A Comprehensive Study of Sarcasm Detection in Bangla Using NLP

Maliha Mussarrat
Computer Science and Engineering
BRAC University
maliha.mussarat@g.bracu.ac.bd

Abrar Qasem
Computer Science and Engineering
BRAC University
abrar.qasem@g.bracu.ac.bd

Hasan Mohammod Tanzir
Computer Science and Engineering
BRAC University
hasan.mohammod.tanzir@g.bracu.ac.bd

Md. Abu Bakar Siddique Khan
Computer Science and Engineering
BRAC University
abu.bakar.siddique.khan@g.bracu.ac.bd

Md Sabbir Hossain
Computer Science and Engineering
BRAC University
md.sabbir.hossain1@g.bracu.ac.bd

Annajiat Alim Rasel
Computer Science and Engineering
BRAC University
annajiat@gmail.com

***Abstract*—Sarcasm serves as a unique approach to express emotions, when words convey meanings opposed to the original intended, typically mixed with humor. Unraveling sarcasm is tough due to its variable textual clues. In the Bangla language, this element remains mostly unexplored. To solve this gap, we utilised cutting-edge NLP approaches, Machine Learning, and Deep Learning on a dataset that already existed. Our study proposes a Humour Recognition AI for the language of Bangla built upon a Deep Learning Architecture. Achieving an outstanding 95% F1 score and accuracy, our model beats existing machine learning methods. This research contributes to Bangla NLP and breakthroughs in sarcasm detection.**

***Index Terms*—NLP, AI, Natural Language Processing in Bangla**

I. INTRODUCTION

Amidst the colorful landscape of social media, the utilization of sarcasm is gradually growing, engaging an increasingly expansive group of individuals. Verbal irony, in which words are purposefully used to convey meanings that are at odds with their literal meaning, is a common form of sarcasm. This kind of expression has its origins in the playful mockery of others, whether in jest or with sincere meaning, adding a dash of humor or even contempt. Sarcasm has firmly knitted itself into the fabric of communication across the worldwide social media landscape.

Since sarcasm is frequently used negatively, it must be prohibited throughout all platforms for social networking and in real-world situations such as conversations over the internet. There is more comprehensive data in online conversations and feedback, which is the reason why it recently gained a lot of popularity.

NLP Research Communities continue to classify Bangla as a LResource Language. For sarcasm detection tasks, there is only one publicly available dataset that we could find. They described the technical challenges they encountered in that paper [1] as a result of the lack of resources needed to produce a sizable dataset. To assess the performance, they simply utilized conventional machine learning models.

In our study, we examined a number of well-liked NLP preprocessing methods. To track the impact of our preprocessing step, we first employed conventional classification models like the SVM, Random Forest model, and Logistic Regression. Then, using a deep learning architecture, we constructed a model employing an ensemble of GRU and CNN. We divided the dataset 80:20. Finally, we tested our model using the test dataset and achieved an accuracy and F1 score of 95% and 95%, respectively.

We have briefly covered the earlier works in part II and will do so again in the next sections. After that, in section III-A, we talked about the dataset that we utilized to train our model. In section III, we will talk more about how the models were trained. In this section, we review the findings that were derived from the instruction of the models. In conclusion, Section V encapsulated the entirety of our discourse and encompassed our prospective undertakings.

II. LITERATURE REVIEW

There is significant interest in creating and utilizing Bangla datasets for sentiment analysis and similar tasks, according to the literature on the subject. For instance, the BanglaSenti dataset, developed by Ali et al. (mentioned in 1), has over 61,000 Bangla words that have been classified as positive, negative, or neutral. Similar to this, Rahman et al. (also referenced in 1) produced two datasets with comments from restaurant and cricket reviews that were used for aspect-based sentiment analysis. Modern machine learning techniques and statistical linguistic analysis were used to analyze these datasets, yielding results with a fair level of accuracy. Our knowledge of sentiment analysis in Bangla still has some gaps, particularly in the area of sarcasm identification. Although this work was not specifically for Bangla, Suhaimin presented a modified method for categorising and detecting sarcasm in analysis of sentiment. Furthermore, it is challenging to create

Identify applicable funding agency here. If none, delete this.

and assess automated systems for this activity because there aren't any annotated datasets available for Bangla.

