Journal of Computer Science

Original Research Paper

A Multi Layer Perceptron Along with Memory Efficient Feature Extraction Approach for Bengali Document Categorization

[1]Quazi Ishtiaque Mahmud, [2]Noymul Islam Chowdhury and [2]Md Masum

[1]Institute of Information and Communication Technology, Shahjalal University of Science and Technology, Sylhet, Bangladesh
[2]Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet, Bangladesh

Article history
Received: 05-01-2020
Revised: 27-02-2020
Accepted: 27-03-2020

Corresponding Author:
Mr. Quazi Ishtiaque Mahmud
Institute of Information and Communication Technology, Shahjalal University of Science and Technology, Bangladesh
Email: rafisustcse@gmail.com

Abstract: In terms of the total number of speakers in the world Bengali stands as the seventh language and it has been used by approximately 265 million people worldwide. Day by day more people are expressing their views and opinions in Bengali in digital platforms like blogs and social media on various topics. Despite this, very little work has been done to structure these electronic documents according to their categories. In this paper, a methodology is developed for automatically categorizing Bengali news among twelve predefined categories using a Multi Layer Perceptron (MLP) model. We also explored the optimization opportunities that lie within the feature space and illustrated the difficulties that arise while handling large feature spaces in neural networks. It has been shown in this paper that the feature space can be optimized to achieve better accuracy. Using our modified feature extraction technique, we reduced the feature space and achieved an accuracy of 93.3%.

Keywords: Document Categorization, TF-IDF, Multi Layer Perceptron, Activation Functions

Introduction

Document categorization is one of the key problems nowadays. It is also important for real-life applications and services. Bengali is one of the most popular languages in the world. So it is also becoming important to work with Bengali document categorization. We had studied different works on document categorization. Numerous works have been done for English document categorization but actually, very little work has been done for Bengali document categorization. But as ours is also a categorization problem we studied some previous works in different languages like English and Arabic text categorization problem.

It has been shown that categorization can be handled in two phases, Hypothesization and Confirmation (Hayes *et al*., 1988). In the Hypothesization phase, depending on the words and content, it attempts to pick out all categories into which the story might fall. When particular words and phrases suggest more than one category, they will contribute to the Hypothesization of each of those categories. Confirmation is the phase in which it attempts to find additional evidence in support of a Hypothesized topic. Using a pattern-matching technique some basic kind of processing is done on both phases. They ran a set of 500 stories through the system and claimed an average recall rate of 93%.

Probabilistic measures are also used to classify English documents (Wu *et al*., 2004). They used Support Vector Machines and generated the probabilities of two classes at an instant.

Srivastava and Bhambhu (2010) also used Support Vector Machine (SVM), rule-based classifier and KNN classifier to solve the problem of categorization. They used four different types of data: Heart data, Diabetes data, Satellite data and Shuttle data to test their model. They collected the data from the "UCI Machine Learning Repository" (Dua and Graff, 2019).

Joachims (1998) used SVM for English document categorization and showed that SVM suits well for the task of text categorization. They also showed a comparison between RBF and Polynomial kernel in SVM.

In their research, Lam and Lee (1999) proposed a technique for feature reduction in neural network based text categorization. They mentioned that neural network performed poorly if it is trained with raw text data. Because textual representation has a high dimensional feature space. They proposed an approach to reduce the feature space into an input space of much lower

dimension. They used Reuters-21578 (Dua and Graff, 2019) test collection to see the effectiveness of their proposed model. They proposed four dimension reduction techniques and among them, they showed that principal component analysis was the most effective one.

Pedresen and Yang (1997) demonstrated a comparative study on feature selection for text categorization. They focused on dimensionality reduction. They evaluated five methods for dimension reduction including Document Frequency, Information Gain, Mutual Information, Chi test and Term Strength. They showed that Mutual Information (MI) has poor performance for its bias towards favoring rare terms and sensitivity to probability estimation error.

Zhang and Zhou (2006) used multi label neural network for text categorization. Each instance in the training set is associated with a set of labels in multi label learning. They proposed back propagation multi label learning to solve the problem. This algorithm is derived from the popular back propagation algorithm using a novel error function that captures the characteristics of multi label learning.

Subramaniyaswamy and Pandian (2012) have done some works on English blog topic categorization. They used the Keyword Frequency - Inverse Topic Frequency (KF-ITF) weighting method to filter out keywords from a blog. Then they used the high-frequency keywords as features for their SVM classifier. They trained their system for 100 blogs and achieved accuracies ranging from 90.00% to 96.73% for different categories.

Chekima and Anthony (2011) categorized English scientific papers among five categories. They proposed a keyword extraction based system that categorizes a document by analyzing the key terms that are present in it.

Mesleh (2007) proposed a method to categorize Arabic documents into nine predefined categories by using SVM classifier along with Chi square feature extraction technique.

Bawaneh *et al*. (2008) proposed a methodology using K-NN and Naïve Bayes to solve the problem of Arabic Text categorization.

Bayesian model along with character-level n-gram was also used to categorize Arabic text documents (Al-Salemi and Ab Aziz, 2011).

Ruiz and Srinivasanan (1999) used a hierarchical neural network for text categorization. Their model uses a divide and conquer approach to defining the smallest problem based on a predefined hierarchical structure.

Mansur *et al*. (2005) used n-gram based approach for Bengali news categorization. For their experiment, they randomly selected 25 test documents. They worked with six categories. And they generated their own corpus. They tested their model for 150 documents.

Naïve Bayes classifier was used for classifying Bengali documents (Chy *et al*., 2014). The authors used a small corpus which contained around 700 documents. They gathered this data by crawling news from the site of the famous Bengali newspaper Prothom Alo. They also proposed a stemming algorithm to extract features.

Quadery *et al*. (2016) used Chi square distribution to select the important features for categorization. Then they built the classifier model by using the Naïve Bayes technique. They used the OSBC (2020) which contains 35000 documents of 12 categories. They trained their model for 6430 documents.

