EAST WEST UNIVERSITY

Project Report

Course Code: CSE400B

Course Title: Capstone Project

Project Title: Innovative Approaches for Automatic Contextual Understanding of Bengali News

Submitted By:

Name	ID
Ronjon Kar	2022-1-60-091
Tanjila Akter	2022-1-60-078
Tausif Ahmed	2022-1-60-315
Md. Touhidur Rahman Limon	2022-1-60-272

Submitted To:

Dr. Anisur Rahman
Proctor
Associate Professor
Department of CSE
East West University

Submission Date: 28 August, 2025

Contents

1 Abstract

This work offers an effective methodology for categorizing Bengali news items through the integration of Natural Language Processing (NLP) approaches with machine learning and deep learning models. We used two big datasets: (i) the Bangla Newspaper Dataset from Kaggle, which has 400,000 articles in 9 categories, and (ii) a custom dataset of 150,000 articles that we got by scraping the web from different sources and sorting them into 13 categories. We used FastText to make word embeddings after merging and preprocessing (removing stopwords, tokenizing, and balancing the classes). FastText was chosen because it works well with morphologically rich languages like Bangla. Multiple models were evaluated, including traditional machine learning algorithms (Logistic Regression, Decision Tree, Random Forest, SVM, XGBoost) and deep learning architectures (ANN, CNN, LSTM, Bi-LSTM, GRU). Experimental results show that **CNN achieved the highest accuracy of 91.93%**, followed closely by ANN (91.86%) and LSTM-based models (91%). Among machine learning methods, XGBoost performed best with 91.14% accuracy. These results show that deep learning models are better for classifying Bengali text and that FastText works well with complicated linguistic features without heavy computational cost.

2 Introduction

Text classification is an essential task in Natural Language Processing (NLP) that involves sorting text into predefined categories. Bengali newspaper classification simply refers to the process of categorizing articles according to their content into one or more predefined classes. It is vital for automating real-world applications like recommendation systems, news aggregation, sentiment analysis, and spam detection. Manually classifying documents is labor-intensive and takes a lot of time, especially with the rapid increase in online news and text content. Nowadays people are more comfortable with online newspapers. Because it is easy to find the news of someone's interested genre. People like to read content from several sources instead of just one. Over the years, researchers have created various methods for text classification, from traditional handcrafted features to modern machine learning algorithms. Common approaches include TF-IDF (4), Naïve Bayes (1), Support Vector Machines (SVM) (3), and neural network-based methods (19). Most of the existing newspaper classification algorithms are tested based on English newspaper dataset. But in Bengali the works are not that much wide. While there has been significant research in English and other widely spoken languages (4), fewer studies focus on Bengali text, even though Bengali is one of the most spoken languages worldwide. Bengali is a difficult language as Bengali is morphologically rich language and the dimension of the feature vector becomes very high while the corpus is very large. It has complex linguistic structures, idiomatic expressions, and rich morphology (18). These features make Bengali text classification a tough challenge, especially when handling long sequences, compound words, and meanings that depend on context. Recent advances in deep learning have greatly enhanced the performance of text classification tasks. Mod-

els like Long Short-Term Memory (LSTM) (11), Bi-directional LSTM (Bi-LSTM) (7), Convolutional Neural Networks (CNNs) (7), and attention mechanisms (20) have gained wide acceptance. Moreover, transformer-based models like BERT (18) and GPT (11) have shown outstanding results by capturing long-range dependencies and context more effectively than older models. Techniques like Integrated Gradients (IG) (19), SHAP (17), LIME (13), and Grad-CAM (13) help researchers identify the key features behind specific classifications, addressing the black-box issue of neural networks. In this study, we are focused to classify the Bengali newspaper classification using NLP (Natural Language Processing). This paper aims to assist Bengali online news readers by providing recommendations for pertinent news utilizing multi-category classification. In this study, we focused on classifying Bengali news from a large and diverse dataset. We used three different datasets, two of which are publicly available on Kaggle and one we created by collecting news from different Bengali newspapers. By using different sources, we are training our model with diverse data so that the model can learn different patterns for different data. We used FastText for vectorization. We trained different machine learning and deep learning models with the same dataset. Our contributions in this work are as follows:

- We create a large, sorted Bengali news dataset by merging multiple sources, which can serve as a benchmark for future research.
- We assess various machine learning and deep learning models, leveraging FastText embeddings to enhance feature representation.
- We analyze the performance of these models regarding classification accuracy and robustness across imbalanced categories, providing insights for practical applications in news recommendation and automated content organization.

The remainder of the paper is organized as follows: Section III reviews Backgroud Information in Bengali text classification and news categorization. Section IV explains the methodology, including data collection, preprocessing, features extraction, word embedding, and model explanation. Section V presents the experimental results and performance analysis. Finally, Section VI concludes the study and outlines future research directions.

3 Background Information

Nowadays, with a huge amount of news coming in the newspapers every day, different researchers have found different ways so that readers can find the news they want. They have tried their best to solve this problem. Below we have highlighted some related work patterns of different authors. We have tried to show from our perspective how they have tried to solve the problem.

In a study[1] researchers used a balanced Bengali news collection of 3,000 articles and divided them into 12 different categories (crime, economics, education, entertainment, environment, international, opinion, politics, scienceTech, sports, accident, art). 250 articles

in each class. Then they compared some machine learning and deep learning models for news classification. They achieved a maximum accuracy of 93.43% for CNN. They also achieved 91% accuracy for linear SVM and 92.60% for Bi-LSTM. Although their accuracy is good, their model is overfitted. They do not get good results for new, different types of data. They do not remove stopwords, which will create a lot of grammatical noise. Their technique will give very bad results for large datasets.

Similarly, Hossain, A et al.[2] A study only worked with titles, not the entire content. They used over 100,000 titles across 8 categories. The classes in this dataset are imbalanced. Since they only analyzed titles, it could not achieve better accuracy. The model could not understand all the patterns in the text either. They trained two deep learning models, GRU and LSTM. The accuracy of LSTM is 82.74% and GRU is 87.48%. Since they trained only two models, they cannot decide which model is best for their study. Also, they may not get better output for real-time data.

In another work, Khushbu, S. A [3] et al. trained only on headlines. For their study, they used 8,602 Bengali news articles divided into 11 categories. The data is not enough for analysis. They trained various machine learning models like SVM, Naive Bayes, Logistic Regression, Random Forest. The models achieved the worst accuracy. None of them exceeded 43%. They also used a deep learning model like neural network. Although neural network achieved 90%, it did not work well for unseen data.

