Bangla Article Classification With TensorFlow

Xinyue,ZHANG
HKUST
Big Data Technology
20750194
xzhangfa@connect.ust.hk

Chennan,FU
HKUST
Big Data Technology
20710780
cfuac@connect.ust.hk

Di,WU
HKUST
Big Data Technology
20733885
dwuau@connect.ust.hk

***Abstract*—This literature is a demonstration of using TensorFlow for text classification in non-English/local languages. Here we choose Bengali as the local language to solve a multi-class classification task where we classify Bengali news articles in 5 categories. The word embedding method - FastText[1] is applied to extract meaningful features. TensorFlow's pretrained embedding exporter can convert the word embedding to a text embedding module first and then use the module to train a classifier to build a small sequential model which is a linear stack of layers. As we can see in the classification_report, finally we gain a 0.96 precision and accuracy after training data set only 5 epochs.**

***Index Terms*—NLP; Document Classification; FastText; Deep Learning.**

I. Introduction

The proliferation of unstructured textual data attracts the natural language research community to extract knowledge by mining textual data. The most common and well-studied problem is to categorize the textual documents. Document classification has paramount importance on several applications like searching, filtering, and organizing the textual documents. During the last decades, several statistical and machine learning approaches have been utilized to extract meaningful features to accurately classify the textual documents.For example, Bag of words, TF-IDF [2], Word embedding(Word2Vec [3]) features have been extracted from the text data in order to train some supervised learning model, for instance, SVM, KNN or Naive Bayes, for classifying the documents. Moreover, various deep learning algorithms, such as Convolutional Neural Network(CNN) and Long Short TermMemory(LSTM) [4], have also been utilized to extract valuable features from the unstructured textual data in order to categorize the documents.

Despite several large data sets on different other languages are available, a few data sets are constructed for the Bengali document classification purpose.Furthermore, most of the state-of-the-art works, which considered the Bengali article classification task, only consider Bag of words or TF-IDF features or Word2Vec and thus did not consider FastText.

In this work, we use a large corpus of Bengali documents, BARD: Bengali Article Data set. Additionally, we also perform an extensive statistical analysis to extract the textual relationships across the different documents categories. Finally create a small Sequential model which is a linear stack of layers to train the above data set and use trained model to complete classification.

The major contributions of this work are summarized below:

- To the best of our knowledge, we selected the largest Bangla articles data-set, which consists of around 3,76,226 articles labeled with five document categories.
- We performed an extensive statistical analysis in order to support word based features usage in training the supervised learning model.
- We adopted the fastText method on the BARD data-set and used a small sequential network to achieve 96% accuracy on the premise of training only 5 epochs

The remaining parts of the paper are organized as follows: Related Works like details of BARD data set as well as FastText have been presented in Section II. Subsequently, the Bengali article classification model and its performance are presented in Section III. Then, in Section IV, complete the prediction. And compare with state-of-art works in section V. Finally, we conclude this work with future directions in Section VI.

II. Related Works

Automated classification of text documents in various languages is a well-studied area. Several supervised learning algorithms have been used to classify text documents, for example,in [6] the author utilizes Support Vector Machine to learn the textual features for document classification. Moreover, Naive Bayes [7] and KNN [8] have been employed in the state-of-the-art works. In addition, word2vec embeddingwith SVM is utilized [9] for categorizing English documents.Furthermore, deep learning models, such as CNN [10], have also been introduced in the literature for this purpose.Compared to other languages, a very few works have been addressed done for Bangla article classification task.In [10], author analyzes the efficiency of n-gram based text categorization for Bangla language with one=year news corpus of Prothom Alo newspaper. Moreover, Naive Bayes based bangla text classification model has been proposed in [11].In addition, in [12], author compares the performance of four supervised learning Methods: Decision Tree, K-Nearest Neighbor, Naive Bayes, and Support Vector Machine (SVM)for Bangla documents categorization. In this work, TF-IDF have been used to construct feature vector by utilizing 1000web documents. Similarly, in [13], author uses TF-IDF to train a supervised learning model on

1960 web documents.To the best of our knowledge, all the state-of-art works, who addressed the Bangla text classification task, utilized a very small corpus of articles which is not very effective to train a supervised learning model. Furthermore, fastText have not been studied to build a supervised model for Bangla article categorization task.

