I.J. Intelligent Systems and Applications, **2019, 8, 56-67**
Published Online August 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2019.08.06

Comparative Study of Convolutional Neural Network with Word Embedding Technique for Text Classification

Amol C. Adamuthe
Department of Information Technology, Rajarambapu Institute of Technology, Rajaramnagar, MS, India
E-mail:amol.admuthe@gmail.com

Sneha Jagtap
Department of Computer Engineering, Rajarambapu Institute of Technology, Rajaramnagar, MS, India
E-mail: jagtap.sneha35@gmail.com

Received: 29 January 2019; Accepted: 20 March 2019; Published: 08 August 2019

Abstract—This paper presents an investigation of the convolutional neural network (CNN) with Word2Vec word embedding technique for text classification. Performance of CNN is tested on seven benchmark datasets with a different number of classes, training and testing samples. Test classification results obtained from proposed CNN are compared with results of CNN models and other classifiers reported in the literature. Investigation shows that CNN models are better suitable for text classification than other techniques. The main objective of the paper is to identify best-fitted parameter values batch size, epochs, activation function, dropout rates and feature maps values. Results of proposed CNN are better than many other classification techniques reported in the literature for Yelp Review Polarity dataset and Amazon Review Polarity dataset. For all the seven datasets, accuracy obtained by proposed CNN is close to the best-known results from the literature.

Index Terms—Convolutional Neural Network, Text Classification, Text mining, Word2Vec.

I. Introduction

In recent year, the volume of text data is increasing rapidly due to digitalization [1]. Application field like life science, social media, business intelligence, healthcare etc needs text mining [2]. The purpose of text mining is multi-fold namely feature selection [1,3], feature extraction [4], web opinion mining [3,5] etc. Text classification applications are present in various domains namely social media [6,7], healthcare industries [8], business intelligence, image processing etc. In text classification system, words related to one specific concept are organized in a document. The main purpose of test classification is to assign a label to document based on the present words. Text classification is designed to assign predefined classes to text documents [9].

Increase in large volume of unstructured text created the need for efficient and effective algorithms on text mining such as classification and clustering [9]. Text classification is a challenging open problem in natural language processing [10]. The broad outline of classification methods is presented in [11]. Yang et al. reported the significance of five techniques for text categorization [12]. Techniques namely support vector machine [13], KNN [14], association-based classification [5], term gram model [15], Bayesian classification [16,17], and neural network [18] used for text classification. Each classification technique has some limitations. Support vector machine has limitations for scale-up with an increase in the number of documents. KNN requires more computation time and has challenges for high dimensions and high samples. Association based classification is subjected to the overfitting problem. The graph representation is a computational task for text classification using term graph model. Naive-Bayes works well only for low dimensions [18].

Better ability to learn complex feature representations is the main advantage of CNN over traditional machine learning approaches [8]. CNN approaches shown good results for many research domains. CNN is effective for applications of natural language processing (NLP) and achieved superior results in sentence modeling, search query retrieval, semantic parsing and other traditional NLP tasks [19]. CNN has given better results for many problems from the medical domain [8], social networks [6], image classification and image analysis [7]. CNN works better for image processing applications than many other neural network structures. CNN found superior for document classification [19]. CNN gives outstanding performance for computer vision. The important objective of CNN is its application to feature extractor [20]. CNN used to extract character to sentence level features for the performance of sentiment analysis [21].

Text classification is a supervised learning task in which labels are assigned to text documents [12]. CNN has the ability to learn complex feature representations which differs it from traditional machine learning approaches [13]. In literature, CNN effectively solved may application in different domains. General applications of CNN to text classification is a well-studied topic in the literature.

The paper presents convolutional neural network with word embedding technique for text classification. The main objective is divided into three sub-objectives,

1. To confirm the performance of CNN for text classification.
2. Researchers have suggested for investigation of new methods for improvement in accuracy of text classification. To perform a comparison of CNN with other classifiers for text classification. This identifies the need for improvement in text classifier.
3. To tune the hyperparameters of CNN to improve the results of text classification

To test the performance of CNN models, seven benchmark datasets with varying classes from 2 to 14 are used. Performance of the proposed CNN model is compared with CNN and other text classifiers reported in the literature.