Likewise, Amin, R., [4] et al. study also worked only on news titles, not full articles. They collected 88,968 Bangla news titles from 7 different categories. The dataset is unbalanced because some classes have more titles than others. They proposed a Parallel 1D CNN with word-level data augmentation and achieved 93.47% accuracy, which is better than previous models like Naïve Bayes, SVM, and MLP. However, their recall was only 90.07%, which is not good enough. Since the model only works on titles, it cannot fully understand the context of news content. They also didn't release their dataset publicly, which limits further research. In the future, they suggested using GANs for better augmentation, classifying full articles, and improving precision and recall.

Furthermore, Rahman, M. M.[5] et al. used 14,400 Bangla news articles from *Prothom Alo* divided into 6 categories. They introduced a model called Text-GCN for classification. The model achieved 96.25% accuracy, which is higher than other models like BiLSTM, GRU-LSTM, LSTM, Char-CNN, and BERT. However, their approach used a lot of memory and worked on a small dataset with limited categories. Since the dataset is small, the model may not perform well on large-scale data. In the future, they suggested using bigger datasets, hybrid models, and better preprocessing techniques.

Similarly, Yeasmin, S et al.[6] collected 12,000 Bangla news articles from *Prothom Alo* divided into 5 categories. They compared traditional machine learning, deep learning, and graph-based models. Text-GCN achieved the highest accuracy of 94.8%, surpassing CNN (92.1%) and BiLSTM (93.4%). However, the dataset comes from a single source and has limited categories, and the model requires high computational resources. In the future, they suggested using multi-source datasets, transformer models, and real-time applications.

In addition, Chowdhury, P. et al.[7] classified 14,000 Bangla news articles into 10 categories using a CNN–LSTM hybrid model with GloVe embeddings. The model achieved

87% test accuracy, outperforming SVM, LSTM, CNN, ANN, and BiLSTM. However, the model suffered from overfitting, dataset imbalance, and misclassification caused by named entities. Future directions include using larger balanced datasets, transformer models, and improved handling of named entities.

Moreover, Worked on 28,666 Bangla news articles from *Daily Ittefaq* divided into 10 categories in another study [8]. They used TF-IDF with machine learning models and found that Random Forest combined with SMOTE achieved 95% accuracy and F1-score. Balancing the dataset significantly improved performance compared to the imbalanced baseline. However, they faced Unicode processing issues, limited generalization, and high computational cost. Future work aims at using embeddings, deep learning models, and larger datasets.

On the other hand, Mouri, A. G. et al.[9] classified 136,811 Bangla news headlines into 6 categories using machine learning, deep learning, and transformer models. XLM-RoBERTa achieved the highest accuracy of 86.50%, outperforming Bangla-BERT and traditional ML/DL methods. However, the dataset was imbalanced and some categories overlapped, which affected performance. Future work suggested using balanced datasets, finer-grained classification, and advanced transformer models.

Additionally, Roy, A[10] et al. introduced a new 38-class Bengali news dataset (Dataset-1) and also used a 6-class dataset (Dataset-2) to benchmark ML and multilingual deep learning models. The best results were achieved with FastText + SVC (92.61%) for Dataset-1 and XLM-RoBERTa-Large (95.12%) for Dataset-2. Challenges included class overlap and dataset limitations. Future directions focus on expanding datasets and exploring advanced deep or hybrid architectures.

A study Ahmad, I. et al.[11] classified Bangla news into 8 categories using ML models (LR, SVM, RF, NB) and DL models (LSTM, Bi-LSTM, CNN, GRU) on datasets totaling ˜50,000 articles. The best performance was achieved with GRU + FastText (91.83% accuracy), outperforming traditional ML models. However, limited and imbalanced datasets were a challenge. Future work suggested creating larger balanced corpora, using transformer-based models, and applying data augmentation.

Building on the need for richer datasets, r Rashid, M. R. [12] introduced a new 12-class comprehensive Bangla news dataset and applied deep generative models (VAE, RBM) for feature extraction. The best result came from VAE + SVM with 94.12% accuracy, outperforming traditional ML methods like TF-IDF and Word2Vec. However, the approach had high computational cost and relied on large labeled datasets. Future work suggested using transformers with generative features, semi-supervised learning, and multimodal classification.

Similarly, to leverage pre-trained transformers, Sikder, M. F. et al.[13] fine-tuned BanglaBERT-Large on a 400k *Prothom Alo* dataset divided into 9 classes. The model achieved ≈92% accuracy on balanced data and used Integrated Gradients for explainability to highlight influential words. Baseline RNN, Bi-GRU, and Bi-LSTM+CNN models performed poorly. Limitations included class imbalance, [UNK] token issues, and using a single-source corpus. Future directions focus on improving robustness, expanding datasets, exploring multiple XAI methods, and deeper per-class evaluation.

On the other hand, Alam, S. et al.[14] used a large open-source Bangla news dataset

(SUST, 12 categories) to compare ANN, CNN, Bi-LSTM, and hybrid CNN+Bi-LSTM models with word embeddings. They achieved 88.56% accuracy for 10 categories and 84.93% for 12 categories. Limitations included a non-standardized dataset, weak Bangla stemming, and class imbalance. Future work suggested creating standardized datasets, improving Bangla NLP tools, exploring transformer models (e.g., Bangla-BERT), and expanding to more diverse text sources.

Addressing the problem of smaller datasets, Hasan, M. K [15] classified 9,913 Bengali news headlines across six categories using LSTM, Bi-LSTM, and Bi-GRU models. The Bi-LSTM model performed best with 77.91% validation accuracy. However, overfitting occurred due to the small dataset and Bengali's morphological complexity. Future work suggested using larger datasets, transformer-based models (BERT), and data augmentation for better generalization.

Taking a hybrid approach, Rana, S et al. [16] proposed NewsNet, a hybrid CNN + GRU + BiLSTM model for Bangla news classification using the Kaggle dataset (˜400k articles, 9 categories). NewsNet achieved 94.57% accuracy, outperforming all baseline ML methods (best SVM: 88.42%). Limitations included single-source data, limited categories, and higher computational cost. Future directions suggested multi-source datasets, more categories, and real-time evaluation.

Similarly, Chowdhury, O et al. [17] classified ˜30,000 Bangla news headlines into 8 categories using ML and deep learning models. GRU achieved the best accuracy at 84.01%, outperforming Bi-LSTM (83.42%), LSTM (82.74%), and classical ML baselines (˜65%). Limitations included using headline-only data, possible labeling noise, and modest dataset size. Future work recommended larger, cleaner datasets and deeper exploration of Bengali NLP challenges.

