Detecting Sarcasm in Bengali Comments using NLP

by

Md. Jamiur Rahman Chowdhury
18101448

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 2023

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. I have acknowledged all main sources of help.

Student's Full Name & Signature:

[illegible]

Md. Jamiur Rahman Chowdhury

18101448

Approval

The thesis/project titled "Detecting Sarcasm in Bengali Comments using NLP" submitted by

1. Md. Jamiur Rahman Chowdhury (18101448)

Of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 19, 2023. .

Examining Committee:

Supervisor:

[illegible]

Mr. Arif Shakil
Lecturer, Department of CSE
BRAC University

Co-Supervisor:

Farig Yousuf Sadeque

Dr. Farig Yousuf Sadeque
Assistant Professor, Department of CSE
BRAC University

Head of the Department:

Dr. Sadia Hamid Kazi
Chairperson and Associate Professor, Department of CSE
BRAC University

Program Coordinator:

Dr. Golam Rabiul Alam
Professor, Department of CSE
BRAC University

Abstract

Natural Language Processing (NLP) is a subset of Machine Learning which resides at the intersection of Linguistics and Computer Science. It deals with the capability of computers to learn and work with human languages. With the emergence of social media platforms, modern-day communication is being digitalized more than ever. To keep up with this rapid flow of development, the advancement of automated text processing and artificial language interpretation has become necessary. These concerns have given birth to a domain called Sentiment Analysis where blocks of text are processed to extract prominent sentiments that are prevalent within them. These sentiments can be happiness, sadness, anger, disgust, etc. Over the past few years, similar studies have garnered the attention of a vast number of computer scientists and linguists but as the study progresses and expands in the form of languages, concentrations, and contexts more and more challenges have started to show up. One of these challenges is the interpretation of figurative language. Figurative language refers to the structure of speech where the actual meaning defers from the literal meaning. The best example of this is Sarcasm which is a sort of figurative language used with an intention of mockery or humor. Detecting sarcasm is considered to be one of the most challenging tasks in the domain of NLP due to the figurative structure and creative nature of sarcastic texts and the lack of relevant data on the internet. Determining sarcasm can often be difficult for even human beings as one has to have a strong understanding of the context to detect sarcasm. However, many studies have achieved respectable results by following the context unaware unimodal methods using classical Machine Learning, Deep and Hybrid Neural Networks. Motivated by such research, the objective of this paper is to take a step toward detecting sarcasm in the Bengali Language domain using Support Vector Machine (SVM), Cogniinsight(Word2Vec), and Bidirectional Encoder Representations from Transformers (BERT) on a novel dataset. To the best of my knowledge, this will be the first-ever initiative taken toward detecting sarcasm in Bengali Language using BERT.

Keywords: Natural Language Processing; Sentiment Analysis; Machine Learning ; Support Vector Machines; Word2vec; BERT

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to my advisor Arif Shakil Sir and Co-advisor Dr. Farig Yousuf Sadeque Sir for their kind support and advice in our work. They helped me whenever I needed help.
And finally to my parents without their throughout support it would not be possible. With their kind support and prayer I am now on the verge of my graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

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

ANN Artificial Neural Network

$BERT$ Bidirectional Encoder Representation of Transformers

$BiLSTM$ Bidirectional Long Short Term Memory

$LSTM$ Long Short Term Memory

MLM Masked Language Modeling

MLP Multi Layer Perceptron

NLP Natural Language Processing

NLU Natural Language Understanding

NSP Next Sentence Prediction

RNN Recurrent Neural Network

SVM Support Vector Machine

$TF - IDF$ Term Frequency-Inverse Document Frequency

Chapter 1

Introduction

This chapter addresses the motives behind this research and elaborates on the importance of analytical exploration in this field. The background section discusses the current state of sentiment analysis and figurative language detection and its importance, the problem statement addresses the issue that this research aims to solve and the research objective section outlines the core targets that this research is conducted around.

1.1 Background

Sentiment analysis is one of the most important applications of Natural Language Processing which mainly focuses on detecting the emotions that are underlying within a block of text by analyzing the words that make up the sentence and their correlation to each other. Here each sentence is analyzed to generate a local context and these contexts are further concatenated to create a global meaning of a paragraph. Sentiment analysis aims to detect the emotions within a sentence by detecting the polarity of the positive and negative nature of the meaning of that sentence. The sentences are scored to create a spectrum of positivity where the distinctions can be: very positive, positive, neutral, negative, and very negative. The very negative statements have a score of 1, negative sentences have a score of 2, neutral statements have a score of 3, positive sentences have a score of 4 and very positive sentences have a score of 5. The main motivation for sentiment analysis is to automate the process of understanding statements and taking decisions accordingly. With the rapid development of internet-based communication, in-person communication is being replaced wherever it is possible. These transactions of information are done in many forms such as text messages, posts, digital blog articles, comments, etc. Even though the volume of information is overwhelming and impossible to operate manually, the need for governing the data to ensure appropriacy and to maintain communication protocols hasn't become invalid. This horribly rigorous task can be made simple with the help of sentiment analysis. Utilizing SA businesses can understand the feedback made by the customers in their e-commerce environment, and social media websites can make sure that the information that is being transmitted via their platform is legitimate and appropriate for the demography of users that their system caters to. Books and transcripts are also being made available online these days for the sake of better monetization and more profitable business. This has created the scope and need for text analysis even more.

The frequent and continuous analysis of language will ultimately be beneficial for linguistic studies also as it contributes to the creation of a new norm of narrative for the upcoming generation. We know the mechanics of a human language stay the same with time but the interpretation of it changes and because of this it is important to understand all possible interpretations of a statement before proceeding towards transmitting it. English is the most widely spoken language in the world but there are many other languages in the world which are used by a huge population of people. Bengali is the 7th most spoken language in the world with over 272 million speakers worldwide. This brings the same complexities of automated language comprehension as English to large demography of people. To satisfy the needs of Bengali speakers a number of studies have been conducted that deal with the classical sentiment analysis problems but in the domain of the Bengali language. Despite the increasing frequency of research in the field, there are a few research gaps in the Bengali language domain that are yet to be addressed properly. One such segment is the detection of figurative language such as Sarcasm in the Bengali Language. To automate the machine interpretation of a human language it is important to figure out patterns within all forms of expressions in that language. So that we can create rules and embed them into learning algorithms and ultimately teach the machines to predict our desired outputs accordingly. Previous works in the English language domain suggest that the task can be done by implementing rule-based algorithms and deep learning. However, this study aims to implement transfer learning to address the issue.

1.2 Research Problem

With the advancement of technology, the occurrence of online communication has increased and conversations that took place in person previously are being done digitally. This has created a huge repository of information that needs to be understood properly so that proper communication protocols can be maintained and advancements in linguistic studies can be done. Lack of understanding within a language domain can create a plethora of problems for people communicating online. Online conversations lack a multitude of components compared to physical communication such as: In an online conversation the body language of a person is often absent which eliminates the entire nonverbal arc of conversations. Another core aspect that is absent in digital communication is the ambiance of the environment where people do not take the environmental components such as people's facial expressions, noise margin of the conversation space, and the perceived collective attitude of people participating in a conversation. On the other hand, offline conversations maintain better records of a sequence of information which ensures better accountability of speech compared to in-person conversations. Even though these differences are subtle, in the larger picture these subtle differences may build up to complex problems. Some problems that may occur from the lack of proper understanding of a language are misrepresentation, the spread of false information, violation of community standards, etc. All of these problems lead to the disruption of harmony in society.
Although these problems might occur regarding all forms of narratives, the concentration of misinterpretation of language is the highest among the statements that follow a figurative form of language. The most frequently used form of figurative language in social media is Sarcasm which is the act of using irony in order to mock

a person or an idea. The difficulty in distinguishing sarcasm occurs because of its figurative structure. Machines can be taught to detect a certain emotion within a sentence of regular structure pretty easily as the literal meaning is the intended meaning of the statement. But it is hard in terms of figurative language as the literal meaning and the intended meaning are different from each other. This means detecting sarcasm without any prior context associated with the statement can be a very difficult task. But this is necessary as we often see sarcasm being misrepresented and misinterpreted in social media and news portals which often leads to extreme problems.
Even though sarcasm is mostly used to mock people or establish a comedic angle in a situation it can work as a layer to encapsulate any type of emotion within a sentence. For example from comments that hurt a certain group of people or a pattern of ideas, we can determine the underlying agenda of the authors, from comments that are self-deprecating and established dark humor through suggestive comments we can detect suicidal tendencies within a person.
Alongside making an approach towards solving the aforementioned issues regarding language understanding one of the core reasons behind initiating this study has been the lack of research in the Bengali Language domain regarding figurative language detection and language analysis in general.Bengali is the native language of Bangladesh and it is widely used in other countries as well but despite the high number of speakers the figurative form of this language hasn't gotten the attention of many researchers yet. in order to encourage the use of this language in social media platform and digital media the development of the analysis field associated to this language needs to prospers even more.

1.3 Research Objectives

My research aims to achieve the following objectives :

- Initiating figurative language detection in the Bengali Language domain
- Developing an optimal Bengali text processing method
- Creating a figurative language detection framework for the Bengali language
- Detecting sarcasm in Bengali comments efficiently
- Encouraging automated language comprehension

Chapter 2

Literature Review and Related work

In this section, I aim to describe the key technologies that are used to conduct my research and previous works that support and describe the functionalities of those technologies. This section also addresses the origins of the approach that I have taken in order to solve the issues that I am working on and how the previously conducted research that is mentioned here have inspired and motivated me to proceed further.

2.1 Natural Language Processing

Natural language processing or NLP is the subset of Artificial intelligence that deals with the process of making machines capable of comprehending human spoken languages such as English, Spanish, French, Bengali, Hindi, etc. It is the intersection of computer science and Linguistics. NLP is a combination of several learning methods such as Statistical Learning, Deep Learning, and Classical Machine learning. These methods are directed by a rule-based model created for a specific language. This gives NLP the ability to understand, manipulate and generate texts in human languages according to the requirements.

