Bangla News Document Categorization using Deep Learning Approaches and Fine-tuned BERT

Muhammad Anwarul Azim
Department of Computer Science and Engineering
University of Chittagong
Chittagong-4331, Bangladesh

Md Gias Uddin
Department of Computer Science and Engineering
University of Chittagong
Chittagong-4331, Bangladesh

Mohammad Khairul Islam
Department of Computer Science and Engineering
University of Chittagong
Chittagong-4331, Bangladesh

Abu Nowshed Chy
Department of Computer Science and Engineering
University of Chittagong
Chittagong-4331, Bangladesh

ABSTRACT
With the explosive growth of text documents available in digital form, document categorization has become a critical challenge in managing digital data effectively and precisely. So, researchers apply supervised, semi-supervised, and unsupervised approaches to categorize text documents. Recently, Transformers-based models show outstanding results in the downstream tasks of natural language processing, such as text classification, sentiment analysis, emotion classification, name entity recognition, spam email detection, etc. As the Bangla language is a widely spoken language, we deploy deep neural networks based CBiLSTM, BiLSTM, FastText, and Transformer-based BERT classifier models to categorize Bangla news documents into predefined categories. We utilize pre-trained fastText word embedding vectors with CBiLSTM, BiLSTM classifier models. The dataset we used has 28,800 news documents with 12 categories. In order to find the best outcome, we fine-tune each model with different hyperparameters. Fine-tuned BERT classifier model manages to achieve the highest accuracy of 94.74% compared to other classifier models. We also compare the accuracy of different classifier models with respect to Bangla news documents.

Keywords
Bangla News Document Categorization; Text Classification; CBiLSTM; BiLSTM; Word embedding; fastText; Fine Tuned BERT

1. INTRODUCTION
Document categorization, which is a classification problem by definition, is an interesting research area in natural language processing (NLP). In the fields of data mining, database, information retrieval, and computer science, document categorization, or text classification, has grown as a popular research area. It's a well-known method for finding, analyzing, comprehending, meaning extracting, and explaining information from text-based data. The process of classifying text documents into prespecified categories or labels based on their textual information and extracted characteristics is regarded as document categorization or text classification. Natural language processing-based document categorization allows one to filter through large amounts of documents without having read them. One may track the sort of document being uploaded and downloaded by assigning it a category, making future retrieval simple and quick. By categorizing documents, we can find documents connected to a single record by searching for specific criteria. Furthermore, the documents to be categorized for interest can be collected from a variety of sources, including news agencies, public websites, social media, government reports, journals, etc. Text documents include emails, newspapers, blogs, newsgroup messages, publications, reports, web pages, and other documents. The documents may be found in a variety of natural languages, have varying lengths and structures, and be produced by a range of authors in a variety of writing styles. Document categorization is particularly handy for publishers, news sites, blogs, and anyone else who manages a large amount of content. Document categorization can be accomplished manually or algorithmically. On the other hand, NLP mechanisms enable computers to derive meaningful information more intelligently, as well as evaluate and identify the meaning of such texts.

With the continuous advancement of the internet and the availability of digital devices, the amount of text materials available on the internet, such as digital libraries, news sources, and workplace intranets, is growing quite rapidly. So, document categorization has a great importance to manage, organize, and analyze huge volumes of data. The application of document categorization includes news categorization, sentiment analysis, emotion classification, spam email detection, fake news detection, name entity recognition, etc. To address these types of problems, many studies have contributed to document categorization using machine learning [1-3] and deep learning approaches [4-6]. Recently, the Transformer based language model has demonstrated remarkable performance in document categorization in the context of classification challenges in NLP [7-9].

The Bangla language has a large number of words in the dictionary that are used in a variety of ways. It's also a word order language with a lot of flexibility and the second-highest number of speakers in the Indian subcontinent. It is spoken by over 210 million people. It is Bangladesh's official language as well as one of the Indian constitution's recognized languages [10]. In the course of time, the portion of Bangla textual information on the internet is growing fast, and it has become a hot research area for many researchers. In the latest research, various machine learning, deep learning approaches have been introduced for Bangla document classification.

In this paper, we collect a publicly available news dataset and preprocess it before feeding it into the classifier models. We

use deep learning approaches and the transformer-based BERT model to categorize Bangla news documents.

In this research work, we make the following contributions regarding Bangla news categorization

- We make a new dataset from a publicly available dataset.
- We fine tune different hyperparameters of Convolutional Bidirectional long short-term memory (BiLSTM), BiLSTM, FastText, and Fine-tuned BERT classifiers to investigate how hyperparameters impact on classifier performance for categorization.
- We explore and observe that the performance of the fine-tuned BERT model outperforms the other classifiers.

We got the highest result from fine-tuned BERT and then the second highest result from the fastText classifier. However, BiLSTM showed a higher accuracy than CBiLSTM.

The remainder of the paper is organized as follows: Section 2 describes the theoretical background of the work; Section 3 refers to some early works of document categorization in Bangla languages; Section 4 casts light on the overall view of the proposed work; Section 5 discusses the analysis of experimental setup and results, and Section 6 presents a short conclusion with some remarks on future works.

2. THEORETICAL BACKGROUND

This section goes over the theory behind the document categorization models we utilize.