In contrast, Mugdha, S. B. S et al.[18] used a balanced 5-class subset of the BARD Bangla news dataset (75,000 articles) and applied a novel rule-based Bangla stemmer with TF-IDF + Logistic Regression. The stemmer improved feature quality, achieving 95.3% test accuracy and outperforming Multinomial NB and Random Forest. However, the study focused on classical ML, had limited categories, and used single-source data. Future work suggested benchmarking against neural models, expanding categories, and testing generalization.

Moving toward ensemble methods, Hossain, T et al.[19] proposed BanglaNewsClassifier, a hybrid stacking model (BiLSTM + SVM meta-classifier) applied to an 8-class Bangla news dataset from Kaggle. It achieved ≈94% accuracy, outperforming standalone CNN, LSTM, and ML models. The study addressed dataset standardization, benchmarking, and error analysis for Bangla NLP. Limitations included single-source data, overlapping categories, and high computational cost. Future directions focused on multi-source corpora, efficient ensembles, and improved preprocessing and interpretability.

On the classical ML side, Jakaria et al.[20] evaluated nine classical ML models for Bangla news classification using an 8-class Kaggle dataset (˜398k articles). SVM (92.76%), Bagging (92.64%), and Logistic Regression (92.26%) were the top performers using TF-IDF features. However, the study only used single-source data, did not include deep or transformer models, and faced potential domain drift. Future work suggested using modern neural models, expanding datasets, and performing cross-domain evaluation.

Expanding beyond classification, Paul, P. C. et al.[21] introduced a BERT-LDA hybrid model for unsupervised topic modeling on a new Bengali news corpus, selecting 11 topics based on coherence. BERT-LDA outperformed traditional methods (C_V = 0.92 vs. LDA 0.61, HDP 0.85, LSI 0.52). However, evaluation was limited to intrinsic metrics and only used a news corpus. Future work suggested neural topic models and human interpretability studies across diverse Bengali domains.
Beyond topic modeling, Ayman, U. et al.[22] introduced BanglaBlend, a 7,350-sentence dataset labeled for two Bangla language forms: "Saint" vs. "Common," aimed at style and variety classification. Baseline experiments with five deep-learning models showed the dataset's utility, though specific performance metrics were not reported. Limitations included binary labeling, sentence-level scope, and incomplete baseline details. Future directions suggested fine-grained labels, document-level context, and full reproducibility reporting.
Finally, addressing multi-label tasks, Sarkar, S et al.[23] introduced BanglaNewsNet, a 7,245-article dataset with 21 human-assigned labels for zero-shot multi-label classification. It benchmarked 32 models, finding that decoder LLMs like Gemini-1.5-Pro (F1 $\approx$ 0.616) outperformed classic encoders (LaBSE $\approx$ 0.40), but overall F1 scores remained modest. Limitations included zero-shot only, single-source corpus, and low performance. Future work suggested Bangla-centric pretraining, multi-domain datasets, prompt engineering, and human evaluation.

4 Methodology

This section presents a simple method for classifying Bengali news articles using Natural Language Processing (NLP) methods. This process includes some important steps: data collection, data pre-processing, word embedding, model architecture and explanation. Its details are given below:

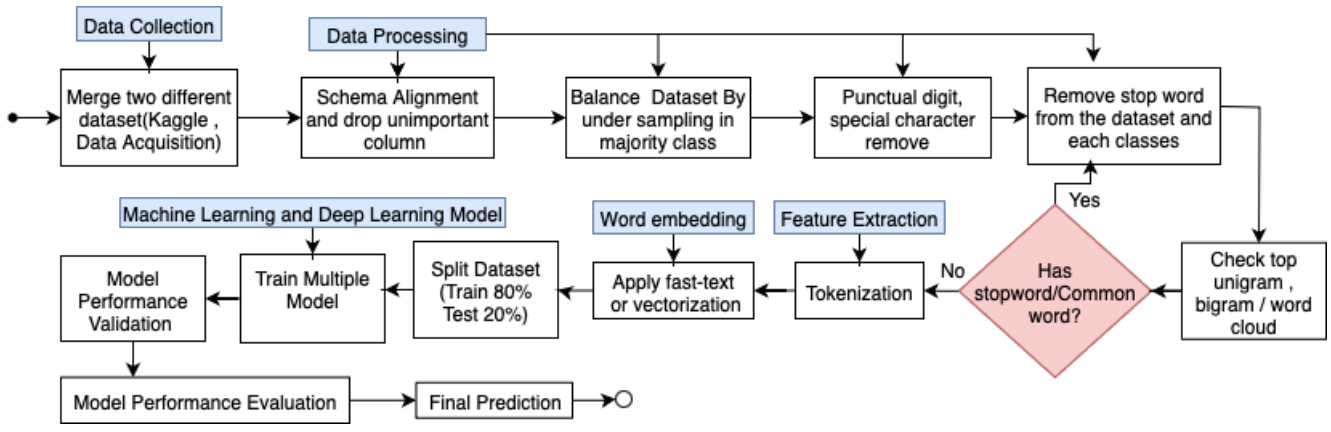


Figure 1: Methodology

4.1 Data Collection

Dataset-1: Bangla Newspaper Dataset
The Bangla Newspaper Dataset, published on Kaggle by Zabir Al Nazi, is a well-structured and richly annotated collection of 400,000 Bangla news articles sourced from *Prothom Alo*, one of the most prominent newspapers in Bangladesh. This corpus provides a clearly defined categorization framework, consisting of 9 news categories such as Bangladesh, International, Sports, Economy, Opinion, Entertainment, Education, Technology, and Lifestyle.