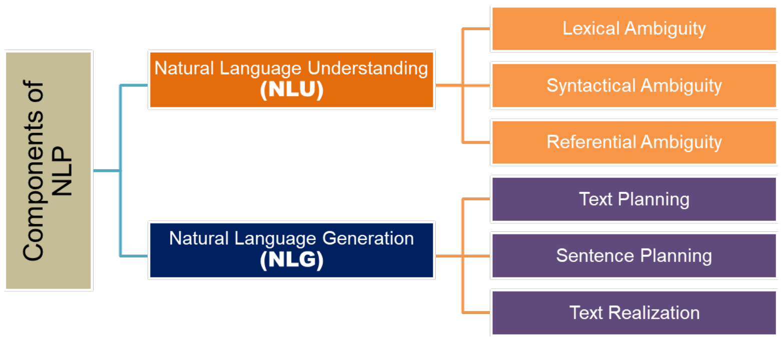


Figure 2.1: Components of NLP.[13]

Modern classical language processing can be divided into two components where one deals with the comprehension of the language and the other works on the generation bit of it. Natural language Understanding or NLU tries to make sense of a sentence fed into the NLP model by looking for lexical ambiguity within it, where there can be multiple interpretations of the literal meaning of a sentence and the actual meaning of the sentence cannot be understood without some sort of additional information associated with the actual sentence. The NLU segment also looks for Syntactic Ambiguity where the order of the words within a sentence is given emphasis while generating interpretations for a sentence. Finally, after making sure that the sentence does not have any syntactic ambiguity or lexical ambiguity the segment looks for referential Ambiguity within the sentence where a unique reference for the specific set of syntax in that sentence is looked for.

The Natural language Generation or NLG segment of an NLP model mainly works with the output that is expected from the entire model. It starts to generate the statements from the base level as texts using the vector representations of the words coming in as an input. The texts are further concatenated to create sentences and the sentences are then used to generate full-fledged statements. After the statements are generated the NLG segment tries to realize the meaning of that sentence and whether it is coherent with the input or not.
In my work NLP mainly facilitates the inputs that are collected in order to detect sarcasm in Bengali comments by providing a toolkit to process that makes the comments more functional and efficient for the models that are in use.

2.2 Sentiment Analysis

One of the most prominent and widely used implementations of NLP is Sentiment Analysis. It is mainly the study of detecting the emotions that are prevalent within a sentence or a block of text. In online communications, some formats do not require the core insights of what a statement represents rather only the positivity or the negativity is taken into account. For example, when a new product is launched into an e-commerce platform for the first time the sellers try to gather user feedback from user comments. Before getting into the constructive information in the comments it can be helpful to get an idea about the positivity: negativity ratio of the comments, this can save time to a larger extent. Sentiment analysis on a higher level harnesses the NLP pipeline in order to distinguish whether the words that are present within a sentence depict a positive or negative meaning. The meanings of the sentences are treated as local entry points initially which are associated with a polarity score and later on the local scores are re-iterated with the concatenated context to calculate the global score which is the indicator of the actual sentiment within that block of a sentence.[4] In a more granular level sentiment analysis works with qualitative distinctions while classifying sentences. A sentence can have a plethora of sentiments embedded into it such as Happiness, Sadness, Anger, Disgust, Surprise, Fear etc

From previous research conducted by Shubham et. al[5], it is suggested that emotion detection can be an effective way of sarcasm detection as sarcasm itself can be treated as an emotion where other emotions are prevalent but not literally expressed. This study shows multiple approaches towards sarcasm detection revolving around sentiment analysis some noticeable methods are :

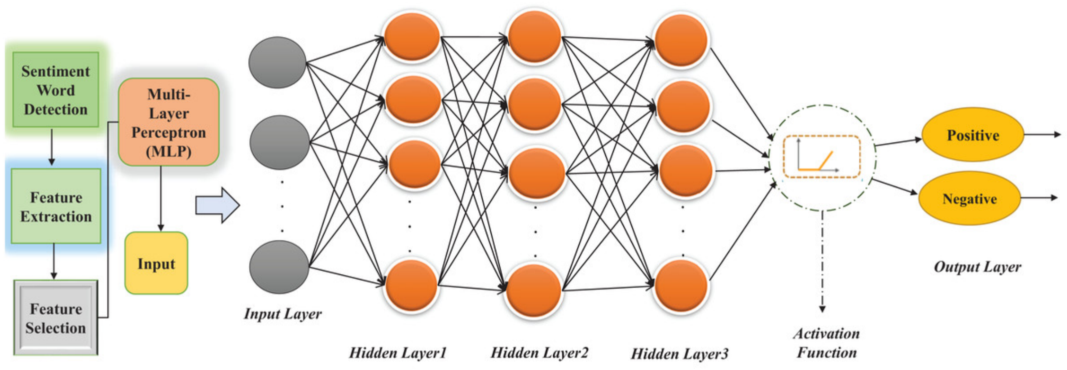


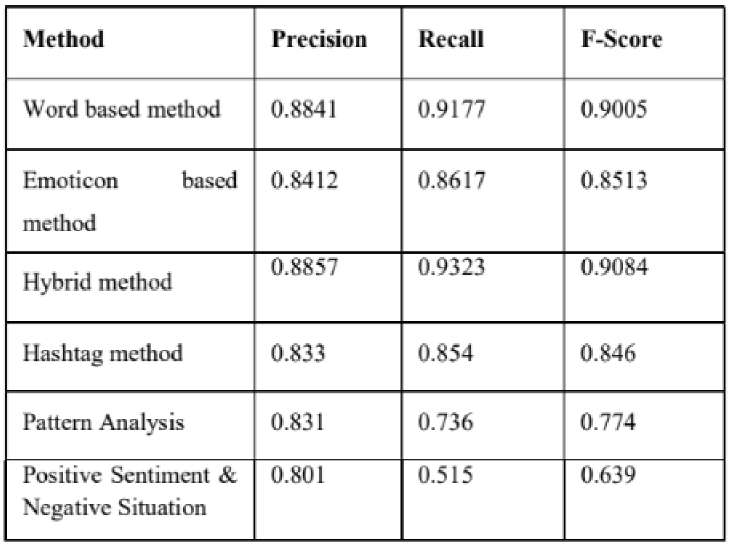
Figure 2.2: Sentiment analysis MLP diagram.[20]

1) A word-based detection approach where the emotions depicted by each word is collected using SentiWordNet to generate a relevant score portraying a specific emotion. The scores are calculated to distinguish a strong emotion for each comment in a feed. After that, the comments are grouped according to their nature of sentiment. The emotion that occurs maximum time is taken as the prominent sentiment of that feed and the other comment groups depicting different emotions are taken as Sarcasm.

2) An emoticon-based approach that emphasizes the emoticons or emojis that are used in a comment. Emoticons can portray an emotion very well without the need for an elaborate description hence, they can be used to classify emotions in a sentence effectively. Using these emoticons, this approach also creates groups of comments that depict a specific emotion. The emotion of the group with the maximum number of comments is taken to be the prominent emotion and the rest of the emotions depicted by the other groups are taken to be Sarcasm.

3) Another interesting approach discussed in this study is a Positive Sentiment and Negative Situation approach where the disparity between the situation and the comments that are reacting to that situation is given priority. At first, the context block is analyzed for a prominent sentiment and then the reaction comments are analyzed in a similar manner as the aforementioned approach to figure out the prominent sentiment within the reaction comments. After that, the sentiments are compared to find out polarity. If the situation is detected to have positive sentiments, the reaction comments that represent the negative sentiments are Sarcastic ones. Other methods in this research are A Hybrid Method, Hashtag Method, and Pattern Analysis Method. Among these methods, the most effective method is found to be the Word-based method which achieves the highest precision score of 0.8841

Another research conducted by J. H. Balanke et. al[7] has suggested that sentiment analysis in sentences is a step forward toward sarcasm detection and it can be done by the popular lexicon Algorithm where words that are fed into the sentiment analysis model are initially compared to the words available in the valence dictionary which mainly contains words associated with a score that indicates the polarity of sentiments in that word. The study addresses the functionality of this



Method	Precision	Recall	F-Score
Word based method	0.8841	0.9177	0.9005
Emoticon based method	0.8412	0.8617	0.8513
Hybrid method	0.8857	0.9323	0.9084
Hashtag method	0.833	0.854	0.846
Pattern Analysis	0.831	0.736	0.774
Positive Sentiment & Negative Situation	0.801	0.515	0.639

Figure 2.3: Method scores [5]

algorithm alongside the need for it to be extended in order to be efficient enough to generate acceptable results. In the process, they have devised systems that extend the lexicon-based algorithm to make it capable of depicting sarcasm more accurately.

1) The first system combines the lexicon-based algorithm with a distinguished sarcasm detection algorithm where the lexicon algorithm performs the polarity calculation of the text data that is fed into the model. The outputs from the lexicon algorithms are grouped into positive and negative statement clusters. The positive statements are then fed into the sarcasm detection algorithm as the authors believe that in order to be sarcastic a statement can only be either positive or neutral in nature.

2) The second system focuses more on the environment in which a statement is generated alongside extending the functionality of the lexicon algorithm. It takes into account the personality, level of education, and narrative of the author while detecting the sentiments present in a statement.

Both systems in the research are mentioned to have similar results with the first system being faster due to fewer calculations per iteration and the second system being a little bit more accurate due to the multi-metric nature of the calculation.

2.3 Figurative Language

Figurative language is the form of language that refers to the intended meaning of a statement and the literal meaning of that statement being different from meach other. With the advancement of technology communication has become digitalised beyond the barrier of formal and casual use only. Nowadays in social media platforms

figurative language is being widely used and the practise is increasing in popularity day by day. This has brought the need for the automation of figurative language comprehension.
Figurative language can be of many forms such as:

1) Metaphore: Here a situation is described with the help of a comparison with another situation to portray the main gist of one situation through its similarities with the other.

