COMPARATIVE STUDY OF SENTIMENT ANALYSIS FOR MOVIE REVIEWS

STUDENT NAME: SANKET TALAVIYA

STUDENT ID: 969198

INTERIM REPORT

LIVERPOOL JOHN MOORES UNIVERSITY – MS IN ML AND AI

DECEMBER 2020

**DEDICATION**

I dedicate this work to my wife Bhumika, my little daughter Praanvi and My parents, who have been there for me throughout this research work. You all have been my biggest supporters. Without your support I couldn’t have continued my dream of doing Masters in Machine Learning and Artificial Intelligence.

**ACKNOWLEDGEMENTS**

I would like to thank Dr. Ahmed for the weekly sessions and guidance in preparing this report. I would also like to thank my thesis supervisor Anurag Maji for his valuable suggestions and feedbacks that helped preparing this research report.

# 

**TABLE OF CONTENTS**

DEDICATION ii

ACKNOWLEDGEMENTS iii

ABSTRACT vi

LIST OF TABLES vii

LIST OF FIGURES viii

LIST OF ABBREVIATIONS x

CHAPTER 1: INTRODUCTION .............................................................................................. 1

1. Background of the Study............................................................................................... 1
2. Problem Statement ……………………….................................................................... 3
3. Aim and Objectives ………………………………....................................................... 3
4. Scope of the Study ........................................................................................................ 3
5. Significance of the Study .............................................................................................. 4
6. Structure of the Study.................................................................................................... 4

CHAPTER 2: LITERATURE REVIEW ................................................................................... 5

1. Introduction ....................................................................................................................5
2. Sentiment Analysis in movie reviews domain................................................................ 6
3. Sentiment Analysis in product reviews domain........................................................... 12
4. Sentiment Analysis in other domains............................................................................ 17
5. Discussion.................................................................................................................... 26
6. Summary ...................................................................................................................... 26

CHAPTER 3: RESEARCH METHODOLOGY ..................................................................... 27

1. Introduction ................................................................................................................. 27
2. Methodology …………………………........................................................................ 28
   * 1. Data Selection ……………………….............................................................. 28
     2. Data Pre-Processing …………………………………………………………. 28
     3. Exploratory Data Analysis…………………………………………………… 29
3. Proposed Method …………………………………..…............................................... 29

3.3.1 Embeddings Considered …………………………………………………….. 29

3.3.2 Traditional machine learning models ……………………………………….. 33

3.3.3 Transformer based model - BERT …….…………………………………….. 34

1. Proposed Flow Diagram .............................................................................................. 35

3.5 Evaluation Metric Used…………………………….…………………….………….. 35

3.6 Expected Outcomes………………………………………………………………...…36

3.7 Required Resources………………………………………………………………...…37

3.7.1 Hardware Resources…………………………………………………………… 37

3.7.2 Software Resources….………………………………………………………… 37

3.7.3 Additional Resources……………………...…………………………………… 37

3.8 Risks and Contingency Plan….……………...……………………………………...... 38

3.9 Summary………………………………………………………………….………….. 38

REFERENCES ........................................................................................................................ 39

APPENDIX A: RESEARCH PLAN ……............................................................................... 43

APPENDIX B: DATASET URL ….…………........................................................................ 44

# ABSTRACT

In today’s world, Sentiment analysis has become the need of the hour for opinion mining on variety of subjects. It has various applications in marketing, e-commerce, advertising, politics, and research. Day by day people expressing their sentiments over internet, different websites and social media are increasing. One such domain is the entertainment industry, especially movies. Users have been expressing their views on a particular movie through various popular websites such as IMDB, Rotten Tomatoes, Twitter etc. These reviews help other people to decide whether they should spend their precious time in watching a particular movie or not. And hence lot of people go through movie review comments to find out whether they should watch a particular movie. Sentiment analysis is the process of classifying these review comments. If one doesn’t apply machine learning and Artificial Intelligence techniques to find out overall sentiments of the topic going through all review comments. It would take humongous effort to label this review comments manually to figure out overall sentiments. So, if we automate sentiment analysis activity using machine learning and Artificial Intelligence, it can help saving a lot of human efforts. So far many experiments have been done using feature engineering and pre-processing using BOW, TF-IDF, Doc2Vec, Word2Vec embeddings and different machine learning models like Logistic Regression, SVM, Random Forest. Although good results have been achieved using these techniques there is still scope of doing better in feature engineering space through applying contextual embedding methods like BERT embeddings which has been introduced by the state-of-the-art model BERT. Context free embedding techniques like TF-IDF, Word2Vec etc. ignore the meaning of the word in different contexts as it creates only one representation of a given word in entire vocabulary. In contrast to that, contextual embedding technique like BERT learns different representations of a given word based on their context. While there are few experiments already done on IMDB movie review dataset using context free embedding techniques. In this research experiments will be carried out using contextual BERT embeddings on Logistic Regression, SVM, NB and Random Forest and transformers-based model – BERT. Then performance comparison of these models will be done and that will be compared against other similar papers.

# LIST OF TABLES

Table 3.1 Dataset columns overview…………....................................................................... 28

Table 3.2 Risk Mitigation Plan……………........................................................................... 38

# LIST OF FIGURES

Figure 2.1 Custom Embedding Layer and Results………………………………………….… 6

Figure 2.2 Proposed Methodology………………………………………………………….…. 6

Figure 2.3 Model Comparisons…………………………………………………………….….. 7

Figure 2.4 Methodology and Results……………………………………………………….…. 8

Figure 2.5 Feature Extractions and Vectorizations………………………………………….… 8

Figure 2.6 Comparison of 8 classifiers………………………………………………………... 9

Figure 2.7 Architecture of Two-Stream Self-Attention for Target-Aware Representations….. 9

Figure 2.8 Methodology Flowchart……………………………………………………….…. 10

Figure 2.9 DNN based architecture………………………………………………………..…. 11

Figure 2.10 Proposed opinion mining architecture………………………………………....... 12

Figure 2.11 Traditional & Deep learning model results ………………………….…….....…. 13

Figure 2.12 Flow Diagram of the System Architecture…………………………………........ 13

Figure 2.13 Accuracy of traditional ML models……………………………………………... 14

Figure 2.14 Classification System Diagram for Sentiment Analysis………………………… 14

Figure 2.15 Comparison of ML model performance…………………………………………. 15

Figure 2.16 Using word2vec and CNN to classify opinions…………………………………. 16

Figure 2.17 Proposed Model & Results……………………………………………………… 16

Figure 2.18 Proposed Methodology and Results……………………………………………... 17

Figure 2.19 The architecture of CBOW and Skip-gram algorithms…………………………. 18

Figure 2.20 Process Flow diagram…………………………………………………………… 18

Figure 2.21 Comparison of ML and DL model performance………………………………… 19

Figure 2.22 Architecture of RNN based model……………………………………………… 20

Figure 2.23 Proposed aspect-level sentiment classification model…………………………... 21

Figure 2.24 Results of fine-tuned BERT for sentiment analysis……………………………... 21

Figure 2.25 Architecture diagram of Research framework…………………………………... 22

Figure 2.26 Architecture for sentiment analysis……………………………………………… 22

Figure 2.27 Proposed architecture with CNN Bi-LSTM……………………………………. 23

Figure 2.28 Overall selection process of the best performing pre-processing techniques…… 24

Figure 2.29 Architecture of DNN and CNN…………………………………………………. 25

Figure 2.30 Architecture of the proposed hybrid deep learning model………………………. 25

Figure 3.1 TF-IDF Representation…………………………………………………………… 31

Figure 3.2 Skip Gram Representation……………………………………………………….. 32

Figure 3.3 BERT embeddings……………………………………………………………..…. 33

Figure 3.4 Proposed Methodology Flow Diagram………………………………………….... 35

# LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| AUC | Area Under ROC Curve |
| BOW | Bag of Words |
| BERT | Bidirectional Encoder Representations from Transformers |
| BN | Bayes Net |
| CNN | Convolutional Neural Network |
| DNN | Deep Neural Network |
| DT | Decision Tree |
| EDA | Exploratory Data Analysis |
| FN | False Negative |
| FP | False Positive |
| FPR | False Positive Rate |
| IDF | Inverse Document Frequency |
| KNIME | Konstanz Information Miner |
| KNN | K-nearest Neighbours |
| LSTM | Long Short-Term Memory |
| NB | Naïve Bayes |
| ME | Maximum Entropy |
| MM | Markov model |
| RF | Random Forest |
| ROC | Receiver Operating Characteristic |
| RRL | Ripper Rule Learning |
| SGD | Stochastic gradient descent |
| SVM | Support Vector Machine |
| TF | Term Frequency |
| TN | True Negative |
| TP | True Positive |
| TPR | True Positive Rate |
| XGBoost | Extreme Gradient Boosting |

# CHAPTER 1

# INTRODUCTION

# 1.1 Background of the Study

Sentiment analysis is useful for finding out how customers think about a particular product or topic. Before computer era started, In the earlier times sentiment analysis was done primarily on non-online modes like paper or written documents as there was no internet services. In the recent times internet services have become part of the human life and hence more and more people have started expressing their views on online forums including social medias, publications, blogs, websites etc. Sentiment Analysis is a process of extracting sentiments from these raw reviews expressed over above various internet forums using NLP and statistics.

As Internet services are evolving more and more each day, Exponential growth has been witnessed in terms of information availability on the internet. This includes opinions and reviews for particular product or topics like movies. (Yousefpour et al., 2014). Because of tremendous value sentiment analysis brings to the table by eliminating manual efforts of parsing all reviews and figure out overall sentiment of the subject in question. Lot of researchers are attracted to this field. This area of research broadly involves NLP and Machine learning models. (Qaisar, 2020). Sentiment analysis in each field had different objectives for example if its e-commerce website reviews for a product. From the sentiment analysis product companies can gather information how good or product is doing in the market by deriving sentiments from the reviews. If the subject is movie reviews, It helps users to decide whether they should view this movie or not based on sentiment analysis that way it helps them save their time and movie if overall sentiments are negative.

Although algorithms have progressed pretty well in last few years, few challenges in this field have not been completely eliminated yet. One of these challenges is to classify statements to positive or negative even though it doesn’t have any clear emotions and sometimes statements can have words which have different meanings based on the contexts so that leads in to ambiguity in the end result of the classification.(Qaisar, 2020)

Film Industries including songs, trailers, movies etc rely heavily on sentiment analysis to figure out how people are feeling about particular song/trailer or movie. This sentiment analysis can also give pointers to other producers on what people don’t like in movies or songs etc based on that they can take actions for their next movies. And existing movie reviews help user to decide on which movie to watch or not to watch etc and film makers in a way that what particular things people like the most and disliked the most. The dataset here is taken from Kaggle and it contains IMDB movie reviews. (IMDB Dataset of 50K Movie Reviews | Kaggle, 2021)

Lot of research has already been done in this area using different context-free embedding techniques like BOW, TF-IDF, Word2Vec with different machine learning models but there is a scope of doing more research on sentiment analysis with recently introduced contextual embedding techniques like BERT embeddings with different machine learning models.

Broadly below are the two types of sentiment analysis techniques.

1. **Machine Learning based techniques**: This technique mainly falls into supervised classification category, in this technique two set of datasets are required, train dataset and test dataset, Train dataset is used to learn various patterns between variables of the subject and test document is used to check the robustness of the model on the dataset that model hasn’t seen yet. Logistic Regression, SVM and Naïve Bayes have achieved good performance in this area. One important decision to make in this method is to do feature selection. The most widely used features in this method are as (1) Term frequency (2) Part of Speech Information (3) Negations (4) Opinion words and phrases.
2. **Lexicon based techniques**: It is unsupervised technique as it requires prior knowledge of sentiment lexicons. In this technique features of a given statements are compared with sentiment lexicons whose values are known in advance. These lexicons contain the word list that represents people’s subjective feelings and opinions. Once we have these lexicons handy next task is to go through document and find out positive and negative lexicons in each statement and if statement has more positives, it’s a positive sentiment otherwise it’s a negative sentiment. There are few methods to build sentiment lexicons (1) Manually (2) From Corpus (3) From Dictionary. (Vohra and Teraiya, 2013)

# 1.2 Problem Statement

This research focuses on saving users’ precious time by suggesting them accurate sentiments of a particular movie. This research also can be used by other movie makers and actors to understand how their movie is being rated by critics. That will help them taking better actions based on these sentiments in the future movies. Most of the existing researches have solved this problem with use of different embedding techniques like BOW, Word2Vec, Doc2Vec, or TF-IDF with different machine learning models like SVM, LR, Decision Tree, Random Forest, Naïve Bayes etc. So there is a scope of solving following problems.

1. Can we get more accurate model that can help predicting movie reviews better to users?
2. Can contextual embedding techniques like BERT Embeddings do better than other traditional embedding techniques like Word2Vec, TF-IDF etc.

# 1.3 Aim and Objectives

This research is aimed to propose a model to classify sentiments of IMDB movie reviews dataset with high accuracy and the goal of this research is to help users who want to get accurate idea about overall sentiments on a particular movie that can help them deciding if they should watch that particular movie.

Based on the goal of the study, Following objectives have been identified.

1. Find out the related research work that has been done so far in sentiment analysis domain for movie reviews.
2. Find out the best word embedding technique feature engineering approach.
3. Build machine traditional machine learning and transformer models.
4. Compare the performance of these models and Identify the most accurate model.

# 1.3.1 Business Objectives

In the recent times, Social media has become the primary channel for expressing views on a particular topic. In the movies domain, These reviews act as input for filmmakers and actors to figure out which part of the film users liked/disliked. Also these reviews act as input for end users who want to decide on whether they should spend few hours on watching particular movie or show.