Dataset-2: Bengali Newspaper Dataset through Web Scarping
There are not many classes in dataset-1. We need more data of different classes to classify more different articles. So We created a dataset of Bengali newspapers through web scraping and stored the data in a CSV file. We collected about 150,000 news items from four different newspapers: *Bangladesh Pratidin*, *Bangla Vision*, *Prothom Alo*, and *Dainik Inqilab*. There were about 20 categories, but most classes did not have enough data. So, we kept the information of only 13 classes. We used web scraping to get the headline, category, publication date, full article, and source of each news item. The dataset is publicly available on Google Drive.
Here is the Drive link: https://drive.google.com/file/d/1OtPyOn-LsceeDPJI5yfK8ekVneHHkeLR/view?usp=share_link

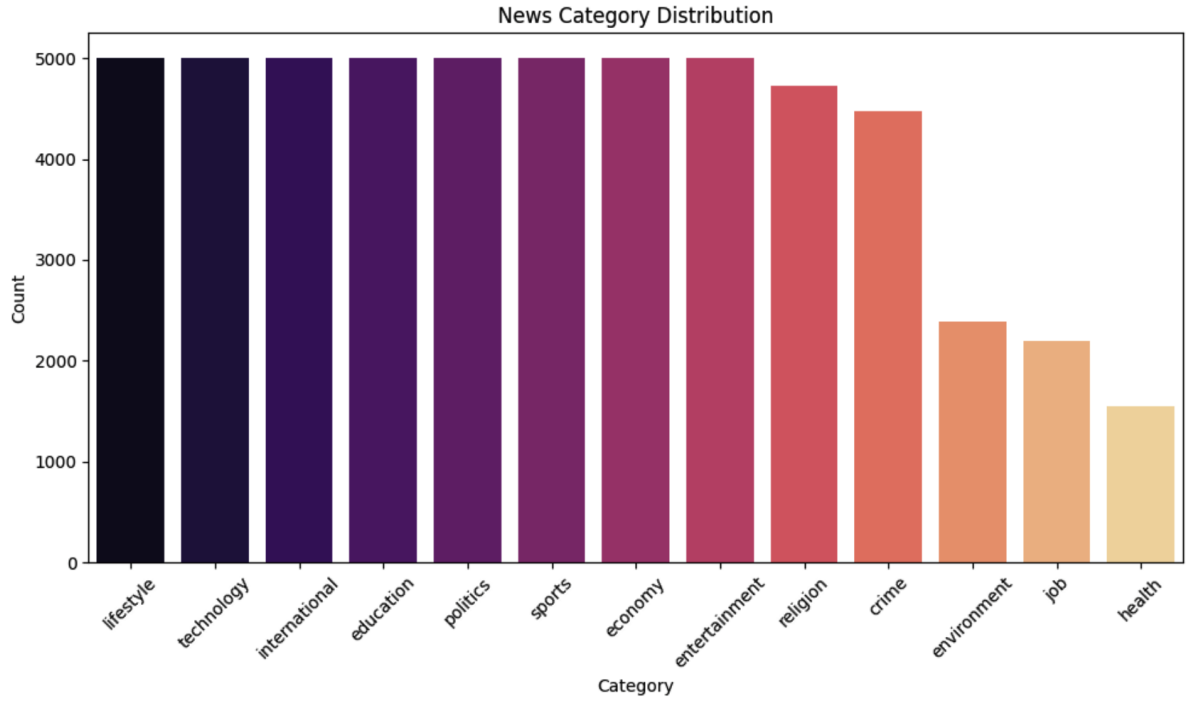


Figure 2: Bar Chart of Category Frequency

4.2 Data Preprocessing

After collecting all the data, we merged the datasets and aligned their schemas. After aligning, we removed some features that were not related to our research. Then we noticed a significant class imbalance, the Sports class had 160,000 records and the Religion class had only 5,000. To achieve balanced results, we applied undersampling. This saved 5,000 records from each class for further processing. Next, we removed common stopwords that appeared frequently in the overall dataset. Additionally, for specific classes we manually removed some stopwords that were not important to that class but might be important for others. After stopword removal, we generated word clouds and examined the top unigrams and bigrams. If any frequent words were still found we removed these. This process was repeated iteratively to ensure that the number of extra common words in the dataset was minimized at each step.

4.3 Features Extraction

Clean text: Before running a model, we always need to clean the dataset first. For our work, we cleaned the content by following these steps. First, we removed 800+ stop words from the overall dataset. Stop words are common words that do not help in classification and do not add much value to understanding the overall content. After removing stop words, we removed non-Bangla words, numbers, URLs, and some special characters from the text. This helps us find rare words related to a particular class from the text.

Tokenization: Before feeding the model, we need to extract specific words related to a particular class. These words help identify patterns in the dataset for efficient classification. To find these specific words, we explore all the records and identify certain token words. Then, we tokenize each record and store it in an array using Unicode (text representation) and the Python library 're'. The total number of tokens in the overall dataset is more than 1 crore. Using these tokens we find patterns from each class and it will be easy to classify a particular category.

4.4 Word Embedding

FastText is a library for word embedding. It was developed by Facebook AI researchers in 2016. It works with letter n-grams by breaking words into subword units. FastText is an upgraded version of Word2Vec that handles rare words more efficiently. It can handle not only English but also morphologically rich languages like German, Spanish, Japanese, Hindi, and even Bengali.
It is the fastest and most efficient version for word embedding. Word2Vec is not efficient for rare words, small misspellings, and morphologically rich languages. On the other hand, Bangla BERT can be used, but it requires a lot of time for training, which makes it very time-consuming and wastes GPU resources. Therefore, we choose FastText for word embedding because it is simple, works with Bengali text, provides fast processing, and does not require GPU resources.

(a) Crime Word Cloude

(b) Economy Word Cloud

(c) Education Word Cloud

(d) Entertainment Cloud

(e) Environment Cloud

(f) Health Word Cloud

(g) International Cloud

(h) Lifestyle Word Cloud

(i) Politics Word Cloud

(j) Religion Word Cloud

(k) Sports Word Cloud

(l) Technology Word Cloud

(m) Job Word Cloud

Figure 3: Word Cloud of 13 Classes

FastText Working Process:

