[1]Chayanika Talukdar
[2]Shikhar Kumar Sarma

Hybrid Model for Assamese Document Classification using Doc2vec for feature extraction

Journal of Electrical Systems

Abstract: - Document level categorization is challenging for texts with a huge number of words, often indicating contradicting categories. This research is particularly useful for vast amount of unorganized digitized text, produced as a side effect of the exponential growth of internet. Many text classification studies have been carried out using various machine learning and deep learning techniques, however, mainly for short text. In this study, we will categorize Assamese documents, a subject that has mostly gone unexplored until now. Here, we propose a hybrid model that combines the advantages of two most popular deep learning models- the CNN and LSTM. Also, Doc2vec has been used to convert documents into numeric vectors of 3 dimensions- 100, 128 and 300. When evaluated on the prepared data set of 780 Assamese documents, the model was found to have worked effectively with an accuracy of 96.5% and an F1-score of 96%, for the vectors with dimension value of 300.

Keywords: CNN, LSTM, Doc2vec, Assamese, SVM, LR, Hybrid.

I. INTRODUCTION

Text categorization involves assigning predefined categories to textual documents written in natural languages. The aim of this task is to identify the text's subject and predict the class to which it may belong. It may present a conceptual outlook of the document collections and is critical for many information retrieval applications. Text classification finds application in a wide range of fields such as sentiment analysis [1], topic categorization, spam filtering [2] and news classification, to name a few.

This field is attracting a great deal of interest from scholars and researchers. Several machine learning algorithms such as Decision Tree [3], Naïve Bayes [4], Support Vector Machine (SVM) [5] etc. are used for text classification. But the emergence of deep learning techniques has advanced this analysis a step further. A lot of research has been undertaken to demonstrate the usefulness of Deep Learning, DL henceforth, in this field. As a matter of fact, CNN (Convolutional Neural Network) [6] and LSTM (Long Short-Term Memory) have proven to be quite effective in classifying documents. Moreover, in NLP (Natural Language Processing), LSTM [7] neural networks are also widely used. Given that CNN is an efficient feature-extracting approach and that it can be integrated more effectively into larger networks, the crucial job that needs to be done is to hybridize and combine these models, in order to maximize their respective advantages [8].

Using deep learning techniques on short texts has yielded impressive results. However, in contrast to classifying a word or a sentence, document classification is more complicated because it comprises of a greater number of words and it is quite a problem to retain the semantic association among the sentences.

With the growing use of internet, the volume of digitized Assamese documents has also increased in the last couple of years. This huge collection of texts can be of great use if preserved with appropriate labeling. However, despite the abundance of text data, there is still a scarcity of research on document classification for complex languages like Assamese, which is spoken by millions of people in a North-Eastern state of India, called Assam. This presents a unique challenge for researchers and practitioners alike due to the lack of freely accessible annotated datasets.

As machine learning techniques can only act upon numeric vectors, documents are usually represented as BOW(Bag-Of-Words) [9]. In BOW, every dimension represents a single word, while the vector's overall dimensionality measures the scope of the vocabulary. However, its main problem is that the resultant vectors have usually a very high dimensionality and are also sparse. Apart from this, it does not cover the semantics of the document.

Another easy yet effective technique for feature extraction based on the bag of words is TF-IDF(Term Frequency Inverse Document Frequency). The primary negative aspect that can be attributed to it is that it does not consider the context of the text, which is very important to understand the meaning of the statement. For example, in the statement “the boy is sitting on the bank of Ganga”, if we consider the word 'bank'(unigram), then

[1]* Corresponding author: Department Of Computer Science, NERIM,email-id:ctalukdar@gmail.com
[2] Author Department of Information Technology,Gauhati University, email-id: sks001@gmail.com

it appears to be ambiguous as it may imply a money transaction bank or a river bank. But if we consider “sitting on the bank of Ganga” (6-gram) then it becomes clear in which context the word 'bank' is used.

In order to enhance the representation of text and maintain the semantic representation of each document, neural embeddings were proposed by Bengio et al. [10] in 2003. They modeled it as a feed-forward neural network model. It represented a word as a real-valued vector of a specific dimension. The association between any two words is measured by using the distance that exists between their corresponding embedded vectors. Quoc Le et al. [11] extended this concept to learn document embeddings and proposed an unsupervised framework named Paragraph Vector, to address the problem related to semantic meaning.

In this research work, a novel approach for Assamese document classification has been proposed by integrating the power of Doc2Vec, CNN and LSTM. The Doc2Vec model is employed to convert documents into vectors, capturing the semantic and syntactic features of the Assamese texts. The resulting vectors are then fed into a CNN-LSTM architecture, which employs convolutional layers to capture local patterns in the data and LSTM layers to capture sequential dependencies. To compare the output produced by the proposed model, a single CNN and a LSTM were constructed separately. Two other traditional models, namely, Logistic Regression and Support Vector Machine were also used to classify the dataset. CNNs are particularly effective in capturing local patterns in data. In the context of text classification, it can effectively detect n-gram features and local word dependencies. However, it fails to grasp the consecutive interrelationships of words in text. LSTMs, on the other hand, are designed to capture long-range relationships in sequential data. They excel at modeling the context and comprehending the meaning of a document or sentence, but fail to retrieve the features simultaneously. Therefore, in order to benefit from the strong points of both, CNN and LSTM have been stacked together in this work.

The key contribution of this research can be highlighted in the following points-

- Constructed a dataset of Assamese documents from a huge unlabeled corpus and attached appropriate labels. Four categories of documents were used for this purpose.
- Applied Doc2vec to extract the semantic features from these documents so that the documents can be classified into their relevant categories.
- Built a hybrid model, named ADC, using a combination of the CNN and LSTM models.
- Built separately, a CNN model and an LSTM model, so as to compare their efficiency with ADC.
- Applied traditional machine learning methods like Logistic Regression and SVM (Support Vector Machine) to classify this dataset.
- Compared the results of the proposed CNN-LSTM model with the other models’ output to assess their efficiency in classifying unseen Assamese text documents.

