Development of a Text Classification Framework using Transformer-based Embeddings

Sumona Yeasmin, Nazia Afrin, Kashfia Saif and Mohammad Rezwanul Huq[a]
Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh

Keywords: Natural Language Processing, Machine Learning, Classification, Transformer-based Embedding, Contextual Similarity.

Abstract: Traditional text document classification methods represent documents with non-contextualized word embeddings and vector space models. Recent techniques for text classification often rely on word embeddings as a transfer learning component. The existing text document classification methodologies have been explored first and then we evaluated their strengths and limitations. We have started with models based on Bag-of-Words and shifted towards transformer-based architectures. It is concluded that transformer-based embedding is necessary to capture the contextual meaning. BERT, one of the transformer-based embedding architectures, produces robust word embeddings, analyzing from left to right and right to left and capturing the proper context. This research introduces a novel text classification framework based on BERT embeddings of text documents. Several classification algorithms have been applied to the word embeddings of the pre-trained state-of-art BERT model. Experiments show that the random forest classifier obtains the highest accuracy than the decision tree and k-nearest neighbor (KNN) algorithms. Furthermore, the obtained results have been compared with existing work and show up to 50% improvement in accuracy. In the future, this work can be extended by building a hybrid recommender system, combining content-based documents with similar features and user-centric interests. This study shows promising results and validates the proposed methodology viable for text classification.

1 INTRODUCTION

Natural language processing (NLP) is the ability of a computer program to understand human language. NLP consists of two significant steps, namely data preprocessing and implementation of algorithms. Preprocessing is mainly done to understand the syntax, i.e., arrangements of words in a sentence to make grammatical sense, and the semantics, i.e., the use of and meaning behind words, of a sentence. Text document classification has become one of the key tasks in the field of NLP since it is very challenging to tag appropriate documents out of a massive corpus. Furthermore, text document classification has a wide range of applications, such as document organization, content identification, information retrieval, and similar document searching (Li et al., 2020).

Typically, descriptors or context are extracted from the document first. Then based on that same context, records are kept together, making it more convenient and easier to retrieve documents. It is not feasible to classify text documents using a traditional classification algorithm which requires the input to be a fixed-length feature vector. It is important to find a better way of representing large text documents. As classification algorithms can only work with numeric values, a challenge arises to represent a vast amount of text in numerical form, capturing both syntax and semantics of the document. There are several representations for vectorizing large textual datasets, such as Word2vec, and TF-IDF which can convert text documents into high dimensional vectors. Despite their popularity in vectorizing large text, these models have drawbacks as they cannot capture the sequence or semantic meaning.

The proposed text classification framework will follow the followings steps:

- Data preparation and preprocessing of a corpus of the text documents.
- Generation of embeddings large text documents using the BERT Model.
- Execution of classification algorithms over the document embeddings, i.e., vectorized forms of

[a] https://orcid.org/0000-0002-4222-2838

Yeasmin, S., Afrin, N., Saif, K. and Huq, M.
Development of a Text Classification Framework using Transformer-based Embeddings.
DOI: 10.5220/0011268000003269
In *Proceedings of the 11th International Conference on Data Science, Technology and Applications (DATA 2022)*, pages 74-82
ISBN: 978-989-758-583-8; ISSN: 2184-285X

