Text Document Classification using Convolutional Neural Networks

Vishnu Panickar#1, Sujit Pradhan#2, Priyanka Kashyap#3, Ashish Kawale#4, Nihar M. Ranjan#5
Department of Computer Engineering, JSPM's Narhe Technical Campus, Pune.

1vishnu.panickar234@gmail.com
2sujitpradhan78@gmail.com
3pkashyap0409@gmail.com
4ashishvk99@gmail.com
5nihar.pune@gmail.com

Abstract: Documents are one of the most common methods for maintaining data and records. Everyday a lot of documents/files are generated with lot of data for future research purposes or for business analytics. These files/documents must should be stored effectively so that it can be retrieved whenever needed. Organizing large documents can be a tedious task as the internal content of the files are not known. Manually organizing each and every file is not practically possible as it may take hours to categorize a file based on its contents and also the accuracy of classification cannot be guaranteed. In the fields like Library Science a huge amount of files are required to be maintained, which can be helpful in future for business decisions or for research purpose. To make this task easier Text Document Classifier can be used. Text Document Classifier can classify a given document based on the contents inside the document and label the document from the pre-defined classes. Unlike traditional classification Techniques in Machine Learning like Support Vector Machine, term frequency-identification and Naïve Bayes Classifier, Neural Networks has better analytical results. Traditional Classification Methods has limitations in terms of effective feature extraction and the dimensionality problem, these limitations can be solved by Convolutional Neural Networks.

Keywords: Convolutional Neural Network, Data Mining, Machine Learning, Text Classification, Word Embeddings.

1. INTRODUCTION

In this era text data is being generated from various sources for business and analytics. There are a lot of social media platforms and each of this platform generates a lot of data. These data are used by many companies and organizations to improve their service quality as well as increase their business. Now the data collected from these sources are very huge in volume and must be organized properly, so that when needed the data can be retrieved efficiently. The data must be arranged according to a category they belong to. But determining the categories of the documents or files can be very difficult and time consuming task. We cannot manually read each and every file and then assign it a category, as there are very chances of human error and also it may be very time consuming. So an effective mechanism is needed to classify these documents.

Text Document Classifier is an automated classification application developed to classify ocuments to a pre-defined category based on their internal contents. Documents are nothing but huge blocks of words, sentences or paragraphs. Keyword matching is one of the technique used in commercial document classification to classify, but the problem in this technique is building a keyword list which is suitable for every category which is not an easy task.

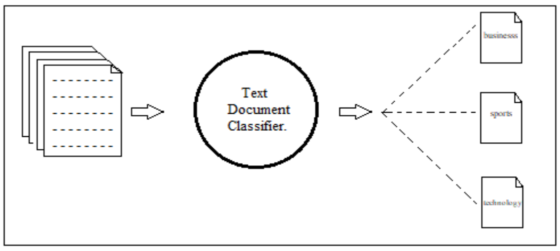

Fig 1.1 Document Classifier

And moreover as keyword matching only consider the different independent keywords without taking the context information into consideration These texts are represented as features of a particular document, and the feature representation is one of the key issues in text classification. Feature representation of traditional machine learning approaches use bag of words [3], [15] or n-gram [16] techniques to generate feature vector as text representation to train classifiers. Natural Language Processing and Machine Learning algorithms are used to develop the application. As we know Machine Learning has a lot of algorithms which can deal with classification very well, but we will use Neural Networks for classifying the documents. Neural Networks are lot faster and more efficient as compared with traditional machine learning techniques.

2. RELATED WORK

Text classification has been one of the oldest research topics in the field of machine learning. Various classification methods were used for classification of text which were quite successful. Sang-Bum Kim [1] in his work used naïve bayes classifier [19], [1] for classification of text. In his research he observed that naïve bayes is not that effective during parameter estimation process, which causes poor results in text classification domain. Naïve Bayes has been one of the popular machine learning method for various years. It is very simple to use hence this framework is widely used in various task, but the learning is based on unrealistic assumptions. In their naïve bayes classifiers, a document is considered as a binary feature vector representing [14], [16] whether each word is present or absent. This naïve bayes is called multivariate Bernoulli naïve bayes. Two problems were encountered during this process. The first one is its rough parameter estimation. In the multinomial model, the word probabilities for a class are estimated by calculating the likelihood in the training documents. This means that parameter estimation in this model is affected more by long documents rather than short documents. The second problem is in handling rare category documents that contain only a few training documents. When a category has only a few training documents, a few informative models that are useful to determine the category of a document. Usually this method cannot give us accurate results as the data may contain relatively large number of noisy terms and their probability estimates can be unrealistic. In this paper, two empirical heuristics: pre-document text normalization and feature weighting method were used. The pre-document text normalization with prior distribution over a positive or negative document set makes the proposed model remarkably different from the traditional multinomial model. This model was able to reduce the problems of rough parameter estimation and handling rare categories that occurred in traditional classifiers [18], but it was little costly in terms of time and space.