The paper is organized as follows -

Section 2 starts with a short introduction to text classification and the process involved therein. Section 3 discusses some of the available classification algorithms. Section 4 discusses the evaluation technique for text classification. Section 5 focuses on the primary challenges faced in text classification. We conclude in Section 6.

II. Background and Context

A. Text Classification

Text categorization is a group of linguistic procedures applied in the artificial decoding of natural language, which are applied on digitized texts, such as news articles and publications etc. It is especially important to improve the organization of documents under specific heads. This can ease out the accessibility and filtering process. Text classification can be used at several granularity levels, specifically:

- Word level analysis: It establishes a word's orientation towards a particular class.
- Sentence level analysis: It establishes a sentence's orientation towards a particular class. It is frequently applied to text classification, mainly to classify news headlines.
- Document level analysis: This analysis establishes a document's orientation towards a particular class. This level is harder than the others because, as the word count rises, noise words also rise, which skews learning. This makes classification more challenging.

B. Word Embedding and Document Embedding

Word embedding is nothing but a word representational technique used in NLP. It represents words as dense vectors of real numbers [12]. Word2vec is a commonly used embedding technique that captures the meanings and contextual relationships between words by learning scattered depiction from a large corpus of text.

Word2Vec models are trained using two primary architectures: Skip-gram and Continuous Bag of Words (CBOW).

An expansion of the Word Vector, the Document Vector captures the entirety of an article or document in a single numerical vector so that similarities between articles or documents can be quickly determined. This approach enables the document representation via two primary models: the Distributed Memory model or DM model and the Distributed Bag of Words model or DBOW model. By leveraging these models, this vector technique enhances our ability to capture the essence of documents and effectively discerns the similarities across a corpus.

- DBOW constructs a vector by arbitrarily predicting each word's probability distribution in a document, using the document's identifier. This approach does not consider the word order.
- DM, as opposed to DBOW, guesses a term or word based on the context of the document. It attempts to anticipate a central word by randomly selecting a group of words in a paragraph using the document id as input.

C. CNN

Convolutional Neural Network (CNNs) is a sort of deep neural network that can recognize information in various positions with remarkable precision. CNNs are extensively utilized in the field of image classification. Of late, it has been successfully applied to many NLP tasks. CNNs are multilayer networks, with one layer's output being the subsequent layer's input. It typically comprises of three layers: an input layer, an output layer and a number of hidden layers.

D. LSTM

LSTM is a form of Recurrent Neural Network (RNN). This is particularly effective for tasks involving sequential data since it can capture long-range dependencies. LSTMs are designed to overcome the vanishing gradient problem of RNNs. It is the memory in LSTMs that allows for reading, writing and deleting data by virtue of three gates. The input gate or the first gate enables or blocks updates, whereas the second or forget gate shuts the neurons based on their relevance as determined by the algorithm's weights. The third component (Output Gate) is the regulatory gate for the neuron's output state.

III. EXISTING LITERATURE

Over the past couple of years, many researchers have come up with different solutions to the problem of automatic text classification. Some came up with solutions using traditional methods, while others gave solutions based on deep learning methods.

A. Text classification based on traditional methods

Shalini et al. in their work [13] proposed a method to categorize Hindi text documents into the two predefined classes. For extracting the features, the membership degree of each of the words were calculated and analyzed with the classes. Then these features were passed to the SVM for classification purposes.

In another study, Mahdaouy et al. [14] attempted to classify Arabic texts using two-way embeddings- word level and document level. They used CBOW, Skipgram and Glove for generating the word vectors and used the doc2vec model. They generated vectors of 6 different dimensions and then evaluated using the SVM on the OSAC data set.

In another study, Kim et al. [15] proposed a model where the feature set for each document was constructed using three feature extraction techniques, viz, TF-IDF, LDA and Doc2Vec. Then, based on the representation approach, 3 different types of classifier models were trained. The training set for the other two models with the confidently predicted label is once more expanded to include the document with the highest prediction score for one of these 3 models. In this way, multi co-training is carried out. To assess the model's efficiency, it was tested using Random Forest and Naïve Bayes algorithms.

Aubaid et al. in [16] came up with a rule-based classification technique for classifying documents into ten different categories. They used Doc2vec technique for generating feature vectors of the documents. They applied three different algorithms, namely, Jrip, One Rule, ZeroR on the Reuters and 20Newsgroup dataset.

Dadgar et al. in one of their studies [17] on the classification of English texts, applied TF-IDF to extract the features and used the SVM as the classification algorithms. When applied to the BBC and 20Newsgroup datasets, their research revealed accuracies of 97.84% and 94.93%, respectively.

Sharma et al. [18] demonstrated a method for classifying Assamese documents with Assamese WordNet. On Assamese documents, this approach achieved an accuracy of 90.27. Assamese WordNet searches for frequently used terms in Assamese documents. For each common term sysnet detected in WordNet, an extended form is identified and linked with it. It scans each predefined class for extended terms found in the testing document. It allocates the test document to the class with the largest number of matched terms.

Gogoi et al. [19] employed the Naïve Bayes classifier for classifying the Assamese documents, which were categorized into Sports, Politics, Law and Science. For their research they employed the multinomial Naïve Bayes approach on a dataset of 200 documents. Bag of words (BoW) was used to extract the features. They achieved a standard precision of 94.41% and a recall of 94.68%.

B. Text classification based on traditional methods

Ferdouse et al. [20] used LSTM in one study on Bengali text classification. They carried out the experiment on the Kaggle Bengali news articles dataset to obtain an accuracy rate of 84%.

