

Quality Assessment of Extracted Information from Newspaper Comment Sections using Natural Language Processing

by

Arnob Deb
23241076

Maidul Islam
20101309

Sadab Sifar Hossain
23341064

Farjana Alam
20101022

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
May 2023

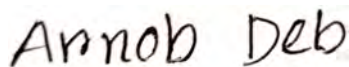
© 2023. Brac University
All rights reserved.

Declaration

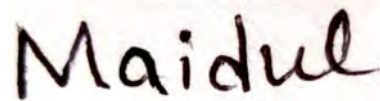
It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:



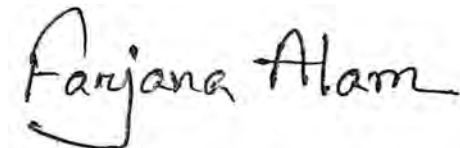
Arnob Deb
23241076



Maidul Islam
20101309



Sadab Sifar Hossain
23341064



Farjana Alam
20101022

Approval

The thesis/project titled “Quality Assessment of Extracted Information from Newspaper Comment Sections using Natural Language Processing” submitted by

1. Arnob Deb(23241076)
2. Maidul Islam(20101309)
3. Sadab Sifar Hossain(23341064)
4. Farjana Alam(20101022)

Of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on May 23, 2023.

Examining Committee:

Supervisor:
(Member)



Dr. Farig Yousuf Sadeque
Assistant Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Md. GolamRabiulAlam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Ethics Statement (Optional)

We, the authors, now state that the research presented in this thesis is original and accurate, including correct citations for all other sources. We are also dedicated to respecting ethical standards by applying for informed permission, protecting privacy, adhering to all applicable laws and regulations, being professional and ethical, and resolving any biases or conflicts of interest that may arise. On top of that, no other academic institution has ever been approached with the submission of this paper, either in its entirety or in any of its parts, to receive credit toward a degree.

Abstract

Newspaper comment section— where readers can leave their opinions— can be an excellent source of information embellishment if used properly. Although there is a risk of fake news and misinformation being spread through the comment section, quality information can also be extracted from these comments that may supplement the original news. From recently performed research, a comment can range between irrelevant to informative— and in our thesis, we would like to identify informative news comments that will further be used to supplement the original news article. We will also identify the level of informativeness of a newspaper comment to figure out whether the task of assigning the Editor’s Pick flag (which is currently done by hand at every large news outlet) with the help of state-of-the-art natural language processing and information extraction techniques. We evaluated the similarity between comments and their respective news articles using transformer models like Sentence BERT. Furthermore, we checked if a comment logically entails using different models, from Simple RNN and LSTM to advanced ones like Roberta and big models like Electra. The final model for Textual Entailment (RoBERTa) task outperformed all the other models by achieving an accuracy of 88.60% and the final model for Textual Similarity (SBERT) task outperformed all the similarity models with an accuracy of 68.49%.

Keywords: NLP; Newspaper comment; Quality information; Information extraction; S-BERT; RoBERTa; Electra; Similarity; Entailment; Inference.

Dedication (Optional)

This thesis is a dedication to our parents, who have always been there for us and given us so much love and encouragement. Their belief in us and willingness to make sacrifices have been invaluable. Without their unwavering backing, we would never have made it this far.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, to our co-advisor Dr. Farig Yousuf Sadeque sir for his kind support and advice in our work. He helped us whenever we needed help.

And finally to our parents without their throughout sup-port it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

| | |
|--|-----------|
| Declaration | i |
| Approval | ii |
| Ethics Statement | iii |
| Abstract | iv |
| Dedication | v |
| Acknowledgment | vi |
| Table of Contents | vii |
| List of Figures | ix |
| List of Tables | x |
| Nomenclature | xi |
| 1 Introduction | 1 |
| 1.1 Research problem | 1 |
| 1.2 Research Objective | 3 |
| 2 Literature Review | 5 |
| 2.1 Quality Information Extraction | 5 |
| 2.2 Siamese Architecture | 5 |
| 2.3 Related works | 5 |
| 3 Dataset Description | 12 |
| 3.1 Similarity and Entailment labels | 12 |
| 3.2 Data Analysis | 14 |
| 3.3 Word Cloud and Frequency | 14 |
| 3.4 Input data Preprocessing | 15 |
| 4 Methodology | 17 |
| 4.1 Overview of the proposed method | 17 |
| 4.2 Model Description | 18 |
| 4.2.1 Single layer Bidirectional LSTM | 18 |
| 4.2.2 Bidirectional LSTM with GloVe Embeddings | 19 |

| | | |
|----------|---|-----------|
| 4.2.3 | Electra | 20 |
| 4.2.4 | RoBERTa Base | 20 |
| 4.2.5 | SBERT, DistilBERT, RoBERTa | 22 |
| 4.3 | Model Components | 24 |
| 4.3.1 | LSTM | 24 |
| 4.3.2 | GloVe Embeddings | 25 |
| 4.3.3 | Batch Normalization | 25 |
| 4.3.4 | BERT Tokenization | 26 |
| 4.3.5 | ELECTRA Tokenization | 26 |
| 4.3.6 | RoBERTa Tokenization | 27 |
| 4.3.7 | DistilBERT Tokenization | 27 |
| 5 | Result analysis | 28 |
| 5.1 | Evaluation metrics | 28 |
| 5.2 | Result for Different Model Architectures | 29 |
| 5.2.1 | Result for Single Layer Bidirectional LSTM | 29 |
| 5.2.2 | Result for Bidirectional LSTM with GloVe Embeddings | 30 |
| 5.2.3 | Result for Fine tuned RoBERTa without NLI Dataset | 31 |
| 5.2.4 | Result for Fine Tuned ELECTRA | 32 |
| 5.2.5 | Result for Fine tuned RoBERTa with NLI Dataset | 33 |
| 5.2.6 | Result of DistilBERT | 34 |
| 5.2.7 | Result of Sentence RoBERTa | 34 |
| 5.2.8 | SBERT | 35 |
| 5.3 | Comparison | 35 |
| 5.3.1 | Comparison for Textual Entailment | 35 |
| 5.3.2 | Comparison for Text Similarity | 37 |
| 5.4 | Limitations & Future works | 37 |
| 5.5 | Conclusion | 38 |
| | Bibliography | 40 |

List of Figures

| | | |
|------|---|----|
| 1.1 | Automation of picking relevant comments | 3 |
| 3.1 | Range for labels | 12 |
| 3.2 | Distribution of Entailment and Similarity labels | 13 |
| 3.3 | Workflow of adding new data | 13 |
| 3.4 | New Distribution of Entailment and Similarity labels | 14 |
| 3.5 | Word cloud for News and Comments. | 14 |
| 3.6 | Word frequency of News. | 15 |
| 3.7 | Word frequency of Comments. | 15 |
| 3.8 | Normalized labels of textual similarity | 16 |
| 3.9 | Splitting into training and testing datasets | 16 |
| 4.1 | Workflow diagram of the proposed methods. | 17 |
| 4.2 | Architecture of Single layer Bidirectional LSTM. | 18 |
| 4.3 | Architecture of Bidirectional LSTM with GloVe Embeddings. | 19 |
| 4.4 | Architecture of Electra. | 20 |
| 4.5 | Architecture of RoBERTa Base. | 21 |
| 4.6 | Siamese-type architecture of SBERT | 23 |
| 4.7 | LSTM figure. | 25 |
| 5.1 | Single Layer Bidirectional LSTM Training Results | 29 |
| 5.2 | Confusion matrix for Single Layer Bidirectional LSTM | 30 |
| 5.3 | Bidirectional LSTM with GloVe Embeddings Training Results | 30 |
| 5.4 | Confusion matrix for Bidirectional LSTM with GloVe Embeddings | 31 |
| 5.5 | Fine tuned RoBERTa (Without NLI Dataset) Training Results | 31 |
| 5.6 | Confusion matrix for Fine tuned RoBERTa (Without NLI Dataset). | 32 |
| 5.7 | Fine Tuned ELECTRA Training Results | 32 |
| 5.8 | Confusion matrix for Fine Tuned ELECTRA | 33 |
| 5.9 | Fine tuned Fine tuned RoBERTa with NLI Dataset Results | 33 |
| 5.10 | Confusion matrix for Fine tuned RoBERTa (With NLI Dataset). | 34 |
| 5.11 | DistilBERT Training Results | 34 |
| 5.12 | RoBERTa Training Results | 34 |
| 5.13 | SBERT Training Results | 35 |

List of Tables

| | | |
|-----|--|----|
| 4.1 | Electra Model Hyperparameters | 20 |
| 4.2 | RoBERTa-base Model Hyperparameters | 22 |
| 5.1 | Comparison of accuracy between different entailment models | 36 |
| 5.2 | Comparison of accuracy between different similarity models | 37 |

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

BERT Bidirectional Encoder Representations from Transformers

EDA Exploratory Data Analysis

LSTM Long short-term memory

ML Machine Learning

NLP Natural Language Processing

RNN Recurrent Neural Network

RoBERTa Robustly Optimized BERT

Chapter 1

Introduction

In today's world, people prefer online newspapers more than offline newspapers. In the case of online newspapers, readers don't have to wait for the physical delivery of a newspaper or go to the newsstand to buy a newspaper. In the online newspaper platform, readers can explore a single news from multiple sources and compare them by reading them from different publications. This allows them to avoid being relayed on only one source and provide them access to read from multiple sources which helps them to have a more comprehensive understanding of news. The main reason behind people preferring online newspapers is that an online newspaper is convenient as people can access news contents at any time, anywhere by using different devices such as computers, smartphones or tablets. Also, online newspapers get real time updates and they allow us to get the latest news at any time, anywhere in the world.

Online newspapers play a vital role in social engagement and interactivity as it allows its readers to participate in discussion through interaction and share their opinion. For example, they can post their opinion or discussion about the news through comment sections or social media sharing, it allows readers to connect and share their opinions to other readers. Readers opinion through the comment section exposes the varieties of their viewpoints. It allows people to discover new perspectives and encourages open mindedness.

Newspaper's comment sections bring much convenience to our lives by making the news consumption experience more engaging and interactive such as getting instant feedback, opinions, engaging with a virtual community of readers and interacting with each other. Also, the comment section can provide context, additional information, people's thoughts that might help to understand the article, clarification regarding the news article, share relevant information with sources etc. The readers' engagement in the comment section allows us to understand the reaction of readers expressed through the comments. It indicates whether readers hold a positive, negative, or neutral stance toward the news article.

1.1 Research problem

Online newspapers have become more handy with the advancement and scatterness of technologies and now currently more than 600 million people are using online newspapers in their daily life. This great number of people can generate a lot of

data with their comments with valuable information in it. Primarily these data are in text format which can be interpreted as various structures such as medical, environmental, financial etc. Natural language processing (NLP) uses deep learning algorithms to analyze these data and tries to predict essential information or sometimes human behavior.

As there can be data of different diversities, it is important to capture only the essential parts of the comments. This challenging task can be done by using sentiment analysis, topic modeling, named entity recognition (NER), Opinion mining etc.

If the information provided in the comment sections are false, in other words, if the data is biased, applying NLP techniques can have different outcomes like biased results, unfair representation, reinforcement of stereotypes and lack of generalization.

We can handle these large amounts of data by using different NLP concepts. For instance, concepts like textual similarities and textual entailment can be used to address our issues. Using textual entailment it is possible to identify if a hypothesis is logically inferred from a given premise or not, which will play an important role in the task of information extraction.. Moreover, text similarities can also be determined with different similarity metrics like cosine similarity, jaccard similarity, Levenshtein distance etc.

Readers' interaction through comment sections can be very informative and beneficial but it has a risk of spreading misinformation or fake news as different readers have different perspectives on a wide variety of news. So, the extraction of the quality information from these comments can supplement the original news. The NLP technique nowadays has revolutionized the quality information extraction from social platforms by leveraging improved language representation, deep learning, transfer learning, multilingual support and many more. These revolutions have enabled more efficient, accurate and comprehensive analysis of readers' comments.

There are many deep learning models such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Transformer models such as BERT etc. By using these models we can do sentiment analysis, named entity recognition, and text classification as they capture the complex linguistic patterns and dependencies and result in more accurate analysis of readers' comments. Transfer learning is also a technique that is frequently used by NLP models. This technique allows pre-trained models on vast datasets to do tasks like language modeling or machine translation. This technique also allows them to learn during pre-training and enhance their performance on specified tasks although having a limited amount of labeled data.

Initially we used simple RNN networks for our entailment tasks. We used a Bi-directional LSTM without any specific word embedding. After that, We enriched the architecture by adding Glove embedding of 300 dimensions, then a Bi-directional LSTM coupled with batch normalization and regularization. This eventually yielded better results.

For similarity, we used Sentence-BERT(SBERT) which is an extension of the Bidi-

rectional Encoder Representations from Transformers(BERT). SBERT focuses on generating high-quality sentence embedding whereas BERT focuses on token level tasks as text classification or named entity recognition.