Dino Isa and Lam Hong Lee [3] in their work introduced support vector machine [15] along with naïve bayes classifier. This work implemented an enhanced hybrid collection through the utilization of the naïve bayes approach and the support vector machine. In this implementation, the bayes formula was used to vectorize a document according to a probability distribution reflecting the probable categories that the document may belong to. This method has been analytically proven better as compared to other classifiers. But, it was not easy to vectorize the text data into numeric form by using support vector machine, and methods like TF-IDF had problems of dimensionality [1], [7], [19]. The bayes formula gives a range of probabilities to which the document can be assigned according to a predetermined set of topics Their system was divided as follows: naïve bayes was used to pre-process the data, i.e., vectorize the data and the support vector was used as a classifier. In this work, the dimensions were reduced from thousands to typically less than 20 through the use of bayes formula and the then this reduced probability distribution was fed to the support vector machine for training and classification purposes. This hybrid system utilized the simplicity of Bayesian formula as a vectorizer and the capability of SVM to generalize efficiently as a classifier. Naïve bayes formula vectorized the documents by using the probability distribution where the dimension of features is based on the number of categories available. The hybrid model had a better performance ratio when compared with the performance of traditional naïve bayes classifier. This model does not perform well where there is high percentage of similarity between the keywords [7]. In such cases the naïve bayes classifier shown greater accuracy as it uses the highest probability category to identify the correct class of the document. The naïve bayes and SVM had a good performance metrics in terms of training time and testing time when compared with other hybrid models and the hybrid model has only one second addition in terms of training and testing time when compared with traditional naïve bayes classification approach.

Jung-Yi Jiang, Liou and Lee [6] in their work proposed a fuzzy similarity based self-constructing algorithm [8] which was used for feature clustering. Feature clustering was able to decrease the dimensionality of feature vectors. The two major approaches, feature selection and feature extraction were used for reducing features and forming clusters of words based on similarity test. By feature selection, a new feature set is obtained, which is the subset of the original word set. This obtained subset is then used as the input set for classification tasks. Feature clustering, which is an efficient approach is then used to group words that are similar to each other, into the same cluster and is characterized by a membership function with statistical mean and deviation. The user need not specify the features in advance. When all the words are fed, the clusters are formed automatically. The clusters formed in this approach are of incremental and self-constructing nature, which is an important factor in the calculation of similarity. No clusters exist at the beginning, and then the clusters are formed according to necessity. The training data in this model uses support vector machine classifier as it is better than other classifying models. In order to make the method more flexible and robust, support vector machine finds the maximum hyperplane in feature space. This algorithm finds the points closest to the boundary values which are known as support vectors [15], [7]. The approach tries to make a decision boundary in such a way that the separation between various classes is widely possible. The fuzzy model gives better results in terms of speed and it can obtain better extracted features when compared with other classification approaches.

Ali arshad, Saman Riaz and Licheng JIAO [8] in their work propose a novel approach using DEFM-MC by utilizing multiple intra clusters to extract the information about the new features which can control the redundancy for the multiple class which are imbalance for classification, here the classification is associated with maximum similarity of the features between the different multiple intra clusters. Further they have improved the classification performance of their model for classification. As its believed that more features are redundant, irrelevant causing more risk by making the system complex and furthermore growths the time and cost. Hence the features can be reduced into two ways of features extraction and then features selection. However, very few works are done for combine features reduction technique to enhance the performance of classification on the multi-class imbalanced problem and easy to implement with efficient and better results [16]. Their main motivation to utilize the combine features reduction techniques is a way to handle all the problems of imbalanced data and also to eliminate all the irrelevant and redundant features and noisy data for the classification of the data by using the proposed DFMC-MC based features extraction technique and feature selection technique such as random under sampling (RUS) and random over sampling (ROS).

3. IMPLEMENTATION DETAILS

Documents are nothing but collection of sentences and paragraphs. To Process these documents, it must be first converted to a suitable format for processing. Text Documents are the easiest format to read and process, so first step is to convert the document into text format and then extract the text into list in python. The classification model was built using Convolutional Neural Network, using keras. Keras is a library in neural networks and provides APIs for Neural Network, which is included in tensorflow which is an open source library for machine learning projects.