Wanet et al. [21] suggested an approach for classifying long length legal documents. Before embedding the documents using Doc2vec, they first split each of the documents into multiple chunks to form multiple chunk embeddings. Then these are merged using a BiLSTM and then attention mechanism is applied for predicting the class.

Rhanoui et al. [22] suggested a technique for extracting sentiments related to documents. They used Doc2vec as a method for feature extraction and then passed these features to a hybrid model based on CNN and Bidirectional LSTM. They obtained an accuracy level of 90.66% accuracy.

Zhou et al. [23] in their research used TF-IDF for extracting the vital features from the text and then built the corresponding word vectors with the help of Word2vec model. They passed the resulting vectors to a hybrid model which they constructed using the CNN and LSTM. The model was applied on THUCNews and Taobao review.

Talukdar et al. [24] employed three models, namely the CNN, multichannel CNN and a hybrid model using CNN with SVM on a dataset that consisted of 634 Assamese documents. They used word2vec to convert the features into numerical vectors. Their proposed technique with CNN recorded the highest accuracy of 96%. Table 1 presents some works that employed varied embedding strategies.

IV. MATERIALS AND METHODS USED

The resources and techniques used in this work are listed in this section.

A. The Assamese dataset

The Assamese corpus created by Gauhati University's NLP Lab served as the foundation for the data set used in this study. This corpus contains numerous Assamese articles that cover different fields. This collection of articles is drawn from a number of well-known Assamese history books, newspapers and periodicals. It even contains some popular Assamese songs. The dataset was created by selecting a few files from this corpus. Some of the files had to be segmented as they contain huge amounts of data. Altogether, the 780 documents make up the entire dataset.

Two impartial annotation experts' assistance was sought for the manual labeling of the textual contents. The process of data annotation involved several iterations over a period of time. Four classes were identified in isolation to which each documents falls - Arts, Children, History and Sports. In light of the annotation phase, the final classes assigned to the 780 documents have been allocated as indicated in Fig1. To determine the level of agreement among the two annotators, the Cohen's Kappa score has been utilized. It measures the level of agreement between two annotators when classifying categorical items. It is particularly useful in scenarios where the agreement between the raters needs to be evaluated while considering the possibility of agreement occurring by chance alone.

It adjusts for the likelihood of agreement occurring by chance and provides a more accurate measure of agreement than simple percent agreement. The formula for Cohen's Kappa coefficient involves calculating the observed agreement between the raters and the expected agreement under the assumption of independence. It is defined as:

$$K = \frac{P_o - P_e}{1 - P_e}$$

where

- Po is the observed agreement between the raters.

- Pe is the expected agreement between the raters under the assumption of independence
- The value of k ranges from -1 to 1, where k=1 indicates total agreement between the raters, k=0 indicates agreement equivalent to that expected by chance and k=-1 indicates perfect disagreement between the raters.

Po and Pe are calculated using the following formulae:

$$P_{o=}\frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{n(n-1)}\sum_{j=1}^{m}(n^2{}_{ij}-n_{ij})\right) \quad (1)$$

$$P_e = \sum_{j=1}^{m} P_j^2 \quad (2)$$

Calculating the Cohen's Kappa for the dataset we obtained:

K = (0.801-0.197)/(1-0.197)= 0.7521

Table 1. Some works using different embedding strategy

Embedding used	Level	Model	Accuracy
TF-IDF	paragraph	SVM[13]	97.84
Word2vec	sentence	BLSTM-2DPooling [25]	88.3
Word2vec,GLOVE	document	SVM[14]	95.8
Doc2vec	document	SVM[14]	94.5
Word2vec	document	CNN-LSTM[24]	96(F1-value)

Which, when compared to the Cohen's interpretability score, is found to fall within the substantial agreement limit. This showed that the two annotators of our dataset agreed in the majority of the cases.

B. The Proposed Model: CNN-LSTM

The specification of the proposed model is thoroughly explained in this section. This model is composed of two DL models, namely the CNN and the LSTM. The intention of merging these two models in our research is to capitalize on the strong points of each one of them. The CNN layers capture the long and short term features which are then passed as inputs to LSTM for constructing a sequential correlation. The document vectors that are constructed using the Doc2vec model is passed as input to the proposed model. The model is trained and tested on the dataset prepared.

Thus, the following architecture is being proposed, which is composed of four parts, as illustrated in Fig. 2.

C. Pre-processing

Data quality has a role to play in the performance of deep learning models. It requires pre-processing steps such as tokenization, stop-word elimination, lowercase conversion and stemming [26].

Therefore, pre-processing of the dataset forms the first part of our proposed model. The raw input text may contain many characters such as English letters, punctuation symbols and numerals which are unhelpful for the classifier. In addition to this, the input text might contain certain frequently occurring words called stop words, which could lead the classifier astray while classifying. Hence, these unwanted words and characters need to be removed, so as to help the classifier. Some of the stop words in Assamese are- বাবে, দুই, তেওঁ, অতি, অথনি. Table 2 demonstrates a sample text before and after preprocessing.

D. Embedding

This step constitutes one of the most significant parts of the classification pipeline. The goal is to convert each preprocessed text into numeral vectors. This conversion is necessary as machine learning models can be applied to only numerical data. Embedding can be realized using a number of techniques including word2vec and doc2vec.