* Discover and algorithm that can identify the polarity of reviews accurately, i.e. whether the review is positive or negative.
* This will help not just the movies domain but other domains as well like sentiment analysis on e-commerce product reviews, that will help end users to find out whether to should buy a particular product or not based on overall sentiments.

# 1.4 Scope of the Study

As the time duration for this research is limited, research scope will be limited to following points:

1. The dataset used here is of IMDB movie reviews and it has been taken from Kaggle, so assuming data is already valid.
2. In this research, scope will be limited to develop and compare the performance of different machine learning classification models using TF-IDF, Word2Vec Embedding and BERT embeddings only.
3. Machine learning Model experiments will be limited to Logistic Regression as interpretable models and SVM, Random Forest as part of black box models. Deep learning models will not be evaluated in this research or one model like BERT will be evaluated only if time permits.

# 1.5 Significance of the Study

Critics today often uses social media and movie rating websites to post their comments or rating for the movies. This data can eventually help users in decision making whether to watch that movie or not and that can really save users from wasting their precious time. Now if user starts reading all movie reviews that are provided on internet or social media sites, it will consume a lot of time as it’s a non-trivial task to go through all reviews manually and figure out overall sentiments on a particular movie. Through machine learning models we can automate the process of sentiment analysis from the pool of movie review comments and figure out what is the overall sentiment for the movie. This involves Natural language processing underneath and benefit that we get is incredible in terms of money and time saved. This sentiment analysis is basically a classification problem whether a particular comment is positive or negative or neutral and then for the whole movie how many comments are positive and how many of them are negative, this way we can achieve overall sentiment for the movie. So, this research would provide more accurate results to end users to help them taking better decisions on whether they should watch a particular movie or not based on sentiments expressed through movie reviews.

# 1.6 Structure of the Study

The first chapter contains an introduction to the issue, a background examination of the research, and issue statements. The research's goals and objectives are also given, followed by a discussion of the study's significance and scope.

The second chapter contains literature review of sentiment analysis in various domains like product reviews, movie reviews etc. It covers existing research that have been done using variety of techniques like data pre-processing, feature extraction, embeddings, machine learning models and deep learning models. Towards the end of this chapter various outcomes have been documented and conclusions have been provided.

# CHAPTER 2

# LITERATURE REVIEW

# 2.1 Introduction

In sentiment analysis field lot of research has already been done so far. A lot of work has been done with traditional machine learning models, deep learning models, Hybrid models, with different embedding techniques like BOW, Word2Vec, FastText etc. So, It is essential that we study what kind of work has been done so far and identify what is still missing in this previous work or there is a scope of improvement. So, in this chapter some of the previous work is captured Also, details are captured around the datasets, methodology and evaluation metrics used during the research.

This chapter is divided into 3 sections, first section provides details around recent research in the movie reviews domain with IMDB and Stanford movie review dataset. Second section outlines research in online product reviews domain with amazon.com product reviews and other online tweets for e-commerce products and third section provides overview of research in other domains with dataset around online tweets, Online Restaurant reviews, Global pandemic COVID-19 dataset etc. Overall, we have tried reviewing previous work that has been done on single dataset as well as on multiple datasets. Some of the research proposes new hybrid models and compare their performances with existing once and some research papers are covering comparative studies of existing models.

In the final section we will summarize the discussion on the literature review and identify gaps where this research will be focused.

# 2.2 Sentiment Analysis in Movie Reviews domain

(Haque et al., 2019) in their research used IMDB dataset and applied CNN and LSTM based models and compared their performances. IMDB dataset contained around 50000 reviews. Results showed that CNN yielded the best performance amongst all compared models. Instead of using pre-trained embeddings they tried to train embedding layer using IMDB training data.

|  |  |
| --- | --- |
| Methodology  Chart, waterfall chart  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.1 Custom Embedding Layer and Results (Haque et al., 2019)

(Moolthaisong and Songpan, 2020) in their research used dataset from ﻿Metacritic website and after pre-processing they changed the unstructured data into structured data using stringtoword vector filter. They used TF-IDF as word embedding technique and applied NB, Random Forest and J48 algorithms and ﻿got accuracy of 80.25%, 79.83% and 68.06% respectively.

|  |
| --- |
| Diagram  Description automatically generated |

Figure 2.2 Proposed Methodology by (Moolthaisong and Songpan, 2020)

(Tripathy et al., 2016) concluded in the research that accuracy of classification model is inversely proportional to the value of n in n-gram. When tests were done using higher n-gram models where n is greater than 3, Accuracy decreased for models like Maximum Entropy, SVM, NB, Stochastic Gradient De-scent(SGD). However unigrams and bigrams results were remarkably better. In this study IMDB dataset was used.

|  |
| --- |
| Table  Description automatically generated |

Figure 2.3 Model Comparisons (Tripathy et al., 2016)

(Sahu and Ahuja, 2016) examined IMDB dataset and proposed an approach to classify sentiment expressions from zero to four on scale and extracted the features and applied ranking and used these features for training multiclass classifier to classify reviews to a correct label. They followed approach based on n-grams and after applying classification models they achieved accuracy of 88.95% as a best one. Naïve Bayes performed the worst with accuracy score of 54.77%. Performance of Decision Tree was very near to Random Forest. Precision, Recall, F1- score, AUC and accuracy were used as evaluation metrics. Following table shows the methodology used in the research and the results.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.4 Methodology and Results (Sahu and Ahuja, 2016)

(Tarımer et al., n.d.) conducted a study on IMDB and twitter dataset, in this study they applied Decision tree, NB, SVM machine learning algorithms on a vector space that was created in KNIME platform. That is one analytics platform. Results they obtained were showing Decision tree was performing better than NB and SVM. And when they tried for other datasets as well and finally concluded that SVM yields better results across different datasets. Paper also highlighted that Decision tree doesn’t work with same accuracy on IMDB data and Twitter data because of twitter dataset had more spelling mistakes than IMDB.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.5 Feature Extractions and Vectorizations (Tarımer et al., n.d.)

(Yasen and Tedmori, 2019) In this research, They used IMDB dataset and applied around 8 different machine learning models after cleaning and pre-processing the dataset and they evaluated using 5 different metrics. Results concluded that Random Forest outperformed the other classifiers and Ripper Rule Learning performed the worst on the datasets. In their research they applied ML models after steps like tokenization, stemming and feature-selection.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.6 Comparison of 8 classifiers (Yasen and Tedmori, 2019)

(Pipalia et al., 2020) in their research presented classification power of pre-trained models like BERT, XLNet, T5 etc. IMDB dataset was used in this research with 50000 movie reviews. In the pre-processing step they removed special characters, HTML tags and emoji characters. Research concluded that XLNet yield the highest accuracy amongst all compared models. Training time of DistilBERT was half compared to BERT and yet not compromised on accuracy much.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.7 ﻿Architecture of Two-Stream Self-Attention for Target-Aware Representations (Pipalia et al., 2020)

(Tripathi et al., 2020) published comparative study of four different machine learning algorithms Naïve Bayes, Logistic Regression, Random Forest, and Decision Tree on IMDB movie reviews dataset. Comparison was done on metrics like accuracy, precision, recall, F1 score and Area under curve. Text Normalization, HTML tag removal, stop words removal, Feature extraction were done as pre-processing step. BOW and TF-IDF were used as vectorization technique. Result showed that TF-IDF with Logistic regression gave the best AUC score of 96%.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results with BOW  Table  Description automatically generated  Results with TF-IDF.  Table  Description automatically generated |

Figure 2.8 ﻿Methodology Flowchart by (Tripathi et al., 2020)

(Kumar Singh et al., 2020) in their research published comparative study of deep learning models with traditional machine learning models on IMDB dataset. NB, Random Forest, SVM, LR, LSTM and CNN were used as machine learning models. In this research Logistic Regression on the word2vec was used as a base model (79.5% accuracy). They used hyper parameters tuning for better performance. Experiments were done around learning rate, different values of epochs, for regularization parameters dropouts, L1, and L2 regularizations, and for activation functions ReLU, tanh, sigmoid functions were used. Overall paper concluded that deep learning models performed better than traditional machine learning models. LSTM with ReLU yielded the best accuracy score of 93.75%.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Chart, bar chart  Description automatically generated |

Figure 2.9 ﻿DNN based architecture (Kumar Singh et al., 2020)

(Wazery et al., 2019) in their research applied traditional machine learning models and deep learning models on IMDB dataset, Amazon dataset and Airline tweets and compared their performances. Word bag and word embeddings were used for feature extraction. Accuracy, Precision, Recall and F1-score were used for model evaluation. Result showed that deep learning based RNN-LSTM model yields the best accuracy score for all the datasets. NB doesn’t do well on Amazon and Airline dataset but on IMDB dataset mostly because of data formats.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Amazon Dataset Results  Table  Description automatically generated  IMDB Dataset Results  Table  Description automatically generated  Airline Dataset Results  Table  Description automatically generated |

Figure 2.10 ﻿Proposed opinion mining architecture (Wazery et al., 2019)

# 2.3 Sentiment Analysis in Product Reviews domain

(Tamara and Milićević, 2018) did the comparative analysis by applying BOW, BOW with n-grams and TF-IDF embeddings to different machine learning models like Logistic Regression, SVM and few deep learning models like LSTM and ConvNet. Glove and Word2Vec embeddings were also tested here. In this research amazon product review dataset was used and paper concluded that deep learning models perform better than traditional machine learning models. LSTM yielded greater than 95% accuracy and BOW n-grams + TDF with SVM model gave around 92% accuracy. BOW with Boosting yielded around 80%+ accuracy. Because review classes were evenly distributed, accuracy was chosen as a performance criterion for this categorization assignment. As a metric, the average F-1 score of the positive and negative groups was used to combine the precision and recall of the two class labels. The test set was used to evaluate all models. In the summary table below, best score is highlighted with blue and worst score with red color.

|  |  |
| --- | --- |
| Traditional Models Results  Table  Description automatically generated | Deep Learning Models Results  Table  Description automatically generated |

Figure 2.11 ﻿Traditional & Deep learning model results (Tamara and Milićević, 2018)

(Dey et al., 2020) presented comparative study of couple of machine learning algorithms on amazon product reviews. They used TF-IDF, frequent noun identifier and relevant noun remover for feature extraction. After that they used NB and SVM as machine learning approaches. Their research showed that SVM works better on amazon product reviews with high accuracy rate. They used precision, recall and F1-Score as well to measure system performance.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Result  Table  Description automatically generated  Table  Description automatically generated |

Figure 2.12 ﻿Flow Diagram of the System Architecture (Dey et al., 2020)

(Bansal and Srivastava, 2018) in their research classified user reviews using CBOW and skip-gram techniques using various machine learning algorithms like SVM, LR, NB, and Random Forest using k-cross validation technique. They concluded that Random Forest with CBOW gets the highest accuracy amongst all. They used dataset consisting of 4 Lacs mobile phone reviews on amazon for different brands. They used spacy for data pre-processing and performed stemming, stop word removal, removal of punctuations etc. Accuracy was used as metric to evaluate models. Below is a summary table showing accuracy of different models with mean and 95% confidence interval of 10-fold cross validation.

|  |
| --- |
| Table  Description automatically generated |

Figure 2.13 ﻿Accuracy of traditional ML models (Bansal and Srivastava, 2018)

(Bayhaqy et al., 2018) in their research did comparative study of 3 machine learning algorithms Decision Tree, KNN and Naïve Bayes on the dataset made up of tweets about e-commerce. During pre-processing stage, they applied cleansing of texts, convert to negation, convert emoticons, tokenization, filtering words that appear in large number but have no meaning, stemming etc and they concluded that Naïve Bayes performed better overall.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Result  Table  Description automatically generated |

Figure 2.14 Classification System Diagram for Sentiment Analysis (Bayhaqy et al., 2018)

(Mohbey, 2021) in his research applied LSTM on amazon product review dataset from Kaggle containing around 1.6 Lacs of records. and got around 93.66% accuracy. Furthermore, paper presented comparative study of LSTM with other machine learning models. Word normalization, stop word removal, tokenization and vectorization were used to pre-process data. Precision, F1-score, AUC, accuracy, recall metrics were used to access the performance of the models. For LSTM model hyper parameters like Batch size, dimensions, Activation function, optimizer, learning rate, number of epochs and dropouts were tuned. Result showed that LSTM depends only on pre-trained word vector and achieves 93% accuracy with 95% F1-score and AUC of 97% that is significantly above all other models in the comparison. Result also showed that SVM and NB do not perform well on massive dataset and takes long time to train.

