Genre Classification of Bangla Poem Using Machine Learning and Deep Learning Techniques

Syed Tangim Pasha*, Ashraful Islam†¶, Mohammed Masudur Rahman‡, Eshtiak Ahmed†§, Md. Ferdouse Ahmed Foysal§, Md Zahangir Alam¶
*Department of Computing and Information System, Daffodil International University, Bangladesh
†Center for Computational and Data Sciences, Independent University, Bangladesh
‡Department of Computer Science and Engineering, International Standard University, Bangladesh
§Department of Computer Science and Engineering, Daffodil International University, Bangladesh
¶Department of Computer Science and Engineering, Independent University, Bangladesh
Corresponding Author: Ashraful Islam (ashraful@iub.edu.bd)

2023 IEEE World AI IoT Congress (AIIoT) | 979-8-3503-3761-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/AIIoT58121.2023.10174592

***Abstract*—The computational analysis of the Bangla poems is a challenging task due to the diverse linguistic, stylistic, and semantic features of the Bangla language. In this work, we prepared a dataset of 1311 Bangla poems of two separate categories: Love and Miscellaneous poem, which contain 500 and 811 poems respectively. We used word or semantic-based features to classify Bangla poems using the TF-IDF feature techniques. We used Logistic Regression, Naïve Bayes (NB), and Support Vector Machine (SVM) models for classification through machine learning, and we used Bayesian optimization techniques for hyperparameters tuning of these three models. We also used LSTM, CNN, and transformer models for this research. For the performance evaluation of the classification models, we used four evaluation metrics of precision, recall, F1-score, and accuracy. We also used the ROC-AUC curve to distinguish between all the machine learning and deep learning models. The experimental results expressed that, the transformer model achieved the highest accuracy compared to all the typical machine learning and deep learning models with an accuracy of 87%.**

***Index Terms*—Bangla Poem, Machine Learning, Deep Learning, Genre identification, Bangla Text Classification**

I. Introduction

The computational analysis of Bangla poetry presents a formidable challenge due to its diverse linguistic, stylistic, and semantic features, despite its crucial role in Bangla literature [1]. Genre classification of Bangla poems can provide valuable insights into the poems' nature, sentiment expressions, and stylistic characteristics, aiding author attribution of poems. Numerous machine learning and deep learning techniques have been employed in Bangla Natural Language Processing (BNLP) for classification tasks [1], [2]. This research focuses on classifying Bangla poems based on their genre, offering a practical application domain for BNLP research.

We initially conducted experiments on typical machine learning models, i.e., Logistic Regression, Naïve Bayes (NB), and Support Vector Machine (SVM), to improve their accuracy through hyperparameter tuning, using Bayesian Optimization techniques. This approach was chosen because it allowed for the combination of prior knowledge about the function with sample-specific knowledge, enabling the derivation of posterior information about the function distribution. Our results showed that the Naïve Bayes model achieved the highest accuracy of 86% after hyperparameter tuning. Additionally, we explored deep learning models, including the transformer model, which is well-suited for text classification tasks. Our proposed transformer-based model outperformed other models tested in our experiments with an accuracy of 87%.

This paper utilized web scraping techniques to collect 1311 poems from the "Banglarkobita" website [3], which were categorized into two collections: Miscellaneous poem (811 poems) and Love poem (500 poems). Due to the substantial amount of special symbols in the Bangla text, various data-cleaning techniques assigned. To extract semantic-based features, the TF-IDF feature techniques with count vectorizer techniques were used. Stop words were not removed to establish linguistic relations between words and long sentences in the Bangla poem. In the deep learning experiment, Keras text preprocessing techniques were applied to preprocess the dataset. The evaluation of our classification models in the result section was based on four metrics - precision, recall, F1-score, and accuracy. To distinguish between the machine learning and deep learning models, we used the ROC-AUC curve. The deep learning models were further evaluated using the confusion matrix, which considered true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values. Additionally, the loss and accuracy graphs of the deep learning models were displayed for further analysis.