The writers discuss the characteristics of sarcasm and the difficulties in recognizing it, including the use of irony, exaggeration, and understatement. Additionally, they go over various strategies for detecting sarcasm, such as rule-based, machine-learning, and hybrid approaches. The authors examine a number of research that have employed supervised, unsupervised, and semi-supervised learning techniques to identify sarcasm. Additionally, they go over the different features that were employed in this research, including lexical, syntactic, and semantic aspects. Overall, the review sheds essential light on the state of the art in sarcasm detection research and emphasizes the need for more study in this field.

Natural language analysis and sentiment mining experts can benefit greatly from this review's findings. It is clear from a survey of research that mockery extraction in NLP is a difficult problem due to the many factors involved. Nonetheless, it's important to remember that algorithmic learning methods have shown promise in solving this problem. More work is needed to improve the precision of sarcasm detectors and to create models that can efficiently deal with a wide variety of sarcastic expressions. The writers analyze the characteristics of mockery and the difficulties in recognizing it, including the utilization of humor, exaggeration, and ambiguity. The researchers also go over other methods used to spot humor, such as governed by rules, machine learning, and mixed approaches. Experts have used monitored, unattended, and semi-supervised training strategies to detect mockery. Some of this study is dissected by the writers of it. The researchers elaborated on the research's methodology, which drew on a wide range of linguistic, syntactic, and cognitive aspects. Overall, the summary sheds light on where the field of mockery identification is at the moment and why more study is needed in this area. Scientists and users in the areas of conversational text analysis and sentiment extraction can benefit greatly from the information presented in this summary. While artificial intelligence has shown promise in tackling this problem, an analysis of the current literature suggests that mockery detection remains a significant obstacle in the field of artificial text analysis.

In order to recognize mockery in words, researchers of the study "Harnessing Cognitive Features for Sarcasm Detection" provide a unique method based on the analysis of users' eye-movement tendencies. The authors contend that conventional language and stylistic traits alone frequently fall short of capturing the cognitive processes needed in sarcasm comprehension, making them insufficient for sarcasm detection. The authors increased sarcasm recognition by 3.7% by combining these data with cognitive variables like fixation length and saccade amplitude. The authors contend that current methods for sarcasm identification, which rely on language and stylistic characteristics, fall short in their ability to model the mental operations necessary for sarcasm interpretation. The authors propose an innovative approach to identify sarcasm by using the eye-movement patterns exhibited by human readers, so addressing this limitation. They contend that eye-movement patterns, which represent the reader's attentional focus and processing techniques, are a rich source of information on the cognitive processes involved in understanding sarcasm. The fixation time, saccade amplitude, and regression path length are just a few of the cognitive variables that the scientists extract from eye movement patterns and employ to train a machine-learning model for sarcasm detection.

A dataset of 821 sentences, including 171 sarcastic tweets and 650 non-sarcastic statements, is used by the authors to gauge the efficacy of their approach. They contrast the potency of their approaches with a number of baseline models that only take into account linguistic and stylistic characteristics. Their method outperforms the starting point estimates, as evidenced by the data, which shows a 75.3 percent accuracy and an F1 score of around 0.74. A thorough examination of the cognitive aspects is also done by the authors, who find that sarcastic words are linked to longer fixation times and smaller saccade amplitudes than non-sarcastic ones. In conclusion, the paper significantly advances the field of sarcasm detection by putting forth a fresh strategy that makes use of cognitive elements from eye movement patterns.

Reactive supervision is a cutting-edge technique that takes advantage of the organic dynamics of online dialogues to gather data about sarcasm. The ability to swiftly and effectively gather vast amounts of data as well as the ability to record the distinctive contextual characteristics of sarcastic statements are just two of the many advantages this technology has over prior approaches. Using reactive supervision, the authors of this work produced a sizable dataset of tweets with prospective labels and contextual variables that they hope will boost sarcasm detection studies. This dataset presents fresh possibilities for sentiment analysis and emotion recognition research in affective computing. In the context of natural language processing (NLP), responsive monitoring represents a novel approach to data acquisition.