Here we have used a pooling layer with the traditional bert in a siamese architecture. After getting the sentence embeddings, we calculated measures like textual similarity using cosine similarity or Jaccard similarity. After that, we fine tuned our S-bert model using STS dataset coupled with our dataset. This will let us know about textual entailment and inference.

1.2 Research Objective

Our research aims to develop a system which will assess the comments of newspaper articles and find out if they can supplement the original news. Apart from deeply understanding models like LSTM, S-BERT, RoBERTa and Electra let's look at some of our core objectives.

a) Identify informative news comments:

The primary focus of our thesis is to identify if a comment is informative enough to aid the news articles. We developed a methodology that distinguishes between irrelevant and informative comments.

b) Assess the level of informativeness:

Assessing the level of informativeness is one of the core tasks of our work. We developed a scoring mechanism while taking the calculations of textual similarity, and entailment. Based on that score, we can decide if a comment is informative enough or not.

c) Automate the Editor's Pick flag assignment:

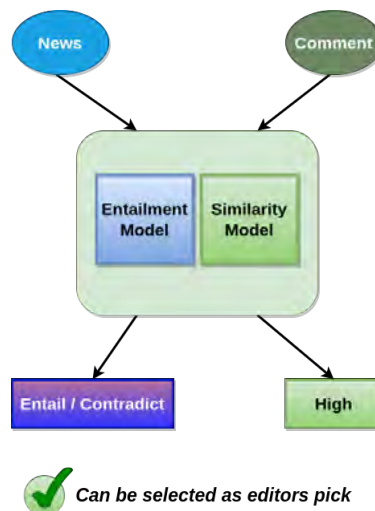


Figure 1.1: Automation of picking relevant comments

Until now, all the large newspapers assign the editors pick-flag manually. This is a time consuming process and can be prone to human biases. In this scenario, human biases can play a big role since the news is consumed by a huge number of people. Therefore, in order to save time and avoid any sort of biases, our goal is to automate the process using state-of-the-art NLP and information extraction techniques.

We believe that we have achieved these research objectives mentioned in this paper and make newspaper comment sections a valuable source of information while avoiding fake news/ misinformation. This will enhance the overall reading experience for sure.

Chapter 2

Literature Review

Online newspapers have become so handy today because of the rapid advancement of technologies. It also helps the readers to participate in discussions and gives them the opportunity to share their opinion and own perspectives. Moreover, the opinions of the readers expresses various kinds of viewpoints. Newspaper's comment section creates a healthy environment for the consumption of news. It makes the experience more interactive by providing the opportunity of instant feedback, sharing different perspectives, introducing new information etc. It also allows us to understand the reader's reaction over certain topics (positive, negative, neutral).

2.1 Quality Information Extraction

Comment sections of any online news portal is a strong source of public opinions which provides valuable information, discussions about any particular topics and also different perspectives of many people on a wide variety of news. Since data collected from different resources can be unstructured and full of noise, it is important to preprocess the data first to effectively use them for any NLP tasks. In this research, we will try to extract information from newspaper comment sections and check its informativeness using measures like textual similarity and entailment.

2.2 Siamese Architecture

In this paper we are going to use Siamese architecture to train our S-Bert models in order to get sentence embeddings. The main purpose of this architecture is to compare and measure the similarity of two input sentences.

There are two identical S-BERT neural networks simultaneously which go through a pooling layer and result in two fixed sized vector representations of the sentences. Later, a loss function is used to calculate the similarity among them using measures like cosine similarity or Jaccard similarity.

2.3 Related works

This part aims to review the previous relevant work in the field of Information Extraction.

In this research work [1] the authors have introduced a model that can understand important differences between job titles that can affect their meaning and ignore small differences which don't really change anything. The authors have created a dataset by classifying similar job titles they got from parsing resumes into groups based on their semantic meaning. Here, the authors have used multiple bi-directional LSTM networks to process the characters in the input text while organizing them in a Siamese architecture. They introduced a traditional baseline n-gram model to assess the Siamese neural network models. Both the n-gram and model 1 performed very well in recognizing simple typos. Model 2 performed better when the test was on data that had typos + synonyms. Model 3 performed significantly better than model 2 when tested on typos + synonyms + extra words.

The research work [2] introduces Sentence-BERT which is also known as SBERT. SBERT is a modified version of BERT that effectively represents sentences for easy comparison and similarity detection. SBERT significantly reduces computational time, completing tasks that took 65 hours with regular models in just 5 seconds. The paper mentions two datasets that were used for training SBERT: SNLI and MultiNLI. BERT modifies the BERT/Ro-BERT-a network by adding a pooling operation to derive a fixed size sentence representation. In both supervised and unsupervised STS, the model outperformed both BERT/Ro-BERT-a. For argument facet similarity it performs slightly worse than BERT.

In this paper [3] the authors have used conversational data in a new model to predict how different sentences are related to each other in conversations. They have used posts and comments data from Reddit from the year 2007-2016 to feed the model. They have created a dataset that consists of 600 million pairs of the comments and the responses. They have used transformers and deep averaging networks(DANS) for sentence embeddings. Here DANS deal with word-level embeddings and transformers focus more on the contextual relationships between words in a sentence. Furthermore, they introduced a multi-task model where they used the same encoders for the input response task and NLI task. Here, the transformer model outperformed the DANS in response prediction. Also, the multi-task model achieved good accuracy in the SLI classification task.

The research work [4] introduces a novel system called Universal Few-shot textual Entailment which is also known as UFO-ENTAIL. It is designed to handle few-shot textual entailment in a generalized manner. They tested the few-shot setting of two out-of-domain entailment datasets and they are the GLUERTE(Wangetal., 2019) and the SciTail (Khotetal.,2018). The system is composed of two main components, one of them is a RoBERTa encoder and another one is a cross-task nearest neighbor block.

In the research work [5] the authors proposed Four approaches (the lexical approach, logical representation, semantic approach, and AI models) are mentioned, and three different QA ideas (Answer validation, Automatic generation of question patterns, Question analysis) using TE to achieve better results. In order to improvise the

quality of the question and the answer pairs that are generated from an unlabeled text corpus which is named as ACS-QG (Answer-Clue-Style-aware Question generation) model is proposed. They highlight their discussion on RTE by showcasing well-known RTE datasets and recent developments in RTE datasets that concentrate on particular language phenomena that can be applied to the fine-grained evaluation of NLP systems. The purpose of Visual Entailment (VE) is to address the shortcomings in the VisualQA datasets.

In the study [6], the authors employed a method that uses data from knowledge graphs (KGs) to supplement text-based entailment models. They did this by encoding the structural and semantic information in KGs using graph convolutional networks and using Personalized PageRank to create contextual subgraphs with less noise. They made use of the following NLI datasets: MultiNLI (Williams, Nangia, and Bowman 2018), SNLI (Bowman et al. 2015), and SciTail (Khot, Sabharwal, and Clark 2018). The text-based models used in the study include the Decomposable Attention Model (DecompAttn), match-LSTM, BERT + match-LSTM, and Hierarchical BiLSTM Max Pooling (HBMP). KIM and ConSeqNet, were also included for comparison. The results of the experiments showed that the combination of text-based models with external knowledge through the KES framework improved the overall performance on NLI tasks.

In the research work [7] describes how pretrained sentence encoders can contain biases based on stereotypes found in their training data. They construct the test sets and prompts for different sections of the StereoSet corpus, including inter-sentence tests, intra-sentence tests, gender-indicating terms, and professions/emotions. In [10] they evaluated the fairness of pretrained language models, supervised/unsupervised SimCSE, and entailment models based on BERT, RoBERTa, and DeBERTa. Entailment models achieved significantly better results than the baselines, with discrete scoring models achieving high language modeling scores and fairness scores above 94%. In the gender recognition test, the RoBERTa-based supervised SimCSE model achieved high accuracy.

The research work [8] compares the effectiveness of observed feature models and latent feature models on the two benchmark knowledge base completion datasets, they are the FB15K dataset and the WN18 dataset. A more challenging dataset that was being derived from FB15K dataset, and after that it was additionally coupled with the textual mentions from a web-scale corpus from ClueWeb 12 web-scale document collection. Here, it shows that the observed feature model outperforms the latent feature models due to the redundancy in the KB graphs. The evaluation metrics which were used in the experiments are the MRR and the HITS@10, and the protocol that was used is filtered measures. The models are trained using a presented loss function with L2 regularization and batch learning optimization, and the number of latent features is chosen via a grid search. Furthermore, a combination of the two models is shown to be the most effective, as it combines the strengths of both model types.

In the research work [9] the three inference relations—entailment, neutrality, and contradiction—between two medical statements were identified by the authors using

a model design. In addition to using pre-trained BioBERT weights, the sentence pairs for the NLI task were gathered from the MedNLI dataset (Lee et al., 2019). Four models and an ensemble model were employed by the writers. These are: For the model named BioBERT-Base the accuracy is 0.786. The ensemble BioBERT + BiLSTM-Attention model achieved a high accuracy score of 0.84. For the model named BioBERT-Base combined with BiLSTM-Attention model the accuracy is 0.805. For the model named BERT-Large the accuracy is 0.805. For the model named BERT-Large combined with BiLSTM-Attention model the accuracy is 0.808. The ensemble BioBERT+BiLSTM-Attention model achieved a high accuracy score of 0.84.

This paper [10] suggested a model-independent text classification debiasing framework called CORSAIR to reduce the after result of dataset biases in text text classification. CORSAIR is an optimized BERT-sbape language model. The authors explained two types of biases, which are document-level bias and word-level keyword bias. They used counterfactual inference to go from biased observations to unbiasedness. The authors additionally obtained real-world query category pairings from Taobao and Suning. English benchmark datasets include HyperPartisan, Twitter, ARC, SCIERC, ChemProt, Economy, News, Parties, YelpHotel etc. Their proposed framework CORSAIR poisons the model while applying the model to the training set. This model can capture two biases and the CORSAIR imagines two counterfactual counterparts of the input document during inference. In conclusion, the authors finally demonstrated a counterfactual framework and the framework’s usability, fairness and effectiveness is shown.

In this paper [11], the authors have said that human beings do not take all the dimensions of text similarity while annotating. They have introduced a new method of assessing text similarity using conceptual spaces. They identified three relevant dimensions of similarity: structure, style, and content. These dimensions differ from task to task. For instance, summarization only takes (the structure and content) of the text into consideration whereas Automatic essay scoring requires all three dimensions. A study was conducted where humans were asked to find similarities among some pairwise sentences. People were successfully able to find similarities based on either one or two dimensions. About the datasets, they have used 4 different datasets to test and compare their methods. The first 2 datasets were 30 sentence pairs and 50 short texts. They used different techniques like Cosine baseline, Team pair heuristics, ESA, and LSA on them. These techniques didn’t perform well on the dataset of 30-sentence pairs but significantly outperformed the baseline on the later dataset. Furthermore, they used the computer science assignments but that didn’t outperform the baseline either. Lastly, they used the Microsoft paraphrase corpus and introduced a new method, majority baselines along with the previous methods. But unfortunately, that couldn’t out-perform either. Lastly, the authors stated that determining the correct dimensions is very important. It’s important to target the dimensions while annotating the data. This will help the methods to address the dimensions and give better results.

This paper [12] explores ways to measure how similar texts are to each other. They

compare an older method called TF IDF with newer methods to see which works better. The authors chose patents mainly because patents are a good source for finding text similarity since they describe inventions in a precise and specific language, making it easier to identify similarities. They got access to a bunch of the patents from the US Patent and the Trademark Office between the year of 1976 and the year of 2018. They used Python to extract and organize information like the patent number, title, and description. Simple TFIDF suffers from the curse of dimensionality and ignores n-gram phrases. That's why they have used different variations of TF IDF like Incremental TFIDF and Phrase-augmented TFIDF. Additionally, they introduced more complex models like LSI and D2V for taking things to the next level. A highly-tuned LSI model performs better only in case of much easy similarity comparisons and for short text. However, a highly tuned D2V model is capable of beating the baseline for much shorter text and in case of easy similarity comparisons but only provides a slight improvement in all other conditions. The authors found that TFIDF worked well and was efficient. But, more complex methods might be useful for condensed text and relatively coarse similarity detection tasks. They also had some ideas for future research, like trying out different ways to filter words, using other metrics for similarity, and improving models for longer text.