II. Related Works

Researchers from all around the world worked on poems through computational analysis in different languages, for

example, English, Panjabi, France, Malay, Persian, Hindi, Arabic, etc. For Panjabi's poem classification, Kaur and Saini experimented on 2034 poems with different machine learning algorithms like NB, K-nearest neighbor (KNN), SVM, Hyper pipes, etc., and among those algorithms, SVM gave the best result of 76% accuracy [4]. They also experimented on 240 Punjabi poems of four categories with the gain ratio technique used for ranking features and algorithms like KNN, NB, SVM, and Hyperpipes (HP) are used and NB gave the best result [5]. Another research was done by Kaur and Saini with 240 Punjabi poetries but used ten different machine learning algorithms and SVM showed the best accuracy of 58.79% [6]. They also experimented on 2034 Punjabi poems with two poetic features of orthographic and phonemic on different machine learning algorithms like NB, SVM, hyper pipes, KNN, and SVM gave the highest accuracy of 71.98% on orthographic features [7]. They also experimented on two linguistic features of lexical features and syntactic features on 2034 Punjabi poems with NB, KNN, SVM, hyper pipes and SVM works best [8].

Lou et al. proposed a model for 7214 poems in English to classify into nine categories with the SVM model, and for feature extraction, they were using TF-IDF and Latent Dirichlet Allocation [9]. Computational analysis of American poems to classify poems based on stylistic features and visualizing them into clusters done by Kaplan and Blei [10]. Researchers also experimented with Malay poetry by using SVM models with Radial Basis Function (RBF) and Linear Kernel Function to classify 1500 Malay poetry and Linear Kernel gave the best performance compared to RBF kernel [11]. Hamidi et al. used SVM with RBF kernel to classify 136 Persian utterance poems and got 91% accuracy on the proposed model [12]. Prafulla and Saini proposed a model for 450 Hindi poetry classification using NB and Random Forest based on TF-IDF features and Random Forest gave a better performance than NB [13]. Researchers proposed a model for Arabic poem classification by using NB, SVM, and Linear Support Vector models, but SVM and NB got good accuracy than others [14].

Deshmukh et al. experimented with 341 Marathi poems with five categories like Friend, Prem, Bhakti, Prerna, Desh, and SVM model was used for the classification of those categories [15]. Kaushika and Patel worked on 154 Hindi poetries by using NLP techniques to classify poems with five different machine learning algorithms of SVM, NB, Decision Tree, Random Forest, and KNN, but SVM, NB, and Random Forest worked better than the other algorithms on that experiment [16].

Geetanjali et al. only worked on Rabindranath Tagore poems who is a famous poet in Bangla literature [17]. They categorized the poems as devotional, love, nature, and nationalism. They did poem classification on 1341 poems with four different categories by using the SVM model and the accuracy was 56.8%. Compared with previous research, we worked on two different categories of poems as Miscellaneous and Love poems, and we collected poems randomly, so our dataset poems are based on random poets, whereas their research was based on the great poet Rabindranath Tagore's poems only.

III. Methodology

Fig. 1 illustrates the proposed methodology that has been utilized in this research work.

A. Data Preprocessing

For data preprocessing, we used techniques like whitespace removal, punctuation marks, special symbol, full stop removal, etc., but we did not use the stop words removal technique in our research. The Bangla language has many types of whitespace marks and symbols. So, we used the Python regular expression library to remove whitespace symbols and clean the text. We also removed double whitespace, single whitespace, etc., and make the poem text into single-spaced long sentences.

Bangla language has different and many types of punctuation marks and special symbols than any other language in the world. We used the Python regular expression library to remove punctuation marks, full stop removal, and special symbols from the text and some of the symbols are [/ ' " ; : ? || | [] () ! , ... < > *].

Stop word removal is an important task in Bangla text classification research. As our research was mainly based on the Bangla text classification techniques, but we didn't use the stop words removal techniques in our research. Because Bangla poems have an interrelation between stylistic and lexical features. So, if we remove stop words, for example, অথচ, আজ, আপনার, আমাদের, মধ্যেই, যাওয়া, তুমি, etc. we break the stylistic and lexical relationship features in the poems. Moreover, we checked it through the experiments and saw the effects in the results after removing stop words from all of the poems, and that is why we didn't remove stop words from our text.