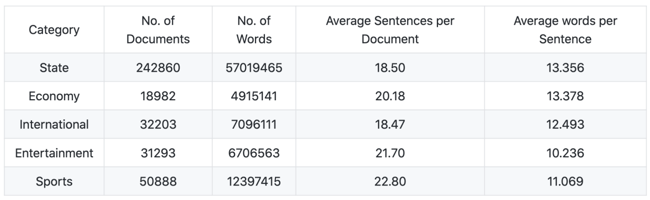
In the following part, data preprocessing is included and then present the details of the proposed data set BARD: Bengali Article Data set. Subsequently, introduce fastText which is used to convert word to vectors.

A. Data Preprocessing

Before tokenizer, remove all the punctuation and digits(1,2,3...). We tokenized the articles using space as delimiter. At the end of this step, we got a set of words and their frequency in each article.

Most frequent words, namely stop words, do not help to categorize the articles. Hence, remove around 25 most frequent words from all the articles and performed the statistical analysis again on the filtered data set, which is presented in Fig2. Now, this filtered frequency distribution shows that each category has some unique distribution of words, which may contribute more to categorize the articles.

B. Data set



Category	No. of Documents	No. of Words	Average Sentences per Document	Average words per Sentence
State	242860	57019465	18.50	13.356
Economy	18982	4915141	20.18	13.378
International	32203	7096111	18.47	12.493
Entertainment	31293	6706563	21.70	10.236
Sports	50888	12397415	22.80	11.069

Fig. 1. Data Set Details.

The distribution of 3,76,226 articles on five categories is shown in Fig1, along with the total number of words in each category. Except the State category, the distribution of articles on different are quite balance. However, in the statistical analysis, we have found out that the State category's articles have the almost unique textual property which can help the supervised learning model to accurately separate this articles from the other category. Additionally, the average number of sentence per document and the average number of words per sentence in each category are quite consistent, which are depicted in Fig1.

C. Splitting data set

To create a Data set using generator we first write a generator function which reads each of the articles from file_paths and the labels from the label array, and yields one training example at each step. We pass this generator function to the tf.data.Dataset.from_generator method and specify the output types. Each training example is a tuple containing an article of tf.string data type and one-hot encoded label. We split the data set with a train-validation split of 80-20 using the skip and take method.