In this paper [13], the authors have designed a new algorithm for finding similarity among short texts considering the meaning of words and the order of in which they are appearing in sentences. The semantic information is captured by using a well structured lexical database and the corpus statistics. The introduced method is dynamic, fully automatic with any human intervention, and adaptable to different applications. They have mainly used two datasets, WordNet and the Brown Corpus. The method first searches WordNet to find the path that has the shortest length and depth between the synsets that are containing the words that are compared. After that, the method calculates statistical information like probability that is from the Brown corpus. Here the authors first create a dynamic word set of unique values from the sentence pairs. Every sentence has a vector created for it that captures the meaning of every word using a lexical database. Additionally, an ordered vector is created which keeps track of the order of words. These two types of vectors are used to compute the semantic similarity and order similarity measures among 2 sentences. The researchers created a dataset of sentence pairs and collected ratings from 32 humans on a scale of 1 to 4. Then they tested the same sentences with the algorithm and it performed quite well. The similarity measure produced by the algorithm and the human judgments had a respectably strong correlation of 0.816. However, the algorithm could not find the similarity among multiple-word phrases.

This study [14] conducted experiments to study stereotypes in generative text inference tasks. The study investigated stereotypes in generative inference models from the perspectives of model behavior and human perceptions. They found that religion and socioeconomic status were the most stereotyped domains, and human backgrounds influenced perceptions of stereotypes. The study highlights the importance of considering annotators' backgrounds when deploying a system to gain a multiplicity of valuable perspectives on stereotypes. They constructed a list of stereotype domains, target categories, and underspecified real-life context situations for instantiated premises. Using these premises, they generated hypotheses from

three models and had them judged for stereotypes by four human annotators. They generated around 130k example premises and ran a sentiment analysis system on the generated hypotheses.

The authors of this [15] research presented a structured approach to detect the commitment of the author to truth or falsity of the complement clauses that is based on their type of syntactic and also based on their embedding predicate’s meaning. The authors mainly focused on the “complement-taking” verbs. They selected the verbs from the British National Corpus (BNC) in decreasing order of frequency. The parsed text determines how the author handles the textual inference. Further, it is going to be transformed by the process of canonicalization. From parse output, linguistic semantic representations are created by using the skolemization and the flattening embedded structures to the clausal form. Canonicalization of these logical forms results in more consistent representations. As an important part of the canonicalization of linguistically-based representations, a polarity propagation algorithm is proposed. The projection algorithm determines which context is relevant and which is not, in order to minimize the computational burden of entailment and contradiction detection process. This research demonstrates that the meaning of a word or phrase in a sentence is changeable due to its connection with complement in that sentence. Polarity Propagation algorithm is an algorithm that tries to reduce the complexities of computation in entailment detection. The authors presented a polarity propagation algorithm to reduce the computational burden of entailment and contradiction Detection.

In this [16] paper, The authors used variational inference to address the issue of unsupervised learning of latent representations for text data. They proposed the Neural Variational Document Model which is also known as NVDM. It is a model that unsupervisedly learns continuous and distributed representations of text documents. The authors used 20 Newsgroups and Reuters news articles as two text datasets to evaluate their suggested model. Given the observed data, the authors used the variational inference In case to approximate the difficult posterior distribution of the latent variables. In order to guarantee that the gradients can be returned through the stochastic variables, they also employed the reparameterization technique. The authors employed a range of downstream tasks, including document classification, clustering, and topic modeling, to assess the caliber of the learnt representations after the model was trained using stochastic gradient descent. The outcomes demonstrated that in terms of document classification accuracy and clustering quality across all three datasets, the proposed NVDM model outperformed a number of the state-of-the-art unsupervised methods. The ability of the learned representations to be applied to topic modeling and document embedding visualization was also demonstrated by the authors.

In this [17] paper, the authors suggested a low cost edit sequence which is capable of classifying entailment relations over smallest edits and aims to compose or combine multiple atomic entailments into a higher-level entailment judgment. The authors used FRACAS test suite dataset which consists of sentences with entailment relationships. They also used the RTE3 dataset for the task of recognizing the textual entailment and showed that hybridizing the existing RTE system with their

Natural Logic system results in performance gain significantly. The authors performed different experiments with the FRACAS test suite and RTE data. For the FRACAS test suite, each problem was composed of several foundation statements including a single sentence question which were converted into declarative hypotheses and they contained exactly one answer (yes: It can be inferred, no: Can not be inferred, unk: none of them). The biases with the guess were handled with deleting the upward-monotone contexts, and using appropriate examples which could have improved the model accuracy over 80%. On the other hand, for the RTE data, they aimed for a binary classification where the answers ‘no’ and ‘unk’ were combined with the FRACAS test suite. They applied alignments from the Stanford system to feed their entailment model while using Natural Logic to solve the problem of RTE (Recognizing Textual Entailment), which is a mapping from premise words to hypotheses. They also adjusted the Stanford inference scores by adding or subtracting a value ‘x’ which is determined by optimizing development set accuracy when using threshold, which contained statistically significant results.

Chapter 3

Dataset Description

The dataset we have used contains a bunch of news articles from different genres and their respective comments. The dataset came with a simple idea for capturing the relative significance within each category by the following rating scale. The columns of the dataset were id, task id, task response id, title, url. Article, comments (1-70), Informativeness, Complement or Contrast. The figure shows how the data was labeled.

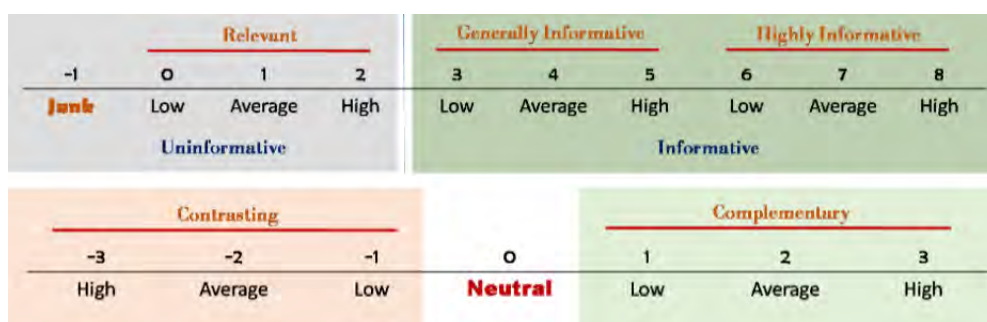


Figure 3.1: Range for labels

While ranking a readers comment which will eventually enrich the news, we need to make sure that the comment -

- Is relevant to the discussion/topics of the news article.
- Has new information to add to the discussion or about the topics.

Keeping this in mind, we have preprocessed our dataset and came up with 2 labels.

- Similarity labels:** High, Low, Medium
- Entailment labels:** Entailment, Neutral, Contradiction

3.1 Similarity and Entailment labels

For the similarity labels, we have used the range that came with the dataset. The range starts from -1 and ends at 8. We have divided the whole thing into 4 parts, namely High, Low, Medium and Junk. Junk : -1 Low: 0 to 2 ; Medium: 3 to 5 ; High: 6 to 8. This is the distribution of data after we have mapped into their

respective labels. From the graph, we can clearly see that there is a data imbalance. The number of lower data is tremendously higher than the other labels. For the entailment labels, we have used the other range that came with the dataset. The range starts from -3 and ends at 3. We have divided the whole thing into 3 parts: Entailment, Neutral and Contradiction. Neutral: 0 ; Entailment: 1 to 3 ; Contradiction: -1 to -2.

- **Similarity labels:** High, Low, Medium
- **Entailment labels:** Entailment, Neutral, Contradiction

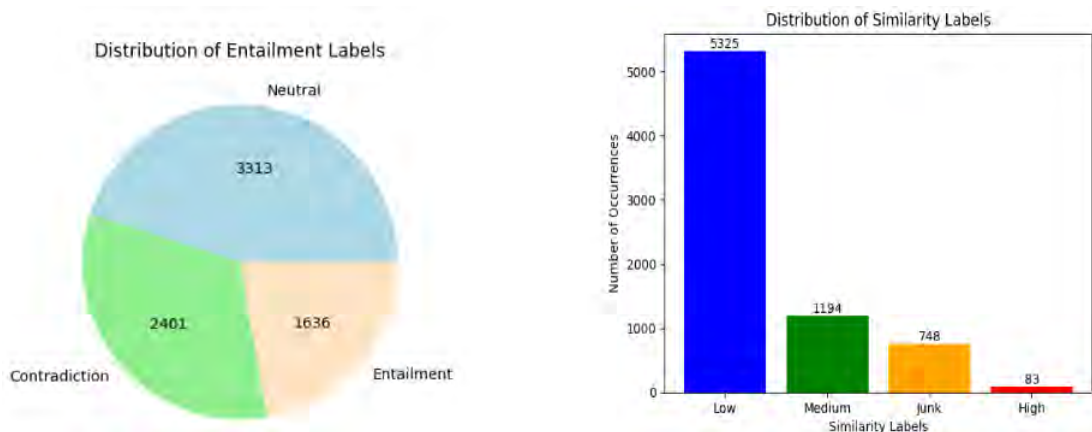


Figure 3.2: Distribution of Entailment and Similarity labels

From the distribution of the entailment labels, we can see that it's quite better than the similarity labels. Number of a single label did not spike leaving the others behind. We saw that in the case of low and high labels of similarity. To fix the

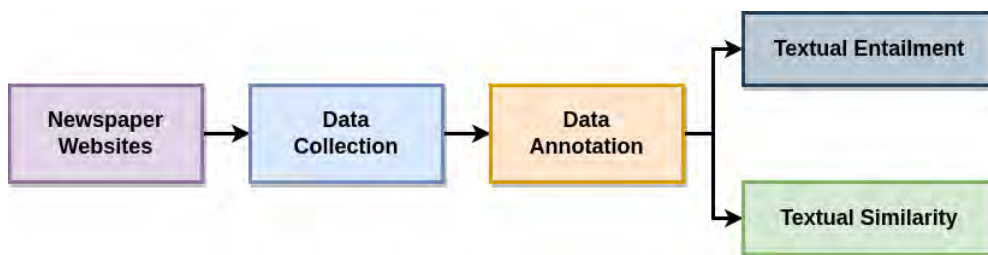


Figure 3.3: Workflow of adding new data

problem of having too few very similar examples in our dataset we looked for more data online. We specifically searched for comments that were highly similar to the newspaper article. After gathering the extra data we annotated and labeled them to improve our dataset.

Now we can see the labels are more balanced than before. We removed all the junk comments from our dataset. And then we started working with this balanced dataset.

using the comments section to talk more about the same stuff. It also means the articles are interesting because they're getting people to comment in a way that relates to what's written.

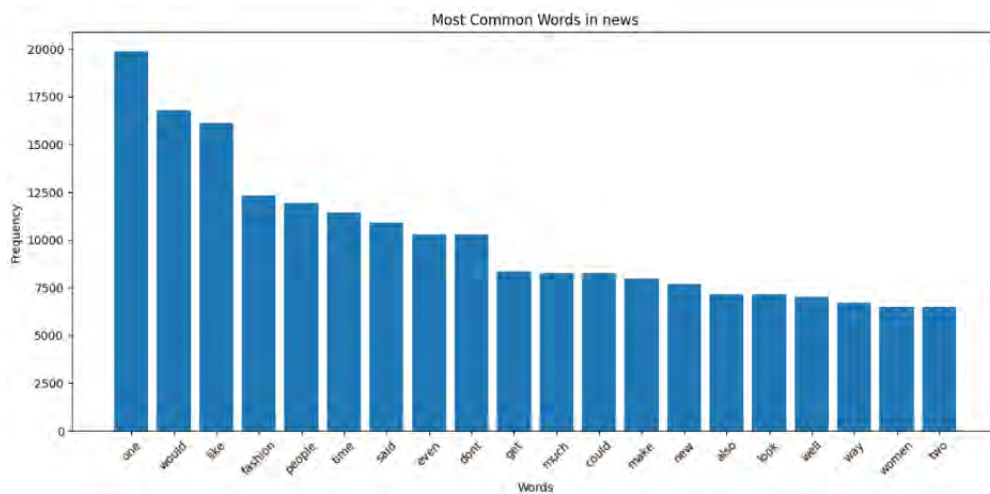


Figure 3.6: Word frequency of News.

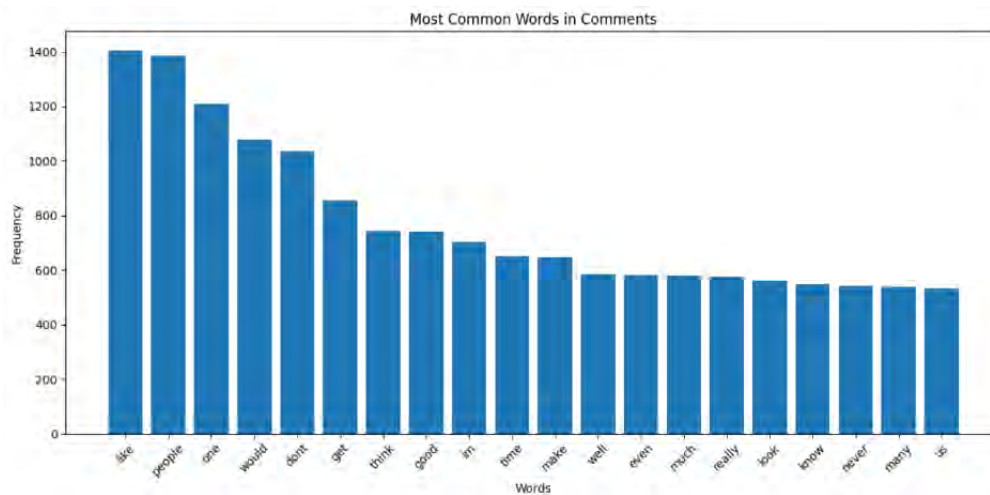


Figure 3.7: Word frequency of Comments.