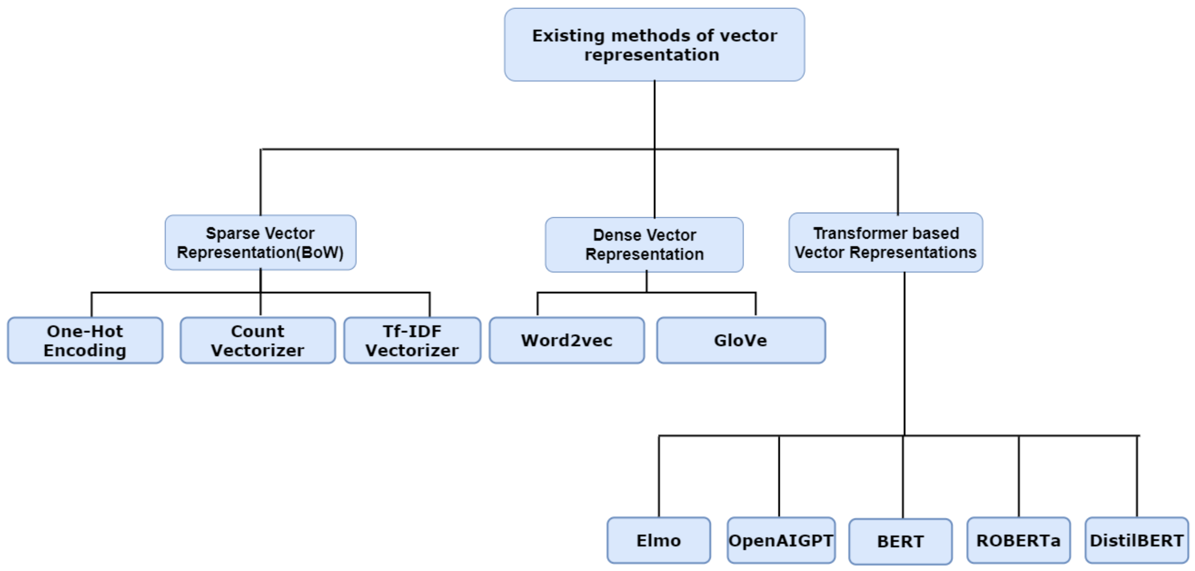
Existing methods of vector representation
Sparse Vector Representation(BoW)
Dense Vector Representation
Transformer based Vector Representations
One-Hot Encoding
Count Vectorizer
Tf-IDF Vectorizer
Word2vec
GloVe
Elmo
OpenAIGPT
BERT
ROBERTa
DistilBERT

Figure 1: A taxonomy of existing methods on word embeddings.

documents.

We introduce a transformer-based text classification framework by embedding the state-of-art BERT (Bidirectional Encoder Representations from Transformers) model, which has mitigated the drawbacks mentioned above. The powerful multi-head mechanism of BERT is used to vectorize text documents with contextual sentence embeddings. After the vectorization process, the text documents are classified with machine learning algorithms such as Decision Tree, Random Forest, and K-Nearest Neighbor (KNN).

The paper is organized as follows. Section 2 describes the background knowledge and existing work, followed by the proposed methodology in Section 3. Section 4 describes the datasets used for the validation, and Section 5 presents the results of our experiments. Eventually, we discuss the possible extensions of this work in the future and summarize the contributions of this study in Section 6.

2 BACKGROUND AND RELATED WORK

This section presents the background and related work relevant to text classification, including embeddings of words. Figure 1 visualizes a taxonomy of existing methods on word embeddings. Text embedding techniques have been broadly classified into three types, which are discussed below.

First, *sparse vector representations*, e.g. Bag-of-Words model, have been studied. Three types of vectorizers fall under this category. *One-Hot encoding* scheme considers the whole corpus of documents and vectorizes each document based on the existence of words, i.e., either true or false.

Count Vectorizer takes the count value of each occurring word in a document and transforms the document into a vector based on each word's frequency in the entire text. Count Vectorizer converts a collection of text documents to a vector of term/token counts.

TF-IDF (Qaiser and Ali, 2018) is a vector representation scheme calculating the importance or the relevance of a word document within a collection of documents. TF-IDF stands for Term Frequency Inverse Document Frequency. Despite being very popular vectorization techniques, both count vectorizer and TF-IDF cannot identify the relationships between words regarding linguistic similarity and word importance for analysis.

Another way to vectorize text documents is *dense vector representation* methods. *Word2vec* is one of the dense vector representation methods for producing word embeddings from a larger text corpus statistically. Another well-known text vectorizer is the *gloVe* vectorizer. The gloVe is an unsupervised algorithm for vector representations for words in which the training performs on aggregated global word-word co-occurrence statistics from a corpus. Word embeddings obtained from dense vector representations group semantically similar words together; however, they cannot capture the contextual meaning of the whole paragraph.

Since the previously presented approaches have significant limitations of not capturing the contextual meaning of the text document, the transformer-based