2) Simile: A figure of speech where a notion or an idea is expressed or associated to a degree with help of another notion or an idea which has an established distinction of nature.

3) Irony: A sort of narrative where the statement widely differs from the related situation which generated the original statement.

4) Sarcasm: A form of irony used in a way to mock an idea or person or to achieve a comedic angle

5) Satire: A form of speech where exaggeration, irony and humor are used used interchangeably in order to express comedy or ridicule someone

6) Hyperbole: Statements that are exaggerated and not meant to be taken seriously.

In this study, I focus on Sarcasm which is the most common form of figurative language currently being used in the social media platforms and in informal communication. As sarcasm depicts a completely different meaning compared to the literal meaning of the sentence used to express sarcasm it is exceptionally hard to detect sarcasm without any context. It is often hard for even human beings to detect sarcasm in realtime conversations.[12] However form previously done researches we can observe that many approaches of sarcasm detection have been developed over the years that may use the context of the statements or may not . Either ways respectable results have been achieved by most of the studies regarding this.
From a research conducted by P. Verma et al[23] we can observe multiple techiniques of sarcasm detection :

Rule based approached to sarcasm detection are the preliminary approach that lays out the base line thought process for the task. Here basic tasks like lexical, semantic and syntactic analysis are done on the sentences that are fed as an input to the detection models. This approach doesn't deal with pitfalls and exception cases , also doesn't take auxiliary features into account therefore are basic and inefficient compared to the modern methods. There are lexicon based methods that may use lexicon dictionaries or corpuses to detect sarcasm , these methods are a bit better than rule based methods.[6]

The smarter methods in the spectrum are the automated approaches to the problem

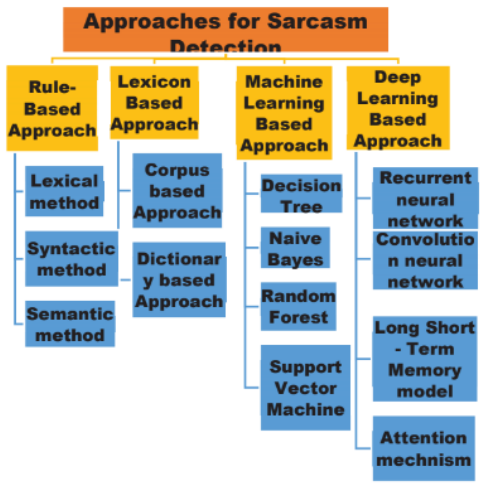


Figure 2.4: Figurative language detection approaches[23]

incorporating machine learning and deep learning methods. From previous studies we can find a pattern based approach towards sarcasm detection. This type of approach is the most common approach in terms of classification problems in the sentiment analysis domain . Here words are embedded to generated scores to depict the polarity of sentiments in them . The scores for a specific sentiment will have a specific range. After the ranges are calculated the scores of new words are used to determine the sentiments within the words. This approach can be found to be more advanced with the introduction of Support Vector Machines within the processing pipeline. This adds a dimensional functionality to the system. After the features are extracted from the dataset through using NLP tools the SVM classifier pins the feature scores and creates clusters with specific distinctions representing each sentiment prevalent in the text data. This enables the model to concurrently distinguish between sentiments and generate accurate outputs.

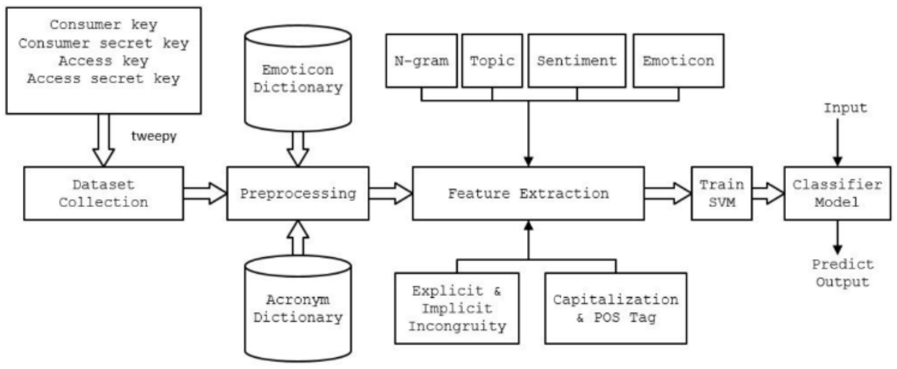


Figure 2.5: SVM based approach in Sarcasm detection[6]

Other approaches include deep learning based methods. Deep learning is a subset of machine learning which aims to mimic the working mechanism of a human brain in machines. These methods introduce multi layer calculation to the sarcasm detection with the help of multi layer perceptrons or MLPs. A study published in 2021 demonstrated that sarcasm detection can be done effectively with the help of Deep Learning. In their research they have used convolutional neural networks to extract features from the data sets that they have gathered.[1] In their study they have associated a lexicon based algorithm in order to generate distinctions between types of words. After that, for the sarcasm detection process they have worked with Incongruity detection where the disagreement between the context and the statement is detected , Hyperbole detection where exaggeration in statements are taken care of, Temporality detection where the time frame of the statement is understood and finally dislike detection. After the detection process a number of machine learning classifiers are implemented in order to generate the accurate outputs . With this process they have achieved a 94 percent accuracy.

A similar study shows that satire can be detected using deep learning methods as well.[22] This could come in handy as satire is closely related to sarcasm and often it can be represented as a subset of it.[27]

Two studies from recent times argue[9] [19] that in sarcasm detection the accuracy and functionalities of the common methods can be extended by introducing multi modal data handling. The common sarcasm detection techniques usually work with only one sort of data at a time such as text data or audio data. But these studies state that if multiple representations of sarcasm are taken into account while building an algorithm the sarcasm detection process can be better in multiple volumes and the detection model will not be limited to only text based data to work with.

One of the studies have used a hierarchical attention layer mechanism where dialog level attention function is used to achieve the context behind the dialogue more efficiently on the other study a CNN and SVM based approach is visible where different data forms are processed for sarcasm detection and finally used to build a model incorporating all outcomes. Both of the models achieve a respectable accuracy while adding volume to the sarcasm detection process.

Even though sufficient amount of techniques are available for sarcasm detection that have shown remarkable results. Still the process of building on the previous methods have to keep going as with time the interpretation of language changes. Something that is considered to be sarcastic today may not be considered sarcastic in the future.[26] As new platforms are emerging and formats of different sorts are being developed sarcasm detection for each format needs to be developed separately. Even though the platforms change most of the approaches will stay the same such as understanding the connection between several emotions and sarcasm , incongruity among statements and their contexts and the figurative nature of sarcastic statements. Keeping these in mind will always help to progress in the process.

2.4 RNN-LSTM

Recurrent Neural Networks or RNNs are neural networks that have a circular input structure where the outputs of the first iteration are used in the second iteration and so on. Due to this repetitive dataflow system, RNNs are a very good fit for sequence modeling problems. A few examples of sequence modeling problems are voice recognition, image recognition, etc. RNNs can be used to detect emotions within a block of text as well because sentiment analysis mainly converts the texts within a sentence fed into the model into vectors and further calculates the nature of the sentiments accordingly. So, for a certain type of sentiment, the embedded values of a sentence will be within a certain range. If we can determine that range the analysis process becomes a pattern recognition process.

Even though RNN has a memorization property it fails to perform when the sequence of data is largely due to the vanishing gradient problem where the iterations of the model don't work towards teaching the model how to generate accurate inputs.This occurs due to the weight calculation in the backpropagation process being too frequent and insignificant because the gradient calculated from the loss function is too small. As the structure of an RNN has only one nonlinear gate there is no workaround to this problem within the model itself. However, to solve this problem a more verbose and effective model has been developed.

Long-Short Term Memory or LSTM Is a variation of RNN where the structure includes three nonlinear gates: The input gate, forget gate, and output gate. These three gates are responsible for the data flow within an LSTM cell. An LSTM cell takes the hypothesis of the previous cell as input alongside the input for the current cell and a self-maintained cell state which indicates the nature of data inside an LSTM cell. The inputs are fed into all three gates where the input gate decides which portion of the data that is fed into the network should be propagated as an input into the cell, the forget get determines which part of the input to keep and which part to forget about and the output gate decides how much of the data that is kept inside the cell should be carried forward to the next LSTM cell as the hypothesis of this cell. This interconnected structure of LSTM makes the network immune to the vanishing gradient problem and helps it to keep a track of long sequences by holding on to necessary data for a long amount of time.

A previous study was done by H M Mahamudul et al[14] which demonstrated that RNN can be useful in Bengali speech recognition and can be further processed to analyze sentiments effectively. They have used Mel-Frequency Cepstrum Coefficient for the delineation of the spectral components of an audio block. Further down the line, they have used SVM and RNN to decide which of the core 6 sentiments are available within that block of audio. They have achieved up to 51.33 percent accuracy by using RNN categorization on top of their generic algorithmic approach.

A previously published journal[21] has explored a multitude of variations in LSTM to achieve a promising 95.30 percent accuracy with a stacked Bidirectional LSTM structure which uses a word embedding model that is based on inverse gravity moment and is term weighted. Bidirectional LSTM learns the context within a language by first forward propagating a sentence from start to finish and then back propagating a sentence from the end to the beginning which gives the model better understanding of the relationships between the words that are being used within the sentence.Stacked LSTM layers increase the accuracy of prediction by gathering better contextual elements taking into account past, present and future data. They have proposed a 3 layer stacked Bi-LSTM model which maximises the performance of LSTM networks using optimised word embedding techniques.