Table

Description automatically generated

Figure 2.15 Comparison of ML model performance (Mohbey, 2021)

(Tammina and Annareddy, 2020) in their research used Amazon product reviews and IMDB movie reviews dataset and applied deep CNN model and compared its performance with traditional machine learning models like SVM, Random Forest and NB. IMDB dataset had 50k records and amazon dataset had 34k records. As a pre-processing step stemming, lemmatization and punctuation removal were done. Accuracy was used as a evaluation metric. To prevent overfitting dropout hyper parameter was used. Results suggested that CNN could efficiently extract information through feature maps and performed better than other models. It took overall 2 hours to train on both datasets with 8 GB RAM and Intel i7 processor.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.16 ﻿ Using word2vec and CNN to classify opinions (Tammina and Annareddy, 2020)

(Hossain et al., 2020) in their research applied SVM, KNN, NB and Gradient boosting classifier on amazon consumer reviews dataset. Dataset had around 24k reviews. Research suggested that NB gets the best accuracy score although other models were also very near in the performance. Accuracy, Precision, Recall, and F1-Measure were used as evaluation metrics. For gradient boosting they used week learners and parameters like min\_samples\_leaf, max\_depth, random\_state, n\_estimator, subsamples, learning\_rate etc.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.17 Proposed Model & Results (Hossain et al., 2020)

(Shehu et al., 2020) in their research proposed optimised LSTM approach using ﻿Pastoralist Optimization Algorithm for sentiment analysis. Amazon product review dataset was used in the research. POA-LSTM was then compared with LSTM for performance and results showed that POA-LSTM performs better for different population sizes. Accuracy, Precision, Recall and F1-score were used as evaluation metrics. As a part of pre-processing, elimination of URLs, special ciphers, break words were done along with tokenization.

|  |  |
| --- | --- |
| Methodology  A picture containing graphical user interface  Description automatically generated | Results  Chart, line chart  Description automatically generated |

Figure 2.18 Proposed Methodology and Results (Shehu et al., 2020)

**2.4 Sentiment Analysis in other domains.**

(Al-Saqqa and Awajan, 2019) in their survey found the in the word embedding techniques CBOW and skip-gram, Skip-gram works better for infrequent words than the other. And CBOW works well with frequent words and is faster compared to skip-gram. They used different datasets like Twitter dataset, Arabic Health Services dataset, Airline tweets, Kaggle competition dataset etc in English, Chinese and Arabic languages.

|  |
| --- |
| Architecture of CBOW and skip-gram  Diagram  Description automatically generated |

Figure 2.19 ﻿The architecture of CBOW and Skip-gram algorithms (Al-Saqqa and Awajan, 2019)

(Singh et al., 2021) in their study applied deep learning models such as simple neural network, CNN and two RNNs like LSTM and GRU on web crawled customer reviews and reported that RNN outperforms simple NN and CNN models. They also plotted various charts measuring loss vs epochs and noticed that simple NN overfits. And other RNN and CNN models do well in comparison. Research also noted that GRU takes lesser time than LSTM to train. Overall GRU yielded the best accuracy score of 85.84%.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.20 Process Flow diagram (Singh et al., 2021)

(Dhola and Saradva, 2021) compared several machine learning as well as deep learning models on twitter dataset with 1.6 Million of tweets for sentiment classification. Dataset had equal number of positive and negative tweets. Tokenization and stemming were done as pre-processing and SVM, Multinomial NB, LSTM and BERT were applied as machine learning and deep learning models. And result showed that deep learning models like BERT and LSTM performed better. They used precision, recall, F1-score, and accuracy metrics for evaluation. Below is a summary table. Right side of the table shows the comparison of accuracy metric.

|  |  |
| --- | --- |
| Table  Description automatically generated | Chart, bar chart  Description automatically generated |

Figure 2.21 Comparison of ML and DL model performance (Dhola and Saradva, 2021)

(Seo et al., 2020) presented comparative study of different deep learning models like CNN and RNN on 13 different datasets. Performances of these models were compared based on word level input and character level input and metric used was Area under ROC. Result showed that sentiment classification performance with the size of dataset irrespective of the model used. Word level input always yielded better performance for both models. They also noted that increasing model complexity is not always giving better results for CNN models but for RNNs. Employing LSTM/GRU unit helped in getting better performance. Overall best model was bidirectional LSTM with word level inputs. Following is the summary of the performances of the models and best scores are highlighted in bold letters.

|  |  |
| --- | --- |
| Architecture of RNN based models  Diagram  Description automatically generated  Architecture word level input  Diagram  Description automatically generated  Architecture character level input  Chart, diagram  Description automatically generated | Results  Table  Description automatically generated  Table  Description automatically generated |

Figure 2.22 Architecture of RNN based model (Seo et al., 2020)

(Zhang and Rao, 2020) in their research proposed to apply BERT output containing rich contextual information as an input of Deep neural network and then used this neural network to classify aspect-level sentiments. They did comparative study of this model with other models on three public datasets and concluded that BERT embeddings as an input to Deep neural network performed better than other baseline models by almost 2%. They used dropouts, batch size, learning rate, max epoch, max sequence length, optimizer as hyper parameters. Below is a summary of comparison between different models on different dataset.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results    Table  Description automatically generated |

Figure 2.23 ﻿proposed aspect-level sentiment classification model (Zhang and Rao, 2020)

# (Devlin et al., 2019) in their research fine-tuned BERT model on labelled sentiment dataset and got accuracy of 87.3% on data from stockwits.com. Dataset contained around 14 million records and BERT base pre-trained model was used with 12 layers, 768 hidden and 12 heads after fine tuning. To overcome catastrophic forgetting problem, they used hyper parameters like learning rate, batch size, optimizer, and number of epochs. Below table shows evaluation metrics for the proposed model.

Table

Description automatically generated

Figure 2.24 Results of fine-tuned BERT for sentiment analysis (Devlin et al., 2019)

(Zahoor et al., 2020) in their research performed sentiment analysis activity on restaurant reviews. Dataset used in this research had around 4000 records and several machine learning algorithms were applied on this dataset such as NB, SVM, LR and Random Forest. Performance comparison of these models showed that Random Forest gives the best accuracy score of 95%. Precision, Recall, F1 score and Accuracy were used as evaluation metrics.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.25 ﻿Architecture diagram of our research framework (Zahoor et al., 2020)

(Tiwari et al., 2020) in their research study performed sentiment analysis on the twitter data with different traditional machine learning models like DT, Random Forest and SVM and reported that Random Forest yields best accuracy score. Performance of Decision Tree was also very comparable to Random Forest.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.26 Architecture for sentiment analysis (Tiwari et al., 2020)

(Mengistie and Kumar, 2021) collected about 40000 tweets about COVID-19 from twitter, used FastText and GloVe pre-trained models for sentiment classification task. They trained hybrid model CNN-Bi-LSTM with both pre-trained models with FastText and GloVe and got 99.33% and 97.55% accuracy respectively. Precision, recall, accuracy and F1 score were used as evaluation metrics. Result showed that FastText model performed better than GloVe model because FastText contains more features

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Result  Table  Description automatically generated  Chart, table, bar chart  Description automatically generated |

Figure 2.27 proposed architecture with CNN Bi-LSTM (Mengistie and Kumar, 2021)

(Pradha et al., 2019) in their research compared pre-processing techniques like stemming, spelling correction and lemmatization to obtain the efficient method. 1.3 Million records were used as dataset retrieved from twitter. Dataset contained reviews of two popular products Amazon Alexa and Google Now. SVM, NB and Deep Learning models were trained on this dataset and result showed that SVM performs better than other models mainly because data was unstructured, and paper claimed that data pre-processing plays very important role in prediction accuracy and computational time. Computational speed was also compared for these models in the results section. SVM took the least time to train amongst all compared. NB took more time compared to others. Computational time of the pre-processing steps were also compared and found that stemming took the least time and spelling correction took the most time.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated  Technique speed Results  Table  Description automatically generated |

Figure 2.28 ﻿Overall selection process of the best performing pre-processing techniques (Pradha et al., 2019)

(Indulkar and Patil, 2020) in their research applied two multi-layer perceptron-based models namely deep feed forward neural network and Convolutional Neural network on Uber and Ola dataset extracted through tweets from twitter. Google’s word2Vec was used to generate word embeddings. Accuracy was chosen as an evaluation metric. Deep NN with 2 layers yielded the best accuracy. Also, it was reported that same model did not perform well for Ola dataset because tweets extracted from ola did not have proper formatting although cleaning was done.

CNN model is used typically in image processing but still it was used in the experiment here to see how well it does on text processing.

|  |  |
| --- | --- |
| Methodology  Deep Feed Forward Neural Network  Diagram  Description automatically generated | Methodology  CNN  Diagram  Description automatically generated |
| Accuracy of Deep Feed Forward NN  Table  Description automatically generated | Accuracy of CNN  Table  Description automatically generated |

Figure 2.29 Architecture of DNN and CNN (Indulkar and Patil, 2020)

(Salur and Aydin, 2020) in their research applied hybrid model that combined word embedding techniques like Word2Vec, FastText and character-level embedding with different deep learning-based models like GRU, CNN, LSTM, BiLSTM and compared their performances. They proposed a model which extracted features of different deep learning methods of word embeddings and combines these features and classifies text for a particular sentiment. Dataset used was made up of Turkish tweets about GSM operator in Turkey. They achieved best classification accuracy of 82.14% using proposed approach of combining CNN+BiLSTM with FastText and character-level embedding.

|  |  |
| --- | --- |
| Methodology  Diagram  Description automatically generated | Results  Table  Description automatically generated |

Figure 2.30 ﻿Architecture of the proposed hybrid deep learning model (Salur and Aydin, 2020)

# 2.5 Discussion

The objective behind study of existing research is to understand different methods that are already applied in sentiment classification problem space and based on that find out a different approach to solve sentiment classification problem on IMDB movie review dataset.

A lot of research has been done using traditional machine learning models like SVM, NB, Random Forest, Decision Tree, Logistic Regression, Maximum Entropy etc. as well as deep learning models like CNN, RNN, BERT, XLNET etc. with word embeddings as TF-IDF, Word2Vec, Doc2Vec, Glove etc. But limited research is done with BERT embedding as input to traditional machine learning models and hence in this research we are going to focus on applying BERT embeddings to some traditional machine learning models like LR, SVM, NB, DT, Random Forest etc and compare their performances based on Accuracy, AUC, Precision, Recall, F1-score etc.

Few researchers have applied state of the art and transformer models like BERT, XLNet, RoBERTa etc. on product reviews sentiment classification domain. So, it makes sense to compare the performance of traditional machine learning models with one or more transformer model.

# 2.6 Summary

This chapter provides an overview of previous work done in sentiment classification domain using various pre-processing steps, vectorization techniques, traditional and deep learning model buildings etc. We have tried covering sentiment analysis research work done in movie reviews domain, product reviews domain and in other domains. Literature review also covered papers that compared machine learning models and evaluated their performances based on accuracy, precision, recall, f1-score etc. And later we discussed the motivation for this research.

# CHAPTER 3

# RESEARCH METHODOLOGY

# 3.1 Introduction

In the recent years there has been a huge leap forward in the natural language processing space with the introduction of transformers. Transformers provide thousands of pre-trained models to perform on text processing tasks like summarization, text generation, extraction of information, question-answering etc. BERT, RoBERTa, XLNET etc are some examples of transformers. Transformers provide user friendly APIs to quickly download pre-trained models and fine tune them based on the dataset. In the real-world results transformers have outperformed all traditional and deep learning models like RNN and CNN. Therefore, in this research we are focusing on extracting rich contextual information from transformer model BERT and then applying this embedding with some traditional machine learning algorithms as well as State of the Art Transformer models like BERT on IMDB movie reviews dataset and compare their performances.

BERT embedding captures the context of the word also that contrasts with other context free embedding techniques like TF-IDF, Word2Vec, Doc2Vec etc. These contextual embeddings achieve state of the art results in natural language processing tasks like sentiment analysis. So here in this research we are going to generate word vector from BERT embeddings and apply it on traditional machine learning models like Logistic Regression, SVM, NB, Random Forest and compare their performance with same models built using TF-IDF and Word2Vec Embeddings.

# 3.2 Methodology

# 3.2.1 Data Selection

This dataset consists of IMDB movie reviews taken directly from Kaggle Website. Dataset has 2 columns Review and Sentiment. Review column represents user reviews some of the are large comments and some of them are small. Sentiment column has two labels “Positive” and “Negative”, Overall, there are 50000 records in the dataset and classes of positive and negative reviews are balanced that is 25000 each. (IMDB Dataset of 50K Movie Reviews | Kaggle, 2021)

Table 3.1: Dataset columns overview

|  |  |
| --- | --- |
| Variable | Definition |
| Review | User’s review statements are given as review column. |
| Sentiment | Classification of sentiment, i.e., positive, negative |

# 3.2.2 Data Pre-Processing

There is an approach called Exploratory Data Analysis (EDA) that helps extracting the information enfolded in the data and based on that we can summarise the main attributes or characteristics of the data that is more important to study for a given problem.

1. Stop Word Removal - Stop words refer to the most common words in a language and they don’t add much value to the overall meaning of a sentence. So, they can be removed to reduce the overhead of processing these words.
2. Tokenization – It refers to breaking a piece of text into smaller units called tokens.
3. Remove HTML content – All HTML tags that are present in the reviews add no meaning to the sentences, so they all need to be removed. hyperlinks also need to be removed.
4. Substituting multiple spaces with single space.
5. Remove words containing numbers.
6. Converting to lowercase.
7. Lemmatization/Stemming - Stemming is the process of transforming the word into its base word by cutting off common prefixes or suffixes. e.g., studies and studying will be transformed to studi. Lemmatization is the process of transforming the word into its dictionary representation. e.g., studies and studying will be transformed to study.
8. Feature Extractions using different embedding techniques like TF-IDF, Word2Vec Embeddings and BERT Embeddings.

# 3.2.3 Exploratory Data Analysis

EDA will help identifying characteristics of dataset, based on EDA we can see need of lemmatization vs stemming etc. It will also help in taking right approach for research methodology. Following data will be analysed during EDA.

1. Distribution of positive and negative sentiments.
2. Missing Values analysis.
3. Unique values of rows in the dataset.
4. Effect of sentence length on label to be predicted.
5. Top Unigram, Bigrams and Trigrams for positive and negative reviews.
6. Word clouds for top Unigram, Bigrams, Trigrams for positive and negative reviews.
7. Automatic EDA report through pandas-profiling library.

# 3.3 Proposed Method

In this research traditional and transformer-based models will be built with 3 different embeddings. Dataset is divided in 70:30 ratio for train and test set. Also, K-fold cross validation strategy is used for training models.

**3.3.1 Embeddings Considered**

Following are the embedding techniques considered for this research.