Table 2. Texts prior to and post cleaning

Prior to cleaning	Post Cleaning
নৱাগতপৰিচালকনস্পর্শৰেমালয়ালমছবিকনৰূপতস জাইছেনৱাগতপৰিচালকে শ্রীকুমাৰএটি Issue Dated :ফেব্রুৱাৰী 20, 2011 একাধিকসংকটসত্বেওমালয়ালমচলচ্চিত্রউদ্যোগৰৰূ পালীপর্তভুমুকিমাৰিছেআশাৰ ৰূপালীডাৱৰে।	নৱাগতপৰিচালকস্পর্শৰেমালয়ালমছবিকৰূপতসজাইছেন ৱাগতপৰিচালকে শ্রীকুমাৰফেব্রুৱাৰীএকাধিকসংকটসত্বেওমালয়ালমচলচ্চিত্র উদ্যোগৰৰূপালীপর্তভুমুকিমাৰিছেআশাৰৰূপালী ডাৱৰে

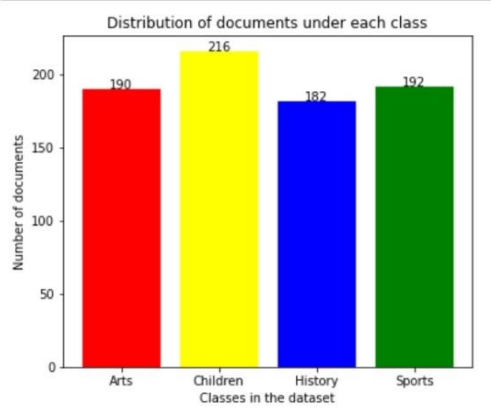


Fig 1. Class wise distribution of documents

To retain the semantic association between distinct documents, Doc2Vec, which is a variant of Word2Vec, attempts to select an appropriate continuous vector for a paragraph or even a document. As in word2vec, here too, each word is expressed by an n-dimensional continuous vector (n<<|V|), where V represents the vocabulary size in the dataset. Additionally, in the same space as word vectors, the document itself is likewise represented as a continuous vector.

Each document in Doc2Vec corresponds to a distinct vector embodied in a column in matrix D. According to the network structure of Doc2vec, the document embeddings for this work have been trained. The document vectors were trained using the PV-DBOW (Paragraph vector- Distributed Bag of Words) model. Vectors of three different dimensions were generated by this model, 100, 200 and 300. Fig. 3 displays the TSNE plot of the document vector generated.

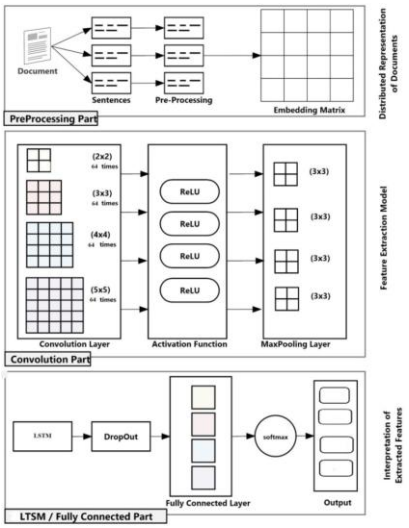


Fig. 2 The proposed CNN-LSTM model

E. Convolutional Layers

The convolution layer seeks to investigate the combinations of the various sentences and paragraphs in the document, employing filters with size a. Built to retrieve features, a CNN is meant to be merged into a bigger network.

In this layer, filters act as n-gram sensors/detectors and look for and identify a certain category of n-grams, giving them high scores. These identified n-grams with the highest scores run via the max pooling operations [27]. Four convolutional layers with each having 64 filters have been employed. The filters used were of sizes 2, 3, 4 and 5. After each filter, a max-pooling layer was applied in order to update and lessen the size of the vectors generated by the convolutional layer. The pooling size was set to 3X3, with a stride of 1 step. The outputs of each max pooling layer are concatenated to generate the input for the LSTM.

F. Activation Layer

The ReLU(Rectified Linear Unit) activation function has been used in each of the four convolution layers. It introduces non-linearity into the network by outputting the input directly if it is positive, and zero otherwise.

G. Regularization

Regularization involves organizing a neural network to prevent overfitting and improve deep learning performance. We use the dropout as a regulizer. Penalizing major weights helps optimize neural networks [28].

H. Optimization

DL techniques employ optimization to update the model weight as well as bias values over multiple iterations. Various optimization algorithms provide the most appropriate and optimal values for various parameters. In this work, we have used Adam (Adaptive Moment Estimation) [29].

I. LSTM

When it comes to extracting the important details from textual material, CNN is incredibly successful. But it fails to make a connection between the recent and earlier information. Being one type of RNN, LSTM can identify the long-term associations that exist in sentences of indefinite length. A layer of LSTM with 64 neurons was employed for this research. A dropout rate of 0.2 has been applied to control the parameters. This aids the model's ability to avoid the overfitting issue.

J. Dense Layer

This layer represents the final layer of this model. It acts on the output of the LSTM to categorize a given document into one of the four classes. A fully connected layer with four neurons and softmax as the activation function are used because multi-valued classification is taken into consideration for this work. Using a normalization technique, the softmax function determines the class to which the document belongs.

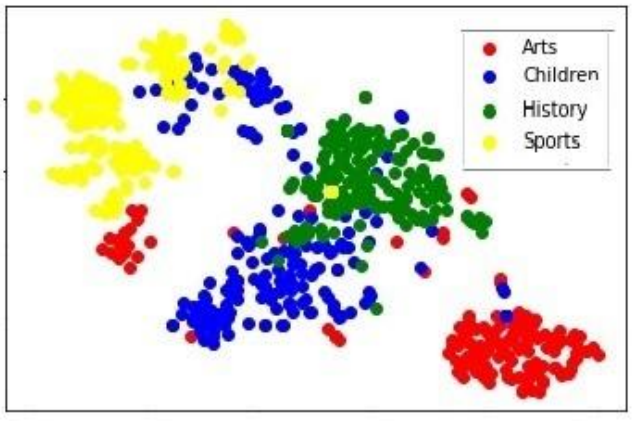


Fig. 3 Visualization of paragraph vectors of the Assamese documents using t-sne

V. A COMPARISON WITH CNN, LSTM, SVM AND LR

With a view to compare and contrast the results of this work, the CNN model and the LSTM model were built separately. Also, the dataset was tested using two traditional methods- the Logistic Regression (LR) and the Support Vector Machine (SVM).