2.1 fastText

fastText is an open-source library for efficient learning of word representations and also performing robust, fast, and accurate sentence classification. It converts text into continuous vectors that can be used for a variety of language-related tasks. Unlike many word embeddings, fastText embeds words by considering each word as being composed of character n-grams rather than a word whole. This capability allows it not only to learn rare words but also out-of-vocabulary words [11].

For text categorization, the document's words are organized into word vectors. By averaging all word vectors, the model's first step is to create a document vector. The number of times a word appears in the document is used to calculate the average for each word vector. The document vector given as the hidden layer's input is multiplied by the hidden layer's linear matrix to produce a classification vector. Finally, to do classification, the softmax function is used to calculate the class label probabilities [12].

2.2 BiLSTM

The recurrent neural network (RNN) is a sequence learning model that operates on the premise of saving a layer's output and feeding it back into the input to predict the layer's output. While training in deep stages, traditional RNN suffers from vanishing and exploding gradient problems. LSTM is an improved version of RNN that solves these problems by introducing an input gate, an output gate, a forget gate, and a memory cell. Using these mechanisms, LSTM can learn longer-term dependencies. BiLSTM is an adaptation of LSTM that consists of two LSTMs. LSTM looks at a sentence from only one direction, whereas BiLSTM looks at a sentence from both directions. One LSTM takes the input in a forward direction, and the other in a backward direction. This technique helps the model to fully learn the relationship between words in a sentence [13] and at any point in time, bidirectional LSTM can preserve information from both past and future by utilizing forward and backward LSTM. So Bi-Directional LSTM is particularly effective in the downstream tasks of NLP.

2.3 Convolutional BiLSTM

The convolutional neural network (CNN) and the BiLSTM network are combined to form the Convolutional BiLSTM (CBiLSTM) network. CNN is a powerful deep neural network that was initially designed for image processing, but CNN shows outstanding results in the field of NLP tasks. In NLP tasks, one-dimensional CNN is used. CNN can extract higher-level sequences of word features from raw text but is unable to accomplish sequential correlation. The main objective of integrating these two models is to generate a model that takes advantage of both CNN and BiLSTM's capabilities, capturing CNN's extracted features and using them as an LSTM input.

2.4 BERT

BERT [14] stands for Bidirectional Encoder Representations from Transformers that can be used for a wide variety of NLP tasks. BERT is based on the transformer architecture [6]. Using the transformer's multi-layer encode module, BERT is able to jointly utilize both left and right contexts across all layers to pre-train its bidirectional representations. Since the pre-trained BERT model was trained on generic corpora, fine-tuning is required for the downstream NLP tasks. The BERT model is initiated using the pretrained parameters, and all of the parameters are fine-tuned using labeled data for specific tasks such as classification, sentence pair classification, spam email detection, sentiment analysis, and question answering tasks. Although the downstream tasks are all initialized with the same pre-trained parameters, they each have their own fine-tuned models.

The encoder of the BERT-base model has a hidden size of 768, 12 transformer blocks, and 12 self-attention heads. BERT generates a representation of the sequence from an input sequence of up to 512 tokens. One or two segments make up the sequence. The first token of the sequence is always [CLS], which contains the special classification embedding, and another unique token [SEP] is used to divide segments.

There are only a few modifications made to the BERT-base configuration when it is fine-tuned for the downstream text classification task. The entire sequence is represented by BERT using the final hidden state h of the first token [CLS]. BERT is topped with a basic softmax classifier to predict the probability of a class label. BERT's fine-tuning on single sequence classification tasks is shown in Figure 1.

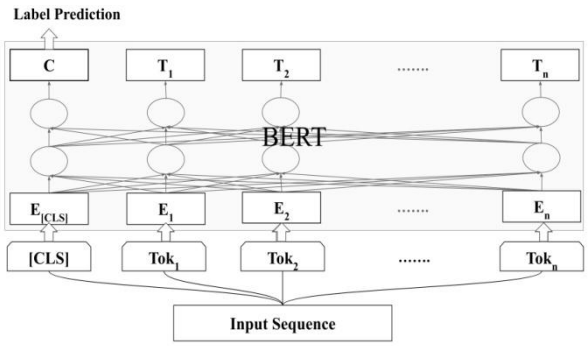

Figure 1: Architecture of the Fine-tuned BERT Model

3. LITERATURE REVIEW

In prior studies, researchers demonstrated a variety of methods to categorize text documents. Despite the fact that numerous research works have been accomplished in English, China, Arabic languages, etc., very few studies have been done in Bangla language, mostly due to the lack of sufficient data and enough technical tools.

In previous studies, different machine learning and deep learning approaches have been used to categorize Bangla text documents.

Chy et al. [15] proposed a Naive Bayes classifier to classify Bangla news documents based on the news code of IPTC. To extract news stories from web pages, they built their own crawler. They also proposed a lightweight stemmer for feature extraction.

Mostakim et al. [16] used TF-IDF to extract textual features and deployed K-Nearest Neighbor, Random Forest, SVM with linear and radial basis kernel, Gaussian NB, and logistic regression classifiers to categorize Bangla documents into 5 predefined classes. They have generated a massive Bangla content dataset that is open to the public for use.

Kabir et al. [17] used a Stochastic Gradient Descent (SGD) classifier with TF-IDF feature extraction techniques for Bangla document categorization. The dataset they used contains 9127 documents with 9 categories. They obtained the highest accuracy compared to other investigated machine learning approaches, such as SVM, Ridge Classifier, Passive-Aggressive, Naive Bayes, and Logistic Regression.