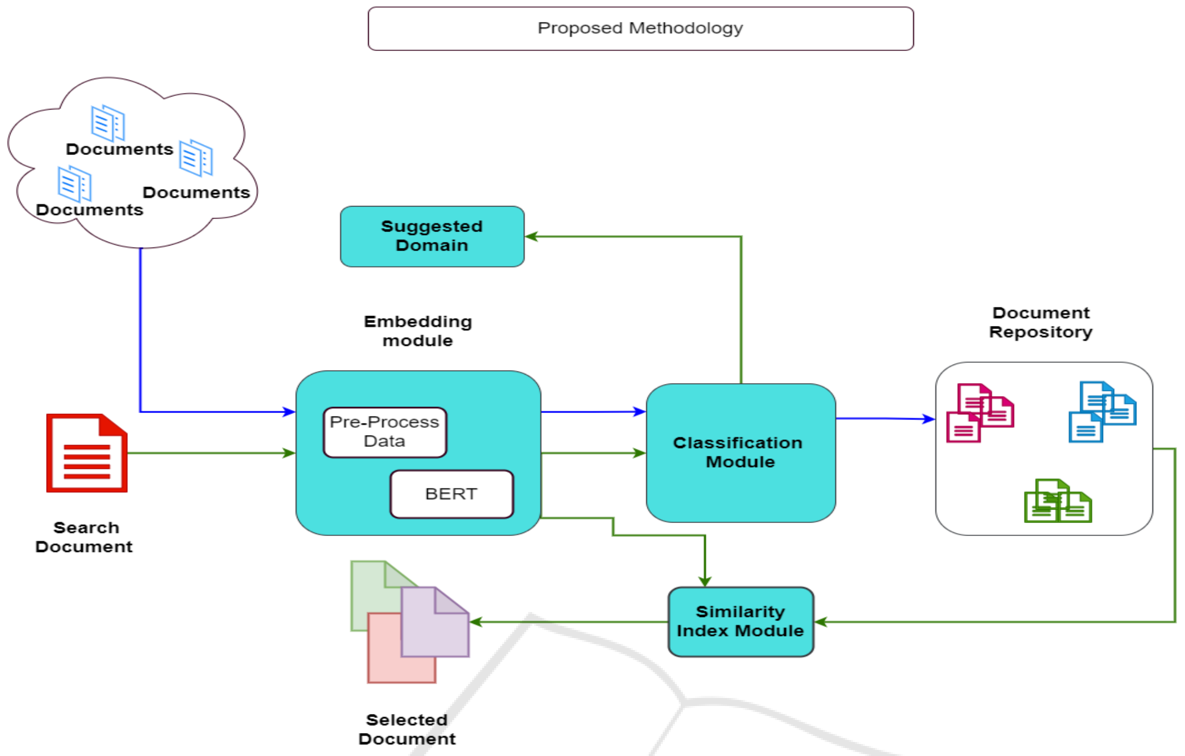

Figure 2: Flow Diagram of the Proposed Framework.

vector representations, which are state-of-the-art text embedding techniques, have been explored and studied in-depth. Transformers have robust word embedding schemes that capture both semantic and contextual meanings of a word. Transformers' idea is to completely handle the dependencies between the input and output with attention and recurrence. Attention mechanisms have become an integral part of captivating sequence modeling and transduction models in various tasks, allowing the modeling of dependencies without regard to their distance in the input or output sequence (Bahdanau et al., 2014), (Kim et al., 2017). *ELMo*, which stands for Embeddings from Language Model, is a bi-LSTM language model (Peters et al., 2018). It captures the deeply contextualized word embeddings created from the Language Models (Devlin et al., 2019). *OpenAI GPT* is a generative pre-trained language model trained on a diverse corpus of unlabeled text (Radford et al., 2018). Furthermore, the model is fine-tuned for each specific task.

Finally, the state-of-art *BERT* model is designed to pre-train deep bidirectional representations from an unlabeled text by jointly conditioning the left, right and proper context in all layers (Devlin et al., 2019). BERT possesses two significant steps to complete its task such as pre-training and fine-tuning. BERT has a few variations and extensions. One of them is RoBERTa which includes a careful evaluation of the effects of hyper-parameter tuning and training set size (Liu et al., 2019). DistilBERT is a small, fast, cheap, and light transformer model based on the BERT architecture (Sanh et al., 2020).

There exist several studies facilitating transformer-based embeddings for text documents. A clustering module of large text documents based on BERT's fine-tuning has been developed (Li et al., 2020). The authors used a weighted scheme and tested their algorithms on the Reuters Dataset (Lewis, 1997).

X-BERT, an extreme Multi-label Text Classification using the BERT model, has been introduced in (Chang et al., 2019). According to this paper, the proposed model, X-BERT, leverages both the label and input text to build label representations, which induces semantic label clusters to better model label dependencies.