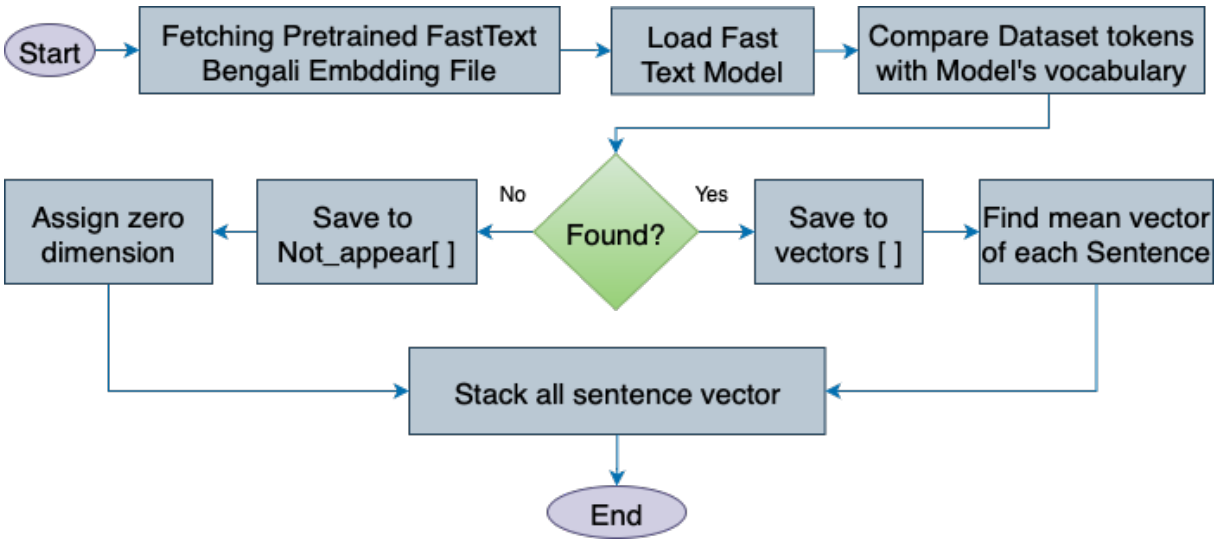


Figure 4: FastText Working Process Diagram

Step 1: Fetching the Pretrained Bangla Word Embedding File

The first step involves obtaining a pretrained word embedding file specifically trained on Bangla text. This file acts as a vocabulary dictionary where each word is represented as a vector of numerical values. These vectors capture semantic and syntactic information about words, enabling the model to understand word meanings in a mathematical form. Using a pretrained file saves time and ensures better performance since it is built from a large corpus of Bangla text.

Step 2: Loading the FastText Model

Once the embedding file is available, the FastText model is loaded into the system. Loading the model means initializing it so that it can access the pretrained vocabulary and their corresponding vectors. At this stage, the model is ready to provide vector representations for words whenever required.

Step 3: Comparing Dataset Tokens with the Model's Vocabulary

After loading the model, the dataset is tokenized into individual words (tokens). Each token is then compared with the vocabulary present in the FastText model:

- If the word exists in the model's vocabulary, its corresponding vector is retrieved and stored.
- If the word is not found in the vocabulary, it is added to a separate list of out-of-vocabulary (OOV) words for further processing.

This comparison step ensures that every word in the dataset is checked for availability in the pretrained embeddings.

Step 4: Assigning Zero Vectors to Unknown Words

For the tokens that are not found in the pretrained vocabulary, a zero vector is assigned. A zero vector is an array of zeros having the same dimensionality as the original word vectors (for example, 300 dimensions). This approach maintains uniformity in the size of vectors across the dataset, even if some words are missing from the vocabulary.

Step 5: Computing the Mean Vector for Each Sentence

After assigning vectors to all words in a sentence (including zero vectors for unknown words), the next step is to compute a single representation for the entire sentence. This is achieved by calculating the **mean vector**:

$$\text{Sentence Vector} = \frac{\sum(\text{word vectors in the sentence})}{\text{Number of words in the sentence}}$$

This process generates one vector per sentence, capturing the overall meaning of the sentence.

Step 6: Stacking All Sentence Vectors

Finally, all the sentence-level vectors are combined into a single matrix (or stack). Each row of the matrix represents one sentence, and each column corresponds to a dimension of the word embeddings. This stacked matrix becomes the input for further processing tasks such as text classification, clustering, or sentiment analysis.

4.5 Model Explanation

1. **Logistic Regression:** Logistic regression is a primary supervised machine learning algorithm used for binary classification tasks. Binary classification means where the outcome is one of two possible categories like true/false or 0/1. This method is widely used in NLP for text tokenization and sentiment analysis tasks.

$$\hat{y} = \frac{1}{1 + e^{-(\mathbf{w}\cdot\mathbf{x}+b)}} \tag{1}$$

Eq (1) refers the logistic function, where $\hat{y}$ symbolize the predicted probability, $\mathbf{w}$ is the weight vector, $\mathbf{x}$ is the feature vector, and b is the bias term.

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{2}$$

Eq (2) refers to cross-entropy loss. Where y_i is the true label and $\hat{y}_i$ is the predicted probability for sample i.

2. **Random Forest:** Random forest is an ensemble machine learning method that constructs multiple decision trees throughout the training session and outputs the class which is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It is well known for its high accuracy, ability to handle large datasets, and resistance to overfitting.

$$\hat{y} = \text{mode}\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_T\} \tag{3}$$

3. **Decision Tree (DT):** A decision tree is a supervised learning method that classifies data by recursively splitting the dataset based on feature values. At each node, the algorithm selects a split that reduces impurity, commonly measured using Gini Impurity (Eq. 4) or Entropy (Eq. 5).

$$\text{Gini}(D) = 1 - \sum_{i=1}^{C} p_i^2 \tag{4}$$

$$\text{Entropy}(D) = -\sum_{i=1}^{C} p_i \log_2(p_i) \tag{5}$$

4. **Support Vector Machine (SVM):** Support Vector Machine (SVM) is a robust classification technique that identifies the optimal hyperplane by maximizing the margin between classes. While highly effective in high-dimensional feature spaces, it can be computationally demanding. For linearly separable datasets, the optimization problem is defined as:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{6}$$

$$\text{subject to } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, \quad i = 1, ..., N \tag{7}$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2) \tag{8}$$

Here, Eq (6) is an objective function, subject to Eq (7). In addition, $\mathbf{w}$ is the weight vector, b is the bias term, $\mathbf{x}_i$ are the feature vectors, and $y_i \in \{-1, 1\}$ are the class labels. For non-linear data, SVM uses kernel functions like the RBF kernel shown in Eq (8). Here, γ is a free parameter.

5. **XGBoost (Extreme Gradient Boosting):** XGBoost is an efficient and scalable machine learning algorithm based on gradient boosting that builds an ensemble of decision trees to minimize loss with regularization for better generalization.

$$\text{obj} = \text{Loss function} + \text{Regularization term} \tag{9}$$

= loss function (e.g., logistic loss, squared error) = regularization term controlling model complexity

6. **CNN:** Convolutional Neural Networks (CNNs) are a class of deep learning architectures primarily developed for the analysis of grid-structured data, such as images.

Through the use of convolutional layers, they automatically and adaptively learn hierarchical spatial feature representations. While extensively utilized in computer vision, CNNs have also demonstrated effectiveness in natural language processing tasks.

$$y_{i,j}^{k} = \sum_{c} \sum_{m} \sum_{n} x_{i+m-1,j+n-1}^{c} \cdot w_{m,n}^{k,c} \tag{10}$$

In Eq (10), $y_{i,j}^{k}$ represents the output at position (i, j) for the k–th filter in the convolutional layer. The input value from the c-th channel of the input feature map at position $(i + m - 1, j + n - 1)$ is denoted by x, and w refers to the weight of the filter.