B. Machine Learning Approaches

To extract features and make a word dictionary for machine learning models we were using *CountVectorizer*

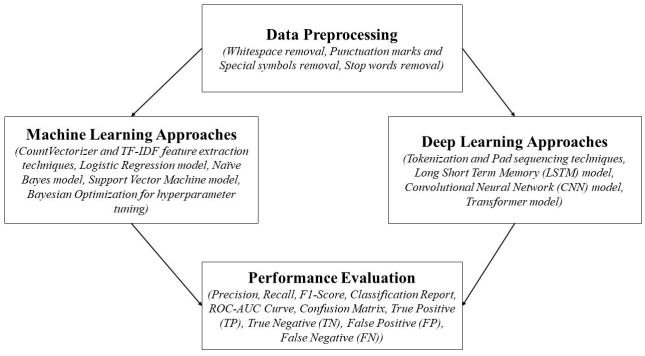


Fig. 1: Our proposed methodology of genre classification

and TF-IDF techniques in our research. *CountVectorizer* split words into tokens from a long text which is called tokenization, and after making tokenization, it encoded it as an integer number which is called vectorization. For example,

sentence = আমি তোমায় ভালোবাসি এবং তাহাকেও ভালোবাসি |

tokenization = 'আমি', 'তোমায়', 'ভালোবাসি', 'এবং', 'তাহাকেও'

vectorization = 'আমি': 0, 'তোমায়':1, 'ভালোবাসি':2, 'এবং':3, 'তাহাকেও':4

Here in the above example, the 'ভালোবাসি' word is present two times in the sentence, so CountVectorizer counts it as a single word. After doing CountVectorizer techniques we applied TF-IDF techniques in our research. TF-IDF means the Term Frequency-Inverse Document Frequency, which evaluates a word in a corpus or a collection of documents. TF or Term Frequency measures how frequently a word occurs in a document. In the target column, we did the Label Encoding by *LabelEncoder()* method and also did the One Hot Encoding by pandas get_dummies() method.

Logistic Regression is a classification-based machine learning algorithm, and in our research, we used it for a Binary Classification task. We classified Bangla poems without hyperparameter optimization and got an accuracy of 82% when we used 'liblinear' as a solver, but after hyperparameter optimization, we got a better accuracy of 84%. The NB is the most popular algorithm for text classification research. Before the Bayesian Optimization, we got 73% accuracy, but after optimization, we got a huge change in the accuracy of 86% which is the highest accuracy of the overall machine learning experiment. Before optimization, the SVM model gave us 81% accuracy, and after optimization, we got 83% accuracy. Table I, II, and III describe our three machine learning models in detail the hyperparameters we chose to optimize, the search space, and the final value after the optimization.

C. Deep Learning Approaches

In our research, we used the Keras *Tokenizer()* method for the tokenization of our dataset. 35622 unique tokens were created after the tokenization. After tokenization, we used the Pad Sequencing techniques with 'post' padding, and for maximum sequence length, we used 200. We used the Label Encoding and the One Hot Encoding in our target column same as we did in our machine learning experiment section.

LSTM is a popular model for text classification research because it tries to capture long-term dependencies between word sequences. We used Keras Sequential API to build our custom model. Fig. 2 illustrates the custom LSTM model's different layers.

For our proposed model, we used 4 different layers:

1) **Embedding Layer:** The embedding layer is the input layer in the model, where we chose Embedding dimensions = 32, input length = 200, and gave tokenized words as input.
2) **LSTM Layer1:** Here, hidden layers = 2, dropout = 0.2, activation function = 'tanh', and return_sequences=True.
3) **LSTM Layer2:** Here, hidden layers = 4, dropout = 0.4, and activation function = 'tanh'.
4) **Dense Layer:** This is the output layer of the model. Here, output class = 2, and activation function = 'sigmoid'.

As we used CNN, which is a popular model for feature extraction and Image research, but recently it is also used in the Natural Language Processing (NLP) field. We chose CNN because it has many kernels and deep layers which help to extract important features and some other important aspects. Figure 3 illustrates the custom CNN model's different layers.

