Genre Classification: A Machine Learning Based Comparative Study of Classical Bengali Literature

Asadullah Al Galib, Maisha Mostofa Prima, Satabdi Rani Debi, MD Muntasir Mahadi,
Nayema Ahmed, Adib Muhammad Amit, Ehsanur Rahman Rhythm, Annajiat Alim Rasel
Department of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
{asadullah.al.galib, maisha.mostofa.prima, satabdi.rani.debi, md.muntasir.mahadi,
nayema.ahmed, adib.muhammad.amit, ehsanur.rahman.rhythm}@g.bracu.ac.bd, annajiat@bracu.ac.bd

Abstract—**Bengali classical literature holds immense cultural and historical significance, encompassing a rich collection of works ranging from poetry to plays, novels, and essays. However, compared to other languages, there has been limited research and development in leveraging NLP techniques for analyzing Bengali classical texts. By addressing this gap, we aim to unlock the potential of NLP in exploring and understanding this literary heritage. The first objective of this research is to compile a comprehensive dataset of Bengali classical literary texts. This dataset will serve as a valuable resource for researchers and developers interested in exploring various aspects of Bengali classical literature using NLP techniques. The second objective is to apply NLP techniques for genre classification of Bengali classical texts. Genre classification is crucial for organizing and categorizing literary works, enabling researchers to easily access specific genres of interest. By employing machine learning algorithms and NLP methods such as text preprocessing, feature extraction, and classification models, we aim to develop an automated genre classification system specifically tailored to Bengali classical literature. The outcomes of this research have broad implications for the preservation and accessibility of Bengali classical literature. By automating the analysis and categorization of these texts, researchers and enthusiasts can efficiently navigate through vast repositories of archival materials, leading to deeper insights into the historical, cultural, and linguistic aspects of Bengali classical literature.**

Index Terms—**Bengali Literature, Naive Bayes, LSTM, Transformer**

I. Introduction

Bengal's literary history is seen to have reached its literary zenith during the 19th century, especially in terms of classical Bengali literature. It was a time when the region saw the rise of notable writers like Kazi Nazrul Islam, Rabindranath Tagore, Bankimchandra Chattopadhyay and Sarat Chandra Chattopadhyay who had a lasting impact on the cultural environment. Exploring the many genres of ancient Bengali literature and examining its traits and themes is an important part of knowing and appreciating its richness.

In order to recognize and comprehend the unique qualities, themes, and creative components present in a given body of work, genre classification is essential to literary study. We learn about the many literary styles and idioms that were in vogue at this time by exploring the genres found in classical Bengali literature.

Poetry is an important literary genre in traditional Bengali literature. Bengali poetry comes in a variety of formats, including lyrical pieces and narrative lines. It displays a wide range of style, structure, and content. One can explore the vivid world of rhymes, sonnets, epics, and other literary forms within the field of poetry, which highlight the mastery of language and imagery by well-known poets of the time.

Drama is an important sub-genre of traditional Bengali literature. The existence of dramatic works, such as plays and theatrical productions, gives the literary landscape a vibrant new dimension. Bengali playwrights created engrossing stories that they then brought to life on stage by examining human emotions, social problems, and philosophical ideas. Drama was a genre that offered a stage for social critique, amusement, and contemplation, making it a crucial component of traditional Bengali literature. We can learn more about the cultural, social, and intellectual climate of the times by examining the traits and subjects of various genres in classical Bengali literature. Each genre presents a distinct viewpoint and highlights the originality, inventiveness, and literary skill of the writers who helped Bengali literature grow.

In this exploration, we will contrast and evaluate the main literary sub-genres of ancient Bengali literature, emphasizing their unique characteristics, theme investigations, and creative contributions. By doing this, we can fully comprehend the complex nature of traditional Bengali literature and recognize its lasting significance in Bengal's and other countries' literary traditions.

II. Related Work

Waleed A. Yousef et al. [1] trained recurrent neural networks (RNN) at the character level on English and Arabic written poems in order to learn and identify the meters that give the poems their phonetic pronunciation. Datasets crawled from non-technical sources were cleaned, formatted and published to a publicly accessible repository for scientific research.