Mahmud et al. [18] used an SVM classifier along with TF-IDF mixed with a term frequency threshold feature selection technique for Bangla document categorization. The used dataset has more than 300,000 text documents with 12 predefined categories. They improved the classifier's performance by greatly reducing feature space and execution time.

Islam et al. [19] used the normalized TF-IDF feature extraction method and Chi-Square distribution feature selection technique with word analyzer and 3 supervised machine learning algorithms Naïve Bayes (NB), Stochastic Gradient Descent (SGD), and Support Vector Machine (SVM) classifier for classifying Bangla text documents into 12 predetermined categories. Besides, they studied feature importance on classifier model and classifier's efficiency.

In another work [20], they applied a TF-IDF weighting feature selection technique with length normalization with a Support Vector Machine classifier with a linear kernel for categorizing Bangla text documents into 12 predefined classes. The proposed method outperformed all other existing classifier methods.

Dhar et al. [21] applied popular supervised machine learning techniques like Naive Bayes, Random Forest, MultiNomial, and Naive Bayes with the TF-IDF feature extraction technique for Bangla document categorization into 5 predefined categories. They utilized a feature dimensionality reduction technique, only 40% of the TF scores were considered in the experiment. The LIBLINEAR model with a 40% TF approach achieved the highest accuracy compared to other models.

In their other work [22], they used Multi Layer Perceptron, Naive Bayes, Multi Naive Bayes, Decision Tree, PART, and K Nearest Neighbor with TF-IDF-ICF feature extraction technique for classifying Bangla news documents into 8 predefined classes. They showed TF-IDF with Inverse Class Frequency improved classifier performance.

Mahmud et al. [23] used a Multi-Layer Perceptron classifier with TF-IDF Feature extraction techniques for Bangla text document classification. Their dataset consists of 31908 text documents with 12 categories. They also showed an optimization method for handling large feature spaces in neural networks and were able to obtain the highest accuracy.

Hossain et al. [24] proposed a Deep Convolutional Neural Network classifier with a Word2Vec word embedding semantic feature extraction technique for Bangla automatic document categorization. The dataset has about 86,199 text documents with 12 categories. They achieved competitive results compared to existing methods.

Mojumder et al. [25] employed deep learning approaches such as Convolutional Neural Network (CNN), Bidirectional LSTM, and Convolutional BiLSTM with fastText word embedding feature extraction technique for categorizing Bangla news documents into 12 predefined categories. The dataset has 113082 text documents. They achieved state of art results using that proposed method.

Rahman et al. [26] implemented Convolutional Neural Networks and LSTM deep learning networks for automatic text document categorization. They used character level tokenization instead of word level tokenization and utilized 3 datasets named BARD, Prothom Alo, and OSBC for the categorization of Bangla text documents.

Amin et al. [27] generated Bangla word embeddings can be helped to build word clusters that represent the semantic relationship of words from the context. They used the clustering words as characteristics in the categorization of Bangla text documents. They used a Support Vector Machine classifier for the categorization task.

Alam et al. [28] applied Logistic Regression, Neural Network, Naive Bayes, Random Forest, and Adaboost classifiers for Bangla news document classification, and for feature extraction, they used TF-IDF, Word2Vec word embedding approaches with those classifiers. They also proposed a publicly available dataset named the BARD dataset that has 3,762,226 news documents with 5 classes.

Khushbu et al. [29] applied recurrent neural network (RNN) and machine learning approaches such as Support Vector Machine (SVM), Naive Bayes, Logistic Regression, and Random Forest classifiers for classifying news documents into 11 predefined categories. The dataset has 80000 news documents. They obtained the highest accuracy with the deep learning approach RNN compared to machine learning approaches.

Tudu et al. [30] deployed supervised machine learning approaches such as Support Vector Machine (SVM), MultiNomial Naive Bayes (MNB), stochastic Gradient Descent (SGD), and Logistic Regression (LR) to classify Bangla news documents. The dataset contains around 84, 906 news documents of 10 categories. They also reported issues, challenges and analyzed the performance of those classifiers.

Recently, some researchers have tried transformer-based classifier models for classification.

Alam et al. [31] fine-tuned Multi-lingual Transformer BERT model for Bangla text classification such as sentiment analysis, emotion detection, news categorization, and

authorship attribution. They used six different domain-specific datasets.

Rahman et al. [32] used two transformer-based models such as Multilingual BERT and ELECTRA BERT and fine-tuned each model with different hyperparameter settings. They used 6 different datasets, including BARD, OSBC, Prothom Alo, Youtube comment dataset, news comment sentiment dataset, and news classification dataset for domain specific tasks.

For the technical domain classification of Bangla text, Ghosh et al. [33] used a pre-trained Bangla BERT (Bangla-Bert-Base) language model and compared performance with other Bangla BERT versions like ELECTRA, Indic BERT, and mBert.

4. METHODOLOGY

This section illustrates the details of our approach for Bangla news document categorization. Figure 2 presents the workflow diagram of our approaches. Initially, we collected data from an open-source data repository. We prepared the data through preprocessing for the different classifier models. Following that, we fine-tuned the classifier models with different hyperparameters with our collected data. Finally, we assessed the models by utilizing various metrics.