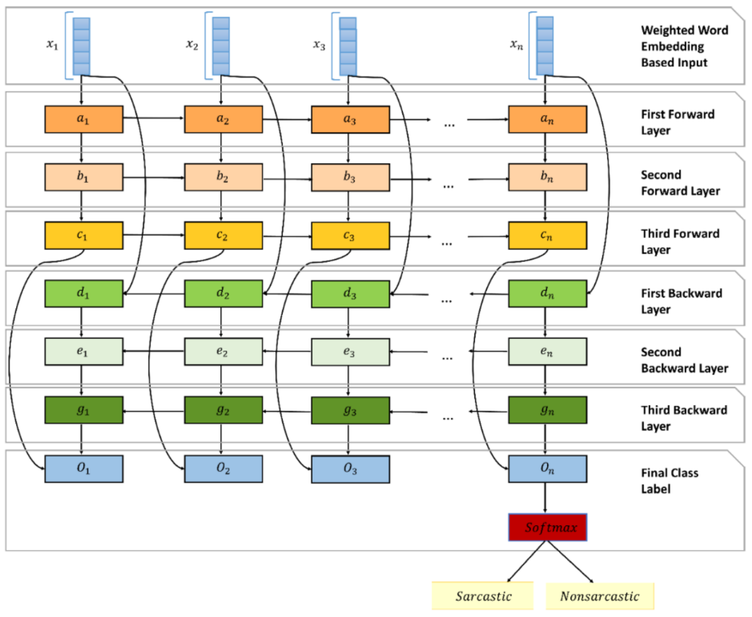


Figure 2.6: Stacked-3 Bi-LSTM diagram[21]

Form their study it has been made evident that the stacked LSTM-3 model is the most effective one among all other LSTM approaches and the inverse gravity moment based term weighting is the most effective method.

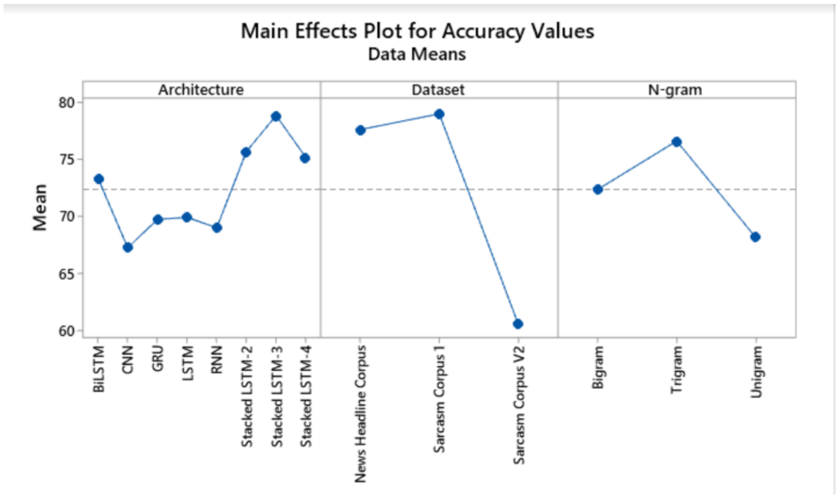


Figure 2.7: accuracy values comparison[21]

Other mentionable approaches incorporating LSTM in sarcasm detection that I have come across are: A research conducted by A. Kumar et al.[16] that proposes a multi head attention based bidirectional LSTM model that is said to outperform feature rich Support Vector Machine based approaches. The structure of the model includes a Bi-LSTM layer that encodes the embedded words that are generated from the input sentences, a multi head attention layer that tries to identify the parts that are responsible for the sarcastic nature of a comment the most instead of detecting only one phrase or word like other single headed LSTM models, a concentration stage where auxiliary features extracted from additional processes are combine with the embedded sentences and a softmax classifier. This approach without the auxiliary features attains the highest precision over SVM and generic Bi-LSTM with a score of 72.63 percent.

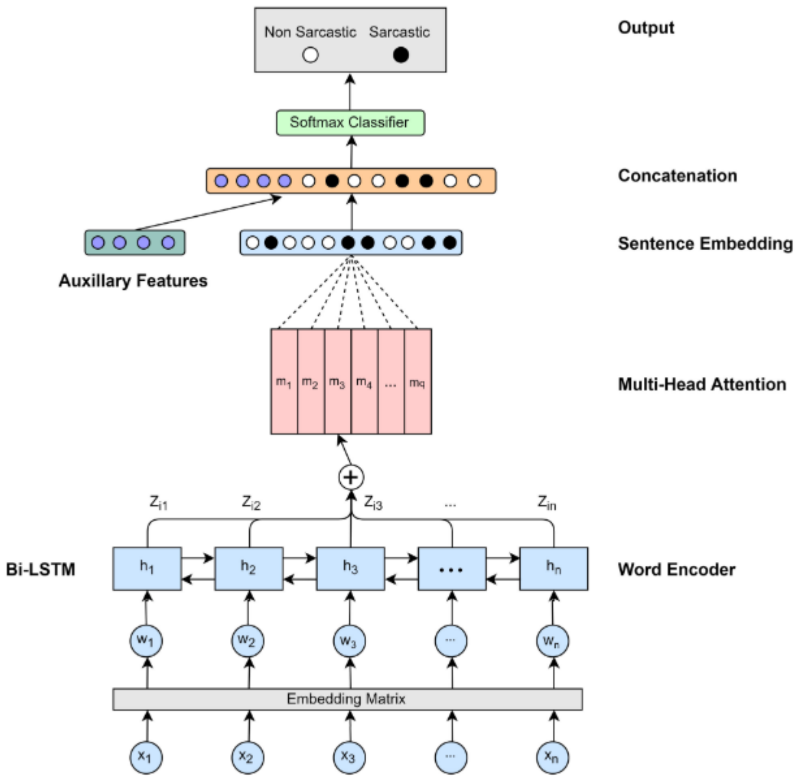


Figure 2.8: Multi Head Attention based BiLSTM structure[16]

A similar study shows that a soft attention layer [11] based model combined with convolutional neural networks can achieve respectable results while detecting sarcasm within a sentence.

Chapter 3

Methodology

In this section I aim to portray a rough outline of the process following which I will be detecting sarcasm on Bengali comments. The core technologies that are used to solve the issue in hand will be discussed here alongside the dataset description.

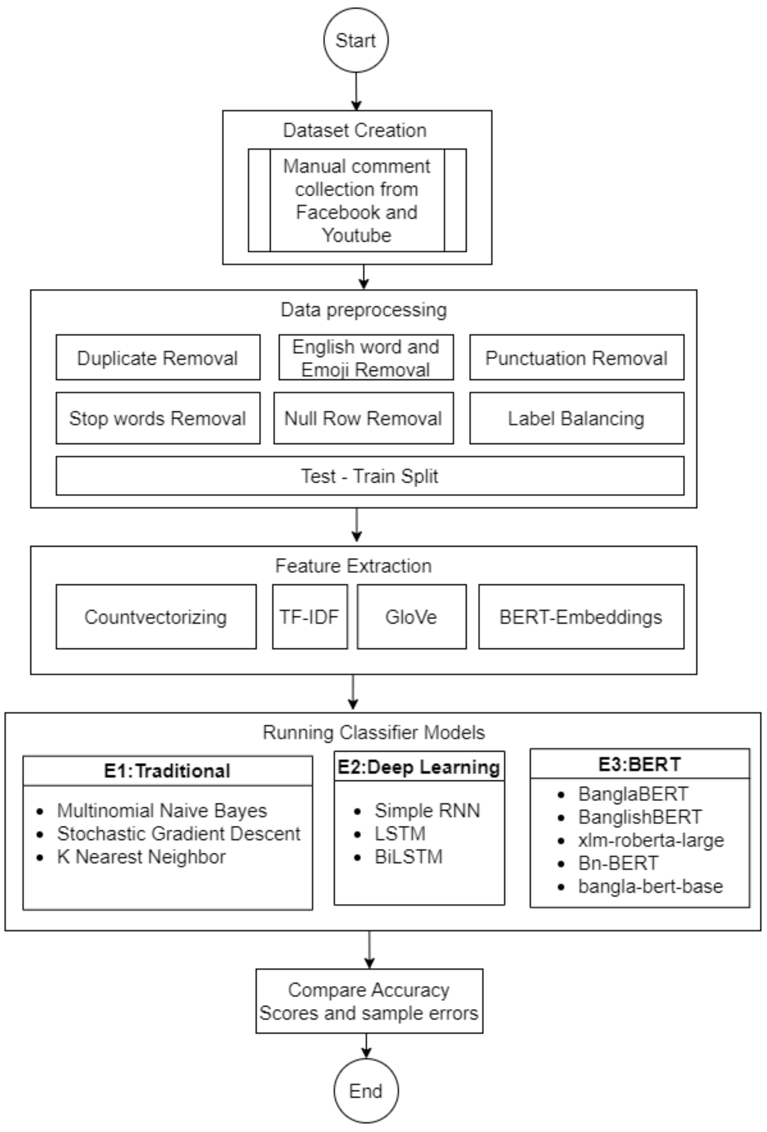
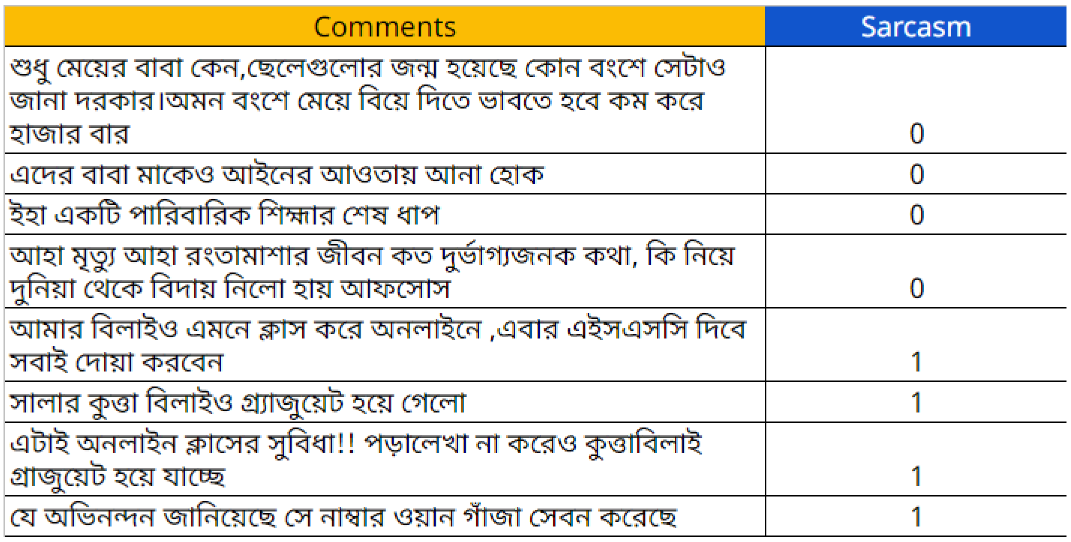