The work of Parilkumar Shiroya et al. [2] compare three machine learning algorithms: K-NN, SVM and LR for classifying books by their genres based on their titles and abstracts. The paper uses a customized dataset that consists of books that are translated to English from Gujarati or Hindi origin books.

The paper reports that SVM achieved the highest accuracy and was fast in processing and predicting output among the three algorithms.

The work of [3] Joseph Worsham et al. develop approaches for genre identification that can be used with complex and exceedingly large literary works. The paper uses the Gutenberg Dataset and compares current models to traditional methods and evaluates various machine learning approaches, including CNN, LSTM, HAN, Naive Bayes, k-Nearest Neighbors, Random Forests, and XGBoost. XGBOOST outperforms all models, achieving the highest accuracy of 84% among deep learning models.

Anshaj Goyal et al. [4] compared machine learning and deep learning approaches for literary genre classification using the same Gutenberg dataset as the previous paper for experiment. The experiment used three statistical machine learning algorithms (Random Forest, Naïve Bayes, and SVM) and four deep learning approaches (LSTM, CNN, RNN, and BERT) on the entire texts for classification. The SVM classifier achieved 85% accuracy in genre prediction among the machine learning models, while the RNN approach performed better than among the deep learning models. The RNN approach outperformed all other models in the evaluation.

III. Methodology

The methodology for this paper involves:

- Dataset analysis to compile a comprehensive collection of Bengali classical literary texts.
- Building baseline model Naive Bayes for genre classification as performance benchmarks.
- Training deep learning models (e.g., LSTM, Transformer) tailored to the characteristics of Bengali classical literature.
- Analyzing results by evaluating metrics like accuracy, precision, recall, and using confusion matrices and classification reports.
- Discussing future directions, including advanced pretraining techniques and exploring related tasks like sentiment analysis or authorship attribution, to advance NLP in Bengali classical literature.

A. Naive Bayes:

Naive Bayes is based on Thomas Bayes' 18th-century Bayes' theorem, which machine learning researchers later improved. This approach computes classification probabilities based on prior information while assuming feature independence. Utilising the Bayes theorem, naive Bayes functions by calculating the conditional probabilities of features given classes.[6]

The well-known classification technique, Naive Bayes, is frequently used for a variety of text categorization applications across several languages. Naive Bayes in text classification: Bayes' theorem is the foundation of the probabilistic approach known as naive Bayes. For simplicity, it assumes feature independence and calculates conditional probabilities of class labels based on document features. It evaluates the chance that a document fits one or more predetermined categories in applications like sorting Hindi poems by mood. Multinomial Naive Bayes assesses probability taking feature occurrences into consideration for sentiment analysis of Bengali tweets, assisting in sentiment categorization. Similar probabilities are determined by Naive Bayes for the classification of texts in Turkish and Urdu. Naive Bayes frequently performs well in text classification scenarios.[5]

Basically, Naive Bayes determines probability based on features. The algorithm plays a crucial role in categorization, sentiment analysis, and document labeling across several languages for text classification tasks.[8]

Text classification is a fundamental task in natural language processing (NLP) that entails categorizing or labeling a piece of text. Various techniques have been developed over the years to address this problem, with recurrent neural networks (RNNs), Long Short-Term Memory (LSTM) networks, and, more recently, Transformers emerging as two powerful approaches. In this post, we'll look at the LSTM and Transformer models in the context of text classification, highlighting their advantages and disadvantages as well as real-world applications. The architecture of the Transformer is encoder-decoder. Long-term dependencies are captured by self-attention in both components, which is excellent for NLP tasks. It is essential for machine translation, language modeling, sentiment analysis, and other sequential data applications due to its parallel processing and complicated linkage handling. Both LSTM and Transformer can be used in text classification tasks, where the goal is to assign a label or category to a given text.

B. LSTM:

1) LSTM in Text Classification: Long Short-Term Memory (LSTM) networks were conceived with the express purpose of capturing and retaining long-range dependencies within sequential data, rendering them highly suitable for a diverse array of Natural Language Processing (NLP) tasks, prominently including text classification. Textual data is inherently sequential, where the arrangement of words within a sentence can dramatically alter its semantic significance.