Convolutional Neural Networks, is a deep learning algorithm widely used for image processing. The idea of classifying text using convolutional neural network was presented by Yoon Kim [6] in the paper "Convolutional Neural Networks for Sentence Classification" The idea was to process the documents in the similar way as image were processed which will be explained later in Embedding Layer.

Dataset: BBC News Dataset is used as the training data for building the model, it can be downloaded from the following link: http://mlg.ucd.ie/dataset.

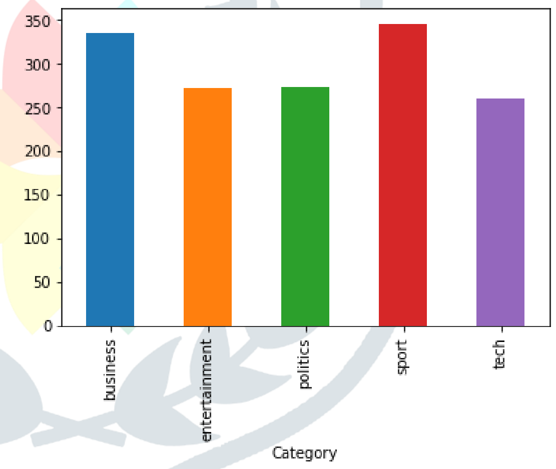

Fig 3.1: Dataset description graph

The dataset consists of 2225 files of text information of the following classes: 'business', 'entertainment', 'politics', 'sports' and 'technology'. The detail of number of files per category is shown in the Fig 3.1.

Data Preparation and pre-processing: The pre-processing done on the data set are remove punctuations, convert to lowercase, remove numbers and remove single characters. Converting texts into lowercase helps in parsing the text better it also helps in reducing the size of the vocabulary. Removing punctuations helps to increase the coverage of the embeddings on our vocab. Removing numbers and removing single characters helps in improving the embedding ratio by 2-3%.

Input Layer: The pre-processed data will be passed on to the input layer. The data consists of 2225 features and max sequence length of 1000. So the input matrix is of the shape (2225,1000). The tokenizer classes provided by the keras library converts text to sequences and provides access to the mapping of the words to integers.

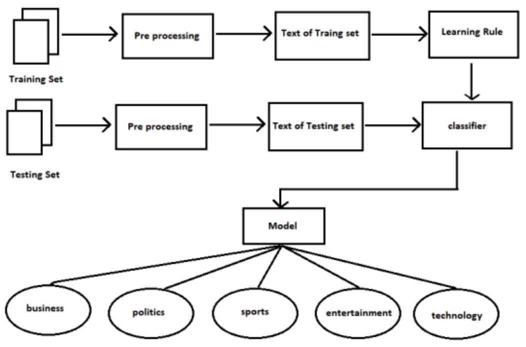

Fig 3.2 Model Flow

The input is then passed to the embedding layer. Embeddings are representation of words into a type such that the words with similar meaning have similar representation. In word embeddings the individual words are represented in the form of real-valued vectors, in the dimension of few hundreds. This contracts to the thousands or millions of dimensions for the representation of sparse words like one hot encoding. The input to Convolutional Neural Network contains categorical features that takes one of k distinct symbols, each word is associated to each possible feature value with a dimension d. Embeddings helps to capture the implicit relations between words, by finding the number of time the word is used in the training documents.

In our model we are using glove (Global Vector for Word Representation). It is a method for efficiently learning word vectors. Glove is an unsupervised learning algorithm for generating the vector representation for the words. The training if performed on the word-word co-occurrence statistics from the corpus, and the representation depicts a linear substructure of the words. Let us represent two sentences into the word vector representation, the max length of the sentence be 5. ’hope to see you soon’ and ‘nice meeting you’ be the two sentences. After embedding these sentences, we will get the results as [1,2,3,4,5] and [6,7,4,0,0] 0 is padded to make the sentence of same length. So our word index will be 0: [1.2, 3.1, 2.5], 1: [1.2, 3.1, 2.5], 2: [1.2, 3.1, 2.5], 3: [1.2, 3.1, 2.5], 4: [1.2, 3.1, 2.5], 5: [1.2, 3.1, 2.5], 6: [1.2, 3.1, 2.5], 7: [1.2, 3.1, 2.5]. So the sentences after converting to word index form will be represented as ‘[1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5]’ and ‘[1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5], [1.2, 3.1, 2.5],’ The word embeddings are optimized by keras during the training phase. In our model we have used glove pre-trained word embedding with the dimension of 100. It has 400 million pre-trained word vectors trained on data from Wikipedia.