Figure 3.1: Methodology workflow diagram

3.1 Dataset Description

The dataset that has been used in this study is called ‘BN-SRCM’ which is a novel dataset developed by me. Due to the lack of open source datasets that would be fit for my task I have built this dataset for the sole purpose of sarcasm detection. The dataset contains 8048 unique entry points in the form of comments made in the Bengali language. The comments are collected from the most widely used social media platform of Bangladesh, Facebook. These comments are further associated with a polarity score that indicates the presence of sarcasm within that comment. Comments that are sarcastic have a polarity score of ‘1’ and comments that are not sarcastic have a polarity score of ‘0’ attached to them. The Binary encoded nature of the dataset makes the detection of sarcasm a bit easier.

The comments that are collected are mainly from famous Bengali social media news portals such as Prothom Alo ,Somoy News , BD News 24, Jamuna News etc. The post that are selected from the facebook pages of these portals are selected randomly on the basis of availability of a huge variety of Bengali comments attached to them. While collecting the comments I have tried to include sarcastic statements that are pretty straightforward to detect as sarcastic and rely less on context. I have also tried to include the variation of comments in a way so that from a single post equal amount of sarcastic and non sarcastic comments can be taken in. By maintaining almost a symmetric distribution of sarcastic and normal comments it can be made sure that the test and train splits of the dataset will have equal amounts of comments belonging to both polarities easily.



Comments	Sarcasm
শুধু মেয়ের বাবা কেন,ছেলেগুলোর জন্ম হয়েছে কোন বংশে সেটাও জানা দরকার।অমন বংশে মেয়ে বিয়ে দিতে ভাবতে হবে কম করে হাজার বার	0
এদের বাবা মাকেও আইনের আওতায় আনা হোক	0
ইহা একটি পারিবারিক শিক্ষার শেষ ধাপ	0
আহা মৃত্যু আহা রংতামাশার জীবন কত দুর্ভাগ্যজনক কথা, কি নিয়ে দুনিয়া থেকে বিদায় নিলো হায় আফসোস	0
আমার বিলাইও এমনে ক্লাস করে অনলাইনে ,এবার এইসএসসি দিবে সবাই দোয়া করবেন	1
সালার কুত্তা বিলাইও গ্র্যাজুয়েট হয়ে গেলো	1
এটাই অনলাইন ক্লাসের সুবিধা!! পড়ালেখা না করেও কুত্তাবিলাই গ্রাজুয়েট হয়ে যাচ্ছে	1
যে অভিনন্দন জানিয়েছে সে নাম্বার ওয়ান গাঁজা সেবন করেছে	1

Figure 3.2: BN-SRCM Dataset snapshot

3.2 Data Cleaning

The data cleaning phase handles the data that is in a form which cannot be dealt with. Here duplicate comments that are generic and always imply the same meaning whether they are sarcastic or not, are removed. Comments that have a broken sentence structure or do not make any sense at all are removed. Comments that have missing sections are removed and finally emojis within comment are also removed. Even though emojis depict a certain sort of sentiment and can be made use of while detecting sarcasm within a text, to reduce complexities of calculations I have chosen to remove them using regular expressions

3.3 Data Preprocessing

The data preprocessing phase contains multiple steps where data is formatted in a way so that the model can make the most sense of the data and efficiently detect sarcasm without any bias.

After collecting the comments and labeling them with a polarity score that indicates the sarcastic nature of a comment a few basic procedures are followed with the help of the open source Python library bnlp toolkit .

1. **Punctuation Removal:**As punctuations do not add any value to the sentiment analysis process all punctuations occurring within a simple sentence are removed using regular expressions.
2. **Stop word removal:** There are a list of words present in all languages that do not provide much value while being parsed through the sentiment analysis pipeline, these words are called stop words. In the preprocessing phase stop words are removed using the remove stopwords method int he bnlp toolkit library.
3. **Tokenization:** In the Tokenization process the words that are supposed to be fed into the model are converted into tokens or little lumps of information. While parsing a large sentence or sequence of sentences the individual meanings of words may be lost due to the computational complexity. Tokenization makes sure such an issue doesn't occur.
4. **Stemming:** Stemming is the process of reducing a word to its root. This is mainly done to reduce the complexity of a sentence and allocate a global score for a certain word which can be used to create a dictionary containing standalone scores for future reference. The words in the dataset are stemmed using the bangla stemmer library available on PyPI.
5. **Lemmatization:** Lemmatization process refers to keeping only the origin of a word if multiple forms of a single word are prevalent within a sentence

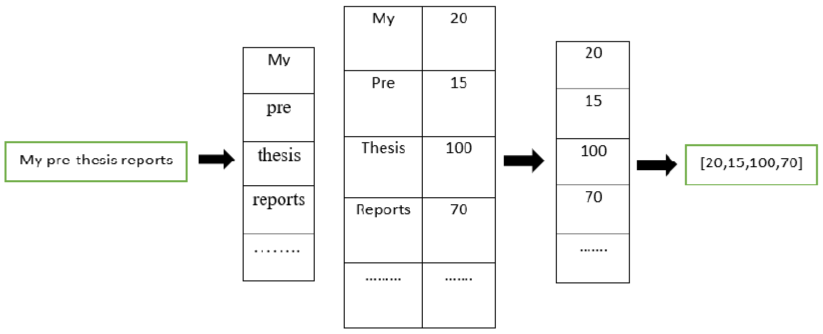


Figure 3.3: Tokenization process[17]

for example if a sentence contains the words ‘Doing’,’Done,’Did’ and ‘Do’ the process will reduce the words to the lemma or the base form of the word which is ‘Do’. As my dataset is small to retain semantic information I have avoided lemmatization.

3.4 Feature Extraction

As sarcasm detection is a text classification problem we will need some features in order to differentiate between the future words that will be fed into the model for generating predictions accurately

3.4.1 Vectorization

Countvectorizer: Count vectorizer is a method that converts text data into numerical data for it to be acceptable for machine learning models. It utilizes the bag of words approach to create a histogram based on the data it is passed into it. A histogram is a sort of an adjacency matrix where the column headers depict each unique word and the row headers depict different data points or comments in this case. The process begins with converting each sentence into tokens and mapping those tokens to the frequency of their occurrence in the document of comments. Based on these frequencies the method sorts the unique words in ascending order and creates the column headers. The comments are fitted onto the row headers and the cross section of each row and column indicates the magnitude of presence of the word of that column in the sentence of that row. 1/2/3 means the word is present once, twice/thrice and 0 means the word is absent in that sentence. For my dataset the size of the histogram is 8048 X 15552. This matrix or histogram can be further split into test and training sets to be fed into machine learning models.

3.4.2 Word Embedding

TF-IDF: Term Frequency - Inverse Document Frequency or TF-IDF is another way of converting text data into vectors. It is comparatively better than countvectorizer as it can hold more semantic information which can be used to make sense of text data more efficiently. Here the term frequency is calculated by the ratio of the

number of repetition of a word in a sentence and the total number of words in that sentence. On the other hand the inverse document frequency of a word is calculated by taking the logarithm of the ratio of the number of sentences in a corpus and the number of sentences that contain that word. Finally the TF-IDF value is calculated by multiplying the two previously calculated values.
This process basically generates separate values for each word within the corpus. If a word has a higher value than that word can be determined to be a distinguishable word that represents its labeled value. For my dataset if there are 8 words in a sentence and the label is 1 which means the sentence is sarcastic by calculating the values of each word through TF-IDF we can find out which words are more likely to depict sarcasm by finding out the words that have a higher value as output through the TF-IDF process. By this manner the TF-IDF vectorization process takes more information into account to generate values that retain better semantic information compared to countvectorizer.

GloVe: Global vectors for word representation or GloVe is one of the most widely used and efficient word embedding techniques in the Natural Language Processing domain.
We have seen that we need to convert words into numerical formats to pass them through machine learning models as they do not understand text data. However, deep learning models give us the added advantage of using multiple layers while calculating our predictions. Among these layers we have a specialized layer known as an embedding layer that helps the deep learning model to convert texts into vectors to generate embeddings for words or sentences.
The GloVe embedding process consists of generating one hot encoded vector of words in sentences and then taking them into consideration to generate a contextualized matrix that has representations of all words within that sentence. The process mainly generates a co-occurrence matrix of each word within a corpus separately in the context of other words. This is generated based on the the hypothesis that if the probability of occurrence of a word given that another word will co occur with it in the same sentence the probability of the output converging towards the trained label of the previous occurrences will increase. For my dataset I have used a pre-trained GloVe model called Bangla GloVe from the bnlp toolkit module to generate embeddings for the corpus based on my dataset. The one hot representations are passed into the embedding layer of the neural networks along with the vocabulary size, max length and padding sizes to generate embeddings for each word in sentences transformed into a vector size that is uniform throughout all data points of the dataset.

