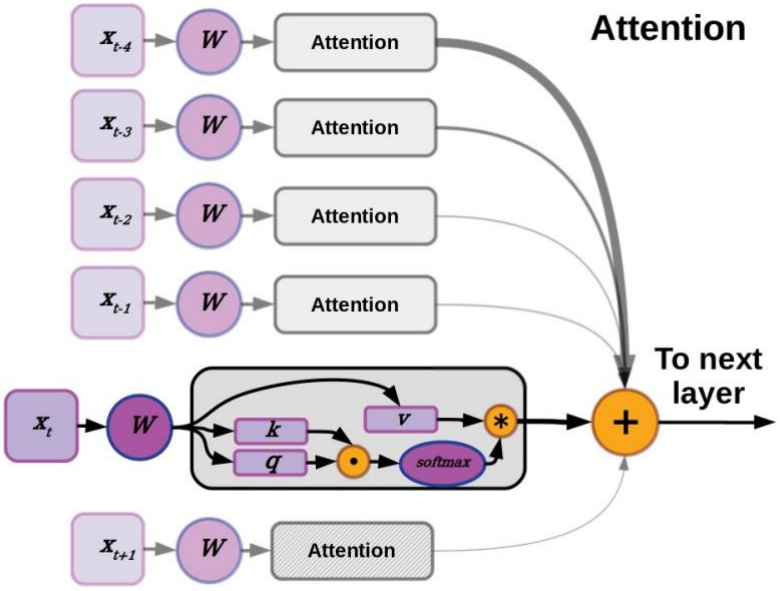

Figure 14. Representation of Attention Mechanism.

3.5.3. Decoder and beyond

After the encoding of the data in through the six layers of feature matrix of global attention is now created. Now let us understand exactly how statement is decoded in the transformer once the encoding is done we feed in the start word to the decoder along with the temporal output of a global attention to the model now according to this we also feed in all the 6 temporal and coding outputs which were initially feed it to the next encoder but now is also fitted into the various segments of the coder namely D1, D2, D3, D4, D5 and D6. These data are now SoftMax and the best perceiving output is located as the next word once we have stimulated the next word in the series it goes as an input to the decoder until and unless we reach the end of sentence or the EOS. Complete life cycle of how a transformer architect took the data processor through encoders and found the decoded into another corpus language. Indeed special technique has its own share of merit today in sentiment analysis and natural language processing text classification subjects the encoder model is an essential essence to decoding the feature vectors not only provides a higher dimensional global attention which is often necessary for classification tasks but also has different biases and weights of each and every local attention. Using a CNN for an overall neighbourhood Kernal global attention we can use a transformer which not only fetches the global attention with the likes of RNN networks and long short term memory networks but also with the local attention models such that this statement output is a rather distribution function of the local sentiments of the world. It can be stated that the utility of a comment involving both critic and benefits of a product are evaluated separately by assigning different weights to each and every critic and compliment and finally did you see whether or not the comment can be classified as a negative neutral or positive comment.

Since we have discussed so thoroughly in Transformers we will try to cover Bert, Robert and disbert. Along with the implementation pipeline of each of the models, also try to elaborate tokenization padding and other basic stop words context in relation to natural language processing in upcoming sections.

3.6. Final Design and Architecture

As we have elaborated above the utilisation of various neural networks starting from the artificial neural networks proceeding to visualising images through kernel spaces in convolutional neural networks and then understanding the summarisation of a Corpus of data into a variant space by recurrent neural networks has its own utilities and design in building the long short term memory networks and the Transformers of today. However in this project we try to demonstrate the utilities of transformer architectures such as BERT in classifying positive and negative sentiment comments from the Amazon review system. Explanation of reducing the data from the Amazon review system would be covered in the upcoming implementation section but as of now we'll try to elaborate the exact procedure of extracting features from the given data set of the Amazon comment sentiment system.

First of all sentiment analysis is a procedure by which we try to determine whether a sentence impacts a given product in a positive or a negative manner. It is a brilliant use case in understanding whether or not the customer likes or not and exactly what patches determine him into this like a given project. The utility of sentiment analysis has been done for decades to understand the mood of the market and release products according to a given sentiment or given mood of the buyer. Today the classification of sentiment analysis is at a very different level from analysing tweets for political and social logical benefits to understanding the ability of a product to encroach the market, sentiment analysis is being used. Since we are already elaborated on the utility of sentiment analysis and the process of sentiment analysing the data, we now celebrate the utility of the path model and the underlying concept of tokenization, padding, pickle filing and other steps.

The diagram below depicts the flow of the remaining steps in our project:

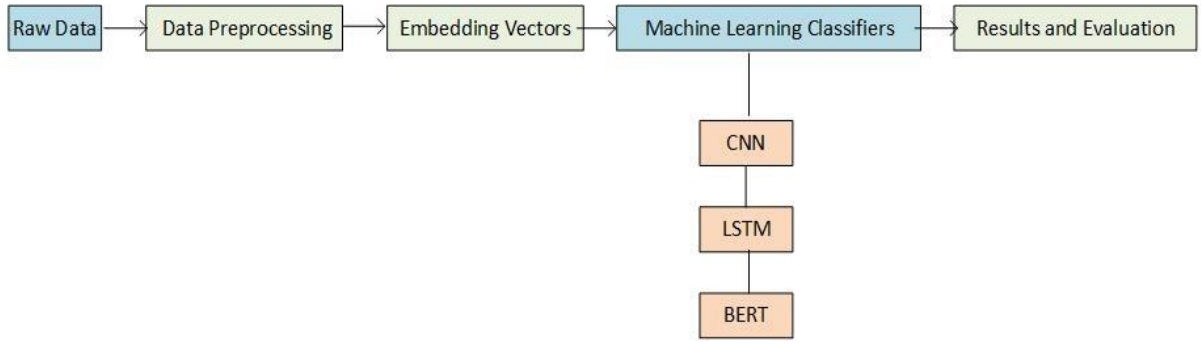

Figure 15. Flow chart of Model Execution

3.7. Dataset used and Data pre-processing

3.7.1. Data Description

The data as is given to us is in a form of csv file, it consists of four attributes:

- Customer
- Customer ID
- Customer comments and
- Ratings that are associated with the comments.

Based on these components, we can analyse and tell that any rating above three can be classified as a positive sentiment and any rating below three can be classified as a negative sentence by the lambda function of the python pandas library. We now have a binary data set with only two levels 0 for positive sentiment and one for negative sentiment; however, a machine cannot understand different variance and length of words. Furthermore, the sentence does not refer to any hex code for the given machine.

As a result, we must convert these words into understandable mathematical numpy language, so that the machine can not only reduce the meaning between the words, but also create an equivalent statement for each input chaining to give an output, which we can then feed into our model.

3.7.2. Data Pre-Processing

Tokenization is the process of converting human language into a mathematical transcript using a hash function. Padding is the process of converting a mathematical transcript into an equally spaced and equal length transcript. In our case, we tokenize 5497 different words and use dollar characters as padding. The final length of the data's padded numerical structure is approximately 74 different characters and 212 in length.

3.7.3. Embedding Vectors

We also need to create attention masks for each of our data points in order to create a valid input for a transformer. To do so, we create a two-dimensional Matrix for n words in a sentence. Where the Matrix contains n - I zeros and rest ones for each sequence length I. This zero value denotes the absence of a specific token, whereas the one value denotes the presence of a specific token. This is done to prevent cheating from the transformer architecture in such a way that it is unable to look ahead in time and predict the country that this aids in utilizing our current work in order to predict the future as well as finally did using the feature matrix to predict the classification.

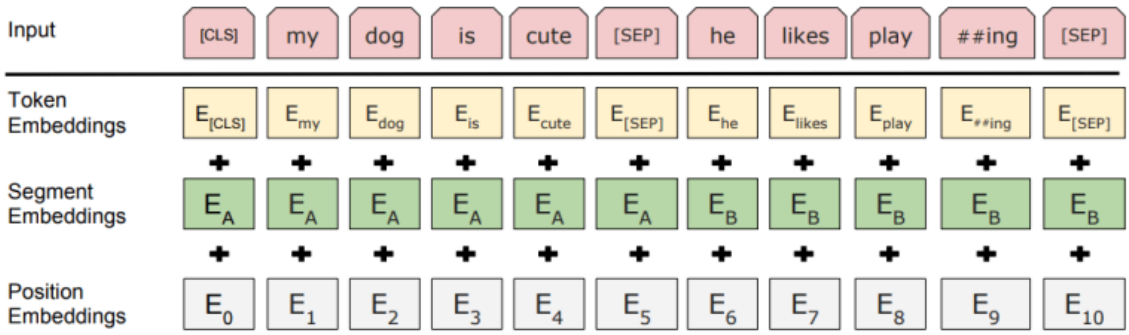

Figure 16. Data Pre-Processing .

Following the generation of embedding vectors, the following steps are taken:

1. After completing the steps we then move forth to create a model.
2. Model Contains an embedding layer which takes all the attention and the input feature matrix from the data and returns 786 vector.
3. This vector is feature extracted and has close correlation between similar words.
4. After this layer the output is passed to various dense layers and then finally passing it to a dense layer pertaining to two outputs
5. The loss function utilised in this case is categorical cross entropy with optimizer Adam and activation function relu.