So, we used 5 different layers in our proposed model:

1) **Embedding Layer:** This is the input layer of the model. Here, we chose Embedding dimensions = 32, input length = 200, and tokenized words as input.
2) **Cov1D Layer:** Here, hidden layers = 16, filters = 5, padding = 'same', and activation function = 'tanh'.
3) **MaxPooling1D Layer:** After the Conv1D layer we downsampled our data by using this layer.
4) **Flatten Layer:** We used this layer to convert the MaxPooling layer pooled feature map into a single column by using Flatten for a fully connected layer.
5) **Dense layer** This is the output layer of the model. Here, output class = 2, and activation function = 'sigmoid'.

We also used the transformer model in our experiment for classifying the Bangla poems. We utilized the transformer model architecture proposed by Google [18]. Transformer employs a self-attention technique that is appropriate for language comprehension. We used a transformer block as an attention layer and also used two

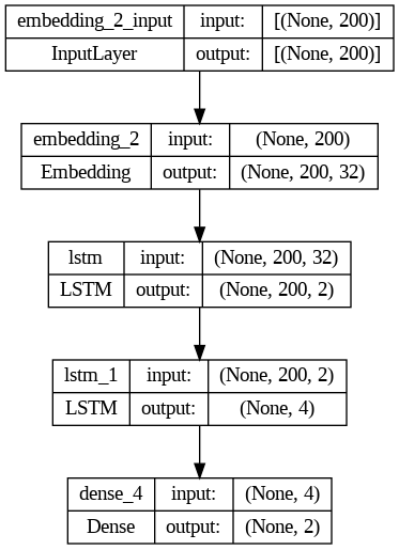


Fig. 2: Our custom LSTM model different layers diagram

Table I: Logistic Regression model hyperparameters value after optimization

Hyperparameters Name	Search Space Values	After Optimization (value)
C	range(0.5, 5)	3.49
penalties	l1, l2	l1
fit_range	True, False	True

Table II: Naïve Bayes model hyperparameters value after optimization

Hyperparameters Name	Search Space Values	After Optimization (value)
alpha	range(0.1, 1.0)	0.34
class_prior	[None, [0.1, 0.9]]	None
fit_range	True, False	False

Table III: Support Vector Machine model hyperparameters value after optimization

Hyperparameters Name	Search Space Values	After Optimization (value)
C	range(1e-6, 100)	0.75
gamma	range(0.00001, 10000)	1
kernel	'rbf', 'linear'	'linear'

embedding layers, one for tokens, and one for the token index. Fig. 4 illustrates the custom transformer model's different layers.

We used 8 different layers in our proposed model:

1) **Input Layer:** This is the input layer of the model. Here, we just input the maximum sequence length = 200
2) **Token and Position Embedding Layer:** Here, maximum sequence length = 200, number of words = 40000, embedding dimensions = 32, and these input send to embedding layers.
3) **Transformer Block:** We used transformer block to call attention layer and input embedding dimensions = 32, number of heads = 2, feed-forward network dimensions = 4.
4) **Global Average pooling 1D:** Downsampling the data.
5) **Dropout Layer_1:** Here, dropout = 0.1.
6) **Dense Layer_1:** Here, hidden layers = 4, and activation function = 'relu'.
7) **Dropout Layer_2:** Here, dropout = 0.1.
8) **Dense Layer_2:** This is the output layer of the model. Here, output class = 2, and activation function = 'softmax'

IV. Results and Discussion

As we used Bayesian Optimization techniques to optimize the hyperparameters of our three machine learning