Since more Bangla is being used online, it's critical to analyse Bangla text data to preserve a harassment-free and safe online environment. Cyberbullying and online harassment are serious issues, and it's already normal practice to threaten people via electronic communication channels like social networking sites. The writers of the PDF file have assembled a dataset of 44,001 Bangla comments from open Facebook posts that has been labelled with various types of harassment in order to address this problem. Machine learning models for identifying online harassment in Bangla text can be created using the dataset. The dataset was also subjected to an exploratory study by the authors to determine the various types of Bengali bullies. The study emphasizes the need for additional investigation in this area to create successful preventive strategies against cyberbullying.

The first paradigm shift is moving away from rule-based procedures and towards statistical techniques that employ characteristics like sentiment shifts and semi-supervised patterns. The second change involves the use of deep learning techniques, which take into account contextual information like the stylometric characteristics of authors and the type

of discussion topics. The third change is a move towards incorporating contextual knowledge through language models. In addition, the authors emphasize numerous methods for introducing context into sarcasm detection algorithms, including author context, conversational context, and topical context. In order to enhance sarcasm detection models, the authors also cover the usage of user embeddings to encode users' stylometric and personality attributes. The PDF lists a number of research that looked into how to recognise sarcasm in a variety of languages, including Czech and English. In its entirety, this PDF offers a thorough examination of the present status of mockery identification research, encompassing a wide range of topics and highlighting areas that remain unresolved.

III. METHIDOLOGY

We first assembled our dataset and used a variety of preprocessing. After that, the preprocessed data was split 80:20. The training data was then balanced, and The data was trained on four models: Support Vector Machine, Random Forest, Logistic Regression , and a combination of GRU and CNN models. Finally, we use test data to evaluate our model.

A. Dataset

As we've already mentioned, there hasn't been much research done on Bengali sarcasm recognition. We looked into using BanglaSarc and made that decision [1]. We selected this dataset since it was compiled from content that was collected from online blogs, Facebook comments, and YouTube videos. This page contains a link to a download of the Dataset of the Bangla Sarcasm. There are 5021 data in this collection.

Label	Number of Data
Sarcastic	1945
Non-sarcastic	3076
Total	5021

B. Pre-Processing

The majority of the dataset of the Bangla Sarcasm was pre-processed to a large extent. Nonetheless, we examined a few well-known preprocessing techniques, that significantly improved our model's functionality. First, using RegEx, we eliminated all of the emoticons and other emojis.

Another issue that was brought to our attention was the unbalanced nature of the BanglaSarc dataset. We increased number of samples of data being collected only in the training split to address this issue and prevent data leaking between the training split and the test split. Oversampling may result in overfitting to an initial training set of data, although it can be easily avoided by ending the analysis early.

C. Embedding

We tested our machine learning model after the preprocessing step was finished using a count vectorizer and a tf-idf, which is called a Support Vector Machine. Embedding in GloVe [6] was utilized for our deep learning models, which were built with GloVe. In the deep learning model, count vectorizer and word2vec were eliminated since the tf-idf and GloVe embedding approaches performed more effectively. We have made use of the pre-trained GloVe embedding that BNLP [7] offers. We decided to use a 300-dimensional embedding on 39 million tokens that had been created via crawling news articles and Wikipedia. The embedding is available for download from this location.

D. Machine Learning Model

As our machine learning models, we went with the four models that we discussed earlier.

Support Vector Machine: It can be found in a variety of kernel function configurations. The process of determining a hyperplane, sometimes referred to as a decision limit, is a fundamental aspect of Support Vector Machine (SVM) models. Classifying each information source according to a predetermined set of characteristics is the major focus of a support vector machine (SVM) framework.