In literature, different text classifiers are investigated for text classification such as Naïve Bayes, support vector machine, logistic regression, stochastic gradient descent, long short-term memory and hybrid models of these. Results indicate that CNN models give the best accuracy for all datasets except dataset 1. The performance of CNN is good to text classification.

Paper presents results of convolutional neural network with variations of hyperparameters namely activation function, batch size, epochs, dropout rates, feature map values for text classification. Results show that the accuracy is increased with increase in epochs and dropout rate to a certain number. Subsequently, it shows a reduction in accuracy. Performance of ReLU activation function is good for all the datasets. CNN accuracy increases with a number of feature maps.

The rest of this paper is organized as follows. Section II is about the discussion of related work on text classification using different machine learning algorithms including convolutional neural network. Section III demonstrated the proposed methodology. Experimental details, datasets, results and discussion are presented in section IV. Section V is present about conclusions.

II. Related Work

In literature, many techniques have experimented for text classification. Researchers used different datasets for performance testing of text classification algorithms.

Papers [6-8, 15, 19, 21] shows the results of different techniques for text classification in these datasets. The Xiang Zhu et al. [6] represented a framework for automatic classification of Chinese article. The methodology uses sentence extraction techniques and word vector model with CNN. Approach tested on Sina news dataset in different categories from the year 2005 to year 2011. CNN with word2vec model trained by wikipedia corpus (wiki + convolutional neural network) and sina news (wiki+sinanews). sinanews+ CNN has given better performance than SVM. Sentiment analysis based on the text using CNN is investigated in [7]. Two datasets namely MR (movie review) and STS Gold dataset used for experimentation. Authors suggested future research by considering word2vec, multilayer CNN and larger training dataset. Mark Hughes et al. represented CNN method to classify two datasets from the medical domain at sentence level [8]. The model is analyzed on various combinations namely logR & Doc2Vec, zero Mean & Word2Vec, elimMean & Word2Vec, bag of words & logR and CNN and Word2Vec. CNN based approach found better than other approaches. Paper [15] represented a survey of various classifiers. Many methods analyzed for feature selection and algorithms have presented the survey on classifiers and text classification. The survey included Rocchio’s algorithm, KNN, naïve bayes, decision rule, decision tree, support vector machine, neural network and LLSF. Results of different variations of convolutional neural network namely CNN-rand, CNN-static, CNN-non-static and CNN multichannel with pre-trained word vectors for sentence-level classification are reported in [19]. Experiments carried on seven benchmark datasets namely MR, SST-1, SST-2, Subj, TREC, CR and MPQA. Yoon Kim et al. reported that CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks. Deep CNN on sentiment analysis of short texts is presented in [21]. It applied for two datasets, Stanford Sentiment Treebank and the Stanford Twitter Sentiment corpus. The SSTb corpus achieved with 85.7% accuracy for single sentence sentiment prediction. For STS corpus obtained sentiment prediction accuracy is 86.4%.