4.1 Dataset Collection

We have used the publicly available Bangla dataset corpus [34]. All of these news documents were collected from several online Bangla newspapers, including prothom-alo.com, bdnews24.com, dailykalerkantha.com, etc. Each news document belongs to a particular category depending on its contents. The corpus has a total of 102097 news documents with 12 categories such as Accident, Art, Crime, Economics, Education, Entertainment, Environment, International, Opinion, Politics, Science & Tech, and Sports.

Considering the balanced dataset and resource issues, we have taken 2400 documents from each category. After that, our new dataset has a total of 28880 news documents. Table 1 represents the distribution of documents per category.

4.2 Dataset Preprocessing

Preprocessing is an important step for enhancing model performance and making raw data models understandable and compatible. Besides relevant information, raw data contains noise and irrelevant information such as punctuation, digits, special symbols, characters, etc. Preprocessing is required before feeding raw data to the model. Different steps are involved in preprocessing, such as tokenization, punctuation

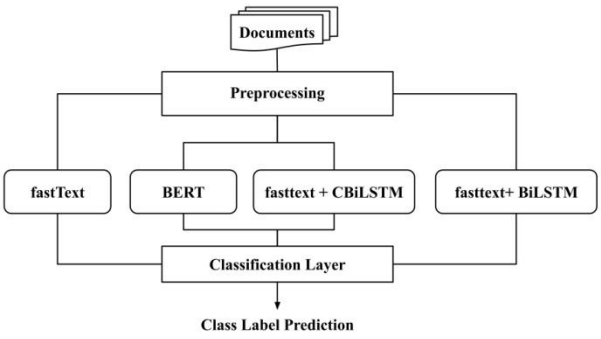

Figure 2: Overview of the research work

and emoticon removal, stemming, stop-word removal, label encoding, etc.

Table 1: Documents' distribution per category

Category	No. of Docs.	Category	No.of Docs.
Accident	2400	Environment	2400
Art	2400	International	2400
Crime	2400	Opinion	2400
Economics	2400	Politics	2400
Education	2400	Science & Tech	2400
Entertainment	2400	Sports.	2400

In our preprocessing stage, we followed various significant steps to make raw data understandable to the classifier models which are described below:

Duplicates and Punctuation Mark Removal: we remove all duplicate values and punctuation marks (.,|';! "!"-:;?_`~,। , etc.), Bangla as well as English digits (English Digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9; The corresponding Bangla Digits : ০ , ১ , ২ , ৩, ৪ , ৫ , ৬, ৭ , ৮, and ৯), some special symbols (#$%&()*+/<=>@[\\]^{|}~), emoticons and unnecessary newlines (\n), white spaces, tabs (\t), etc.

Stop Word: We don't remove stop-words as it causes loss of structural information and also makes context understanding difficult for the model. The experimental result justifies this not removing the stop words policy. In other words, if the stop-words are removed, the context is not represented enough to produce the good result.

Tokenization: Tokenization is the way of breaking up a given sentence or text into meaningful units called tokens. After cleaning documents, each word in the news document is tokenized.

For example, the sentence,

আমিবাংলায়কথাবলি

After tokenization:

'আমি', 'বাংলায়', 'কথা', 'বলি'

The main object of the tokenization class is to represent every word in such a way that the machine can map them to numerical values.

Padding: Before feeding to deep neural networks and BERT models, sequence length must be equal. Only the first 350 tokens were considered in our experiments, except for the fastText classifier. So, sequences that were too long were truncated, and sequences that were too short were padded with zeros.

Label Encoding: We use the Sklearn LabelEncoder() function to transform our non-numerical categories to numerical labels. For example, accident-0, art-1, crime-2, economics-3, etc.

Dataset Split: After preprocessing, we divide our dataset into two parts, 80% for training and 20% for testing purposes. We use 20% of training data as a validation set

Table 2: The distribution of dataset

Model	#Train	#Dev	#Test	Total
CBiLSTM	18432	4608	5760	28800
BiLSTM	18432	4608	5760	28800

Fine-tuned BERT	18432	4608	5760	28800
fastText	23040		5760	28800

For the fastText classifier, we only divide the dataset into two parts, 80% for training and 20% for testing purposes. The distribution of the dataset is described in Table 2.

4.3 Model Architecture

4.3.1 fastText

Word representation vectors are utilized as input into the input layer. Words in documents are represented by n-dimensional word vectors. By averaging all word vectors, the classifier model generates document vectors. A linear bag of the document's words makes up the vector. Each word vector is averaged the same number of times as the number of times it appears in the document. The hidden layer receives the input and multiplies it by a matrix to generate the classification vector. Finally, the document classes are determined using the loss function, which allows for the construction of probability distributions across preset class labels. Different hyperparameters are used during training such as learning rate, wordNgrams etc. the hyperparameters and their values are described in Table 3. The remainder of the hyperparameters are left at their default values.

Table 3: Hyperparameter settings for fastText

Hyperparameter Settings	Value
Learning rate	[0.1, 0.2, …, 0.9, 1.0]
WordNgrams	[1, 2, 3, 4, 5]
Epoch	[5, 10, ..., 45, 50]

4.3.2 BiLSTM

To start the embedding layer, we utilize 300-dimensional fasttext word embedding pretrained vectors, and we generate embeddings for our dataset (vocabulary size=297130). As we consider the first 350 tokens in a sequence, the model's embedding layer receives an input document as a series of 350 tokens. The embedding layer transforms the token into 300-dimensional weighted token vectors. To extract higher-dimensional features from the embedding layer, we employ bidirectional LSTM. The weighted token vectors are used as the inputs of the BiLSTM model. We used different hidden units and recurrent dropouts to overcome overfitting in the BiLSTM Layer.