The inception of LSTM can be traced back to 1997 when two German scientists proposed this novel architecture. At its core, LSTM sought to address a significant shortcoming observed in vanilla Recurrent Neural Networks (RNNs) – the problem of vanishing and exploding gradients[19]. What sets LSTM apart is its ingenious incorporation of memory cells and gating mechanisms. These mechanisms encompass input gates, forget gates, and output gates. It's this architecture that fundamentally differentiates LSTM from its predecessors.

The crux of LSTM's capability lies in its ability to retain crucial information across extended sequences, a boon for NLP and time series forecasting alike. The concept of gating mechanisms, a linchpin of LSTM, is pivotal. These mechanisms effectively manage the flow of information by selectively capturing dependencies within sequential data over time. By orchestrating this flow, LSTMs excel in tasks necessitating the comprehension of prolonged, intricate dependencies. This inherent strength makes them particularly valuable for grappling with sequential data characterized by intricate patterns and interconnections.

In the realm of text classification, LSTM-based models have left an indelible mark. Applications like sentiment analysis, spam detection, and document categorization have all reaped the benefits of LSTM's prowess. Beyond these foundational tasks, LSTMs have ventured into more sophisticated domains, including machine translation and speech recognition, where the capacity to model and comprehend intricate dependencies within sequential data is of paramount importance.

In summary, LSTM networks represent a watershed moment in the field of neural networks, particularly in the context of NLP. Their innate ability to grapple with extended dependencies within sequential data has propelled them to the forefront of text classification and numerous other applications where understanding complex, long-term relationships is pivotal.

C. Transformer:

1) Transformer in Text Classification: Natural language processing (NLP) has many diverse real-world applications, and one of these applications is text classification. Its main goal is to classify textual items into predetermined labels or tags, ranging from single lines to extensive documents. Numerous fields, including sentiment analysis, news categorization, user intent classification, content moderation, and others have found great use for this analytical skill. Despite the fact that there is an abundance of textual data from sources like web pages, social media, emails, online news portals, user reviews, and customer service interactions, preparing this data for classification remains a difficult task because of its inherently unstructured and noisy nature.

Text classification has consistently relied significantly on traditional machine learning algorithms. In order to complete this procedure, it was necessary to laboriously choose and engineer characteristics by hand, such as the bag of words or n-grams, which were then fed into classification algorithms like Naive Bayes (NB), Support Vector Machine (SVM), and more. This lengthy procedure frequently required deep feature engineering knowledge as well as domain expertise.

However, the availability of large-scale datasets and the rise of deep learning algorithms have caused a paradigm shift in the approach to text categorization in recent years. Deep learning models have the amazing capacity to learn representations on their own from unprocessed data, reducing the requirement for complex feature engineering. Additionally, these models show that they are capable of being transferable across a variety of jobs.

Transformers and their related pre-trained language models have become powerful tools in the most recent era of text classification [21]. These models take advantage of the enormous amount of unlabeled data to construct sophisticated representations. The ability to adapt makes their abilities stand out; they excel at both understanding language nuance and adjusting to various downstream duties.

The BERT (Bidirectional Encoder Representations from Transformers) model is one demonstration of these transformers [22]. BERT portrays a strong knowledge of contextual and semantics due to its extensive pre-training on text corpora. On particular text categorization tasks, fine-tuning BERT has produced amazing results, frequently outperforming conventional methods.

These pre-trained language models' performance depends on their capacity to understand the nuanced nature of human language. They capture the text's underlying semantic meaning and context alongside its surface-level properties. This deep understanding enables them to perform brilliantly in a wide range of text classification tasks, such as sentiment analysis, where it's important to recognize subtleties in sentiment, and content moderation, where spotting incorrect material requires a delicate command of language.

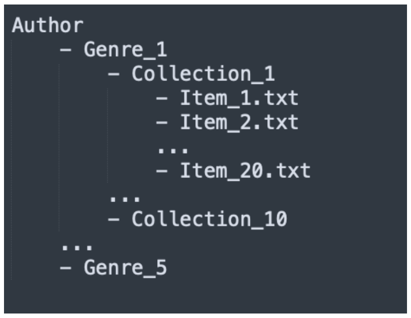
In essence, text categorization has evolved dramatically because it was formerly a difficult operation requiring manual feature engineering and the use of traditional algorithms. Transformers and pre-trained language models in particular from deep learning models have changed the area.