DocBERT, a fine-tuned BERT model for Document classification based on the distilled BERT model, is reported in (Adhikari et al., 2019). In another work, the authors developed a clustering method of large documents based on an improved K-means algorithm with a density peak (Hu et al., 2021). Authors have used the embeddings of the BERT model for the clustering tasks. Haoxiang Sh et al. proposed document clustering based on BERT with data augmentation (Shi and Wang, 2020). The authors introduced self-supervised contrastive learning (SCL) and few-shot contrastive learning (FCL). These learning techniques with unsupervised data augmentation (UDA) are used for text clustering. The authors have also introduced data augmentation methods, back translation, and random masking. Another work by Daniela Godoy et al. proposed an intelligent agent called a personal searcher, which collects documents from the world wide web and filters out the results based on the high probability relevant to the target user (Godoy and Amandi, 2000).

Our in-depth, comprehensive analysis of these works discovered some limitations of these approaches. The study (Li et al., 2020) has not included any sentence-level feature in the weighted plan to improve its overall performance. The work reported in (Chang et al., 2019) introduced SCL(Self-supervised Constructive Learning). SCL shows a problem of negative sampling, which limits the performance of their algorithm. In another work (Adhikari et al., 2019), the authors did not provide distillation effects over a range of neural network architectures. Also, the authors did not explore the model compression techniques in the context of transformer models. Our proposed framework addresses the limitations mentioned earlier and bridges the research gaps to provide a robust text document classification framework.

3 PROPOSED METHODOLOGY

This section explains the methodology of our proposed text classification framework. Figure 2 depicts the overall flow diagram of the framework. The proposed framework consists of four central components: embedding module, classification module, similarity index module, and suggested domain module. Colored arrows denote different workflows of these components (see Figure 2).

First, documents are collected and scraped from the internet and are preprocessed, if required. Afterward, we provide the preprocessed data to the BERT model as input for producing word embeddings. The pre-trained BERT model is the core of the *embedding module*. The pre-trained language representation model BERT will generate contextualized sentence embeddings. Generally, this module maps each variable-length sentence in text documents to a 768-dimensional fixed-length sentence embedding.

BERT models' embeddings are then used to perform classification tasks. The embeddings from BERT are provided to the classification algorithms such as Decision tree, Random Forest, and KNN(k-Nearest Neighbor). The *classification module* comprising these algorithms is executed, and finally, The documents are classified based on their topic and context and stored within the cloud-based repository facilitated by the framework. The blue-colored path represents this whole workflow in Figure 2.

The following workflow of the proposed framework is to find contextually similar documents. Represented by green-colored paths, the target document is considered and passed through the embedding module. Then the *similarity index module* is used to retrieve similar documents from the repository. A predicted domain is also tagged based on the classification module. The similarity index module retrieves the most similar documents. We have used the cosine similarity measures, and the proposed framework retrieves the five most similar documents. The definition of cosine-based similarity is mentioned below, in which A and B are vectors representing two documents.

$$Cosine\ Similarity(A,B) = \frac{A \cdot B}{(|A|)(|B|)}$$

In this paper, we mainly focus on the classification tasks and discuss the performance of the classifiers in the coming sections.

4 DATASETS PREPARATION AND PREPROCESSING

This section presents the two datasets we have used in this paper for the experiments. First, the Reuters dataset as a benchmark dataset (Lewis, 1997), and second, the dataset we collected from the Daily Star news portal[1], a leading English newspaper in Bangladesh.

Table 1: Reuters Dataset Description - Training Size vs. Execution Time (used for comparison with (Li et al., 2020)).

Dataset Name	Dataset Size	Number of Topics	Time (in seconds)
DS4_v2	200	4	372.45
DS5_v2	500	5	784.57
DS8_v2	1000	8	4391.68
DS15_v2	5000	15	8786.54

4.1 Reuters Dataset

The Reuters dataset from Reuters-21578, Distribution 1.0 in Natural Language Toolkit (NLTK) (Lewis, 1997), was considered for testing the classification module. The Reuters-21578 dataset is one of the most

[1]https://www.thedailystar.net/

widely used data collections for text categorization research. Documents existing in the Reuters-21578 are from the Reuters newswire in 1987. We have taken this dataset as a benchmark for the experiments, analysis, and later framework validation. From Reuters-21578 datasets, the documents tagged with only one topic are considered in our experiments.