* TF-IDF
* Word2Vec
* BERT Embeddings

**Why These 3 Embedding techniques only?**

We saw during literature review that, A lot of research already has been done using TF-IDF and Word2Vec embedding techniques with traditional machine learning models. But a very limited research is done with BERT embeddings with Traditional machine learning models. TF-IDF and Word2Vec are context free embedding techniques. They ignore the meaning of the word in different contexts as it creates only one representation of a given word in entire vocabulary. In contrast to that, contextual embedding technique like BERT learns different representations of a given word based on their context. So, applying these three techniques on same traditional machine learning models would help us learning which embedding technique works better as well as we can compare our model’s performance with other base papers that have experimented with TF-IDF and Word2Vec embeddings already on IMDB dataset. (Tripathi et al., 2020)

Below is a brief understanding around these embedding techniques.

**TF-IDF:**

Assume that we have a problem at hand where we want to figure out a which document is the most relevant for the review statement or comment that we want to search in these documents. So one naïve solution for this problem is to simply gather words of the review statement or comment and figure out how many documents do not include all the words of the review statement or comment and simply eliminate them. But since its very naïve approach intuitively we can think of that still a lot of documents will still remain relevant for the searching operation. So the next addition we can do in filtering is to count frequency of each term in the document. This is called Term Frequency.

|  |  |
| --- | --- |
| A picture containing letter  Description automatically generated | (1) |

Document frequency is similar to term frequency, It represents the count of a particular term in the set of document. In other words, It is the number of documents where the given term is present.

|  |  |
| --- | --- |
| Text  Description automatically generated | (2) |

In the term frequency, all the terms are given equal importance or weightage. But in practical scenarios some words occur too frequently like “is”, “are”, “of” etc. and that has very less to no importance, hence we need to rationalise or weigh down high frequency terms and weigh up the low frequency terms, IDF is an factor which helps achieving this by lowering weightage of high frequency terms and increasing weightage for low frequency terms.

|  |  |
| --- | --- |
| Logo  Description automatically generated | (3) |

Here N is total number of documents and DF is number of documents containing given term.

|  |
| --- |
| Table  Description automatically generated |

Figure 3.1: TF-IDF Representation (BoW Model and TF-IDF For Creating Feature From Text, 2021)

**Word2Vec Embeddings:**

Word vector helps representing the words as vectors. So you can vectorize these words and match these vectors to find similarities, But this sounds impractical if you have millions of words. So we might need to find similarities between these encoded vectors.

Word2Vec is very well known predictive model that strives to output a target word provided a context or the context words from the target. It’s a vector representation of words such that the cosine distance between those vectors reflects the semantic “distance” between the words they represent. For example if word1 and word2 have a similar meaning, then cos distance of word1 vector and word2 vector should be small.

Word2Vec has two implementations 1) Skip-gram 2) CBOW. In the skip-gram model, prediction of context is done based on the representation of the input word. And in the CBOW model, prediction of the word in the middle is done based on surrounding words.

(NLP 101: Word2Vec — Skip-gram and CBOW | by Ria Kulshrestha | Towards Data Science, 2021)

Word embeddings help achieving dimensionality reduction so number of feature required to build the model is less. Also it captures the meanings of the words and semantic relationships.

|  |
| --- |
| Diagram  Description automatically generated |

Figure 3.2: Skip Gram Representation (NLP 101: Word2Vec — Skip-gram and CBOW | by Ria Kulshrestha | Towards Data Science, 2021)

**BERT Embeddings:**

BERT Embeddings comprises of 3 different embeddings.

1. Position Embeddings are used to understand the location index information of the input. This can overcome the shortcomings of RNN which doesn’t keep location information.
2. Segment Embeddings are unique embeddings for different sentences that is learnt by BERT. In these embedding different sentences are given different embeddings.
3. Token Embeddings are unique embeddings in which BERT learn specific token from the word token vocabulary.

|  |
| --- |
| A screenshot of a game  Description automatically generated with medium confidence |

Figure 3.3: BERT embeddings (What is BERT | BERT For Text Classification, 2021)

# 3.3.2 Traditional machine learning models

Following traditional machine learning models will be built with all three embedding techniques mentioned in section 3.3.1. Also training will be done using K-fold cross validation technique.

* Logistic Regression
* SVM
* Naive Bayes
* Decision Tree
* Random Forest

**Why only these models?**

Based on literature review, Lot of research has already been done with above traditional machine learning models with TF-IDF and Word2Vec embeddings on IMDB dataset. So in order to compare performance of these models with relatively recent embedding technique like BERT we would train all these traditional machine learning model with TF-IDF and Word2Vec embeddings and BERT embeddings and then compare the performance of these models to infer which embedding technique works better.

**Why BERT embeddings only?**

BERT is as a deeply bi-directional model by design. That means it captures the meaning of the word from both left to right and right to left contexts from the first layer itself to the last layer. Because of this BERT generates embeddings that has more than 1 vectors for a given word based on its context and hence it can handle the issue of ambiguous words with more than one meanings based on context successfully.

# 3.3.3 Transformer based model - BERT

BERT has been released with two versions, BERT-Base and BERT-Large. They have 12 layers with 110M parameters and 24 layers 340M parameters respectively. BERT is pre-trained using large corpus of Toronto book corpus and Wikipedia. BERT is trained using two objectives.

* Some words from the input are masked and model learns to predict these words. This is called Masked Language Model. In this, from each sequence 15% of the tokens are processed with 80% probability of token being replaced with MASK, 10% probability of token being replaced by random token and 10% probability of the token is unchanged.
* Two sequences are fed as input and the model is trained to predict if one sequence follows the second or not. This is called Next sequence Prediction. In this two sentences X and Y are separated in a such a way that 50% of time Y would be actual next statement and other 50% of time it would be a random statement. (A Light Introduction to BERT. Pre-training of Deep Bidirectional… | by constanza fierro | dair.ai | Medium, 2021)

To identify between two sentences following pre-processing steps are done before feeding into model.

1. CLS and SEP tokens are inserted at the beginning and end of the statement.
2. Sentence embedding that can indicate if it’s the same statement or different statement.
3. Positional embedding that can indicate the index or position in the statement.

**Why BERT model?**

BERT is a deeply bi-directional model and it has achieved state-of-the-art results in NLP tasks such as sentiment classifications, question answering etc. Before transformers got the attention for NLP tasks, RNN and CNN were used to get state-of-the-art results. But RNN is not efficient for long sequences as it tend to forget the contents or mixes the contents in some cases. Also sequential nature of execution makes it more time consuming to train as it’s hard to leverage GPUs or TPUs. LSTM offers advantage over RNN by adding a GATE mechanism to decide which information the cell should remember and which it should forget. But still LSTM can’t be trained in parallel. That’s a major drawback. CNN has an advantage over RNN as it can be trained in parallel but capturing dependencies for long sequence can be pretty complex and impractical. Transformers solves these problem as it can learn long term dependencies very well and it can be parallelised to so it reduces time to train models.

(BERT Explained: State of the art language model for NLP | by Rani Horev | Towards Data Science, 2021)

# 3.4 Proposed Flow Diagram

|  |
| --- |
| Diagram  Description automatically generated |

Figure 3.4: Proposed Methodology Flow Diagram

# 3.5 Evaluation Metric Used

Confusion matrix will be created to evaluate model’s performance because they allow us to derive metrics like True Positives, True Negatives, False Positives, and False Negatives easily.

It allows comparing expected values with actual values and that’s the reason it will help in this sentiment classification task. Since sentiment classification is a binary classification problem, we are going to use AUC (Area Under Curve) – ROC (Receiver Operating Characteristics) as a metric to evaluate various model performances. By calculating AUC-ROC, Goodness of the model can be determined. If the curve is sticky towards upper left side, it means that the model is very good and if it is more towards the 45-degree diagonal, it means that the model is almost completely random. So, if AUC is larger if model is good.

1. Accuracy - It’s a ratio of accurately predicted reviews to total number of reviews.

|  |  |
| --- | --- |
|  | (4) |

1. Precision – It’s a ratio of reviews classified accurately for a given sentiment to the total count of reviews classified as for that sentiment.

|  |  |
| --- | --- |
| Text  Description automatically generated with low confidence | (5) |

1. Recall - It is opposite of precision, it measures false negatives against true positives.

|  |  |
| --- | --- |
| A picture containing text  Description automatically generated | (6) |

1. F1-Score – It is the harmonic mean of precision and recall.

|  |  |
| --- | --- |
| A picture containing graphical user interface  Description automatically generated | (7) |

1. TPR – It’s a ratio of accurately predicted sentiments to the sum of

|  |  |
| --- | --- |
|  | (8) |

1. FPR – it measures the frequency at which model predicts Positive where a Negative is observed.

|  |  |
| --- | --- |
|  | (9) |

Macro Average and Weighted average for precision, recall and F1-score will also be calculated. In that macro average is exactly same as usual average that we observe and weighted average considers how many of each class were present in its calculation, this is particularly helpful when dataset is imbalance. So if one of the predicted class is fewer in dataset, it would have less impact on precision, recall and f1-score.

# 3.6 Expected Outcomes

The research is targeted to get better accuracy in sentiment classification task using contextual word embedding techniques like BERT embeddings. Logistic Regression, Decision Tree, SVM , NB and Random Forest model will be built with TF-IDF, Word2Vec and BERT embeddings and we would compare results of these embedding techniques on these models. Then Transformer model BERT will be built with BERT embeddings and it will be compared with traditional machine learning models. As a outcome we would get the best model for this sentiment classification task that can also be leveraged for other domains like product reviews and other similar domains.

# 3.7 Required Resources

This research needs experiments with traditional models and transformer model So, Google Colab GPU and Personal Computer will be used.

**3.7.1 Hardware Resources**

Following hardware resources will be required.

* Operating System: MacOS BigSur.
* Processor: 2.2 GHz 6-Core Intel Core i7
* Memory: 16 GB

**3.7.2 Software Resources**

Following software resources will be required.

* Python Version - 3.3
* Anaconda Package – It provides pre-defined libraries.
* Jupyter Notebook - 2.2.6 – It provides web based interactive development environment. It allows creation and sharing of documents that contain code, visualizations, narrative text etc.
* Google Colaboratory - This is web-based interface that allows executing code on the cloud. It’s helpful in training machine learning models as it offers GPU and TPU for processing for educational purposes.

**3.7.3 Additional Resources**

* UpGrad team has provided GPU enabled Nimble box for processing needs. So that will be used for building transformer models.

# 3.8 Risk and Contingency Plan

Following are the risks identified and potential plan of mitigation.

Table 3.2 Risk Mitigation plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Description | Probability of  Occurrence | Loss Size  (Days) | Risk Exposure  (Days) | Mitigation Plan |
| Software Resources | 25% | 8 | 2 | Will build models on personal computer. |
| Covid Impact | 15% | 10 | 2 | Might need to optimise time by reducing number of experiments on models |
| Data Quality issues | 15% | 5 | 1 | Cleaning data and more pre-processing. |
| Model Building might take more time than anticipated because of unknown issues, need extra reading etc. | 30% | 10 | 3 | Will ask for help from Supervisor, professor |

Total Risk exposure is around 8 days that is not very difficult to mitigate.

# 3.9 Summary

This chapter detailed out the proposed research methodology to perform sentiment analysis and classification on IMDB movie review dataset. As a summary, Dataset will be downloaded from Kaggle, After EDA and pre-processing steps, three different word embeddings will be created Word2Vec, TF-IDF and BERT embeddings and then traditional machine learning models will be created, and performance of these models will be compared. In the second phase, Transformer model BERT will be created, and its performance will be compared with other trained models.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 Introduction**

This chapter covers exploratory data analysis on the IMDB movie reviews dataset. (IMDB Dataset of 50K Movie Reviews | Kaggle, 2021). In the EDA, distribution of positive and negative sentiments, effect of text length on target class labels, Top Unigrams, Bigrams and Trigrams are derived. Word clouds are also built as part of EDA that graphically represents word frequency, so it captures the words that appears more frequently in the given text. Data pre-processing and feature engineering are carried out and post that TF-IDF, Word2Vec and BERT embeddings are created. Machine learning models like Logistic Regression, Random Forrest, SVM, Naïve Bayes, Decision Tree are created with all these 3 embedding techniques and BERT transformer model is also built post pre-processing and tokenization. Various hyper parameters are used for traditional and transformer models.

**4.2 Exploratory Data Analysis**

EDA is a very useful method to perform investigation on the dataset so that interesting features and insights can be derived from the dataset. It also helps in finding anomalies and testing hypothesis or assumptions. There are different ways to get insights into data. Here for this study we have considered bar chart, box plots, Histograms to visualise the data. Python programming language has few libraries that helps in EDA process and creates different visualisations out of the box. One of such library named pandas profiling is used here in this research.

**4.2.1 Distribution of the Sentiments**

Distribution of the sentiments plays very important role in deciding right pre-processing technique. If the distribution of sentiments are not balanced, Machine learning models don’t give equal priority to each class. So in this research we have used bar plot to visualise distribution of the sentiments. There are total 50,000 rows in the dataset with two target labels as positive and negative. Around 842 records are duplicate entries so after clean-up of these records we are left with 49158 unique records.

|  |
| --- |
| Chart, bar chart  Description automatically generated |

Figure 4.1 Distribution of the sentiments.

From the above figure it’s clear that dataset is balanced. Both positive and negative sentiments are having frequency of 24764 and 24394 records respectively.

**4.2.2 Text Statistics**

This section explores fundamental characteristics of the text data which can be done through word frequency or sentence length analysis.

**4.2.2.1 Word Frequency Analysis**

Frequency distribution of number of words can play a significant role in the feature engineering, hence we have counted the number of words per sentiment class and compared the numbers. Most of the positive sentiments are found in reviews having word count less than 400, and frequency of positive sentiments is higher when word numbers are between 20 and 140.

|  |
| --- |
| **Chart  Description automatically generated with medium confidence** |

Figure 4.2 Distribution of the word count for the reviews with positive sentiments.