IV. Dataset Compilation

Even though the world of Large Language Models has ushered in a new era in every domain of human knowledge, the sheer advancement in the field of LLM is predominated by only a handful of languages. And in most cases, the datasets for these languages were collected from various internet crawling services such as Common Crawl [9]. And even when the datasets are in a partially covered language, they are mostly collected from various news or blog sites on the internet [10]. Conducting any significant literary analyses in a language requires verified and authentic datasets in a readily usable format.

To facilitate research and explore the untapped riches of classical Bengali literature, we initiated a project to compile a comprehensive collection of classical literary works. Previously, we compiled the datasets for Rabindranath Tagore and Kazi Nazrul Islam [11]. To compile the datasets from an authentic source, we choose the digitized literary works published by the Department of Information Technology & Electronics, Government of West Bengal, India [12]. So far, they have digitized the entire collections of four authors, namely Rabindranath, Nazrul, Sarat, and Bankim. The overall content hierarchy for each author is described in Figure 1.

In order to handle this complexity and dynamic content hierarchy, we designed the crawler in such a way that, given the landing URL for each genre, it can automatically crawl all the collections, items inside the collections, and all the pages for an item. There are two parts to dataset compilation, crawling & parsing content and aggregating & formatting the content to usable formats. Both of these segments are described below:



```
Author
    - Genre_1
        - Collection_1
            - Item_1.txt
            - Item_2.txt
            ...
            - Item_20.txt
        ...
        - Collection_10
    ...
    - Genre_5
```

Fig. 1: Content hierarchy

A. Crawling & parsing content

The crawler is created using *requests* and *beautifulsoup* libraries. We used the *requests* library to make the GET requests to fetch the raw HTML page content and *beautifulsoup* was used to parse and filter the relevant sections from the HTML contents. To identify and extract specific sections from the HTML, we used CSS selectors for its simplicity to use. Starting from the landing page of a genre, the crawler can independently crawl through all the navigation links of that content and create separate text files for each content for further downstream processing.

B. Aggregating & formatting content

Given an author, the aggregator starts from the root directory of that author where the text files were stored during the crawling step. it then traverses the nested directories for different genres and collections and reads the contents of individual items from the text files. For each content, cleaning and basic preprocessing steps are run. Finally, separate CSV and TXT files are created for each genre comprising all individual items found in that genre.

C. Challenges

While crawling the content from the mentioned site, we faced some difficulties due to the HTML structure and layout used in the site. Some of these challenges are described here:

- Each author repository had its own content layout and hierarchy structure for different genres.
- Some genres started content from the starting URL, whereas others showed some introductory metadata, and the actual content started from the next page.
- Pagination links were broken for some contents which made it difficult for the crawler to crawl the content of a single item in a sequential order.

To overcome these challenges, we used different handlers for specific genres of each author. A fallback mechanism was added to deal with broken pagination links so that if the crawler failed to parse which page to visit next using the default resolver, it used the secondary resolver to identify next-page links.

V. Dataset Analysis & Preprocessing

After conducting an initial analysis of all four datasets, we excluded Bankim from the training data due to its lack of diverse genre distribution. Moreover, the story genre seemed to include other genres that might introduce incorrect labeling in the training data. We aggregated the remaining three datasets, namely Nazrul, Rabindra, and Sarat into a single CSV file. When we refer to the all-collection dataset, we are pointing to this aggregated version.

Since different authors have different genre distributions and some authors do not have any work in certain genres, we decided to consider the genres that have a good distribution across all three authors. After analysis, we discarded works from all other genres except these four categories - novel, essay, story, and poem. In our classification tasks, we aim to predict the genre class for a given text from these four categories.

A. Analysis

First, we conduct some basic exploratory analysis on the aggregated dataset. From Figure 2, we can see that Rabindranath has written more stories than both Saratchandra and Nazrul. The sheer number of poems by Rabindranath compared to Nazrul is very high. Saratchandra, however, has written more novels compared to the other two. Out of the three, only Rabindranath has a gigantic number of essays, whereas both Saratchandra and Nazrul have written only a handful of essays.