Table 4: Hyperparameter settings for BiLSTM

Hyperparameter Settings	**Value**
No. of BiLSTM layer	[1, 2]
Hidden unit	[128, 256, 512]
Dropout	[0.1, 0.2, 0.3, 0.4]
No. of Dense layer	[3, 4]
Final activation function	[softmax, softplus]
Optimizer	[adam, adamax, Rmsprop]
Batch size	[64, 128, 256]
Epoch	[10, 15, …, 145, 150]

The high dimensional feature of BiLSTM's output is fed into the dense layer, which consists of a different hidden unit and a relu activation function. We converted the dense layer result into one dimensional vector. Finally, the output layer, which is the last layer with multiple activation functions, generates a probability distribution that results in 12 different classes. We utilized different hyperparameters such as activation function, dropout, hidden unit, recurrent dropout, batch size, epoch, etc. The different hyperparameters values are described in Table 4. Our model was optimized using the sparse categorical cross-entropy loss function and the adam, adamax, and RMSprop optimizers with different batch sizes and epoch numbers. The remainder of the hyperparameters are left at their default values.

4.3.3 CBiLSTM

Like the BiLSTM Model, we begin the embedding layer by generating embeddings for our dataset with a vocabulary size of 297130 using 300-dimensional fastText word embedding pretrained vectors. These weighted word vectors were input into the CNN layer, which extracted a series of higher-level phrase representations. Different filter sizes, kernel sizes, and activation functions make up the CNN(Conv1d) layer. These important characteristics are then passed into the BiLSTM layer, which retrieves high-dimensional properties. The BiLSTM features were input into the dense layer, which has a number of hidden units and a relu activation function.

Table 5: Hyperparameter settings for CBiLSTM

Hyperparameter Settings	Value
No. of CNN layer	[1, 2]
Filter Size	[512, 768, 1024]
Kernel size	[2, 3, 4, 5]
Activation function	[selu, elu, relu, tanh]
No.BiLSTM layer	[1, 2]
Hidden unit	[128, 256, 512]
Dropout	[0.1, 0.2, 0.3, 0.4]
No. of Dense layer	[3,4]
Final activation function	[softmax, softplus]
Optimizer	[adam, adamax, Rmsprop]

The dense layer outputs are converted into one converted into a one-dimensional vector. Finally, the output layer generates a probability distribution for prediction. The hyperparameter settings for the CBiLSTM model are given in Table 5. The model was optimized using the sparse categorical cross-entropy loss function and adam, adamax, and RMSprop optimizers with different batch sizes and epoch numbers.

4.3.4 BERT

In our experiment, we used Indic-Transformers Bangla BERT, which is a monolingual BERT model pre-trained on 3 GB of the monolingual training corpus. The pre-training data was majorly taken from OSCAR. This model is publicly available in the HuggingFace Library [35].

We use BERT's custom tokenizer, BertTokenizer, for tokenization, which includes masking and padding the sequences. For BERT fine-tuning, we use HuggingFace's wrapper BertForSequenceClassification, which is a normal Bert model with an added single classifier linear layer on top for classification. For classification tasks such as ours, only the output of the classification token [CLS] is used. BertForSequenceClassification utilizes the special [CLS] classifier token as the first token in every sequence. This token contains the classification embedding of the sequence. BERT uses the final hidden state of the [CLS]. During the

training, this additional untrained classification layer is trained on our specific task.

Table 6: Hyperparameter settings for fine-tuned BERT

Hyperparameter Settings	Value
eval_steps	500
train_batch_size	[8, 16, 32]
eval_batch_size	[8, 16, 32]
num_train_epochs	[2,3, ..., 9, 10]
Learning rate	[2e-5, 3e-5, 4e-5, 5e-5, 6e-5]
Weight decay	[0.1, 0.01, 0.001, 0.0001]
Save steps	3000

For classification tasks such as ours, only the output of the classification token [CLS] is used. BertForSequenceClassification utilizes the special [CLS] classifier token as the first token in every sequence. This token contains the classification embedding of the sequence. BERT uses the final hidden state of the [CLS]. During the training, this additional untrained classification layer is trained on our specific task. While optimizing, we fine-tuned Weight decay, Eval step, Train batch size, Test batch size, Learning rate, Save step, Batch size and Epoch hyperparameters. The best intermediate model is stored using the checkpoint and used to make predictions on the test data. The remainder of the hyperparameters are left at their default values.

5. EXPERIMENTS AND EVALUATIONS

5.1 Experimental Environment

For our experiment, we used the Python programming language on the "Google Colaboratory" platform. "Google Colaboratory" provides both GPU and TPU services. We utilized CUDA-enabled GPU services. Besides Pytorch, for data representation and visualization, we use Python libraries like NumPy, pandas, matplotlib, etc. For measuring classification performance, we used the sklearn library. We also utilize the Keras library with a Tensorflow background for the extrinsic evaluation.

To increase the performance of a model, Fine-tuning the model with different hyperparameter settings according to the given task and the respective dataset is an effective method. In our experiment, we use different hyperparameter settings to select the optimal configuration for all classifier models. Table 7 describes all optimal hyperparameter settings for all classifier models.