The same thing is applicable for the word frequency graphs. This suggests good relevance and reader engagement in the comment section. This is exactly what we want since it suggests the number of Junk comments is low.

3.4 Input data Preprocessing

In our text similarity task the labels initially ranged from -1 to 8. We cleaned out irrelevant or unnecessary data. Now the new range of labels are from 0 to 8. To align our data with the Semantic Textual Similarity (STS) we adjusted our labeling by dividing the values by 8. This normalization process made our labeling range now go from 0 to 1 which matches with STS standards. The Semantic Textual Similarity (STS) data we are working with originally ranged from 1 to 5. To suitably prepare this data for our SBERT model we normalized it by dividing all the values by 5.

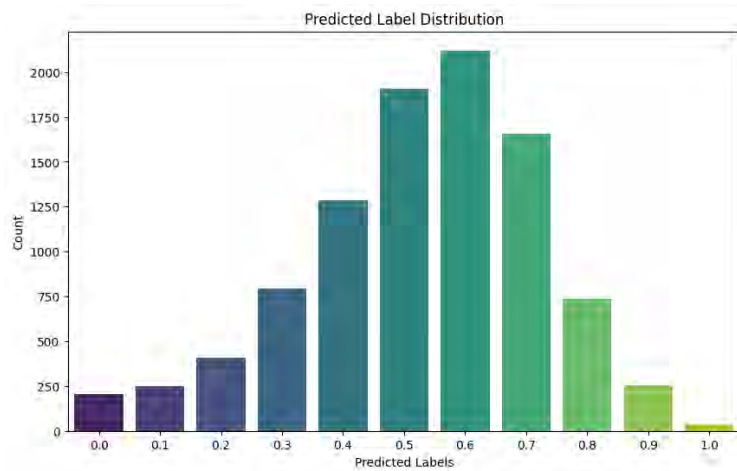


Figure 3.8: Normalized labels of textual similarity

Now we divided the whole dataset into training and testing.

Total data in the main dataset: 9650

Training and Testing Split:

- **Training Data:** 7720 samples
- **Testing Data:** 1930 samples

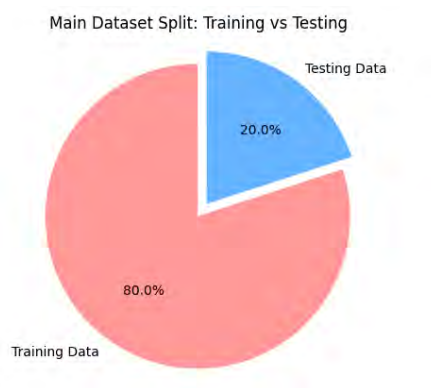


Figure 3.9: Splitting into training and testing datasets

Chapter 4

Methodology

The purpose of our thesis is to identify informative news comments that will further be used to supplement the original news article.

4.1 Overview of the proposed method

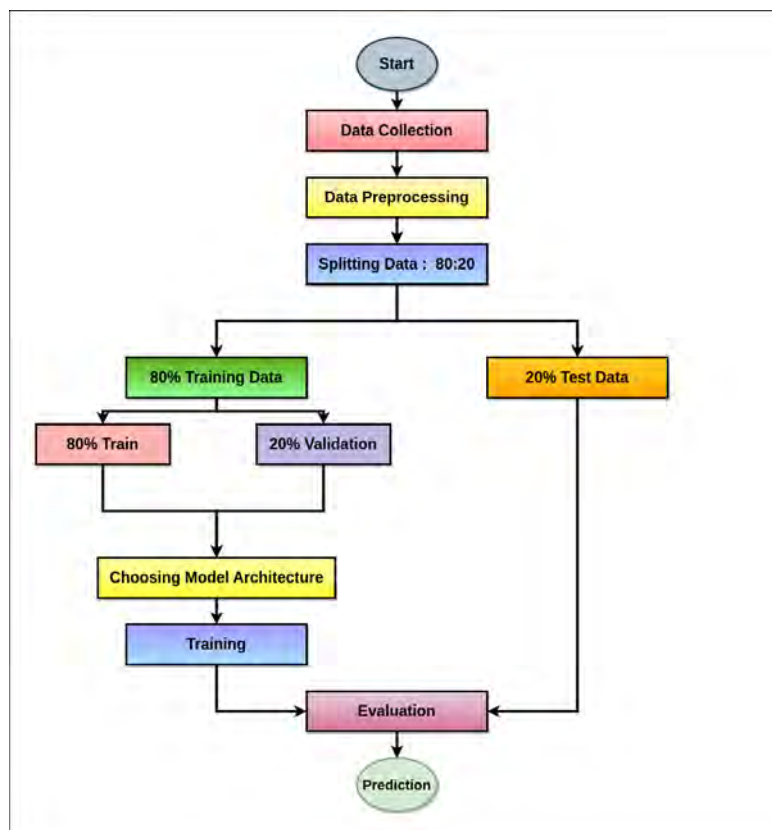


Figure 4.1: Workflow diagram of the proposed methods.

In order to do that, we have built several models-

- a) Single layer Bidirectional LSTM.
- b) Bidirectional LSTM with GloVe embeddings.
- c) RoBERTa

- d) Electra
- e) DistilBERT
- f) Sentence RoBERTa
- e) SBERT

4.2 Model Description

4.2.1 Single layer Bidirectional LSTM

The first layer of this model is an Embedding layer which has an output dimension of 200, representing 200 dimensional vectors for each word. For the next step, we have used a Bidirectional LSTM (Long Short-Term Memory) layer. It is a specialized type of recurrent neural network (RNN) which is designed to capture intricate sequential patterns in the data. By "bidirectional," we mean that this layer processes sequences in both the forward and backward directions. In this Bidirectional LSTM layer we have used 128 LSTM units, each functioning as a memory cell capable of retaining information over variable time steps. This layer is added with a dropout and recurrent dropout of 0.3 which is responsible for capturing sequential information both from forward and backward. Finally, the last layer is a dense layer or fully connected layer with 3 units because of the 3 unique labels in the classification task which are Entailment, Contradiction and Neutral. This layer was performed with the 'softmax' activation layer which is a commonly used activation function in multi class classification tasks. During the training phase, the model undergoes an optimization process to fine-tune its parameters. The Adam optimizer is employed for this purpose because it efficiently adapts the learning rate throughout training and accelerating convergence. For the loss function we have chosen categorical cross-entropy as it is ideal for multi-class classification tasks.

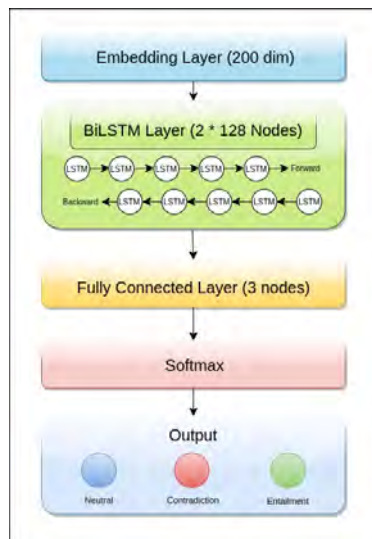


Figure 4.2: Architecture of Single layer Bidirectional LSTM.

The architecture of the Single Layer Bidirectional LSTM Network is shown in figure 4.1

4.2.2 Bidirectional LSTM with GloVe Embeddings

The architecture of the Bidirectional LSTM with GloVe Embeddings and Batch Normalization is shown in figure 3.4

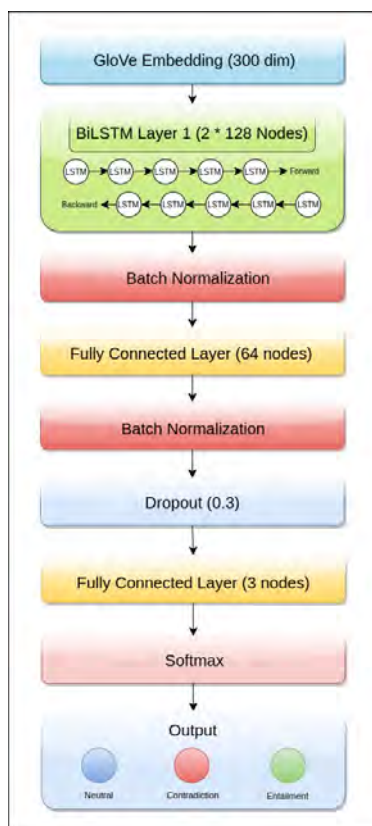


Figure 4.3: Architecture of Bidirectional LSTM with GloVe Embeddings.

The second approach is using GloVe Embedding followed by Bidirectional-LSTM as well as some batch normalization and dense layers added with dropouts. Firstly, some pre-trained word embeddings (GloVe embeddings glove.6B.300d) were used to represent the words as vectors. As the embedding layer is trainable, the model can fine-tune the embeddings during the training process. Secondly, the Bidirectional LSTM was used after the Embedding layer which is responsible for processing the sequences in both forward and backward manner. This layer includes 256 units along with dropout and regularization for improving the performance. Thirdly, after the Bidirectional Layer, comes the batch normalization layer which is performed for speeding up the training process. A dense layer or fully connected layer is also being used after the batch normalization which consists of 64 units and ReLU (Rectified Linear Unit) activation, followed by another batch normalization layer. During the training process, to ensure the model does not face any overfitting issues, a 0.3 dropout layer is added with the model architecture. Finally, the equal number of classes (Entailment, Contradiction and Neutral), that is three classes are used as units in a fully connected dense layer along with softmax activation function to produce class probabilities. This architecture was compiled using the Adam optimizer and for the loss function, categorical cross entropy loss was used.

4.2.3 Electra

We have chosen ELECTRA as our third model. The ELECTRA model is a variant of the transformer architecture developed by Google. It implements the "replaced token detection" technique wherein certain tokens within the text are substituted with a token generator. The model discerns whether the token is original or generated. This self-supervised learning method involves training transformer networks with relatively minimal computational resources by distinguishing between "real" input tokens and "fake" ones. In our process, firstly set the maximum sequence

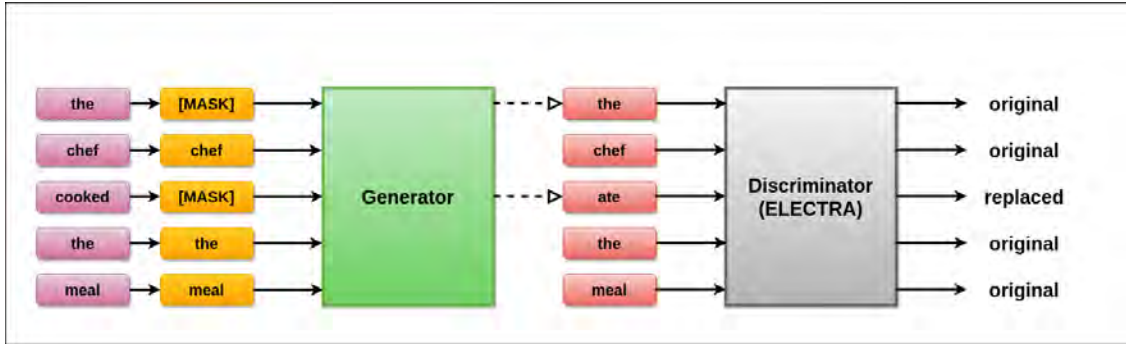


Figure 4.4: Architecture of Electra.

length as 512 for tokenization, balancing context retention and computational efficiency. Longer sequences may offer more context but might demand increased computational resources. Random seed was set to 100 to ensure reproducibility in model training. It allows consistent results across multiple runs. The learning rate determines the step size during model parameter updates. A smaller learning rate permits finer adjustments to model weights during training which potentially leads to better convergence and avoids overshooting the optimal solution. So we set the ideal learning rate as $1e-5$. To optimize memory utilization and training efficiency we have used dynamic batch sizing considering the number of replicas in sync. Sparse Categorical Crossentropy was set as the loss function. Each component, from hyperparameter settings to specific functions, contributes to the model's efficiency, performance and adaptability to the classification task at hand. Fine-tuning hyperparameters and crafting efficient data pipelines are critical aspects that impact the model's learning process and eventual performance in handling text-based classification problems.

| Hyperparameter | Value |
|----------------|--------|
| Max Length | 512 |
| Learning Rate | $1e-5$ |
| Epochs | 30 |
| Batch Size | 16 |

Table 4.1: Electra Model Hyperparameters

4.2.4 RoBERTa Base

RoBERTa (A Robustly Optimized BERT Pretraining Approach) Base, a robust pre-trained language model, stands out for its versatile prowess in handling various

NLP tasks. Built upon the Transformer architecture, this model thrives on a massive dataset of text and code, allowing it to capture intricate linguistic relationships and extract meaningful insights. RoBERTa Base, with its exceptional capabilities in NLI, offers a powerful foundation for building systems that effortlessly navigate the complexities of language

Our final model consists of a RoBERTa Base model that performs contextual embedding of the input token IDs which are then passed to a classifier that will return probabilities for each of the possible three labels "entailment" (0), "neutral" (1), or "contradiction" (2). The classifier consists of a regular densely-connected neural network.