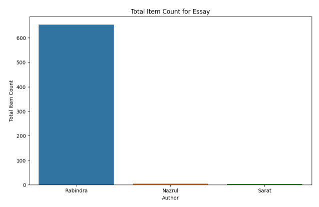
When we look into the average word count across genres for each author, we notice something interesting. Even though Rabindranath has written more essays, out of a handful of essays written by Saratchandra, the average word count in the essay genre is much higher than both Nazrul and Rabindranath. All three authors have a similar word count for the novel, with Nazrul having the lowest average word count. For both poem and story genres, Nazrul has a much higher average word count compared to the other two. The average word count in the poem genre for Nazrul is higher than Rabindranath. See Figure 3.

Individually, all three authors have a higher average word count in novels compared to other genres. This is expected since the novel genre has a longer format than the rest.

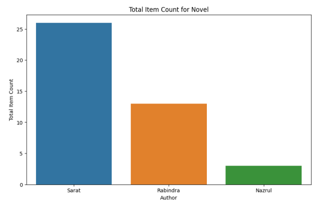
B. Preprocessing

After aggregating all three datasets, we conduct further preprocessing to remove unnecessary symbols and punctuations as well as splitting the longer items into shorter versions retaining the genre and author intact. To create shorter segments of items from longer genres such as novels, stories, and essays, the text is first split using Bengali line-ending punctuations such as '—', '?', and '!'. Then, using a predefined mapping, the individual sentences are merged into shorter segments. For example, if we have a novel of 1000 lines, and we split it into segments of 10 lines each, then after the split-and-merge step, we will have 100 novel segments.

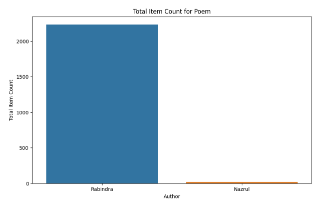
This splitting step allows us to create a uniform sequence length for each genre since training an LSTM requires inputs



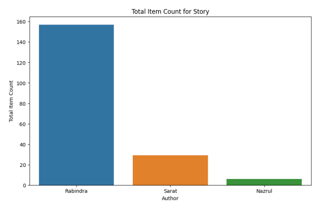
(a) Essay



(b) Novel

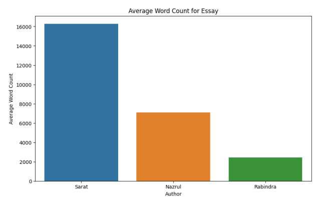


(c) Poem

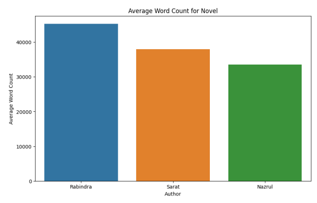


(d) Story

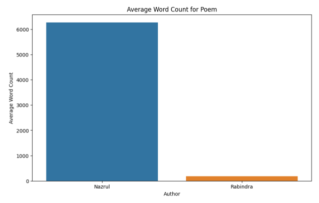
Fig. 2: Total item count by genre



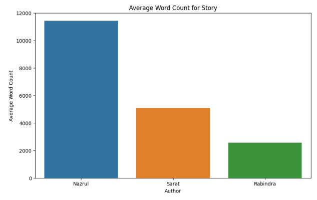
(a) Essay



(b) Novel



(c) Poem



(d) Story

Fig. 3: Average word count by genre

of certain lengths. Without splitting the content like this, we would need to either pad shorter contents to match the maximum length of longer content or cut longer contents to match the shorter items. From Figure 4 (a, b), we can see the effect of preprocessing the dataset by splitting and merging. Figure 4 (c) shows the impact of splitting by creating items in all genres with a much closer average word count.

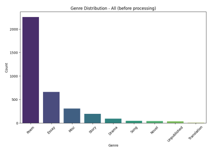
VI. Model Building & Training:

A. *Naive Bayes*

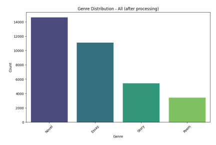
To perform genre classification on classical Bengali literature, we employed some machine learning approaches and one of them involved Multinomial Naive Bayes (MNB) classification. First, the raw textual content was tokenized using the Natural Language Toolkit (NLTK) tokenizer for the Bengali language. Then stopwords and digits were removed from the tokenized content to retain only meaningful words. The preprocessed content was then joined to form coherent documents. Each genre label was encoded into numerical values using the Label Encoder from the scikit-learn library. We employed the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique to convert the textual data into numerical feature vectors. The TF-IDF vectorizer was fit on the training data to learn the vocabulary and IDF values. We performed hyperparameter tuning using Grid Search with cross-validation. The hyperparameters considered were 'alpha' and 'fit prior' for the Multinomial Naive Bayes classifier. Notably, we employed stratified sampling (stratify=y) during the train test split due to data imbalance. The Multinomial Naive Bayes classifier was trained on the training data using the best hyperparameters obtained from the grid search.