In literature, many researchers used IMDB dataset with 2 classes for performance testing of proposed algorithms. Results of different techniques for text classification in IMDB dataset are available in [3, 5, 23-37]. IMDB dataset for web opinion mining is investigated in [5]. The paper shows the results of lexicon method with 71% classification accuracy. Y. He, et al. [23] tested on multi-domain sentiment dataset and movie review data using self -training from labeled features. Oracle labeling and naïve Bayes achieved best results for sentiment analysis with 81.36% and 82.53% respectively. Better results are reported than existing weakly supervised sentiment classification. Various techniques and approaches for sentiment analysis and opinion mining are reported in [23]. The paper shows result using maximum entropy, naïve Bayes and support vector machine techniques. SVM achieved 82.9% classification accuracy which shows greater results than others. B. Pang, et al. [18] shows the results of unigram support vector machine method for sentiment analysis. Performance of this technique achieved 86.9% classification accuracy. Paper [3] used text classification for web forum opinions in multiple languages namely English and Arabic. Support vector machine method tested on IMDB dataset and achieved 88.04% classification accuracy. Performance of the method increased with better classification results than [5,22-24]. A. Maas, et al. [24] present a technique to learn word vectors for capturing semantic term by using a mixture of supervised and unsupervised techniques. The paper shows the result on various methods namely bag of words, Latent Semantic Analysis, Latent Dirichlet Allocation etc. Semantic bag of words achieved 88.28% classification results which shows better performance than others. Paper [25] represents for sentiment classification on IMDB dataset. Paper implements comprehensive attention recurrent neural network technique. Comprehensive Attention Long Short-Term Memory and Comprehensive Attention Gated Recurrent Network methods show 90.1% classification accuracy respectively. The technique improves classification performance compared with the standard recurrent method. Y. Long, et al. [26] implemented a cognition-based attention model for sentiment analysis. Tested model on four datasets namely IMDB, Yelp14, Yelp13 and IMDB2 with a different number of classes. Paper [27] tested techniques on IMDB dataset and achieved 90.3% accuracy. The paper shows better results on recurrent convolution neural network with highway layers technique. Naïve Bayes and support vector machine classifiers tested on nine benchmark datasets [28]. Support vector machine with naïve Bayes feature bidirectional method achieved 91.2% classification result with improved performance. R. Johnson, et al. [29] evaluated one hot bidirectional long short-term memory technique on four benchmark datasets namely IMDB, 20-newsgroup, Elec, and RCV1. The paper shows 91.86% improved classification accuracy. Paper [30] shows fast dropout training technique for multilayer neural network, regression and classification. Technique evaluated on IMDB dataset and achieved 91.1% classification result. Jiachen Du et al. [31] introduced CNN attention approach with RNN. Tested on various five datasets with different class labels, vocabulary size and average length. Pretrain convolutional-recurrent attention network gives 92.1% result on IMDB dataset. P. Liu, et al. [32] implemented recurrent neural network (RNN) with multitasking learning for text classification. 91.3% classification accuracy achieved on IMDB dataset. Q. Le et al. [33] shows paragraph vectors perform better than bag of word and other text presentation technique. Achieved good results on several text analysis and text classification tasks. The paper shows 92.7% classification results on IMDB dataset. The paper [34] presented adversarial training methods for semi-supervised text classification. Tested methods on five datasets namely IMDB, RCV1, Rotten tomatoes, DBpedia and Elec. The paper shows result of virtual adversarial method with 94.09% accuracy. Performance of text classification improved than previously reported papers in the literature. J. Hong, et al. [35] surveyed and implemented various algorithms namely averaged paragraph vector, long short-term memory, logistic regression with paragraph vector and 2-layer multi-layer perceptron with paragraph vector. Algorithms tested on three text datasets and achieved better performance for text classification. Performance of 2-layer multi-layer perceptron with paragraph vector algorithm gives 94.5% result which shows better than literature.

In literature, many researchers used Yelp polarity review dataset with 2 classes for performance testing of proposed algorithms. Paper [36-38] shows the result of different techniques for text classification in Yelp polarity review dataset. D. Yogatama, et al. [36] presented results on different models of naïve Bayes and long short-term memory. Tested results on six datasets with a different number of class labels and shows better performance than other literature. Discriminative Long Short-Term Memory model achieved 92.6% classification accuracy on Yelp polarity review dataset. The paper [37] evaluated various techniques on three datasets namely TripAdvisor, Yelp, Yelp* with a different number of classes. Performance of n-gram consistency $activity^{+}$ model achieved 91.09% than previously reported literature. Salinca [38] implemented various approaches for automatic sentiment classification by using four machine learning models and two feature extraction methods. Unigram stochastic gradient descent with stop words removing punctuations and handling negations method gives 92.6% classification results on Yelp polarity review dataset.

In literature, many researchers used Amazon review polarity dataset with 2 classes for performance testing of proposed algorithms. Paper [39-42] shows the result of different techniques for text classification in Amazon review polarity dataset. B. Nguy, et al. [39] used text classification for evaluating helpfulness purpose in Amazon reviews. The paper shows result of the deep learning models namely recurrent neural network and long short-term memory. One Layer recurrent neural network and two-layers LSTM achieved 65% result on Amazon review polarity dataset. Paper [40] implemented support vector machine and random forest for regression and classification. Tested models on Amazon review polarity dataset SVM and random forest achieved 57.7% and 75.4% classification accuracy respectively. M. Agarwal, et al. [41] represents Amazon reviews for predicting the usefulness of product review. The paper shows result of various methods namely multinomial model, stemming with stop words removed, logistic regression, bigrams and trigrams with 66.02%, 78.11%, 79.66% and 54.65% classification results respectively. J. Vijaybhaskar, et al. [42] used for sentiment classification or user review purpose. Tested models on Amazon review polarity dataset. Paper shows result on four methods namely multinomial naïve Bayes, logistic regression, stochastic gradient descent classifier and linear support vector machine with 84%, 92%, 91% and 93% classification results respectively.