While evaluating, to obtain the improved performance, we fine-tuned our CBiLSTM, BiLSTM classifier models based on the number of CNN layers, BiLSTM layers, activation function, kernel size, hidden unit, dropout, batch size, final activation function, number of dense layers, epoch, and optimizer. For the vector representation of news document texts, we used the pre-trained 300-dimensional fastText embedding model at the embedding layer. We use the early-stopping method to overcome the model's overfitting on the training dataset. In the fastText classifier model, we fine-tuned the number of epochs, learning rate, and wordNgrams. While fine-tuning BERT, we optimize the number of epochs, train_batch_size, and eval_batch_size, learning rate, and weight decay. The rest of the parameters are set to their default values for all classifier models. In this work, we reported the results based on these settings.

Table 7: Optimal hyperparameter settings for classifier models

Model	Optimal Selection
CBiLSTM	No. of CNN layer: 1 Filter Size: 512 Kernel size: 3 Activation function: elu No.BiLSTM layer: 2 Hidden unit = 512 Dropout = 0.2 No. of Dense layer = 3 Final activation func.: softplus Optimizer: adamax Batch size = 128 Epoch = 50
BiLSTM	No. BiLSTM layer: 2 Hidden unit = 512 Dropout = 0.2 No. of Dense layer = 3 Final activation func.: softplus Optimizer: adamax Batch size= 128 epoch= 95
Fine-tuned BERT	per_device_train_batch_size = 32 per_device_eval_batch_size = 32 num_train_epochs = 3 learning_rate = 6e-5 weight_decay = 0.001 Model check-point = 1500
fastText	Learning rate = 1.0 wordNgrams = 2 epoch = 25

5.2 Evaluation Measures

A confusion matrix graphically represents the results of a classification process. The result of a confusion matrix can be better comprehended using four parameters: accuracy, precision, recall, and f1-scoreIn the equations for these parameters, there are a few common variables. True Positives (TP), True Negatives (TN), False Negatives (FN), and False Positives (FP) are the four types. To evaluate the performance of our experiment, evaluation metrics such as precision, recall, F1-score, and accuracy are used. To estimate the performance of a classifier model, we used macro average. An evaluation metric evaluates the performance of a predictive model. Precision measures the positive patterns that are correctly predicted from the total predicted patterns in a positive class. Recall measures the fraction of positive patterns that are correctly classified. The F1-score metric represents the harmonic mean between recall and precision values. Accuracy measures the ratio of correct predictions over the total number of instances evaluated.

5.3 Results and Discussion

In this work, we evaluated deep learning approaches such as CBiLSTM, BiLSTM, fastText, and Fine-tuned BERT for Bangla news document categorization based on our dataset.

Table 8: Performance comparison among the models

Model	Precision (%)	Recall (%)	F1-score (%)
CBiLSTM	91.70	91.67	91.66
BiLSTM	92.02	92.03	92.01
fastText	93.50	93.50	93.50
Fine-tuned BERT	94.75	94.73	94.74

We deployed the entire training dataset for training our proposed model and the validation dataset for hyperparameter tuning.

Table 9: Class-wise evaluation criteria for fine-tuned BERT Model

Category	Precision (%)	Recall (%)	F1-Score (%)
Accident	95.67	95.26	95.46
Art	89.76	91.60	90.67
Crime	92.21	91.28	91.74
Economics	95.77	93.68	94.71
Education	98.72	98.51	98.62
Entertainment	96.69	95.58	96.13
Environment	94.96	93.12	94.03
International	95.42	96.62	96.02
Opinion	92.06	92.46	92.26
Politics	93.63	95.40	94.51
Science & Tech	95.09	96.67	95.88
Sports	97.01	96.63	96.82
Macro Average			**94.74**

We analyzed the performance of individual models based on the test dataset. The comparative performance of our classifier model based on the test dataset is shown in Table 8.

Here, we can see that the Fine-tuned BERT model achieved an f1 score of 94.74%, where the f1 scores of the CBiLSTM, BiLSTM, and fastText classifiers are 91.66%, 92.01%, and 93.50%, respectively. From these results, we can say that Fine-tuned Bert outperforms all the other three models. Table 9 demonstrates the individual class performance of the fine-tuned BERT model.

In the confusion matrix shown in the Figure 3, each column indicates the instances in a predicted class and each row indicates the instances of an actual class.

From all of our experiments, we observe that the interclass relations between the classes can be seen in all of the confusion matrices from that dataset, implying that documents from one category can be categorized into another. According to the classification report and confusion matrix of each classifier model, the art category scored poorly, but the education category performed relatively well. Following an examination of the dataset, it was discovered that the Art category contains some incorrectly categorized items, as well as some documents that are semantically connected to the Entertainment category. Crime category and Accident category share similar content. Furthermore, the documents in the Politics and Crime categories also share similar content. If we can train the deep learning models with more data, they will be able to extract complicated characteristics from the data much more successfully. FastText classifier employed more train data than CBiLSTM, BiLSTM in this study, which has an effect on CBiLSTM, BiLSTM performance.