BERT-Embedding: Bert embedding is a word embedding method which is built into the BERT model. This method has a similar function compared to GloVe except that it works with sub segments of texts called word-pieces.In this study both of the embedding techniques are made use of for better vectorization of the scores for each word and better output generation.

3.5 Experiment- 1

In experiment 1 I have applied traditional machine learning algorithms on my dataset to detect sarcasm. From previous studies I have observed that for text classification tasks under the Natural Language Processing domain two of the best performing models are the Multinomial Naive Bayes and Stochastic Gradient Descent algorithm. I have used the extracted features from the TF-IDF vectorizer to calculate the predictions in the aforementioned models. Another credible model for tasks of this sort can be the K Nearest Neighbor algorithm which I have paired with the countvectorizer to calculate predictions.

3.5.1 Multinomial Naive Bayes

The Naive Bayes classifier basically uses the bayes theorem to classify features into labels.In text classification problems it implements a probabilistic method of calculating the likelihood of a piece of text belonging to a certain category depicted by a certain label. The model takes a lot of things into consideration while predicting the labels for vectorized sentences that are passed into it. To put things into perspective of my task, to classify whether a sentence is sarcastic or not we mainly have to calculate using the Bayes theorem that what is the probability of the sentence being Sarcastic/ Not sarcastic given that the featured are x1,x2,x3 and so on. Here the values of the features will be the vectors generated by the TF-IDF vectorizer for each word in the sentence that has been passed for classification. The vectors will contain the information depicting which words are more important while representing the label that it is associated with.
To calculate this we will need the global probability of sentences being Sarcastic/ Not Sarcastic and the conditional probability of a feature appearing given that the label is Sarcastic - 1 or Not Sarcastic - 0. These probabilities will be then multiplied and normalized to a value with a max range of 1 by dividing the product by the total probability of the features appearing in the dataset as per the bayes theorem. Even though it is a lightweight and simple classification model perfect for baseline evaluations in text classification tasks the Naive Bayes classifier has a downside called the zero frequency problem. Here the model assigns a 0 value to words that have not appeared in the train set of the dataset that has been passed into the model.Which makes evaluations for new data unreliable and faulty.

$$P(class/features) = \frac{P(class) * P(features/class)}{P(features)}$$

- P(class/features) : Posterior Probability
- P(class) : Class Prior Probability
- P(features/class) : Likelihood
- P(features) : Predictor Prior Probability

3.5.2 Stochastic Gradient Descent

The Stochastic Gradient Descent classifier utilizes a linear transformation equation consisting of the input data in forms of vectorized text which are labeled in numerical format, a learning rate the slope of the equation curve, the iterative prediction values alongside a loss value which can be calculated in many different ways. The model mainly takes a random guess about the possible label for the text data that has been passed into the model and further tries to converge towards an optimal value to finally determine the actual label for the text by minimizing the loss iteratively.
Here the model calculates the linear transformation function and plots the points of the range in a plane which allocates the graph of the function. After the random estimation value is taken the iterations of the calculation guides the position of the value towards the point with the lowest level of error. Each iteration takes normalization of outputs and penalization for errors into account to calculate the latest values.

3.5.3 K nearest Neighbor

K nearest Neighbor or KNN is another popular model for text classification tasks which basically predicts the label of a training data based on the K closest data points with the passed data found in the training dataset.
In the perspective of my task once the data is vectorized and split into the test and training set the classification task can be initialized. Here when a new piece of text data is passed into the model from the test set the model calculates the similarity between the passed test data and the stored training data points and calculates K number of datapoints that have the most similarity with the passed data by calculating similarity scores. Then the model utilizes these data points to predict whether the data point that has to be classified is Sarcastic or Not. For example if we take into consideration 10 nearest comments in the perspective of a random comment and 7 of them are Sarcastic, chances are the input text which is similar to these data points will be Sarcastic as well so it will predict the label of that text as Sarcastic.
The accuracy of the KNN algorithm is based on the value of K . If the value is low than the predictions may be incorrect as the increasing amount of data points with different biases will not be taken into account and a possibility of getting a skewed answer will increase on the other hand if we take the K value too high than the computation will take larger amount of time and hence the model will be slow.

3.5.4 Evaluation

From the table we can see that the multinomial naive bayes algorithm is the best performing algorithm and the KNN algorithm is the worst performing one.

Model	F1- Score	Precision	Accuracy
Multinomial Naive Bayes:	**64.41%**	**64.52%**	**64.48%**
Stochastic Gradient Descent:	61.86%	63.15%	62.43%
K Nearest Neighbor:	56.70%	58.51%	54.60%

Table 3.1: Traditional models Performance measure

3.6 Experiment- 2

In experiment 2 I have applied deep learning models to solve the problem that I am working with.In the traditional machine learning model when the data points are converted into numerical format using the countvectorizer or TF IDF. vectorizer and passed into the models the sequential information is lost. This mainly means that the machine learning models do not keep track of the information of which sentence comes after which and what was the output of the evaluation of the previous datapoint. Because of this a good amount of semantic information is lost and this can affect the outcome of the evaluations drastically on a large scale. The recurrent neural network based deep learning models address these problems by taking the hypothesis generated from the previous datapoint evaluation into account. There are three different deep learning models that I have implemented for my task. For all of the models I have made use of the embedding layer available in Keras. Before that we need to create a uniform input which is done by firstly embedding the text inputs using the previously discussed GloVe embedding method. Here I have used the bnlp toolkit to access the Bangla GloVe module which is a pre-trained GloVe model. This model has two versions one of which is trained on Bengali news datasets and the other one is trained on Bengali wiki text data. I have used the Bengali news dataset version as it converges with the type of data in my dataset. Further the embeddings are put into an uniform length as the Keras embedding layer accepts input of uniform length. This is done by padding the embeddings generated from GloVe.
I have set the maximum length of the input comments as the base length and padded any other comment with lesser length with zeros at the end of the list.

3.6.1 Simple-RNN

In all RNN models there are three stages of evaluating an input to generate output.The first stage involves the input layer which keeps all of the inputs or the data points from the test set moving forward to the hidden layer where the necessary calculations are made to generate the predictions for the inputs. On the second stage the predicted values are compared with the actual values that were originally associated with the text at the output layer. This process incorporates the loss function. The loss function is a performance measure of the deep learning models of the loss- function generates a low value, the model is working well. If the loss function generates a high value the model is not performing well.
The final stage is the calculation of gradient for each datapoint and updating the weights of nodes using the calculated gradients. The first two stages are a part of

forward propagation and the last stage is a part of backwards propagation.
The simple RNN consists of only one tan hyperbolic layer which normalizes the input value within a range. Even though the RNN model is a good approach to solve sequential text classification problems it suffers from a problem called the vanishing gradient problem where due to very small adjustments in the weights for the nodes the gradient generated becomes smaller and smaller. After a point the gradient becomes so small that it doesn't impact the weight adjustments at all and the learning rate drops to almost zero. The model is rendered functionless at this point. This problem is addressed by the LSTM model

3.6.2 LSTM

The Long Short Term memory or LSTM model has a similar structure and working procedure to the simple RNN model. Only the hidden layer calculations are more efficient and verbose in this model. here the hidden layer consists of three gates: the forget gate which is responsible for which part of the data to forget that is passed into the cell. This gate consists of a sigmoid activation function, an input gate which takes in the inputs and consists of a sigmoid function and a tan hyperbolic function that rationalizes the data within a range. Finally the output gate that calculates the output depending on the previous cell hypothesis and outputs from the other gate. Since the vanishing gradient problem occurs in the back propagation part. The LSTM architecture prevents the values of the gradient to go below a certain threshold. This eliminates the vanishing gradient problem.
As the gates work simultaneously to generate the prediction there is more precision and control available in the LSTM model compared to the RNN models. This makes it more versatile and efficient.

The formula for the gates inside an LSTM cell:

$$\begin{gathered}\text{Input gate: } i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \\ \text{Forget gate: } f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ \text{Output gate: } o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \\ \text{Cell state: } c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c)\end{gathered}$$

In the picture the three inputs coming into the LSTM cell are the cell state, the hypothesis from the previous cell and the input data itself. And the output are the cell state for the next cell connected to this cell, hypothesis of this cell and the hidden state output for the next cell

3.6.3 biLSTM

The BiLSTM model is exactly the same as the LSTM model except that. The Bidirectional LSTM model generates context for the text data from both sides. It runs the text through the model in the correct order through forward propagation and in the backwards order in the backwards propagation. This generates better context and better understanding of the meaning or the passed text within the model

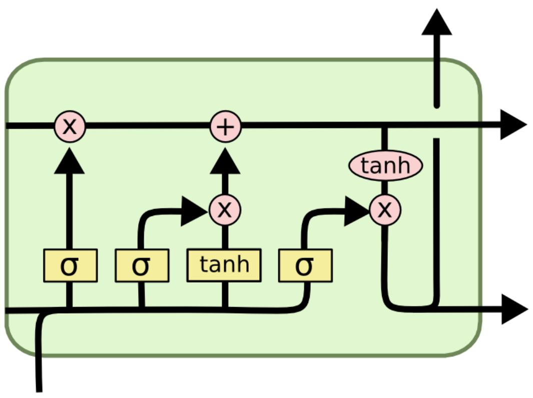


Figure 3.4: LSTM Structure

compared to the single directional LSTM model. It also parses data faster as the propagations take place simultaneously.

3.6.4 Evaluation

From the table we can see that the best performing model is the Bidirectional LSTM model with GloVe embeddings. This comes obvious as it is a faster and more efficient model with a better working procedure compared to the other two models.