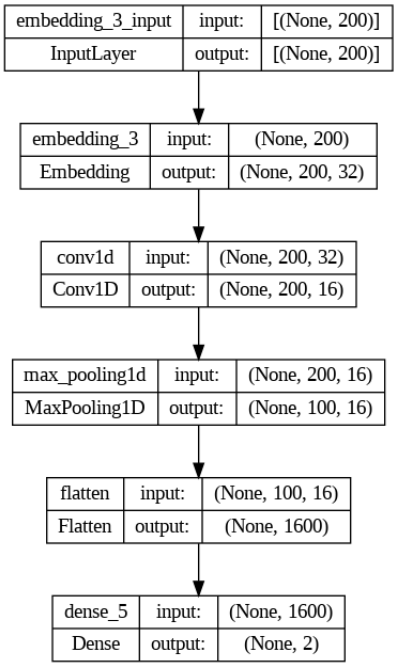


Fig. 3: Our custom CNN model's different layers

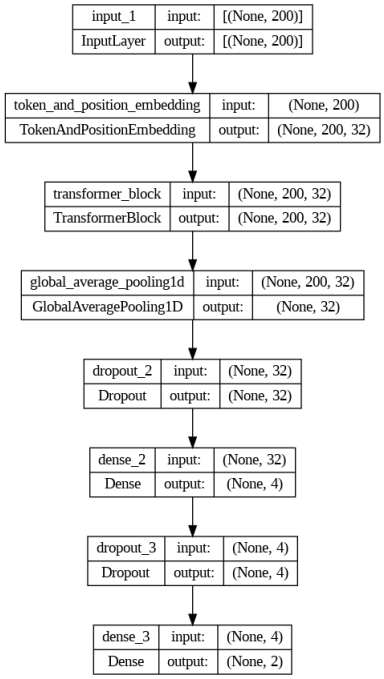


Fig. 4: Our custom transformer model's different layers

Table IV: Different machine learning models before and after optimization accuracy

Model Name	Accuracy Before Optimization	Accuracy After Optimization
Logistic Regression	82%	84%
Naïve Bayes	73%	86%
Support Vector Machine	81%	83%

models and we got some good accuracies which we showed in Table IV. The highest accuracy changes happened in the NB model, where accuracy improved from 73% to 86%, but the other two models also performed better after the optimization.

From Table V, we see the highest accuracy achieved by the NB algorithm, whereas the Logistic Regression derived 84% and the SVM 83% accuracy. The NB gave the best result because the Bayes Theorem as Bayes Theorem computes the conditional probability by using the joint probability and marginal probability of the features and possible labels. An interesting point is, NB works on the Bayes theorem and we used the Bayesian Optimization techniques which are also based on the Bayes theorem. Here, Miscellaneous poem section every algorithm's Recall values are higher than the Love poem, because of the 311 extra poems in the Miscellaneous section. As data increases False Negative (FN) decreases. On the other hand, the Love section's Precision values are higher than the Miscellaneous section, which indicates that the Miscellaneous section has low False Positive (FP) rate than the Love section. From the ROC curve in Fig. 5, we can state that the NB model performance surpassed the other two model's results. Logistic Regression performed well like the NB from the very beginning displayed in the graph. The SVM also acted well and on a single point it tried to surpass the Logistic Regression model's performance, but it performed poorly than the other two algorithms.

In our deep learning experiment for the proposed three models, We used the 'Adam' as an optimizer, and 'Binary Cross Entropy' as a loss function. We experimented on different epochs and batch size values, but finally, we got a better result at epochs = 15 and batch size = 32 and took validation split = 0.1. For the experiment, we took 80% of the data as training, for the validation we took 10% from that 80% of the training data and 20% of the data for the test purpose. Table VI showed the three deep learning models classification reports and the transformer achieved the highest accuracy of 87% over all the other models. The CNN took the second position with 86% and the LSTM achieved 81% accuracy. Here, in the Miscellaneous section, all the model's recall values are higher than the Love section which is a similar incident to the machine learning part.

The ROC curve in Fig. 6 also states that the transformer achieved the highest performance than the other two models. CNN overtook in a single point but the False Positive (FP) rate is higher than the transformer model. From the Loss graphs in Fig. 7 of all three models, we saw validation loss is higher than the training loss, because of the small amount of data in the validation phase as well as in the overall experiment. On the other hand, the accuracy graph of Fig. 8 shows that, for the high loss in the validation phase, they got rough accuracy in the validation phase but overall all the models worked quite well under this small amount of data because of the different data preprocessing steps. So, the confusion matrix of our proposed three models in figs. 9 to 11 showed that the Love section False Negative (FN) 0.34 is higher than the Miscellaneous section 0.10 for the LSTM model, but for the CNN model the False Negative (FN) percentage is lower for the Love section, and in the transformer model, the percentage becomes lower in the Love section, as because our transformer model performed better than the other two models.