After building the model we try to create a stable visualisation of all the different parameters used in the model and then create a layer diagram in order to showcase the passing of different layer outputs from one to another.

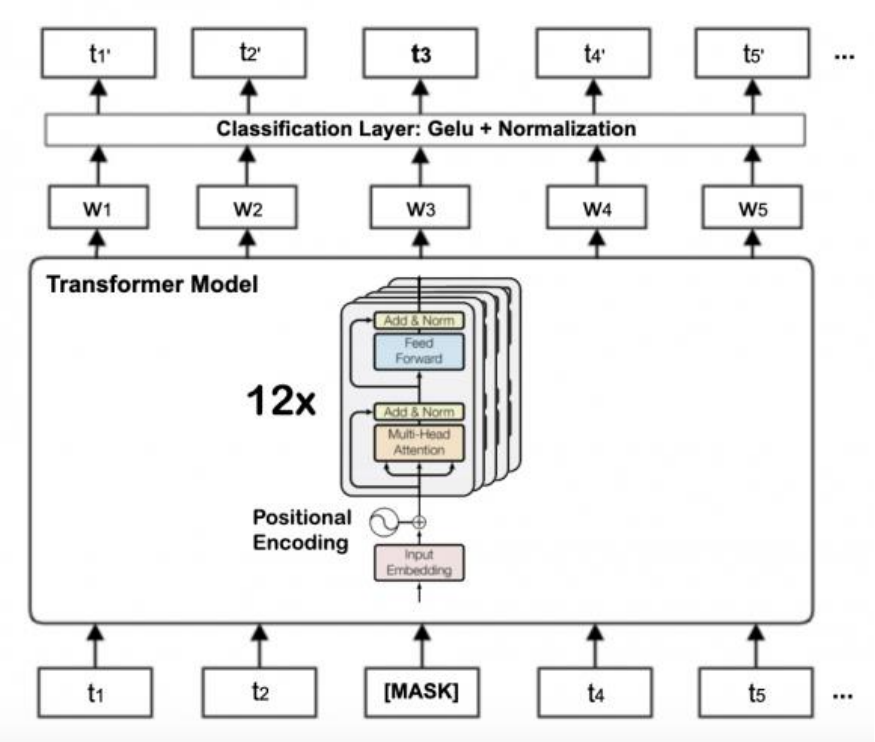

Figure 17. Full Transformer Architecture.

Before beginning the training testing and evaluation we divide the data into 80% training, 20% testing and 20% validation sets. The validation set is taken out from training randomly at every iteration the model tries to recreate the feature Matrix which it has not seen priorly. After each epoch during the backpropagation the validation set as well as a training set is used to create a loss error this loss is then propagated through the model and into the batch model in order to retrain the birth model as well as between the newly modified word embedding layer of the dense and the other artificial neural networks. Take a 64 input batch size for training our network in order to create a global gradient descent in the model.

Before training we try to take a few precautions in order of failure resistive model training. Thus save all the files, masks and labels as pickle and also create a call-back to save the best model.

```
Preparing the pickle file.....
Pickle files saved as  dbert_inp.pkl dbert_mask.pkl dbert_label.pkl
```

```
Loading the saved pickle files..
Input shape (89981, 34) Attention mask shape (89981, 34) Input label shape (89981,)
```

```
Train inp shape (71984, 34) Val input shape (17997, 34)
Train label shape (71984,) Val label shape (17997,)
Train attention mask shape (71984, 34) Val attention mask shape (17997, 34)
```

Figure 18. Pickle Files.

3.8. Machine Learning Classifiers used in Sentiment Analysis

As previously stated, the BERT model is used in conjunction with the pickle file and other standard training procedures. Let us examine the utility of models based on convolutional neural networks and recurrent neural networks.

3.8.1. Convolution Neural Network (CNN)

The convolution neural network model is based on the concept of convolutional networks. First, the data set Corpus is passed through the word to vector embedding Corpus of the tensor flow, from which we can obtain a two-dimensional output of word embeddings. This word embedding is now processed by a one-dimensional convolution network. As previously stated, the filter generally attempts to capture the most significant pixel of a given window in a 1D array or a 1D tensor, rather the pixels are taken in a sequential length by length window. Following this, the output is also a one-dimensional tensor, but it has been convoluted by the evolution via back propagation operation. This convolution is held together by a series of max-pooling layers along the way. We use three different convolution neural network segments, then pass the output to a flatten or deconvolution layer, and finally prove the similar dense column as described in the BERT model. Finally, we consider the loss function as a categorical cross entropy activation function as a relu and SoftMax, Adam's optimizer. The model's layer architecture is described below.

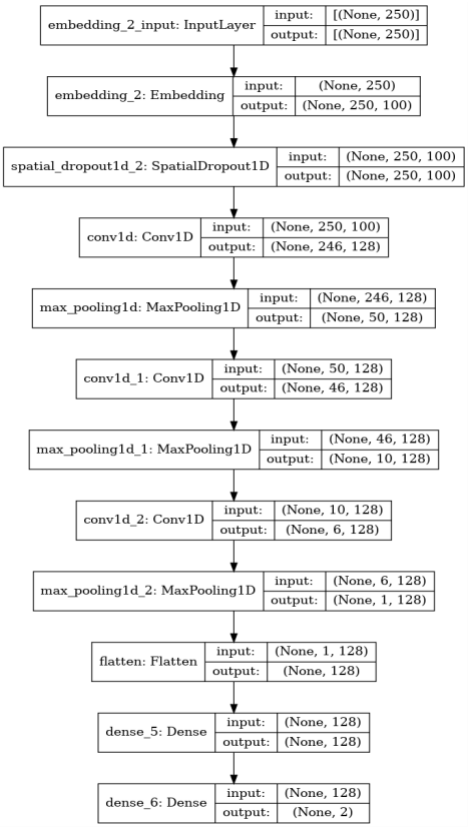

Figure 19. CNN Model Architecture.

3.8.2. Long Short Term Memory (LSTM)

For recurrent neural networks, we follow a similar procedure: we pass the initial tensor through the word embedding vector, and then from the word meaning activate, we pass it to the LSTM or long short term memory, and finally to the artificial neural network. The layer architecture of the model is also described below.

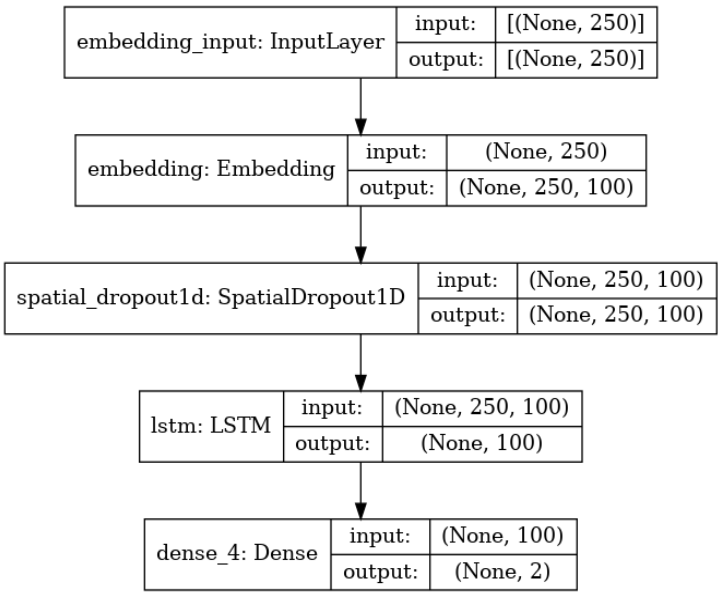

Figure 20. LSTM Architecture.

3.8.3. Bidirectional Encoder Representations from Transformers (BERT)

The transformer architecture comes in various utilities and designs. Bert architecture was one of the noble ideas of using bi-directional Transformers. We utilise this transfer learning technique in our implementation of the transformer. As in the diagram given below, we feed the scraped data which goes through two different operations namely the architectural composition of 786 feature vectors, and slicing lambda function. The slicing lambda function however creates the two-dimensional output into a single of flatten array output of the attention maps which it is fed to this sentence and finally as explain with the other models then output of two is taken from the model.

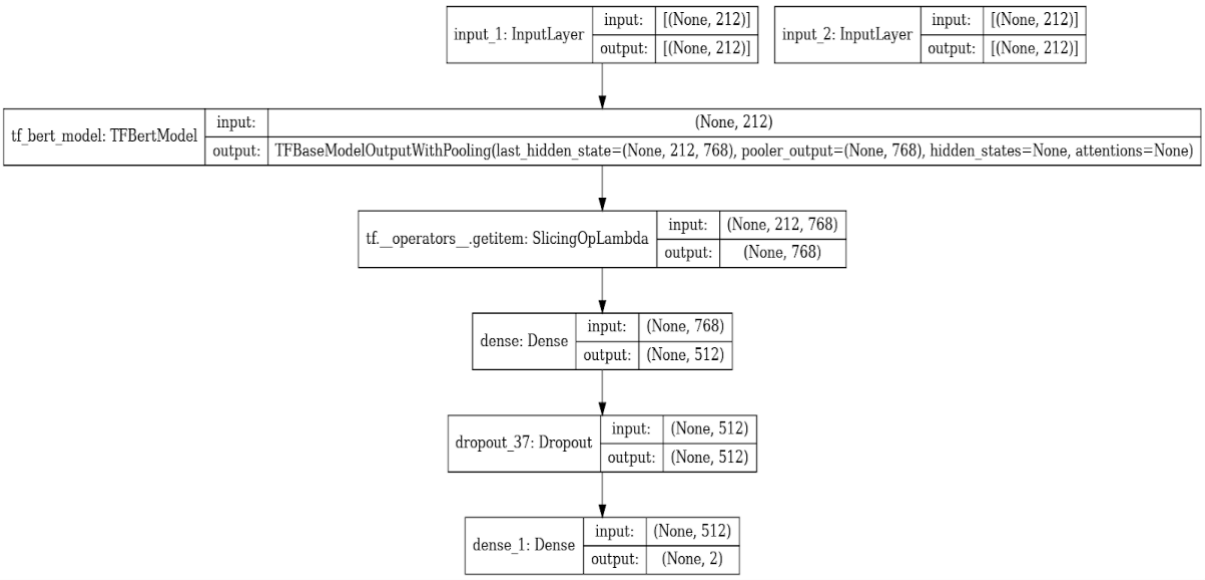

Figure 21. BERT Architecture.