Model	F1- Score	Precision	Accuracy
Simple RNN	51.98%	52.51%	52.30%
LSTM	54.01%	53.93%	54.03%
Bi-LSTM	**57.72%**	**57.64%**	**57.71%**

Table 3.2: RNN Performance measure

3.7 Experiment- 3

In experiment 3 I have implemented five different pretrained transformer based BERT models specialized for working with Bengali data. The difference between these models are the bases which are different variants of BERT with slightly different architectures. This also brings in different tokenizers and encoding processes for the text data that is passed into the models.Another difference is the training dataset that these models are trained on.
The implementations of these models are done with the help of either Tensorflow and keras or the Simpletransformers library and Pytorch.

3.7.1 Transformers

Sequence modelling tasks are tasks where long sequences of data is handled with for example : image recognition, voice recognition , pattern recognition are prominent problems of the sequence modelling domain. A state of the art approach for solving these problems is RNNs where the circular input-output structure of the cells are used to keep track of memory. These models are known as sequence to sequence models which are perfect for tasks such as translating a block of text from one language to another. Despite the advanced functionalities offered by these models there are some limitations with them. One of them being the previously mentioned vanishing gradient problem, for which the model cannot work with sequences of data that are too long. Another problem is the inability of parallelization due to the sequential nature of the models.[2] To address these issues the Transformers model was invented at Google Brain back in 2017. This model introduces a Encoder - Decoder structure which works parallel to each other in order to make sense of the sequence of data that is being fed into the transformer model.

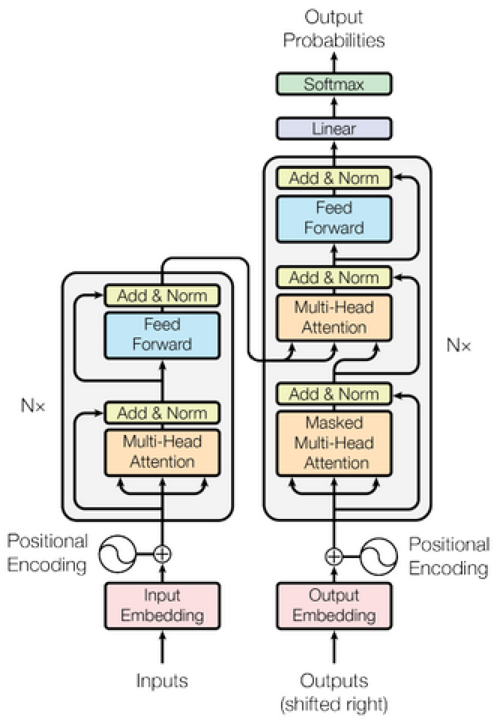


Figure 3.5: Transformer Structure[10]

Here the little block on the left is the encoder and the long block in the right is the decoder both of which have multi head attention layers and feed forward network layer attached to an addition and normalization function. The encoder takes in the input information as embedded words the inputs then go through the stacked encoder cells in the encoder block and after the data passes through the last encoder it is propagated to all of the decoder cells in the decoder block together which further calculates the output of the operation. Here a self attention layer is present considers one word within a sentence and calculates its co-relation with all of the other words withing that sentence to get a better understanding of the context within that sentence. This self attention is calculated through three different vectors called the Query Vector, Key Vector and Value Vector. The multi head attention layer enables the transformer model to calculate multiple representations of a single word in multiple positions of the corpus concurrently.

3.7.2 BERT

Bidirectional encoder representation for Transformers of BERT is transformer based model developed at Google AI with the intention of language modelling. The core functionality of BERT is making the training of transformers technically bidirectional. Due to the increased iteration process bidirectional models can grasp the context of a block of text better than that of a unidirectional model. BERT includes a novel method called Masked Language modelling which enable the bidirectional training of the transformers in the model.

We know that there are two parts within a transformer one is the encoder and the other one is the decoder. Here the encoder part deals with input data to make sense of the words that makeup the input sentences and the decoder works with output data which is basically the prediction task. BERT is a model built with the intention of language modelling only therefore here the decoder part is not need and we only have to work with the encoder part.The BERT model takes in a sequence of tokens as an input and faces the training challenge of having a defined prediction goal from the beginning of the calculation. This challenge is dealt with using two training strategies within BERT:

1. **Masked Language Model(MLM):** The MLM part of the BERT model mainly converts 15 percent of the words in the sequences that are being fed into the model into into masked tokens after that the model tries to predict the words that are masked from the context that it gathers from the sentences around it. This require the addition of a classification layer over the encoder output in the blocks of encoder, transforming the output vectors into the dimension of the vocabulary present in the context of the sentences and implementing a softmax classifier to calculate the probability of each occurring word.Here the loss function of the BERT model only takes into consideration the masked words in a sentence the unmasked words are ignore for this matter.

2. **Next Sentence Prediction(NSP):** The NSP part of the model deals with the co relation between a pair of sentences that are fed as an input into the model. Here the model takes in pairs of sentences as input and tries to predict whether the second sentence is consequent to the other.in the process half of the sentences are original pairs of sentences prevalent in the actual block of text and the other half is randomly selected sentences from the corpus. The

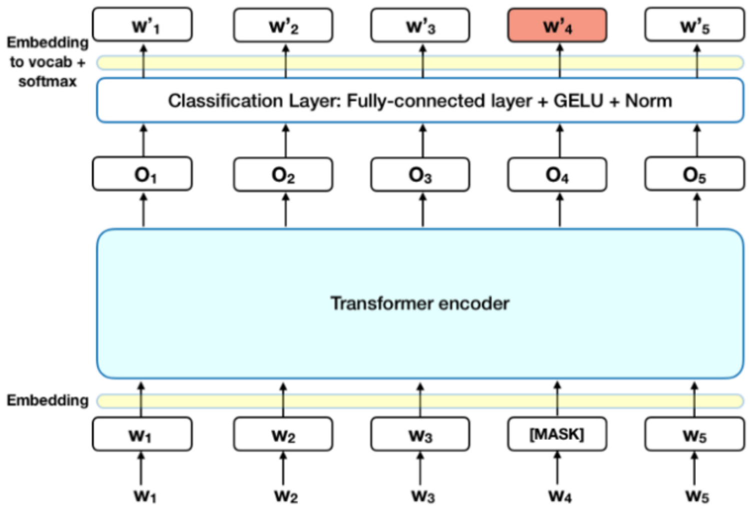

W'1
W'2
W'3
W'4
W'5
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
W1
W2
W3
[MASK]
W5
W1
W2
W3
W4
W5

Figure 3.6: Masked Language Model Structure [3]

two halves are further compared with the assumption that they are not connected to each other. Here a CLS token is added to the starting of the first sentence and a SEP token is added to the ending of the second sentence. An embedding articulating the position of the sentence is then added to in order to keep a track of the sequence

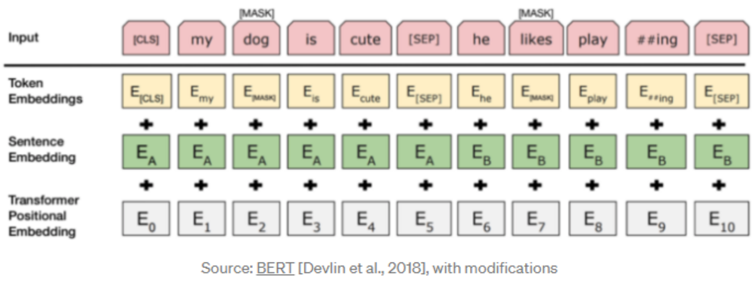


Figure 3.7: Next Sentence Prediction Structure [3]

The sarcasm detection model is a classification task where the inputs are taken in to predict whether the sentences are sarcastic or not. For this task the BERT model can be fine tuned according to the Next sentence prediction bit of the model by adding a classification layer on top of the encoder output.

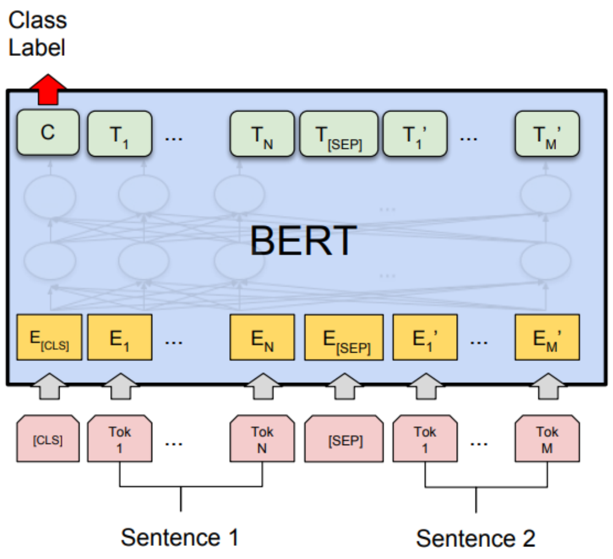


Figure 3.8: General structure of BERT [3]

3.7.3 csebuetnlp/banglabert

The banglabert model by csebuetnlp is the most popular transformer based model on the Huggingface website specialized for working with Bengali data points it based on the widely used model Electra which, like the other BERT based models has a generator model which is used to replace tokens within a datapoint. Also it trains another model called a discriminator model which can be used to find out which of the tokens have been replaced by the generator within a sentence. This makes the the model perfect for text my text classification task as the model is trained with special normalization pipeline developed by the csebuetnlp trained on Bengali data collected from the Bengali part of the OSCAR dataset available of Huggingface.[24]

3.7.4 csebuetnlp/banglishbert

Like the banglabert model the Banglish bert model is developed around Electra as well. The only difference is that banglishbert is trained on banglish data which basically means the mixture of Bengali and English data. This model is implemented with my dataset with a different pre-processing method where only one step has been altered. The step where I removed the Banglish components from the dataset. As this model enables us to work with banglish words I have kept the likely banglish words within the dataset. Both of the Electra based models uses a tokenizer called ElectraTokenizer which is used to automatically encode text data into numerical format that is acceptable to the model.[25]

3.7.5 neuralspace-reverie/indic-transformers-bn-bert

indic -transformers-bn-bert is a pre-trained BERT based model which is trained on 3 gigabytes of bengali data taken from the OSCAR dataset that is available on the Hugging Face website. It is finetuned for downstream tasks such as text classification and parts of speech tagging. Which makes it even more credible for our task . it uses the BertTokenizer to encode data that is passed into it.[15]

3.7.6 Xlm-roberta-large

Xlm-roberta-large is the multilingual version of the popular RoBERTa model which itself is a modified version of the original BERT model. This model has a few changes in the hyperparameters and the embedding technique. The main advantage of using this model is the amount of data it is trained with is 2.5 TB of common crawl data which are filtered and represented in 100 languages including Bengali. This gives the model a huge flexibility advantage over models with multilingual or low resource monolingual datasets from the huge training corpus. It uses the RobertaTokenizer to encode the datapoints.[8]

3.7.7 sagorsarker/bangla-bert-base

Bangla-bert-base is another pretrained BERT based model used to execute tasks on Bengali language centered datasets. It is trained with the OSCAR datasets bengali part and the Bengali Wiki dump dataset. It also uses the BertTokenizer to encode the text data before evaluating them and predicting the labels.[18]

3.7.8 Evaluation

From the table we can see that the best performing model is the BanglaBERT model from buetcsenlp. The second best model is the bn bert model by indic transformers. The xlm roberta multilingual model also performs well in the dataset.