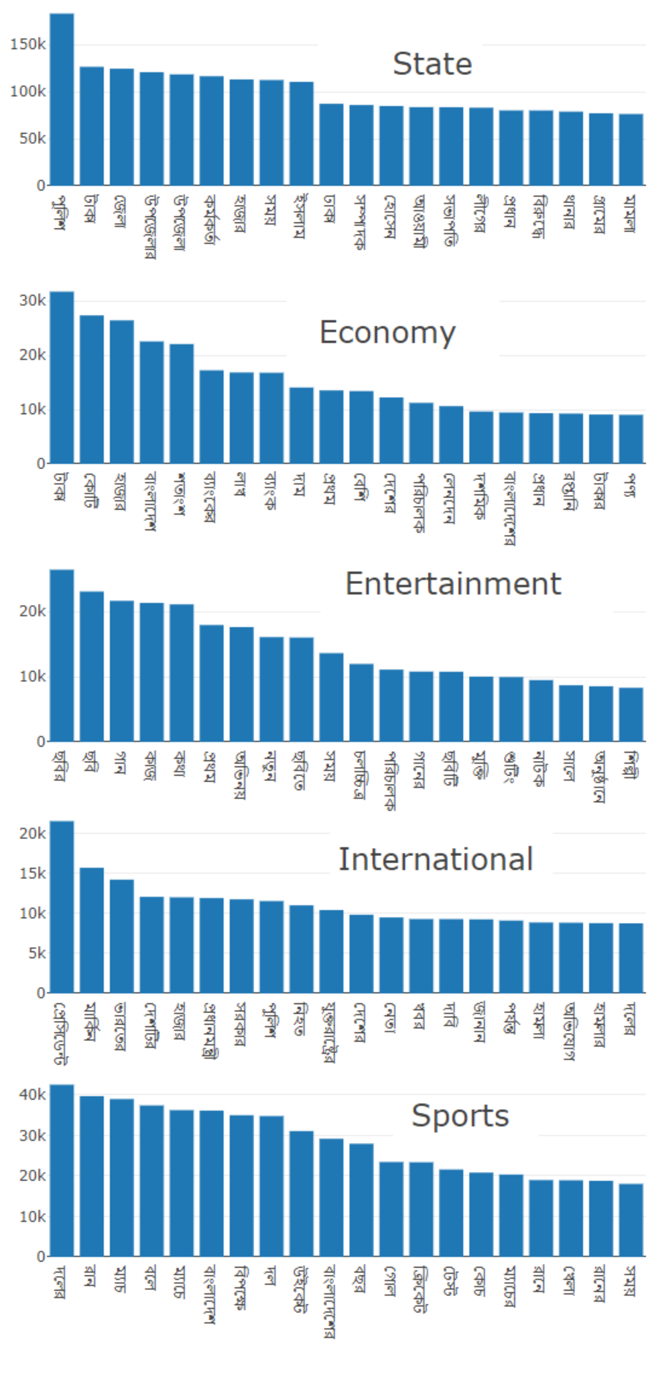


Fig. 2. After removing stop words.

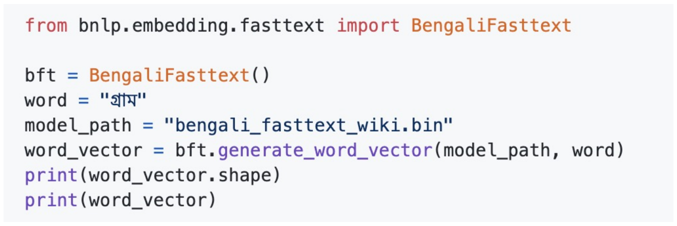
D. FastText

The FastText model was first proposed in the paper Enriching Word Vectors with Sub-word Information and it was based on the skip-gram model. The key idea is replacing a simple word vector with the sum of vectors of n-grams in a word.

In fact, if a word appears less frequently, the quality of the vector obtained will not be ideal. To solve this problem, FastText uses sub-word information to solve it. For example, for a word, 'love', its sub-word is all n-grams of 'love' itself.

When n equals to two, the two-gram of ‘love’ is ’lo’, ’ov‘, ’ve’. The suffixes are usually added to the method, so it may become love, then all its two-grams are #l, lo, ov, ve, e#.

As mentioned earlier, FastText uses the skip-gram model of Word2Vec, and the main difference lies in its characteristics. Word2Vec uses word as the most basic unit, that is, predicting other words in the context through the central word, while the sub-word model uses n-gram as the unit. In order to improve efficiency, low-frequency n-grams are generally filtered. The value of n in FastText is 3 6. In this way, each word can be expressed as a string of n-grams, and the embedding of a word is expressed as the sum of all n-grams.

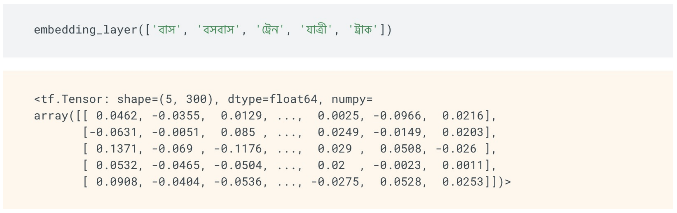


```
from bnlp.embedding.fasttext import BengaliFasttext

bft = BengaliFasttext()
word = "গ্রাম"
model_path = "bengali_fasttext_wiki.bin"
word_vector = bft.generate_word_vector(model_path, word)
print(word_vector.shape)
print(word_vector)
```

Fig. 3. Example code for FastText.

The whole word embedding process is: sub-words are formed into a dictionary, and their n-grams are mapped into integers from 1 to k through the hash function. A word is composed of the index in the dictionary and the hash value of its n-grams. But in the actual operation of this project, we only use the index encoding as the key of the query vector, and the final output is still a vector like the output of Word2Vec.We used FastText trained by Facebook research. Each vector has a dimension of 300.



```
embedding_layer(['বাস', 'বসবাস', 'ট্রেন', 'যাত্রী', 'ট্রাক'])
```

```
<tf.Tensor: shape=(5, 300), dtype=float64, numpy=
array([[ 0.0462, -0.0355,  0.0129, ...,  0.0025, -0.0966,  0.0216],
       [-0.0631, -0.0051,  0.085 , ...,  0.0249, -0.0149,  0.0203],
       [ 0.1371, -0.069 , -0.1176, ...,  0.029 ,  0.0508, -0.026 ],
       [ 0.0532, -0.0465, -0.0504, ...,  0.02  , -0.0023,  0.0011],
       [ 0.0908, -0.0404, -0.0536, ..., -0.0275,  0.0528,  0.0253]])>
```

Fig. 4. FastText for specified words in our project.