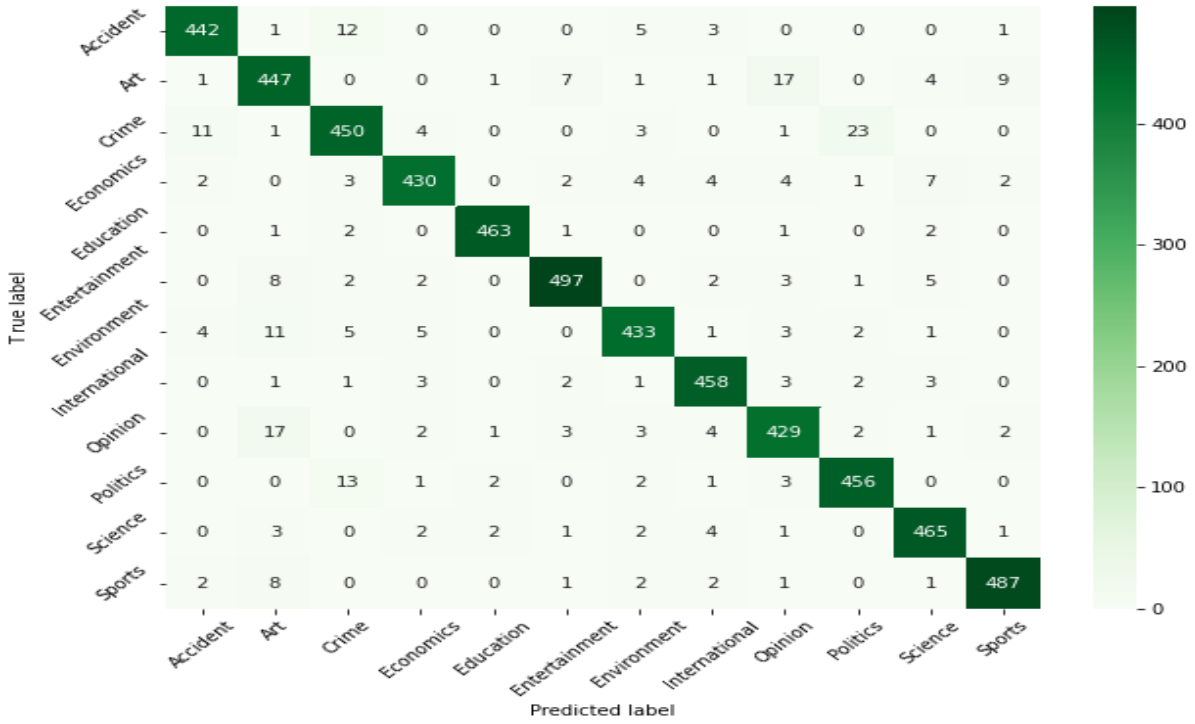

Figure 3: Confusion Matrix of Fine-tuned BERT model's prediction

6. CONCLUSION AND FUTURE WORK

Bangla language is spoken by a good number of people all over the world. With the great advancement of the internet and technology, the number of Bangla news text data in digital form is increasing exponentially. To manage, organize, and access valuable information from these huge volumes of data is a complex task. To handle this challenge, in this work, we collect publicly available Bangla corpus dataset and preprocess it before giving it to classifier models. We demonstrate deep learning approaches and a transformer-based fine-tuned BERT model for Bangla news document categorization. In our deep learning approaches, we utilize CBiLSTM, BiLSTM, and fastText classifiers. We use pre-trained FastText word embeddings with CBiLSTM, BiLSTM. We optimize these models with different hyperparameters. Then we fine-tune the Transformer based BERT model with different hyperparameters. We determine that the fine-tuned BERT model outperforms all the other models by a significant margin and achieved an accuracy of 94.74%. We use evaluation metrics such as F1 score, precision, recall, macro average, and confusion matrix for measuring model performance.

In the future, we plan to re-evaluate our approaches with the full dataset corpus. We also plan to explore different preprocessing and feature extraction methods and more complicated deep learning approaches and Transformer based models like DistilBERT, RoBERTa, XLM-Roberta, ELECTRA to improve performance and computational resources.

7. REFERENCES

[1] Durgesh, K. SRIVASTAVA, and B. Lekha. "Data classification using support vector machine."Journal of theoretical and applied information technology12, no. 1 (2010): 1-7.

[2] El Kourdi, Mohamed, Amine Bensaid, and Tajje-eddine Rachidi. "Automatic Arabic document categorization based on the Naïve Bayes algorithm." In proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages, pp. 51-58. 2004.

[3] Lewis, David D., and Marc Ringuette. "A comparison of two learning algorithms for text categorization." InThird annual symposium on document analysis and information retrieval, vol. 33, pp. 81-93. 1994.

[4] Johnson, Rie, and Tong Zhang. "Effective use of word order for text categorization with convolutional neural networks."arXiv preprint arXiv:1412.1058 (2014).

[5] Lai, Siwei, Liheng Xu, Kang Liu, and Jun Zhao. "Recurrent convolutional neural networks for text classification." In Twenty-ninth AAAI conference on artificial intelligence. 2015.

[6] Zhou, Chunting, Chonglin Sun, Zhiyuan Liu, and Francis Lau. "A C-LSTM neural network for text classification."arXiv preprint arXiv:1511.08630 (2015).

[7] Sun, Chi, Xipeng Qiu, Yige Xu, and Xuanjing Huang. "How to fine-tune bert for text classification?" In China national conference on Chinese computational linguistics, pp. 194-206. Springer, Cham, 2019.

[8] Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter."arXiv preprint arXiv:1910.01108 (2019).

[9] Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. "Roberta: A robustly optimized bert pretraining approach."arXiv preprint arXiv:1907.11692 (2019).

[10] Britannica, T. Editors of Encyclopaedia. "Bangla language." Encyclopedia Britannica, July 28, 2017. https://www.britannica.com/topic/Bangla-language.