Most of the negative sentiments are found in reviews having word count less than 400, and frequency of negative sentiments is higher when word numbers are between 50 and 120. This range is overlapped with positive sentiment as well but the number of positive sentiments in this range is more than 2000 on average whereas negatives sentiments are below 2000 on average.

|  |
| --- |
| **Chart  Description automatically generated** |

Figure 4.3 Distribution of the word count for the reviews with negative sentiments.

**4.2.2.2 Sentence Length Analysis**

From the below figures, we can say that positive and negative reviews are having almost overlapping range in terms of sentence length. Both positive and negative classes have the greatest number of reviews in the range of 50 to 1000 characters. Below figure illustrates sentence length distribution of reviews with positive sentiments.

|  |
| --- |
| A picture containing chart  Description automatically generated |

Figure 4.4 Distribution of the sentence length for the reviews with positive sentiments.

Below figure illustrates sentence length distribution of reviews with Negative sentiments. From this figure it’s evident that there is no clear distinct range in terms of number of characters for positive and negative sentiments.

|  |
| --- |
| Chart  Description automatically generated with medium confidence |

Figure 4.5 Distribution of the sentence length for the reviews with negative sentiments.

**4.2.2.3 Stop words distribution**

Below figures illustrates the distribution of valid word counts that doesn’t include stop words for the reviews with positive sentiments. From the figure we can notice that when such words are having frequency in the range of 40 to 120. Number of positive sentiments are higher.

|  |
| --- |
| **Chart  Description automatically generated with medium confidence** |

Figure 4.6 Stop words distribution for positive sentiments

Below figure illustrates valid words which doesn’t include stop words for negative sentiments. From figure we can notice that frequency of negative sentiments is higher (>2000) when such words have frequency in the range of 50 to 80. This range is a sub range we noticed where positive sentiments were higher. But if we compare the number of counts in the same range there are more positive sentiments than the negative sentiments in this range.

|  |
| --- |
| **Chart  Description automatically generated** |

Figure 4.7 Stop words distribution for negative sentiments

**4.2.3 Top N-Gram Analysis**

N-Gram model is a probabilistic language model. It is used in NLP field very extensively. Basic objective of using N-Gram is to capture the language structure from the statistical point of view. Here N represents the number of words used as observations. So, we can get more context with higher N numbers. Unigram, Bigram and Trigram are the most used N-gram models. Unigram represents single word, Bigram represents two words, Trigram represents three words and so on. Below figure shows one example to illustrate how different N-grams like unigram, bigram, trigram etc. can be generated from a given statement.

|  |
| --- |
| **Graphical user interface, website  Description automatically generated** |

Figure 4.8 Illustration of different N-gram windows for the same sentence (Generate Unigrams Bigrams Trigrams Ngrams Etc In Python | Arshad Mehmood, 2021)

To build these different N-gram we have used CountVectorizer class of scikit learn library in python.

**4.2.3.1 Top Unigrams Analysis**

Below Figure illustrates top positive unigrams present in the dataset. Like, Good, Great, Character, Scene, Story etc. keywords have frequency of around 10000 and shows positive sentiments.

|  |
| --- |
| **Chart, bar chart  Description automatically generated** |

Figure 4.9 unigram for the reviews with positive sentiments.

Below figure illustrates top negative unigrams present in the dataset. Bad, character, scene, story etc. are few of the top unigrams that people use in reviews to express negative sentiments as well. Since these words are common in positive and negative reviews, it would make sense to explore bigrams, so that more context around the words can get captured.

|  |
| --- |
| **Chart, bar chart, funnel chart  Description automatically generated** |

Figure 4.10 unigram for the reviews with negative sentiments.

**4.2.3.2 Top Bigrams Analysis**

Below figure illustrates top positive bigrams present in the dataset. From these bigrams we can notice that good movie, great movie, special effects, main character etc are some of the positive bigrams.

|  |
| --- |
| **Chart  Description automatically generated** |

Figure 4.11 Bigram for the reviews with positive sentiments.

Below figure illustrates top negative bigrams. Low budget, Main character etc seem to be mentioned by users in negative reviews with frequency above 1000.

|  |
| --- |
| **Chart, funnel chart  Description automatically generated** |

Figure 4.12 Bigram for the reviews with negative sentiments.

**4.2.3.3 Top Trigrams Analysis**

From the below figure we can see that people liked movies based on true story, movies around second world war etc. Also, from trigram it appears, many people recommend the movie to other if they really like it.

|  |
| --- |
| Chart, funnel chart  Description automatically generated |

Figure 4.13 Trigram for the reviews with positive sentiments.

As per below figure, Top negative trigrams contain keywords like worst movie seen, bad acting, low budget movie, waste time money etc.

|  |
| --- |
| Chart, bar chart, funnel chart  Description automatically generated |

Figure 4.14 Trigram for the reviews with negative sentiments.

**4.2.4 Word Clouds**

Word clouds represent the graphical representation of the word with high frequency. The larger the word is in the picture, the more frequent the word is in the data set. This kind of visualization helps in analysing text by highlighting terms that are often found in a series of reviews, documents, or other text. It can also be used in reporting to communicate the key points or topics of the words that could affect the polarity of the sentiment. From below figure it seems that special effect is present in both positive and negative sentiments, that means if special effects are good people likes movie, otherwise they dislike the movie. New York is the most common word in positive sentiments. People are not enjoying low budget movies much.

|  |  |
| --- | --- |
| **Positive**  **Text  Description automatically generated** | **Negative**  **Text  Description automatically generated** |

Figure 4.15 Word Clouds for positive and negative sentiments

**4.3 Data Preparation and Pre-processing**

This stage includes analysing dataset attributes and make necessary changes to the dataset before applying machine learning models. These changes may include removing/treating outliers, removing duplicates, treating missing values, drop columns or rows that are not relevant for the research etc. Here in this research, we have used the dataset for IMDB reviews, obtained from Kaggle (IMDB Dataset of 50K Movie Reviews | Kaggle, 2021). This dataset has two columns, one for reviews and one for sentiment labels.

**4.3.1 Removing Duplicate data**

There are 50,000 rows in the IMDB movie review dataset obtained from Kaggle. Upon analysis we found that 824 records were the duplicates. So, after removing these rows dataset was left with 49176 records, containing 24,774 records for positive sentiments and 24,402 records for negative sentiments. Pandas library of python was used for dropping duplicate records.

**4.3.2 Text Data Pre-processing**

Pre-processing stages for text data would include cleansing and converting raw data to structured data, so that the data is suitable for feature extraction techniques and contains more information, which could aid in the development of high-accuracy and high-quality models.

Some of the review comments included HTML tags, hyperlinks, Special characters, numbers, and inconsistent punctuations etc. So following steps were performed.

1. Remove duplicates
2. Remove HTML tags
3. Remove all single characters
4. Substituting multiple spaces with single space
5. Remove hyperlinks
6. Remove numbers
7. Converting to lowercase
8. Removing stop words
9. Lemmatization
10. Tokenization

NLTK library was used for the tokenization and Lemmatization. Regular expressions were used for removing HTML tags, single characters, hyperlinks etc.

**4.3.2 Pre-processing for transformer-based BERT model.**

We have transformed sentiment column values to numbers like 0 and 1 for positive and negative sentiments as transformer model requires label text in numerical format. Below table captures the sentiment label and corresponding numerical value.

Table 4.1 Sentiment Labels Transformation

|  |  |
| --- | --- |
| Sentiment Label | Corresponding Number Format |
| Positive | 0 |
| Negative | 1 |

Apart from this, pre-trained BERT model expects data in the specific format. So, we would need additional things as mentioned below.

1. Special token like CLS, that is used to mark beginning of the text and other special token like SEP, that is used to mark the end of a sentence of separation between two sentences.
2. Token Id for the tokens from BERT’s tokenizer.
3. Mask Ids to identify which elements in the sequence are tokens and which one are padding elements.
4. Segment IDs to distinguish different sentences.
5. Positional Embeddings to represent position of token within sequence.

Tokenizer.encode\_plus function of transformer interface takes care of all above steps. From the below figure, we can notice that majority of the reviews have length lesser than 500, So we have used 500 as maximum length to pad in the “encode\_plus” function.

Chart, histogram

Description automatically generated

Figure 4.16 Distribution of length of sentences

As an output of “encode\_plus” method, we got two outputs, attention masks and input ids, Both of these were converted to tensors. Input ids are the token ids which works as an input to the model. Attention masks is an array of indices that helps model to decide on which indices model should pay attention to. Input sequences are aided with 1 and padded sequences are labelled with 0, These masks help in differentiating between two.

Overall, We have divided dataset into 80:20 ratio for train and test set. And from training set we are using 10% as validation set that is nearly 4000 records.

**4.4 Feature Engineering**

In general, Feature engineering has following two goals.

* Preparing appropriate dataset that is compatible for the machine learning algorithm requirement.
* Aid in to improving performance of machine learning models.

In this research, We are going to create three vectorization techniques like TF-IDF, Word2Vec and BERT Embeddings as part of feature engineering. All embeddings are created with two variants, with stop words and without stop words to see what is the difference in the performance of the models between these two variants.

**4.4.1 Preparing TF-IDF**

TF-IDF suggests the importance of the word in the whole vocabulary corpus as well as in the document. TF represents term frequency in the document and IDF represents the frequency of a particular word in the whole vocabulary corpus. TfidVectorizer class of scikit learn library was used to generate TF-IDF vector representation of the dataset. Following table shows the parameters used to initialise this class.

Table 4.2 Parameters used with TF-IDF

|  |  |  |
| --- | --- | --- |
| Parameter Name | Parameter Value | Parameter Significance |
| n\_gram\_range | (1,3) | The lower and upper boundary of the range of n-values for different n-grams to be extracted. |
| max\_features | 20000 | If not None, build a vocabulary that only consider the top max\_features ordered by term frequency across the corpus. |

Before generating TF-IDF vector for the dataset, We have removed all words that occurs less than 5 times in the dataset as they may not add value in the overall output of the model and dimension space of the TF-IDF vector can also be reduced. Sample snippet is provided as below.

*vectorizer = TfidfVectorizer(ngram\_range=(1, 2), max\_features=20000)*

*tfidf\_with\_stop\_training\_features = vectorizer.fit\_transform(train\_data["clean\_text\_normalized\_wo\_stop"])*

*tfidf\_with\_stop\_test\_features = vectorizer.transform(test\_data["clean\_text\_normalized\_wo\_stop"])*

*tfidf\_no\_stop\_training\_features = vectorizer.fit\_transform(train\_data["clean\_text\_normalized"])*

*tfidf\_no\_stop\_test\_features = vectorizer.transform(test\_data["clean\_text\_normalized"])*

**4.4.2 Preparing Word2Vec Embeddings**

The basic idea of Word2Vec is words that occur in similar context tend to be closer to each other in vector space. Word2Vec utilizes two architectures, Continuous Bag of Words and Skip Gram. Gensim library was used to generate Word2Vec Embeddings. Following table shows the parameters used.

Table 4.3 Parameters used with Word2Vec

|  |  |  |
| --- | --- | --- |
| Parameter Name | Parameter Value | Parameter Significance |
| min\_count | 5 | Ignores all words with total frequency lower than this. |
| window | 5 | Maximum distance between the current and predicted word within a sentence. |
| size | 100 | Dimensionality of the word vectors |

As an output, we got vector of size 40000x100 for training set and 10000x100 for test set as the dataset was split into 80:20 ration for train and test data. Sample snippet is provided as below.

*def embeddToWord2Vec(text):*

*words = word\_tokenize(text)*

*if embedding is 'WORD2VEC\_WITH\_STOP':*

*result = [w2v\_with\_stop\_model.wv[w] for w in words if w in w2v\_with\_stop\_model.wv.vocab]*

*else:*

*result = [w2v\_no\_stop\_model.wv[w] for w in words if w in w2v\_no\_stop\_model.wv.vocab]*

*##* *we use mean operator on overall word vectors that are of fixed length and use just a vector to represent the review*

*feature = [mean(x) for x in zip(\*result)]*

*return feature*

*## apply word\_tokenize function from NTLK librar*

*words = train\_data['clean\_text'].apply(word\_tokenize)*

*w2v\_with\_stop\_model = gensim.models.Word2Vec(words, min\_count = 5 ,size=100, window = 5)*

*w2v\_no\_stop\_model = gensim.models.Word2Vec(words, min\_count = 5,size=100, window = 5)*

*## Get Word2Vec embeddings for data with stop words*

*embedding = 'WORD2VEC\_WITH\_STOP'*

*word2vec\_with\_stop\_training\_features = train\_data['clean\_text'].apply(embeddToWord2Vec)*

*word2vec\_with\_stop\_test\_features = test\_data['clean\_text'].apply(embeddToWord2Vec)*

*#Get Word2Vec Embeddings for data without stop words*

*embedding = 'WORD2VEC\_NO\_STOP'*

*word2vec\_no\_stop\_training\_features = train\_data['clean\_text\_w2v\_no\_stop'].apply(embeddToWord2Vec)*

*word2vec\_no\_stop\_test\_features = test\_data['clean\_text\_w2v\_no\_stop'].apply(embeddToWord2Vec)*

**4.4.3 Preparing BERT Embeddings**

BERT Embeddings are rich contextual embeddings that can capture the meaning of the words based on the context. BERT offers an advantage over other embedding techniques like Word2Vec because each word has a fixed representation under Word2Vec regardless of the context in which word appears. BERT creates representation of the words based on the words surrounding them. SentenceTransformer library was used to create sentence vectors. It maps sentences and paragraphs to a 768 dimensional dense vector space. We can pass a model name while initialising this class as mentioned in the snippet below. Below table describes the list of various pre-trained sentence embedding models.