This brings two benefits:

- For low-frequency words, the generated word vectors will have a better effect. Because their n-grams can be shared with other words.
- For words outside the trained vocabulary, their word vectors can still be constructed. That is, their n-gram vectors are superimposed.

III. Model Training and Evaluation

A. *Introduction of Model*

After processing the data set, it’s time to train them to achieve a satisfactory result. In this project we use the module to train a classifier with tf.keras which is high level user friendly API to build deep learning models.

Actually keras itself is not capable of low-level computing. Hence, it should be combined with TensorFlow which is a open source machine learning framework based on python and also the most common back-end of the keras. The sequential model we finally applied containing three kinds of layers as follows:

- Input layer: the layer is considered as an entry point into a Network. Note that it is generally recommend to use the functional layer API via Input which creates an input layer without directly using input layer.
- Embedding layer: after obtaining the word embedding from FastText, we use embedding exporter and run the exporter script on our embedding file .Then according to hub.KerasLayer, we get the embedding layer.
- Dense layer: it is a commonly used fully connected layer. The fully connected layer is to identify features for classification. The number of neurons increases, the complexity of the model increases; the number of fully connected layers increases, and the nonlinear expression ability of the model increases. In theory, it can improve the learning ability of the model. Under other ideal conditions, if Training Accuracy is high and Validation Accuracy is low, it means over-fitting. Try to reduce the number of layers or parameters. If the Training Accuracy is low, it means that the model is not well studied. Try to increase the parameters or the number of layers.

In these two hidden layers, the neurons needs to be activated. Relu which is rectified linear unit activation function is applied in the model on account of its characteristics.

Before training the data set, it is important to deploy the loss function and optimizer which is to solve for minimum loss. Keras provides abundant losses including mean-squared-error, mean-absolute-error, mean-absolute-percentage-error, categorical cross-entropy etc. Finally, SparseCategoricalCrossentropy is implemented considering the cross-entropy is more suitable for classification problem.

B. *Evaluation*

After training we can visualize the accuracy and loss curves for training and validation data using the history object returned by the fit method which contains the loss and accuracy value for each epoch.

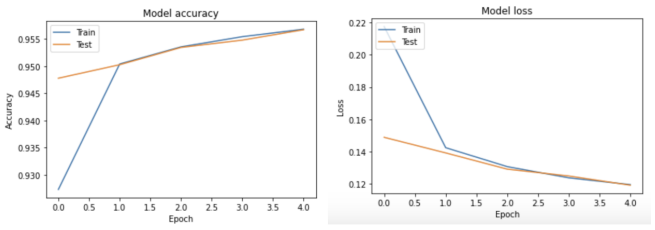


Fig. 5. Accuracy & loss values.

IV. Prediction

This step we get the predictions for the validation data and check the confusion matrix to observe the model's performance for each of the 5 classes, so as to test the rationality of our model.

Randomly read all the strings of the three files through file_paths, predict the category to which they belong, output the ground truth category and the predicted category. In the process of forecasting, the np.argmax, which is a method in numpy that returns the index of the maximum value, has been used to return the class label which holds maximum probability in the n-dimensional array.

Next compare our predictions with the correct labels for validation data in order to get the classification_report, which is a text report showing the main classification metrics.

Generally, in order to evaluate the classification performance of a classifier, some evaluation indicators need to be introduced. To assess the performance of the prediction model we utilized Precision, Recall and F1-score as evaluation metrics.

Precision refers to the proportion of real positive samples in the samples judged by the classifier as positive, that is, how many of all samples judged as positive by the classifier are real positive samples. Calculated as follows:

$$Precision = \frac{TP}{TP + FP}[5]. \tag{1}$$

Recall refers to the proportion of positive samples correctly determined by the classifier to the total positive samples, that is, how many of all positive samples are judged by the classifier as positive samples. Calculated as follows:

$$Recall = \frac{TP}{TP + EN}. \tag{2}$$

$$F1- \text{ score } = \frac{2 * (\text{ Recall } * \text{ Precision })}{\text{Recall } + \text{ Precision}}. \tag{3}$$

Here, TP = True Positives , FP = False Positives and FN =False Negatives.