B. *Deep Learning*

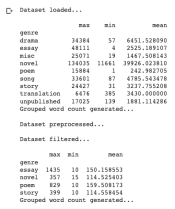
For the genre classification task, we used two deep learning models, namely LSTM and transformer-based fine-tuned BERT. For LSTM, we considered both the regular and bidirectional LSTM. We also explored the model performance



(a) Essay



(b) Novel



```
Dataset loaded...

                max    min          mean
genre
drama          34384     57   6451.528090
essay          48111      4   2525.189107
misc           25071     19   1467.508143
novel         134035  11661  39926.023810
poem           15884      1    242.982705
song           33601     87   4785.543478
story          24427     31   3237.755208
translation     6476    385   3430.000000
unpublished    17025    139   1881.114286
Grouped word count generated...

Dataset preprocessed...

Dataset filtered...

        max  min        mean
genre
essay  1435   10  150.158553
novel   357   15  114.525403
poem    829   10  159.508173
story   399   10  114.558454
Grouped word count generated...
```

(c) Novel

Fig. 4: Effect of content splitting & merging

for single-layer and stacked-LSTM layers for different hyper-parameter values.

1) LSTM: In order to build and train an LSTM model, first, all the contents are tokenized using a ‘max vocabulary size’. Then texts are converted to sequences and padded to match a given ‘max sequence length’. Then a dynamic custom model builder is used to create a model architecture based on different hyperparameter values. We used Adam Optimizer with a small learning rate. For the loss function, we use ‘categorical cross-entropy and ‘accuracy’ as the performance metric.

Initially, we trained both the regular and bi-directional LSTM models for different combinations of various hyper-parameters. Some of these are - learning rate, max vocabulary size, max sequence length, embedding dimension, number of LSTM units, batch size, dropout, regularization, number of LSTM layers, and number of epochs. After extensive experimentation with different values, we finalized the training architecture of LSTM models. For training with the aggregated dataset, we varied the model architecture by considering batch sizes of 32 and 64. we ran both the regular and bi-directional stacked LSTM layers for each batch size. To measure the effectiveness of stacked layers, we ran another version of the models by keeping the batch size at 32 but varying the number of LSTM layers. We also carried out hyperparameter tuning by tweaking the max vocabulary size and lax sequence length.

2) Transformers: For the transformer-based classification, we considered three pre-trained language models trained on the Bengali language and then fine-tuned the models for the genre classification task. BERT and similar encoder architectures consider each token of a text in the context of all the tokens left to it as well as all the tokens right to it [13]. We used the *Simple Transformers* [18] library and its classification model to fine-tune the pre-trained language models for the genre classification task. For all three models, we enabled CUDA for faster training.

Bangla-Electra: It is a language model trained on the Bengali language using ELECTRA from Google [17]. The model was trained on OSCAR crawled dataset and Bengali Wikipedia dump [14].

Bangla BERT Base: This is a pre-trained language model using mask language modeling and trained on the same dataset as Bangla-Electra [15].

BERT Multilingual Base Model (Uncased): This model also used masked language modeling, but was trained on Wikipedia for the top 102 languages [16].