Islam *et al*. (2017) used a stemmer developed by Urmi *et al*. (2016) to reduce the number of features and achieve higher accuracy. They used TF-IDF technique to sort out the relevant features and SVM with linear kernel as the classifier. They claimed an accuracy of 92.57% for the OSBC (2020) dataset.

Authors Mahmud *et al*. (2018) improved the model of authors Islam *et al*. (2017) and showed that better accuracy can be achieved using much fewer features on the same dataset (OSBC, 2020).

So, not a lot of researches are carried out for Bengali document categorization. We found that the existing methodologies can be improved by analyzing features more carefully and applying more sophisticated classification algorithms like Multi Layer Perceptron (MLP) model.

Methodology

In this section, the steps of our work will be discussed. The first subsection describes the categories that are considered. In the second subsection, an overview of the corpus is given. The third and fourth subsection describes the preprocessing that has been carried out and the feature extraction technique that is used. A description of our Multi Layer Perceptron (MLP) model is provided in the fifth subsection. In the next two subsections, the memory issues and how it is solved by reducing the feature space are discussed. A comparison of two feature extraction methods is given in the eighth subsection. The algorithm for building the modified TF-IDF matrix is described in the ninth subsection. And in the last subsection, an analysis is given of the final feature matrix.

What is Classification

Classification means assigning documents to their appropriate classes or groups. It is different from clustering by the fact that in clustering the number of classes cannot be known in advance but in the case of classification, the number of classes is defined. Categorization among twelve classes is considered in this research. They are accident, art and literature, crime, environment, entertainment, education, international, politics, opinion, science and tech and sports.

Data Collection

In all of our experiments, the OSBC (2020) is used. Table 1 below represents the information about the corpus. To make sure that the dataset is not biased towards a specific category, same number of documents are taken for each category.

Data Preprocessing

Data preprocessing is a necessary step for all machine learning techniques. In this research, some preprocessing techniques are applied. Firstly, words are identified from news. Then punctuation symbols are removed. Finally, stemming is applied on Bengali words using the stemmer developed by authors (Urmi *et al*., 2016).

Feature Extraction

For feature extraction, the TF-IDF score of every unigram in our dataset is used. Equation 1 is used to calculate TF-IDF score of a feature:

$$tf - idf(t,d) = tf(t,d) * idf(t) \tag{1}$$

where term frequency $tf(t,d)$ means the number of times a word or term(t) appears in the document(d). Equation 2 is used to calculate the $idf(t)$ from the following formula:

$$idf(t) = \log \frac{1 + n_d}{1 + df(d,t)} + 1 \tag{2}$$

Here n_d means the total number of documents and $df(d,t)$ means the total number of documents that have the term(t). Then L2 normalization is used to normalize the TF-IDF scores so that our classifier converges early using Equation 3:

$$v_{norm} = \frac{v}{\| v \|_2} \tag{3}$$

The documents are converted into n-dimensional input vectors using this TF-IDF transformation. Where n is the number of features. Suppose there are two documents:

Doc1 = "সাকিব ভালো খেলে"
Doc2 = "তিনি আজ ভালো খেলেন নাই"

Table 2 refers to the TF-IDF matrix of these two documents.

For illustration purposes, all the words are considered as features. Now after converting the documents into TF-IDF matrix each of the term (word or Unigram) will have a TF-IDF score. The top 30000 terms are selected as features for our model. Now each of the documents will have a label associated with them. The documents are given as inputs to our neural network, after that the weights are initialized, then activation functions are selected and outputs of the hidden layer and also the final layer are calculated. Finally, our result is compared with the actual output, then the error is calculated and the weights are updated accordingly. Figure 1 represents how documents are used as inputs to the neural network. Suppose Doc1 is given as input to our network. As there are 7 features in total, the input layer will contain 7 neurons. Suppose there are 2 classes. So a single neuron or 2 neurons at the output layer can be used. There may be as many hidden layers as we want. But 1 hidden layer with 3 neurons is chosen. The number of output neurons is determined by the number of classes and the number of hidden layer neurons is determined by (the number of input layer neurons)/2 (Marsland, 2014).

Neural Network Configuration

Neural networks are indeed the most popular techniques in the field of machine learning. In this research, the MLP model is used because using Platt's method (Platt, 1999) authors Mahmud *et al*. (2018) showed that the OSBC (2020) contains non-linear characteristic. And single layer neural networks cannot predict non-linear functions (Marsland, 2014). Table 3 refers to the configuration of our neural network.

Memory Issue

Our feature matrix consists of 28717 documents and it contains 30000 features. These 30000 features are chosen according to their TF-IDF scores. That means that the size of our feature matrix is 28717*30000. So we needed a matrix whose dimensions are 28717*30000. And the data type of the matrix is float because they contain values between 0 and 1. Each float takes 4 Bytes memory. So, (4*28717*30000)/10^9 = 3.45 GB memory is required to store the TF-IDF matrix.

Our MLP model has an input layer of 30000 neurons, a hidden layer of 15000 neurons (because hidden layer neuron number = input layer neuron number/2) (Marsland, 2014) and an output layer of 12 neurons (because there are 12 categories). So, firstly 3.45 GB memory is required to store the TF-IDF matrix. Now ((30000*15000+15000*12)*4)/10^9 = 1.8 GB memory is needed to store the weights. That means in total 5.25 GB memory is required only for storing the TF-IDF matrix and the weights. The machine that is used to conduct the experiment has only 8 GB RAM available. So, when the algorithm is executed on this machine after 5-10 min it stops responding. So, this much weight cannot be used. Now the reduction of weights means the reduction of the number of features. So, our features have to be analyzed again and scopes for optimization needed to be found.