7. **ANN:** Artificial Neural Networks (ANNs) are computational models inspired by the structure and function of the human brain. They consist of interconnected layers of nodes (neurons) that process inputs through weighted connections and activation functions to learn complex patterns. ANNs are widely applied in both classification and regression tasks.

$$y = f\left(\sum_{i} w_i x_i + b\right) \tag{11}$$

x_i = input features, w_i = corresponding weights, b = bias term, $f(\cdot)$ = activation function (e.g., sigmoid, ReLU), y = neuron's output.

8. **LSTM:** Long Short-Term Memory (LSTM) networks are a specialized type of Recurrent Neural Network (RNN) designed to capture long-term dependencies in sequential data. They incorporate memory cells and gating mechanisms that enable the selective retention or forgetting of information across extended sequences. Due to their effectiveness in handling sequential data, LSTMs are widely applied in tasks such as language modeling and machine translation.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{12}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{13}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{14}$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{15}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{16}$$
$$h_t = o_t * \tanh(C_t) \tag{17}$$

9. **Bi-LSTM:** Bidirectional Long Short-Term Memory (Bi-LSTM) is an extension of the traditional LSTM that processes sequential data in both forward and backward directions. By combining information from past (backward pass) and future (forward pass) contexts, Bi-LSTMs capture richer dependencies, making them effective for tasks such as speech recognition, text classification, and machine translation.

$$\overrightarrow{h_t} = \text{LSTM}_{\text{forward}}(x_t, \overrightarrow{h_{t-1}}) \tag{18}$$

$$\overleftarrow{h_t} = \text{LSTM}_{\text{backward}}(x_t, \overleftarrow{h_{t+1}}) \tag{19}$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{20}$$

10. **GRU:** Gated Recurrent Units (GRUs) are a simplified variant of Long Short-Term Memory (LSTM) networks, designed to capture long-term dependencies while using fewer parameters. Unlike LSTMs, GRUs combine the cell state and hidden state into a single vector and utilize only two gates—update gate and reset gate—making them computationally more efficient while maintaining competitive performance. The internal working of GRU can be described by:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{21}$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{22}$$

$$\tilde{h}_t = \tanh(W_h \cdot [r_t * h_{t-1}, x_t] + b_h) \tag{23}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{24}$$

5 Experimental Results

5.1 Result Analysis

Table 1: Model-wise Test Accuracy

Model	**Accuracy (%)**
CNN	91.93
ANN	91.86
LSTM	91.24
Bi-LSTM	91.22
GRU	90.68
XGBoost	91.14
SVM	89.40
Random Forest	88.92
Decision Tree	71.15
Logistic Regression	88.05

Table 1 shows the performance of different models based on the dataset. Among all the models, deep learning models achieved better results than traditional machine learning models.

The **CNN (Convolutional Neural Network)** achieved the highest accuracy. CNN is a very powerful technique for learning complex patterns. In this case, CNN learned

the patterns very well and showed better results than others, achieving an accuracy of **91.93%**. Another deep learning model, **ANN (Artificial Neural Network)**, also performed very well with an accuracy of **91.86%**. Its accuracy is slightly lower than CNN. The difference in accuracy between CNN and ANN is only **0.07%**, and both are able to find very complex relationships from the dataset.

Then we have **LSTM (Long Short-Term Memory)** and **Bi-LSTM (Bidirectional Long Short-Term Memory)** models. They both achieved almost similar accuracy: LSTM achieved **91.24%** and Bi-LSTM achieved **91.22%**, which is a very good result. Both of these models can handle sequence data and remember previous data during prediction. The **GRU (Gated Recurrent Unit)** model also performed well with **90.68%** accuracy, although it is slightly lower than LSTM or Bidirectional LSTM.

Among the traditional machine learning models, only **XGBoost** has been able to achieve good accuracy, capturing better patterns than some deep learning models. The accuracy of XGBoost is **91.14%**. Among the machine learning models, XGBoost is one of the most powerful techniques as it uses the boosting technique to combine many weak learners and create a strong model.

SVM is another powerful algorithm, but its accuracy is not satisfactory for this dataset, achieving **89.40%**. The accuracy of **Logistic Regression** is **88.05%**, which is relatively good. Among the tree-based models, **Random Forest (88.92%)** achieved moderate accuracy, but **Decision Tree** performed poorly. Since Decision Tree is based on a single tree, it cannot find appropriate patterns for large datasets.

Evaluation Metrics in Machine Learning

1. Accuracy:

Accuracy measures the proportion of correctly classified instances (both positive and negative) among all instances in the dataset.

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

Where:

- TP = True Positives
- TN = True Negatives
- FP = False Positives
- FN = False Negatives

2. Precision:

Precision is the fraction of correctly predicted positive observations among all observations predicted as positive. Precision is often called the Positive Predictive Value.

$$\text{Precision} = \frac{TP}{TP + FP}$$

It answers the question: *"Of all instances predicted as positive, how many are actually positive?"*

3. Recall:

Recall is the fraction of correctly predicted positive observations among all actual positive observations. It is often called Sensitivity or True Positive Rate.