A. CNN

The following parameters were used to configure this model:

- Three one-dimensional convolution layers, the first two of which are arranged one on top of the other, are used to build this model.
- 32 filters of varying sizes were applied to each of the convolution layers. The first one was used to look at 3 successive tokens at a time. The second filter was set up to look at 5 successive tokens, while the third filter uses 32 six-gram filters.
- Tanh was applied as the activation function in each of the three convolutional layers.
- Each of the second and third convolution layers was followed by a 1-D maxpooling, to retain just the most noticeable features from the feature map.
- A flatten layer was placed next to it, turning the 2X2 matrices into one-dimensional vectors.
- Two dense layers were used; one with 128 neurons, and the other that followed the first dense layer had 4 neurons that indicated the class count.
- As for the optimization function, Adam was used.
- Softmax was used as the activation function in the last layer. Connecting the given findings with the proper class was made possible using the softmax function.
- The loss function is essential for forecasting the probabilities of classes. We have used the cross-entropy loss function for this model.

B. LSTM

When evaluating long-text data, LSTMs are renowned for their capacity to maintain the order of chronology between the data. This model was built with 3 layers.

The following parameters were used to configure this model:

- The spatialdropout layer, with a dropout value of 0.2, was used as the initial layer.
- The LSTM layer was placed next to it, which consisted of 64 neurons. A recurrent dropout rate of 0.2 was applied with a view to updating the cells of the LSTM.
- A dense layer with 4 units was applied, indicating the number of classes used in the dataset.
- Sparsecrossentropy is utilized as the loss function and Adam for optimizing the output.

C. Experimental environment.

The proposed model CNN-LSTM was built using Python 3.11.3, with Keras and Tensorflow libraries. Scikit-learn 0.20.3 was used for SVM and logistic regression. Gensim library 0.3.1 was used to build the document embeddings. Two sets were created from the prepared dataset: a training set and a validation set.

80% of it is in the training set, while the remaining 20% is in the testing set. Therefore, a total of 624 documents formed the training set and 156 formed the testing set.

D. Results and Discussion

In order to assess the effectiveness of the proposed CNN-LSTM model, we tested the dataset on the other 4 machine learning models, namely, CNN, LSTM, SVM and LR, as well.

a) Results of Proposed CNN-LSTM model

The first experiment was conducted on the proposed model whose testing and training accuracy rate met its saturation point after 30 epochs. It exhibited an accuracy of 96.5%, 91.6% and 93.3% when applied to document vectors of size 300, 128 and 100 respectively. Fig. 4(a) displays the accuracy graph for the training and validation sets. This showcases the satisfactory performance of the model on both these sets. It also showed a significant reduction in validation and training loss, which can be seen in Fig. 4(b). Here, a loss rate of 0.15 can be observed, when applied to document vectors of size 300. For document vectors of size 128, the loss was 0.26% and for 100 dimensions, the loss was 0.3%. The highest performance rate of this model is recorded for vectors of dimension

Table 3. Classification report of CNN-LSTM

Dimension	Precision	Recall	F1-score
100	95	94	94
128	92	91	92
300	96	96	96

300, followed by that with 100 dimensional vectors. Figs 5(a), (b) and (c) represent the confusion matrix for the model on the 100, 128 and 300-dimensional vectors, respectively. Table 3 displays the classification report of this model. As can be observed, this model performed quite satisfactorily, recording an F1-value of 92% and above on all the three dimensions.

b) Results of CNN

This model recorded an accuracy rate of 93%, 94% and 92% when tested on the test dataset having 100,128 and 300 dimensions, respectively. The model was exposed to 30 epochs each for the three different dimensional vectors. The performance report of the model can be seen in Table 4.

c) Results of LSTM

The model registered the highest accuracy score of 92% for test data, having vector dimension 300, followed by vectors of 128 dimensions, which outputted 91% accuracy. The lowest accuracy rate was registered by 100-dimensional test vectors with an accuracy rate of 90%. The model was exposed to 35 epochs after which the accuracy got saturated Table 5 summarizes the model's performance.

d) Results of other machine learning models(SVM and LR)

The SVM recorded the highest accuracy rate of 89% for both the 100 and 300-dimensional test data, while LR outputted the highest accuracy of 90% for the 100- dimensional vectors.

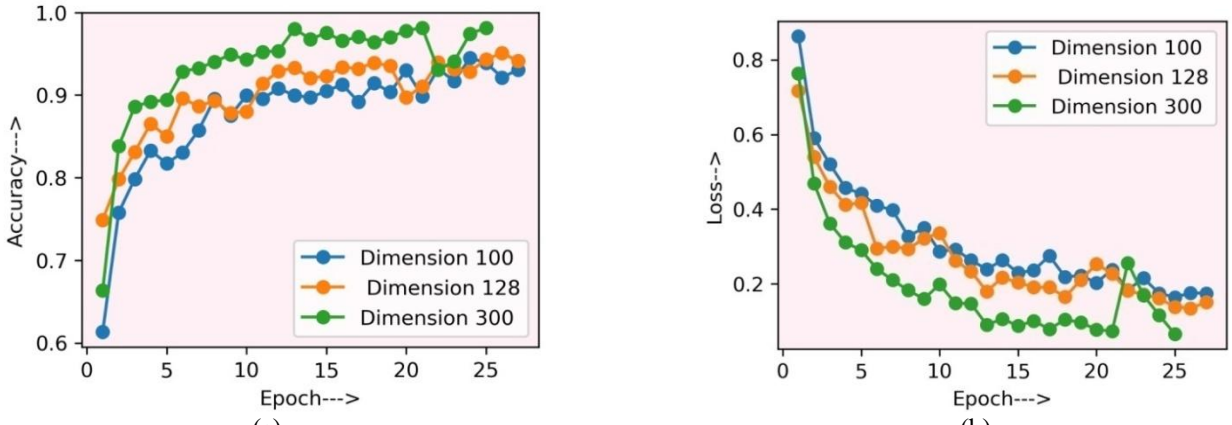
The F1-score of all the models has been presented as shown in Fig. 6.