The word vector is passed in the form of matrix X and defines a soft constraint for each pair

$$w_i^T w_j + b_i + b_j = \log(X_{ij})$$

i represents how the words appear in context of word j. wi is the vector for main word and wj is the vector for the context and bi, bj are the bias. The cost function of glove embedding J is represented as:

$$J = \sum_{i=1}^{V} . \sum_{j=1}^{V} f(X_{ij}) \left(w_i^T w_j + b_i + b_j - logX_{ij}\right)^2$$

The word vector $l, l \in R^d$ and d is the dimension of the word vector. The entire document is represented as $D \in R^{nd}$ in the form of matrix. n is the number of words in it. A sentence of max length is padded where ever necessary.

$$l_n = l_1 \oplus l_2 \oplus l_3 \oplus \ldots\ldots \oplus l_n$$

l_n refers to the concatenation of words. The convolution involves a field W which is applied to the window tom form a feature C. The convolution is defined as:

$$W \in R^{hd}$$

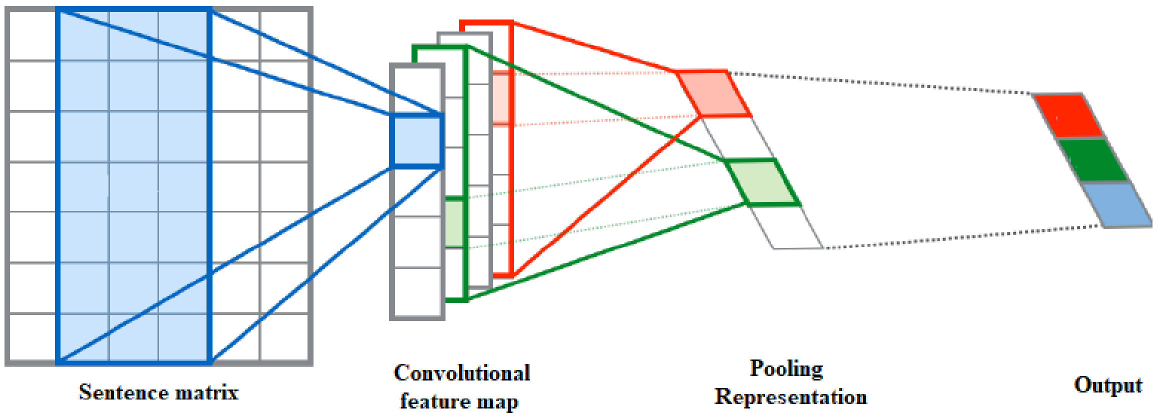

Fig 3.3 Representation of CNN Flow.

where h is the number of words the convolution will span, i.e. the size of the stride. The equation can be denoted as-

$$W * D_{j:j+h-1} = \sum_{i=j}^{j+h-1} . \sum_{k=0}^{d-1} Wi, kDi, k$$

Suppose y is a feature than, y will be represented as follows:

$$y = f(W.X_{i:i+h-1} + b)$$

b is the bias term and f is a non-linear function. The filter is applied to the window in every possible way to produce a feature map. To obtain the feature map, we add the bias term and apply activation function. After convoluting the entire document, we will have a convoluted feature map C, such that

$$C(W) = [C_1, C_2, C_3, \ldots . C_{n+h+1}]$$

The feature map is the forwarded to the pooling layer to down sample the feature map. We are using max-pooling layer for down sampling, it includes extracting some representation from the feature map. Max pooling chooses the maximum of the value from the input feature map. This is done to reduce the number of vectors in the feature map. The pooling layer gives a single feature that corresponds to the feature map w, and is represented by-

$$\bar{c}w = [\bar{C}w_1 + \bar{C}w_2 + \cdots + \bar{C}w_k]^T$$

This feature is then passed to the softmax activation layer that converts the output into the probabilities that result into 1. The output Y can be denoted as:

$$Y_c = \frac{e^{zc}}{\varepsilon_{d=1}^{c} e^{zd}}$$

And the loss function can be optimized by categorical cross-entropy by using the equation

$$\text{Loss} = -\sum_{i=1}^{0} . \sum_{j=1}^{c} \bar{y}jlogy_i^j + \lambda\|\theta\|^2$$

The values given to the fully connected neural network layer, CNN goes through the back propagation process to determine the most accurate weights. Each neuron receives a weight that prioritizes the most appropriate label after passing through the activation function the output is then converted into probabilities which results in the summation of 1. The model with the highest accuracy is saved in the hierarchical data format and the file is updated if the accuracy of the model increases from the previous value.