Table 1: Our experiment corpus

Categories	Number of documents	Training/Testing
Accident	2659	2412/247
Art	2659	2364/295
Crime	2659	2401/258
Economics	2659	2384/275
Education	2659	2384/275
Entertainment	2659	2379/280
Environment	2659	2407/252
International	2659	2378/281
Opinion	2659	2414/245
Politics	2659	2401/258
Science\& Tech	2659	2400/259
Sports	2659	2393/266
Total	31908	28717/3191

Table 2: Document to TF-IDF matrix

	আজ	খেলে	খেলেন	তিনি	নাই	ভালো	সাকিব
Doc1	0.00	0.63	0.00	0.00	0.00	0.44	0.63
Doc2	0.47	0.00	0.47	0.47	0.47	0.33	0.00

Table 3: Neural network configuration

Number of hidden layers	1
Number of input layer neurons	12
Number of hidden layer neurons	5500, 6000, 6500, 7000 (Four different configurations were used)
Number of output layer neurons	12
Initialization of weights	Between $-\frac{1}{\sqrt{n}}$ and $\frac{1}{\sqrt{n}}$; n = number of features (Marsland, 2014)

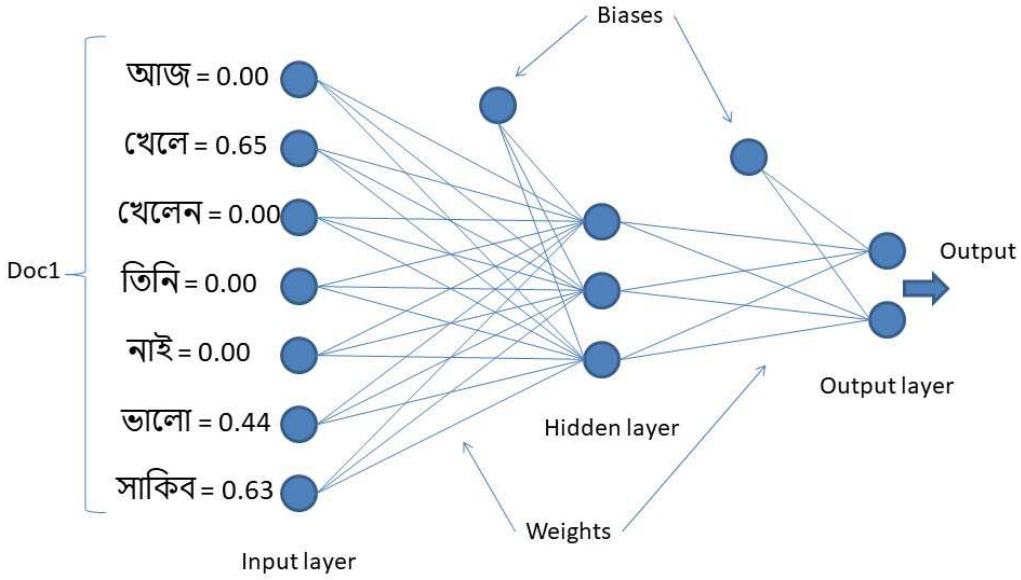

Fig. 1: TF-IDF matrix with MLP

Reducing Feature Space

In this section, the experiments that are carried out to reduce our feature space for our MLP model are discussed and also an analysis of the performance is given. Using TF-IDF based feature extraction technique the words that have better TF-IDF scores are selected. But there are a few problems with that approach. Firstly, using that technique, it is not possible to identify how many features have been selected from a particular class. Secondly, it can be biased towards a specific category. Because as we are taking features based on their frequencies there can be a lot of features for a particular category whose TF-IDF scores will fall behind the TF-IDF scores of other categories' features. As a result, the number of features selected will not be the same for all the categories. Table 4 Refers to a sample TF-IDF score matrix.

Each of the rows in Table 4 represents each of our documents and each of the columns corresponds to each of our features. Now the value at each position of the matrix is the TF-IDF score for that feature in that document. Now, these features are sorted according to their average TF-IDF scores. Table 5 represents the sorted features.

This matrix is created for all of the documents of all of the categories combined. Then the features with lower TF-IDF scores are removed. Initially, the first 30000 features were chosen. But now we will consider taking features separately from all of our categories. Then all of our features will be combined to make the final feature matrix. For example, suppose there are 12 categories of documents and our goal is to select the top 24 features. If the traditional approach is followed, then a TF-IDF matrix will be built for all the documents and the matrix needs to be sorted so that the first 24 features can be chosen. But in our proposed approach 12 separate TF-IDF matrices need to be created for 12 categories. Then each of the individual matrices needs to be sorted and from each matrix, the first 2 features will be selected.

For our research, 12 sorted TF-IDF matrices are created for 12 categories. Then the first 1000 features for each category are chosen. That means a total of 12000 features in total (12 categories). But there were a few noises in the features. So after removing those noisy words, our total feature count was 11577. So, it can be seen that using this technique, our feature space is reduced to more than half of our previous feature space of 30000 features.

Comparing the two Approaches

If features are chosen without creating separate feature matrix for each category, then it can be seen from Table 6 that there exist features like ″হয়″ (happens), ″হয়েছে″ (happened), ″হচ্ছে″ (happening) which don't represent a specific category. To get good results using this type of feature selection technique, a huge number of features need to be considered.

Table 7 and 8 show the 24 features that are selected using our modified feature extraction technique. The modified feature extraction technique produces much more relevant features. For example, there are no features like "হয়েছে" (happened), "হচ্ছে" (happening). When using traditional feature extraction technique, some very important words were missed for example: "দুর্ঘটনা" (accident), a very important feature for accident class, "গল্প" (story), "কথা" (tale), important for art-literature class, "নেতা" (leader) important for politics category, "নতুন্" (new), "তথ্য" (information), important for science and tech category, "এলাকা" (area), "পরিবেশ" (environment), important for environment category. Also, how each of the categories contributed to our feature list can be precisely known. The same amount of features has been taken from each of the categories. So, the final feature list will have an equal number of features from each of the categories of the news. So, it will not be biased towards a specific category.

