# Automated Reference Validation for Scholarly Publications using NLP

by

A S M Nasim Khan
19101623
Mohammad Nasif Sadique Khan
19201084
MD. Adnan Howlader
19201076
Ayan Roy
19201043

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 and Engineering

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
January 2024

# Declaration

It is hereby declared that

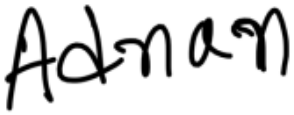
1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material 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:**



A S M Nasim Khan
19101623

Mohammad Nasif Sadique Khan
19201084



MD. Adnan Howlader
19201076

Ayan Roy
19201043

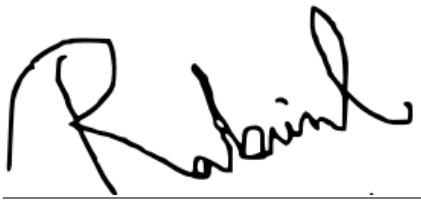
# Approval

The thesis/project titled "Automated Reference Validation for Scholarly Publications using NLP" submitted by

1. A S M Nasim Khan(19101623)
2. Mohammad Nasif Sadique Khan(19201084)
3. MD. Adnan Howlader(19201076)
4. Ayan Roy(19201043)

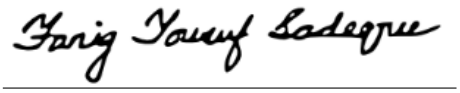
**Examining Committee:**

Supervisor:
(Member)



Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor-1:
(Member)



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

Co-Supervisor-2:
(Member)

Mr. Rafeed Rahman
Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
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

# Abstract

Accurate references in scholarly publications are a crucial aspect of scientific writing. The manual validation of references can be a time-consuming and error-prone process. This research introduces an updated version of the automated referencing validation model that makes the peer review process efficient. The proposed model utilizes the capabilities of Natural Language Processing generating sentence embeddings which uses an efficient algorithm. Our model first breaks down the scholarly article into sections and uses topic modeling to group every section according to their context properly. After that, It generates sentence embeddings for each section. By making sets of embeddings, they are used to calculate the semantic similarity between the query and the referred article. Additionally, this methodology addresses the valid references for non-contextual scenarios such as having common name entities. Lastly, strategic feature engineering is also being used for better performance. We have created a dataset of scholarly papers with manually verified references to evaluate the efficiency and accuracy of our model. This improved version of the referencing validation model aims to outperform traditional models such as Document-BERT, BERT, and SBERT regarding efficiency and accuracy. The model can be used in interactive real-time systems, providing quick and reliable feedback to peer reviewers. This study aims to make a contribution to the field of automated referencing validation in scholarly publications. The model offers a solution to the limitations of manual validation which makes it a valuable tool for peer reviewers and researchers.



**Keywords:** Automated Referencing Validation ; Natural Language Processing; Scholarly Publications; Semantically; Research; XLNet; Context Similarity; NER; Top Modeling; Tokenize.

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Over the past few decades, the modern world has witnessed a transition to online publication from traditional print publication with the remarkable rapid development of technology. In this era of digital publication. With this blessing, the scholarly publication also has found its way to expansion. Almost every research is built upon some existing research, so providing references in scholarly publications ensures the accuracy of the information cited and the reliability of the information that is being used. We can prevent plagiarism and academic misconduct with the help of reference validation. Reviewers need to ensure that proper credit was given to the original authors and sources. For this, proper reference is an essential part of any scholarly publication. It also helps the readers to follow the tracks of similar ideas, theories, and knowledge. The traditional way of validating the authenticity of the study and information used in scholarly papers is by checking the references and referred articles manually by meticulously reading every referred paper. With an exponentially increasing rate of research being conducted daily, this process is becoming more time-consuming, inefficient, and expensive for peer reviewers. In such a situation, the touch of automation can help to reduce the hefty process of cross-validation.

In our paper, we are using the power of Natural Language Processing (NLP) techniques in order to address the manual reference-checking process of scholarly articles by automating the entire process. By harnessing the abilities of Current Natural Language Processing techniques and optimizing them, we propose a highly efficient automated reference validation system that can accurately verify the authenticity, relevance, and accuracy of references by determining the semantic similarity of the reference with its referred article.

Our proposed method involves breaking down the referred document into sections, then grouping and tokenizing a section according to their context. For this, we have used topic modeling. Afterward, We generate embeddings. The next step is to downsize the embeddings and calculate the similarity between them using cosine similarity. If the similarity score meets the threshold requirements, we return the classified result. Otherwise, we repeat the process with the next section of the following document. This process is very efficient in terms of computation because it does not require pre-processing the whole paper together.

There are numerous probable perks of automating the cross-validation process of scholarly article references. One of them is the significant reduction of tedious reading and understanding of papers by the reviewers. Instead, they can better use their time and resources in evaluating other aspects of the papers such as the scientific diligence, methodology, and overall quality of the research paper. This will result in faster dissemination of newly found knowledge, allowing scientific advancements to flourish.

In short, the automation process in reference validation for scholarly papers is a new area of research. Automating this sector can greatly benefit the reviewers as well as scholars by allowing their research to be verified faster and speeding up the publication process overall. This research aims to provide a practical solution to automate reference validation with the proposed advanced and optimized NLP framework.

## 1.1 Research Statement

The proliferation of research publications has increased knowledge dissemination significantly. However, it brought a noteworthy challenge to the academic community. Reference validation involves reading the referred paper and getting a proper understanding of the topic, it usually takes a huge amount of time to validate the references of a scholarly article by a peer reviewer.

### 1.1.1 The Challenge of Escalating Publications

As the number of researchers and their publications continues to grow, so does the need for validating citations. Reference Validation of a large number of publications can be a tiresome job to do. Along with increasing the number of researchers and their publications, the number of references being used in an article is comparatively higher than ever because of the greater availability of information sources requiring the number of citations to be validated has increased exponentially. This creates a significant strain on the peer reviewers. However, the need for checking the authenticity of the references because of the probability of human error or intentional academic misconduct remains. With increasing interest in research among a large population, the quantity of research papers is going to increase at a rate when it will become quite impossible to check and validate every research paper reference manually in the near future. Since there is no currently established automated reference validation system for scholarly papers, our effort in creating one such system is to aid peer reviewers, making their lives easier and making the entire process faster, highlighting the importance and novelty of this research.

### 1.1.2 AI-Generated Reference

Modern Natural Language Processing (NLP) has revolutionized the field of language modeling. Today, machines not only understand languages properly, But they can generate contextually relevant responses quickly and flawlessly. For instance, ChatGPT using GPT-4, and Bard AI using LLM are great examples. Language models now can offer informative suggestions, and they have the ability to write large articles that are inseparable from human writing. Although these models have excellent

articulate capabilities, they still have some drawbacks when it comes to referencing. Whenever they struggle to find supportive information they tend to generate deceptive information and attach irrelevant references to it. This provides us with another crucial reason, highlighting the absolute necessity for reference validation, and automating this process with advanced NLP will make reference validation less tedious. Increasing use of the NLP models among researchers for their convenience and ease of finding and organizing information has raised many questions about the authenticity and accuracy of the recent research that is being conducted.

## 1.2 Research Objectives

Our research has made significant and noble contributions to the field of reference validation, addressing a fundamental and important task in the world of scholarly publications. Our work reflects our dedication to improving the quality and trustworthiness of academic communication. Here, we highlight the key aspects of our research contributions:

- **Automated Reference Validation System:** Our research has led to the development of a highly advanced automated system specifically designed for the crucial task of reference validation. We understand that the accuracy and reliability of references are pivotal for maintaining the integrity of academic discourse. Thus, our system serves as a valuable tool for researchers, academics, and publishers, offering them a means to ensure the correctness of citations within scholarly works. By doing so, we contribute to safeguarding against unintentional errors and fraudulent practices, ultimately bolstering the credibility of academic discussions.

- **Innovative Methodology:** Our research contributions are characterized by an innovative methodology that sets us apart. While traditional approaches often rely solely on semantic similarity, our methodology takes a more comprehensive approach by considering both semantic and contextual similarity. By incorporating advanced techniques like topic modeling and named entity recognition, we move beyond basic word matching to capture the intricate nuances essential for accurate reference validation. This innovative approach not only enhances the accuracy and efficiency of reference verification but also exemplifies our commitment to pushing the boundaries of what can be achieved in this field.

- **Dataset Creation:** One of the most notable and commendable aspects of our research is the creation of a specialized dataset tailored specifically for reference validation. Recognizing the glaring lack of comprehensive datasets dedicated to this particular task, we embarked on a rigorous journey of manual annotation. This painstaking process involved the meticulous examination and validation of numerous scholarly papers, resulting in the construction of a unique and invaluable dataset. Our carefully curated dataset, designed to be diverse and representative, not only supports our research but also serves as a fundamental resource for the wider academic community. Our dedicated effort in crafting this dataset is a clear demonstration of our steadfast commitment

to advancing the scholarly field and aiding future research in the validation of references.

In summary, our research contributions in the field of reference validation are motivated by a noble goal to raise the standards of scholarly communication. Our automated system, inventive approach, and the development of a specialized dataset collectively reflect our commitment to the pursuit of knowledge, academic rigor, and the integrity of scholarly discourse. Through these contributions, our goal is to enable researchers, foster transparency, and ultimately reinforce the foundation upon which scientific and academic advancements are constructed.

## 1.3 Thesis Organization

The thesis is organized into several key sections to provide a clear and structured presentation of the research. The introductory chapter (Chapter 1) begins with an overview (1.1) and identifies the research problem (1.2), highlighting the challenges posed by escalating publications and the role of AI-generated references. The research objectives (1.3) are then outlined, followed by an overview of the thesis organization (1.4). The literature review (Chapter 2) explores related works in the field. The methodology section (Chapter 3) is comprehensive, starting with a description of the dataset (3.1) and then detailing the various stages of feature extraction (3.2) including text handling, context-based chunking, topic modeling with BERTopic, chunk merging, semantic similarity calculation, and Named Entity Recognition (NER). The model specification (3.3) further delves into XLNet tokenization, model selection, architecture, loss function, and optimization algorithm. Results and discussions (Chapter 4) focus on performance metrics and the evaluation of model performance (4.1.1). Finally, the thesis concludes (Chapter 5) by summarizing the findings and their implications.

# Chapter 2

# Literature Review

## 2.1 Survey Methodology

While exploring the field of reference validation automation, we encountered a scarcity of papers on the topic in Google Scholar. Our search led us to relevant works on semantic and contextual similarity in different journals. Then we stumbled upon a relevant paper in the private repository of Brac University, accessible through "Dspace." To look into citation accuracy data, we also explored papers related to manual reference validation. Since the reference validation automation topic is new, our search revealed a limited pool of relevant papers. The lack of extensive research in this area led us to broaden our understanding through related studies on semantic, contextual similarity, and manual reference validation.

## 2.2 Background Studies

Recurrent Neural Networks are a type of neural network architecture that works well with sequential data by maintaining the memory of past inputs. The ordering of data points matters a lot in RNN. However, since RNN works on sequential data, it is computationally expensive to train large datasets and RNN struggles to capture long-term dependency as the architecture can hold the memory of the previous input only.[3] the paper introduces a new type of RNN architecture that is presented to fix the vanishing gradient problem and difficulty in modeling long-term dependencies. By storing important long-term data in a unit known as a "memory cell", LSTM networks address dependency issues. The main problem is that it processes inputs sequence by sequence, and it can limit their capability to take advantage of parallel computing. The amount of data that can be processed simultaneously and training times may be slowed as a result. In contrast, Transformers are highly parallelizable, since each token can be processed independently.

In [10] published in 2017 by Google Researchers, researchers address the limitations of RNN and LSTM by introducing Transformers. While making predictions, the Transformer employs an attention mechanism that enables it to selectively emphasize different parts of the input. The encoder part uses self-attention to compute the representation of each input token that considers its relationship with other tokens in the input. A word is represented using a vector of 512 dimensions using a word embedding method so that similar words have vectors with less distance. Then posi-

tional encoding is used to store positional information of the vector. Then multi-head attention and normalization are applied and output is forwarded to the feed-forward network then again normalization. For machine translation both encoder and decoder have almost similar steps except the decoder applies linear transformation and softmax function at the end. The author created two different models of this architecture, a base model and a big model. Both models have 6 encoders and 6 decoders stacked. But for the big model, the dimension of embedding is 1024 and has $213 \times 10^6$ parameters compared to $65 \times 10^6$ parameters of the base model. The model scores so well in English to German interpretation that even the base model beats every other previous model. In English to French translation, The big transformer model generates a cutting-edge BLEU score of 41.8 beating all the single models that have been published before, and the training cost is a lot less.
For the reference validation system, we need to use the attention mechanism so that when the system takes the referenced paper as input, it can emphasize on most relevant words and compute the results faster instead of sequential input taking which is computationally expensive.