In literature, many researchers used AG's News Topic Classification, Yahoo! Answers Topic Classification and DBpedia Ontology Classification Datasets with 4, 10 and 14 classes respectively for performance testing of proposed algorithms. Paper [36] shows result of different techniques for text classification in these datasets. D. Yogatama, et al. [36] presents results on different models of naïve Bayes and long short-term memory. Tested results on six datasets with different number of class labels and shows better performance than other literature. Discriminative Long Short-Term Memory model achieved 92.1%, 73.7% and 98.7% classification accuracy on AG's News Topic Classification, Yahoo! Answers Topic Classification and DBpedia Ontology Classification Datasets respectively.

In literature, many researchers used Amazon Review Polarity, Yelp Review Polarity, AG's News Topic Classification, Yahoo! Answers Topic Classification and DBpedia Ontology Classification Datasets with 2, 2, 4, 10 and 14 classes respectively for performance testing of proposed algorithms. Paper [43-46] shows result of different techniques for text classification in these datasets. C. Qiao, et al. [43] implemented two methods of region embeddings namely word context region embedding and context word region embedding for CNN for text classification. Tested on five datasets and achieved better performance than other methods from the literature. Context word region embedding achieved better results for 2, 4 and 14 classes of datasets. Word context region embedding achieved better results for 2 and 10 classes of datasets. The paper [44] implemented text classification on FastText with convolutional neural network. The paper shows better result on various datasets. Performance for text classification evaluated on two sub-techniques namely h=10 and bigram h=10. Paper [45-46] implemented CNN by using character level vector for text classification. Achieved better results on various datasets.

In literature, many researchers used IMDB datasets for performance testing of proposed algorithms. Paper [10, 47-50] shows result of different parameters on CNN for text classification in IMDB datasets. Paper [51] presents results of CNN (random, static, non-static) on MR, SST-1, SST-2, Subj, TREC, Irony, Opi, Tweet and Polite datasets. Different hyperparameters of CNN are tuned for better accuracy. Simple convolutional neural network with one layer of convolution, trained with top of Word2Vec embeddings, performs remarkably well on sentence classification tasks. Paper [52] represented an application of recurrent neural networks and CNN. Tested models on Reuters-21578 and RCV1-V2 datasets. CNN MODEL is used for extracting text features and RNN used for multi-label prediction.

III. Methodology and Implementation Details

This section presents algorithmic and implementation details of convolutional neural network and word embedding technique (Word2Vec) for text classification. Deeplearning4j library is used for implementing the techniques. It is distributed and open source library for deep learning algorithms in java programming language.

A. Word Embedding technique- Word2Vec

Classification algorithm requires input in numeric format. To implement CNN we need to convert words into numeric format. Word embedding techniques assign one-hot vector to each word in the document. This vector put unique index for each word except all zeros [51]. In literature, different word embedding techniques namely random vectors, Word2Vec, Glove (Global vectors for word representation), FastText [48] and Character vector [46] are experimented to construct input matrix for CNN classifier. Word2Vec is a predictive model [51]. Word2Vec is a two-layer neural network that processes text for numeric conversion. The goal of Word2Vec is to combine vectors of similar words in vector space. This vector is used as input for deep learning network or queried to detect the relationship between words. In this paper, skip-gram model is used to predicts source context-words from the target words [51]. Algorithm 1 describes the working of Word2Vec technique.