The objective in a space with N dimensions is to identify the hyperplane that maximally separates the data points of two distinct classes. This hyperplane can take on a variety of shapes depending on the dimensions of the space. The expense variable associated with the Support Vector Machine (SVM) model is expressed in a specific mathematical formulation that may be represented mathematically as follows: The objective function $J(\theta)$ is represented as:

$$J(\theta) = \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

$$\theta^T x^{(i)} \geq 1, y^{(i)} = 1$$

$$\theta^T x^{(i)} \leq -1, y^{(i)} = 0$$

The code that was just presented makes use of a linear kernel. In most cases, fitting multidimensional or data points that are difficult to simply separate requires the employment of a kernel. In light of the circumstances at hand, we have employed many models including sigmoid support vector machine (SVM), kernel support vector machine (SVM) with polynomial kernel, Gaussian support vector machine (SVM), and basic linear support vector machine (SVM).

Logistic Regression: The sigmoid function of f(x) is the logistic regression function, denoted by p(x), which is used in logistic regression:

$$p(x) = \frac{1}{1 + e^{-f(x)}}$$

As a consequence of this, it's frequently quite close to either 0 or 1. To categorize problems into multiple or binary categories, the intuitive equation required for doing so is provided by a logistic regression model, which is used. Because we are classifying text based on such a large set of features, this is the case, resulting in a simple binary choice (which is actually true or untrue). While a number of factors were investigated prior to acquiring the highest possible precisions from the LR model, we utilized hyperparameter tuning in order to gain the most desirable results possible for every individual dataset. A

theoretical description for the LR assumption function can be provided as such:

$$h_\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Using a sigmoid function, the output of logistic regression is transformed into a likelihood ratio. Achieving the highest probability is the goal while simultaneously minimizing the cost function. The following can be deduced from the calculation of the cost function:

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)), & \text{if } y = 1 \\ -\log(1 - h_\theta(x)), & \text{if } y = 0 \end{cases} \quad (1)$$

Convolutional Neural Networks (CNNs): In order to interpret information having a grid-like structure, like a picture, a specific type of artificial system was developed. They are made up of several "convolutional" layers, with each one implementing a different number of filters to the input fed into the system. These filters are small matrices that are slid over the input data, the data is multiplied element by element, and the results are added together. Specific patterns which are in the collected data are found using this method. Before being fed into the subsequent layer, using a nonlinear activating functioning, for instance, ReLU, to function the output of the layers of convolution is first processed. Only then it is provided to the following layer. Additionally, "pooling" layers in CNNs are in charge of downsampling the final results of the layers that perform convolution. Applying a preset function of a window to the resulting output data does this, such as maximum or average pooling. This helps to lessen the likelihood of the model being overfit by lowering the total number of parameters it uses. Additionally, it lowers the dimensionality of the data, which makes it easier to analyze using computer resources.

Random Forest Classifier: It uses a Random Forest Classifier. A sort of ensemble learning technique called random forests can be applied to regression, classification, and numerous other kinds of tasks. As part of the process training phase, this approach builds a variety of decision trees. The category that was picked by the vast majority of trees is what the random forest produces when applied to classification issues. The outcome of regression analysis is the calculation of an overall or typical forecast for each member of the tree. Random choice forests address and resolve the issue of decision trees' propensity for overfitting to their training sets. In a typical basis, random forests exhibit superior performance compared to decision trees, while their precision is shown to be lower to that of gradient-boosted trees. Nevertheless, the effectiveness of their performance can vary based on the specific qualities of the data.

FFNN: There is no cycle in the connections between the nodes of this, which functions like a synthetic neural network. Also known as a feed-forward network. In this model, we have tried out a variety of activation functions to see which one works best. In order to create the FFNN, we relied on the relu function.

$$f(x) = \max(0, x)$$

Max Pooling: Downsampling the output of the convolutional layers in CNNs is accomplished with the help of a method called max pooling. The process involves applying a maximum filter on the layers' of the convolutional output. A preset frame of data is examined by this filter, which then delivers the value that is highest overall. Because most of the values are thrown away and only the highest ones are kept, the amount of output data is effectively reduced in size as a result of this.CNN becomes more resilient to minute translations in the information that is input as a result of max pooling. In addition, it is beneficial to mitigate overfitting by decreasing the quantity of parameters utilized in the model. It also has the impact of making CNN more accurate.