The paper [12] describes a pre-existing model that performs at the cutting edge in numerous NLP tasks, including classification, name entity recognition, and question answering, etc. BERT uses self-attention to recognize the relationship between words of a sentence and is based on transformer architecture. Traditional old models like RNN and LSTM process inputs sequentially, but BERT uses the bidirectional nature of the transformer. It considers both the current and following words at the same time when encoding word representations, and it allows a computer to understand the context better. By fine-tuning both the right context and left context simultaneously across all layers, BERT is intended to pre-train extensive bidirectional characterization from an unlabeled corpus. It predicts the next masked word in a sentence and understands connections between sentences. It uses masked language model objective and next sentence prediction object to understand general language representations. BERT is fine-tuned for specific downstream tasks following pre-training. To accomplish this, task-specific layers must be added to the pre-trained BERT. Then, for the specific task, a smaller labeled dataset is used to fine-tune the entire model. One of the main benefits of BERT is it can transfer information gained from pre-preparing to different downstream tasks. $BERT_{base}$ has 12 transformer layers and $BERT_{large}$ has 24 transformer layers. The parameter size of $BERT_{base}$ is $110 \times 10^6$ and $BERT_{large}$ is $340 \times 10^6$. BERT performance on SQUAD v1.1&v2.0, EM score is 87.433, and F1 score is 93.16 beating previous state-of-art models. For our reference validation system, we would like to work with BERT architecture because of the attention mechanism. Since BERT captures language nuances by processing inputs in both directions, we can utilize the power of our system to improve accuracy and efficiency.

Since the release of BERT, there have been an increasing number of papers that have built on the technology following the BERT model. RoBERTa is one of the most popular ones. RoBERTa is a state-of-the-art model that builds upon the work done by its predecessor BERT (Bidirectional Encoder Representations from Transformers). The Facebook AI researchers who released the paper [13] in July 2019 insisted on the fact that BERT was significantly under trained and if improved upon, it can

match or exceed the performance of any other model that came before and after it during the time of the release of this paper. RoBERTa uses a similar model akin to BERT but makes some key optimization strategies to enhance the effectiveness of the system.

The Facebook AI researchers noted that to improve performance on downstream tasks, using more data on pre-training is necessary. So, they use similar datasets as BERT but also use CC-NEWS which covers 63 million news articles in the English language scraped between September 2016 and February 2019, and other datasets totaling over 160 GB uncompressed text compared to BERT's original 16 GB uncompressed data. Another type of improvement can also be seen as BERT relies on static, masking which is done once during data preprocessing. Even though BERT has protocols to minimize this issue, the end result being the same mask four times being reported in each training sequence. RoBERTa by using dynamic masking which means generating a new masking every time it feeds a sequence to the model solves this issue. One other thing that RoBERTa does is abandon Next Sentence Prediction(NSP) completely as removing NSP loss slightly improves or at least matches downstream task performance. NSP loss might have improved the original BERT's implementation performance, but the researchers concur this was a speculative error on the part of the original BERT implementation. RoBERTa's performance on SQuAD [8] v1.1 & v2.0, EM score is 88.9, and F1 score is 94.6 beating BERT and other state-of-art models. RoBERTa also achieves state-of-the-art results in RACE [9] and the highest average score to date in GLUE [11]. Since RoBERTa is a modified version of BERT and has better performance, it can be used as an alternative to BERT.

Now that we can observe the fact that BERT only gets better when it is fed more data in its pre-training process, we can also assume that one can also improve its performance in a specific field if we feed data to BERT regarding the specific work. This is fine-tuning as the pre-trained BERT model improves and can adapt its knowledge and parameters to the specific task. While fine-tuning BERT for text classification is not necessary for improving performance in tasks related to semantic similarity detection, learning how to fine-tune BERT for text classification will help one to understand how to use the advantages of fine-tuning BERT to one's advantage. The paper [14] shows us different ways to fine-tune text classification and also gives us a general solution for this issue. This proposed solution has obtained cutting-edge scores on eight widely researched text classification datasets.

The researchers suggest that since BERT is trained in the general domain then it is only a natural progression to further pre-train BERT in the task-specific domain as it will have a more relevant data distribution. In the paper, it is found that in-domain pre-training yields better results than within-task pre-training as the model detects more nuisances, and cross-domain pre-training does not improve performance as BERT is already trained on a general dataset. Furthermore, the original BERT-base model involves an embedding layer, a 12-layer encoder, and a pooling layer. So, for the task at hand, the most effective layer needs to be chosen. The layers can be fine-tuned with different learning rates. In the paper, it is found that the top layer of BERT is more useful for text classification. Another thing the

paper emphasizes is multitasked fine-tuning as it is an effective approach to learning knowledge from several other supervised tasks. All the tasks share the same embedding and BERT layers except the final layer which is the private classification layer. All of these experiments were done by checking error rates on eight different text classification datasets.

On the topic of fine-tuning BERT, to improve BERT-based sentiment analysis, researchers have put together different strategies to analyze BERT-based models for sentiment analysis. The paper [17] presents three strategies where the first is fine-tuning the BERT-based pre-trained models. The second and third strategies are to create an ensemble model that develops different types of BERT variants and uses a compressed model named Distil BERT respectively. The researchers have worked with three types of BERT for sentiment analysis in the field of software engineering which are the original BERT, RoBERTa, and ALBERT. They have fine-tuned all the models in different datasets for a comparative study of the intended strategies. To ensemble all these models together, a weighted voting scheme was used, and a confidence score was considered for each model to get the final weighted prediction. Apart from this, a distilled model would be crucial if one wanted to increase the performance of the models. The distillation framework would contain two models where one is large, and another is small. The smaller model would mimic the larger model by learning from the probabilities generated from it until the final activation function. While retaining 97% of the language understanding capabilities, The Distill BERT model is also able to reduce its size to 40% of the BERT Base model. The extensive trials conducted by the researchers would demonstrate superior performance compared to existing tools where it achieves higher f1-scores of 0.88, 0.84, and overall 0.92 for the Stack Overflow, Jira, and GitHub datasets respectively. This research can be further extended to domain-specific research and still has scope to improve.

Now, following our topic, we need to learn different ways of calculating text and semantic similarity. Since it is a crucial part of our research, an overview is needed. In [21], researchers present an architecture based on LSTM to learn a text similarity calculation on different length input character sequences. The given model is built with layers of character-level bidirectional LSTM added with a Siamese network. It gets trained to display different-sized strings into embedding space with fixed dimensions by using knowledge about the closeness between a set of two strings. Semantic matching between texts is a fundamental problem in the field of Information Extraction (Martin and Jurafsky,2000). Normalization is very important for this type of task so that texts can be compared to other texts. A simple example is "12 pm", "12.00h", and "noon" all map to the same representation. Normalization is crucial for retrieving useful data from the unstructured corpus. The paper provided a system for job title normalization which is a common task for information extraction from thousands of CVs and social media analysis.

The job is to receive an input text and map it to finite sets of job codes that are predefined. For example, "react developer" can get mapped to "frontend developer". It is a multi-class classification problem, but the paper limited the class to synonymous jobs only. This approach is more flexible and helps to find closely related jobs.

Siamese networks are two-branch networks with fixed weights. The two branches get copied and merged with an energy function. In this network $x_1$, and $x_2$ are input sequences and y is the output ($y = 1$ when similar and $y = 0$ when dissimilar). The energy function $E = cos(fw(x_1), fw(x_2))$. The network has four layers of LSTM nodes which are bidirectional. The final BLSTM layer produces a fixed dimensional output which is forwarded to the feedforward layer. The input strings are padded to make the input of 100 characters. Adam's formula is being used to enhance the parameters of the model (Kingma and Ba,2014). The experiment showed embedding spaces captured invariance of input. The model is invariant to spelling differences, superfluous words, and synonym replacements. We would like to experiment with the behavior of this network on standard textual similarity for our reference validation system. Our reference validation system can take advantage of mapping the important words to map different classes so that the system can keep track of the relevant words in both papers. Also, the baseline here is pretty weak, so comparison to stronger baselines would help to increase the accuracy and efficiency of the proposed model and our reference validation system.

One of the more important topics that we need to focus on is the topic of using cosine similarity to get accurate results. In document-level sentiment classification, each document is represented as a fixed-length vector, and document embedding models map each document to a compact and meaningful vector in a continuous vector space with lower dimensions. In[15], researchers hypothesized that training document embeddings using cosine similarity instead of dot product will lead to better results in the field of sentiment classification. The reasoning behind this is twofold. Firstly, cosine similarity is widely employed to find document similarity and enhances the representation of similar documents by having a lower angle between corresponding vectors which indicates greater similarity. Moreover, it acts as a regularization technique by disregarding vector magnitudes. Since cosine similarity focuses on the angle between the vectors unlike dot product which considers the magnitudes of input and output vectors, it mitigates the incentive for magnitudes to dominate the computation of similarity. The researchers trained a neural network to accurately predict the words and N-grams in a given document which in the process helped them learn useful document embeddings. They compared the models which were benchmarked in the IMDB dataset which contains 25000 training documents, 25000 test documents, and 50,000 unlabeled documents, and found out that using cosine similarity as the similarity measure instead of dot product increases accuracy across the board. Even though these experiments resulted in success, it is essential to benchmark these experiments on more datasets to combat the possibility of overfitting.

As cosine similarity is used extensively now for its enhanced representation of documents, it will be a crucial part of our research. Cosine similarity can also help in many estimation methods. However, in[7], the authors stated that the Siamese network is inefficient because of the syntax complexity and word correlation. The paper proposes a similarity estimation method that mixes headword and semantic attention mechanisms.

To begin, contextual data is extracted using a BiGRU model. Then by using the

attention mechanism and character splicing, two sentences are collected which are semantically enhanced. The author used the convolutional neural network to merge embedding data and contextual data together. The author used nouns or verbs as headwords because they represent the main data of the sentence. Then, the result of the BiGRU is calculated and updated after receiving a set of headwords. In Information Fusion, part word sequences are fused after splicing. Then the cosine similarity function is used as an evaluation formula to calculate text similarity. The author syntactically analyzed the sentence with the Language Technology Platform in order to obtain the headwords. It is always possible that significant information will be erased because the convolution kernel has a fixed-sized window. Even though multiple windows of varying sizes can be used to solve this issue, doing so will require more calculations. The author combined the data from the spliced words using a CNN network (DCNN). Finally, the author used the cosine distance formula to calculate whether the two texts are similar in semantics after obtaining the vector representation of the sentences. The author compared HA-RCNN to cutting-edge models used in the same application to demonstrate its efficacy. The model achieved a 0.93 value in recall and 0.81 in F1 value. By looking at the comparison, it is obvious that the change has little effect on the model's performance; however, the model is trained faster. For our reference validation system, we can use the head attention mechanism to experiment if our model gives better output in terms of
In contrast, the paper [19] takes a different approach to semantic matching and also highlights the need for syntactic matching. The author proposes an improved NLP model to tackle some common issues of syntactic and semantic matching such as the highly complex and ambiguous nature of many prosodic phrasings and the scarcity of training datasets. The model uses bidirectional long short-term memory (BiLSTM) deep network layers and an attention mechanism to achieve more precise and contextually-aware representations of text sequences.

The BiLSTM layers in the encoder stage of the model process input text sequences in both forward and backward directions. By examining the words that come before and after a sentence, the bidirectional processing method makes sure that the model can capture contextual information and any word dependencies that are present in the sentence. This opens up a realm of possibilities for the model as they can get a profound understanding of semantic and syntactic characteristics of the words. This allows the model to encode these characteristics into meaningful representations.

To enhance the model's performance, the attention mechanism must be used because it plays a critical role. Context vectors are generated using the attention mechanism. This vector consists of computed alignment scores at different word positions. Priority is given to the words that carry substantial contextual information by the scores. By focusing on a subset that are relevant words, syntactic and semantic matching is improved in the model. This results in a more efficient performance of accurate predictions.

To test this hypothesis and asses its effectiveness, the researchers conducted experiments on small datasets that represent real-life applications. These datasets encompass a variety of syntactic and semantic matching tasks, enabling comprehensive testing and comparison with an alternative algorithm that lacks an attention mech-

anism. The results obtained demonstrate the superiority of the proposed model, surpassing the performance of models that do not incorporate attention mechanisms. The proposed model excels in capturing intricate word relationships and achieves higher accuracy in syntactic and semantic matching.