We have clearly described the design of using different models and architectures for sentiment analysis. We will now proceed to the implementation or coding aspect of creating a web application to demonstrate the same. In the sections, we see the exact procedure for implementing web scraping code and node.js server architecture with python TensorFlow and keras. Discuss the findings, conclusions, and discussion, as well as a brief future note on the technology. Let's get right into the Implementation section

Chapter 4

Implementation

As previously explained in the design and literature review sections, we will now proceed to explain the model's implementation in this chapter. We are creating a web application for the Implementation of the trained model. The key features of the web app are as follows:

- A dashboard which takes amazon product reviews URL as an input.
- A scraper which helps in collecting the reviews of that specific product from the URL in a structured format.
- Predict the sentiment (+ve or -ve) for the reviews using pre-trained RNN, CNN & BERT model.
- Visualisation of the final results using tables, bar graphs and word cloud.

Let us first get a firm grasp on our model's critical flow charts and operation. As previously stated, we save a data.csv file in our local stack memory. After creating the csv file, it is loaded into an evaluation.py file, which evaluates the recurrent neural network, convolutional neural network, and BERT. First, we describe the sequence length and other hyper parameters, which will be discussed further below, before loading the model and obtaining the output for the given data from the csv file. Once the data has been processed and the changes have been made, the file is saved as dataop.csv. This file is then used to create a positive to negative bar plots and word cloud for each of the different positive and negative words of the data. The below figure illustrates the training and the flow cycle of our implementation.

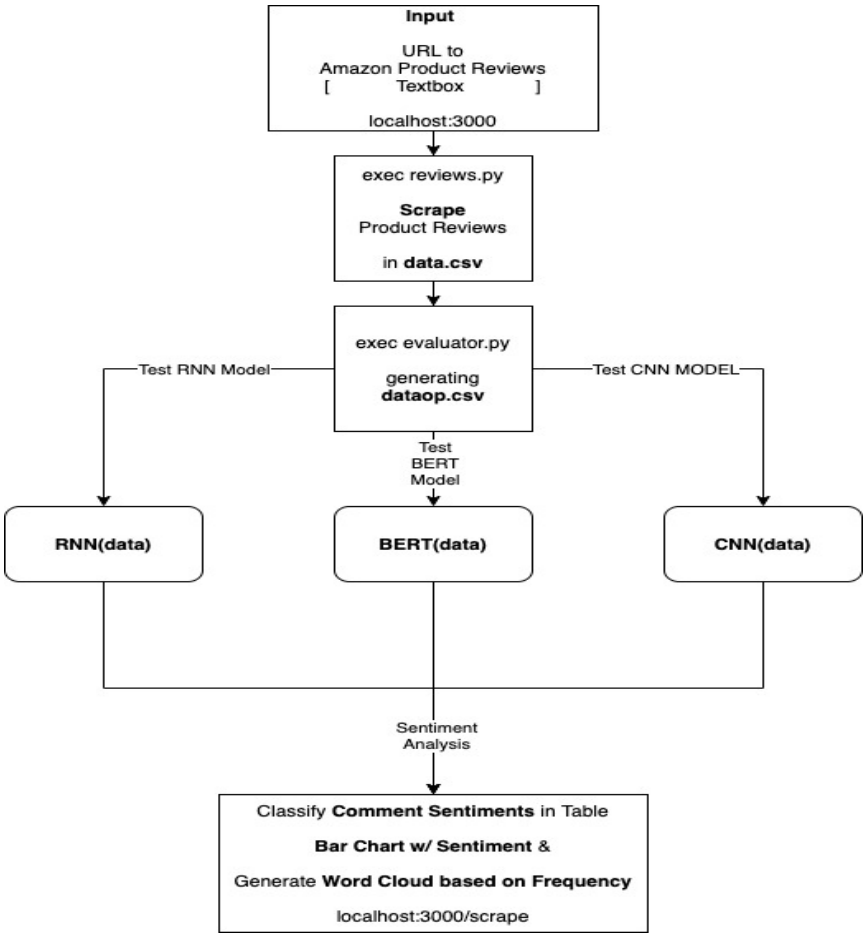

Figure 22. Flow chart of Web Application.

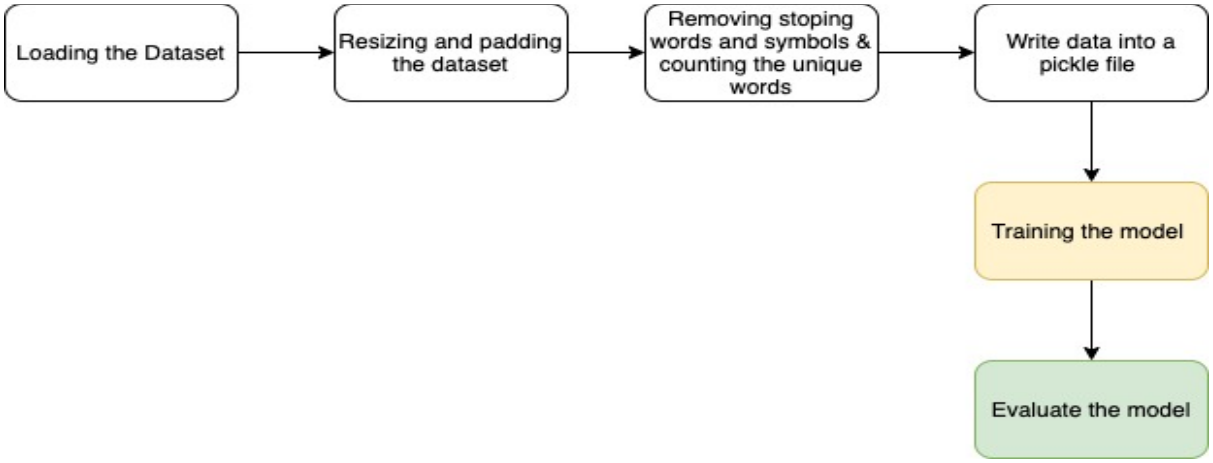

Figure 23. Flow chart of Training and Evaluation.

4.1. Scraping the data from Amazon's product reviews

In this section we will explain how we scrape the data, creating the HTML parser format to extract the data from it and finally storing into a data.csv file. This data scraping process makes use of a function that describes the characterization of HTML protocol files and is established with the Amazon API port which attempts to pass the data from the given website automatically through a testing passer.

4.1.1. Importing Libraries

For the following implementation of this section we are importing some libraries. The libraire names are mentioned in the picture below.

```
from selectorlib import Extractor
import requests
import json
from time import sleep
import csv
from dateutil import parser as dateparser
from string import Template
```

Figure 24. Import Libraries.

4.1.2. Establishing Connection

Once we start scraping the first step is to understand the port number from where the data has to be fetched. Pillow code illustrates the idea of getting the port number, creating a yaml file and saving the information, retrieving the data from the given port number. To create a dashboard to view the results of the models, first a web scraper is made, the scrapper parser through the url.txt file which has all the URLs listed for which the reviews will be scrapped. Then the headers of the network request are written and the appropriate selectors are written which go through the rendered DOM and parse the reviews and save the necessary information in a csv file.

```
# Create an Extractor by reading from the YAML file
e = Extractor.from_yaml_file('selectors.yml')

def scrape(url):
    headers = {
        'authority': 'www.amazon.com',
        'pragma': 'no-cache',
        'cache-control': 'no-cache',
        'dnt': '1',
        'upgrade-insecure-requests': '1',
        'user-agent': 'Mozilla/5.0 (X11; CrOS x86_64 8172.45.0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.64 Safari/537.36',
        'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9',
        'sec-fetch-site': 'none',
        'sec-fetch-mode': 'navigate',
        'sec-fetch-dest': 'document',
        'accept-language': 'en-GB,en-US;q=0.9,en;q=0.8',
    }
```

Figure 25. Defining and establishing the port.