Parameters namely batch size, minimum word frequency, layer size and learning rate are initialized. Batch size refers to the number of words that the algorithm process at a time. Minimum word frequency is the minimum number of times a word must appear in the corpus. Layer size equals the number of dimensions in the feature space.

```
Algorithm1: Working of Word2Vec technique
Input:  $D$ = input text document.
Output: $W$ = word vector document.
  Step 1.  Load $D$
  Step 2.  while ($D$! = null) do
               Whitespace stripping.
               Tokenization.
               Initialize the model parameters.
               Fit the model to given input.
Store vectors into $W$ file.
```

Algorithm 1. Working of Word2Vec technique.

```
Algorithm 2: Convolutional neural network for text classification
E number of epochs, B number of batch size, F rate of feature maps,
V size of word vectors and $F_s$ filter size.

Input:  $D_{Tr} = \{d_1, d_2, d_3 \dots, d_n\}$ set of n training documents.
        $D_{Te} = \{d_1, d_2, d_3 \dots, d_n\}$ set of n testing documents.
        $W$ = word vector document.
Output: $C$ = class label
  Step 1.  for each ( $D_i$ in $D_{Tr}$) do
             for E epochs do
               while eof ($D_i$) do
                 Transfer sentences into matrix considering B and W
                   Generate F from V and $F_s$ for kernel size on matrix
                   Produce max-pooling on each F
                   Perform fully connected layer with dropout
                   Save $D_i$ into trained module
  Step 2.  for each ( $D_i$ in $D_{Te}$) do
             while eof ($D_i$) do
               Transfer sentences into matrix with B and W
               Load trained module
               Evaluate $D_i$ on trained module.
```

Algorithm 2. Working of convolutional neural network for text classification.

B. Convolutional Neural Network

The output of Word2Vec in the format of numeric is given as input to convolutional neural network.

Tokenization, stemming and stop-word removal operations are performed on the input text. The proposed convolutional neural network model for text classification is on word level. The output layer provides a class label for the given input file and accuracy for training/testing. Algorithm 2 describes the working of convolutional neural network for text classification.

Convolutional neural network has experimented on various parameters. The proposed method used 3 convolutional layers with windows sizes 3, 4, 5 respectively and the stride is 2. After that, a max-pooling layer is activated to the whole sequence on each filter. Finally, the output of each kernel is merged into a unique vector and fed to a fully connected layer. L2 regularization and learning rate values are 0.1 and 0.001 respectively. Word2Vec vectors size is 300. Adam optimizer is used for experimentation. Experiments are conducted with different batch sizes 10, 20, 40, 60, 80, and 100 and different number of epochs 10, 50 and 100. Sigmoid, Tanh, Softplus, SoftMax and ReLU are activation functions are tested. The best number of feature map is depending on datasets. 10, 50, 100, 200, 300 feature map values are used for each region size.

IV. Experimental Details, Results, and Discussion

This section presents experimental details on varying datasets, obtained results and discussion. To test the performance of the proposed methodology, seven benchmark datasets are used.

A. Dataset

Datasets vary with respect to the number of classes, number of training and testing samples. Datasets are collected for implementation purpose from [53]. The details are given in table 1.

Table 1. Datasets used for text classification

	Name of Dataset	Number of Classes	Number of Train Sets	Number of Test Sets
Dataset 1	IMDB	2	25,000	25,000
Dataset 2	Amazon review full score	2	30,00,000	6,50,000
Dataset 3	Amazon review polarity	2	3,000,000	650,000
Dataset 4	Yelp review polarity	2	560,000	38,000
Dataset 5	AG news topic classification	4	1,20,000	7,600
Dataset 6	Yahoo! Answers topic classification	10	1,400,000	60,000
Dataset 7	DBpedia ontology classification	14	1,800,000	200,000

Performance of classifiers is tested using accuracy measure (equation 1). Accuracy depends on true positive, true negative, false positive and false negative values.

$$Accuracy_i = \frac{TP_i+TN_i}{TP_i+TN_i+FP_i+FN_i} \quad (1)$$

Where, TP is True Positive, TN is True Negative, FP is False Positive and FN is False Negative.

B. Experimental Results of CNN

Experiment No. 1: Identifying the suitable number of epochs and batch size

An epoch is one forward pass and backward pass of all training samples. Table 2 shows the accuracy on seven benchmark datasets with varying batch size and epochs. For dataset 1, 4 and 5 best text classification accuracy is obtained at 100th epoch. For remaining datasets, the best accuracy obtained at 50th epoch. The best accuracy obtained for datasets are 90.01%, 62.45%, 94.28%, 93.54%, 88.16%, 72.77% and 96.67% respectively. The accuracy is increased with increase in epoch up to a certain number. Subsequently, it shows a reduction in accuracy.