In short, the paper presents a promising NLP model which combines BiLSTM deep network layers and an attention mechanism in order to tackle some important challenges related to syntactic and semantic matching. The model's architecture enables the encoding of text sequences into contextually-sensitive representations, while the attention mechanism enhances its ability to focus on relevant words. However, it does not explore how the model can be applied to tasks involving context matching between two different sentences or groups of sentences. This limits the practical applicability of the model in applications such as reference validation, where understanding the relationship between multiple sentences is crucial.

Unlike other papers, paper [1] determines semantic text similarity with both word similarity and string similarity. According to the author, text similarity measurement is a crucial task in Natural Language Processing(NLP) and related fields, with applications ranging from information retrieval to data integration but most existing methods suffer from the drawback of being domain-dependent. To address this limitation, researchers have proposed a fully automatic method that can be used independently of the domain. The proposed method combines a corpus-based measure of semantic word similarity with a modified version of the Longest Common Subsequence (LCS) string-matching algorithm.

The proposed method differentiates itself from the rest by including many components to find the different aspects of similarity. It offers an innovative idea of combining a corpus-based measure for semantic word similarity and a modified version of the LCS for a string-matching algorithm. So, this fusion lets the model go beyond the limitations of traditional NLP models, whose current approaches mainly focus on large documents or individual words. This method gives us a more effective way to measure the similarity between short texts because the method considers both word-level and string-level similarity.

Moreover, this method holds another important trick up its sleeve, as it can address common-word order similarity problems in short texts. This NLP model can address common-order similarity as this model can find normalized differences of common order. As we know that different syntactic orders can hold the same value, so the model considers the syntactic information to capture subtle short text variations.

Since it was established that syntactic information helps in common-word similarity problems, the method's ability to accurately assess text similarity is increased. It is acknowledged that the order in which the word comes into play in a sentence can definitely influence the overall message of the said sentence. This can be seen even more in short texts where the effect of syntactic orders can be observed more frequently.

So, we can say that the model offers a more effective and nuanced approach to text

similarity measurement. By combining both syntactic and semantic situations in NLP, it is possible to provide a more accurate assessment of the similarity in various text-related applications.

In conclusion, the proposed model's creative combination of a corpus-based measure of semantic word similarity and a modified version of the LCS string-matching algorithm offers a promising solution for measuring text similarity in various applications with short texts involving textual knowledge representation and knowledge discovery. However, this model is not suitable for reference validations because it is not optimized to handle very large texts. Reference validations require processing and understanding the entire referred article, which is too much text for this model to handle. Therefore, it is not appropriate for tasks that involve handling extremely large texts like complete articles.

In[20], the authors presented problem-independent greedy text similarity, which is computationally efficient on big datasets with any word embedding method. Based on an evaluation rule that calculates word similarity and significance to the texts. GTSM plots words in two texts. In the k-nearest neighbors text classification problem, the authors compared it to the well-known word mover's distance algorithm and discovered that it produces comparable or superior results. It was shown that GTSM is a good substitute for both sentence-level and word-level compared to WMD and sentence mover's similarity due to correlation score. The similarity measurement is used in NLP to determine the degree to which words convey the same meaning in a given context and to capture semantic relations between them. The terms are semantically close if the interval of vector embeddings is lesser than a fixed threshold. Different word representations are used for GSTM, for example, Co-Occurrence, Weight of Words, Centroid, Vector embeddings, TF-IDF Weights, and LDA. The primary concept of GTSM is to evaluate and normalize the plotting of words between two files by removing important word relations and considering the importance of relevant words. It has two parts: truncated plotting and normalization. The decreased plotting (nominator) checks all potential relevance between word pairs and is trimmed by presenting fixed boundaries. Authors developed a standard form of the "ideal" relation between two text documents. The obtained similarity score is significant and $> 1$ if the estimated truncated mapping is more significant than the standard mapping. GTSM use this algorithm to calculate similarity metric:

1. Vector embeddings of words,
2. Estimate a matrix of pairwise similarity,
3. Evaluate words weight using one of the techniques,
4. Perform a truncated mapping,
5. Fix a threshold by analyzing which word-pairs have a significant connection.
6. Evaluate whether words from two documents have an important connection or not by the mentioned function.

The time complexity of GTSM is o($n^2$) because pairwise similarities have to be calculated. This algorithm's appeal lies in its ease of parallelization, adaptability to specific data, relatively low computational costs (average complexity time is O($n^2$) in relation to the number of unique words in documents), and acceptable F1 score.

In[16], When it comes to pretraining large language models such as BERT, It is generally seen that denoising autoencoding pretraining approach works better than autoregressive language modeling pretraining approaches. However, BERT relies on masking the input in the former pretraining approach. By doing this, It neglects the dependency between the positions that are masked. So, XLNet was proposed by the researchers at Google. Firstly, XLNet lets bidirectional learning be possible by maximizing the log-likelihood over all possible permutations of the factorization order. Also, by being an autoregressive formulation, it overcomes the lacking of BERT. XLNet is able to capture more contextual information compared to BERT. Moreover, XLNet introduces the transformer XL into the pretraining process, which is a state-of-the-art autoregressive model. In short, XLNet is a model that has combined the advantages of the autoregressive language modeling approach as well as the advantages of the autoencoding language modeling approach. The neural architecture that is designed for this model is synergistic with the autoregressive objective. If we compare the performance of this model with the ones that came before, it becomes clear that this model is the one that achieves substantial improvement over the models that came before.

## 2.3 Related Works

Referencing is crucial in scholarly papers as it acknowledges the sources used and provides credibility to one's work. By validating references, it is possible to ensure the accuracy and reliability of the information presented as well as maintain the integrity of the research. However, reference validation at the contextual level is still a new area of research.

In[5], It is stated that the author is mainly responsible for the accuracy of referencing others' work. However, since relying on the author for accuracy is not enough; throughout the publishing process, authors, peer reviewers, editors, and publishers may collaborate to avoid the majority of reference errors. Reference error problems have been happening for decades and it is still increasing. Referencing a paper helps to skip redundant work and improve the previous work or studies. Sharing of ideas and information may be facilitated by acknowledging all relevant sources. Therefore, it is essential to reference peer-reviewed papers that have been checked by multiple editors and reviewers. Finding the correct research paper to cite takes a lot of skills. An author should be experienced and skilled enough to find proper research papers to cite. The author should understand the motive and research goals of the referenced paper before citing it. If an author cites a paper inaccurately, it will often negatively impact scholarly journals and their indexability. If the citation is accurate, it will help readers to find the source of the referenced paper. Wrong and irrelevant citations will represent sources in unwanted manners. In the digital age, reference linking allows one to insert clickable source links so that users can

easily navigate to the main source. Besides authors, editors have a big role in selecting, formatting and verifying resources. Editors should have a deep knowledge about the impact of references and the consequences of wrong citations. Peer reviewers recheck and validate every citation, and they inform us whether we should add other citations or remove any redundant citations. Editorial software is being used by more and more publishers to digitize the reviewers' reference validation. Such a methodology might build the productivity of distributing dependable and profoundly instructive reference lists.

In another research paper [2], the writers discussed the importance of obtaining publication records and citation indices accurately, to evaluate academic performance. Obtaining complete information about one's publication and citation records can be a challenging and time-consuming task to do manually. With that thought, they proposed an automated and reliable method for estimating the records of an individual's publication citation and his relevant citation indices. This automated method estimates the publication-citation information of any particular individual by taking a set of domains such as an individual's name, publication titles, individual's year range, and the URL of that writer's publication list. Using these filters on top of various search services such as Google Scholar and Web of Science, the researchers were able to significantly improve the estimation retrieval of one's publication records.

To measure the performance of this method, 30 faculty members were selected at random from all wide individuals with scientific backgrounds and academic institutions. The method was able to show an average sensitivity and specificity rate of 97.5 % and 72.1 % respectively. One major drawback of this method is it determines publication ownership within the domains mentioned above, where it should verify this by using context similarity.

A study [4] from 2011 investigated whether online reference managers such as CiteULike and Mendeley can be used as measurement tools for academic influence. Online reference managers are platforms that let their users download, read, and exchange reference information from within reference libraries online. Web of Science is the bibliographic database that was used as a benchmark to compare the number of users in CiteUlike and Mendeley and make an educated assumption about whether WoS citations in a paper have any scholarly influence by assessing users who have saved the paper in one or more online reference managers. This type of research has relevance to the paper at hand as this shows us the alleged influence of bibliographical databases when assessing academic influence. This is something this paper could take into account while checking reference validation.

By using a sample consisting of 1613 papers published in Nature and Science in the year 2007, meaningful correlation analysis was attempted. A correlation was found between WoS citations and CiteUlike and Mendeley users but it would seem that the research would have major shortcomings. First of all, assuming a paper has value to a user just because it was in a saved reference manager was not proven. Moreover, the study was only conducted on Nature and Science and it might yield different results in other fields. Furthermore, it can be said that all research papers saved do not hold equal value to the user. So, the correlation might indicate the

research paper was successful in measuring academic influence, but further research needs to be done to establish this research paper as completely valid. Also, the researchers suggested that the number of users is relatively small and so this cannot pose a serious threat to traditional citation indexes. This ends the interest of this paper compared to the paper at hand.

The author of the paper [6] conducted great research highlighting the need for a better reference validation system as he discusses the validation of reference lists of already published scholarly articles in English-language biomedical journals where he has found an 18% error rate. The study analyzed the references used in ten biomedical journals that were published in 2005, 2008, and 2011. For this study, 15 references from each article were arbitrarily selected, making a total of 450 references. After analysis, 81 out of 450 references turned out to be inaccurate. These inaccurate references are classified into two categories: major and minor errors. Considering the level of inaccuracy, the author classified 48 references as minor errors, and the rest of the 33 errors as major errors. So with the conducted study, we can easily conclude that even after manually reviewing, it is hard to detect incorrect validation as it is really hard to read fully refereed papers with a great understanding in such a short period.

While most of the above research focused on reference validation without considering contextual level or took a manual approach, Plabon Dutta and his co-researchers [18] in a paper named "Document-BERT: A Transformer-based Automated Referencing Validation Model for Scholarly Publications". D-BERT (Document-BERT) is a proposed model that aims to generate efficient sentence embeddings for large documents in order to find validity or scholarly articles using semantic similarity measurement value. The model builds upon $BERT_{BASE}$, a widely used transformer-based language model. The authors introduce D-BERT as a means to overcome the computational challenges associated with encoding large documents using BERT.

The literature review starts by highlighting the importance of context-aware models for generating word embeddings. While traditional methods like Word2Vec provide fixed vector representations for words, they lack consideration for contextual information. In contrast, models like BERT create dynamic word representations influenced by surrounding words, resulting in more accurate and contextually rich embeddings.

However, when working with large documents, relying solely on word embeddings becomes tedious and limits the information extracted. This is where sentence embedding techniques come into play. Sentence embeddings translate sentences into real-number vectors, allowing for similarity comparisons in vector space. Several existing models, such as SBERT, XLNet, and Universal Sentence Encoder have been proposed for generating sentence embeddings. In this study, the focus is on building a BERT-based model for generating sentence embeddings.

The proposed architecture of D-BERT can be applied to any BERT-based model, such as BERT, RoBERTa, DistilBERT, or XLNet. However, for demonstration purposes, an uncased version of $BERT_{BASE}$ with 110 million parameters is considered.

The authors emphasize the need to optimize time, space, and computing power by tokenizing each input sequence with a maximum length of 300 tokens. Texts longer than 300 tokens are truncated to ensure efficiency.

D-BERT tackles the challenge of converting large tensors into fixed-size sentence embeddings. For $BERT_{BASE}$ models, each encoder layer outputs a dense vector of dimension 768. By applying a mean pooling layer, the output embeddings for all tokens are compressed into a 768-dimensional vector, representing a sentence embedding. Non-real tokens are ignored during the pooling operation using attention masks. A fully connected dense layer with a tanh activation function further down-projects the 768-dimensional vector to 512 dimensions. This reduction in dimensionality significantly improves runtime without sacrificing much accuracy.

The model, dBERT, produces a tensor containing the sentence embeddings for each set of Sentence Array for Embedding (SAFEs) within a document. Initially, 5 sets of SAFEs are being generated where 1 set contains only one sentence and the 4 set contains a combination of 4 sentences in a way so that if there is only a context spread through any 4 sentences, one of these sets of SAFEs will contain them. So by using these embeddings, D-BERT can capture the contextual information and temporal properties of the input text in an efficient manner.

To measure the similarity between SAFEs (Sets from the scholarly reference and Sets from the referred article), cosine similarity is employed. Cosine similarity calculates the cosine of the angle between two vectors and provides a value between 1 (for overlapping vectors) and -1 (for opposite vectors). By comparing the cosine similarity scores, the model identifies the most similar sentences to a given query.