4.1.3. Downloading the Relevant Details from the Page

Once we are able to reduce the port number within use information to understand how many rows are present in the reviews web page of Amazon. Once that accomplished we try to hinder the processing of the HTML blogs in the page. Doing so we also get the structure of this collection of the webpage such that we can retrieve the data in tabular format and save it for future references.

```
    # Download the page using requests
    print("Downloading %s"%url)
    r = requests.get(url, headers=headers)
    # Simple check to check if page was blocked (Usually 503)
    if r.status_code > 500:
        if "To discuss automated access to Amazon data please contact" in r.text:
            print("Page %s was blocked by Amazon. Please try using better proxies\n"%url)
        else:
            print("Page %s must have been blocked by Amazon as the status code was %d"%(url,r.status_code))
        return None
    # Pass the HTML of the page and create
    return e.extract(r.text)
```

Figure 26. Downloading the page.

Once we have recreated the data into a csv or tabular format, we can save it to data.csv file for further use and use it to get the outputs from our model using Panda's library.

```
# product_data = []
with open("urls.txt",'r',newline='') as urllist, open('data.csv','w') as outfile:
    writer = csv.DictWriter(outfile, fieldnames=["title","content","date","variant","images","verified","author","rating","product","url"],quoting=csv.QUOTE_ALL)
    writer.writeheader()
    for url in urllist.readlines():
            for x in range(1,4):
                try:
                    print(x)
                    t = Template('$url&pageNumber=$x')
                    turl = t.substitute({'x' : x,'url':url})
                    data = scrape(turl)
                    if data:
                        for r in data['reviews']:
                            r["product"] = data["product_title"]
                            r['url'] = url
                            if 'verified' in r:
                                if 'Verified Purchase' in r['verified']:
                                    r['verified'] = 'Yes'
                                else:
                                    r['verified'] = 'Yes'
                            r['rating'] = r['rating'].split(' out of')[0]
                            date_posted = r['date'].split('on ')[-1]
                            if r['images']:
                                r['images'] = "\n".join(r['images'])
                            r['date'] = dateparser.parse(date_posted).strftime('%d %b %Y')
                            writer.writerow(r)
                        # sleep(5)
                except:
                    continue
    print('Done')
```

Figure 27. Creating CSV File and Saving It for further use.

Once the data.csv file is created and stored in the local stack it can be viewed as in the below image.

Final Scraped Data:

title	content	date	variant	images	verified	author	rating	product	url
Stay Away	Update after 1 m	22 Feb 2021	Colour: Reverb R	https://images-n. https://images-n.	Yes	Ankit	1	OnePlus Bullets V	https://www.ama
No Quick Switch	It's mentioned in	20 Oct 2020	Colour: Reverb R	https://images-n. https://images-n.	Yes	Arun Veer	1	OnePlus Bullets V	https://www.ama
Great for calls an	So coming from	05 Nov 2020	Colour: Reverb R		Yes	satya	1	OnePlus Bullets V	https://www.ama
Not for me (too r	After viewing toc	14 Dec 2020	Colour: Reverb R		Yes	Prajwal G	5	OnePlus Bullets V	https://www.ama
Best Wireless Ne	Just received my	16 Apr 2021	Colour: Reverb R	https://images-n. https://images-n. https://images-n.	Yes	Gopal Singh Ran	5	OnePlus Bullets V	https://www.ama
Average bullet , r	After seeing all th	02 Nov 2020	Colour: Reverb R	https://images-n. https://images-n.	Yes	Roshan hp	3	OnePlus Bullets V	https://www.ama
The Perfect VFM	Bought this for M	26 Nov 2020	Colour: Reverb R		Yes	Rajiv P Rai	4	OnePlus Bullets V	https://www.ama
Best purchase bu	**Important** rea	04 Jan 2021	Colour: Reverb R	https://images-n. https://images-n.	Yes	Vishal	5	OnePlus Bullets V	https://www.ama
Not great suppor	The earphones th	28 Jan 2021	Colour: Reverb R		Yes	Dora	1	OnePlus Bullets V	https://www.ama
This is different fi	Cons: Few featur	12 May 2021	Colour: Reverb R		Yes	Amazing Custom	4	OnePlus Bullets V	https://www.ama
Awesome, Better	Review from an a	28 Oct 2020	Colour: Reverb R		Yes	Tarun Soni	5	OnePlus Bullets V	https://www.ama
Quality is not at p	Oneplus has beco	24 Dec 2020	Colour: Reverb R		Yes	Sachin S.	1	OnePlus Bullets V	https://www.ama

Figure 28. Structure of the Scraped Data.

4.1.4. Evaluation Python Script

Defining the constraints of the dataset. Then removing the Stop words. In this step we try to first map the dictionary consisting of all the different stop words once the stop was identified with the right to remove the source stop from the data.csv file. It is also interesting to note that the encoding format of our data.csv file is cp1252. This encoding format provides much more security in handling various comments and authenticity for our review. This is also a very important gesture for modelling any AI or data driven technologies today.

```
MAX_NB_WORDS = 50000
MAX_SEQUENCE_LENGTH = 250
EMBEDDING_DIM = 100
def data_creation():
  with open('token.json') as json_file:
      data = json.load(json_file)
  df = pd.read_csv('data.csv',sep=",", encoding='cp1252')
  X = []
  for i in df.index:
    S = ''
    for j in df.iloc[i,1].split(' '):
      if j in data:
        S += j+' '
    X.append(S)
  return X,df
```

Figure 29. Starter Code and word length

4.1.5. Modelling Function (RNN, CNN , BERT Output generator)

We are using BERT, RNN and CNN output generator. The utility of recurrent neural networks and convolutional neural network models can be seen in the image below. In both cases, we use it to tokenize a function of text to sequence modelling and then pad the given sequences of power input. Once the input has been converted into a feedable format, the model is loaded into the trained model variable. We allot a maximum value to arguments by allocating negative or positive remarks to an output. The API call returns the prediction variable as an output.

The flow of the diagram is consistent across all the RNN, CNN and BERT models. All of the models begin by taking the input as a padded sequence. After loading the model, tokenize it with the prestored dictionary. Predict results using the saved weights and architecture of the model. Then convert it into negative and positive comments and return the array list.

```
def RNN_output(X):
    with open('tokenizer.pickle', 'rb') as handle:
      tokenizer = pickle.load(handle)
    X = tokenizer.texts_to_sequences(X)
    X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
    trained_model = load_model('RNN_mod.h5')
    preds = trained_model.predict(X,batch_size=16)
    pred_labels = preds.argmax(axis=1)
    Prediction = []
    for i in pred_labels:
      if i == 0:
        Prediction.append('Negative Comment')
      else:
        Prediction.append('Positive Comment')
    return Prediction
def CNN_output(X):
    with open('tokenizer.pickle', 'rb') as handle:
      tokenizer = pickle.load(handle)
    X = tokenizer.texts_to_sequences(X)
    X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
    trained_model = load_model('cnn_mod.h5')
    preds = trained_model.predict(X,batch_size=16)
    pred_labels = preds.argmax(axis=1)
    Prediction = []
    for i in pred_labels:
      if i == 0:
        Prediction.append('Negative Comment')
      else:
        Prediction.append('Positive Comment')
    return Prediction
```

Figure 30. Creating Output for RNN and CNN generator

```
def Bert_output(X):
    with open('tokenizer.pickle', 'rb') as handle:
      tokenizer = pickle.load(handle)
    X = tokenizer.texts_to_sequences(X)
    X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
    trained_model = load_model('Bert_mod.h5')
    preds = trained_model.predict(X,batch_size=16)
    pred_labels = preds.argmax(axis=1)
    Prediction = []
    for i in pred_labels:
      if i == 0:
        Prediction.append('Negative Comment')
      else:
        Prediction.append('Positive Comment')
    return Prediction
```

Figure 31. Bert Model Output generator.

4.1.6. Prediction Function

As we can see in the below image we are trying to use a Predictor function which takes in our input variable the predicted variable and the name of the model. First we declare a positive and a negative array list and then it rate over the number of output variables. Then we create a word cloud for the positive and negative sentences from the given direction. Pyplot diagram of the bar plot and the word cloud received from the word cloud function described below.

```
def WC(X,Y):
  from wordcloud import WordCloud, STOPWORDS
  import matplotlib.pyplot as plt
  import pandas as pd
  comment_words = ''
  stopwords = set(STOPWORDS)
  for val in X:
      val = str(val)
      tokens = val.split()
      for i in range(len(tokens)):
          tokens[i] = tokens[i].lower()
      comment_words += " ".join(tokens)+" "
  wordcloud = WordCloud(width = 800, height = 800,
                  background_color ='white',
                  stopwords = stopwords,
                  min_font_size = 10).generate(comment_words)
  return wordcloud
```

```
def predictor(X,Y,name='Default'):
  df['Predictions'+name] = Y
  df.to_csv('dataop.csv')
  pos = []
  neg = []
  for i in range(len(Y)):
    if Y[i] == 'Positive Comment':
      pos.append(X[i])
    else :
      neg.append(X[i])
  p = WC(pos,'Positive Sentiment Words')
  n  = WC(neg,'Negative Sentiment Words')
  plt.figure(figsize = (40, 30), facecolor = None)
  plt.rcParams.update({'font.size': 22})
  plt.subplot(2, 2, 1)
  plt.bar(['Positive','Negative'], [len(pos),len(neg)],color=['green','blue'])
  plt.rc('xtick', labelsize=45)
  plt.rc('ytick', labelsize=60)
  plt.title('Bar Chart ',fontsize = 30)
  plt.subplot(2, 2, 3)
  plt.imshow(p)
  plt.axis("off")
  plt.title('Positive Sentiment Words',fontsize = 30)
  plt.subplot(2, 2, 4)
  plt.imshow(n)
  plt.axis("off")
  plt.title('Negative Sentiment Words',fontsize = 30)
  plt.savefig("public/assets/"+name+".png")
```

Figure 32. Prediction and output of web app.