Dataset 1, 3, 7 got better results on 100 batch size. Dataset 2, 4 achieve improved results on 40 batch. Dataset 5 and 6 achieved better results on 20 and 60 batch sizes respectively. There is no significant impact of batch size on the accuracy of the classifier. Batch size and epochs effects on the accuracy of datasets but it's not common for all datasets.

Experiment No. 2: Comparison of different activation functions

Performance of CNN is tested with different activation functions such as Sigmoid, Tanh, SoftMax, ReLU and Softplus. Table 3 represent effects of activation functions on the accuracy of seven datasets. Results show that the experimented five activation functions gives good accuracy for all datasets except dataset 2 and 6.

For datasets 1, 3, 5 and 6 best results are obtained with ReLU activation function. For the remaining three datasets, hyperbolic tangent activation function gives the best result. Performance of ReLU activation function is good for all the datasets. It is best or close to best in all cases.

Experiment No. 3: Effect of dropout rate on the performance of CNN MODEL

Dropout is a regularization method used to reduce overfitting problem in full connected layer of convolutional neural network. Table 4 shows the results of convolutional neural network with varying dropout rate from 0.1 to 0.7. For dataset 1, 3 and 4 best accuracy obtained with 0.2 dropout rate. For dataset 2, 5 and 7 best accuracy obtained with 0.5 dropout rate. The accuracy is increased with increase in dropout rate to a certain number. Subsequently, it shows a reduction in accuracy.

Table 2. Accuracy (in %) on various batch size and epochs

No. of Epochs	No. of Batch Size	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Dataset 6	Dataset 7
10	10	76.56	54.00	76.23	74.34	76.34	61.36	83.34
	20	77.20	55.65	77.87	75.87	76.89	62.38	84.04
	40	76.43	54.20	78.20	77.85	78.43	64.43	86.64
	60	76.33	52.34	79.26	78.65	80.60	65.22	88.95
	80	77.60	53.10	82.20	79.98	81	65.23	89.17
	100	78.89	54.89	85.11	83.42	82.87	66.85	92.53
50	10	80.55	55.55	87.13	84.92	84.10	69.33	92.94
	20	82.10	60.10	87.32	85.44	85	71.83	95.44
	40	83.45	**62.45**	88.23	89.68	85.76	68.84	95.81
	60	85.76	57.70	90.52	90.84	86.80	**72.77**	94.29
	80	88.32	60.25	93.65	90.44	87.90	69.55	94.35
	100	88.30	61.31	**94.28**	92.34	87.43	64.56	**96.67**
100	10	88.44	56.60	94.23	91.17	88.10	60.56	94.87
	20	89.20	62.33	92.45	92.69	**88.16**	69.23	96.46
	40	89.61	60.20	91.44	**93.54**	87.30	69.57	95.72
	60	89.50	60.76	90.33	90.78	87.40	70.45	95.34
	80	89.40	60.25	90.76	90.78	87	68.84	95.24
	100	**90.01**	60.30	89.25	89.76	86.23	67.66	94.34

Table 3. Comparison of five activation functions (Accuracy is in %)

Datasets	Activation Functions				
	Sigmoid	Tanh	Softplus	SoftMax	ReLU
Dataset 1	89.45	88.22	89.75	89.12	**90.01**
Dataset 2	62.15	**62.58**	62.35	62.45	62.06
Dataset 3	89.46	91.26	93.57	93.28	**94.27**
Dataset 4	92.58	**93.53**	91.25	92.65	93.05
Dataset 5	84.25	85.06	85.52	86.8	**88.16**
Dataset 6	70.45	71.89	70.85	70.55	**72.77**
Dataset 7	94.56	**96.67**	95.88	95.76	94.98