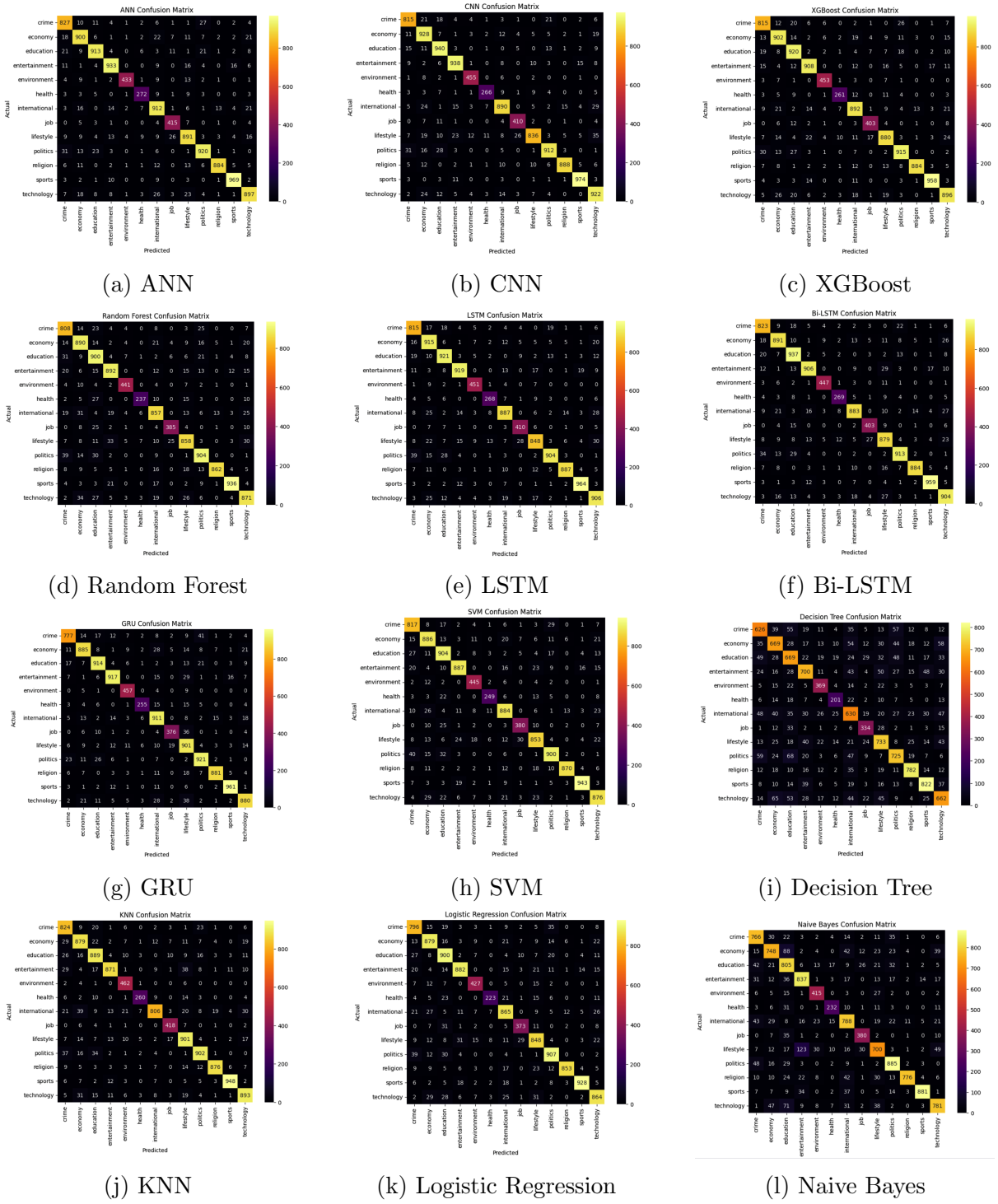
$$\text{Recall} = \frac{TP}{TP + FN}$$

It answers the question: *"Of all actual positives, how many did we correctly identify?"*

4. F1 Score:

F1 Score is the average of Precision and Recall, using a method that gives more weight to smaller values. It helps balance Precision and Recall, especially when the dataset is Imbalance.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$



(a) ANN (b) CNN (c) XGBoost

(d) Random Forest (e) LSTM (f) Bi-LSTM

(g) GRU (h) SVM (i) Decision Tree

(j) KNN (k) Logistic Regression (l) Naive Bayes

Figure 5: Confusion Matrix of All Model

5.2 Discussion

Firstly, we used only 11 classes in our study. Although we have more than 5 lakh data samples, due to class imbalance we were forced to use only 55 thousand. Moreover, our accuracy is less than 95%, so maybe we should do more analysis to get the best result. Additionally, we wanted to remove 100% stopwords for better results, but around 5% stopwords remain, which can make the model a little misleading. Also, sarcasm or idioms language are not properly handled. Finally, Bangla articles are complex, so tokenization and preprocessing may lose meaning.
The current work has some limitations. The number of news classes is limited, which affects the diversity of classification. More diverse classification can be achieved by collecting additional data. The preprocessing technique can be enhanced to improve data quality. Fast-Text word embedding also needs improvement by assigning custom vectors to out-of-vocabulary (OOV) words based on related words instead of defaulting to zero. Additionally, the model architecture can be update for better results by changing some parameters. Finally, there is no web application yet, which is needed for efficiently displaying the model results.

5.3 Experimental Setup

The following tools and technologies are used in the process of developing and implementing the Bangla news classification system:
We use Python to run our project and utilize powerful libraries such as `Pandas`, `NumPy`, `Matplotlib`, and `Seaborn` for data processing and visualization. For building Machine Learning and Deep Learning models, we use `sklearn.metrics` and `tensorflow.keras`. We also calculate performance evaluation metrics using these libraries. We use `Kaggle` Notebooks for setting up and running Python. Kaggle provides GPU support when needed.

6 Conclusion and Future Work

Initially, we collected more than one lakh data samples, but after processing, only 50 thousand could be used. Moreover, about 95% of the stopwords were manually removed from the dataset. Clear visualizations have been provided so that viewers can easily understand the data. Since stopwords were manually removed, there is less chance of losing useful context from the content. Finally, we have used multiple metrics for fair evaluation. In the future, we plan to add more classes and collect additional data to ensure more diversity. By increasing the dataset, the accuracy of the model is expected to improve. All stopwords will be removed to achieve the highest possible results. The model is planned to be expanded to classify social media posts, blogs, and other informal sources. Additionally, a website will be created so that users can easily view news categories. `Bangla BERT` will be used to achieve better results. The model will be continuously updated to handle upcoming news topics and trends.We also aim to improve processing time for faster predictions and explore more advanced preprocessing techniques to enhance data

quality. User feedback will be incorporated to continuously refine the model. Extensive documentation and tutorials will be prepared to help new users understand and use the system effectively.
Finally, preprocessing techniques will be enhanced for improved data quality, and FastText word embedding will be improved by assigning custom vectors to out-of-vocabulary (OOV) words based on related words instead of defaulting to zero. Additionally, we plan to update the model architecture by tuning parameters for better performance.