4. RESULTS

The trained model is able to classify text document into the five pre-defined classes (‘business’, ‘entertainment’, ‘politics’, ‘sports’ & ‘technology’). The trained model has an accuracy of 97.75% and the model is able to find the co-relation between the texts. Based on the context of the text it can classify the document into the pre-defined classes. e.g.: “the ball is on the table” would be classified into sports category and the sentence “the entries was deleted from the table” this sentence will be classified into tech as the model is able to classify the file based on the context of words. Fig 4.1 shows the graphical representation of the model accuracy for training as well as testing data.

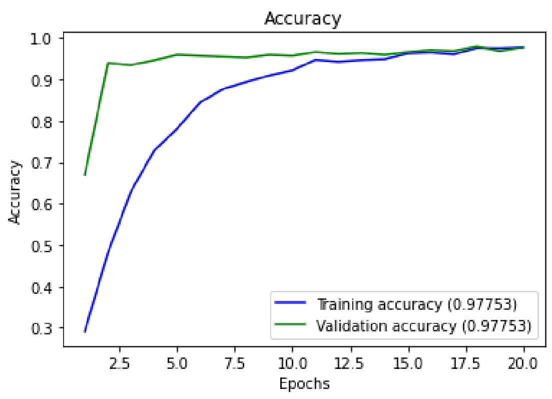

Fig 4.1 Accuracy graphical representation

The model was trained with 1780 files and was validated against 445 files. Fig 4.2 shows the graph for loss function of the model. The performance of a model is described using the metrics accuracy, precision, recall and f1-score.

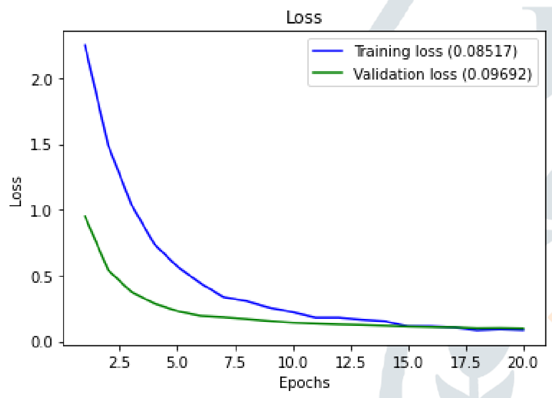

Fig 4.2 Loss graphical representation

1.**Accuracy**: It is calculated using the ratio of total no. true positive and true negative to the ratio of the total observations.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

The accuracy of the trained model is 97.75 %.

2. **Precision**: It is the number of true class prediction actually belong to the true class.

$$\text{Precision} = \frac{TP}{TP+FP}$$

The precision of the model is 96.8.

3.**Recall**: Recall is the ratio between correctly predicted observation to the total observation.

$$\text{Recall} = \frac{TP}{TP+FN}$$

The recall of the model is 98.8.

4.**f1-Score**: It is the weighted average of recall and precision.

$$\text{f1-score} = \frac{2TP}{(2TP+FP+FN)}$$

The f1-score of the model is 98.6.

5. CONCLUSION & FUTURE WORK

Our model is able to classify documents with an accuracy of 97.75% on BBC News Dataset. The model is able to classify files on the basis of five classes 'business', 'entertainment', 'politics', 'sports' and 'technology'. Using word embedding along with Convolutional Neural Network helped in maintaining the correlations between the words.

Using CNN for classifying text always has a scope of improving the accuracy as the amount of data increases and embeddings helps to maintain the correlation between the text. The model could be improved using different pre-trained word-embedding such as google-news embedding or the embedding can be created manually according to the vocab of our training data. As a future work extra classes can be added to increase the area of classification beyond five classes. Classifying the text information using images could be done by extracting the contents from images. The model can be trained to classify files into more specialised categories such as the file belongs to cricket or football in sports category.

REFERENCES

[1] Sang-Bum Kim, "Some Effective Techniques for Naive Bayes Text Classification" PhD, Department of Computer Science, Korea University, 2006.

[2] Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) Convolutional Neural Networks for Sentence Classification by Yoon Kim.