Overall, D-BERT presents an efficient approach for generating sentence embeddings and calculating similarity in large documents. By leveraging $BERT_{BASE}$ and incorporating pooling layers, the model significantly reduces computation time while preserving the accuracy of sentence representations. The proposed architecture provides a practical solution for information retrieval and semantic matching tasks within large-scale document analysis. One of the key disadvantages of this approach is it can not identify the context properly if the context is spread over 5 or more sentences. So, in such cases, this may give an incorrect classification result. Moreover, the authors have only claimed a small performance gain only if the document is too large over SBERT while it falls behind when the document's size is on the smaller side. However, while testing and implementing the author did not share any classification data table questioning how this will actually perform on real datasets.

# Chapter 3

# Methodology

Our proposed Automated Reference Validity Scheme seamlessly integrates accessible data for dataset construction, employing strategic feature extraction to implement relevant features optimally in various Machine Learning Models, RRN, and Transformer Models. It comprises three distinct phases; (1) Dataset Collection, (2) Feature Extraction, and (3) Classification. Fig. 3.1 represents the top-level architecture of our Automated Reference Validity and the following text here explains the purpose of its different phases.

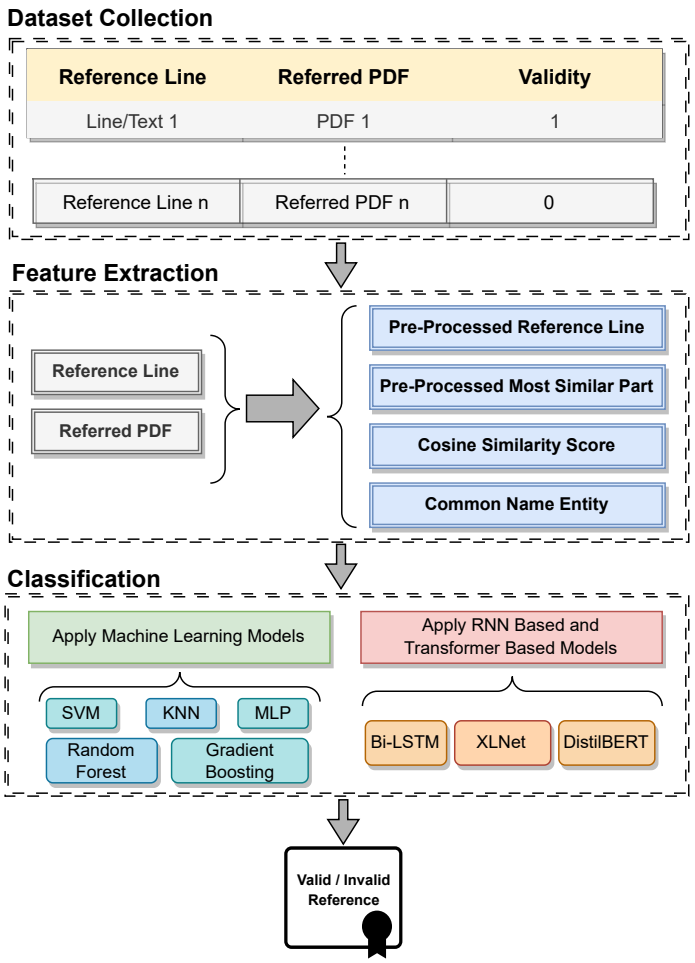



Figure 3.1: Top-level layout of proposed Automated Reference Validity technique.

## 3.1 Dataset Collection

We collected our data from IEEE Xplore which is the largest journal for scientific research. We looked for famous research papers that have been cited multiple times. By selecting these famous research papers, we have created a dataset much more swiftly than expected. We have looked at the papers that are citing our selected papers and checked them individually for the citation line that is referencing the original paper. The citation line along with the links to the selected papers are stored in the dataset for our convenience. Furthermore, we checked that the citation string matches contextually to any segment of the main reference paper, and we set the reference validity value (target variable) according to the match results. Moreover, we have made sure to cross-validate our dataset to ensure quality and remove any margin of error from the dataset. All individuals in the team undertook the cross-validation process to verify the dataset collection process has no room for mistakes.

To solidify the authenticity of our dataset, we have also collected data from Pubpeer which is a website for discussing and reviewing research after publication. It is a peer-review website that highlights the shortcomings of scientific research. We have added the invalid reference issues that were found in Pubpeer and the reasoning behind them in the dataset. These reasonings are the expert opinions of peer reviewers that have helped us to enrich the dataset.

After collecting the data, we found that our dataset had an imbalanced ratio of reference validity value (target variable). Data where the reference validity value was false or "0" were underrepresented in the dataset. To combat this, we have made synthetic instances and added them to our dataset. Two types of synthetic data were made. The first type was made using ChatGPT. Firstly, We made a topical reference line and a random paper using AI but since we cannot check a made-up paper, we attached a random paper to it. This would increment a "0" reference validity value in the dataset. The second type was made by intentionally writing wrong reference lines and attaching a paper to the line. All these new instances helped the dataset by tackling the imbalance problem.

| userstring | Main_Reference Path | Reference Validity |
|---|---|---|
| For example, the article [193], and [194] used bridged non-parallel modalities using neural networks. | MULTIMODAL REPRESENTATION LEARNING G: ADVANCES, TRENDS AND CHALLENGES.pdf | 2 |
| Multimodal emotion detection methods can be classified into early fusion methods and late fusion methods [43]. | MULTIMODAL REPRESENTATION LEARNING G: ADVANCES, TRENDS AND CHALLENGES.pdf | 1 |
| Most professions such as teacher, operator, doctor, and driver require a high degree of attentiveness to complete the work efficiently and effectively [1], [2]. | Using Support Vector Machines to Classify Student Attentiveness for the Development of Personalized Learning Systems.pdf | 0 |

Figure 3.2: Primary Dataset Sample

### 3.1.1 Dataset Description

The dataset contains 1786 instances of referred paper and reference string with a target variable that determines validity. This section provides an overview of the dataset used in our research, highlighting the characteristics, future improvements, etc. Our primary focus of the research was the IEEE paper reference validation. If

the citation string matches contextually with any segment of the main paper, we have set the reference validity value to 1 which means the reference is valid. We have also set the reference validity value to 1 if there is no contextual match, but the citation string contains the named entity of the main paper, as we plan on doing named entity recognition. But, When the citation string is too ambiguous, we have set the reference validity to 2. If all the conditions fail, we have set the reference validity to 0 which means the reference is invalid. Our dataset contains 2 primary features such as reference string and referred paper and the target variable is reference validity.

Our dataset has a total number of 1786 instances for now. Out of all rows, 1237 instances have a reference validity value of 1, 485 instances have a reference validity value of 0 and the rest of 64 instances have a reference validity value of 2. Out of 485 instances of a reference validity value of 0, 250 instances are synthetic, and 235 instances are collected from IEEE Xplore and Pubpeer. Increasing our dataset could potentially lead to further improvements in Automated Reference Validation.

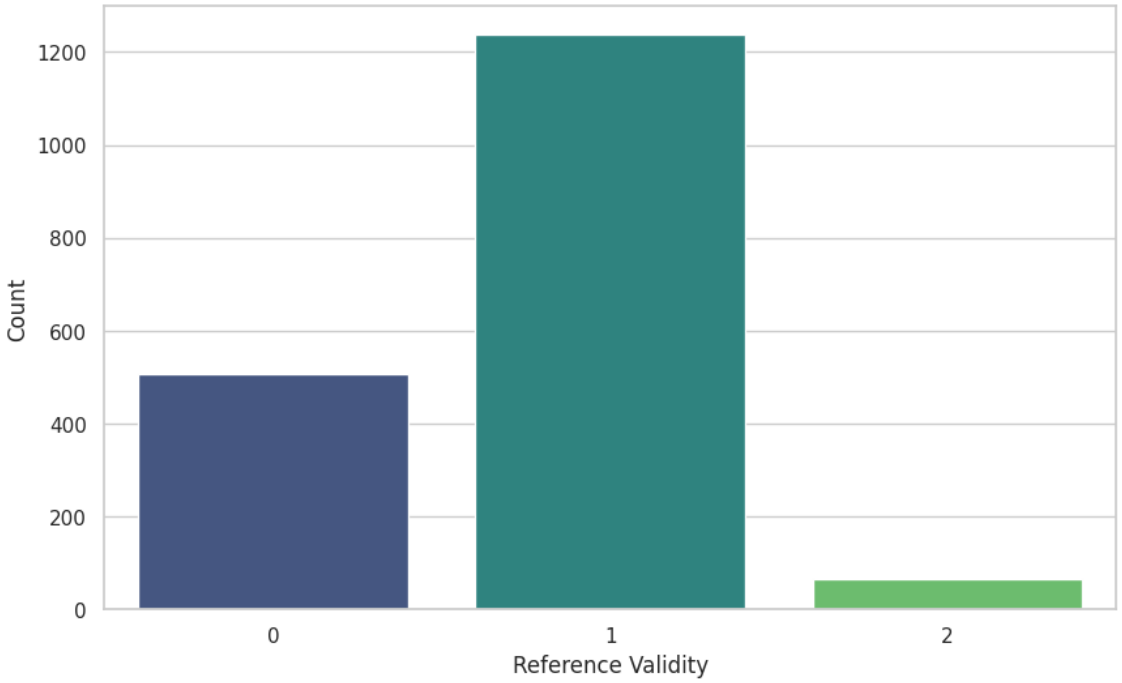



Figure 3.3: Count of Annotated Labels

We anticipate that increasing our dataset could potentially lead to further improvements in Automated Reference Validation.

## 3.2 Feature Extraction

To develop a robust automated system, we proposed a multi-phase feature extraction strategy to extract features for evaluating reference lines with referred PDFs. This strategy includes text preprocessing, named entity recognition (NER), context-driven segmentation, and the application of topic modeling, etc. After extracting

the features, a new dataset is created to train a model with four new features. Subsequently, the features are: Pre-Processed Reference Line, Pre-Processed Most Similar Part from the referred PDF, Cosine Similarity between the pre-processed reference line and pre-processed most similar part from the referred PDF, and Common between the pre-processed reference line and pre-processed most similar part from the referred PDF. To get these following features we followed a systematic approach which is described below:

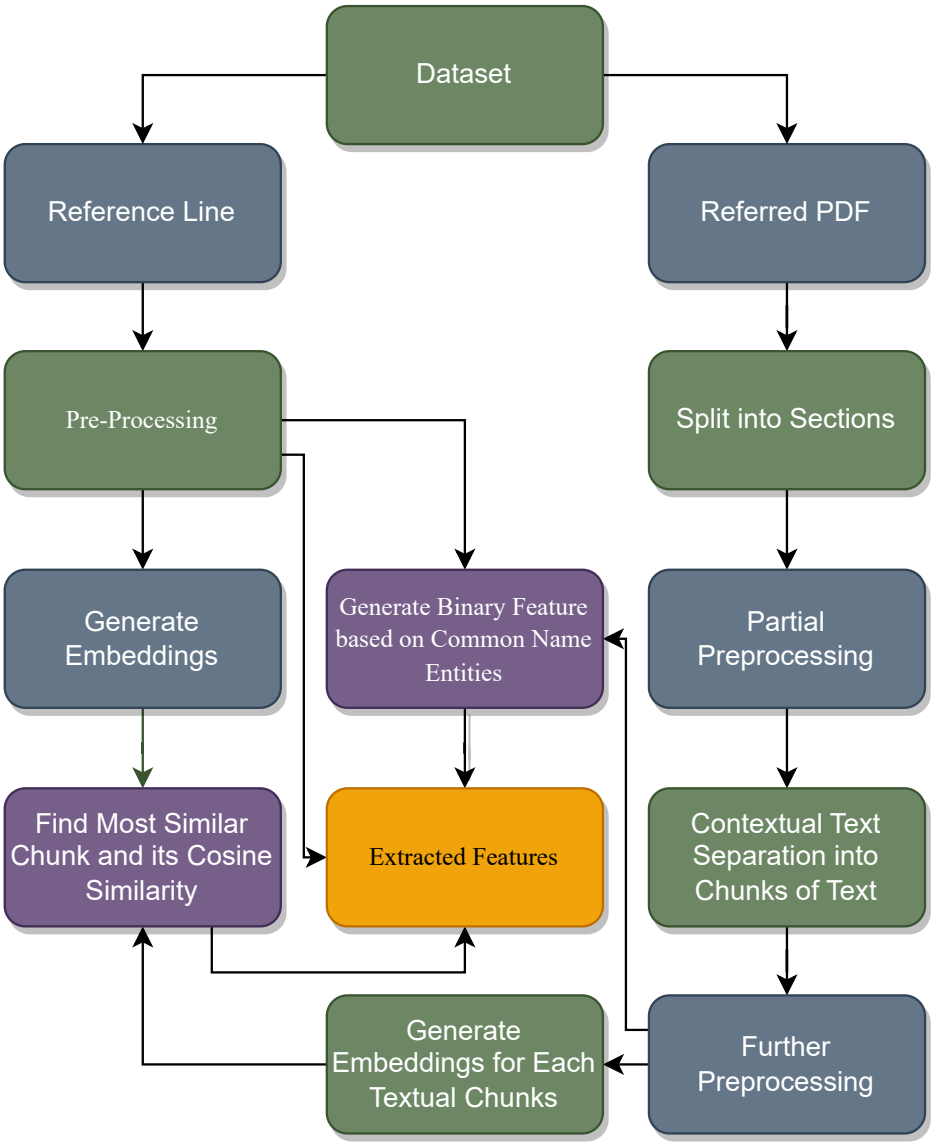



Figure 3.4: Feature Extraction Process.