Table 4. Classification report of CNN

Dimension	Precision	Recall	F1-score
100	94	93	93
128	94	94	94
300	92	92	91

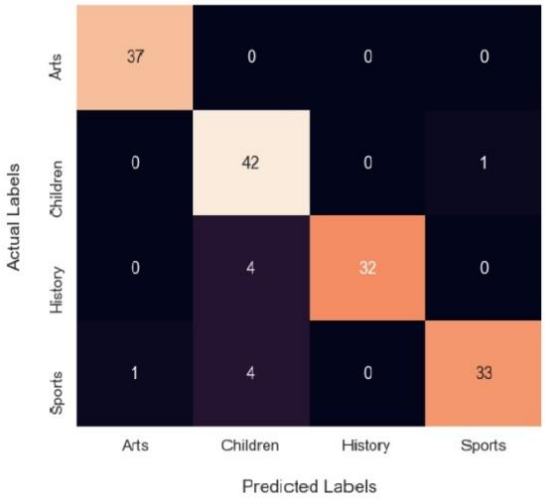
Table 5. Classification report of LSTM

Dimension	Precision	Recall	F1-score
100	95	89	90
128	92	91	91
300	96	91	91

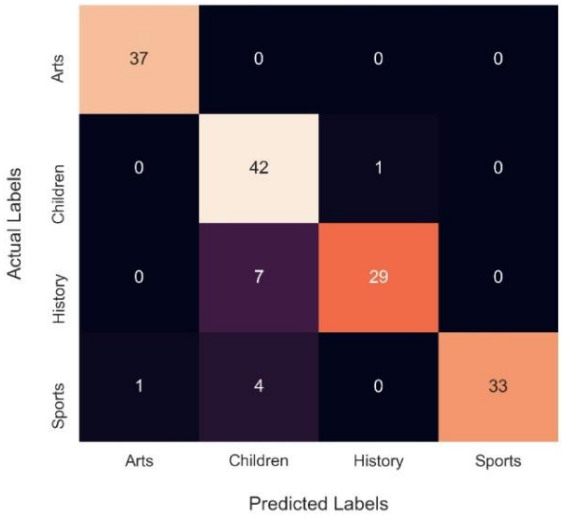


(a) (b)

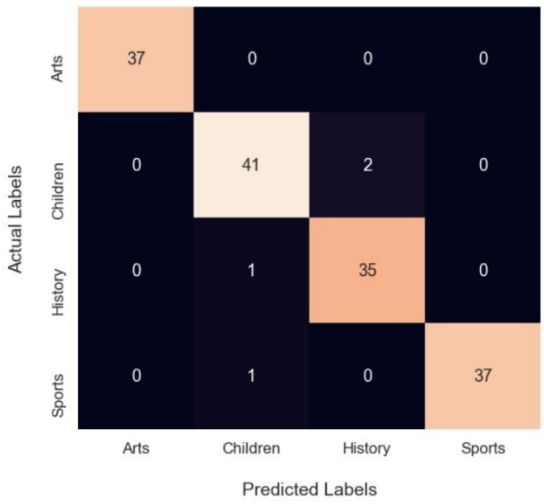
Figs. 4(a) and (b) Accuracy and Loss graph of CNN-LSTM model under the 3 dimensions



(a)



(b)



(c)

Fig. 5(a), (b) and (c) presents the Confusion matrices of the proposed model on 100,128 and 300 dimensional vectors respectively.

Table 6 provides a selection of some of the most significant publications in the area of text classification for various languages worldwide. It is evident from the table that our proposed work registered higher F1-score than the works of [19] and [24] and on Assamese texts.

e) Performance analysis of proposed model

Although the CNN-LSTM model proposed for Assamese text categorization exhibited an accuracy of 96.5%, it did not produce results without errors. The classification error of 3.5% can be attributed to the inherent ambiguous nature of the text, since every text has a contextual meaning. Even the length of the text has a role to play in giving an ambiguous concept of the document. A lengthy text on a particular category may also have references to other categories, which can mislead the classifier. For instance, a document under the children category may entail historical references. So, the classifier may classify it under the history category, instead of children.