GRU: It is a type of recurrent neural network (RNN) that is specifically designed to process consecutive input, such as historical data or plain language. Its primary use is in the field of artificial intelligence. It is made up of "gates," that regulate the transmission of data through and out of the unit. This feature enables the system to use discretion in retaining or discarding data from preceding time steps. The "reset" gate is responsible for determining which information from the preceding phase should be disregarded, while the "update" gate is responsible for determining which information should be retained. Tanh activation functions that are used to create these gates produce values between 0 and 1 as their output. These values indicate the level of security provided by the gate.

Bidirectional GRU: A sort of GRU known as a bidirectional GRU is one that has the ability to process data in sequence either forward as well as rearward. This enables it to consider the circumstances on both ends at a particular time phase. When doing tasks such as processing natural language, in which the meaning of a word can change depending on the words that follow before and after it, this can be advantageous. Two independent GRUs are utilized in the process of implementing a bidirectional GRU. The input data are processed forward by the first GRU, whereas the second GRU conducts reverse data processing. In the subsequent stage, the merged results of both of these Gated Recurrent Units (GRUs) enable the model to take into account the contextual information pertaining to the data present on the information from all the left and opposite sides.

The lost algorithm used in the analysis that we made use of was binary cross-entropy, and the metrics that we made use of were binary accuracy.

$$\text{LBCE} = -\frac{1}{n}\sum_{j=1}^{n}\left(Y_i \cdot \log(\hat{Y}_i) + (1 - Y_i) \cdot \log(1 - \hat{Y}_i)\right) \quad (2)$$

Due to the fact that the training of the weight of the sample size was enhanced, we have a sneaking suspicion indicating the model might have a problem with overfitting. We choose to track the validity loss in an effort to discover a solution to this issue and terminate the model before its completion.

IV. INVESTIGATION AND OUTCOMES

Our group has conducted many evaluations, including initial processing and concept assessments. We attempted to get rid of the emoji, punctuation, and other extraneous strings by utilizing a variety of different regular expressions. This led to an improvement in our performance. Our investigation demonstrated that our preprocessing phase was successful, as we discovered a significant increase in the quality of the results produced by our machine learning models. Both the Logistic Regression model, Random Forest model have provided us with an accuracy of 88%. It is an increase of 25% contrasted with the Logistic Regression model used in the BanglaSarc study. The Support Vector Model also exhibited an improvement in accuracy that was 12% higher. Nevertheless, our ensemble model for sarcasm detection, which is comprised of Gated Recurrent Units and Convolutional Neural Networks, performed the best compared to all of the other models. As a validation set, we made use of twenty percent of our total training data. In training, the model had an accuracy of 95%, and in validation, it had an accuracy of 95%. In the test set, it had an accuracy rate of 95% and received a score of 95% on the Macro Average F1. The comprehensive evaluation is presented the subsequent table III for your perusal.

Name Of the Model	Accuracy	Recall	F1 ranking
LR (Logistic Regression)			
Non-Sarcastic	87%	93%	91%
Sarcastic	88%	79%	83%
Macro Average	88%	88%	87%
Accuracy	88%		
SVM (Support Vector Machine)			
Non-Sarcastic	83%	97%	90%
Sarcastic	92%	66%	77%
Macro Average	88%	81%	83%
Accuracy	87%		
RF (Random Forest)			
Non-Sarcastic	85%	99%	91%
Sarcastic	97%	68%	80%
Macro Average	91%	83%	86%
Accuracy	89%		
GRU + CNN			
Non-Sarcastic	96%	96%	96%
Sarcastic	94%	93%	94%
Macro Average	95%	95%	95%
Accuracy	95%		

This table potentially presents multiple instances showcasing the practical application of our Bangla humour identification technology. The following chart provides a comparative analysis of four distinct approaches previously discussed. For us, the two types of LR and random forest models achieve the same degree of accuracy, which comes to 89% here. Less accurate than other methods, support vector classification achieves an 87% success rate. On the other hand, the GRU model as well as CNN model produced the highest outcome with an accuracy of 95%. This was achieved.