### 3.2.1 Extraction of Text from the referred PDF and Initial Data Handling

Scholarly articles are in general found in PDF format. So we started our work by extracting the texts from the referred PDF file, followed by its careful partitioning

into discrete segments. This division of the text into sections is pivotal because it acknowledges that the concluding statement of one section and the opening statement of the next are unlikely to be thematically connected. This segmentation allows us to create manageable units for analysis, with each section representing a distinct part of the scholarly paper, such as the introduction, methodology, results, discussion, and conclusion. After segmentation, we embark on text preprocessing, an essential step in standardizing the text data by lower-casing and eliminating extraneous noise. Our text preprocessing comprises several vital steps:

- **Lowercasing:** To ensure uniformity in treatment, we convert all text to lowercase. This step prevents distinctions based solely on capitalization, ensuring that "word" and "Word" are treated equivalently.
- **Punctuation Removal:** Punctuation marks are systematically removed. These symbols, while important in written language, often do not substantially contribute to textual meaning and can be safely omitted.
- **In-text Citation Number Removal:** Scholarly papers frequently contain in-text citations, which we remove to focus exclusively on the core content, simplifying subsequent analysis.

During this preprocessing, some common data-cleaning strategies like stopword removal and lemmatization were avoided as we used BERTopic in the topic modeling step for the context-based separation, and BERTopic can utilize the presence of stopwords and non–non-lemmatized words in sentences for better topic modeling.

### 3.2.2 Context-Based Textual Separation

At this point, we introduce a novel step in the process: context-based textual separation. Recognizing that topic changes can occur within sections, we aim to divide the preprocessed text into multiple chunks based on words or phrases that signal shifts in the discussion, such as "but," "however," "on the other hand," "next," and others. This division, while informative, introduces a challenge—how can we be certain that these separated text segments maintain consistent contextual themes? To address this challenge, we employ topic modeling using BERTopic for each section's multiple chunks of text which we acquired in the above-mentioned ways. As we mentioned earlier during the preprocessing, we intentionally avoid advanced preprocessing techniques like lemmatization and stopword removal at this stage. This decision is driven by the need to retain stopwords, as they play a critical role in segmenting the text into multiple chunks. Furthermore, BERTopic is optimized to work with non-lemmatized data and data that includes stopwords as it was trained on similar data patterns. Topic modeling with BERTopic is a pivotal component of our feature extraction process. BERTopic helps us uncover the principal themes or topics present within each chunk of text. Underneath the BERTopic, it uses the BERT language model to extract context-rich information by capturing the nuanced context of texts. Upon applying BERTopic, each chunk of text from all the sections gets labeled for specific topics. These labels indicate the primary topic/theme of each of the text chunks.

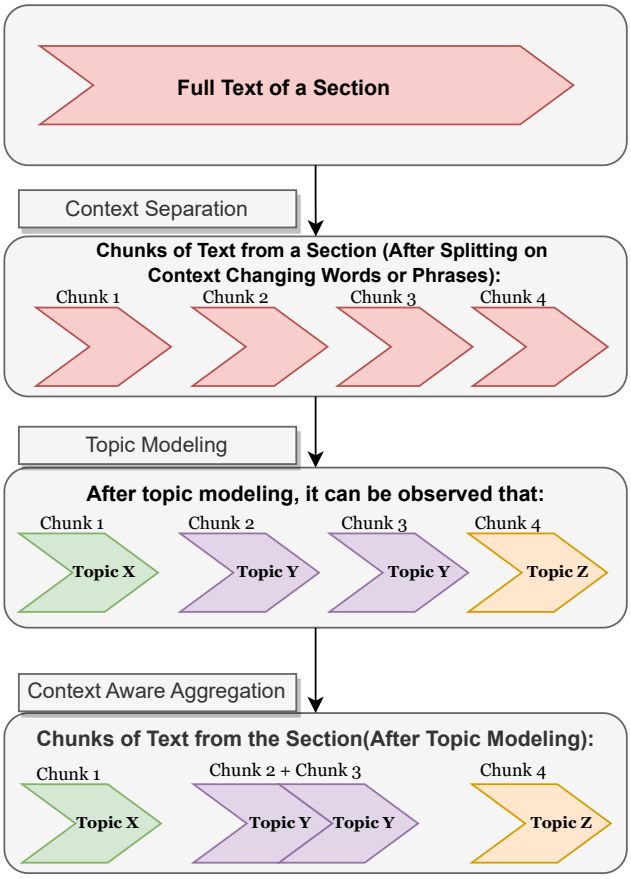



Figure 3.5: Context Based Separation

### 3.2.3 Semantic Similarity Calculation

Our next step is to find out which of these coherent context-based chunks share the most similarity with our reference line and how much semantic similarity is there between them. However, since we are now going to calculate the semantic similarity, we are going to do some preprocessing again as a part of data cleaning by removing stop words and applying lemmatization. Calculating semantic similarity is one of the most crucial steps of our work as this allows us to quantify the extent of alignment of the reference line to the content of a specific chunk. To compute semantic similarity, we rely on embeddings generated by the MiniLM-L6-v2 model. These embeddings are generated separately for both the reference line and each chunk. Embeddings are vector representations that encapsulate the semantic essence of the text, facilitating the assessment of their similarities.

$$Cosine(x, y) = \frac{x \cdot y}{|x||y|} \tag{3.1}$$

We utilize the cosine similarity metric to measure the degree of similarity between the reference line embedding and the embedding of each chunk. Cosine similarity computes the cosine of the angle between two vectors, yielding a score between -1 and 1. However, we have used the normalized range which is from 0 to 1 for the purpose of our use. A higher cosine similarity score indicates a greater degree of semantic alignment. The outcome of this phase provides three pivotal features for each data point in our dataset:

1. The preprocessed reference line.

2. The most contextually similar chunk of text from the referenced scholarly paper.

3. The cosine similarity score quantifies the semantic alignment between the reference line and the selected chunk.

### 3.2.4 Named Entity Recognition (NER)

It's essential to acknowledge that references may not always align with the surrounding context in a purely semantic manner. References can be triggered by specific named entities mentioned in the text. For instance, a reference might be prompted by the presence of a prominent entity such as the "BERT model," leading any author to cite the original BERT paper.
To capture this phenomenon, we incorporate Named Entity Recognition (NER) into our feature extraction process. We employ Spacy's en_core_web_lg model to perform NER on both the preprocessed reference line and the entire scholarly paper. Once these name entities are identified, we proceed to compare the named entities detected in the reference line with those within the scholarly paper. If we find common named entities between the reference line and the paper, we generate a binary feature called "AutomatedNER" This feature is assigned a value of 1 when common named entities are identified and 0 when there are no shared named entities. Automated NER adds an additional dimension to our reference validation process, capturing scenarios where named entities play a crucial role in confirming the validity of the reference.

| preprocessedUserstring | mostSimilarPart | automatedNer | cosineSimilarity | Reference Validity |
|---|---|---|---|---|
| research paper analyzed sentiment business review classification used dataset million data provided yelp used two feature extraction method four different machine learning model linear support vector classification logistic regression naive bayes stochastic gradient descent classifier splitting dataset go 80 training 20 testing used natural language toolkit nltk preprocessing data | research area sentiment analysis opinion mining sentiment mining sentiment extraction ha gained popularity last year online review becoming important criterion measuring quality business paper present sentiment analysis approach business review classification using large review dataset provided yelp yelp challenge dataset work propose several approach automatic sentiment classification using two feature extraction method four machine learning model illustrated comparative study effectiveness ensemble method review sentiment classification sentiment analysis opinion mining classification text review | 1 | 0.8196310997 | 1 |
| also evaluate piccolo model number existing work specifically train 103 gru model yelp dataset used focus classification task | evaluating accuracy used test business review dataset e filtered yelp challenge dataset co n million text review apply preproces stop word apply stemming handling n unigrams feature vector representatio n previous section | 0 | 0.4483153522 | 2 |
| day social site like facebook twitter generally utilized posting client audit various thing example film news food style governmental issue considerably charu nanda writes research sentiment analysis film audit hindi language examined online audit receivedfamiliarity individual making choice assistance future choice based artificial intelligence ai similarly many creator examination client audit various cycle manner business future anticipated | sentiment analysis ha become important research area understanding people opinion matter analyzing large amount information million people express thought various service product using social networking site blog popular review site active feedback people valuable company analyze customer satisfaction monitoring competitor | 0 | 0.6113451719 | 1 |

Figure 3.6: Data Sample after feature Extraction.

In short, our comprehensive feature extraction process for validating references within scholarly articles considers both contextual similarity and the presence of common name entities. Through a rigorous analysis of the reference line and the referenced scholarly paper, we derive features that form a sturdy foundation for automating the reference validation process. Using these features will allow advanced NLP techniques to use relevant and optimized features to learn patterns allowing models to learn complex patterns of scholarly papers. This methodology not only enhances the precision and efficiency of our reference validation model but also reinforces the robustness of our research efforts.

## 3.3 Model Specification

After extracting the four crucial features outlined in the previous process, we employed them to train an XLNet model for the task of validating scholarly paper references. Here's a detailed breakdown of the steps involved in training the model:

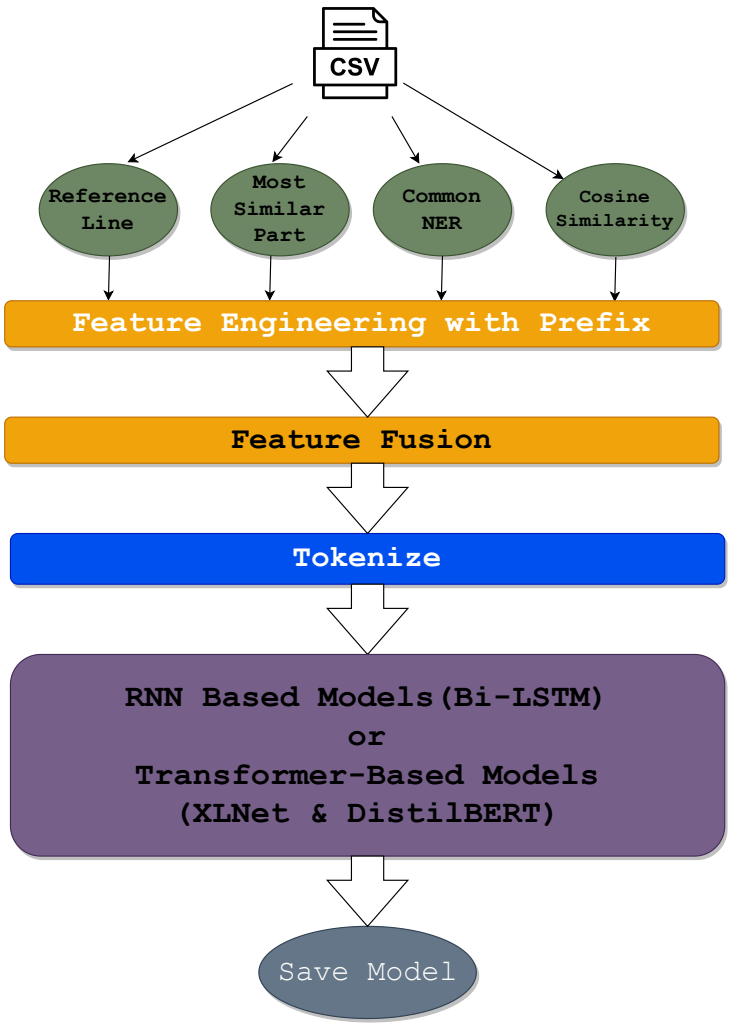



Figure 3.7: Training Phase of Sequence-Based Models

### 3.3.1 Feature Engineering with Prefix

To further enhance the capabilities of learning patterns and distinguish the extracted features accurately, we introduced another novel approach by adding such prefixes that allow NLP models to get more context of the data while training. Without this approach, when we concatenate columns for training, it becomes hard for a model to learn patterns by distinguishing the referred string and most similar chunks texts once these two columns need to be concatenated before tokenizing to feed into models. As a solution, we propose to add prefixes between these columns such as "Part 1:", "Part 2:", "Cosine Similarity:", and "Common Name Entity:". By inserting these markers, a model gains the ability to differentiate between features even after the concatenation process which ensures a clear understanding of data.