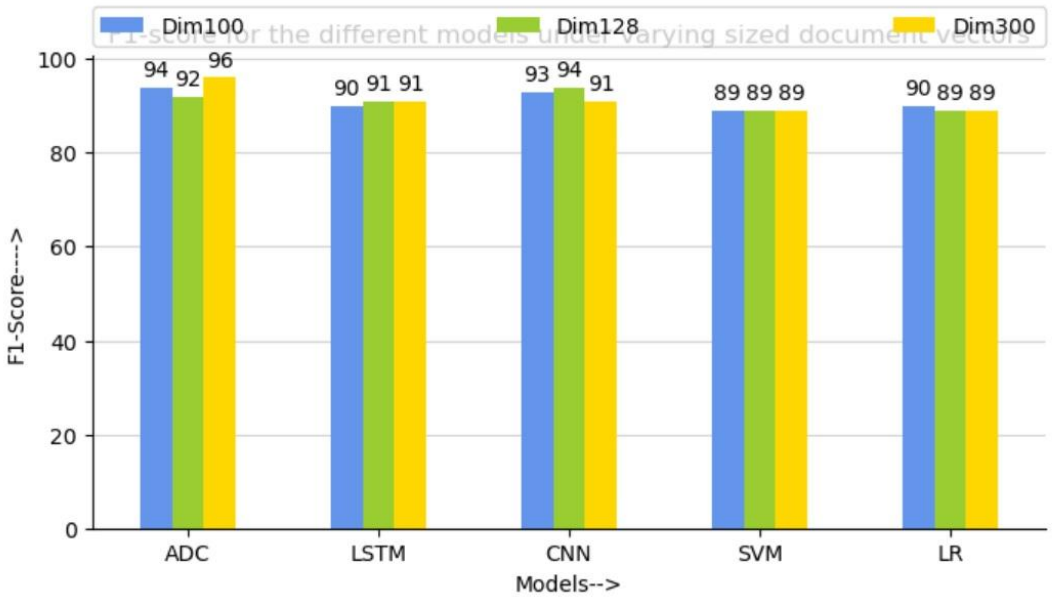


Fig. 6 comparative performance of the models in F1-value.

Table 6. Few prominent works on text classification

Reference	Model Used	Language	Dataset type	Accuracy/F1-score
[30]	Ensemble model	English	Twitter data	92.4
[31]	LR, Doc2vec (DBOW)	English	Security policy documents	96.8
[32]	CNN,Doc2vec	Indonesian	Twitter	65.08
[33]	LR,SVM(Doc2vec)	Indonesian	Twitter	87
[34]	CNN	Turkish	TTC-3600	94.17
[35]	LSTM	Marathi	Marathi dataset	91.1
[36]	BiLSTM,Doc2vec	Bengali	Facebook Post	77.85
[18]	Wordnet	Assamese	Wornet	94.4(F1)
[19]	Naïve Bayes	Assamese	Assamese Articles	95.4(F1)
[24]	CNN+LSTM	Assamese	Assamese Articles	95(F1)
Proposed Model	CNN + LSTM	Assamese	Assamese documents	96.5(F1)

VI. CONCLUSION

As the years pass by, the usage of the Internet is increasing at a rapid rate. This has contributed to the generation of a huge amount of digitized text even in low-resource languages such as Assamese. This data can be of great use if preserved properly and stored under the best fitted category depending on the content. This can make searching easier. This paper presents a novel approach to Assamese document classification and demonstrates the effectiveness of the proposed model, constructed by fitting CNN on top of the LSTM layers with document vectors. This paper also presents an annotated dataset of 780 documents classified under four categories namely Arts, Children, History and Sports, with a substantial agreement between the two annotators used to tag categories to the dataset. The model was tested on this dataset, which outputted an accuracy of 96.5%

and F1 value of 96%. Four other models were built using only CNN, LSTM, SVM and LR and tested on the dataset to compare the results produced by the CNN-LSTM model. However, the CNN-LSTM model was found to have outperformed the other models in terms of all the performance parameters, namely, F1-score, Precision, Accuracy and Recall.

We believe that this model could perform better on a larger dataset. As a future recommendation, the dataset size could be increased by adding more documents into it. Also, new categories can be introduced to the dataset and tried with different models.

REFERENCES

[1] X. Glorot, A. Bordes and Y. Bengio, Domain adaptation for large-scale sentiment classification: A deep learning approach, in Proceedings of the 28th international conference on machine learning (ICML-11), 2011, pp. 513–520.doi:https://doi.org /10.5555/3104482.3104547.

[2] I. B´ır´o, J. Szab´o and A. A. Bencz´ur, Latent dirichlet allocation in web spam filtering,in Proceedings of the 4th international workshop on Adversarial information retrieval on the web, 2008, pp. 29–32. Doi:https://doi.org 10.1145/1451983.1451991

[3] S. Dumais, J. Platt, D. Heckerman and M. Sahami, Inductive learning algorithms and representations for text categorization, in Proceedings of the seventh international conference on Information and knowledge management, 1998, pp. 148–155.https://doi.org/10.1177/14789299241265084

[4] T. Joachims, Text categorization with support vector machines: Learning with many relevant features, in European conference on machine learning, Springer1998, pp. 137–142. Doi: https://doi.org/10.1007/BFb0026683

[5] A. McCallum, K. Nigam et al., A comparison of event models for naive bayes text classification, in AAAI-98 workshop on learning for text categorization, 752(1), Madison,WI 1998, pp. 41–48.

[6] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882 (2014). https://doi.org/10.3115/v1/D14-1181

[7] Y. Yan, Y. Wang, W.-C. Gao, B.-W. Zhang, C. Yang and X.-C. Yin, Lstmˆ 2 2: Multi-label ranking for document classification, Neural Processing Letters 47 (2018) 117–138. http://dx.doi.org/10.1007/s11063-017-9636-0

[8] Y. Goldberg, Neural network methods for natural language processing (Springer Nature, 2022). https://doi.org/10.1007/978-3-031-02165-7.

[9] G. Salton, A. Wong and C.-S. Yang, A vector space model for automatic indexing,Communications of the ACM 18(11) (1975) 613–620. https://doi.org/10.1016/B978-0-44-329238-5.00017-2

[10] Y. Bengio, R. Ducharme and P. Vincent, A neural probabilistic language model, Advances in neural information processing systems 13 (2000). https://doi.org/10.1162/153244303322533223

[11] Q. Le and T. Mikolov, Distributed representations of sentences and documents, in International conference on machine learning, PMLR2014, pp. 1188–1196.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, Distributed representations of words and phrases and their compositionality, Advances in neural information processing systems 26 (2013). https://doi.org/10.5555/2999792.2999959.