Table 4.4 Pre-trained Sentence Embedding models (Pretrained Models — Sentence-Transformers documentation, 2021)

|  |  |  |
| --- | --- | --- |
| Model Name | Average Performance over all tasks | Speed (Encoding speed, sentences/sec on V100 GPU) |
| paraphrase-mpnet-base-v2 | 76.84 | 2800 |
| paraphrase-multilingual-mpnet-base-v2 | 75.39 | 2500 |
| paraphrase-TinyBERT-L6-v2 | 75.36 | 4500 |
| paraphrase-distilroberta-base-v2 | 75.15 | 4000 |
| paraphrase-MiniLM-L12-v2 | 74.81 | 7500 |
| paraphrase-MiniLM-L6-v2 | 74.38 | **14200** |
| paraphrase-albert-small-v2 | 73.94 | 5000 |
| paraphrase-multilingual-MiniLM-L12-v2 | 73.80 | 7500 |
| paraphrase-MiniLM-L3-v2 | 73.55 | 19000 |
| nli-mpnet-base-v2 | 72.45 | 2800 |
| stsb-mpnet-base-v2 | 72.12 | 2800 |
| distiluse-base-multilingual-cased-v1 | 70.43 | 4000 |
| stsb-distilroberta-base-v2 | 70.07 | 4000 |
| nli-roberta-base-v2 | 70.00 | 2300 |
| stsb-roberta-base-v2 | 69.89 | 2300 |
| nli-distilroberta-base-v2 | 69.86 | 4000 |
| distiluse-base-multilingual-cased-v2 | 69.59 | 4000 |
| average\_word\_embeddings\_komninos | 61.57 | 22000 |
| average\_word\_embeddings\_glove.6B.300d | 60.52 | 34000 |

For this research "paraphrase-MiniLM-L6-v2” model is used as it is the quickest high quality model available for sentence embeddings. Following table describes the parameters used for creating sentence embeddings.

Table 4.5 Parameters used with SentenceTransformer

|  |  |  |
| --- | --- | --- |
| Parameter Name | Parameter Value | Parameter Significance |
| model\_name\_or\_path | paraphrase-MiniLM-L6-v2 | If it is a file-path on disc, it loads the model from that path. If it is not a path, it first tries to download a pre-trained SentenceTransformer model. If that fails, tries to construct a model from Huggingface models repository with that name |
| device | gpu | Device (like ‘cuda’ / ‘cpu’) that should be used for computation. If None, checks if a GPU can be used |

Following is the snippet for creating BERT sentence embeddings. We are using “encode” method of SentenceTransformer class for encoding sentences and mean of the vector is calculated. Later this vectors were transposed and used as an array to feed into different models.

*bert\_transformers = SentenceTransformer(‘paraphrase-MiniLM-L6-v2’)*

*def embeddToBERT(text):*

*sentences = re.split('!|\?|\.',text)*

*sentences = list(filter(None, sentences))*

*result = bert\_transformers.encode(sentences)*

*feature = [mean(x) for x in zip(\*result)]*

*return feature*

*## get bert embeddings for training dataset*

*bert\_sentence\_training\_features = train\_data['clean\_text\_bert'].apply(embeddToBERT)*

*feature = [x for x in bert\_sentence\_training\_features.transpose()]*

*bert\_sentence\_training\_features = np.asarray(feature)*

*## get bert embeddings for test reviews*

*bert\_sentence\_test\_features = test\_data['clean\_text\_bert'].apply(embeddToBERT)*

*feature = [x for x in bert\_sentence\_test\_features.transpose()]*

*bert\_sentence\_test\_features = np.asarray(feature)*

**4.5 Model Implementations**

There are total 6 machine learning models created in this research. One of them is the transformer model and the rest are traditional machine learning models. All the traditional models are created with TF-IDF, Word2Vec and BERT embedding techniques as described in previous section.

**4.5.1 Traditional Machine learning models**

In this research we have covered multi-class classification models like Logistic Regression, SVM, Decision Tree, Naïve Bayes and Random Forest.

**4.5.1.1 Logistic Regression Model**

All three embeddings TF-IDF, Word2Vec and BERT were divided into 80:20 ratio for the train and test dataset. Logistic Regression models are built on top of training dataset for these embeddings. After that models are used to make predictions on the test dataset. Hyper parameters were tuned to get the models with highest accuracy and f1-score. GridSearchCV class of scikit learn library was used for finding the best parameters. 10-fold cross validation was used for the best results. Different performance metrics were calculated for the models like Accuracy, precision, recall, f1-score, and AUC-ROC. In the next chapter results are discussed in greater detail.

**4.5.1.2 Naïve Bayes Model**

For the Naïve Bayes Model, Dataset was divided into 80:20 ratio for the train and test data. Three Naïve Bayes models were created for all 3 embedding techniques TF-IDF, Word2Vec and BERT. After training, Models were used to predict on the test dataset. Multinomial Naïve Bayes was used with TF-IDF as features were having discrete values. And Gaussian Naïve Bayes was used for Word2Vec and BERT embeddings as features were in decimal form. Different performance metrics were calculated for the models like Accuracy, precision, recall, f1-score, and AUC-ROC. In the next chapter results are discussed in greater detail.

**4.5.1.3 Decision Tree Model**

For the Decision Tree Model, Embeddings TF-IDF, Word2Vec and BERT were created and 80% of the data was used for training. After training, Models were used to predict on the test dataset. Different performance metrics were calculated for the models like Accuracy, precision, recall, f1-score, and AUC-ROC. In the next chapter results are discussed in greater detail.

**4.5.1.4 Random Forest Model**

For Random Forest model, Again dataset was divided into 80:20 ratio for train and test data and RandomizedSearchCV from scikit learn library was used for searching best hyper- parameters. After training models on training data, Models were used to predict on the test dataset. Different performance metrics were calculated for the models like Accuracy, precision, recall, f1-score, and AUC-ROC. Following hyperparameters are tuned to get the best score.

**4.5.1.5 Support Vector Machine Model**

For the SVM Model, Embeddings TF-IDF, Word2Vec and BERT were created and 80% of the data was used for training. After training, Models were used to predict on the test dataset. Different performance metrics were calculated for the models like Accuracy, precision, recall, f1-score, and AUC-ROC. In the next chapter results are discussed in greater detail.

**4.5.2 Transformer based Machine learning Model**

Transformer library has various pre-trained models for Natural Language Understanding tasks. BERT, XLNet, ELECTRA are few of such pre-trained models. Since training a transformer model is a time consuming task, we have used google colaboratory, so that this training time can be reduced by using GPU. We have used following libraries for training transformer based model.

Table 4.11 Libraries used for training transformer model

|  |  |
| --- | --- |
| Library/Language Name | Library Version |
| Python | 3.8.0 |
| Transformers | 3.0.12 |
| Torch | 1.4.0 |
| Pandas | 1.2.4 |
| Numpy | 1.21.0 |
| Nltk | 3.6.2 |
| Scikit-learn | 0.24.2 |

**4.5.2.1 BERT Model**

There are many BERT variants available on hugging face repository like BERT-base, BERT-large etc. (BERT — transformers 4.7.0 documentation, 2021). These models can be accessed by users without really need of downloading it. Following table shows BERT models available as pre-trained models.

Table 4.12 Pre-trained BERT model variants

|  |  |
| --- | --- |
| BERT-base-uncased | 12-layer, 768-hidden, 12-heads, 110M parameters.  Trained on lower-cased English text. |
| BERT-large-uncased | 24-layer, 1024-hidden, 16-heads, 336M parameters.  Trained on lower-cased English text. |
| BERT-base-cased | 12-layer, 768-hidden, 12-heads, 109M parameters.  Trained on cased English text. |
| BERT-large-cased | 24-layer, 1024-hidden, 16-heads, 335M parameters.  Trained on cased English text |
| BERT-base-multilingual-uncased | 12-layer, 768-hidden, 12-heads, 168M parameters.  Trained on lower-cased text in the top 102 languages with the largest Wikipedia. |
| BERT-base-multilingual-cased | 12-layer, 768-hidden, 12-heads, 179M parameters.  Trained on cased text in the top 104 languages with the largest Wikipedia |
| BERT-large-uncased-whole-word-masking | 24-layer, 1024-hidden, 16-heads, 336M parameters.  Trained on lower-cased English text using Whole-Word-Masking. |
| BERT-large-cased-whole-word-masking | 24-layer, 1024-hidden, 16-heads, 335M parameters.  Trained on cased English text using Whole-Word-Masking. |
| BERT-large-uncased-whole-word-masking-finetuned-squad | 24-layer, 1024-hidden, 16-heads, 336M parameters. |
| BERT-large-cased-whole-word-masking-finetuned-squad | 24-layer, 1024-hidden, 16-heads, 335M parameters. |

In this research we have used BERT-base-uncased model for the experiment. This model is trained through multiple epochs and different hyper parameters were used like batch size, learning rate etc. Once we got the best accuracy with the model on train data, we used the model to validate against the test data. Detailed results are outlined in the next section.

**4.6 Summary**

In this chapter we performed exploratory data analysis by visualising distribution of word frequency, sentence length etc. We also visualised the distribution of stop words and other words for both positive and negative sentiments. After that, N-Gram plots were used to visualise top unigrams, bigrams and trigrams that contributes the most for positive and negative sentiments. Later word clouds were plotted to get an idea of most frequent words for both the target classes. As a part of feature engineering, three different word embeddings were created namely TF-IDF, Word2Vec and BERT embeddings. Various multi-class classification machine learning algorithms are built on top of these embeddings like Logistic Regression, Decision Tree, Naïve Bayes, Random Forest, SVM etc and one of the transformer model BERT is also built.

**CHAPTER 5**

**RESULTS AND EVALUTATIONS**

# 5.1 Introduction

This chapter presents results obtained from traditional machine learning models and transformer-based machine learning model. Hyper parameters are tuned for all these models to get the best accuracy. Confusion matrix is plotted for all these models. Classification reports are also built to analyse the aggregated report parameters such as macro average and weighted average etc. Accuracy and performance metrics are then compared with base papers.

**5.2 Model Output and Evaluation**

In the traditional machine learning algorithms, Logistic Regression, SVM, Decision Tree, Naïve Bayes, and Random Forest are built on top of three word embedding techniques like TF-IDF, Word2Vec and BERT Embeddings. And then for transformer mode, state-of-the-art BERT model is built. For measuring the model performance, We have considered Precision, Recall, F1-Score, Accuracy and AUC score so that performance of the best model based on these embedding techniques can be compared with base paper (Tripathi et al., 2020) as they have used similar metrics in the research. Also weighted average and Macro average we have not captured as both positive and negative target classes have equal proportion in the dataset, So score of these metrics will be same as score of positive or negative labels.

**5.2.1 Logistic Regression based on different word embeddings**

After text pre-processing and creation of different embeddings like TF-IDF, Word2Vec and BERT, Logistic Regression model was built for all three embeddings and hyperparameters are tuned to get the best accuracy and f1-score on the validation dataset. Following hyper parameters were tuned for this model.

Table 5.1 Hyper Parameters used for Logistic Regression Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Penalty | Used to specify the norm used in the penalization |
| C | Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization. |
| Solver | Algorithm to use in the optimization problem. |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.2 Logistic Regression model output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Embedding** | **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy** | **AUC** | **Support** |
| TF-IDF | Positive | 0.89 | 0.91 | 0.90 | 89.50 | 0.9585 | 5007 |
| Negative | 0.90 | 0.88 | 0.89 | 4993 |
| Word2Vec | Positive | 0.86 | 0.87 | 0.87 | 86.49 | 0.9329 | 5007 |
| Negative | 0.87 | 0.86 | 0.86 | 4993 |
| BERT | Positive | **0.91** | **0.90** | **0.91** | **90.64** | **0.9664** | 5007 |
| Negative | **0.90** | **0.91** | **0.91** | 4993 |

From all three word embedding techniques, TF-IDF, Word2Vec and BERT with Logistic Regression model, BERT embeddings yielded the best performance as highlighted in different colour in above table.

Chart

Description automatically generated

Figure 5.1 Confusion matrix for Test dataset for LR model

Above figure represents the confusion matrix for the LR model with BERT embeddings, it shows that LR model predicted sentiments equally well for both positive and negative sentiments.

Table 5.3 Comparison Table with the base paper for LR Model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **Embeddings** | **Precision** | **Recall** | **F1-Score** | **Accuracy**  **(%)** | **AUC** |
| Base Paper (Tripathi et al., 2020) | BOW | 0.8708 | 0.877 | 0.8742 | 87.28 | 0.94 |
| Base Paper (Tripathi et al., 2020) | TF-IDF | 0.882 | 0.905 | 0.8936 | 89.14 | 0.96 |
| Proposed Approach | **BERT** | **0.90** | **0.91** | **0.91** | **90.64** | **0.9664** |

As per above table, If we compare the performance of this BERT Embedding based Logistic Regression model with base paper (Tripathi et al., 2020), We have observed that BERT based LR model built in this research yields better performance than BOW and TF-IDF based embeddings that is used in the base paper. This base paper suggested that BOW with LR yielded 87% accuracy and 0.94 as AUC. While TF-IDF yielded 89% Accuracy and 96 as AUC. We got similar score with TF-IDF embeddings, and with BERT embeddings accuracy is around 90.64% and AUC is around 0.9664.

**5.2.2 Naïve Bayes based on different word embeddings**

After text pre-processing and creation of different embeddings like TF-IDF, Word2Vec and BERT, Naïve Bayes model was built with all three embeddings and hyperparameters are tuned to get the best accuracy and f1-score on the validation dataset. Following hyper parameters were tuned for this model.