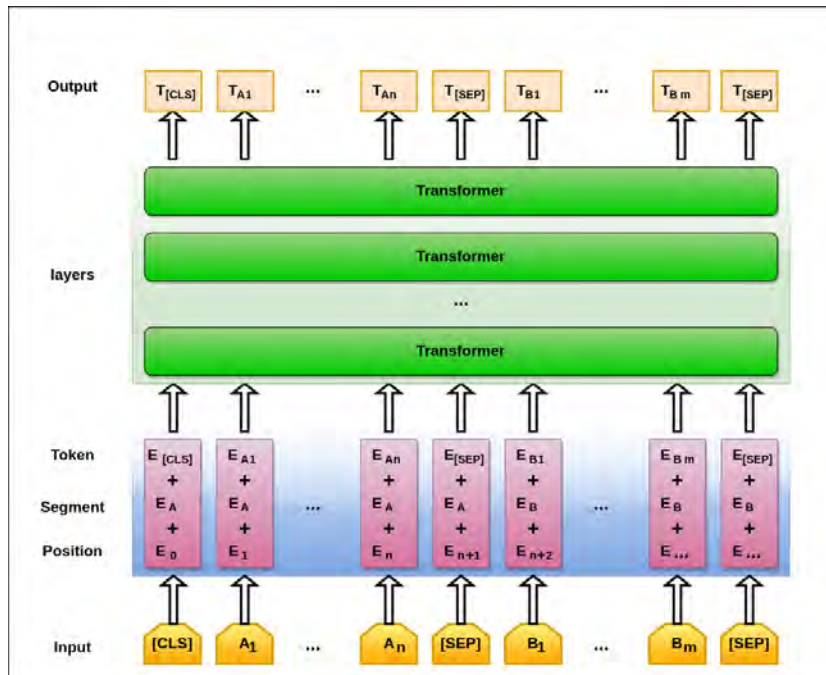


Figure 4.5: Architecture of RoBERTa Base.

Firstly, the RobertaTokenizer is applied to tokenize the newspaper articles and comments from our dataset. The advantage to other types of embeddings is that the RoBERTa embeddings are contextualized. Predictions with contextualized embeddings are more accurate than with non-contextualized embeddings. After receiving the embeddings for the words in our text from RoBERTa, we put them into the classifier which will then in turn return the prediction labels 0,1, or 2.

RoBERTa uses three kinds of input data which are input word IDs, input masks and input type IDs. These allow the model to know that the premise and hypothesis are distinct sentences and also to ignore any padding from the tokenizer.

We set the maximum sequence length of 150 tokens. This tokenization allows for the efficient processing and encoding of the text data. After tokenization and encoding we extracted ids, masks and labels. It facilitates the organization and feeding of data into the subsequent stages of the model implementation. Then we processed our dataset for training and validation purposes with a batch size of 32 and shuffled to enhance model performance. Our model architecture includes dropout layers with a dropout rate of 0.3, dense layers with rectified linear unit (ReLU) activation functions and an output layer with a softmax activation function which aims for

| Hyperparameter | Value |
|----------------|-------|
| Max Length | 150 |
| Dropout Rate | 0.5 |
| Learning Rate | 1e-5 |
| Epochs | 30 |
| Batch Size | 32 |

Table 4.2: RoBERTa-base Model Hyperparameters

multiclass classification (Entailment, Neutral, Contradiction). After that the model is compiled using the Adam optimizer with a learning rate set to 1e-5, employing sparse categorical cross-entropy as the loss function and accuracy as the evaluation metric. The choice of these hyperparameters is based on our multiple trails and errors. We used dropout layers aids in preventing overfitting by randomly dropping neurons during training. A lower learning rate was set to enhance the model’s ability to converge to a better optimum.

Fine-tuning these hyperparameters and architectural choices based on empirical observations and domain expertise enhanced the model’s performance.

4.2.5 SBERT, DistilBERT, RoBERTa

Our proposed model, SBERT is a transformer model made on top of BERT to generate rich sentence level embeddings and predict the similarity among news and comments. The architecture combines a BERT-based model with a pooling layer to generate sentence-level embeddings, followed by a cosine similarity calculation.

The model comprises two main components:

- **BERT Model :** The BERT model is employed to encode the input text, which consists of pairs of news articles and comments. We use the 'bert-base-uncased' variant from the sentence-transformers library. The model tokenizes the input, including special tokens such as [CLS] (classification token) and [SEP] (separator token), and generates 768-dimensional token-level embeddings for each sentence.
- **Pooling Layer:** To obtain sentence-level embeddings, we incorporate a pooling layer from the sentence-transformers library. By default, this layer performs a mean operation on the token embeddings, resulting in a single 768-dimensional vector for each sentence.

Our model adopts a Siamese-type architecture, where two identical sub-networks share the same weights and parameters. This architecture facilitates the processing of pairs of sentences and ensures that the BERT model acts consistently on both the news and comment inputs.

Cosine similarity,

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}} \quad (4.1)$$

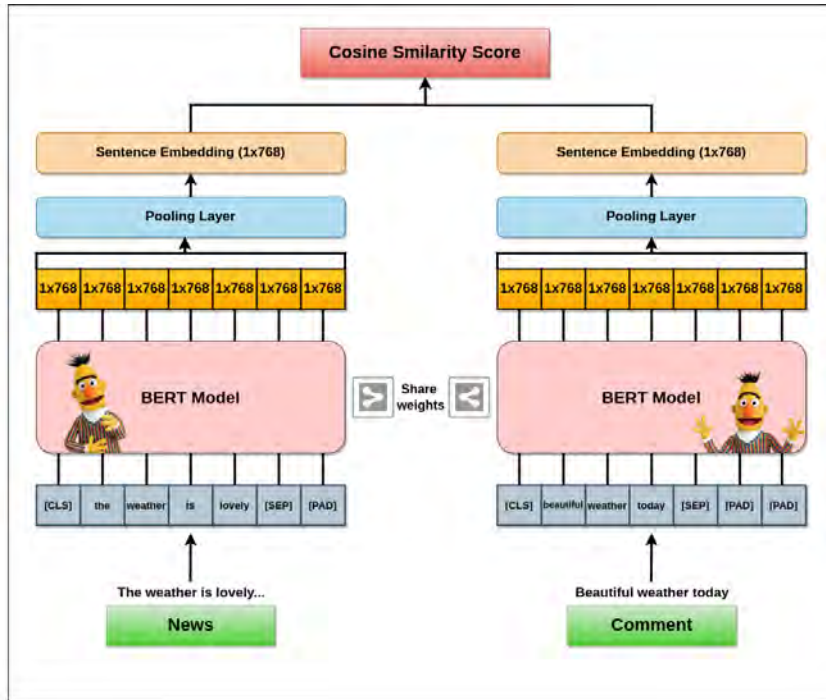


Figure 4.6: Siamese-type architecture of SBERT

The cosine similarity formula serves as a metric for gauging the similarity between two vectors within a high-dimensional space, particularly within the context of our model’s semantic textual similarity analysis. This metric calculates the cosine of the angle formed by two vectors. It reveals whether these vectors align closely in direction. The vectors represent the embeddings of news articles and comments generated by the SBERT model. A high cosine similarity score indicates that the pair of sentences exhibit similar semantic meanings which facilitate the identification of shared contextual information and underlying relationships. This cosine similarity measure proves invaluable in assessing the degree of similarity between news articles and corresponding comments within our dataset. For model development, we fine-tuned the SBERT (Sentence-BERT) architecture on the Semantic Textual Similarity (STS) dataset. The pre-training involved leveraging the STS dataset. This dataset is widely recognized for its diverse and comprehensive sentence similarity annotations. Our objective was to adapt SBERT to capture the semantic relationships present in our news and article. After that we utilized our dataset for model evaluation. This dataset consists of news and comments pairs for which we obtained similarity scores using the fine-tuned SBERT model. These similarity scores were then compared against the ground truth labels present in our dataset. It provides a quantitative assessment of the model’s performance in capturing the semantic similarity between news and comments.

In this scenario all three models, DistilBERT, RoBERTa, and SBERT comes from the powerful BERT architecture. Their strengths and focus areas diverge significantly. DistilBERT and RoBERTa aim to refine BERT’s performance and efficiency for general NLP tasks like text classification and sentiment analysis. They achieve this through modified training processes and reduced model size. This modified training process makes them faster and more lightweight. SBERT takes a different approach which is specialized in generating high-quality sentence embeddings. Its

training revolves around comparing and contrasting sentences that result in embeddings which effectively capture semantic relationships and similarities. So, while DistilBERT and RoBERTa perform better at handling diverse NLP tasks with improved efficiency SBERT shines in understanding and comparing the meaning of sentences at a deeper level.

Algorithm 1 Scoring mechanism of the implemented SBERT

```

1: testArray  $\leftarrow$  [] {Predicted by the model}
2: testActual  $\leftarrow$  [] {Original Labels}
3: testDataset  $\leftarrow$  ReadCSV("Similarity Dataset.csv")
4: pairs  $\leftarrow$  zip(testDataset['sentence1'].tolist(), testDataset['sentence2'].tolist())

5: testActual.extend(testDataset['similarity_score'].tolist())
6: bin_edges  $\leftarrow$  [0, 0.33, 0.66, 1.0]
7: bin_labels  $\leftarrow$  ['Low', 'Medium', 'High']
8: testActual_bins  $\leftarrow$  pd.cut(testActual, bins=bin_edges, labels=bin_labels, include_lowest=True)
9: for i in pairs do
10:   score  $\leftarrow$  round(predict_similarity(i), 2)
11:   testArray.append(score)
12: end for
13: bin_edges  $\leftarrow$  [-1, -0.333, 0.333, 1]
14: bin_labels  $\leftarrow$  ['Low', 'Medium', 'High']
15: testArray_bins  $\leftarrow$  pd.cut(testArray, bins=bin_edges, labels=bin_labels)
16: c  $\leftarrow$  0
17: for i in range(len(testArray_bins)) do
18:   if testArray_bins[i] == testActual_bins[i] then
19:     c  $\leftarrow$  c + 1
20:   end if
21: end for
22: accuracy  $\leftarrow$   $\frac{c}{\text{len}(\text{testArray\_bins})} \times 100\%$ 

```

4.3 Model Components

4.3.1 LSTM

A Recurrent Neural Network (RNN) type, Long Short-Term Memory (LSTM) has the capability to efficiently capture long-term dependencies in sequential data. A standard RNN processes sequential data by passing the information from one time step to the next through a hidden state, however, this hidden state can only remember information for a short period of time. This makes it difficult for RNNs to accurately model data with long-term dependencies, such as in natural language processing tasks where the meaning of a word depends on the context of the entire sentence. LSTMs address this problem by introducing an additional memory cell, which can store information over a longer period of time. A memory cell in LSTM is managed by three gates: input, forget, and output gates. The input gate decides the

portion of newly arrived data being saved in the memory cell, the forget gate determines the amount of previous information to be kept and the output gate controls the amount of information to be passed on to the following time step. Additionally, LSTMs introduce a new structure called the "hidden state" which is used to store information that is passed from one-time step to the next. At each time step, the hidden state is revised according to the current input, previous hidden state and current memory cell state. A single cell of LSTM is shown in figure 3.2

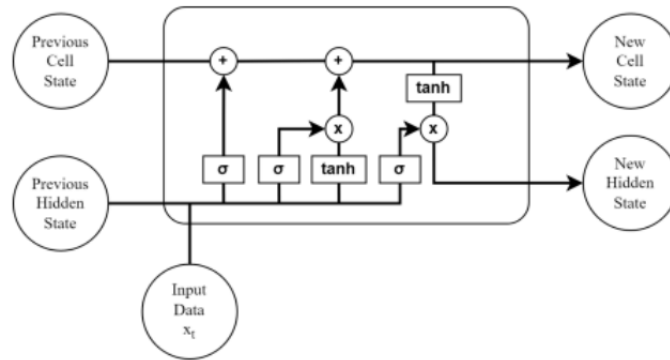


Figure 4.7: LSTM figure.