Fig. 12 represents the overall experimental results of our whole research. As we worked on 6 different models from

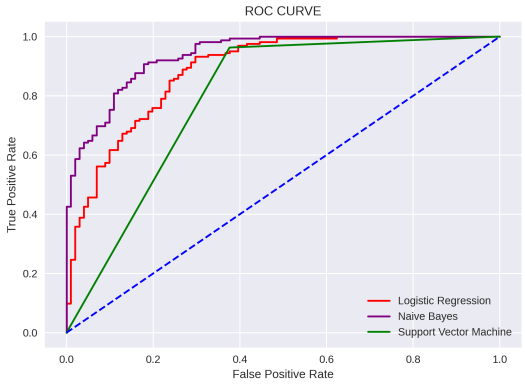


Fig. 5: ROC curve of all the machine learning models

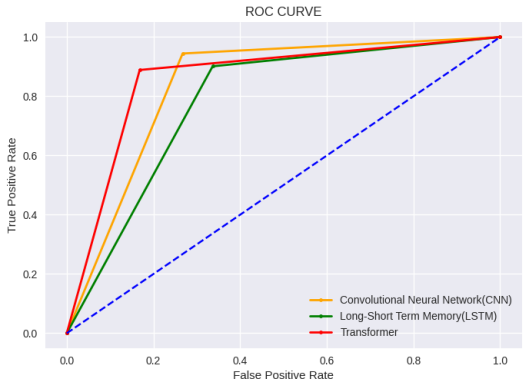


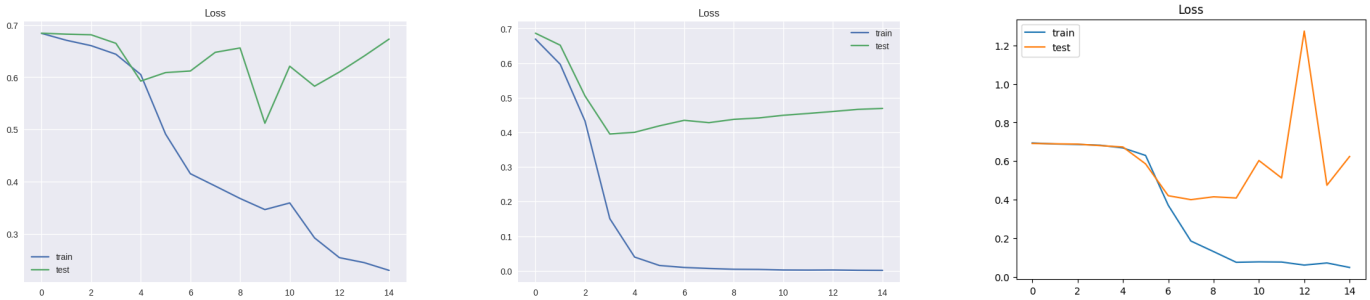
Fig. 6: ROC curve of all the deep learning models

Table V: Classification report of our machine learning experiments

Model Name	Love Poem			Miscellaneous Poem			Accuracy
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	
Logistic Regression	0.87	0.70	0.78	0.83	0.93	0.88	84%
Naïve Bayes	0.95	0.68	0.79	0.84	0.98	0.90	86%
Support Vector Machine	0.91	0.62	0.74	0.80	0.96	0.88	83%

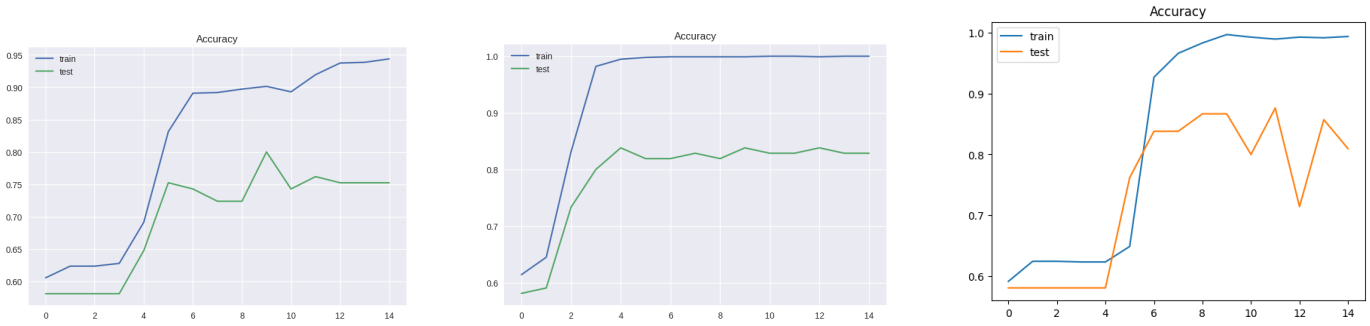
Table VI: Classification report of our deep learning experiments