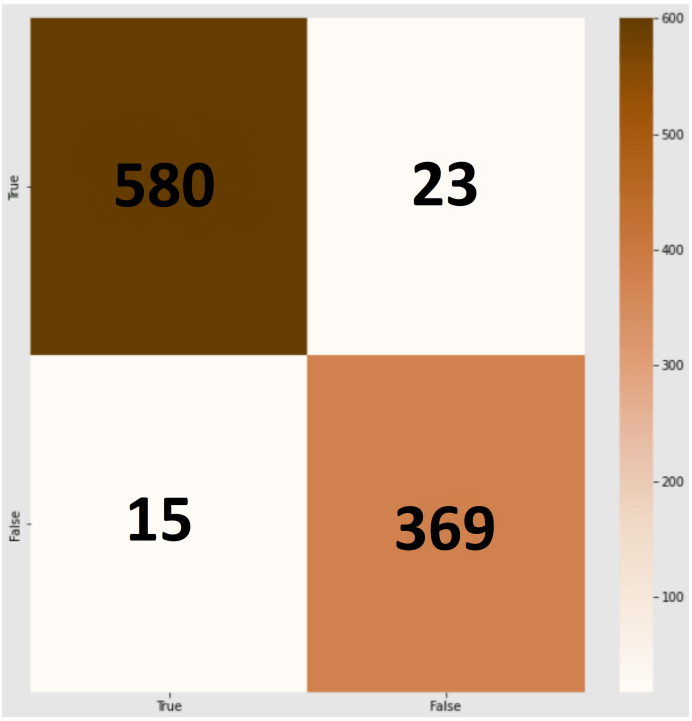

Fig. 1. Confusion Matrix of Model

In addition, to have a better understanding of our model's performance, we have plotted a confusion matrix, which can be found in Figure 3. It is possible to deduce from the matrix that the genuine positive and true negative values are, respectively, 580 and 369. The occurrences of both inaccurate results and negative results are relatively low, specifically 23 and 15, respectively, making them inconsequential. We can observe that our model is marginally overfitted, but this difference is not significant at all.

V. CONCLUSION

We implemented cutting-edge NLP strategies and Deep Learning architectures to provide an "AI for Bengali Sarcasm Detection" in this study. As anticipated, a balanced dataset and improved preparation led to a higher score for us. Despite using a rather straightforward Deep Learning model, we achieved significantly better performance by using contemporary embedding techniques like GloVe. Future research on more complicated Deep Learning models like BERT, which can use a variety of architecture combinations, would be intriguing to investigate. Additionally, we propose developing a a task-specific high-quality dataset that combine hostile language identification with sarcasm recognition.

REFERENCES

[1] Apon, T. S. (2022, September 27). BanglaSarc: A dataset for Sarcasm Detection. arXiv.org. https://arxiv.org/abs/2209.13461

[2] Human verification. (n.d.). https://www.semanticscholar.org/paper/A-Review-on-Sarcasm-Detection-from-Machine-Learning-Wicana-Ibisoglu/e1b6c4cb181eb934574207f0207e7874a12b3655
[3] Hossain, M. Z. (2020, April 19). BanFakeNews: A dataset for detecting fake news in Bangla. arXiv.org. https://arxiv.org/abs/2004.08789
[4] Yaghoobian, H. (2021, July 5). Sarcasm Detection: A Comparative study. arXiv.org. https://arxiv.org/abs/2107.02276v1
[5] Mishra, A. (2017, January 19). Harnessing cognitive features for sarcasm detection. arXiv.org. https://arxiv.org/abs/1701.05574
[6] Wright, B. (2020). Sarcasm detection using machine learning algorithms in Twitter: A systematic review. Coventry. https://www.academia.edu/82766703/Sarcasm_detection_using_machine_learning_algorithms_in_Twitter_A_systematic_review
[7] Ahmed, M. F. (2021, February 4). Bangla Text Dataset and Exploratory Analysis for Online Harassment Detection. arXiv.org. https://arxiv.org/abs/2102.02478