Table 4: A sample TF-IDF matrix

	Feature1	Feature2	Feature3	Feature4	Feature5
Doc1	0.33	0.00	0.63	0.00	0.01
Doc2	0.00	0.00	0.33	0.00	0.02
Doc3	0.94	0.00	0.56	0.27	0.09
Doc4	0.00	0.39	0.00	0.31	0.01

Table 5: Sorted TF-IDF matrix

	Feature3	Feature1	Feature4	Feature2	Feature5
Doc1	0.63	0.33	0.00	0.00	0.01
Doc2	0.33	0.00	0.00	0.00	0.02
Doc3	0.56	0.94	0.27	0.00	0.09
Doc4	0.00	0.00	0.31	0.39	0.01

Table 6: Top 24 features (without creating separate feature matrix for each category)

উত্তর (answer)	দেশ (country)	শেষ (end)
কথা (tale)	পুলিশ (police)	সময় (time)
করেছে (done)	প্রথম (first)	সরকার (government)
কাজ (work)	বছর (year)	সাল (year)
জানান (tell)	বাংলাদেশ (bangladesh)	হচ্ছে (happening)
টাকা (money)	মন (mind)	হয় (happens)
দল (team)	মানুষ (human)	হয়েছে (happened)
দিন (day)	রাত (night)	হাজার (thousand)

Table 7: Top 2 features from accident, art, crime, economics, education and entertainment category

দুর্ঘটনা (accident)	Accident features	বাংলাদেশ (Bangladesh)	Economics features
জানান (tell)		দেশ (country)	
গল্প (story)	Art features	শিক্ষার্থী (student)	Education features
কথা (tale)		ঢাকা (Dhaka, name of a city)	
পুলিশ (police)	Crime features	ছবি (film)	Entertainment features
গতকাল (yesterday)		বছর (year)	

Table 8: Top 2 features from environment, international, opinion, politics, science and technology and sports category

পরিবেশ (environment)	Environment features	বিএনপি (BNP, a political party)	Politics features
এলাকা (area)		নেতা (leader)	
খবর (news)	International features	নতুন (new)	Science and Technology features
হয় (happens)		তথ্য (information)	
সরকার (government)	Opinion features	দল (team)	Sports features
মানুষ (human)		ম্যাচ (match)	

Building the New Feature Matrix

As a new TF-IDF matrix has been created for every category and then sorted features are taken from each category, there needs to be a way to somehow combine all these features and make the complete feature matrix. Now one thing is for certain is that the TF-IDF values cannot be used because they have been calculated separately for each of the categories. The features need to be represented by something different rather than TF-IDF scores. Please refer to the pseudocode for our proposed algorithm for creating the feature matrix.

To have a better understanding, first all the features are selected using our proposed methodology discussed above. Then for every document, we find every word of that document and if that word is present in our feature list then the value of that position is set to 1 else it will be 0. The same documents are considered for building the new feature matrix. Here also:

Doc1 = "সাকিব ভালো খেলে"
Doc2 = "তিনি আজ ভালো খেলেন নাই"

For illustration purposes, all the words are considered as features this time too. Please note the difference between Table 2 and 9. The pseudocode is used to build the feature matrix. In Table 2 the matrix has floating-point values. But in Table 9, the matrix has only values 0 and 1. At first, it might appear that some information is lost because the TF-IDF scores which represent the importance of a word are not present in the feature matrix. But it is not true because even though TF-IDF scores are not present in the final feature matrix but the features are chosen according to the average TF-IDF scores.

Pseudocode for building the new feature matrix is given below:

1. set rows to number of documents
2. set column to number of features
3. create a 2D matrix total_matrix(rows, columns)
4. initialize each cell of total_matrix to zero
5. create a map named all_features that will contain the mapping of a feature to an integer
6. set all_categories to the name of the twelve categories
7. set i to zero
8. for every category in all_categories
9. set docs to all documents belonging to this category
10. for every document in docs
11. set doc_words to all the words in that document
12. for every word in doc_words
13. if that word is present in all_features
14. find the value of the word from all_features
15. set j to value obtained in step 14
16. set total_matrix(i, j) to 1
17. endif
18. endfor
19. increase value of i by 1
20. endfor
21.endfor

Table 9: Document to TF-IDF matrix using

	আজ	খেলে	খেলেন	তিনি	নাই	ভালো	সাকিব
Doc1	0	1	0	0	0	1	1
Doc2	1	0	1	1	1	1	0

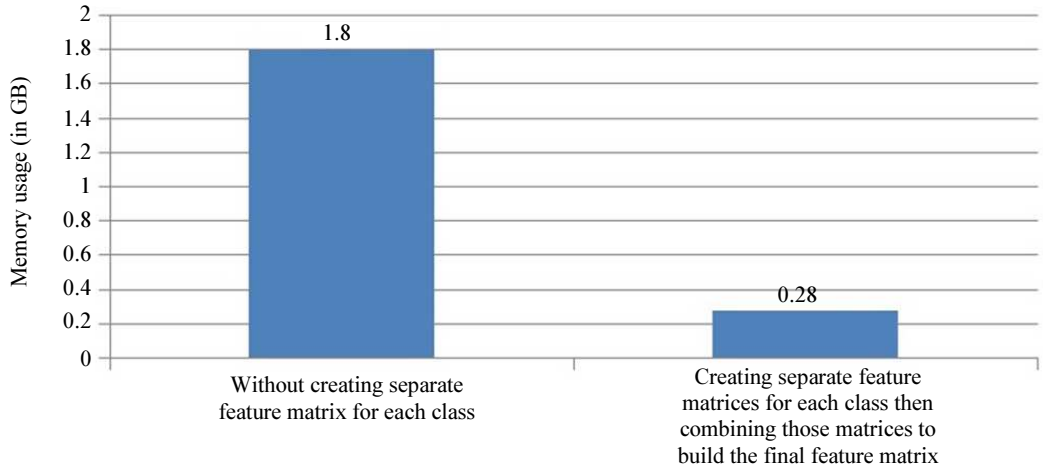

Fig. 2: Comparing memory usage for the two feature extraction methods

Analyzing the Feature Matrix

After building the feature matrix, the memory consumed by the feature matrix is analyzed. The new feature matrix consists of 11577 features and 28717 documents. So, the feature matrix that is built using our new feature extraction technique consists of 28717 rows and 11577 columns. That means the new feature matrix requires (((11577*6000)+(6000*12))*4)/10^9 = 0.28 GB memory. Here 6000 is chosen as our hidden layer size. Figure 2 represents how much memory space is reduced by using the new feature extraction approach.