As described in Table 1, the whole dataset is partitioned into four different datasets. The dataset DS4_v2, DS5_v2, DS8_v2, DS15_v2 consist of 200, 500, 1000, 5000 documents respectively, with 4, 5, 8, 15 most frequent topics. The dataset size and number of topics are chosen based on the general principle of dataset preparation as described in work (Li et al., 2020). Later, the obtained experimental results of the classification module of the proposed framework are compared with the experimental results reported in (Li et al., 2020). The classifier's performance analysis is described in Section 5.2.

Table 2: Reuter Dataset Description (for evaluating performance of classifiers).

Dataset Name	Dataset Size	Number of Topics
DS1	200	5
DS2	500	5
DS3	1000	5
DS4	5000	5

Table 2 presents the Reuters dataset used in this study for evaluating the performance of different classification algorithms. As mentioned earlier, documents labeled with only one topic are considered while building these four different datasets. Furthermore, the topics with a document frequency count of more than 250 are included in these datasets. This threshold value is assumed to be an experimental value suitable for a balanced dataset. The five most frequent topics are chosen to build these datasets, as shown in Table 2.

Figure 3 shows the execution time of the BERT model's vector embedding generation for different dataset sizes of the Reuters dataset. This study has worked with four Reuters datasets (DS4_v2, DS5_v2, DS8_v2, DS15_v2). It has been observed that, with the increasing size of the dataset, the time taken by the BERT model increases at a higher rate. When the dataset size rises from 500 to 1000 samples, we see a sharp increase in the execution time of the BERT model. Interestingly, the execution time increases at a much lower ratio for larger datasets indicating that the pre-training of BERT takes less time with the larger dataset. This analysis implies that the execution time increment is logarithmic against the size of the dataset.

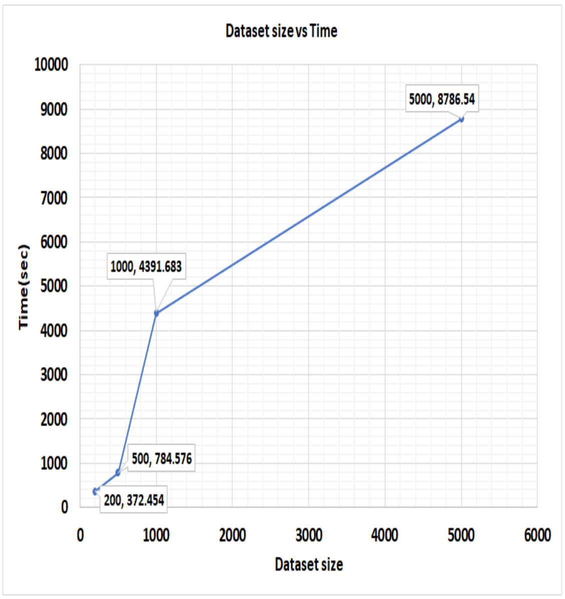

Figure 3: Reuters dataset size vs. execution time.

4.2 Daily Star Dataset

Table 3: Daily Star Dataset Description.

Dataset Name	Dataset Size	Number of Topics
DS415	415	7
DS212	212	7

As mentioned earlier in Section 4, the proposed framework's classifiers performance has also been evaluated using the Daily Star dataset (Yeasmin et al., 2022). The Daily Star portal is an online news portal containing hundreds of news articles on various domains. We have collected the datasets via web scraping. The news content under different domains were fetched using Python's Beautiful-Soup library[2]. Only the text part of each news article was considered. Next, the contents were merged with respective domain names to make it a complete and labeled dataset. The collected and preprocessed dataset has been divided into two sets, namely, DS415, DS212 containing 415 and 212 documents and associated domain names, respectively, for the experimental purpose of this paper. The documents pertaining to Business, Entertainment, Sports, Live living, Tech news, Youth, and Environment, a total of seven domains/topics from the Daily Star news portal, have been considered.

[2]https://beautiful-soup-4.readthedocs.io/en/latest/#

4.3 Dataset Preprocessing

Large text files can contain garbage values and characters that should be eliminated before producing word-level embeddings. Therefore, preparing and preprocessing the uncategorized raw text before providing them to the BERT model is a key task for a robust classification result. The following data preprocessing steps were performed over the datasets used for the experiments.