After running all the plotting diagrams, we save the images into the model name PNG files. Finally we save the dataop.csv file for displaying on the web hook server of the node JS.

author	rating	product	url	Predictions RNN	Predictions CNN	Predictions BER
Ankit	1	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme
Arun Veer	1	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme
satya	1	OnePlus Bullets V	https://www.ama	Positive Commer	Positive Commer	Positive Commer
Dora	1	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme
Sachin S.	1	OnePlus Bullets V	https://www.ama	Positive Commer	Positive Commer	Positive Commer
Nag Deep	2	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme
Saumya	2	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme
Amazon Custom	2	OnePlus Bullets V	https://www.ama	Positive Commer	Positive Commer	Positive Commer
Reuben Robert	2	OnePlus Bullets V	https://www.ama	Positive Commer	Negative Comme	Positive Commer
Roshan hp	3	OnePlus Bullets V	https://www.ama	Negative Comme	Negative Comme	Negative Comme

Figure 33. Output from Evaluation

4.2. Full - Stack for the Dashboard of Web Application

The following section demonstrates the use of Node, and Express JS for the development of the application dashboard. When we hit the end point the view rendered has a text-input box where the user can enter the URL of any amazon product reviews that needs to be scrapped. Once we click on the submit button a post request is sent to the Node server which receives the URL that needs to be scrapped and writes that URL to the urls.txt file which is later used by the web-scraping function.

```
app.set("view engine", "ejs");
app.use(express.static(__dirname + "/public"));
app.use(bodyParser.urlencoded({extended: true}));
app.use(bodyParser.json());

app.get("/",function(req,res){
    res.render("index",{url:'',flag:false})
});

app.post("/scrape",function(req,res) {
    console.log(req.body);
    fs.writeFileSync('urls.txt',req.body.url);
    const python = spawn('python', ['reviews.py']);
    python.on('close', (code) =>
    {
        console.log("Scrapping Completed");
        const python1 = spawn('python',['evaluator.py'])
        python1.on('close',(ecode) => {
            console.log("Files Generated");
            const input = fs.readFileSync('dataop.csv');
            const records = parse(input, {
            columns: true,
            skip_empty_lines: true,
            to_line:7,
            });
            res.render("index",{url:req.body.url,flag:true,records});
        });
    });
});

app.listen(3000,function() {
    console.log('Server is running');
});
```

Figure 34. Node Server.

Once that is done, we spawn a child python process which starts the scrapping function and an event listener is attached to that which waits for the child process to terminate. On the close event of that python process the scraped data is available in the data.csv file which now can be used by the models for prediction and so we spawn a second child python process to which uses the data.csv file for predictions and all the word cloud and charts are saved in file system.

Node.js is a very common and widely used server creation library. Only the utility of the node JS is quite efficient and fast in creating the server, but its dynamic ability to run programs in different languages using spawn is another added advantage. In the below function we could see that we try to restrict the function call with the link of the data. Data is taken from the given utility link, once the link is fetched we spawn into our reviewz.py program in python and extract the scraping from there.

In the second python child process the pickled files of the models are loaded which has the already trained models and they are loaded into the memory and we do the predictions and save the results in a second csv file dataop.csv which has the predictions using all the three models all this while the node server waits for the response. On the close event of this process the node server parses the newly created dataop.csv file and parses that file to show few of the predictions on the dashboard and the view is re-rendered with all the results. The view loads the images saved in the file system to show in the dashboard.

As we have seen in which section you're not only implementing three different architectures from transformer, recurrent neural network and convolutional neural network. But also implemented a webhook web application using node.js server and an effective dashboard to showcase the reasoning for different sentiments using the word cloud and the bar chart.

In the next section we would be looking at the results of the specific architectures, their training and testing accuracies and finally the look of our web application made in node.js.

Chapter 5

Evaluations

In this chapter, we will understand and evaluate the different research objectives of this dissertation. The chapter is divided into sections that examine various model architectures before moving on to web architectural design UI.

The first section of the evaluation discusses the understanding and demonstration of the functionality implemented to extract data of various sentimental comments from any Amazon Shopping website, the HTML layout of the code, and the exact execution procedure. Following that, we discussed how to use different word clouds and data representations of the data set obtained through scripting.

Finally, we analyse the training time, training accuracy, testing time, testing accuracy, and classification report for various network model results. Furthermore, all three referenced models are discussed, as well as the best performing models and the logic behind the model's working protocol is explained in detail.

In the third section, we will create a node JS application with a locally hosted server that will intern with the python program consisting of all three models, scrape the data set, feed the data set into the python program to get the reference convolution and transformer based model outputs.

Finally, create a dashboard of the top 5 comments from the given data set, along with their predicted sentiment analysis and the percentage of positive and negative comments. It will also display the sampling of the word as a result of which a specific sentence has been marked as a positive or negative data set using word clouds in the dashboard.

5.1. Neural Network Models

Neural Networks are networks used in Machine Learning that function similarly to the human nervous system. It is intended to function similarly to the human brain, in which many things are linked in various ways. Some of the neural networks models used in our work are explained below:

5.1.1. BERT Model

As previously stated, the BERT architecture is one of the most advanced modelling techniques for creating a hyper dimensional kernel space vector of model transmission of lateral features. This not only uniquely identifies the positive and negative sentiments in the data set, but it also recreates hyperspace, which can be used to create feature boundaries between them. Images show a significant decrease in the loss function, indicating that the Adam optimizer in categorical cross-entropy is capable of not only creating this hyperspace but also reducing the loss consecutively for the first 10 epochs and producing a Hilton curve model.

```
Epoch 1/10
500/500 [==============================] - 231s 441ms/step - loss: 5.9922 - accuracy: 0.8587 - val_loss: 4.3980
Epoch 2/10
500/500 [==============================] - 218s 435ms/step - loss: 3.9239 - accuracy: 0.9707 - val_loss: 3.0076
Epoch 3/10
500/500 [==============================] - 218s 435ms/step - loss: 2.5350 - accuracy: 0.9882 - val_loss: 2.0318
Epoch 4/10
500/500 [==============================] - 218s 436ms/step - loss: 1.6155 - accuracy: 0.9890 - val_loss: 1.3761
Epoch 5/10
500/500 [==============================] - 218s 436ms/step - loss: 1.0024 - accuracy: 0.9937 - val_loss: 0.9741
Epoch 6/10
500/500 [==============================] - 218s 436ms/step - loss: 0.6125 - accuracy: 0.9973 - val_loss: 0.7751
Epoch 7/10
500/500 [==============================] - 218s 436ms/step - loss: 0.3792 - accuracy: 0.9946 - val_loss: 0.5803
Epoch 8/10
500/500 [==============================] - 218s 436ms/step - loss: 0.2345 - accuracy: 0.9967 - val_loss: 0.4918
Epoch 9/10
500/500 [==============================] - 218s 436ms/step - loss: 0.1521 - accuracy: 0.9955 - val_loss: 0.5051
Epoch 10/10
500/500 [==============================] - 218s 437ms/step - loss: 0.1014 - accuracy: 0.9969 - val_loss: 0.3893
```

Figure 35. BERT Epochs.

The BERT dense model's training graph is depicted below which indicates a decrease in curvature in both training and validation loss. It exhibits a strong increasing accuracy in both training and validation accuracy, as well as a direct proportionality of loss reduction and increase in accuracy for the same.

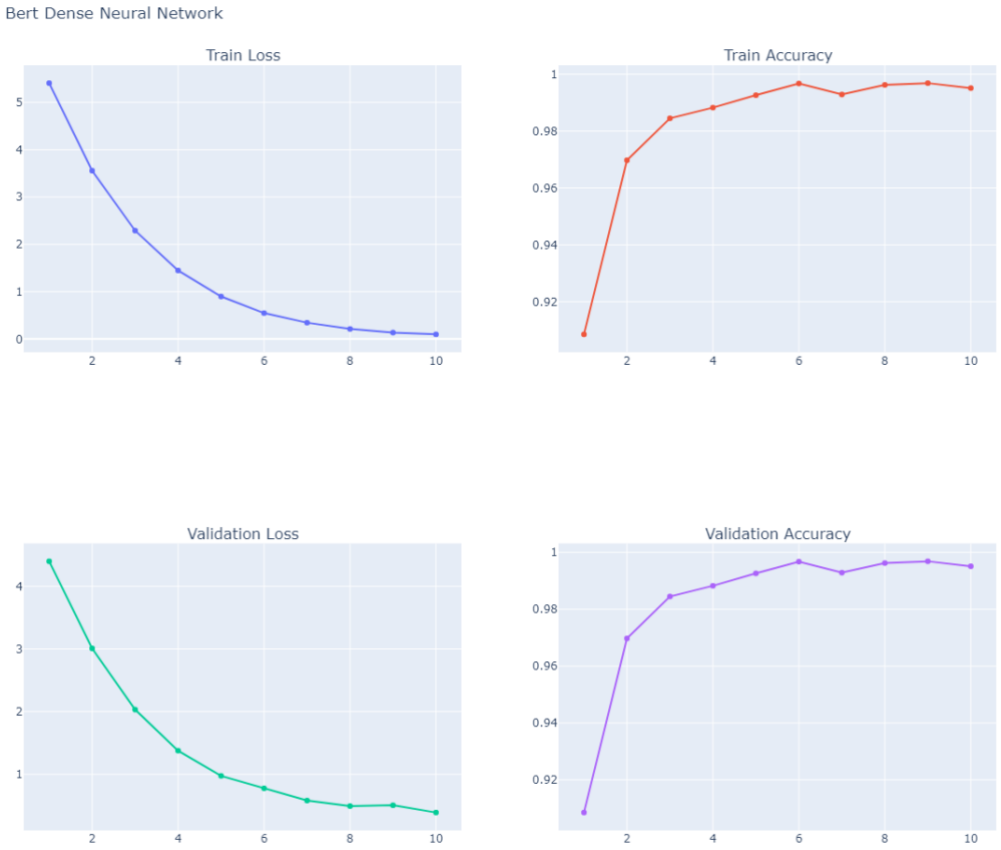

Figure 36. BERT Graphs.