Experiment and Result Analysis

Implementing MLP with Our New Feature Extraction Technique

After reducing our feature space, the MLP model is implemented. This time our model consists of 11577 input layer neurons and our output layer neuron contains 12 neurons for 12 categories. This time our model consists of 6000 neurons in the only hidden layer. Four different activation functions are implemented. Activation function simply indicates how we are deciding whether a neuron fires or not. Figure 3 shows the results of our experiment. The best accuracy (93.3%) is achieved with the Relu activation function.

Experimenting with Different Hidden Layer Size

After getting very good accuracies using this model, some experiments are carried out with different hidden layer sizes. Figure 4 shows the comparison graph for our experiment. The best performance is achieved by using the traditional technique for choosing the hidden layer size which is half of the number of hidden layer neurons (Marsland, 2014).

Dealing with Overlapping Features

Then the overlapping features are removed. That means features that belong to at least two classes. Figure 5 represents our findings. But after removing overlapping features our performance decreases. So, the overlapping features are kept. But in our feature matrix, any duplicate features are not allowed. That means all overlapping features are considered only once.

According to Fig. 5 the accuracy decreases because feature distribution is biased. The number of overlapped features for each category is not the same. Many information is lost when we remove all the overlapping features. It can be seen that the number of unique features is very low. It is hard to make predictions based on such a small feature set. So it may seem that overlapping features are unnecessary but actually, they are not. Figure 6 represents how many overlapped features are there in a different number of categories. For example, 1198 features are present in the feature list of two categories, 525 features are present in the feature list of 7 categories and 1536 features are present in the feature list of all the 12 categories. When all the overlapping features are removed, some important information is lost too. That's why our accuracy decreases.