Moreover, this approach allows the model to get valuable insights into the nature of other numerical features present in the dataset. As a result, the overall performance of most of the models improved for this application due to recent models advanced context-capturing capabilities.

### 3.3.2 vectorization and Tokenization

For Machine Learning models, we encoded the textual data into a numerical representation. In contrast, when preparing the textual data for feeding into the Sequence models, we tokenize the text using the corresponding tokenizer. These tokenizers are specifically designed to handle large-scale language understanding tasks and can handle the complex contextual relationships present in scholarly papers.

### 3.3.3 Model Architecture

Random Forest model works in a method that combines the predictions of multiple Decision Trees. Each decision tree in this algorithm is trained on randomly selected data and features to decrease over-fitting and increase robustness. This model creates multiple Decision Trees from the dataset by creating random subsets of the original dataset. This technique is known as bootstrapping. Gini Impurity is calculated for making these Decision trees. If the probability of samples belonging to class $i$ at a specific node is $P_i$. Gini Impurity,

$$Gini = \sum_{i=1}^{C} P(i) \times (1 - (P_i)^2) \tag{3.2}$$

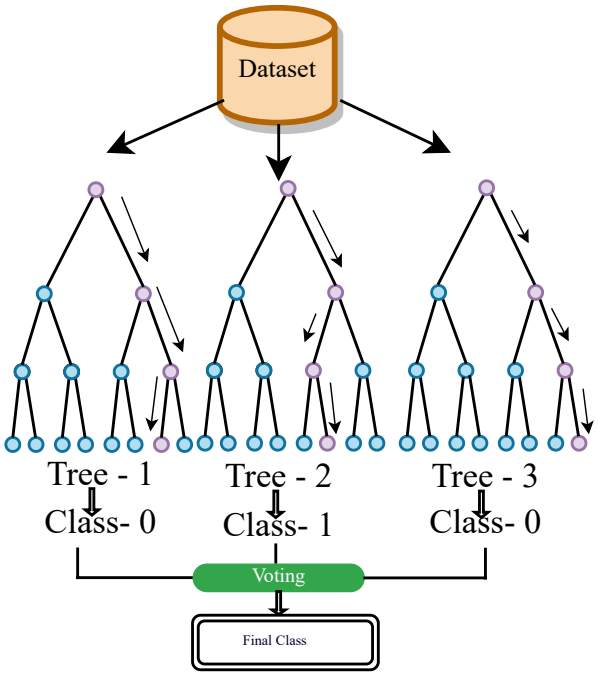



Figure 3.8: Decision process of Random Forest Model.

SVM models construct a hyperplane in high-dimensional space to separate classes to find the hyperplane that maximizes the margin between classes. These types of models can work with both linear and non-linear decision boundaries through kernel functions. Linear SVM When there are two classes, SVM can separate them using a single straight line (2D). The Linear Equation is-

$$wx + b = 0 \tag{3.3}$$

When there are more than two classes, the model can not separate them using a straight line thus this is called Non-Linear SVM. The hyperplane is the best decision boundary. The number of dimensions of the hyperplane is $n - 1$, where $n$ is the number of features.

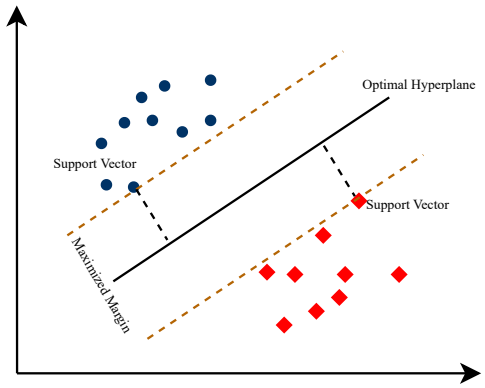



Figure 3.9: SVM Model for Binary Classification

KNN is one of the simplest Machine Learning algorithms based on the Supervised Learning technique. This algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well-suited category by using the KNN algorithm. It does not learn from the training dataset initially, it stores the dataset and at the time of classification, it acts on the dataset. For this reason, it is also called the lazy learner algorithm.

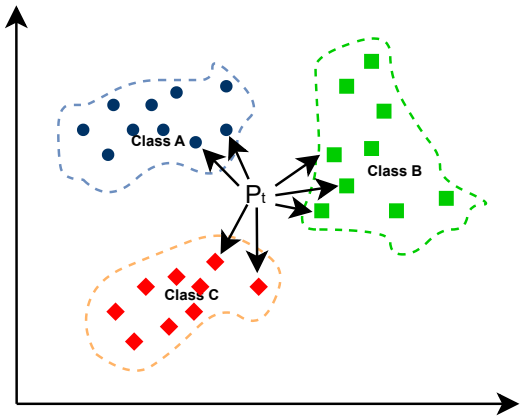



Figure 3.10: Classification Process overview using KNN

MLP is one type of neural network model. It has three types of layers—the input layer, output layer, and hidden layer. An arbitrary number of hidden layers that are placed in between the input and output layers are the true computational engine of the MLP. The neurons in the MLP are trained with the backpropagation learning algorithm.

Gradient descent is an optimization algorithm which is used to minimize the cost function in linear regression. In this algorithm, the model's parameters are iteratively updated by computing the partial derivatives of the cost function with respect to each parameter and adjusting them in the opposite direction of the gradient.

Bidirectional LSTM (BiLSTM) is an RNN-based sequence model. This model consists of two unidirectional LSTMs which process the sequence in both forward and backward directions. BiLSTM architecture has two separate LSTM networks. Between them, one gets the sequence of tokens in the forward direction while the other one gets in the backward direction. The final output is the model is the combination of the probabilities given by these Unidirectional networks. It can be represented as:

$$p_t = p_t^{forward} + p_t^{backward} \tag{3.4}$$

Where $p_t^{forward}$and $p_t^{backward}$ are the returned probabilities from both unidirectional LSTM.

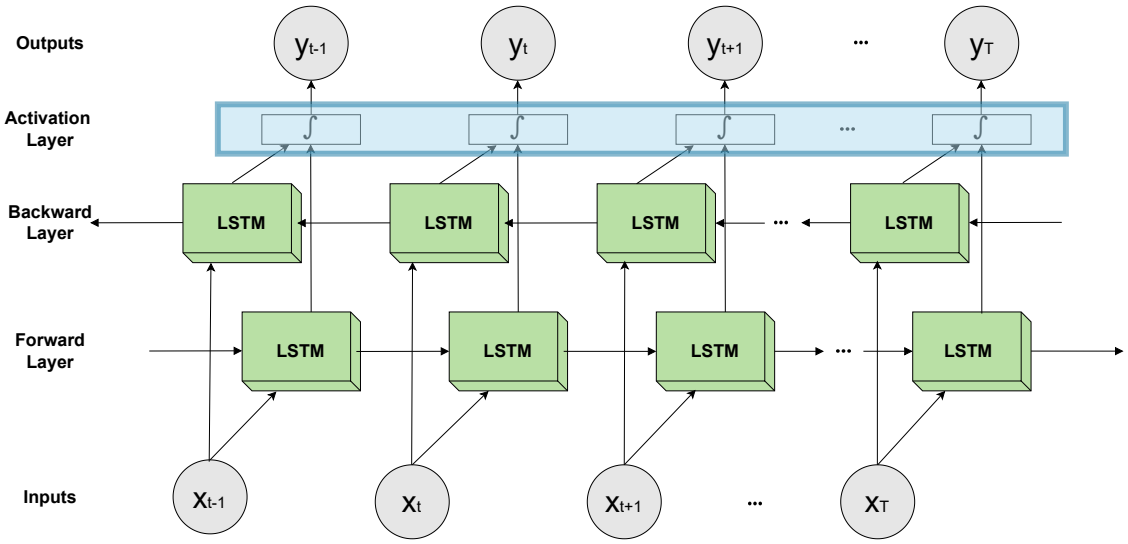



Figure 3.11: Architecture of BiLSTM

DistilBERT and XLNet are transformer-based models that excel in sequential tasks, making them well-suited for understanding the context of the text. These models have bidirectional capabilities, allowing them to consider both the left and right context when making predictions, which is advantageous for tasks like reference validation. DistilBERT is a distilled version of BERT model, it is a compact moels that runs faster than BERT while preserving more than 95% performance. In contrast,

XLNet is an autoregressive. This model captures bi-directional context using Permutation Language Modeling (PLM) method. In this model, the upcoming token depends on all previous tokens.

### 3.3.4 Loss Function

For training, we employed the CrossEntropy Loss (also known as log loss). This loss function measures the dissimilarity between predicted class probabilities and true class labels, making it suitable for binary classification tasks like reference validation. Our Dataset three-class classification and when we drop the confused class, it becomes binary classification from multiclass classification. If the number of classes is represented by $M$, and the number of classes $M$ equals 2, we need a binary classification Cross Entropy formula, where Binary Cross-Entropy(BCE) can be calculated as:

$$-(y \log(p) + (1 - y) \log(1 - p)) \tag{3.5}$$

On the other hand, when $M > 2$, We calculate loss using multiclass classification formula,

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{3.6}$$

Where y = binary indicator (0 or 1) if class label c is the correct classification for observation o and p = predicted probability observation o is of class c

### 3.3.5 Optimization Algorithm:

We used the Adam optimizer, a popular choice for training deep neural networks. Adam adapts the learning rate during training to converge efficiently and reach a better local minimum.

## 3.4 Implementation

In this section, we delve into the practical execution of our proposed solution. This allowed us to truly visualize how our theoretical holistic approach performs in real life as we were able to get tangible results through our comprehensive implementation.

### 3.4.1 Implementation of Feature Extraction

At first, we used the Pandas library to convert our CSV into Panda's Dataframe and then read each row one by one. For each row, we used some specific steps to extract relevant features for training our model and those steps are-

One of the important columns of our dataset is used to contain a Referred PDF, so to validate it with a reference line, we need to first extract data from the corresponding referred PDF. To do that, we have used the PyPDF2 library to Parse PDF data into text. Then, we used an advanced regular expression to find section headers such as 'Introduction', 'Methodology', etc. and used these section headers to split the entire corpus into manageable smaller sections. Before breaking this

into even smaller chunks of text, we did some data-cleaning steps using regular expression and the nltk library. It starts with sentence tokenization, breaking down the text into individual sentences. Numbers within square brackets are removed, and words are tokenized. The text is then converted to lowercase, and punctuation is removed. This cleaning process results in a refined version of the text, ready for further analysis.

Since our approach involves breaking down these texts into smaller chunks by using our collection of topic-changing lists of words or phrases, to break down the above-preprocessed data, we applied the same data cleaning steps to these topic-changing lists of words or phrases. Later on, we used these small sections of text to break down into even smaller chunks using those preprocessed topic-changing lists of words or phrases. But to ensure the contextual flow, we incorporated the idea of topic modeling on each smaller chunk of texts to observe if any consecutive texts share the same topic using BERTopic. This is used to merge the consecutive chunks of texts with the same topic to maintain the contextual flow of the part. Later on, the finalized chunk of text collection is further preprocessed using the nltk library to remove stop words and lemmatized the text. Similarly, in a parallel manner, we use all the preprocessing steps on the reference line to keep the chunk of text collection and reference line in the same format and make our first extracted feature by storing this into a new CSV file.

Now from here, we take 2 separate initiatives, one, we find a similarity score between the reference line and each of the chunks of text, and second, we find if there is any common name entity between the reference line and all the chunks of texts to create a binary feature of common name entity existence. Firstly, to find the similarity between the reference line and each chunk of text, we first generate the embeddings of all of them using the 'sentence-transformers/all-MiniLM-L6-v2' model in n-dimensional vector space and then calculate the cosine similarity of these vectors/tensor using pytorch_cos_sim() this function. After that, the next step is to find the highest cosine similarity and its corresponding preprocessed chunk of text. To use these two things as our new feature we also stored these two things in the CSV file in the same row of the preprocessed reference line. Secondly, we used Spacy's 'en_core_web_lg' as our NER model to identify the name entities of reference lines and chunks of text collection. Upon finding any common name entity we created a binary feature by storing 1 in a new column while 0 for having no common Name Entity. This is how we used various concepts of NLP techniques to extract relevant features for training different ML and NLP models.