1. Stop words removal: Stop words are the most frequent word, and they can reduce the integrity of the context of a document. Therefore, Python's NLTK library functions are used to discard the stop word for strong contextual corpus generation. The entire document is divided into words and then if the word exists in the list of stop words provided by NLTK, it is removed.
2. Removing special, unexpected symbols: Uncategorized data can come with all kinds of symbols that are not expected during model training. Python's Regex function is used to remove special symbols and characters from the dataset.
3. Tokenizing the words: Tokenization is an important step before feeding the word embedding to the BERT model. Word tokenization is the process of splitting a large text into words. Tokenizing the words is needed because each word needs to be captured and subject to further analysis like classifying the document and counting them for a particular sentiment. The BERT pretrained model's tokenizer was used for this purpose.
4. Padding the token vectors: Padding is a special form of masking where the masked token are at the start or the end of a sequence. Padding makes all sequences in a batch fit a given standard length. The BERT model's padding schemes are used to pad the corpus.
5. Masking the tokens: Masking is a way to tell sequence-processing layers that some inputs are missing; this helps to train the model better. We have applied masking with the BERT model.
6. Embedding with the BERT model: The final step before applying the classification algorithms is to get the BERT's contextual word embeddings. The numeric representation of documents with appropriate padding and masking are provided to the BERT model to produce powerful contextual embeddings of documents.

Table 4: Accuracy (in percentage) of different classification algorithms on Reuters dataset.

Dataset Name	Dataset Size	Decision Tree	Random Forest	KNN
DS1	200	76.5	**88.0**	81.5
DS2	500	75.8	**87.0**	85.6
DS3	1000	67.7	**88.0**	88.5
DS4	5000	72.8	**90.0**	89.8

5 EXPERIMENTAL RESULTS

The experimental results of classification algorithms were obtained and analyzed based on two datasets, namely the Reuters Dataset and the Daily Star dataset in this paper. Classification algorithms such as Decision Tree, Random Forest, and KNN (k-nearest neighbors) were applied over these datasets. Furthermore, the results of the experiments have been validated by a linguistic expert showing the viability of the proposed framework.

The experimental results demonstrate that the random forest algorithm outperforms all other algorithms for the classification task and achieves around 90% accuracy for Reuters dataset and 75% accuracy for the Daily Star dataset. Furthermore, the classifiers performance of the proposed framework provides better accuracy than the work reported in (Li et al., 2020) demonstrating the robust and solid performance of the proposed framework. This section presents all the details in this regard.

5.1 Performance of Classifiers on Reuters Dataset

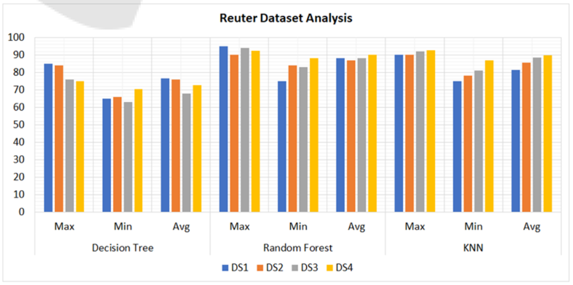

Figure 4: Accuracy distribution (max, min and avg.) of different classifiers on Reuters dataset.

This section presents the result analysis for the Reuters dataset with the proposed method. Several classification algorithms are applied to analyze the performance of the classifiers and the influence of the dataset size in the classification task.

It has been observed that the accuracy of the clas-

Table 5: Performance comparison in terms of accuracy with the target research (Li et al., 2020).

Dataset Name	Dataset Size	Topic number	Decision Tree	Random Forest	KNN	Target Accuracy
DS1	200	4	76.0	**91.5**	91.0	*75.0*
DS2	500	5	74.4	**86.4**	85.2	*63.0*
DS3	1000	8	66.2	**82.6**	81.5	*53.7*
DS4	5000	15	59.8	**84.7**	85.0	*67.9*

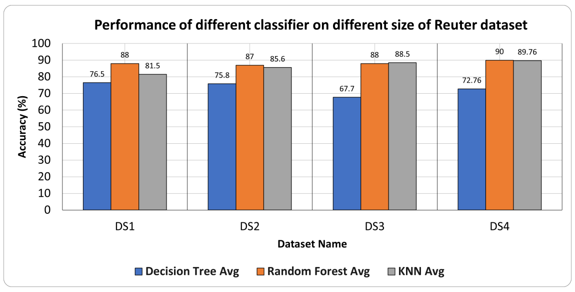

Figure 5: Average accuracy of different classifiers vs. different Reuters dataset.