[11] Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. "Enriching word vectors with subword

information."Transactions of the association for computational linguistics 5 (2017): 135-146.

[12] Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. "Bag of tricks for efficient text classification."arXiv preprint arXiv:1607.01759 (2016).

[13] Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." IEEE transactions on Signal Processing 45, no. 11 (1997): 2673-2681.

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

[15] Chy, Abu Nowshed, Md Hanif Seddiqui, and Sowmitra Das. "Bangla news classification using naive Bayes classifier." In 16th Int'l Conf. Computer and Information Technology, pp. 366-371. IEEE, 2014.

[16] Al Mostakim, Sadek, Faiza Ehsan, Syeda Mahdiea Hasan, Sadia Islam, and Swakkhar Shatabda. "Bangla content categorization using text based supervised learning methods." In 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1-6. IEEE, 2018.

[17] Kabir, Fasihul, Sabbir Siddique, Mohammed Rokibul Alam Kotwal, and Mohammad Nurul Huda. "Bangla text document categorization using stochastic gradient descent (sgd) classifier." In 2015 International Conference on Cognitive Computing and Information Processing (CCIP), pp. 1-4. IEEE, 2015.

[18] Mahmud, Quazi Ishtiaque, Noymul Islam Chowdhury, and Md Masum. "Reducing feature space and analyzing effects of using non linear kernels in svm for bangla news categorization." In 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1-6. IEEE, 2018.

[19] Islam, Md, Fazla Elahi Md Jubayer, and Syed Ikhtiar Ahmed. "A comparative study on different types of approaches to Bangla document categorization." arXiv preprint arXiv:1701.08694 (2017).

[20] Islam, Md Saiful, Fazla Elahi Md Jubayer, and Syed Ikhtiar Ahmed. "A support vector machine mixed with TF-IDF algorithm to categorize Bangla document." In 2017 international conference on electrical, computer and communication engineering (ECCE), pp. 191-196. IEEE, 2017.

[21] Dhar, Ankita, Niladri Sekhar Dash, and Kaushik Roy. "Application of tf-idf feature for categorizing documents of online bangla web text corpus." In Intelligent Engineering Informatics, pp. 51-59. Springer, Singapore, 2018.

[22] Dhar, Ankita, Niladri Sekhar Dash, and Kaushik Roy. "Categorization of Bangla web text documents based on TF-IDF-ICF text analysis scheme." In Annual Convention of the Computer Society of India, pp. 477-484. Springer, Singapore, 2018.

[23] Mahmud, Quazi Ishtiaque, Noymul Islam Chowdhury and Mohammad Masum. "A Multi Layer Perceptron Along with Memory Efficient Feature Extraction Approach for Bangla Document Categorization." Journal of Computer Science 16 (2020): 378-390.

[24] Hossain, Md, and Mohammed Moshiul Hoque. "Automatic Bangla document categorization based on deep convolution nets." In Emerging Research in Computing, Information, Communication and Applications, pp. 513-525. Springer, Singapore, 2019.

[25] Mojumder, Pritom, Mahmudul Hasan, Md Hossain, and K. M. Hasan. "A study of fasttext word embedding effects in document classification in Bangla language." In International Conference on Cyber Security and Computer Science, pp. 441-453. Springer, Cham, 2020.

[26] Rahman, Md Mahbubur, Rifat Sadik, and Al Amin Biswas. "Bangla document classification using character level deep learning." In 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp. 1-6. IEEE, 2020.

[27] Ahmad, Adnan, and Mohammad Ruhul Amin. "Bangla word embeddings and it's application in solving document classification problem." In 2016 19th international conference on computer and information technology (ICCIT), pp. 425-430. IEEE, 2016.

[28] Alam, Md Tanvir, and Md Mofijul Islam. "Bard: Bangla article classification using a new comprehensive dataset." In 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1-5. IEEE, 2018.

[29] Khushbu, Sharun Akter, Abu Kaisar Mohammad Masum, Sheikh Abujar, and Syed Akhter Hossain. "Neural network based Bangla news headline multi classification system: Selection of features describes comparative performance." In 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1-6. IEEE, 2020.

[30] Tudu, Ronald, Shaibal Saha, Prasun Nandy Pritam, and Rajesh Palit. "Performance analysis of supervised machine learning approaches for Bangla text categorization." In 2018 5th Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), pp. 221-226. IEEE, 2018.

[31] Alam, Tanvirul, Akib Khan, and Firoj Alam. "Bangla text classification using transformers." arXiv preprint arXiv:2011.04446 (2020).

[32] Rahman, Md Mahbubur, Md Aktaruzzaman Pramanik, Rifat Sadik, Monikrishna Roy, and Partha Chakraborty. "Bangla documents classification using transformer based deep learning models." In 2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI), pp. 1-5. IEEE, 2020.

[33] Ghosh, Koyel, and Apurbalal Senapati. "Technical domain classification of bangla text using BERT." Biochemistry (bioche) 2 (2021): 741.

[34] Open Source Bangla Dataset Corpus. https://scdnlab.com/corpus/

[35] "Neuralspace-Reverie/Indic-Transformers-BN-Bert · Hugging Face." neuralspace-reverie/indic-transformers-bn-bert · Hugging Face. Accessed July 23, 2022. https://huggingface.co/neuralspace-reverie/indic-transformers-bn-bert.