Model Name	Love Poem			Miscellaneous Poem			Accuracy
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	
Long Short Term Memory (LSTM)	0.81	0.66	0.73	0.81	0.90	0.85	81%
Convolutional Neural Network (CNN)	0.89	0.73	0.80	0.85	0.94	0.89	86%
Transformer with MultiHeadAttention	0.82	0.83	0.83	0.89	0.89	0.89	87%



(a) LSTM (b) CNN (c) Transformer

Fig. 7: All the deep learning models' loss graphs



(a) LSTM (b) CNN (c) Transformer

Fig. 8: All the deep learning models' accuracy graphs

machine learning and deep learning, we got the highest accuracy from the transformer model. The Naïve Bayes model also achieved better accuracy and performance was close to the transformer model and CNN also performed close to the NB model shown in the figure. Overall, the rest of the models performed well shown in the figure.

V. Conclusion and Future Work

In this paper, we proposed for the first time Bangla poem genre classification research with machine learning and the deep learning experiment. From the overall experiment, the transformer model achieved the best performance of 87%. We tuned our hyperparameters by using the Bayesian Optimization techniques which also provided us with better performance in the machine learning section. In the deep learning part, we built our own custom models for the experiment. The limitations of our research, we experimented on a small amount of Bangla poems, and there is also an imbalance between class datasets. As we just only worked on semantic-based features, in the future, we plan to work on other text-based features. Our future goal is to collect more data so that we can make a huge dataset for future Bangla poem research.

References

[1] N. N. Ontika, M. F. Kabir, A. Islam, E. Ahmed, and M. N. Huda, "A computational approach to author identification from bengali song lyrics," in *Proceedings of International Joint Conference on Computational Intelligence: IJCCI 2018*. Springer, 2020, pp. 359–369.