[3] Dino Isa, Lam Hong Lee, "Text Document Pre-processing with the Bayes Formula for Classification Using the Support Vector Machine", IEEE TRANSACTIONS on data engineering 2008

[4] Nihar Ranjan, Rajesh Prasad, "Automatic Text Classification using BP Lion- Neural Network and Semantic Word Processing", Imaging Science Journal Print ISSN: 1368-2199, Online ISSN: 1743-131X

[5] D.D. Lewis, “Representation and Learning in Information Retrieval,” PhD dissertation, Dept. of Computer Science, Univ. Of Massachusetts, Amherst, 1992.

[6] Jung-Yi Jiang, Ren-Jia Liou, “A Fuzzy Self-Constructing Feature Clustering Algorithm for Text Classification”, IEEE Transactions on Data Science 2011.

[7] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, ``Improving word representations via global context and multiple word prototypes," in Proc. ACL, Jeju-do, South Korea, Jul. 2012, pp. 873_882.

[8] Hisham Al-Mubaid and Ali Arshad, “A New Text Categorization Technique Using Distributional Clustering and Learning Logic”, 2006.

[9] Nihar Ranjan, Rajesh Prasad,” Author Identification in Text Mining for Used in Forensic”, International Journal of Research in Advent Technology E-ISSN: 2321-9637

[10] A. Rios and R. Kavuluru, ``Convolutional neural networks for biomedical text classification: Application in indexing biomedical articles," in Proc. ACM-BCB, Atlanta, GA, USA, Sep. 2015, pp. 258_267

[11] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, ``A convolutional neural network for modelling sentences," arXiv:1404.2188, 2014.

[12] S. Block, D. Medin, and D. Osherson, P.A. Flach, E. Gyftodimos, and N. Lachiche, “Probabilistic Reasoning with Terms,” technical report, Univ. of Bristol, Louis Pasteur Univ., 2002.

[13] Y. Xia, W. Liu, and L. Guthrie, “Email Categorization with Tournament Methods,” Proc. Int’l Conf. Application of Natural Language (NLDB), 2005.

[14] X. Su, “A Text Categorization Perspective for Ontology Mapping,” technical report, Dept. of Computer and Information Science, Norwegian Univ. of Science and Technology, 2002.

[15] J.R. Quinlan, H. Kim, P. Howland, and H. Park, “Dimension Reduction in Text Classification with Support Vector Machines,” J. Machine Learning Research, vol. 6, pp. 37-53, 2005.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ``Distributed representations of words and phrases and their compositionality," in Proc. NIPS, Carson City, NV, USA, Dec. 2013

[17] Nihar Ranjan, Rajesh Prasad,” LFNN: Lion fuzzy neural network-based evolutionary model for text classification using context and sense based features”, Applied Soft Computing ISSN: 1568-4946

[18] Ning Zhong, Yuefeng Li, and Sheng-Tang Wu, “Effective Pattern Discovery for Text Mining” IEEE TRANSACTIONS 2012.

[19] A. McCallum and K. Nigam, “A Comparison of Event Models for Naıve Bayes Text Classification,” J. Machine Learning Research 3, pp. 1265-1287, 2003.

Nihar M. Ranjan obtained B.E. in computer engineering from North Maharashtra University, Jalgaon, Maharashtra, M.E. in computer science and engineering from V.T.U, Belgaur, Karnataka and Ph. D in Computer Science from SPPU, Pune, Maharashtra in 2000, 2008 and 2019 respectively.
He is currently working as the head of computer department in JSPM Narhe Technical Campus, Pune. His research interests are data mining, text mining and text analytics. He has more than 10 publications in various international journals and conferences.

Vishnu Prathapan Panickar is currently in the final year of his bachelor’s degree in computer science. He is pursuing this degree from JSPM Narhe Technical Campus, Savitribai Phule Pune University, India.
His area of interest is in Data Mining, Machine Learning and Artificial Intelligence.

Sujit Pradhan is currently in the final year of his bachelor’s degree in computer science. He is pursuing this degree from JSPM Narhe Technical Campus, Savitribai Phule Pune University, India. His current interest is in data mining and machine learning.

Priyanka Kashyap is currently in the final year of her bachelor's degree in computer science. She is pursuing this degree from JSPM Narhe Technical Campus, Savitribai Phule Pune University, India. Her area of interest is in data mining and machine learning.

Ashish Vitthal Kawale is currently in the final year of his bachelor's degree in computer science. He is pursuing this degree from JSPM Narhe Technical Campus, Savitribai Phule Pune University, India. His area of interest is in data mining and machine learning.