Table 5.4 Hyper Parameters used for Naïve Bayes Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| var\_smoothing | Portion of the largest variance of all features that is added to variances for calculation stability. |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.5 Naïve Bayes model output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Embedding** | **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy**  **(%)** | **AUC** | **Support** |
| TF-IDF | Positive | **0.86** | **0.89** | **0.87** | **87.07** | **0.9409** | 5007 |
| Negative | **0.88** | **0.86** | **0.87** | 4993 |
| Word2Vec | Positive | 0.76 | 0.76 | 0.76 | 75.80 | 0.8366 | 5007 |
| Negative | 0.76 | 0.75 | 0.76 | 4993 |
| BERT | Positive | 0.86 | 0.83 | 0.85 | 84.84 | 0.9052 | 5007 |
| Negative | 0.83 | 0.87 | 0.85 | 4993 |

From all three word embedding techniques, TF-IDF, Word2Vec and BERT with Logistic Regression model, TF-IDF embeddings yielded the best performance, and Word2Vec yielded the worst performance as highlighted in different colour in the above table.

**Chart

Description automatically generated with medium confidence**

Figure 5.2 Confusion matrix for Test dataset with NB model

Above figure represents the confusion matrix for the Naïve Bayes model with BERT embeddings, it shows that NB model positive sentiments better than negative sentiments overall.

Table 5.6 Comparison Table with the base paper for Naïve Bayes Model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **Embeddings** | **Precision** | **Recall** | **F1-Score** | **Accuracy**  **(%)** | **AUC** |
| Base Paper (Tripathi et al., 2020) | **BOW** | **0.8566** | **0.865** | **0.8612** | **85.94** | - |
| Base Paper (Tripathi et al., 2020) | TF-IDF | 0.8285 | 0.817 | 0.823 | 82.28 | 0.85 |
| Proposed Approach | BERT | 0.85 | 0.85 | 0.85 | 84.84 | **0.905** |

As per above table, If we compare the performance of this BERT Embedding based Naïve Bayes model with base paper (Tripathi et al., 2020), We have observed that BERT based NB model doesn’t yield better performance than BOW based embeddings that is used in the base paper. This base paper suggested that BOW with NB yielded 85.94% accuracy. While TF-IDF yielded 82.28% Accuracy and 0.85 as AUC. We got slightly better score with TF-IDF embeddings, that is 87.07% as mentioned in the above table and with BERT embeddings accuracy is around 84.84% and AUC is around 0.905.

**5.2.3 Decision Tree based on different word embeddings**

After text pre-processing and creation of different embeddings like TF-IDF, Word2Vec and BERT, Decision Tree model was built with all three embeddings and hyperparameters are tuned to get the best accuracy and f1-score on the validation dataset. Following hyper parameters were tuned for this model.

Table 5.7 Hyper Parameters used for Decision Tree Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| max\_depth | The maximum depth of the tree |
| min\_samples\_split | The minimum number of samples required to split an internal node |
| max\_leaf\_nodes | The minimum number of samples required to be at a leaf node |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.8 Decision Tree model output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Embedding** | **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy**  **(%)** | **AUC** | **Support** |
| TF-IDF | Positive | 0.73 | 0.73 | 0.73 | 72.81 | 0.7280 | 5007 |
| Negative | 0.73 | 0.73 | 0.73 | 4993 |
| Word2Vec | Positive | 0.76 | 0.70 | 0.73 | 73.91 | 0.8094 | 5007 |
| Negative | 0.72 | 0.78 | 0.75 | 4993 |
| BERT | Positive | **0.83** | **0.82** | **0.83** | **82.85** | **0.9004** | 5007 |
| Negative | **0.82** | **0.84** | **0.83** | 4993 |

From all three word embedding techniques, TF-IDF, Word2Vec and BERT with Decision Tree model, BERT embeddings yielded the best performance as highlighted in different colour in the above table. Word2Vec embeddings showed better performance than TF-IDF with decision tree model.

**Chart

Description automatically generated with medium confidence**

Figure 5.3 Confusion matrix for Test dataset with Decision Tree model

Above figure represents the confusion matrix for the Decision Tree model with BERT embeddings, it shows that Decision Tree model predicted positive and negative sentiments equally well.

Table 5.9 Comparison Table with the base paper for Decision Tree model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **Embeddings** | **Precision** | **Recall** | **F1-Score** | **Accuracy** | **AUC** |
| Base Paper (Tripathi et al., 2020) | BOW | 0.7218 | 0.701 | 0.7014 | 71.34 | 0.71 |
| Base Paper (Tripathi et al., 2020) | TF-IDF | 0.7098 | 0.709 | 0.7114 | 70.66 | 0.71 |
| Proposed Approach | **BERT** | **0.83** | **0.83** | **0.83** | **82.85** | **0.9004** |

As per above table, If we compare the performance of this BERT Embedding based Decision Tree model with base paper (Tripathi et al., 2020), We have observed that BERT based DT model yields better performance than BOW and TF-IDF based embeddings that are used in the base paper. This base paper suggested that BOW with DT yielded 71.34% accuracy and 0.71 as AUC. While TF-IDF yielded 70.66% Accuracy and 0.71 as AUC. We got slightly better score with TF-IDF embeddings, that is 72.81% as mentioned in the above table and the best score with BERT embeddings, with accuracy around 78.98% and AUC of 0.7898. Below is a summary table for the comparison.

**5.2.4 Random Forest based on different word embeddings**

After text pre-processing and creation of different embeddings like TF-IDF, Word2Vec and BERT, Random Forest model was built with all three embeddings and hyperparameters are tuned to get the best accuracy and f1-score on the validation dataset. Following hyper parameters were tuned for this model.

Table 5.10 Hyper Parameters used for Random Forest Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| N\_estimators | The number of trees in the forest. |
| max\_depth | The maximum depth of the tree. |
| min\_samples\_split | The minimum number of samples required to split an internal node |
| max\_leaf\_nodes | Grow trees with max\_leaf\_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes. |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.11 Random Forest model output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Embedding** | **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy** | **AUC** | **Support** |
| TF-IDF | Positive | 0.85 | 0.88 | 0.86 | 86.22 | 0.9320 | 5007 |
| Negative | 0.87 | 0.85 | 0.86 | 4993 |
| Word2Vec | Positive | 0.82 | 0.85 | 0.84 | 83.53 | 0.9153 | 5007 |
| Negative | 0.85 | 0.82 | 0.83 | 4993 |
| BERT | Positive | **0.87** | **0.88** | **0.87** | **87.09** | **0.9429** | 5007 |
| Negative | **0.88** | **0.86** | **0.87** | 4993 |

From all three word embedding techniques, TF-IDF, Word2Vec and BERT with Random Forest model, BERT embeddings yielded the best performance as highlighted in different colour in the above table.

**Chart

Description automatically generated with medium confidence**

Figure 5.4 Confusion matrix for Test dataset with Random Forest model

Above figure represents the confusion matrix for the Random Forest model with BERT embeddings, it shows that Random Forest model predicted positive and negative sentiments equally well.

Table 5.12 Comparison Table with the base paper for Random Forest model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **Embeddings** | **Precision** | **Recall** | **F1-Score** | **Accuracy** | **AUC** |
| Base Paper (Tripathi et al., 2020) | BOW | 0.862 | 0.855 | 0.8589 | 85.84 | 0.93 |
| Base Paper (Tripathi et al., 2020) | TF-IDF | 0.8597 | 0.853 | 0.8568 | 85.62 | 0.93 |
| Proposed Approach | **BERT** | **0.87** | **0.87** | **0.87** | **87.09** | **0.9429** |

From the above table, If we compare the performance of this BERT Embedding based Random Forest model with base paper (Tripathi et al., 2020), We have observed that BERT based Random Forest model yields better performance than BOW and TF-IDF based embeddings that are used with Random Forest model in the base paper. This base paper suggested that BOW with RF yielded 85.84% accuracy and 0.93 as AUC. With TF-IDF embeddings RF yielded 85.62% Accuracy and 0.93 as AUC. We got slightly better score with TF-IDF embeddings, that is 86.22% accuracy and 0.93 as AUC as mentioned in the above table, We got the best score with BERT embeddings, with accuracy around 87.09% and AUC of 0.9429. Below is a summary table for the comparison.

**5.2.5 SVM based on different word embeddings**

After text pre-processing and creation of different embeddings like TF-IDF, Word2Vec and BERT, SVM model was built with all three embeddings and hyperparameters are tuned to get the best accuracy and f1-score on the validation dataset. Following hyper parameters were tuned for this model.

Table 5.13 Hyper Parameters used for SVM Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| C | Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty |
| kernel | Specifies the kernel type to be used in the algorithm. It must be one of ‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’ or a callable. If none is given, ‘rbf’ will be used |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.14 SVM model output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Embedding** | **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy** | **AUC** | **Support** |
| TF-IDF | Positive | 0.89 | 0.90 | 0.90 | 89.64 | 0.9593 | 5007 |
| Negative | 0.90 | 0.89 | 0.90 | 4993 |
| Word2Vec | Positive | 0.86 | 0.86 | 0.86 | 86.07 | 0.9327 | 5007 |
| Negative | 0.86 | 0.86 | 0.86 | 4993 |
| BERT | Positive | **0.91** | **0.90** | **0.91** | **90.64** | **0.9656** | 5007 |
| Negative | **0.90** | **0.91** | **0.91** | 4993 |

SVM model yielded the best accuracy from all traditional machine learning models that we tried, that is 90.64% with AUC as 0.9656 with BERT embeddings. This is well beyond all the other traditional machine learning models that were used in this research.

Chart

Description automatically generated

Figure 5.5 Confusion matrix for Test dataset for SVM model

Above figure represents the confusion matrix for the SVM model with BERT embeddings, it shows that SVM model predicted sentiments equally well for both positive and negative sentiments.

**5.2.6 Transformer based BERT model**

There are quite a few variants of transformer-based models available. For this research we have built BERT model. Model is trained using AdamW optimizer, model is tuned with multiple combinations of hyperparameters such as Batch Size, Learning Rate, Number of epochs. Below table shows the hyperparameters used for tuning BERT model.

Table 5.15 Hyper Parameters tuned for the BERT Model

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Batch Size | Number of training examples used in one training step. |
| Learning Rate | controls how much to change the model in response to the estimated error each time the model weights are updated |
| Number of Epochs | Number of complete passed through the training dataset |
| Optimizer | Method used to minimize error function |

Other parameters were kept with default values provided by scikit-learn library. Following table shows the model outputs.

Table 5.16 BERT model output

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Label** | **Precision** | **Recall** | **F1-score** | **Accuracy** | **AUC** | **Support** |
| Positive | 0.93 | 0.94 | 0.94 | 93.66 | 0.9832 | 5007 |
| Negative | 0.94 | 0.93 | 0.94 | 4993 |

BERT model yielded the best performance amongst all the models that are used in this research. Model has the best accuracy of 93.66% with AUC as 0.9832. Precision, Recall and F1-score are around 0.94 that is also the best so far amongst all other models.

**Chart

Description automatically generated**

Figure 5.6 Confusion matrix for Test dataset for BERT model

Above figure represents the confusion matrix for the BERT model, it shows that BERT model predicted sentiments equally well for both positive and negative sentiments.

**5.3 Summary**

In this chapter, Traditional machine learning models as well as transformer-based models were built. Traditional machine learning models like Logistic Regression, Naïve Bayes, Decision Tree, Random Forest and SVM were built using three embedding techniques TF-IDF, Word2Vec and BERT embeddings. Hyper parameters were tuned for all the models to get the model with high accuracy. Confusion matrix were plotted for the best model amongst all three embedding techniques. Also, performance metrics for the best performing model were compared with the base paper.

**CHAPTER 6**

**CONCLUSIONS AND RECOMMENDATIONS**

**6.1 Introduction**

This chapter presents the discussion of the results obtained for all machine learning models used in this research based on different parameters like generalizability, model training time, model size, resources required to train these models. This chapter also compares the results obtained in this research with the base paper, explains the differences and provides recommendation and the problem are where this research model can be used.

**6.2 Discussion and Conclusion**

We have built one transformer-based BERT model and five traditional machine learning models like Logistic Regression, Naïve Bayes, Decision Tree, Random Forest and SVM in this research. In the previous section 5, We produced the performance metrics for each of these models based on three embedding techniques TF-IDF, Word2Vec and BERT embeddings.

To summarize this comparison, we can easily argue that the test accuracy, precision, recall and F1-score provides a clear overall picture of the model’s output for this sentiment classification task. Here dataset is balanced with positive and negative sentiments, so weighted average and macro average of these metrics will be same as the score on positive or negative label. We have also compared the performance of all models. In general, Transformer based BERT model have outperformed traditional machine learning models, because transformer-based BERT model captures all bi-directional contextual and semantic relationships and classify sentiments with high accuracy. Transformer based BERT model yielded better accuracy, precision, recall, f1-score and AUC than any traditional machine learning model. Since we have used traditional machine learning models based on BERT based embeddings, the results are better than the results of base paper (Tripathi et al., 2020) which are based either on TF-IDF or BOW embeddings. And this proves that BERT based word embeddings language model outperforms traditional word embeddings such as TF-IDF, BOW, Word2Vec etc. in terms of handling Polysemic words. Following table captures the performance of BERT embeddings with different machine learning models that are used in this research.

Table 6.1 Performance of models

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **F1-Score** | **Accuracy** | **AUC** |
| Logistic Regression based on BERT embeddings | 0.90 | 0.91 | 0.91 | 90.64 | 0.9664 |
| Naïve Bayes based on BERT embeddings | 0.85 | 0.85 | 0.85 | 84.84 | 0.905 |
| Decision Tree based on BERT embeddings | 0.83 | 0.83 | 0.83 | 82.85 | 0.9004 |
| Random Forest based on BERT embeddings | 0.87 | 0.87 | 0.87 | 87.09 | 0.9429 |
| SVM based on BERT embeddings | 0.90 | 0.90 | 0.90 | 90.64 | 0.9656 |
| BERT | **0.94** | **0.94** | **0.94** | **93.66** | **0.9832** |

As per above table, Logistic Regression and SVM yielded the good performance from the set of traditional machine learning models used in this research. But the BERT yielded the best performance overall. There are other few parameters we need to consider for evaluating models such as model complexity, resources needed to train models etc.