The architecture of the Single Layer Bidirectional LSTM Network is shown in figure 4.2.

The ability of LSTMs to efficiently capture long-term dependencies in sequential data makes them useful for a variety of tasks, including speech recognition, natural language processing, and other sequential data processing tasks.

4.3.2 GloVe Embeddings

GloVe stands for global vectors for word representation. It is an unsupervised learning algorithm developed by Stanford for generating word embeddings by aggregating global word-word co-occurrence matrices from a corpus. The resulting embeddings show interesting linear substructures of the word in vector space.

The underlying concept that distinguishes man from woman, i.e. sex or gender, may be equivalently specified by various other word pairs, such as king and queen or brother and sister. To state this observation mathematically, we might expect that the vector differences between man - woman, king - queen, and brother - sister might all be roughly equal. This property and other interesting patterns can be observed in the above set of visualizations.

4.3.3 Batch Normalization

Batch Norm is a normalization technique done between the layers of a Neural Network instead of in the raw data. It only adds another network layer between one hidden layer and the following hidden layer. Prior to sending them on as the input of the next hidden layer, it has the responsibility of normalizing the outputs from the previous hidden layer.

4.3.4 BERT Tokenization

BERT (Bidirectional Encoder Portrayals from Transformers) utilizes a tokenization procedure known as WordPiece tokenization to handle text based information. This strategy includes separating words into more modest subword units to deal with OOV words and better catch the profundity of language. The BERT comprises a huge number of subword tokens which were made during pre-preparing. It includes familiar words, subwords, prefixes, and postfixes. The tokenization cycle in BERT follows a few stages. At first, the info text is portioned into essential tokens including words, accentuation and whitespace. Every fundamental symbol then goes through WordPiece tokenization and further partitions it into subword tokens. Unique tokens like [CLS] (start of arrangement) and [SEP] (separator between sentences) are acquainted with help with understanding sentence connections for different NLP assignments. In order to guarantee consistent sequence lengths, BERT employs padding and truncation, especially in scenarios involving batch processing. Once tokenized then every token is planned to a comparing vector portrayal inside BERT's pre-prepared embeddings. It catches semantic data and logical connections. In addition, BERT incorporates positional embeddings to encode token positions within sequences and segment embeddings to differentiate between sentences in a pair. By and large, BERT's tokenization system changes crude message into a configuration reasonable for the model's feedback that empowers it to understand language subtleties, handle OOV words, and create thorough portrayals of literary information.

4.3.5 ELECTRA Tokenization

The ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) model acquaints a novel methodology with pre-preparing text encoders inside the domain of regular language handling (NLP). Created by researchers at Google, ELECTRA proposes an exceptional preparation strategy that varies from the customary generative pre-preparing models. In the traditional pre-preparing models like GPT where the model figures out how to foresee the following word in a sentence or to remake covered tokens inside the message. Text sequences must be generated by the model in this generative approach, which can be time- and cost-intensive. ELECTRA takes on a discriminative pre-preparing strategy as opposed to a generative one. The center thought behind ELECTRA includes preparing a discriminator to recognize "real" tokens from the text and "fake" or supplanted tokens. It utilizes a changed rendition of the GAN (Generative Adversarial Networks) system where a generator is prepared to supplant a few tokens in the info text and a discriminator is all the while prepared to separate between the first tokens and the supplanted ones. The ELECTRA model's tokenization process involves replacing a predetermined proportion of the input text's tokens with generated tokens. For instance, irregular tokens in the information succession are subbed with tokens produced by the model. Then the discriminator is prepared to perceive among genuine and supplanted tokens and is entrusted with figuring out which tokens in the grouping are valid and which ones are supplanted/created. The generator's goal is to create substitutions that the discriminator is bound to mark as genuine tokens. Basically provoking the discriminator to turn out to be better at recognizing genuine and counterfeit tokens. This antagonistic preparation arrangement urges the gener-

ator to produce more practical substitutions while pushing the discriminator to turn out to be seriously knowing. Concerning explicitly, the substitution of tokens is a significant stage in ELECTRA's preparation cycle. It includes subbing a negligible part of tokens with model-produced tokens, in this way making a changed contribution for the discriminator. Through this adversarial procedure, the model fine-tunes the discriminator and generator in order to produce a text representation that is both more effective and accurate. ELECTRA's tokenization process, which sets it apart from traditional generative pre-training models in NLP by replacing tokens in the input text with tokens generated by the model, enables adversarial training between the generator and discriminator and improves text encoding capabilities without requiring extensive text generation.

4.3.6 RoBERTa Tokenization

ROBERTa- the abbreviated form of Robustly Optimized BERT approach is a state of the art natural language processing (NLP) model built on transformer architecture. This Facebook AI developed model is designed to fix some shortcomings of the original BERT model. One significant aspect of ROBERTa is how it tokenizes its text. Tokenization consists in dividing a text into individual tokens or subwords, which then serve as input for the model. Formally, in ROBERTa's tokenization, the input text is first segmented into words and these words are further broken down into subword tokens using a Byte Pair Encoding (BPE) tokenizer. BPE is a data compression method that recursively merges the most frequent pairs of consecutive bytes within a sequence, effectively creating a vocabulary that adapts to the specific phonetic details of the dataset. However, this tokenization strategy makes RoBERTa be more adaptable to different linguistic contexts and variations that improves its overall robustness in various NLP tasks.

4.3.7 DistilBERT Tokenization

DistilBERT is another version of BERT made with simplicity and computational efficiency in mind while maintaining high-performance scores on typical NLP tasks. DistilBERT accomplishes this proficiency to a limited extent through a worked on tokenization technique. Tokenization, the most common way of changing over a text into more modest units or tokens, assumes a vital part in language model preparation. In DistilBERT's tokenization, the info text is sectioned into subwords utilizing the WordPiece tokenization calculation. This calculation partitions words into more modest units, empowering the model to deal with intriguing or out-of-vocabulary words really. Furthermore, DistilBERT utilizes a vocabulary decrease method during preparation, where it holds just a

Chapter 5

Result analysis

5.1 Evaluation metrics

For evaluating and analyzing our results for the Textual Entailment and Text similarity task, we took help from several evaluation metrics. Such as,

- **Accuracy:** Accuracy is a fundamental metric in classification tasks that measures the overall correctness of a model. It is calculated as the ratio of correctly predicted instances to the total number of instances. It assesses the model's ability to correctly classify both positive and negative instances.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.1)$$

- **Precision:** Precision is a crucial metric in binary and multiclass classification that focuses on the accuracy of positive predictions. The computation is done as a fraction of the accurate positive instances which were predicted to be positive. Precision is especially useful where the cost of a false alarm is high, such as in medical diagnosis where high precision would mean that healthy people will be less likely to be classified ill.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (5.2)$$

- **Recall:** Recall evaluates the capacity of a model to capture all relevant positive instances. This is evaluated by calculating the proportion of accurately predicted positive instances out of all actual positive cases. Recall has significance when it comes to situations where missing a classifying error for a negative instance is expensive.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (5.3)$$

- **F1 score:** The F1 Score acts as a balancing act between precision and recall. F1 score provides an overall performance measure for a classifier model. It

is specifically beneficial when there are imbalanced class distributions. The range criterion for the F1 Score ranges from 0 to 1 with 1 representing perfect precision and recall.

$$\text{F1 score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5.4)$$

5.2 Result for Different Model Architectures

In our study we implemented five diverse models for the Textual Entailment Task. It was observed that models employing hyperparameter tuning and leveraging pre-trained models significantly outperformed traditional RNN-based models. The research showed that using pre-made models like Roberta and Electra is much better than using the RNN-based models. These pre-made models were already trained on a lot of data and had special ways of understanding words in sentences. And for that reason they are way better at predicting the labels right compared to the older models like RNNs.

5.2.1 Result for Single Layer Bidirectional LSTM

The result from first model (Single layer Bidirectional LSTM) :

We started our Textual Entailment task with a Single Layered BiLSTM model.

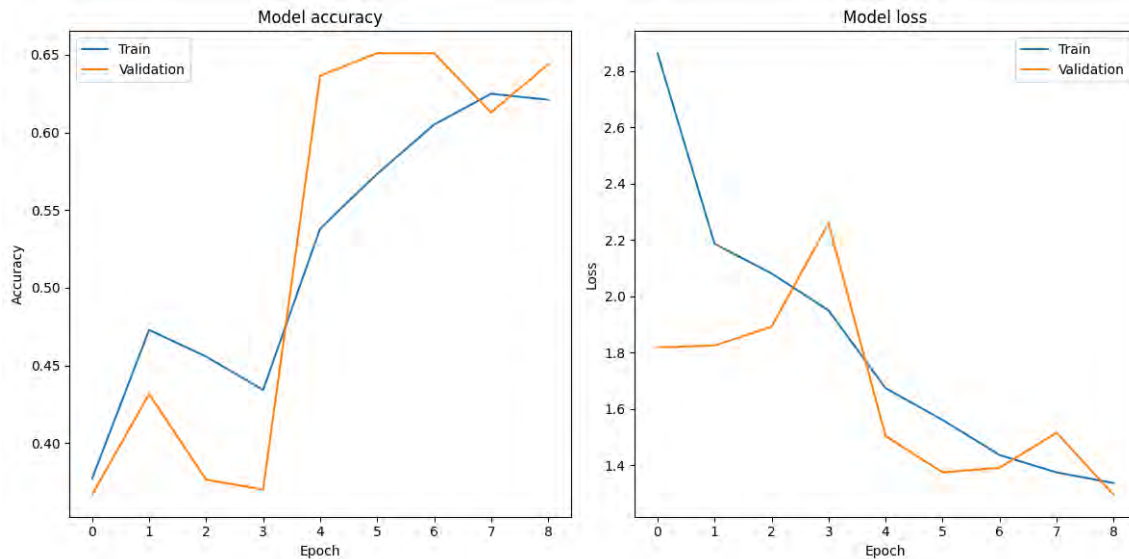


Figure 5.1: Single Layer Bidirectional LSTM Training Results

The highest validation accuracy our first model received is 64.37%. This lack of improvement in the validation accuracy curve indicated inability of the model to learn further from the data. After training for 9 epochs we generated a confusion matrix graph to observe how many labels were correctly classified.

From the confusion matrix we can see that our model correctly predicted 513 Entailment labels, 0 Neutral labels and 520 Contradiction labels. We can also see that 241 Neutral labeled and 99 Contradiction labeled data were misclassified as Entailment.

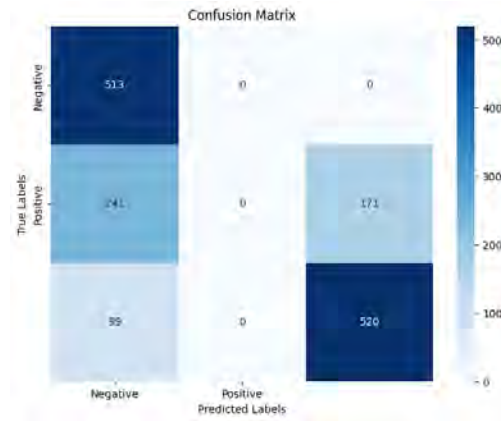


Figure 5.2: Confusion matrix for Single Layer Bidirectional LSTM

Our basic RNN model could not classify a single Neutral labeled data. Moreover 171 Neutral labeled data were misclassified as Entailment.

5.2.2 Result for Bidirectional LSTM with GloVe Embeddings

The result from second model (Bidirectional LSTM with GloVe Embeddings) : After observing the shortcomings of our initial model, we developed a second model as an improvement strategy. This second model is also a RNN model: Bidirectional LSTM with 300-dimensional GloVe embeddings.