[13] S. Puri and S. P. Singh, An efficient hindi text classification model using svm, in Computing and Network Sustainability: Proceedings of IRSCNS 2018 , Springer 2019,pp. 227–237. https://doi.org/10.1007/978-981-10-5780-9_11

[14] A. El Mahdaouy, E. Gaussier and S. O. El Alaoui, Arabic text classification based on word and document embeddings, in Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016 2 , Springer 2017, pp. 32–41. Doi: https://doi.org/10.1007/978-3-319-48308-5_4

[15] D. Kim, D. Seo, S. Cho and P. Kang, Multi-co-training for document classification using various document representations: Tf–idf, lda, and doc2vec, Information sciences 477 (2019) 15–29. https://doi.org/10.1016/j.ins.2018.10.006

[16] A. M. Aubaid and A. Mishra, A rule-based approach to embedding techniques for text document classification, Applied Sciences 10(11) (2020) p. 4009 . https://doi.org/10.3390/app10114009

[17] S. M. H. Dadgar, M. S. Araghi and M. M. Farahani, A novel text mining approach based on tf-idf and support vector machine for news classification, in 2016 IEEE International Conference on Engineering and Technology (ICETECH), IEEE2016, pp. 112–116. https://doi.org/10.1109/ICETECH.2016.7569223

[18] J. Sarmah, N. Saharia and K. Shikhar, A novel approach for document classification using assamese wordnet, in 6th International Global Wordnet Conference, 2012, pp. 324–329.

[19] M. Gogoi and S. K. Sarma, Document classification of assamese text using na¨ıve bayes approach, International Journal of Computer Trends and Technology 30 (2015) 182–186. http://dx.doi.org/10.14445/22312803/IJCTT-V30P132

[20] M. F. Ahmed Foysal, S. Tangim Pasha, S. Abujar and S. Akhter Hossain, Bengali news classification using long short-term memory, in Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Volume 2 , Springer2021, pp.329–338.

[21] L. Wan, G. Papageorgiou, M. Seddon and M. Bernardoni, Long-length legal document classification, arXiv preprint arXiv:1912.06905 (2019). https://doi.org/10.48550/arXiv.1912.06905

[22] M. Rhanoui, M. Mikram, S. Yousfi and S. Barzali, A cnn-bilstm model for document level sentiment analysis, Machine Learning and Knowledge Extraction 1(3) (2019) ,832–847. https://doi.org/10.3390/make1030048

[23] H. Zhou, Research of text classification based on tf-idf and cnn-lstm, in Journal of Physics: Conference Series, 2171(1), IOP Publishing 2022, p. 012-021. 10.1088/1742-6596/2171/1/012021

[24] C. Talukdar and S. K. Sarma, Hybrid model for efficient assamese text classification using cnn-lstm, International Journal of Computing and Digital Systems 14(1) (2023), pp-10183–10192. 10.12785/ijcds/140191

[25] B. Jang, M. Kim, G. Harerimana, S.-u. Kang and J. W. Kim, Bi-lstm model to increase accuracy in text classification: Combining word2vec cnn and attention mechanism, Applied Sciences 10(17) (2020) p. 5841. https://doi.org/10.3390/app10175841

[26] A. K. Uysal and S. Gunal, The impact of preprocessing on text classification, Information processing & management 50(1) (2014) 104–112. . https://doi.org/ 10.1016/j.ipm.2013.08.006

[27] A. Jacovi, O. S. Shalom and Y. Goldberg, Understanding convolutional neural networks for text classification, arXiv preprint arXiv:1809.08037 (2018). https://doi.org/10.48550/arXiv.1809.08037

[28] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv preprint arXiv:1207.0580 (2012). https://doi.org/10.48550/arXiv.1207.0580

[29] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014). https://doi.org/10.48550/arXiv.1412.6980

[30] M. Lansley, F. Mouton, S. Kapetanakis and N. Polatidis, Seader++: social engineering attack detection in online environments using machine learning, Journal of Information and Telecommunication 4(3) (2020) 346–362. https://doi.org/10.1080/24751839.2020.1747001,

[31] M. S El .Rahmany, E. Hussein Mohamed and M. H Haggag, Semantic detection of targeted attacks using doc2vec embedding, Journal of Communications Software and Systems 17(4) (2021) 334–341. https://doi.org/10.24138/jcomss-2021-0113

[32] S. T. Laxmi, R. Rismala and H. Nurrahmi, Cyberbullying detection on indonesian twitter using doc2vec and convolutional neural network, in 2021 9th International Conference on Information and Communication Technology (ICoICT), 2021, pp. 82–86. http://dx.doi.org/10.1109/ICoICT52021.2021.9527420.

[33] T. H. J. Hidayat, Y. Ruldeviyani, A. R. Aditama, G. R. Madya, A. W. Nugraha and M. W. Adisaputra, Sentiment analysis of twitter data related to rinca island development using doc2vec and svm and logistic regression as classifier, Procedia Computer Science 197 (2022) 660–667. https://doi.org/10.1016/j.procs.2021.12.187

[34] H. B. Dogru, S. Tilki, A. Jamil and A. A. Hameed, Deep learning-based classification of news texts using doc2vec model, in 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA), IEEE2021, pp. 91–96. https://doi.org/10.1109/CAIDA51941.2021.9425290

[35] F. Eranpurwala, P. Ramane and B. K. Bolla, Comparative study of marathi text classification using monolingual and multilingual embeddings, in International Conference on Advanced Network Technologies and Intelligent Computing, Springer2021, pp. 441–452. http://dx.doi.org/10.1007/978-3-030-96040-7_35

[36] M. T. Hoque, A. Islam, E. Ahmed, K. A. Mamun and M. N. Huda, Analyzing performance of different machine learning approaches with doc2vec for classifying sentiment of bengali natural language, in 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), IEEE2019, pp. 1–5. https://doi.org/10.1109/ECACE.2019.8679272