Model	F1-Score	Precision	Accuracy
csebuetnlp/banglabert:	**74.53%**	**74.48%**	**74.52%**
csebuetnlp/banglishbert:	72.61%	72.57%	72.63%
neuralspace-reverie/indic-transformers-bn-bert:	71.89%	71.86%	71.88%
Xlm-roberta-large	70.51%	70.55%	70.54%
sagorsarker/bangla-bert-base:	69.78%	69.75%	69.77%

Table 3.3: BERT performance measure

Chapter 4

Scope and Analysis

4.1 Comparison and Observation

From comparing all of the models and accuracy scores we can see that the bert models perform better in sarcasm detection compared to the other models. The traditional models paired with the countvectorizer or TF-IDF vectorizer perform close to the bert models and the RNN based deep learning models perform the worst. This is partially due to the embedding technique that is used with the RNN based models. The pretrained GloVe model generates embedding for the texts in the dataset based on previously trained sentences used to generate the vector values for each word. These words are further matched with the words in my dataset. Therefore the words may match but the spatial meaning and the contextual semantic values may not be accurate as the reference sentences are different. Also for words that are not available in the corpus that was used to train the GloVe model, the model cannot generate numerical values.

4.2 Conclusion and Future Work

From the study we can conclude that the best set of models for this task would be models that have a better chance of being contextualized for the classification class in the Bengali Language domain. Because of its spatial meaning awareness it is able to understand the text data better than the other models.
In the future I want to broaden the dataset in volume and follow better labeling methods with multiple evaluators which would ensure the semantic structure of a category of labeling to be uniform and coherent. I would also like to generate better embeddings for the dataset using the Continuous Bag of Words based vectorising

Model	F1- Score	Precision	Accuracy
csebuetnlp/banglabert	**74.53%**	**74.48%**	**74.52%**
Multinomial Naive Bayes	64.41%	64.52%	64.48%
Bi-LSTM	57.72%	57.64%	57.71%

Table 4.1: Comparison of models

model Word2Vec and train the model based on a corpus that will contextualize the model and prepare it for my dataset. This would ensure better features and better semantic meaning retention while passing the data into the models. Training the word embedding models on custom data will eliminate the possibilities of word embeddings not being found. This would significantly improve the accuracy of the RNN based models.

Bibliography

[1] M. Bouazizi and T. Otsuki Ohtsuki, "A pattern-based approach for sarcasm detection on twitter," *IEEE Access*, vol. 4, pp. 5477–5488, 2016. DOI: 10.1109/ACCESS.2016.2594194.

[2] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/pdf/1706.03762.pdf.

[3] R. Horev, "Bert explained: State of the art language model for nlp," in Nov. 2018, pp. 2–5. [Online]. Available: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

[4] S. Lee, "Figurative language in emotion expressions," in Jan. 2018, pp. 408–419, ISBN: 978-3-319-73572-6. DOI: 10.1007/978-3-319-73573-3_37.

[5] S. Rendalkar and C. Chandankhede, "Sarcasm detection of online comments using emotion detection," in *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2018, pp. 1244–1249. DOI: 10.1109/ICIRCA.2018.8597368.

[6] K. Sreelakshmi and P. C. Rafeeque, "An effective approach for detection of sarcasm in tweets," in *2018 International CET Conference on Control, Communication, and Computing (IC4)*, 2018, pp. 377–382. DOI: 10.1109/CETIC4.2018.8531044.

[7] J. H. Balanke and H. V., "Extension of the lexicon algorithm for sarcasm detection," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, 2019, pp. 1063–1068. DOI: 10.1109/ICCMC.2019.8819845.

[8] A. Conneau, K. Khandelwal, N. Goyal, *et al.*, "Unsupervised cross-lingual representation learning at scale," *CoRR*, vol. abs/1911.02116, 2019. arXiv: 1911.02116. [Online]. Available: http://arxiv.org/abs/1911.02116.

[9] D. Das, "A multimodal approach to sarcasm detection on social media," Ph.D. dissertation, Aug. 2019.

[10] P. Joshi, "How do transformers work in nlp? a guide to the latest state-of-the-art models," in Jun. 2019, pp. 1–6. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models.

[11] L. H. Son, A. Kumar, S. R. Sangwan, A. Arora, A. Nayyar, and M. Abdel-Basset, "Sarcasm detection using soft attention-based bidirectional long short-term memory model with convolution network," *IEEE Access*, vol. 7, pp. 23 319–23 328, 2019. DOI: 10.1109/ACCESS.2019.2899260.

[12] M. Abulaish, A. Kamal, and M. Zaki, "A survey of figurative language and its computational detection in online social networks," *ACM Transactions on the Web*, vol. 14, pp. 1–52, Jan. 2020. DOI: 10.1145/3375547.

[13] D. Banerjee, "Natural language processing (nlp) simplified : A step-by-step guide," in Apr. 2020, pp. 1–6. [Online]. Available: https://datascience.foundation/sciencewhitepaper/natural-language-processing-nlp-simplified-a-step-by-step-guide.

[14] H. M. M. Hasan and M. A. Islam, "Emotion recognition from bengali speech using rnn modulation-based categorization," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 1131–1136. DOI: 10.1109/ICSSIT48917.2020.9214196.

[15] K. Jain, A. Deshpande, K. Shridhar, F. Laumann, and A. Dash, "Indic-transformers: An analysis of transformer language models for indian languages," in Nov. 2020.

[16] A. Kumar, V. T. Narapareddy, V. Aditya Srikanth, A. Malapati, and L. B. M. Neti, "Sarcasm detection using multi-head attention based bidirectional lstm," *IEEE Access*, vol. 8, pp. 6388–6397, 2020. DOI: 10.1109/ACCESS.2019.2963630.

[17] A. Pai, "What is tokenization in nlp? here's all you need to know," in May 2020, pp. 1–6. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/.

[18] S. Sarker, "Banglabert: Bengali mask language model for bengali language understading," in 2020. [Online]. Available: https://github.com/sagorbrur/bangla-bert.

[19] M. Bedi, S. Kumar, M. Akhtar, and T. Chakraborty, "Multi-modal sarcasm detection and humor classification in code-mixed conversations," *IEEE Transactions on Affective Computing*, vol. PP, pp. 1–1, May 2021. DOI: 10.1109/TAFFC.2021.3083522.

[20] S. Janjua, G. Farooq, M. Sindhu, and U. Rashid, "Multi-level aspect based sentiment classification of twitter data: Using hybrid approach in deep learning," *PeerJ Computer Science*, vol. 7, Apr. 2021. DOI: 10.7717/peerj-cs.433.

[21] A. Onan and M. A. Toçoğlu, "A term weighted neural language model and stacked bidirectional lstm based framework for sarcasm identification," *IEEE Access*, vol. 9, pp. 7701–7722, 2021. DOI: 10.1109/ACCESS.2021.3049734.

[22] M. S. Razali, A. A. Halin, L. Ye, S. Doraisamy, and N. M. Norowi, "Sarcasm detection using deep learning with contextual features," *IEEE Access*, vol. 9, pp. 68 609–68 618, 2021. DOI: 10.1109/ACCESS.2021.3076789.

[23] P. Verma, N. Shukla, and A. Shukla, "Techniques of sarcasm detection: A review," in *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2021, pp. 968–972. DOI: 10.1109/ICACITE51222.2021.9404585.

[24] A. Bhattacharjee, T. Hasan, W. Ahmad, *et al.*, “BanglaBERT: Language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla,” in *Findings of the Association for Computational Linguistics: NAACL 2022*, Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 1318–1327. [Online]. Available: https://aclanthology.org/2022.findings-naacl.98.

[25] A. Bhattacharjee, T. Hasan, K. Mubasshir, *et al.*, “Banglabert: Lagnuage model pretraining and benchmarks for low-resource language understanding evaluation in bangla,” in *Findings of the North American Chapter of the Association for Computational Linguistics: NAACL 2022*, 2022. arXiv: 2101.00204. [Online]. Available: https://arxiv.org/abs/2101.00204.

[26] M. Bouazizi and T. Ohtsuki, “Sarcasm over time and across platforms: Does the way we express sarcasm change?” *IEEE Access*, vol. 10, pp. 55 958–55 987, 2022. DOI: 10.1109/ACCESS.2022.3174862.

[27] M. S. Razali, A. Abdul Halin, Y.-W. Chow, N. Mohd Norowi, and S. Doraisamy, “Context-driven satire detection with deep learning,” *IEEE Access*, vol. 10, pp. 78 780–78 787, 2022. DOI: 10.1109/ACCESS.2022.3194119.