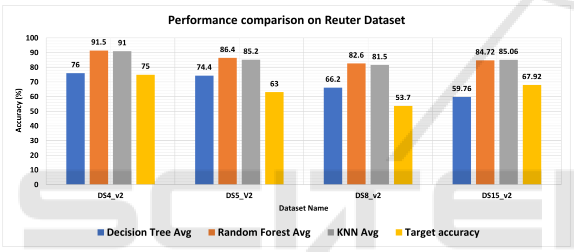

Figure 6: Performance comparison of research work's accuracy with the targeted research work on Reuters dataset.

sifiers increased when the dataset size also increased. The cross-validation method is used as a validation technique. Stratified 10-fold cross validation is used for the result analysis. The cross-validation helped us to achieve optimal results for classification algorithms used in this research. The overall classifier algorithm performance is shown in table 4. Random forest classifier outperforms all other classifiers in all cases. Figure 4 shows the maximum, minimum and average accuracy obtained for different classifiers over different partitions of Reuters dataset, used in this paper. It can be seen from Figure 4 that the random forest classifier provides the optimal performance in all cases. Figure 5 presents comparative visualization of the performance of the classifier algorithms in terms of average accuracy obtained from 10-fold cross-validation on the Reuters Dataset. The random forest classifier, yet again, outperforms all other algorithms.

5.2 Comparative Performance Evaluation on Reuters Dataset

In this section, the performance and classification accuracy of the proposed framework is compared with the weighted BERT model reported in (Li et al., 2020). The dataset mentioned in Table 1 is used for this comparative study. The proposed method achieves higher classification accuracy than the target research in (Li et al., 2020). The comparative analysis is presented in table 5. The proposed framework is simple yet effective. The classification module, proposed in this research, outperforms the accuracy reported in (Li et al., 2020).

Figure 6 provides a visual representation of the comparative analysis with the target study mentioned in (Li et al., 2020). As mentioned earlier in Section 4.1, the partitioning of the dataset was kept identical to the target research, reported in (Li et al., 2020), for a fair comparison and validation of the results. The classification module of the proposed framework achieves the best accuracy by applying the random forest classifier on all datasets. The accuracy of the random forest classifier of the proposed method is at least 22% higher than the accuracy reported in the target research. In the best case, the random forest classifier shows more than 50% accuracy than the work in (Li et al., 2020).

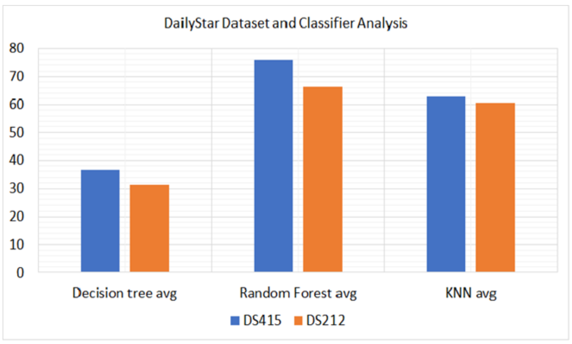

Figure 7: Different Classifier analysis on Daily star dataset.

Table 6: Unseen test data analysis of Daily Star dataset.

			Predicted Class Label		
Sample No.	System Generated Class Label	Domain Expert Identified Class Label	Decision Tree	Random Forest	KNN
1	environment	business	tech_news	business	environment
2	environment	environment	environment	business	environment
3	sports	sports	sports	sports	sports
4	sports	sports	tech_news	sports	sports
5	business	business	entertainment	business	business
6	business	tech_news	tech_news	business	environment
7	entertainment	entertainment	environment	business	sports
8	entertainment	entertainment	entertainment	entertainment	entertainment
9	tech_news	tech_news	tech_news	tech_news	tech_news
10	tech_news	business	tech_news	business	business
11	life_living	life_living	entertainment	life_living	life_living
12	life_living	life_living	environment	tech_news	tech_news
13	youth	tech_news	tech_news	entertainment	youth
14	youth	youth	entertainment	life_living	youth

Table 7: Accuracy (in percentage) of different classification algorithms on Daily Star Dataset.

Dataset Name	Decision tree	Random Forest	KNN
DS415	36.61	**75.92**	62.86
DS212	31.17	**66.47**	60.47

Table 8: Comparative analysis of classifiers based on System Generated and Domain Expert tagging.

	Predicted Accuracy (%)		
	Decision Tree	Random Forest	KNN
System Generated Class Label	35.7	50.0	64.2
Domain Expert Identified Class Label	42.8	57.1	64.2

5.3 Performance of Classifiers on Daily Star Dataset

This section presents the performance of the classification algorithms applied over the Daily Star dataset used in this paper. Two datasets, namely, DS415 and DS212, are used for this experiment. Stratified 10-fold cross-validation method is applied while obtaining the accuracy. The random forest classifier performs the best in both datasets, as described in table 7.