### 3.4.2 Implementation of Models

After extracting the four crucial features outlined in the previous process, we employed them to train the models for the task of validating scholarly paper references. Here's a detailed breakdown of the steps involved in training the model:

#### Model Definition

In our research, we used mainly two types of models. One is general Machine Learning Models and another is Sequence Models for Natural Language Process-

ing. Among general Machine Learning Models, we used a Support Vector Machine (SVM), Gradient Boosting Classifier, K-nearest neighbors, Random Forest Classifier, and MLP Classifier. These models can not understand human languages or text very well which is why texts are represented by vectorization but as our features include long text sequences, we used some Sequence Models for Natural Language Processing which are Bidirectional LSTM, DistilBERT, and XLNet. These models are designed to understand and process human language.

### General ML Models Implementation

To train our general ML models, first, we separated the numerical and categorical columns, so that we can vectorize the textual features of our dataset into numerical representations. For vectorization, we used TF-IDF which was imported from the 'sklearn' library. From this library, we also used StandardScaler to standardize the numerical features of our dataset. As our dataset has a mix of both numerical and categorical features, we used ColumnTransformer to encapsulate all the vectorized and standardized columns of different types of features into a single transformer. After splitting the dataset into train, test, and validation datasets in a 7:2:1 ratio, we defined our models and used another function Pipeline to streamline the process to train the model.

### Sequence Models for Natural Language Processing

For Bidirectional LSTM (BiLSTM), we have used the TensorFlow Keras library and some of its functions. After splitting the dataset into train, test and validation datasets in a 7:2:1 ratio, we used the Tokenizer function tensorflow library to tokenize the categorical columns. We ensured the length of all the training sequences was equal, to ensure that we used 'pad_sequences' which is a common step on NLP while with RNNs models. We have used the same function to standardize numerical features, encapsulate all the encoded and standardized columns, and streamline the process.
To implement DistilBERT and XLNet, we used the PyTorch library along with the transformers library developed by Hugging Face. For tokenizers 'distilbert-base-uncased'and 'xlnet-base-cased' used as different models are compatible with their tokenizer. For both cases, we used a learning rate of $2 \times 10^{-5}$ with a batch size of 32. In the case of epochs, we used 25 to observe the best possible epoch. For DistilBERT's best epoch, we got 22 and 23 for XLNet. Before tokenization, we combined the feature columns with added prefixes. After that, with the help of TensorDataset we created a dataset from the tokenized tensors, and using DataLoader, we set up a data loader to efficiently feed batches of data to the model during training and validation.

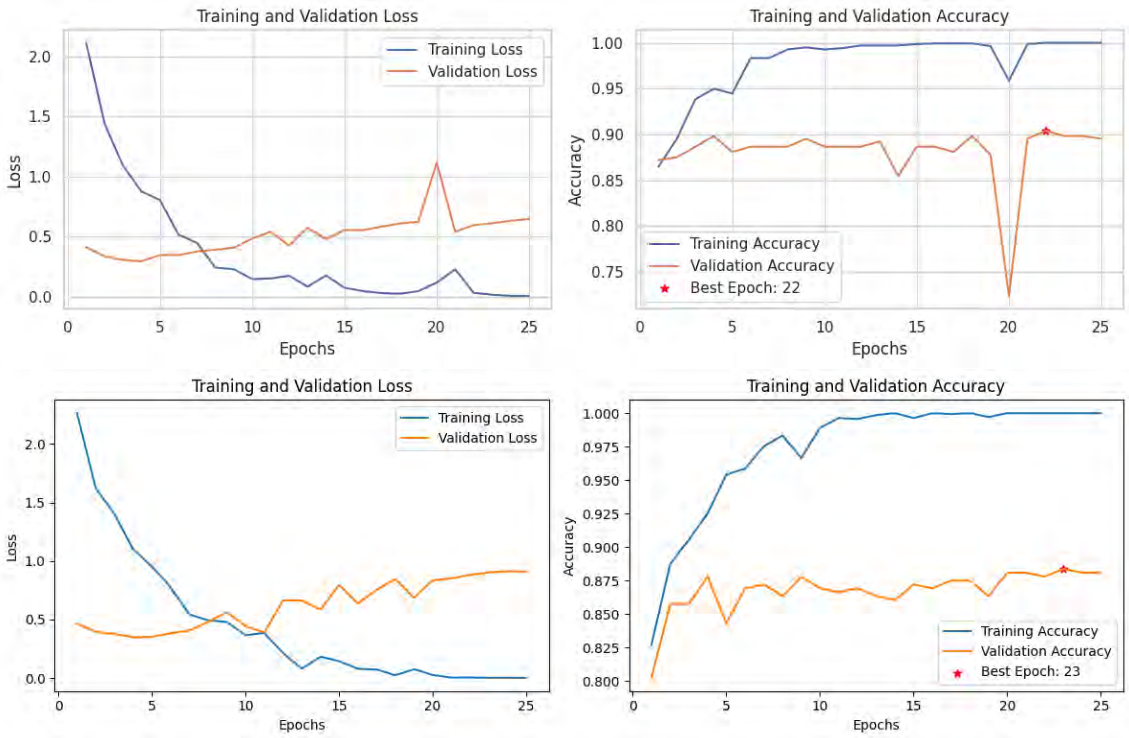



Figure 3.12: Accuraccy and Loss of DistilBERT and XLNet

After that, this data was fed into the models for training. During training, for optimizing the training 'adam' optimizer was used, and 'binary_crossentropy' was used as a loss function. After training the models, we evaluated these models by calculating the accuracy and Classification Report, accuracy_score, and classification_report were used to calculate our evaluation metrics. The weighted average of these metrics was considered in our research due to having imbalanced data. Lastly, the confusion matrix was plotted using the confusion_matrix function from 'sklearn' and matplotlib library.

# Chapter 4

# Results and Discussion

## 4.1 Performance Metrics

To evaluate the model's performance, we considered various metrics, including:

- **Accuracy:** Measures the proportion of correctly classified references out of the total.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

  Although accuracy provides an overall sense of model performance, it may not be suitable for imbalanced datasets.

- **Precision:** Quantifies the proportion of true positive references out of all predicted positives, indicating how reliable positive predictions are.

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

  Precision is emphasized when the cost of false positives is high. It reflects the model's ability to avoid making incorrect positive predictions.

- **Recall:** Measures the proportion of true positive references out of all actual positives, indicating the model's ability to identify valid references.

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

  Recall is particularly important when the cost of missing positive instances is high. It offers valuable insights into how well the model can detect and capture all instances belonging to the positive class.

- **F1-Score:** A harmonic mean of precision and recall, providing a balanced measure of a model's performance.

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4.4}$$

  F1-score is valuable when there is an uneven class distribution or when the consequences of false positives and false negatives are not equal. It provides a comprehensive measure of a model's performance.

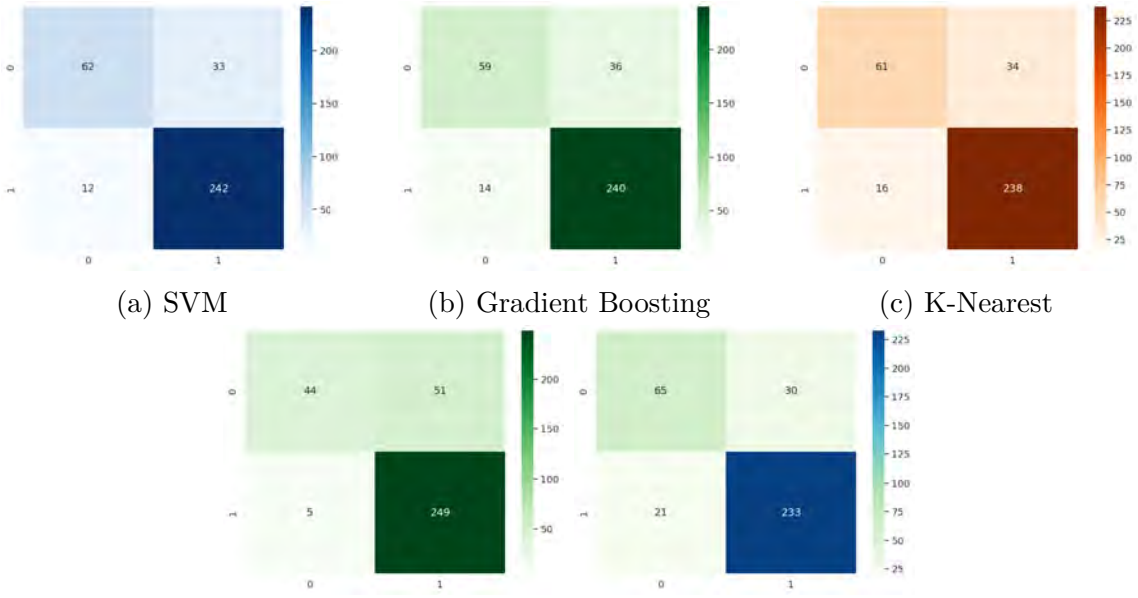
## 4.2 Model Performance Evaluation

In this binary classification scenario, these class labels are used: "Not Valid," "Valid,". The confusion matrix effectively summarizes several machine learning models' performance (Fig 4.1) with respect to these classes.

Out of 286 instances, the SVM model identifies 241 instances as True positives and 9 instances as True Negatives. The false positive score is 34 and the false negative score is 2. In binary classification, the SVM model's overall accuracy is 87%; precision and recall are similar to the accuracy. However, the F1 score decreased to 84%.

The gradient boosting model's true positive score is 243 and its true negative score is 6. The false positive score is 37 and there is no False negative. The gradient-boosting classifier model's accuracy and recall are similar to that of the SVM model. But the precision score is the highest, which is 89%. However, the F1 score decreased greatly from the precision score, which is 83%.

The KNN model's true positive score is 239 which is slightly less than the previous two models. But the true negative score is 12 which is slightly higher than SVM and Gradient boosting algorithms. The KNN model wrongly identified 31 instances as positive and 4 instances as negative. The KNN model's overall accuracy, recall, and F1 score are slightly higher than the SVM and gradient boosting model but the precision score decreased which is 86%.




(a) SVM (b) Gradient Boosting (c) K-Nearest

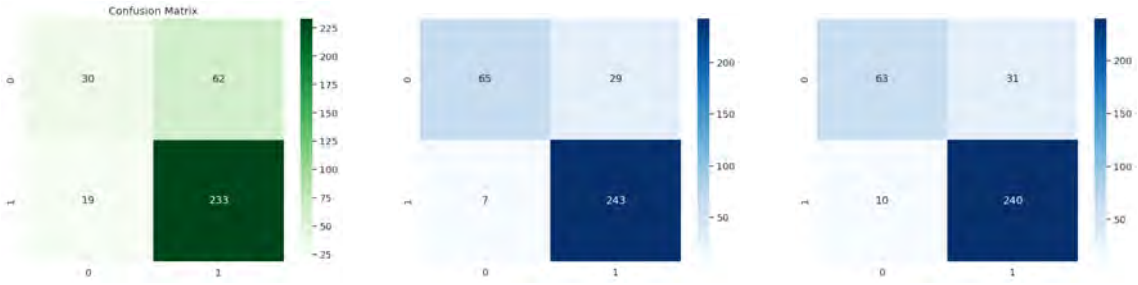
(d) Random Forest (e) MLP

Figure 4.1: Confusion Matrix of Machine Learning Models

The random forest model correctly identified 243 instances as positive and 9 instances as positive. The model wrongly identified 37 instances as positive. The performance of the random forest and gradient boosting model is identical.

MLP model's true positive score is 225, the worst score among other models. But the true negative score is 21 which is the highest score. The model wrongly identified 18 instances as negative. The accuracy and precision score are 86% which is the lowest score among other models. The recall score is much lower than other models, which is 85%. But the model's F1 score is the highest, which is 86%.

Overall we can say that the gradient boosting, the random forest, and the KNN model performed better in terms of accuracy, precision, and recall score. And MLP performed better in terms of F1 Score.




(a) BiLSTM (b) DistilBERT (c) XLNet

Figure 4.2: Confusion Matrix of Sequence Models

The BilSTM model accurately identified 228 instances as positive and 19 instances as negative. The false positive score is 24 and the False negative score is 15. The accuracy and Recall score is 87% which is the same as the Support Vector Machine. The F1 score is similar to KNN which is 85%. So performance did not improve compared to the machine learning models.
The DistilBERT model's true positive score is 240 which is an improvement from the BilSTM model. The False positive score is 24 and the False negative score is 15. The accuracy and the recall score are 89% which is much better than the BilSTM model. But the F1 score is lower which is 84%.
The XLNet Model accurately identified 249 instances as positive which is higher than the previous two models. The false Negative score is 5 which is also the lowest score. The model's accuracy and recall is 89% which is the same as DistilBERT. The precision score is significantly improved from the previous two models which is 90%.