7 Reference

[1] Hossain, M. R., Sarkar, S., & Rahman, M. (2020). Different Machine Learning Based Approaches of Baseline and Deep Learning Models for Bengali News Categorization. *International Journal of Computer Applications, 176*(18), 10–16. https://doi.org/10.5120/ijca2020920107

[2] Hossain, A., Chaudhary, N., Rifad, Z. H., & Hossain, B. M. (2021). Bangla News Headline Categorization. *International Journal of Education and Management Engineering, 11*(6), 39–48. https://doi.org/10.5815/ijeme.2021.06.05

[3] Khushbu, S. A., Masum, A. K. M., Abujar, S., & Hossain, S. A. (2020). Neural Network Based Bengali News Headline Multi Classification System: Selection of Features Describes Comparative Performance. In *Proc. 11th Int. Conf. Computing, Communication and Networking Technologies (ICCCNT)*, 1–6. https://doi.org/10.1109/ICCCNT49239.2020.9225611

[4] Amin, R., Sworna, N. S., & Hossain, N. (2020). Multiclass Classification for Bangla News Tags with Parallel CNN Using Word Level Data Augmentation. In *Proc. 2020 IEEE Region 10 Symposium (TENSYMP)*, Dhaka, Bangladesh, 174–177. https://doi.org/10.1109/TENSYMP50017.2020.9230981

[5] Rahman, M. M., Khan, M. A. Z., & Biswas, A. A. (2021). Bangla News Classification using Graph Convolutional Networks. In *Proc. 2021 Int. Conf. Comput. Commun. Informatics (ICCCI)*, Coimbatore, India, 1–6. https://doi.org/10.1109/ICCCI50826.2021.9402567

[6] Yeasmin, S., Kuri, R., Rana, A. R. M. H., Uddin, A., Pathan, A. Q. M. S. U., & Riaz, H. (2021). Multi-category Bangla News Classification using Machine Learning Classifiers and Multi-layer Dense Neural Network. *Int. J. Adv. Comput. Sci. Appl., 12*(5), 757–764. https://doi.org/10.14569/IJACSA.2021.0120588

[7] Chowdhury, P., Eumi, E. M., Sarkar, O., & Ahamed, M. F. (2021). Bangla News Classification Using GloVe Vectorization, LSTM, and CNN. In *Proc. Int. Conf. Big*

Data, IoT, and Machine Learning, Singapore, 95, 723–731. https://doi.org/10.1007/978-981-16-6636-0_54

[8] Rahman, S., Mithila, S. K., Akther, A., & Alam, K. M. (2021). An Empirical Study of Machine Learning-based Bangla News Classification Methods. In *Proc. 12th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, Bangalore, India, 1–6. https://doi.org/10.1109/ICCCNT51525.2021.9579655

[9] Mouri, A. G., Talukder, P., Anik, T. R., Rahman, I. S. I., Joy, S. K. S., Shawon, M. T. R., Ahmed, F., & Mandal, N. C. (2022). An Empirical Study on Bengali News Headline Categorization Leveraging Different Machine Learning Techniques. In *Proc. 25th Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dhaka, Bangladesh, 312–317.

[10] Roy, A., Sarkar, K., & Mandal, C. K. (2023). Bengali Text Classification: A New multi-class Dataset and Performance Evaluation of Machine Learning and Deep Learning Models. ResearchGate. https://doi.org/10.21203/rs.3.rs-3129157/v1

[11] Ahmad, I., AlQurashi, F., & Mehmood, R. (2022). Machine and deep learning methods with manual and automatic labelling for news classification in Bangla language. arXiv. https://doi.org/10.48550/arXiv.2210.10903

[12] Ur Rashid, M. R., Azam, S., & Jonkman, M. (2023). Feature extraction using deep generative models for Bangla text classification on a new comprehensive dataset. arXiv preprint arXiv:2308.13545. https://arxiv.org/abs/2308.13545

[13] Sikder, M. F., Ferdous, M., Afroz, S., Podder, U., Fatema, K., Hossain, M. N., Hasan, M. T., & Baowaly, M. K. (2023). Explainable Bengali Multiclass News Classification. *IEEE Access, 11*, 12345–12356. https://doi.org/10.1109/ACCESS.2023.10441218

[14] Alam, S., Haque, M. A. U., & Rahman, A. (2023). Bengali Text Categorization Based on Deep Hybrid CNN–LSTM Network with Word Embedding. *IEEE Access, 11*, 98765–98775. https://doi.org/10.1109/ACCESS.2023.9775913

[15] Hasan, M. K., Islam, S. A., Ejaz, M. S., Alam, M. M., Mahmud, N., & Rafin, T. A. (2023). Classifying Bengali Newspaper Headlines with Advanced Deep Learning Models: LSTM, Bi-LSTM, and Bi-GRU Approaches. *Asian J. Res. Comput. Sci., 16*(4), 372–388. https://doi.org/10.9734/ajrcos/2023/v16i4398

[16] Rana, S., Haque, M. I., Sultana, N., Amid, A. F., Hosen, M. J., & Islam, S. (2024). NewsNet: A Comprehensive Neural Network Hybrid Model for Efficient Bangla News Categorization. In *Proc. 15th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, Mandi, India, 522–527. https://doi.org/10.1109/ICCCNT61001.2024.10725173

[17] Chowdhury, O., Ahmed, M., Ara, M. T., Reno, S., & Alam, A. (2023). Bengali News Headline Categorization: A Comprehensive Analysis of Machine Learning and Deep Learning Approach. *BAIUST Academic Journal, 4*(1), 26–44. https://doi.org/10.63307/BAJ.4.1.S3

[18] Mugdha, S. B. S., Khan, Z. H., Uddin, M., & Ahmed, A. (2024). Accurate Prediction of Bangla Text Article Categorization by Utilizing Novel Bangla Stemmer. *Int. J. Autom. Smart Technol., 14*(1), 1–7. https://doi.org/10.5875/xbzrk013

[19] Hossain, T., Islam, A.-R., Mehedi, M. H. K., Rasel, A. A., Abdullah-Al-Wadud, M., & Uddin, J. (2025). BanglaNewsClassifier: A machine learning approach for news classification in Bangla Newspapers using hybrid stacking classifiers. *PLoS One, 20*(6), e0321291. https://doi.org/10.1371/journal.pone.0321291

[20] Jakaria, A. J. M., Roy Chowdhury, R. R., Konia, J. J., Roy, D., & Meem, N. T. A. (2025). A Comparative Study on different Machine Learning Approaches for Categorizing Bangla Documents. *International Journal of Computer Applications, 186*(61), 32–39. https://doi.org/10.5120/ijca2025924391

[21] Paul, P. C., Rahman, M., Begum, A., Ahmed, M. T., Chakraborty, D., & Rahman, M. S. (2025). Combining BERT with LDA: Improved Topic Modeling in Bengali Language. *IAENG International Journal of Computer Science, 52*(2), 383–393.

[22] Ayman, U., Saha, C., Rahat, A. M., & Khushbu, S. A. (2024). BanglaBlend: A large-scale novel dataset of Bangla sentences categorized by saint and common form of Bangla language. *Data in Brief, 58*, 111240. https://doi.org/10.1016/j.dib.2024.111240

[23] Sarkar, S., Hasan, M. N., & Karmaker, S. (2025). Zero-Shot Multi-Label Classification of Bangla Documents: Large Decoders Vs. Classic Encoders. arXiv preprint. https://arxiv.org/pdf/2503.02993