Figure 7 shows the visual representation of the classifiers performance over the Daily star dataset. Decision tree did not provide a good result while the random forest classifier achieved 75.9% accuracy in DS415 dataset which is the best among all classifiers.

Table 6 presents the class label prediction of the classifiers of the proposed framework over 14 unseen Daily Star news portal article which are considered as the test dataset. These articles have also been manually classified by a linguistic expert for a better understanding of the performance of the classifiers.

Table 8 shows the accuracy of the classifiers based on the unseen test dataset as mentioned earlier. The accuracy is measured in both ways - by using the system generated class labels and domain expert identified class labels. In both cases, KNN provides better accuracy than the decision tree and the random forest classifier. Interestingly, the performance of classifiers are better or at least equal when compared to the domain expert identified class labels than the system generated class labels. It shows that the classifiers of the proposed framework is better aligned with the domain expert's understanding implying the suitability of the framework for real, unseen test dataset. The authors of this paper will incorporate more unseen test data samples and apply the classification module over those samples, with the expectation of achieving a better and conclusive results.

6 CONCLUSION AND FUTURE WORK

This paper introduces a novel text classification framework through transformed-based embeddings. The study of embedding schemes of different models concluded that the transformer-based architectures are better suited for generating contextual word embeddings.

This research has applied the classification algorithms to a benchmark Reuters dataset and web-scrapped Daily Star dataset. The experiments with the random forest classifier and KNN show promising results. The random forest classifier achieves around 90% accuracy over the Reuters dataset and 75% accuracy over the Daily Star dataset. The performance of the classifiers has been compared with existing work and it shows up to 50% improvement in accuracy.

The extended version of the framework will include a hybrid recommender system integrating user-centric preferences. The hybrid recommender system facilitates both content-based and collaborative filtering. Moreover, multiple repositories of text documents can be added to the framework for smooth user experience of information browsing. The framework will eventually allow users to browse and submit documents for finding contextually similar records. The framework will will be more robust as the time passes by through a monitoring scheme, keeping the updated user profiles based on the search history. The authors of this paper have already been working towards building such an extended system.

In sum, the proposed text classification framework shows great results for classification tasks and envisions an all-around platform for text processing jobs including classification, clustering and document recommendation. The framework brings state-of-art algorithms, tools and techniques under the same umbrella to provide a unique user experience in text categorization.

ACKNOWLEDGEMENT

We want to thank S. M. Keramat Ali, Assistant Engineer at the Department of Transportation in California, USA, completed MA in Language & Linguistics from the University of Dhaka, for supporting us to validate the classification results on Daily star dataset.

REFERENCES

Adhikari, A., Ram, A., Tang, R., and Lin, J. (2019). Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Chang, W.-C., Yu, H.-F., Zhong, K., Yang, Y., and Dhillon, I. (2019). X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers. *arXiv preprint arXiv:1905.02331*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (24 May 2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*.

Godoy, D. and Amandi, A. (2000). Personalsearcher: an intelligent agent for searching web pages. pages 43–52.

Hu, W., Xu, D., and Niu, Z. (2021). Improved k-means text clustering algorithm based on bert and density peak. pages 260–264.

Kim, Y., Denton, C., Hoang, L., and Rush, A. M. (2017). Structured attention networks. *arXiv preprint arXiv:1702.00887*.

Lewis, D. (1997). Reuters-21578 text categorization test collection, distribution 1.0. *http://www.research.att.com/lewis/-reuters21578.html*.

Li, Y., Cai, J., and Wang, J. (2020). A text document clustering method based on weighted bert model. 1:1426–1430.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (26 July 2019). Roberta: A robustly optimized bert pretraining approach. *ArXiv*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. in naacl.

Qaiser, S. and Ali, R. (2018). Text mining: use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181(1):25–29.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding with unsupervised learning.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (March 2020). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *Hugging Face*.

Shi, H. and Wang, C. (2020). Self-supervised document clustering based on bert with data augment. *arXiv preprint arXiv:2011.08523*.

Yeasmin, S., Afrin, N., Saif, K., and Huq, M. R. (2022). Daily star dataset. *https://github.com/Sumona062/Daily-Star-Datasets.git*.