### 4.2.1 Performance of Machine Learning Models

To evaluate our model, in classification reports, the weighted average score of precision, recall, and f1-score is considered, besides the overall accuracy of the model's performance for all the classes. The weighted average provides a representative measure of the model's performance by giving more weight to classes that have a larger number of samples and less weight to classes that have a small number of samples. It was considered that the class distribution is imbalanced in our dataset.

Among our used models, From the Support Vector Machine (SVM) we got an accuracy of 0.83 with precision, recall, and f1-score of 0.81, 0.83, and 0.81 respectively.

Table 4.1: Classification Report (Weighted Avg) of Machine Learning models

| Model Name | With Confused Label | | | | Without Confused Label | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F1-Score | Accuracy | Precision | Recall | F1-Score |
| Support Vector Machine (SVM) | 0.83 | 0.81 | 0.83 | 0.81 | 0.87 | 0.87 | 0.87 | 0.87 |
| Gradient Boosting Classifier | 0.84 | 0.81 | 0.84 | 0.82 | 0.86 | 0.85 | 0.86 | 0.85 |
| K-Nearest Neighbors | 0.84 | 0.81 | 0.84 | 0.82 | 0.86 | 0.85 | 0.86 | 0.85 |
| Random Forest Classifier | 0.83 | 0.81 | 0.83 | 0.80 | 0.84 | 0.85 | 0.84 | 0.82 |
| MLP Classifier | 0.84 | 0.81 | 0.84 | 0.82 | 0.85 | 0.85 | 0.85 | 0.85 |

Gradient Boosting Classifier was able to perform with 0.84 accuracy 0.81, 0.84, and 0.82 as precision, recall, and f1-score which outperformed SVM in all these performance metrics. K-Nearest Neighbors then performed similarly with an accuracy of 0.84 and precision, recall, and f1-score of 0.81, 0.84, and 0.82 outperforming the Random Forest Classifier, having an accuracy of 0.83 with precision, recall, and f1-score of 0.81, 0.83 and 0.80 respectively. Lastly, the MLP Classifier same as K-Nearest Neighbors and Gradient Boosting. This performance of the MLP Classifier drops significantly when we drop an output label and repeat the whole process. In all these cases above, our dataset has three output labels which are 0,1,2 indicating invalid, valid, and confused reference validity. We train and evaluate these models again without the confused output label (class 2) with the hope of better performance. After dropping the confused label, all the models were able to give better results in all these performance metrics. But in this case, SVM secured the place of most well-performed model among these with accuracy, precision, recall, and f1-score of 0.87, 0.87, 0.87, and 0.87. In terms of all our metrics, both the K-Nearest Neighbors Classifier and Gradient Boosting Classifier's performance performed equally having 0.86, 0.85, 0.86, and 0.85 as accuracy, precision, recall, and f1-score. Besides, these two models Random Forest Classifier performed with an accuracy, precision, recall, and f1-score of 0.84, 0.85, 0.84, and 0.82 respectively, it was outperformed by the previous two models.

### 4.2.2 Performance of Sequence Models

Among Sequence Models, We worked with Bidirectional LSTM, DistilBERT and XLNet. We tested these models with and without our confused output label and examined the performance difference. In all these Sequence Models, overall perfor-

mance was improved after dropping the confused label entries. Before the drop, Bidirectional LSTM showed 0.76, 0.74, 0.76 and 0.74 and after that, it improved to 0.83, 0.82, 0.83, and 0.82 as accuracy, precision, recall, and f1-score respectively. This way, the remaining two models also showed significant improvement in performance metrics. The performance of DistilBERT jumped to 0.89, 0.89,0.89, and 0.89 from 0.84, 0.83, 0.84, and 0.83. Our XLNet model showed 0.83, 0.81, 0.83, and 0.76 before dropping the confused class, and after dropping it showed 0.88, 0.87, 0.88, and 0.87 as accuracy, precision, recall, and f1-score.

Table 4.2: Classification Report (Weighted Avg) of RNN based and Transformer models

| Model Name | With Confused Label | | | | Without Confused Label | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F1-Score | Accuracy | Precision | Recall | F1-Score |
| Bidirectional LSTM | 0.76 | 0.74 | 0.76 | 0.74 | 0.83 | 0.82 | 0.83 | 0.82 |
| DistilBERT | 0.84 | 0.83 | 0.84 | 0.83 | 0.89 | 0.89 | 0.89 | 0.89 |
| XLNet | 0.83 | 0.81 | 0.83 | 0.76 | 0.88 | 0.87 | 0.88 | 0.87 |

While General Machine Learning and Sequence Models exhibit comparable overall performance, DistilBERT's exceptional results highlight the benefits of advanced NLP approaches. Continuous exploration and fine-tuning of Large Language Models (LLMs) could further enhance their performance in binary classification tasks.

# Chapter 5

# Limitations and Future Works

This research paper presents a pioneering approach to address reference validity using various ML and NLP techniques. However, there are several limitations that exist. In the following discussion, we focused on these constraints while outlining some additional avenues for future works that can be built upon this paper and further enhance the entire automated reference validity system of scholarly articles.

One of the limitations of the paper is that the dataset is not of a sizable quantity. As more instances in the dataset are added, we expect the models to improve, as then they would have enough data to learn from. For now, we have to work with the data that was collected to implement this model. Moreover, our work does not account for image data. As this research was mostly focused on NLP, we decided to omit the image-processing tasks needed to decode the image or image-based table data. Decoding image data in the future might yield better results. Also, we have done the research for the English language only, meaning that tracing our steps for other languages might not give the same results that we have gotten. There is an interest in expanding this work to other languages in the future.

Since we have collected published paper references only, we have found a lot fewer invalid or confused references, as the papers in this library have gone through a rigorous peer review process. This has been the cause of imbalanced data in our dataset. To solve this problem, we have used few-shot learning and undersampling techniques, but it was to no avail. The reasoning is that our most minority class data also known as the confused label has insufficient representation in the few-shot samples. Another problem arises in the fact that the referencing paper will sometimes summarize the referenced paper into a paragraph. Since our methods involve finding the most similar part, it will assign the most similar part to where the summary might have the most contextual similarity. Thus, some references might seem falsely irrelevant. Since the abstract or conclusion of a paper usually has a summary of the paper, this problem is solved automatically most of the time. Even though we have worked on this issue and handled it if the referred parts are close, there is still work that needs to be done.

Addressing the limitations of this work and moving on to solve these problems is critical to further advance this research to the next stage. Some potential future works are suggested below.

Since our dataset has not been of a sizable quantity for this field of research, we have inferred that an increase in instances of data will help the models to learn. Thus, this can be identified as one of the future works that can be beneficial for improving automated reference validation. Moreover, the dataset imbalance nature can be handled more effectively in the future. Next, recent advanced LLMs from recent times have exhibited great performance in contextual understanding. So, newer LLMs such as GPT-4, Gemini Ultra, etc. can be applied for further potential improvements in accuracy and precision. Additionally, creating a fusion of NLP and Computer Vision in order to extract relevant data from images will definitely make the entire system more robust. As image-processing methods will help us to analyze images and read table data that we could not read before, it will certainly boost our overall results.

# Chapter 6

# Conclusion

Reference validation stands out as an exceptionally demanding and time-consuming aspect of the peer review process, wielding considerable influence over its overall efficiency. The difficult nature of this task can pose significant hurdles, potentially impeding the timely dissemination of valuable scientific knowledge. In solving these challenges, our research is dedicated to tackling this issue with a commitment to the development of an automated reference validation system. Our aim is not only to simplify but also to accelerate the peer review process. In doing so, we intend to remove the substantial burden that reviewers bear and, in turn, catalyze the accelerated sharing of groundbreaking scientific discoveries. This undertaking represents a pivotal stride forward in our collective efforts to refine and enhance the peer review system. By automating reference validation, we aspire to bring about a transformation that will elevate the scientific publishing experience to new heights. This transformation will not only benefit the academic community but also contribute to the broader advancement of knowledge dissemination across various fields of study. In conclusion, our automated reference validation system has streamlined peer review, saving time and effort for reviewers, enhancing accuracy, improving publication efficiency, and reducing academic misconduct. This innovation represents a significant leap forward in optimizing the peer review process and fostering a more robust and trustworthy scientific publishing ecosystem.

# Bibliography


[1] A. Islam and D. Inkpen, "Semantic text similarity using corpus-based word similarity and string similarity," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 2, Jun. 2008, ISSN: 1556-4681. DOI: 10.1145/1376815.1376819. [Online]. Available: https://doi.org/10.1145/1376815.1376819.

[2] D. Ruths and F. A. Zamal, "A method for the automated, reliable retrieval of publication-citation records," *PLOS ONE*, vol. 5, no. 8, pp. 1–8, Aug. 2010. DOI: 10.1371/journal.pone.0012133. [Online]. Available: https://doi.org/10.1371/journal.pone.0012133.

[3] A. Graves, "Long short-term memory," in *Supervised Sequence Labelling with Recurrent Neural Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 37–45, ISBN: 978-3-642-24797-2. DOI: 10.1007/978-3-642-24797-2_4. [Online]. Available: https://doi.org/10.1007/978-3-642-24797-2_4.

[4] X. Li, M. Thelwall, and D. Giustini, "Validating online reference managers for scholarly impact measurement," *Scientometrics Scientometrics*, vol. 91, no. 2, pp. 461–471, 2012. DOI: https://doi.org/10.1007/s11192-011-0580-x. [Online]. Available: https://akjournals.com/view/journals/11192/91/2/article-p461.xml.

[5] E. Barroga, "Reference Accuracy: Authors', Reviewers', Editors', and Publishers' Contributions," *Journal of Korean Medical Science*, vol. 29, no. 12, pp. 1587–1589, Dec. 2014. DOI: 10.3346/jkms.2014.29.12.1587. [Online]. Available: https://doi.org/10.3346/jkms.2014.29.12.1587.

[6] S. Guraya, "Accuracy of references in scholarly journals: An analysis of 450 references in ten biomedical journals," *European Science Editing*, vol. 40, pp. 88–90, Aug. 2014.

[7] 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.

[8] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2383–2392. DOI: 10.18653/v1/D16-1264. [Online]. Available: https://aclanthology.org/D16-1264.

[9] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, "RACE: Large-scale ReAding comprehension dataset from examinations," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 785–794. DOI: 10.18653/v1/D17-1082. [Online]. Available: https://aclanthology.org/D17-1082.

[10] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1706.03762.

[11] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. DOI: 10.18653/v1/W18-5446. [Online]. Available: https://aclanthology.org/W18-5446.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. [Online]. Available: https://aclanthology.org/N19-1423.

[13] Y. Liu, M. Ott, N. Goyal, *et al.*, *Roberta: A robustly optimized bert pretraining approach*, 2019. arXiv: 1907.11692 [cs.CL].

[14] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification?" In *Chinese Computational Linguistics*, M. Sun, X. Huang, H. Ji, Z. Liu, and Y. Liu, Eds., Cham: Springer International Publishing, 2019, pp. 194–206, ISBN: 978-3-030-32381-3.

[15] T. Thongtan and T. Phienthrakul, "Sentiment classification using document embeddings trained with cosine similarity," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 407–414. DOI: 10.18653/v1/P19-2057. [Online]. Available: https://aclanthology.org/P19-2057.

[16] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, *Xlnet: Generalized autoregressive pretraining for language understanding*, 2020. arXiv: 1906.08237 [cs.CL].

[17] H. Batra, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, "Bert-based sentiment analysis: A software engineering perspective," in *Database and Expert Systems Applications*, C. Strauss, G. Kotsis, A. M. Tjoa, and I. Khalil, Eds., Cham: Springer International Publishing, 2021, pp. 138–148, ISBN: 978-3-030-86472-9.

[18] P. Dutta, M. Alam, M. Rahman, M. Abdullah-Al-Wadud, M. Islam, and M. Dewan, "Document-bert: A transformer-based automated referencing validation model for scholarly publications," *Brac University*, 2021.

[19] F. Abbas, "An Improved NLP for Syntactic and Semantic Matching using Bidirectional LSTM and Attention Mechanism," May 2022. DOI: 10.5121/csit.2022.120906. [Online]. Available: https://doi.org/10.5121/csit.2022.120906.

[20] A. Jangabylova, A. Krassovitskiy, R. Mussabayev, and I. Ualiyeva, "Greedy texts similarity mapping," *Computation*, vol. 10, no. 11, 2022, ISSN: 2079-3197. DOI: 10.3390/computation10110200. [Online]. Available: https://www.mdpi.com/2079-3197/10/11/200.

[21] M. Ji and X. Zhang, "A Short Text Similarity Calculation Method Combining Semantic and Headword Attention Mechanism," *Scientific Programming*, vol. 2022, pp. 1–9, May 2022. DOI: 10.1155/2022/8252492. [Online]. Available: https://doi.org/10.1155/2022/8252492.