Finally, on a separate test, training and validation set. BERT architecture achieves near-perfect accuracy on 2,000 samples of data taken completely apart from the training set. This demonstrates its one-of-a-kind ability to deduce feature differences in high altar determination.

```
Classification Report
              precision    recall  f1-score   support

         Neg       1.00      1.00      1.00      4074
         Pos       1.00      1.00      1.00      3926

    accuracy                           1.00      8000
   macro avg       1.00      1.00      1.00      8000
weighted avg       1.00      1.00      1.00      8000

Training and saving built model.....
```

Figure 37. BERT Classification report.

It is also important to note that the BERT architecture is trained on over 100 Million data sources and sentences thus very genuinely gives an accuracy of 100%. This should not be considered overfitting as it's not the training accuracy but rather the test accuracy. The training is however at 96.57% and testing is at 100%.

Understand the epoch cycles and the graph shown above. Epoch cycles attempts to train on the training data and then validate on the validation data; this plate is approximately 20% accurate, as shown by the visible accuracy of 99.69% after the epoch. It eventually converges to 99.69 %, which is the decimal value of the same. In this case, the graph depicts the Hilton curve for neural networks. This shows how the loss function has steadily decreased while the accuracy has risen. This is due to the use of the Adam optimizer for the same.

It is important to note that validation graphs are derived from data samples that were not used for training by the model. In other words, as accuracy and loss decrease, the model adapts to do the same in the validation or unknown set of data variables.

5.1.2. CNN Model

As previously stated, CNN is not as effective as our bi-architecture or recurrent neural network in language translation modelling. With such problems and difficulties, the convolutional neural network is able to demonstrate a lower loss function and increasing accuracy over the first 10 epochs.

```
Epoch 1/10
500/500 [==============================] - 29s 47ms/step - loss: 0.6930 - accuracy: 0.5079 - val_loss: 0.6916 -
Epoch 2/10
500/500 [==============================] - 23s 46ms/step - loss: 0.6884 - accuracy: 0.5449 - val_loss: 0.6870 -
Epoch 3/10
500/500 [==============================] - 23s 46ms/step - loss: 0.6848 - accuracy: 0.5497 - val_loss: 0.6764 -
Epoch 4/10
500/500 [==============================] - 23s 46ms/step - loss: 0.6575 - accuracy: 0.6355 - val_loss: 0.5395 -
Epoch 5/10
500/500 [==============================] - 23s 46ms/step - loss: 0.4607 - accuracy: 0.7947 - val_loss: 0.4077 -
Epoch 6/10
500/500 [==============================] - 23s 46ms/step - loss: 0.2922 - accuracy: 0.8880 - val_loss: 0.3669 -
Epoch 7/10
500/500 [==============================] - 23s 46ms/step - loss: 0.2197 - accuracy: 0.9179 - val_loss: 0.3569 -
Epoch 8/10
500/500 [==============================] - 23s 46ms/step - loss: 0.1741 - accuracy: 0.9351 - val_loss: 0.3559 -
Epoch 9/10
500/500 [==============================] - 23s 47ms/step - loss: 0.1406 - accuracy: 0.9504 - val_loss: 0.3669 -
Epoch 10/10
500/500 [==============================] - 23s 46ms/step - loss: 0.1170 - accuracy: 0.9609 - val_loss: 0.3998 -
```

Figure 38. CNN Epochs.

The convolution neural network function shows an initial slope function in which the loss and accuracy did not increase over the first 4-5 epochs but instead took a sharp instrument for accuracy and a decrement for loss. It's interesting to note that around the 8th episode of the lowest point laws appeared, followed by a gradual loss. In the ninth and tenth epochs, there is an increase. The same cannot be said for validation accuracy.

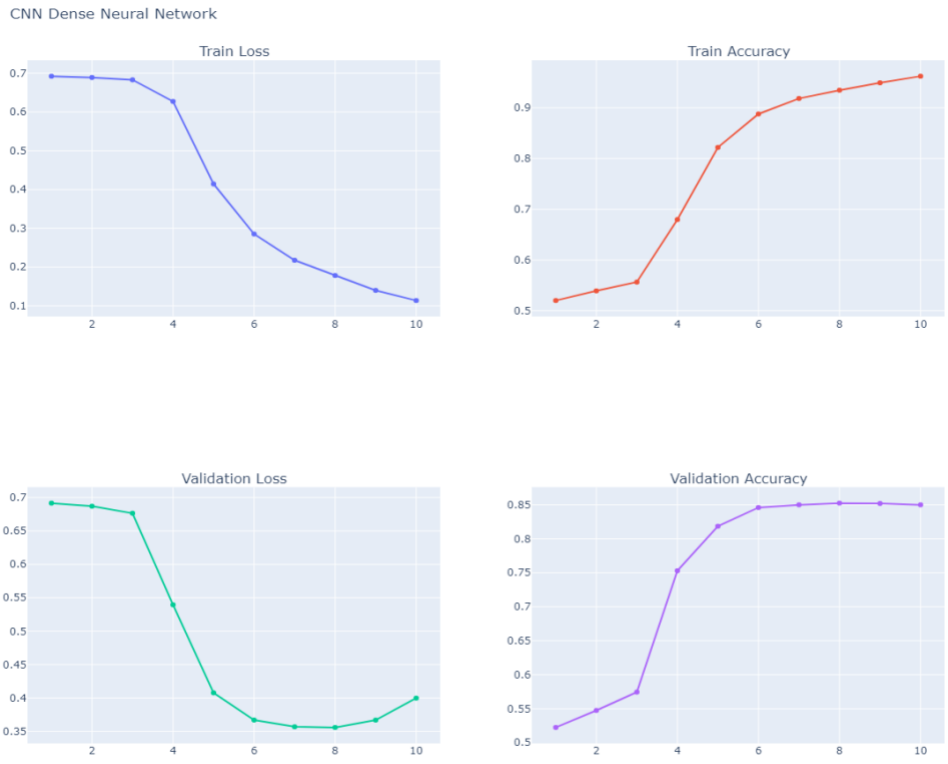

Figure 39. CNN Graphs.

In comparison to the bi-architecture, the convolution neural network achieves an 85 % accuracy in the test data set over 2,000 samples from the training data set.

Classification Report

	precision	recall	f1-score	support
Neg	0.89	0.81	0.85	1043
Pos	0.81	0.89	0.85	957
accuracy			0.85	2000
macro avg	0.85	0.85	0.85	2000
weighted avg	0.85	0.85	0.85	2000

Figure 40. CNN Classification report.

As explained in the BERT modelling for graphs, a similar architecture of Hilton curve is seen in this case as well, but because convolution neural networks are concerned with kernel neighbour concepts rather than time series. Thus, we find that the accuracy for the last epoch is close to 0.9670 or 96.7 % in training but 0.85 or 85 % in testing, which is its major disadvantage. Finally, we can discuss how the validation accuracy shows a slight increase after the eighth epoch, implying that the optimal training cycle would be 7 or 8 epochs. We take about 30 seconds per epoch for training and 60 seconds for testing.

5.1.3. RNN Model

As previously discussed, the recurrent neural network is one of the very first models of describing languages into computer coded networks. In our research and analysis, we discovered that recurrent neural networks are still a very valid and utility-based network, with an accuracy of around 86 % for 2000 unknown samples. Not only does it have the ability to create different colour spaces, as described above, but it also demonstrates how simple and conceptually clear models, such as recurrent neural networks, can provide a good amount of accuracy without training at 10 million parameters.