**6.2.1 Is model generalizable?**

**6.2.2 Model training time**

In this research, we have built models using google colaboratory, also used GPU for training the BERT model. Following table shows the time taken for training the model.

Table 6.2 Model complexity and training time

|  |  |  |
| --- | --- | --- |
| Model | Tuned Hyper parameters | Training Time |
| Logistic Regression with BERT embeddings | Penalty, C, Solver | Around 40 Minutes for extraction of BERT embeddings and around 40 minutes for hyper parameter tuning |
| Naïve Bayes with BERT embeddings | Var\_smoothing | Around 40 Minutes for extraction of BERT embeddings and around 25 minutes for hyper parameter tuning |
| Decision Tree with BERT embeddings | Max\_depth, min\_samples\_split, max\_leaf\_nodes | Around 40 Minutes for extraction of BERT embeddings and around 40 minutes for hyper parameter tuning |
| Random Forest with BERT embeddings | N\_estimators, max\_depth, min\_samples\_split, max\_leaf\_nodes | Around 40 Minutes for extraction of BERT embeddings and around 70 minutes for hyper parameter tuning |
| SVM with BERT embeddings | C, Kernel | Around 40 Minutes for extraction of BERT embeddings and around 45 minutes for hyper parameter tuning |
| BERT | Parameters: 12-layer, 768-hidden, 12-heads, 110M parameters  Hyper Parameters: Batch Size, Learning Rate, Number of Epochs, Optimizer | Around 2 hours 25 minutes |

As per the above table, Transformer based BERT model took significantly more time than other models. Extracting BERT embeddings took around 40 minutes of time and based on the number of hyper parameters used for tuning the model it took variably between 30 minutes to 1 hour for traditional models and more than 2 hours for the BERT model.

**6.2.3 Model size and Resources required**

Model training time plays a key role in deciding which is the right model to use for a particular use-case. Transformer based model BERT is a state-of-the-art model, It gives the best performance but it has lot of trainable parameters compared to traditional machine learning models. To increase the speed of training, Transformer based model can use more GPU and RAM but that would incur more costs compared to traditional machine learning models which needs CPU and RAM for the training. So, we need to trade-off between the training time, cost and how frequently model needs to be re-deployed while deciding right model to fit in the use cases.

To summarize, if model needs to be deployed on edge devices, model need to be light weight and should take lesser time in building and deploying, So in this use-case Logistic Regression and SVM models with BERT embeddings can be used. Whereas in different use-case where accuracy is more important and cost as well as model size is of not much importance, BERT model can be suggested.

**6.3 Contribution to the knowledge**

In this research, we used IMDB movie review dataset and built around 6 different machine learning models to classify movie review sentiments to positive or negative. EDA covered word frequency as well sentence frequency distributions, N-gram exploration etc.

To get the best performing model we built traditional machine learning models like Logistic Regression, Decision Tree, Naïve Bayes, Random Forest, SVM with all three embedding techniques TF-IDF, Word2Vec and BERT Embeddings. Then we built transformer-based BERT model as well to compare the performance of traditional machine learning models to the state-of-the-art model. We also compared other factors like model parameters, Time taken to train models, Model size etc.

Models built in this research are not limited to just classify movie review sentiments they can be used in other text classification tasks as well like online product reviews sentiment analysis, online Airline tweets classification, Question answering systems etc.

**6.4 Future Recommendations**

In this research, we used transformer-based model BERT-base on google colaboratory, this requires heavy resources, so GPU was used to train the model. So as a future experiment, Other transformer models like RoBERT, DistilBERT, XLNET can be applied. Also, as part of sentence embeddings model we used quick and high quality pre-trained model **paraphrase-MiniLM-L6-v2**. So, there is a future scope of applying other pre-trained models which are not quick but ensures improvement in quality and see if we can get a model with better performance.

On the traditional machine learning models, we tried LR, NB, DT, Random Forest, and SVM

With three embedding techniques TF-IDF, Word2Vec and BERT. And with BERT embeddings we got quite better performance than the base paper obtained with the use of TF-IDF and BOW embeddings. So as a future enhancement ULMFit based embeddings can be experimented and from the model perspective, XGBoost can be experimented.

# REFERENCES

Al-Saqqa, S. and Awajan, A., (2019) The Use of Word2vec Model in Sentiment Analysis: A Survey. *ACM International Conference Proceeding Series*, June 2020, pp.39–43.

Anon (2021) *A Light Introduction to BERT. Pre-training of Deep Bidirectional… | by constanza fierro | dair.ai | Medium*. [online] Available at: https://medium.com/dair-ai/a-light-introduction-to-bert-2da54f96b68c [Accessed 19 Jun. 2021].

Anon (2021) *BERT — transformers 4.7.0 documentation*. [online] Available at: https://huggingface.co/transformers/model\_doc/bert.html [Accessed 19 Jun. 2021].

Anon (2021) *BERT Explained: State of the art language model for NLP | by Rani Horev | Towards Data Science*. [online] Available at: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 [Accessed 19 Jun. 2021].

Anon (2021) *BoW Model and TF-IDF For Creating Feature From Text*. [online] Available at: https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/ [Accessed 11 Apr. 2021].

Anon (2021) *Generate Unigrams Bigrams Trigrams Ngrams Etc In Python | Arshad Mehmood*. [online] Available at: https://arshadmehmood.com/development/generate-unigrams-bigrams-trigrams-ngrams-etc-in-python/ [Accessed 4 Jul. 2021].

Anon (2021) *IMDB Dataset of 50K Movie Reviews | Kaggle*. [online] Available at: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews [Accessed 9 Apr. 2021].

Anon (2021) *NLP 101: Word2Vec — Skip-gram and CBOW | by Ria Kulshrestha | Towards Data Science*. [online] Available at: https://towardsdatascience.com/nlp-101-word2vec-skip-gram-and-cbow-93512ee24314 [Accessed 11 Apr. 2021].

Anon (2021) *Pretrained Models — Sentence-Transformers documentation*. [online] Available at: https://www.sbert.net/docs/pretrained\_models.html [Accessed 8 Jul. 2021].

Anon (2021) *What is BERT | BERT For Text Classification*. [online] Available at: https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/ [Accessed 11 Apr. 2021].

Bansal, B. and Srivastava, S., (2018) Sentiment classification of online consumer reviews using word vector representations. *Procedia Computer Science*, [online] 132, pp.1147–1153. Available at: https://doi.org/10.1016/j.procs.2018.05.029.

Bayhaqy, A., Sfenrianto, S., Nainggolan, K. and Kaburuan, E.R., (2018) Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes. *2018 International Conference on Orange Technologies, ICOT 2018*.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1, pp.4171–4186.

Dey, S., Wasif, S., Tonmoy, D.S., Sultana, S., Sarkar, J. and Dey, M., (2020) A Comparative Study of Support Vector Machine and Naive Bayes Classifier for Sentiment Analysis on Amazon Product Reviews. *2020 International Conference on Contemporary Computing and Applications, IC3A 2020*, pp.217–220.

Dhola, K. and Saradva, M., (2021) A comparative evaluation of traditional machine learning and deep learning classification techniques for sentiment analysis. *Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering*, pp.932–936.

Haque, M.R., Akter Lima, S. and Mishu, S.Z., (2019) Performance Analysis of Different Neural Networks for Sentiment Analysis on IMDb Movie Reviews. *3rd International Conference on Electrical, Computer and Telecommunication Engineering, ICECTE 2019*, pp.161–164.

Hossain, R., Ahamed, F., Zannat, R. and Rabbani, M.G., (2020) Comparative Sentiment Analysis using Difference Types of Machine Learning Algorithm. *Proceedings of the 2019 8th International Conference on System Modeling and Advancement in Research Trends, SMART 2019*, pp.329–333.

Indulkar, Y. and Patil, A., (2020) Sentiment Analysis of Uber Ola using Deep Learning. *Proceedings - International Conference on Smart Electronics and Communication, ICOSEC 2020*, Icosec, pp.21–27.

Kumar Singh, P., Sharma, S. and Paul, S., (2020) Identifying hidden sentiment in text using deep neural network. *2nd International Conference on Data, Engineering and Applications, IDEA 2020*, pp.14–18.

Mengistie, T.T. and Kumar, D., (2021) COVID-19 Public Reviews.

Mohbey, K.K., (2021) Sentiment analysis for product rating using a deep learning approach. *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*, pp.121–126.

Moolthaisong, K. and Songpan, W., (2020) Emotion Analysis and Classification of Movie Reviews Using Data Mining. *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics, DATABIA 2020 - Proceedings*, pp.89–92.

Pipalia, K., Bhadja, R. and Shukla, M., (2020) Comparative analysis of different transformer based architectures used in sentiment analysis. *Proceedings of the 2020 9th International Conference on System Modeling and Advancement in Research Trends, SMART 2020*, pp.411–415.

Pradha, S., Halgamuge, M.N. and Tran Quoc Vinh, N., (2019) Effective text data preprocessing technique for sentiment analysis in social media data. *Proceedings of 2019 11th International Conference on Knowledge and Systems Engineering, KSE 2019*.

Qaisar, S.M., (2020) Sentiment Analysis of IMDb Movie Reviews Using Long Short-Term Memory. *2020 2nd International Conference on Computer and Information Sciences, ICCIS 2020*, pp.7–10.

Sahu, T.P. and Ahuja, S., (2016) Sentiment analysis of movie reviews: A study on feature selection and classification algorithms. *International Conference on Microelectronics, Computing and Communication, MicroCom 2016*, pp.0–5.

Salur, M.U. and Aydin, I., (2020) A Novel Hybrid Deep Learning Model for Sentiment Classification. *IEEE Access*, 8, pp.58080–58093.

Seo, S., Kim, C., Kim, H., Mo, K. and Kang, P., (2020) Comparative Study of Deep Learning-Based Sentiment Classification. *IEEE Access*, 8, pp.6861–6875.

Shehu, S.A., Mohammed, A.D. and Abdullahi, I.M., (2020) An Optimized Customers Sentiment Analysis Model Using Pastoralist Optimization Algorithm ( POA ) and Deep Learning. pp.132–139.

Singh, H., Yadav, A., Bansal, R. and Mala, S., (2021) Understanding brand authenticity sentiments using big data analytics. *Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering*, pp.304–308.

Tamara, K. and Milićević, N., (2018) Comparing Sentiment Analysis and Document Representation Methods of Amazon Reviews. *SISY 2018 - IEEE 16th International Symposium on Intelligent Systems and Informatics, Proceedings*, pp.283–288.

Tammina, S. and Annareddy, S., (2020) Sentiment analysis on customer reviews using convolutional neural network. *2020 International Conference on Computer Communication and Informatics, ICCCI 2020*, pp.20–25.

Tarımer, İ., Çoban, A. and Kocaman, A.E., (n.d.) *Sentiment Analysis on IMDB Movie Comments and Twitter Data by Machine Learning and Vector Space Techniques*.

Tiwari, S., Verma, A., Garg, P. and Bansal, D., (2020) Social Media Sentiment Analysis on Twitter Datasets. *2020 6th International Conference on Advanced Computing and Communication Systems, ICACCS 2020*, pp.925–927.

Tripathi, S., Mehrotra, R., Bansal, V. and Upadhyay, S., (2020) Analyzing Sentiment using IMDb Dataset. *Proceedings - 2020 12th International Conference on Computational Intelligence and Communication Networks, CICN 2020*, pp.30–33.

Tripathy, A., Agrawal, A. and Rath, S.K., (2016) Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications*, [online] 57March, pp.117–126. Available at: http://dx.doi.org/10.1016/j.eswa.2016.03.028.

Vohra, M.S.M. and Teraiya, P.J.B., (2013) Journal of Information, Knowledge and Research in Computer Engineering a Comparative Study of Sentiment Analysis Techniques. *Journal of Information,Knowledge and Research in Computer Engineering*, pp.313–317.

Wazery, Y.M., Mohammed, H.S. and Houssein, E.H., (2019) Twitter sentiment analysis using deep neural network. *ICENCO 2018 - 14th International Computer Engineering Conference: Secure Smart Societies*, pp.177–182.

Yasen, M. and Tedmori, S., (2019) Movies reviews sentiment analysis and classification. *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology, JEEIT 2019 - Proceedings*, pp.860–865.

Yousefpour, A., Ibrahim, R., Hamed, H.N.A. and Hajmohammadi, M.S., (2014) A comparative study on sentiment analysis. *Advances in Environmental Biology*, 813, pp.53–68.

Zahoor, K., Bawany, N.Z. and Hamid, S., (2020) Sentiment analysis and classification of restaurant reviews using machine learning. *Proceedings - 2020 21st International Arab Conference on Information Technology, ACIT 2020*.

Zhang, Y. and Rao, Z., (2020) Deep neural networks with pre-train model BERT for aspect-level sentiments classification. *Proceedings of 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference, ITOEC 2020*, Itoec, pp.923–927.

**APPENDIX A: RESEARCH PLAN**

Following Gantt chart provides high level overview of the tasks that are performed with timelines and the plan ahead till the final report submission.

Chart

Description automatically generated

APPENDIX A: Project Plan Gantt Chart

**APPENDIX B**

**DATASET URL**

<https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews>

**MACHINE LEARNING MODELS BUILT**

https://github.com/sdtalaviya/masters\_research