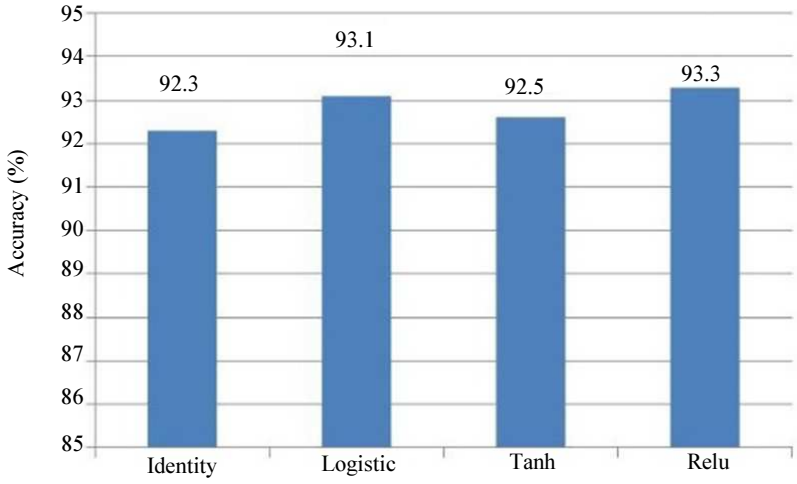

Fig. 3: Comparing different activations for MLP

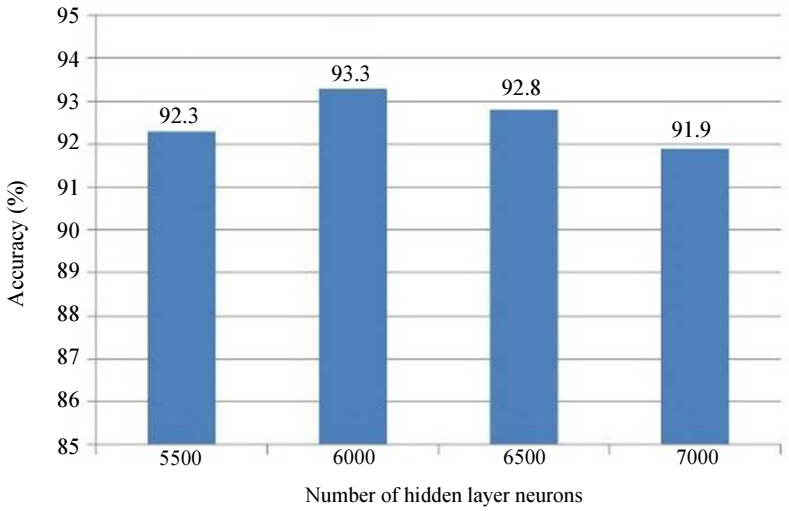

Fig. 4: Comparing among different hidden layer sizes

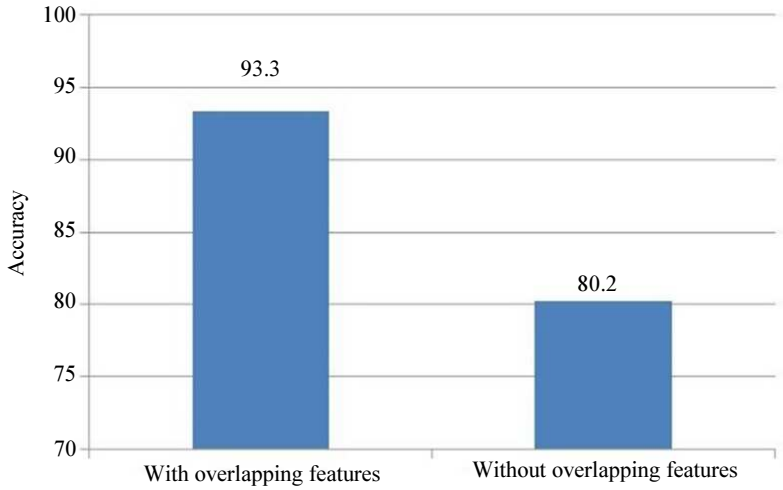

Fig. 5: Comparing the effects of overlapping features

Fig. 6: Number of overlapped features

Performance Analysis

Three metrics are used to analyze the performance of the proposed model. Firstly, accuracy is considered to help us identify the standard of the proposed model compared to other existing methods. Then the number of features is considered to identify how well feature space is optimized. Lastly, code execution time is considered to compare the amount of training time that is required to train our model with the training time of other existing models. These three metrics are chosen to have a clear picture of all the aspects of our model. In all the comparisons the OSBC (2020) dataset is used.

Comparison of Accuracy

The accuracy of the proposed model compared to the works of Chy *et al*. (2014), Quadery *et al*. (2016), Islam *et al*. (2017) and Mahmud *et al*. (2018) is shown in Fig. 7. As mentioned above the OSBC (2020) is used to conduct all the experiments. Chy *et al*. (2014) and Quadery *et al*. (2016) both used Naïve Bayes as their classifier. Chy *et al*. (2014) used TF-IDF as their feature selection technique. But, Quadery *et al*. (2016) used Chi square feature extraction approach. The model of Chy *et al*. (2014) and Quadery *et al*. (2016) achieved 87.52% and 87.18% accuracy on the OSBC (2020) dataset respectively. Islam *et al*. (2017) used SVM (linear kernel) along with TF-IDF feature selection technique. Their model achieved 92.57% accuracy. Mahmud *et al*. (2018) also used SVM with linear kernel but they reduced the TF-IDF matrix by only considering terms (features) such that the frequency of the term passes a certain threshold value known as the Term Frequency (TF) threshold. Using their modified feature extraction approach they achieved 92.79% accuracy. After comparing all these previous models, it can be seen that the proposed model performs fairly well compared to other models and it has an accuracy of 93.3%. It can be seen from the comparison that Bayesian models are outperformed by SVM and MLP in terms of accuracy.

Comparing the Number of Features

Figure 8 represents the comparison graph for the number of features that are used. The proposed model performs quite well when the number of features used by other architectures is considered. As can be seen from Fig. 8 the proposed model uses only 11577 features which is far lower than the models of Chy *et al*. (2014), Quadery *et al*. (2016), Islam *et al*. (2017) and Mahmud *et al*. (2018) where 100000, 96900, 216576 and 30000 features are used respectively.

Comparing Code Execution Time

Figure 9 represents our findings regarding the code execution time. If code execution time is considered, then the model of Mahmud *et al*. (2018) turns out to be better than all the other models. For Bayesian models, like the models used by Chy *et al*. (2014) and Quadery *et al*. (2016), a major portion of the time is needed to find the right number of features. But for all other models, training the models is the most time-consuming task.

It can be said that the proposed model performs pretty well despite the fact that it takes too much time

while learning. But neural networks always learn slowly. When the number of features used and the accuracy are considered then our model performs better.

Precision, Recall and F1 Score

Very often in machine learning, the F1 score is used to measure our actual performance because in many cases accuracy does not always give us the right indication about performance. Table 10 represents our precision, recall and F1 scores. From Fig. 10 it can be observed that our F1 score surpasses the models of Chy *et al*. (2014), Quadery *et al*. (2016), Islam *et al*. (2017) and Mahmud *et al*. (2018).

Confusion Matrix

Table 11 shows that the categories are mapped to integers. Table 12 refers to our confusion matrix for the 12 categories.

Table 10: Precision, recall and F1 score

Category	Precision	Recall	F1 Score
Accident	0.93	0.96	0.94
Art	0.93	0.93	0.93
Crime	0.94	0.93	0.93
Economics	0.93	0.95	0.94
Education	0.97	0.97	0.97
Entertainment	0.94	0.96	0.95
Environment	0.93	0.93	0.93
International	0.92	0.93	0.93
Opinion	0.93	0.90	0.92
Politics	0.95	0.92	0.93
Science	0.93	0.94	0.92
Sports	0.96	0.95	0.95
Avg.	0.94	0.94	0.94

Table 11: Integer-category mapping

Category	Number
Accident	0
Art	1
Crime	2
Economics	3
Education	4
Entertainment	5
Environment	6
International	7
Opinion	8
Politics	9
Science	10
Sports	11

Table 12: Confusion matrix for 12 categories

	0	1	2	3	4	5	6	7	8	9	10	11
0	194	12	1	0	5	2	9	0	0	0	2	2
1	6	240	4	0	0	1	9	0	1	4	0	0
2	0	0	193	1	0	1	3	2	1	8	12	1
3	0	1	0	261	6	3	1	0	2	0	5	1
4	0	0	1	0	239	5	3	0	2	0	4	4
5	0	0	0	0	13	223	3	0	1	1	1	0
6	0	3	0	0	4	1	240	0	3	0	0	0
7	1	2	4	0	1	2	3	220	2	0	6	1
8	2	2	8	0	13	3	1	8	203	1	1	2
9	0	25	5	0	0	0	3	2	1	224	0	0
10	3	0	5	0	2	1	3	0	0	0	220	3
11	0	0	0	0	8	2	3	1	1	0	0	257