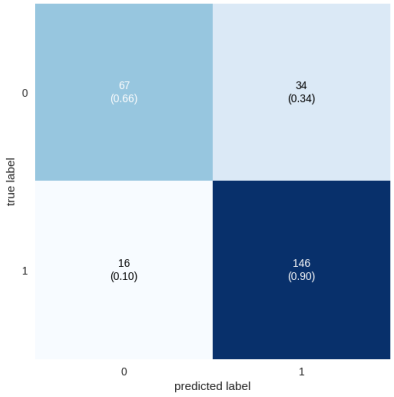


Fig. 9: LSTM model's confusion matrix

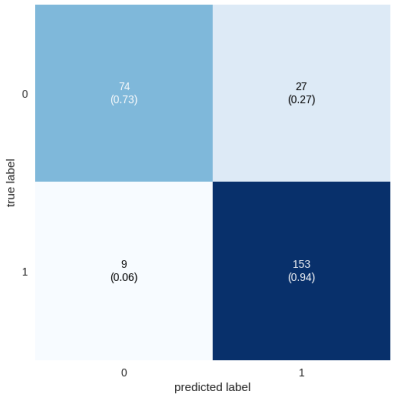


Fig. 10: CNN model's confusion matrix

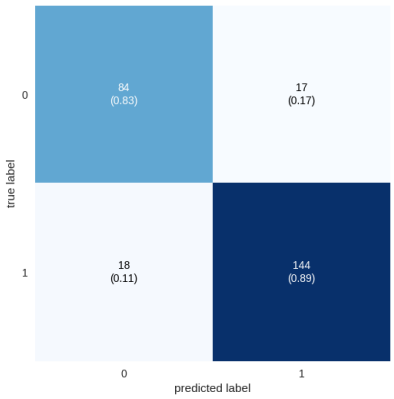


Fig. 11: Transformer model's confusion matrix

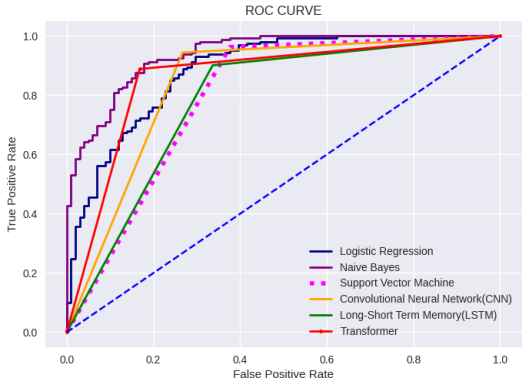


Fig. 12: ROC curve of all the machine learning and deep learning models

[2] O. Sen, M. Fuad, M. N. Islam, J. Rabbi, M. Masud, M. K. Hasan, M. A. Awal, A. Ahmed Fime, M. T. Hasan Fuad, D. Sikder, and M. A. Raihan Iftee, "Bangla natural language processing: A comprehensive analysis of classical, machine learning, and deep learning-based methods," *IEEE Access*, vol. 10, pp. 38 999–39 044, 2022.

[3] B. Kobita, "Banglar kobita," https://banglarkobita.com/, Feb. 2023.

[4] J. Kaur and J. R. Saini, "Designing punjabi poetry classifiers using machine learning and different textual features." *Int. Arab J. Inf. Technol.*, vol. 17, no. 1, pp. 38–44, 2020.

[5] J. Kaur and J. R. Saini, "Automatic punjabi poetry classification using machine learning algorithms with reduced feature set," *International Journal of Artificial Intelligence and Soft Computing*, vol. 5, no. 4, pp. 311–319, 2016.

[6] J. Kaur and J. R. Saini, "Punjabi poetry classification: the test of 10 machine learning algorithms," in *Proceedings of the 9th international conference on machine learning and computing*, 2017, pp. 1–5.

[7] J. Kaur and J. R. Saini, "Automatic classification of punjabi poetries using poetic features," *International Journal of Computational Intelligence Studies*, vol. 7, no. 2, pp. 124–137, 2018.

[8] J. Kaur and J. R. Saini, "Pupocl: Development of punjabi poetry classifier using linguistic features and weighting." *INFOCOMP: Journal of Computer Science*, vol. 16, 2017.

[9] A. Lou, D. Inkpen, and C. Tanasescu, "Multilabel subject-based classification of poetry," *Nature*, vol. 2218, pp. 30–7, 2015.

[10] D. M. Kaplan and D. M. Blei, "A computational approach to style in american poetry," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 2007, pp. 553–558.

[11] N. Jamal, M. Mohd, and S. A. Noah, "Poetry classification using support vector machines," *Journal of Computer Science*, vol. 8, no. 9, p. 1441, 2012.

[12] S. Hamidi, F. Razzazi, and M. P. Ghaemmaghami, "Automatic meter classification in persian poetries using support vector machines," in *2009 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. IEEE, 2009, pp. 563–567.

[13] P. Bafna and J. R. Saini, "Hindi poetry classification using eager supervised machine learning algorithms," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*. IEEE, 2020, pp. 175–178.

[14] M. A. Ahmed, R. A. Hasan, A. H. Ali, and M. A. Mohammed, "The classification of the modern arabic poetry using machine learning," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 5, pp. 2667–2674, 2019.

[15] R. Deshmukh, S. Kore, N. Chavan, S. Gole, and A. Kumar, "Marathi poem classification using machine learning," *International Journal of Recent Technology and Engineering*, vol. 8, no. 2, pp. 2723–2727, 2019.

[16] K. Pal and B. V. Patel, "Data classification with k-fold cross validation and holdout accuracy estimation methods with 5 different machine learning techniques," in *2020 fourth international conference on computing methodologies and communication (ICCMC)*. IEEE, 2020, pp. 83–87.

[17] G. Rakshit, A. Ghosh, P. Bhattacharyya, and G. Haffari, "Automated analysis of bangla poetry for classification and poet identification," in *Proceedings of the 12th international conference on natural language processing*, 2015, pp. 247–253.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.