VII. Result Analysis

A. Naive Bayes

After training the genre classification model, we evaluated its performance using various metrics to gauge its effectiveness in distinguishing between different literary genres. The following analysis summarizes the results of the Multinomial Naive Bayes model: The accuracy of the model on the testing data was calculated, revealing how well the model predicts the correct genre labels. The confusion matrix provided insights into the distribution of true positive, true negative, false positive, and false negative predictions for each genre. Class-wise accuracy was computed to measure the model’s precision for individual genres. The accuracy of predictions for each genre was presented, highlighting the model’s performance for different genres. The classification report detailed precision, recall, F1-score, and support for each genre. The report provided a comprehensive overview of the model’s performance metrics. Overall, the results of our model training and evaluation process shed light on its potential applicability for genre classification in classical Bengali literature. Notably, our analysis yielded the following insights for each author: For Nazrul, optimal hyperparameters were observed as ’alpha’ = 0.01 and ’fit prior’ = True, yielding an accuracy of 64.71%. Similarly, for Rabindranath, the best hyperparameters were ’alpha’ = 0.01 and ’fit prior’ = True, resulting in an accuracy of 87.5%. Correspondingly, Saratchandra demonstrated optimal settings at ’alpha’ = 1 and ’fit prior’ = True, with an accuracy of 43.48%. Aggregating across authors, the best performing hyperparameters were ’alpha’ = 0.01 and ’fit prior’ = True, resulting in an accuracy of 84.44%. Moreover, we observed that by using multiple random state values, applying 42 as the random state consistently yielded higher accuracy. The analysis demonstrates the strengths and limitations of the model in distinguishing between different literary genres.

B. LSTM & Fine-tuned BERT

We generated performance measures for all the variants of LSTM models under different hyperparameter values. From Figure DDD, we can see that the performance of all variants on the aggregated dataset is quite similar, measured by accuracy, precision, recall, and f1-score. We trained two variants of the models for regular LSTM layers by using precision and recall as performance metrics instead of accuracy. However, the overall accuracy across the variants is around 70%. For

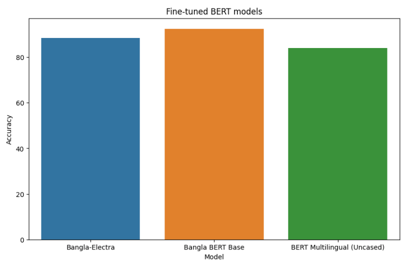


Fig. 5: Transformer model accuracy

precision, recall, and f1-scores, we considered the weighted average score to take into account the class distribution in the dataset. See Figure 5.

After training the Bangla-Electra model for 15 epochs, we gained an accuracy of 88.4%. After training both Bangla BERT Base and BERT Multilingual Base Model for only 5 epochs, we gained accuracy of 92.5% and 84% respectively. For all three models, we noticed a significant drop in loss just after 3 epochs. Figure 6 shows the difference in accuracy for the different models.

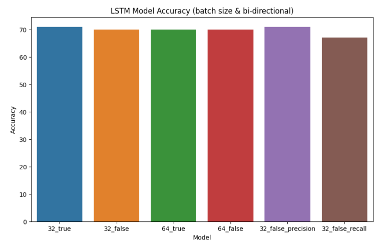
We can see that different variants of LSTM and Bi-directional LSTM models produced similar classification scores across different metrics. Transformer-based BERT models produced much higher accuracy than simple LSTM models trained from scratch. Pre-training language models played a significant role in the performance boost of the downstream task, genre classification.

VIII. Future Works

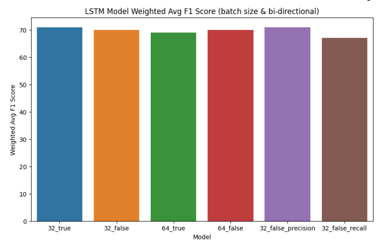
In the future, we plan to try more recent pre-trained models such as the Multilingual Representations for Indian Languages from Google. We also want to conduct a more comprehensive hyperparameter tuning to exhaust all variables and possibilities to measure the model performance under different values.

IX. Conclusion

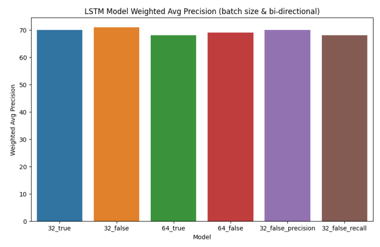
In this paper, we presented a comparative analysis of genre classification of traditional Bengali literature based on machine learning. We compiled a comprehensive dataset of Bengali classical texts from various genres and authors, which can serve as a valuable resource for future NLP applications. We applied three different classification models, namely Multinomial Naive Bayes, LSTM, and Transformer, to perform genre classification on the dataset. We used various metrics such as accuracy, accuracy for each class, confusion matrix, and classification report which included precision, recall, F1-score, and support to compare the performance of these models. We discussed the possible reasons for these challenges and suggested some ways to overcome them in future work. We also highlighted the implications of our research for the preservation and accessibility of Bengali classical literature. By developing an automated genre classification system, we enabled researchers and enthusiasts to efficiently navigate through vast repositories of archival materials, leading to deeper insights into the historical, cultural, and linguistic aspects of Bengali classical literature. We hope that our research will inspire more NLP applications for Bengali classical literature and other under-resourced languages.