It's also worth noting that the origin of this extensive training accuracy is a divergence of decreasing loss and increasing accuracy in the first ten epochs of training. One must also consider the recurrent neural network's health in space graphs, as shown in the figure below, which shows a complete loss reduction and increase in accuracy even during training samples and randomization of the initial weights and biases.

```
Epoch 1/10
500/500 [==============================] - 215s 427ms/step - loss: 0.6105 - accuracy: 0.6960 - val_loss: 0.5160
Epoch 2/10
500/500 [==============================] - 209s 418ms/step - loss: 0.4322 - accuracy: 0.8195 - val_loss: 0.4225
Epoch 3/10
500/500 [==============================] - 208s 417ms/step - loss: 0.3432 - accuracy: 0.8618 - val_loss: 0.3911
Epoch 4/10
500/500 [==============================] - 210s 419ms/step - loss: 0.2924 - accuracy: 0.8885 - val_loss: 0.3782
Epoch 5/10
500/500 [==============================] - 212s 424ms/step - loss: 0.2551 - accuracy: 0.9080 - val_loss: 0.3767
Epoch 6/10
500/500 [==============================] - 212s 424ms/step - loss: 0.2299 - accuracy: 0.9195 - val_loss: 0.3921
Epoch 7/10
500/500 [==============================] - 212s 424ms/step - loss: 0.2028 - accuracy: 0.9264 - val_loss: 0.3690
Epoch 8/10
500/500 [==============================] - 214s 428ms/step - loss: 0.1782 - accuracy: 0.9380 - val_loss: 0.3609
Epoch 9/10
500/500 [==============================] - 214s 427ms/step - loss: 0.1575 - accuracy: 0.9482 - val_loss: 0.3543
Epoch 10/10
500/500 [==============================] - 216s 431ms/step - loss: 0.1470 - accuracy: 0.9509 - val_loss: 0.4043
```

Figure 41. RNN Epochs.

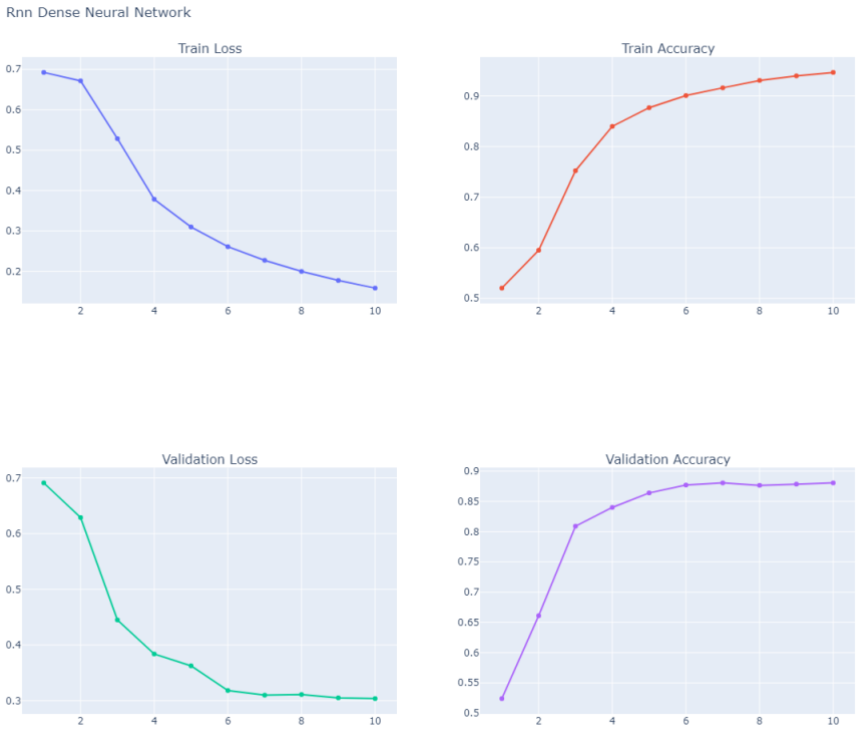

Figure 42. RNN Graphs.

Classification Report

	precision	recall	f1-score	support
Neg	0.86	0.88	0.87	1036
Pos	0.87	0.85	0.86	964
accuracy			0.86	2000
macro avg	0.86	0.86	0.86	2000
weighted avg	0.86	0.86	0.86	2000

Figure 43. RNN Classification Report.

5.2. Web Application Output and Dashboard

The webhook makes use of node JS functionality, which provides a dashboard feature for all reference networks such as the recurrent neural network, convolutional neural network, and transformer all the encoder-decoder neural network. In the three images below, we show how the recurrent convolutional and the transformer produced different results for the same data set components. It is also a validation of the accuracy with which each of them can process and achieve. Finally, we can reduce and highlight word clouds for words that are relevant in contextualizing the positivity or negativity of a comment.

One of the most creative approaches to understanding how a neural network works in relation to a specific example of output. It also aids in the comprehension or creation of explainable artificial intelligence and neural network versioning systems, which is a feature use case for this.

5.2.1. Results from RNN Model

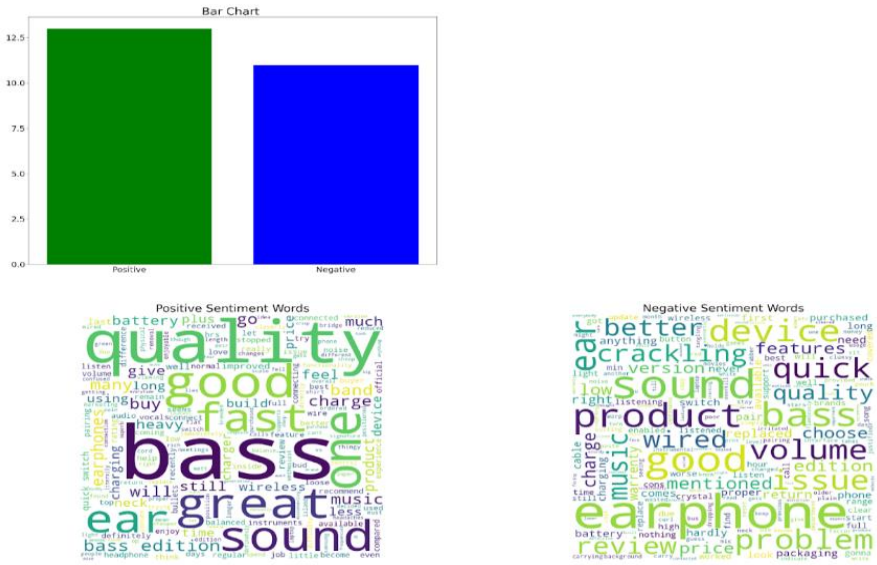

Figure 44. RNN Dashboard.

In the recurrent neural network dashboard we are able to deduce the number of positive and negative comments predicted by the model. The word clouds for the positive and negative comments showcase the different words corresponding to the prediction of positive or negative comment. This in turn also showcases the ability of the recurrent neural network model.

5.2.2. Results from CNN Model

Figure 45. CNN Dashboard.

Convolution neural network model dashboard showcases the number of users for both positive and negative comments and then illustrates the reason for a comment to be negative or positive bye by creating two separate word clouds for the same.

5.2.3. Results from BERT Model

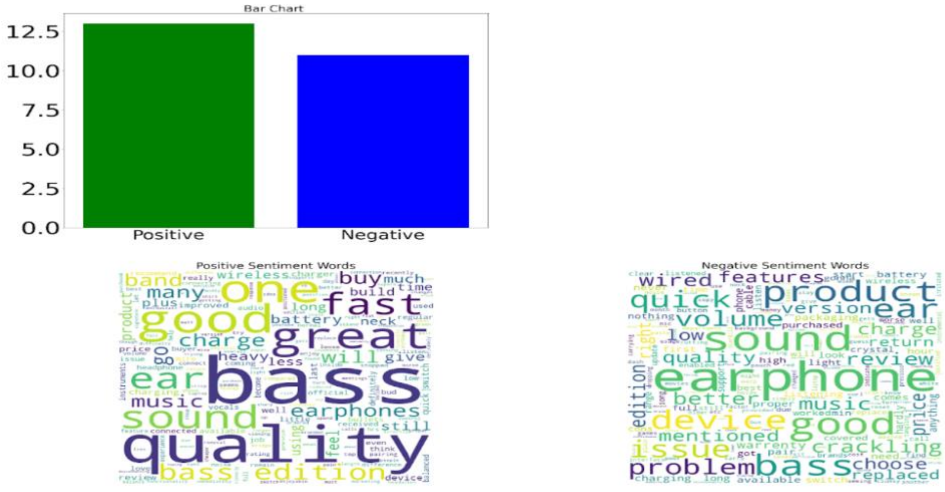

Figure 46. BERT Dashboard

Explain the ability of artificial intelligence is a rather important subject to talk about. As we can see above for the BERT architecture, the dashboard showcases the number of positive and negative comments along with their respective word clouds as explained above. This however at the very preliminary level showcases how AI can be reduced through various visualisation techniques.

5.2.4. Final View of the Dashboard

As we have already discussed the dashboard contains a textbox where the user can provide the link for scraping the data. Once the data is scraped it is then fed into all the three different architectures. Then the top 10 comments along with the dashboard of recurrent neural network, convolutional neural network and Bidirectional Encoder Representations from Transformers model is displayed.

The Picture below shows how the user would view the dashboard of the web application.