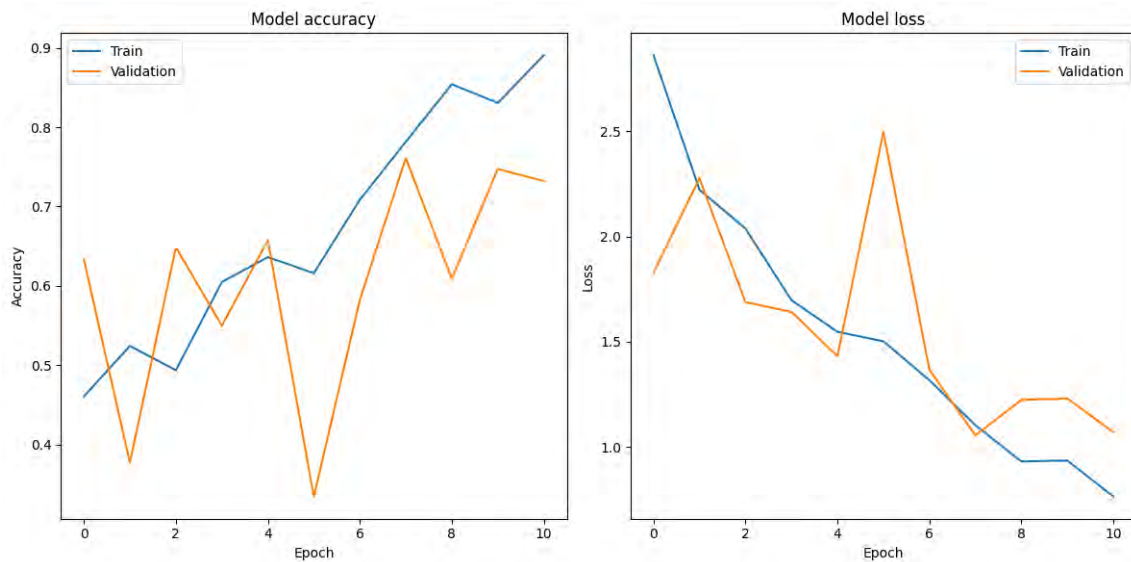


Figure 5.3: Bidirectional LSTM with GloVe Embeddings Training Results

The second model displays more promising signs in terms of accuracy and loss in the initial epochs. The highest validation accuracy achieved is around 73.2%. From the confusion matrix we can see that 512 Entailment labeled data were correctly predicted and 2 Neutral labeled data were misclassified. Only 72 Neutral labeled data were correctly predicted while 338 Neutral labeled data were misclassified. Lastly 616 Contradiction labeled data were correctly predicted by our model.

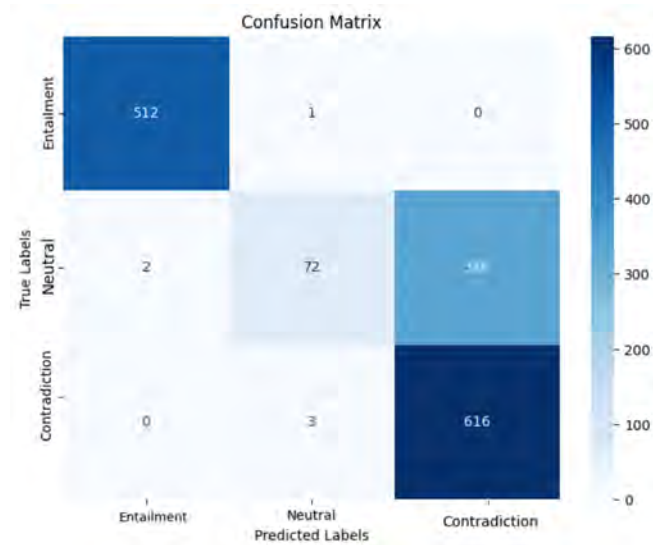
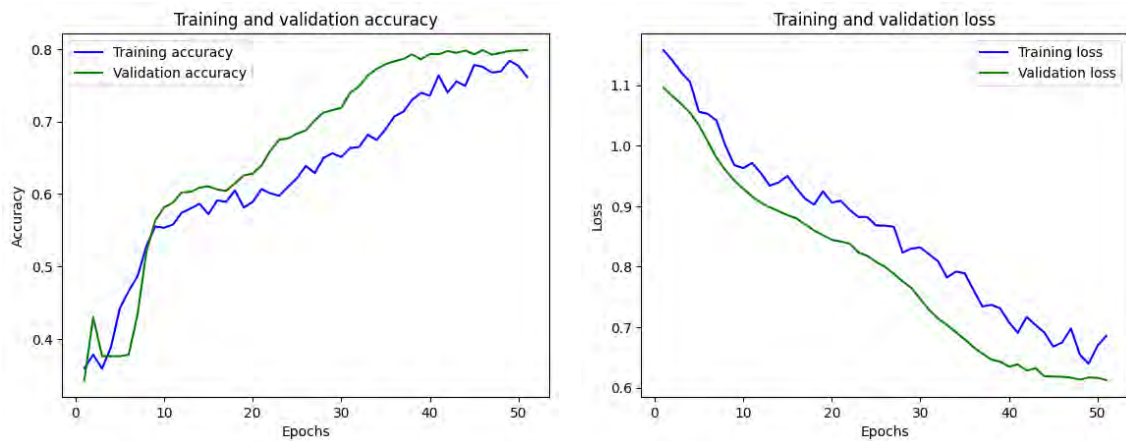


Figure 5.4: Confusion matrix for Bidirectional LSTM with GloVe Embeddings

5.2.3 Result for Fine tuned RoBERTa without NLI Dataset

The result from third model (Fine tuned RoBERTa Base Result) : For our third model, we shifted from RNN models to pre-trained models.



(a) Training and Validation Accuracy

(b) Training and Validation Loss

Figure 5.5: Fine tuned RoBERTa (Without NLI Dataset) Training Results

Our third model’s ability to correctly classify instances within the validation dataset with a notable accuracy of 79.86%.

From the confusion matrix of the third model, we can see 642 data labeled as Entailment, 578 data labeled as Neutral and 264 data labeled as Contradiction were accurately classified.

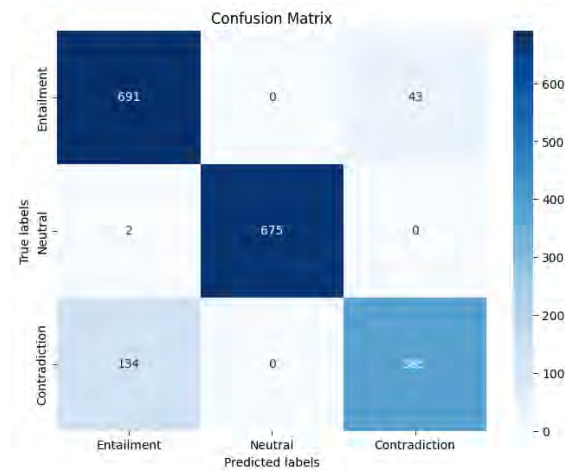
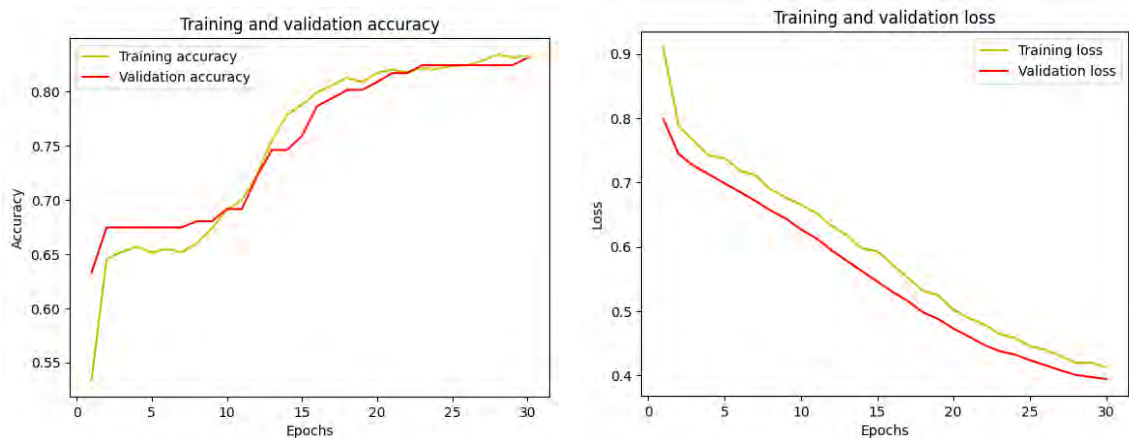


Figure 5.6: Confusion matrix for Fine tuned RoBERTa (Without NLI Dataset).

5.2.4 Result for Fine Tuned ELECTRA

The result from fourth model (Fine Tuned ELECTRA) : For our fourth model we used a MLM pre-trained model named ELECTRA.



(a) Training and Validation Accuracy

(b) Training and Validation Loss

Figure 5.7: Fine Tuned ELECTRA Training Results

Coupled with a larger dataset, our fourth model: ELECTRA achieved its highest accuracy at 83.11%. Our fourth model got much better compared to the earlier ones.

From the confusion matrix of the third model, we can see 622 data labeled as Entailment, 677 data labeled as Neutral and 292 data labeled as Contradiction were accurately classified.

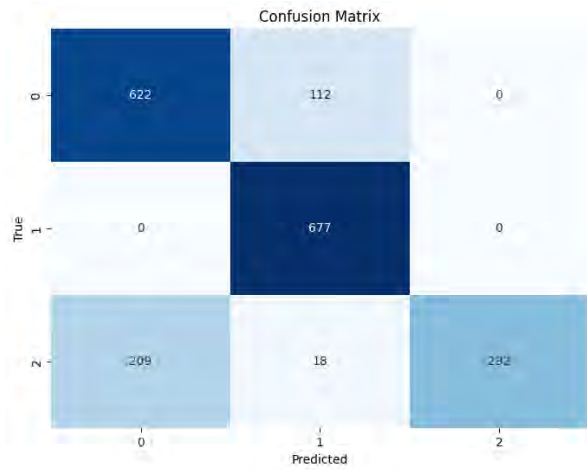


Figure 5.8: Confusion matrix for Fine Tuned ELECTRA

5.2.5 Result for Fine tuned RoBERTa with NLI Dataset

For our last model, we used a Fine Tuned RoBERTa base with concatenation of NLI data.

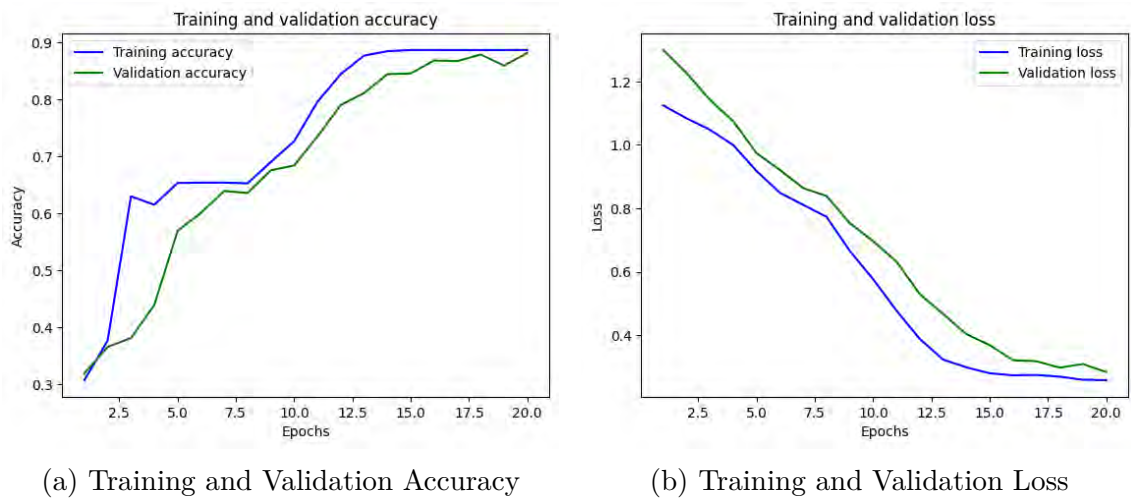


Figure 5.9: Fine tuned Fine tuned RoBERTa with NLI Dataset Results

In our latest model, we achieved the highest accuracy of 88.60%. This model outperformed all our previous models.

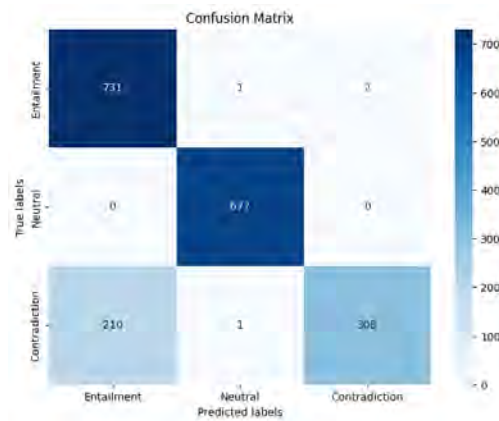


Figure 5.10: Confusion matrix for Fine tuned RoBERTa (With NLI Dataset).

The confusion matrix for the third model reveals that it accurately classified 622 instances labeled as "Entailment," 677 instances labeled as "Neutral" and 292 instances labeled as "Contradiction."

5.2.6 Result of DistilBERT

Upon testing our test dataset DistilBERT achieved an accuracy of 66.37%.