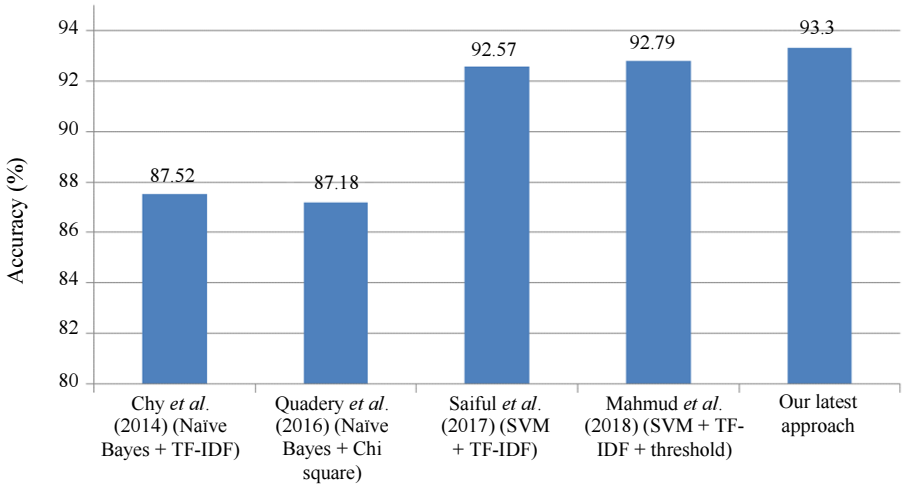

Fig. 7: Comparison of accuracy

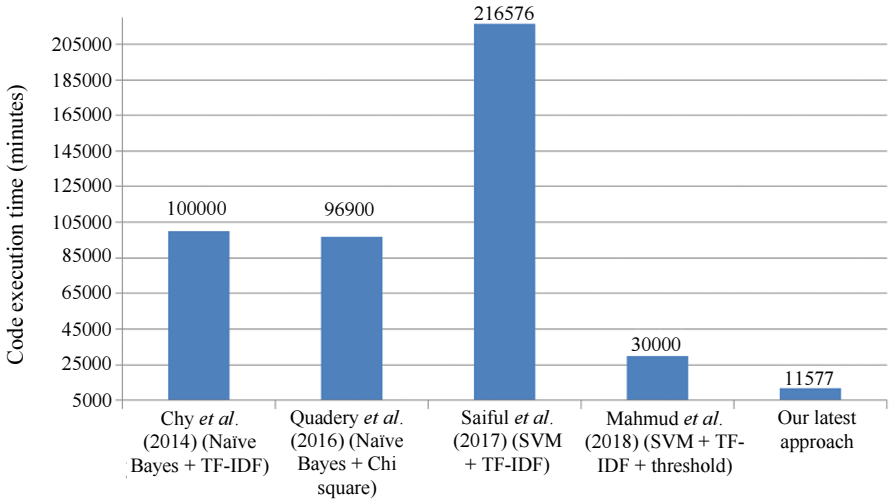

Fig. 8: Comparing the number of features

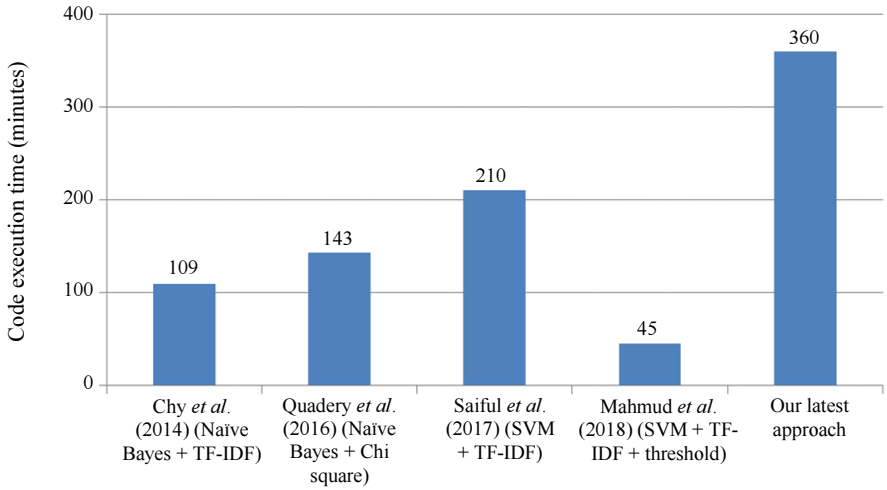

Fig. 9: Comparing code execution time

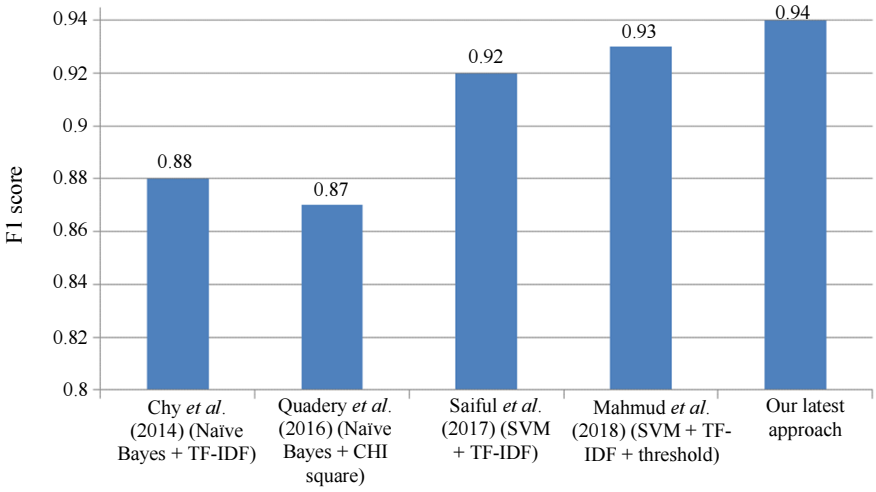

Fig. 10: Comparing F1 score

Conclusion

To conclude we say that this topic of text categorization is explored to quite an extent in this research. Many challenges are faced while implementing the neural network, for example, the huge feature space to work with, the memory issue that needed to be solved. But there are still scopes for improvement. For example, our future work will consist of using Deep Learning techniques such as RNN and using Word Embedding as our feature selection technique. But as there were not many works regarding Bengali document categorization using neural networks we used it and showed that using features efficiently good accuracy can be achieved without using advanced algorithms.