Sentiment Analysis Using RNN, CNN & BERT

https://www.amazon.in/OnePlus-Bullets-Wireless-Bass-Black/product-reviews/B092ZJVB6Z/r Submit

#	Review Content	Predictions RNN	Predictions CNN	Predictions Bert
1	Update after 1 month of use. There is crackling sound in the background that comes during calls games music anything randomly. During calls, once the crackling starts, it continues. It is known issue with the buds, and you need to reset the device to get it back, and then again the crackling starts in 3-10 hours. Orignal review was for 3 stars as below: Pros: + Premium packaging + Quick pairing with OP phone + Stylish + (Hopefully) good battery + USB C interface - no need for me to carry another charging cable + Enviromental noise cancellation + Quality of mic Cons: - Very very poor sound quality. A 399/- pair of wired earphones have better sound quality. - The red cable provided is a sham. It is not warp or dash charging enabled. Also, it is too short! The cost cutting in this department at 2k is not justified - Clumsy, the rubber and buttons are plain clumsy. They never sit properly, even curl some neck clumsily. On dropping the bus down, they hardly ever	Negative Comment	Negative Comment	Negative Comment

#	Review Content	Predictions RNN	Predictions CNN	Predictions Bert
1	Bakwas phone	Positive Comment	Positive Comment	Positive Comment
2	Few issues are resolved that are mention below..but still this mobile hasn't the charm of OnePlus.. It is like a normal android..not an OnePlus.. Low processing. Exceptional features of OnePlus are not there in this phone.. Took so long time to delete items.. Battery is draining fast.. Charging is not as fast as claimed.. Earlier I was using OnePlus 5T..I bought it to upgrade but unfortunately it degrades....	Negative Comment	Negative Comment	Negative Comment
3	The best part of the phone is it's super simple and user friendly operating system. Camera doesn't oversaturate colours and captures decent photos. Battery charging is super fast. People argue that this is overpriced in this segment but I guess ease of using the phone and mainly the OS wouldn't be so good in other comparable phones. It is super light and handy. Edit: taking away 1 star from the review after a month's usage. Phone app lags all the time. Phone starts ringing when I receive a call while the name of caller appears 5 seconds later. This is very annoying. Similar issue while calling. Hope some software update fixes this issue.	Positive Comment	Positive Comment	Positive Comment
4	Phone looks good but features are 10k phone Camera is worst Performance is not good I recommend to go like vivo or oppo	Negative Comment	Positive Comment	Negative Comment

Bert Model

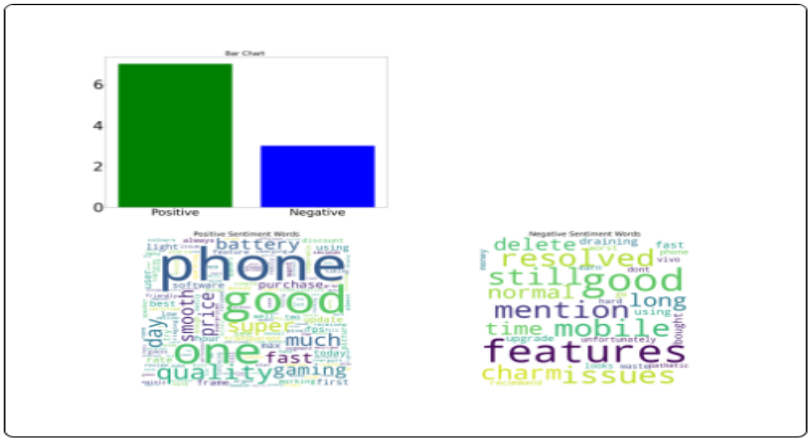

Figure 47. Final Dashboard.

Chapter 6

Discussion & Conclusion

6.1. Discussion

In this dissertation we have discussed the modelling of various different architectures like neural network convolutional neural network and the transformer architecture. Not only have we elaborately discussed the utility of these architectures over sentiment analysis data set but also evaluated on different parameters like training time, accuracy precision and F1 score. However there is a lot of debate about the utility of Transformers and the memory leakage issue they have. We learned in this dissertation that a transformer, which is a general and encoder decoder versioning system, is memory and stack size dependent. Which is generally quite difficult to come by, which is why training a transformer could be very inefficient compared to the standard GPU and TPU available on the market.

We analysed and described the challenges and problems in the following section and discussed the lack of maximum RAM memory, which is required for training a transformer. Even if we consider all of the disadvantages that a transformer may perceive, we cannot dismiss the ability to create a hyper-dimensional kernel space to create boundary separation between positive and negative sentiments.

6.1.1. Challenges

Deep learning is the science of extracting features from a given data sequence, and it is fraught with challenges:

As we all know, in the world of Pico bytes, this data can be massive. As a result, heavy computation is required on parallel processing tasks. In this case, we trained on multiple GPUs in the same time slot, achieving high accuracy and parallel threshold while also incurring high costs from data, we only learn about pitfalls in data after training, wasting GPU time. As a result, it's equally important to spend quality hours manually understanding the data and creating sample kernel spaces by deriving the derivatives of backpropagation so that model time and GPU cycles aren't wasted.

Around 3-4 hours of GPU time is utilised for each model. In a way of modelling, debugging and finally getting the results from each case. This too on the amazon EC2 high end server.

6.1.2. Future Scope

The potential scope of sentiment analysis lies among the discussion of transformers and its memory utilization problem. In this research we have already identified the utility of sentiment analysis and the creation of higher dimensional kernel spaces is a potential market for the future. It is also critical to say with certainty that the current transformer architecture is not an ideal utility space for deploying the models. The absence of a low memory initialization and execution cycle is one of the current problems in transformer architecture. As a result, there is a strong reliance on the availability of higher and graphical processing units, as well as tensor processing units. In general, they are inaccessible to more than 99.9% of the world's population.

It can only be stated through the high utility of cloud architecture as the transformer's potential can be utilized, but there is also a large scope for decreasing the transformer's memory utility. As previously stated, the current sentiment analysis model works on the Amazon sentiment system's dynamo DP and prefers long short term memory networks. Finally, we conclude that the future of sentiment analysis is still as bright as it was 20 years ago, but the complexities of developing a highly accurate model have given way to a highly deployable model today. This also highlights the current

growth of the artificial intelligence and deep learning industries, as well as the promise of a low memory cost network in the future.

6.2. Conclusion

In today's market, it is critical to comprehend and analyse the significance of sentiment analysis. This dissertation has not only concentrated on the utility of sentimental analysis of the market scenario of various products listed on the Amazon Web page but aims to use cutting-edge deep learning methodologies such as the recurrent neural network, convolution neural network, transform architecture, and long short term memory network. We were able to not only reduce the utility of the node.js server during implementation, but also create a spawn function to run our python files using node.js. As shown in the flowchart, we can not only scrape data from a given link, but also run all relevant networks and obtain the desired output. On the web server, it is displayed as a dashboard.

It is also worth noting that the transformer architecture has been trained over 100 million parameters and has one of the highest accuracy rates of around 99.9 percent. However, the recurrent neural network and the convolution neural network perform extremely well in temporal data segments, and the parameter is provided. We have separated our samples into training, testing, and validation sets. The solution neural network achieves approximately 85 percent accuracy on the test set, while the recurrent neural network achieves approximately 86 percent accuracy on the test set. In the results section above, the validation and training loss graphs are also shown to submerge at a Hilton curve. Also, the epoch by epoch decrease in loss and increase in accuracy is very persistent, indicating that the model's ability to submerge the hyperdimensional spaces of understanding the reviews as positive or negative is quite efficient.

The submission of the project includes the evaluations, report and the project code. Also, Consent Right is available for public testing and modification in GitHub repo.

Bibliography

[1] Elli, Maria Soledad, and Yi-Fan Wang. "Amazon Reviews, business analytics with sentiment analysis." 2016

[2] Gurmeet kaur and Abhinash Singla, “Sentimental Analysis of Flipkart reviews using Naïve Bayes and Decision Tree algorithm”, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5 Issue 1, January 2016.

[3] Konstantin Bauman, Bing Liu and Alexander Tuzhilin, “Aspect Based Recommendations: Recommending Items with the Most Valuable Aspects Based on User Reviews”, KDD ‟17, August 13-17, 2017, Halifax, NS,Canada

[4] Xing Fang and Justin Zhan, “Sentiment Analysis using Product Review Data”, Journal of Big Data 2015

[5] Zhi Li, Rui Li, GuanghaoJin,”Sentiment Analysis of Danmaku Videos Based on Naïve Bayes and Sentiment Dictionary”, 2020

[6] Li Yang, Ying Li, Jin Wang, ”Sentiment Analysis for E-Commerce Product Reviews in Chinese Based onSentiment Lexicon and Deep Learning”, 2019

[7] Sosa, P. M. (2017). Twitter sentiment analysis using combined LSTM-CNN models. Eprint Arxiv, 1-9.

[8] J. Wehrmann, W. Becker, H. E. L. Cagnini, and R. C. Barros, “A character-based convolutional neural network for language-agnostic Twitter sentiment analysis,” Proc. Int. Jt. Conf. Neural Networks, vol. 2017–May, pp. 2384–2391, 2017.

[9] P. K. Patil and K. P. Adhiya, “Automatic Sentiment Analysis of Twitter Messages Using Lexicon Based Approach and Naive Bayes Classifier with Interpretation of,” pp. 9025–9034, 2015.

[10] Subakan, C., Ravanelli, M., Cornell, S., Bronzi, M., & Zhong, J. (2021, June). Attention is all you need in speech separation. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 21-25). IEEE.

[11] Haque, T. U., Saber, N. N., & Shah, F. M. (2018). Sentiment analysis on large scale Amazon product reviews. 2018 IEEE International Conference on Innovative Research and Development, ICIRD 2018. https://doi.org/10.1109/ICIRD.2018.8376299

Note: There are few images in this dissertation which have been taken from Google images.