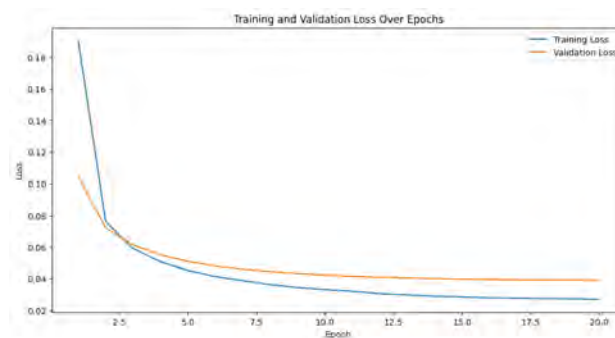


Figure 5.11: DistilBERT Training Results

5.2.7 Result of Sentence RoBERTa

RoBERTa resulted in a slightly lower accuracy of 65.69%.

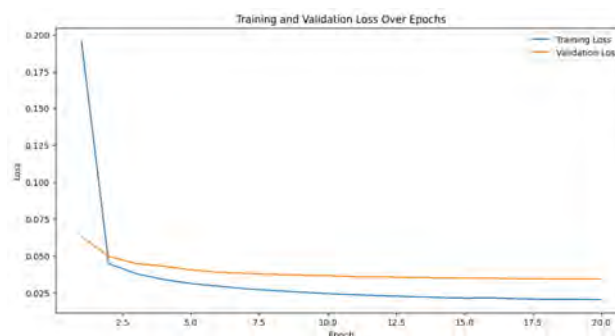


Figure 5.12: RoBERTa Training Results

5.2.8 SBERT

SBERT outperformed both models with an accuracy of 68.49%.

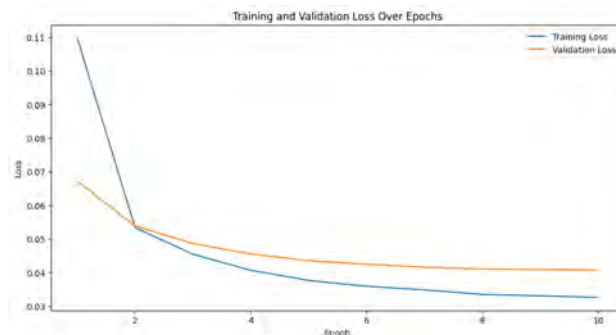


Figure 5.13: SBERT Training Results

5.3 Comparison

5.3.1 Comparison for Textual Entailment

- **Single Layer Bidirectional LSTM:** Our first model, a Single Layer Bidirectional LSTM received a validation accuracy of 64.37% but it struggled with instability in learning, erratic behavior in metrics and most notably it was unable to predict any “Neutral” labels. This inconsistency signifies challenges in convergence or difficulties in the model’s learning process. And by analyzing the confusion matrix we observed that despite ensuring a balanced distribution across all classes within the dataset, our model could not correctly classify instances for Neutral. This indicates that the fault lies within the model’s capability to adequately discern and differentiate features specific to the ‘Neutral’ class.
- **Bidirectional LSTM with Glove Embeddings:** To overcome those problems we came up with another Bidirectional LSTM model. But this time we used Glove Embeddings. It was introduced to address the limitations identified in our first model’s performance. We saw an improvement with a validation accuracy of 73.2% but still faced challenges in accurately classifying labels.

Then we shifted to using pretrained models like ELECTRA and RoBERTa. The reason we switched from RNN models to pretrained models is that RNNs had trouble understanding the complex ways words work together in sentences. They could not catch all the different meanings and relationships between words very well. On the other hand, pretrained models already learned a lot from reading many different types of texts. They are better at understanding how words fit together and what they mean in different situations. This made them better at understanding sentences and their meanings overall.

- **Fine Tuned RoBERTa:** Our first transformer model RoBERTa did not have a lot of data to learn from - only 7719 training data. Even though the increase in accuracy was not huge, the RoBERTa model helped us to fix the overfitting

problem that we faced with our previous models. RoBERTa is doing a better job at predicting compared to the RNN models. It shows improvements in how well it understands and predicts the relationships between sentences. It achieved a validation accuracy of 79.86%.

- **Fine Tuned ELECTRA with NLI Data concatenation:** We had a breakthrough idea when we discovered additional NLI (Natural Language Inference) datasets available online which was similar to our dataset format. We combined these new datasets with our existing training data. This helped our model to learn more effectively. And this time we used another pretrained model ELECTRA, developed by GOOGLE. The impact was significant: our validation accuracy reached 83.11% which surpassed the performance of our previous models by a considerable margin. The reason behind this improvement was the utilization of a larger dataset that enabled our model to learn more accurately. Moreover, the occurrences of misclassification reduced notably with the ELECTRA model.
- **Fine Tuned RoBERTa with NLI Dataset:** For our last model we again used RoBERTa. But this time we trained it with more NLI data. This ultimate model achieved an accuracy of 88.60%. The significant improvement in accuracy was attributed to training the model with a larger dataset. Additionally, we conducted hyperparameter tuning which proved surprisingly impactful in achieving this highest score. This combined approach of using a larger dataset along with fine-tuning model parameters played a crucial role in achieving the best performance.

| Model Name | Acc | F1 | Precision | Recall |
|--|--------|--------|-----------|--------|
| Single Layer Bidirectional LSTM | 64.37% | 52% | 45% | 61% |
| Bidirectional LSTM with GloVe Embeddings | 73.20% | 69% | 86% | 72% |
| Fine tuned RoBERTa without NLI Dataset | 79.86% | 78% | 77% | 80% |
| Fine Tuned ELECTRA | 83.11% | 81.60% | 84.78% | 82.4% |
| Fine tuned RoBERTa with NLI Dataset | 88.60% | 87% | 92% | 86% |

Table 5.1: Comparison of accuracy between different entailment models

In conclusion, the shift from RNNs to pretrained transformers resulted in remarkable improvements in the Textual Entailment task. The significant improvement in accuracy and reduced misclassification of data underscore the transformative impact of leveraging advanced models trained on diverse datasets. It marks a pivotal advancement in accurately understanding and predicting relationships between textual sentences.

In our study focusing on textual similarity, we employed SBERT. We also used DistilBERT and RoBERTa but SBERT got the highest accuracy of 68.49%. Due to the necessity of normalization to align our dataset with the Semantic Textual Similarity (STS) benchmark dataset, some data underwent modifications which resulted in data loss. Before merging our dataset with the STS (Semantic Textual Similarity) dataset, we encountered suboptimal results in our task of measuring textual similarity. But after combining the dataset for training we observed a significant

improvement in accuracy. So it highlights the effectiveness of merging these datasets in enhancing the performance of our model.

5.3.2 Comparison for Text Similarity

In our study focusing on textual similarity, we employed SBERT. We also used DistilBERT and RoBERTa but SBERT got the highest accuracy of 68.49%. Due to the necessity of normalization to align our dataset with the Semantic Textual Similarity (STS) benchmark dataset, some data underwent modifications which resulted in data loss. Before merging our dataset with the STS (Semantic Textual Similarity) dataset, we encountered suboptimal results in our task of measuring textual similarity. But after combining the dataset for training we observed a significant improvement in accuracy. So it highlights the effectiveness of merging these datasets in enhancing the performance of our model.

| Model Name | Accuracy |
|------------------|----------|
| DistilBERT | 66.37% |
| Sentence RoBERTa | 65.69% |
| SBERT | 68.49% |

Table 5.2: Comparison of accuracy between different similarity models

5.4 Limitations & Future works

We will use ensemble models to enhance the comment moderation system’s performance. Ensemble models involve combining predictions from multiple NLP models to achieve a more robust and accurate outcome. By integrating various models each with its strengths and weaknesses, the overall system can benefit from their complementary capabilities that will lead to improved moderation accuracy. Additionally, introducing ensemble techniques helps mitigate the risk of relying too heavily on a single model and it ensures better generalization to diverse comment types and contexts.

Secondly, another crucial aspect to consider when it comes to improving comment moderation is expanding the dataset. Increasing the diversity of data by including examples from multiple sources can help in creating a more complete training set. This will not only expose the models to more language patterns and contexts but also minimize overfitting. Overfitting takes place when a model becomes too specialized in training data thereby negatively affecting its performance on new and unseen data. By having more diverse examples, models will have a better understanding leading to improved moderation outcomes for a wider range of comments.

Addressing the inherent limitation of the current model’s token limit is essential for handling longer textual inputs effectively. Right now, it can only understand up to 512 words, which can be a problem with lengthier comments. We will try to use different models that can take up to 1024 tokens. It will help to understand the news and its corresponding comments more accurately and thus help to find the entailment relationship between them.

Also, it's crucial to teach our models to understand more languages. Currently, they only know English but if we teach them other languages too, they can be better at understanding comments from all over the world. This makes our comment extraction more useful for handling conversations in different languages and dealing with different ways people express themselves. So, by combining different models, getting more examples, adjusting the token length limit and adding more languages we can make our models better.

5.5 Conclusion

Newspaper comment sections can be a great source of information which can further be used to supplement the news article. Till now, assigning the Editor's Pick flag to comments is done manually at large news outlets. This process is time-consuming and subject to human biases. In order to solve this issue, in this paper we have used state of the art NLP techniques like RNN, LSTM, SBERT, DistilBERT, RoBERTa, ELECTRA and different fine-tuned versions of it. We have used these models to calculate Textual Similarity, Entailment. Using these measurements, we can predict if a comment is informative enough to aid the news article. In conclusion, this paper proposes automating comment moderation and extracting informative insights from newspaper comment sections. The method optimizes workflows, enhances user involvement, and boosts readability of news.

Bibliography

- [1] P. Neculoiu, M. Versteegh, and M. Rotaru, “Learning text similarity with Siamese recurrent networks,” in *Proceedings of the 1st Workshop on Representation Learning for NLP*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 148–157. DOI: 10.18653/v1/W16-1617. [Online]. Available: <https://aclanthology.org/W16-1617>.
- [2] N. Reimers and I. Gurevych, *Sentence-bert: Sentence embeddings using siamese bert-networks*, 2019. arXiv: 1908.10084 [cs.CL].
- [3] Y. Yang, S. Yuan, D. Cer, *et al.*, *Learning semantic textual similarity from conversations*, 2018. arXiv: 1804.07754 [cs.CL].
- [4] W. Yin, N. F. Rajani, D. Radev, R. Socher, and C. Xiong, *Universal natural language processing with limited annotations: Try few-shot textual entailment as a start*, 2020. arXiv: 2010.02584 [cs.CL].
- [5] A. Poliak, *A survey on recognizing textual entailment as an nlp evaluation*, 2020. arXiv: 2010.03061 [cs.CL].
- [6] P. Kapanipathi, V. Thost, S. S. Patel, *et al.*, *Infusing knowledge into the textual entailment task using graph convolutional networks*, 2019. arXiv: 1911.02060 [cs.CL].
- [7] H. Luo and J. Glass, *Logic against bias: Textual entailment mitigates stereotypical sentence reasoning*, 2023. arXiv: 2303.05670 [cs.CL].
- [8] K. Toutanova and D. Chen, “Observed versus latent features for knowledge base and text inference,” in *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 57–66. DOI: 10.18653/v1/W15-4007. [Online]. Available: <https://aclanthology.org/W15-4007>.
- [9] L.-H. Lee, Y. Lu, P.-H. Chen, P.-L. Lee, and K.-K. Shyu, “Ncuae at medqa 2019: Medical text inference using ensemble bert-bilstm-attention model,” in *Proceedings of the 18th BioNLP workshop and shared task*, 2019, pp. 528–532.
- [10] C. Qian, F. Feng, L. Wen, C. Ma, and P. Xie, “Counterfactual inference for text classification debiasing,” in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online: Association for Computational Linguistics, Aug. 2021, pp. 5434–5445. DOI: 10.18653/v1/2021.acl-long.422. [Online]. Available: <https://aclanthology.org/2021.acl-long.422>.

- [11] D. Bär, T. Zesch, and I. Gurevych, “A reflective view on text similarity,” in *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, 2011, pp. 515–520.
- [12] O. Shahmirzadi, A. Lugowski, and K. Younge, “Text similarity in vector space models: A comparative study,” in *2019 18th IEEE international conference on machine learning and applications (ICMLA)*, IEEE, 2019, pp. 659–666.
- [13] Y. Li, D. McLean, Z. A. Bandar, J. D. O’shea, and K. Crockett, “Sentence similarity based on semantic nets and corpus statistics,” *IEEE transactions on knowledge and data engineering*, vol. 18, no. 8, pp. 1138–1150, 2006.
- [14] A. Sotnikova, Y. T. Cao, H. Daumé III, and R. Rudinger, “Analyzing stereotypes in generative text inference tasks,” in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 4052–4065.
- [15] R. Nairn, C. Condoravdi, and L. Karttunen, “Computing relative polarity for textual inference,” in *Proceedings of the fifth international workshop on inference in computational semantics (icos-5)*, 2006.
- [16] Y. Miao, L. Yu, and P. Blunsom, “Neural variational inference for text processing,” in *International conference on machine learning*, PMLR, 2016, pp. 1727–1736.
- [17] B. MacCartney and C. D. Manning, “Natural logic for textual inference,” in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, 2007, pp. 193–200.