Acknowledgement

We would like to thank the SCDN lab of Shahjalal University of Science and Technology for providing us with the corpus. Also thanks Md Saiful Islam of Shahjalal University of Science and Technology for his valuable bits of advice throughout this research.

Author's Contributions

Quazi Ishtiaque Mahmud: Contributing to the conceptualization of the research, developing the methodology**,** analyzing and investigating different aspects of the model, preparing the final manuscript.

Noymul Islam Chowdhury: Studying related works in the field, contributing to data collection and preparation and data cleansing, contributing to the proofreading of the manuscript.

Md Masum: Supervising the research.

Ethics

It is testified by the authors that this article has not been published anywhere else and contains no ethical issues.

References

Al-Salemi, B. and M.J. Ab Aziz, 2011. Statistical bayesian learning for automatic Arabic text categorization. J. Comput. Sci., 7: 39-45. DOI: 10.3844/jcssp.2011.39.45

Bawaneh, M.J., M.S. Alkoffash and A.I. Rabea, 2008. Arabic text classification using K-NN and Naive Bayes. J. Comput. Sci., 4: 600-605. DOI: 10.3844/jcssp.2008.600.605

Chekima, K. and P. Anthony, 2011. Categorizer agent for electronic computer science academic papers. Am. J. Econ. Bus. Admin., 3: 213-218. DOI: 10.3844/ajebasp.2011.213.218

Chy, A.N., M.H. Seddiqui and S. Das, 2014. Bangla news classification using naive Bayes classifier. Proceedings of the 16th International Conference Computer and Information Technology, Mar. 8-10, IEEE Xplore Press, Khulna, Bangladesh. DOI: 10.1109/ICCITechn.2014.6997369

Dua, D. and C. Graff, 2019. UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA.

Hayes, P.J., L.E. Knecht and M.J. Cellio, 1988. A news story categorization system. Proceedings of the 2nd Conference on Applied Natural Language Processing, (NLP' 88). pp: 9-17. DOI: 10.3115/974235.974238

Islam, M.S., F.E. Jubayer and S.I. Ahmed, 2017. A support vector machine mixed with TF-IDF algorithm to categorize Bengali document. Proceedings of the International Conference on Electrical, Computer and Communication Engineering, Feb. 16-18, IEEE Xplore Press, Cox's Bazar, Bangladesh. DOI: 10.1109/ECACE.2017.7912904

Joachims, T., 1998. Text categorization with support vector machines: Learning with many relevant features. Mach. Learn. ECML, 98: 137-142. DOI: 10.1007/bfb0026683

Lam, S. and D.L. Lee, 1999. Feature reduction for neural network based text categorization. Proceedings of the 6th International Conference on Advanced Systems for Advanced Applications, Apr. 21-21, IEEE Xplore Press, Hsinchu, Taiwan. DOI: 10.1109/dasfaa.1999.765752

Mahmud, Q.I., N.I. Chowdhury and M. Masum, 2018. Reducing feature space and analyzing effects of using non linear kernels in SVM for Bangla news categorization. Proceedings of the International Conference on Bangla Speech and Language Processing, (SLP' 18). DOI: 10.1109/icbslp.2018.8554844

Mansur, M., N. UzZaman and M. Khan, 2005. Analysis of n-gram based text categorization for Bangla in a newspaper corpus. Center for Research on Bangla Language Processing, BRAC University.

Marsland, S., 2014. Machine Learning: An Algorithmic Perspective. 2nd Edn., CRC Press, ISBN-10: 1466583339, pp: 457.

Mesleh, A.M., 2007. Chi square feature extraction based SVMS Arabic language text categorization system. J. Comput. Sci., 3: 430-435. DOI: 10.3844/jcssp.2007.430.435

OSBC, 2020. Open source Bengali corpus Bangla dataset (Corpus).

Pedresen, J.O. and Y. Yang, 1997. A comparative study on feature selection in text categorization. Proceedings of the International Conference on Machine Learning, (CML' 97).

Platt, J., 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Microsoft Research.

Quadery, F., A.A. Maruf, T. Ahmed and M.S. Islam, 2016. Semi supervised keyword based Bengali document categorization. Proceedings of the 3rd International Conference on Electrical Engineering and Information Communication Technology, Sept. 22-24, IEEE Xplore Press, Dhaka, Bangladesh. DOI: 10.1109/ceeict.2016.7873040

Ruiz, M.E. and P. Srinivasan, 1999. Hierarchical neural networks for text categorization (poster abstract). Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (DIR' 99). pp: 281-282 DOI:10.1145/312624.312700

Srivastava, D.K. and L. Bhambhu, 2010. Data classification using support vector machine. J. Theoret. Applied Inform. Technol.

Subramaniyaswamy, V. and S.C. Pandian, 2012. An improved approach for topic ontology based categorization of blogs using support vector machine. J. Comput. Sci., 8: 251-258. DOI: 10.3844/jcssp.2012.251.258

Urmi, T.T., J.J. Jammy and S. Ismail, 2016. A corpus based unsupervised Bangla word stemming using N-gram language model. Proceedings of the 5th International Conference on Informatics, Electronics and Vision, May 13-14, IEEE Xplore Press, Dhaka, Bangladesh. DOI: 10.1109/iciev.2016.7760117

Wu, T.F., C.J. Lin and R.C. Weng, 2004. Probability estimates for multi-class classification by pairwise coupling. J. Mach. Learn. Res., 5: 975-1005.

Zhang, M.L. and Z.H. Zhou, 2006. Multilabel neural networks with applications to functional genomics and text categorization. IEEE Tran. Knowledge Data Eng., 18: 1338-1351. DOI: 10.1109/tkde.2006.162