In general, the higher the accuracy, the lower the recall rate. In order to evaluate a classifier comprehensively, F-score is introduced. However, the evaluation indicators discussed above are applicable to two classification problems and we are studying a multi-class problem, so we add the macro average (averaging the unweighted mean per label) and weighted average (averaging the support-weighted mean per label).

As we can see in the TableI, our model performs well in the class of state and sports, all indicators reached an average of 0.98. However, it does relatively poor performance in economic categories only up to 80%, which may be the amount of support, the number of occurrences of each label, is too small. Overall, we gain a 0.96 accuracy after training.

V. Comparison

In TableII, the performance of state-of-art works, we can easily identify that neural network with word2vec is superior(96%) to other supervised classification models. The reason behind is that word2vec can capture the semantic and syntactical features of the words in the text.

TABLE I
Performance of our model

	Precision	Recall	F1-score	Support
Economy	0.79	0.82	0.81	3897
Sports	0.98	0.99	0.99	10204
Entertainment	0.92	0.94	0.93	6256
State	0.97	0.97	0.97	48512
International	0.93	0.93	0.93	6377
Accuracy			0.96	7524
Macro avg	0.92	0.93	0.92	75246
Weighted avg	0.96	0.96	0.96	75246

TABLE II
Performance of state-of-art model

Features	Model	Accuracy
Word2Vec	Logistic regression	0.95
Word2Vec	Neural Network	0.96
TF-IDF	Logistic regression	0.94
TF-IDF	Neural Network	0.96
TF-IDF	SVM	0.89

From TableI, we can clearly observe that our simple classification model have outperformed the state-of-the-art Bangla article classification models. Because the model in state-of-art works are adjusted well, while our model only composed of five layers and only fit 5 epochs.

VI. Summary and Future work

In this work, we solved the problem of multiple classification of Bengali news articles. During the process, it was proposed to use FastText for word embedding, and chose to implement the whole project on Tensorflow. The final experimental result 96% shows that our model achieves a good classification effect on Bengali news articles.

In future, if we attempt to set trainable as True, achieving higher accuracy in less time may come true. Because our original set means the embedding weights will not be updated during training.

References

[1] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word-vectors with subword information," CoRR, vol. abs/1607.04606, 2016.

[2] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," Inf. Process. Manage., vol. 24, no. 5, pp. 513–523, Aug.1988.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean,"Distributed representations of words and phrases and their composi-tionality," in Advances in Neural Information Processing Systems 26,C. J. C. Burges, L. Bottou, M. Welling, Z.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural-Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[5] Olson, David L.; and Delen, Dursun (2008); Advanced Data Mining Techniques, Springer, 1st edition (February 1, 2008), page 138, ISBN 3-540-76916-1.

[6] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in Proceedings of the 10th EuropeanConference on Machine Learning, ser. ECML'98. Berlin, Heidelberg:Springer-Verlag, 1998, pp. 137–142.

[7] I. Rish, “An empirical study of the naive bayes classifier,” Tech. Rep.,2001

[8] V. Tam, A. Santoso, and R. Setiono, “A comparative study of centroid-based, neighborhood-based and statistical approaches for effective document categorization,” in ICPR, 2002.

[9] J. Lilleberg, Y. Zhu, and Y. Zhang, “Support vector machines and-word2vec for text classification with semantic features.” in ICCI*CC,N. Ge, J. Lu, Y. Wang, N. Howard, P. Chen, X. Tao, B. Zhang, andL. A. Zadeh, Eds. IEEE Computer Society, 2015, pp. 136–140.

[10] Y. Kim, “Convolutional neural networks for sentence classification,” inProceedings of the 2014 Conference on Empirical Methods in NaturalLanguage Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar,A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp.1746–1751.

[11] A. N. Chy, M. H. Seddiqui, and S. Das, “Bangla news classification us-ing naive bayes classifier,” in 16th Int’l Conf. Computer and InformationTechnology, March 2014, pp. 366–371.

[12] A. K. Mandal and R. Sen, “Supervised learning methods for bangla webdocument categorization,” CoRR, vol. abs/1410.2045, 2014.

[13] A. Dhar, N. S. Dash, and K. Roy, “Application of tf-idf feature forcategorizing documents of online bangla web text corpus,” in IntelligentEngineering Informatics. Singapore: Springer Singapore, 2018, pp.51–59.