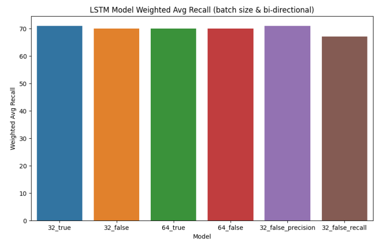
(a) Accuracy



(b) F1 Score



(c) Precision



(d) Recall

Fig. 6: Accuracy of LSTM model variants

References

[1] W. A. Yousef, O. M. Ibrahime, T. M. Madbouly, and M. A. Mahmoud, "Learning meters of Arabic and English poems with Recurrent Neural Networks: a step forward for language understanding and synthesis" (2019) 7(6), e1218.

[2] P. Shiroya, D. Vaghasiya, M. Soni, V. Patel, and B. Y. Panchal, "Book Genre Categorization Using Machine Learning Algorithms (K-Nearest Neighbor, Support Vector Machine and Logistic Regression) using Customized Dataset" IJCSMC, Vol. 10, Issue. 3, March 2021.

[3] J. Worsham, and J. Kalita, "Genre Identification and the Compositional Effect of Genre in Literature" Conf. Proceedings of the 27th International Conference on Computational Linguistics, pages 1963–1973, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[4] Goyal, Anshaj, and V. Prem Prakash. Statistical and Deep Learning Approaches for Literary Genre Classification. Advances in Data and Information Sciences, Springer Singapore, 2022, pp. 297–305.

[5] K. Pal, and B. V. Patel, ”Data Classification with k-fold Cross Validation and Holdout Accuracy Estimation Methods with 5 Different Machine Learning Techniques” 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC).

[6] K. Sarkar, and M. Bhowmick, ”Sentiment polarity detection in bengali tweets using multinomial Naïve Bayes and support vector machines” 2017 IEEE Calcutta Conference (CALCON).

[7] A. Deniz, and H. E. Kiziloz, ”Effects of various preprocessing techniques to Turkish text categorization using n-gram features” 2017 International Conference on Computer Science and Engineering (UBMK).

[8] I. Rasheed, V. Gupta, H. Banka, and C. Kumar, ”Urdu Text Classification: A comparative study using machine learning techniques” 2018 Thirteenth International Conference on Digital Information Management (ICDIM).

[9] Common Crawl Homepage, https://commoncrawl.org/, last accessed 19-08-2023.

[10] Kakwani, D., Kunchukuttan, A., Golla, S., Gokul, N. C., Bhattacharyya, A., Khapra, M. M., Kumar, P. (2020, November). IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 4948-4961).

[11] Kaggle Datasets, https://www.kaggle.com/aagalib/datasets, last accessed 19-08-2023.

[12] Literature Digitization Project Introduction, https://rabindra-rachanabali.nltr.org/node/16140, last accessed 19-08-2023.

[13] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[14] Bangla-Electra Model Homepage, https://huggingface.co/monsoon-nlp/bangla-electra, last accessed 19-08-2023.

[15] Bangla BERT Base Model Homepage, https://huggingface.co/sagorsarker/bangla-bert-base, last accessed 19-08-2023.

[16] BERT Multilingual Base Model (Uncased) Homepage, https://huggingface.co/bert-base-multilingual-uncased, last accessed 19-08-2023.

[17] Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555.

[18] Simple Transformers Homepage, https://simpletransformers.ai, last accessed 19-08-2023.

[19] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] F A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with LSTM,” Neural Computation, vol. 12, no. 10, pp. 2451–2471, 2000

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in Neural Information Processing Systems 30. Curran Associates, Inc., 2017, pp. 5998–6008

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018