Table 4. Comparison of accuracy (in %) with different dropout rates

Datasets	Different Number of Dropout				
	0.1	0.2	0.3	0.5	0.7
Dataset 1	89.92	**90.01**	86.75	88.74	89.15
Dataset 2	61.15	61.89	60.2	**62.45**	61.59
Dataset 3	93.56	**94.28**	93.88	93.76	93.98
Dataset 4	92.58	**93.53**	91.25	92.65	93.05
Dataset 5	84.25	85.06	85.52	**88.16**	86.22
Dataset 6	72.45	72.52	72.57	72.77	**74.27**
Dataset 7	93.46	94.26	93.57	**96.67**	94.27

Table 5. Effects of feature maps on accuracy (%) of CNN MODEL

Datasets	Number of Feature Maps				
	10	50	100	200	300
Dataset 1	89.89	89.94	90.0	90.00	**90.01**
Dataset 2	62.21	62.35	62.55	62.56	**62.6**
Dataset 3	94.01	94.12	94.28	**94.31**	94.30
Dataset 4	93.23	93.34	93.54	93.57	**93.59**
Dataset 5	88.04	88.12	88.16	88.18	**88.19**
Dataset 6	72.65	72.71	72.77	**72.89**	72.76
Dataset 7	96.21	96.38	96.67	96.70	**96.73**

Table 6. Parameter details of CNN model for best results (parameter value, accuracy of obtained best result in %)

Datasets	Epoch	Batch Size	Activation Function	Dropout rate	Feature Map
Dataset 1	(100, 90.01)	(100, 90.01)	(ReLU, 90.01)	(0.2, 90.01)	(300, 90.01)
Dataset 4	(50, 62.45)	(40, 62.45)	(Tanh, 62.58)	(0.5, 62.45)	(300, 62.6)
Dataset 3	(50, 94.28)	(100, 94.28)	(ReLU, 94.27)	(0.2, 94.28)	(200, 94.31)
Dataset 2	(100, 93.54)	(40, 93.54)	(Tanh, 93.53)	(0.2, 93.53)	(300, 93.59)
Dataset 5	(100, 88.16)	(20, 88.16)	(ReLU, 88.16)	(0.5, 88.16)	(300, 88.19)
Dataset 6	(50, 72.77)	(60, 72.77)	(ReLU, 72.77)	(0.7, 74.27)	(200, 72.78)
Dataset 7	(50, 96.67)	(100, 96.67)	(Tanh, 96.68)	(0.5, 96.67)	(300, 96.73)

Experiment No. 4: Selection of the number of feature maps

Table 5 reported classification accuracy on different number feature maps on convolutional neural network classifier. Results show that the performance of CNN changes with values of feature maps. Best results are obtained with 200 and 300 feature maps. Table 5 indicates that performance increases with a number of feature maps. Increase in feature maps takes a longer time to train the model.

Table 6 summaries the best results obtained with different CNN parameters experimented in this section.

C. Comparison of results with techniques other than CNN

This subsection presents a comparison of CNN with literature.

Table 7. Comparison of results for dataset 1 (IMDB dataset)

Reference	Techniques other than CNN	Accuracy in %
[5]	Lexicon	71
[22]	Lexicon Labeling	66.9
	Heuristic Labeling	71.2
	Self-instance instances	73.2
	Self-Learned features	74.7
	Oracle Labeling	81.36
	Naïve Baye’s	82.53
[23]	Maximum Entropy	81
	Naïve Baye’s	81.5
	Support Vector Machine	82.9
[18]	Support Vector Machine – unigram	86.9
[3]	Support Vector Machine	88.04
[24]	Latent Dirichlet Allocation	67.42
	Bag of words (bnc)	87.8
	Bag of words (bΔt’c)	88.23
	Latent Semantic Analysis	83.96
	Full + Bag of words (bnc)	88.33
	Full + Un-labelled + Bag of words	88.89
	Semantic	87.30
	Full	87.44
	Full + Additional Unlabeled	87.99
	Semantic + Bag of Words (bnc)	88.28
[25]	Comprehensive Attention-Recurrent Neural Network	89
	Comprehensive Attention - Long Short-Term Memory	90.1
	Comprehensive Attention - Gated Recurrent Network	90.1
[26]	Long Short-Term Memory + Cognition Based Attention + Local Text Context Based Attention Model$_{parallel}^{GECO}$	90.1
[27]	Recurrent Convolution Neural Network-Highway	90.3
[28]	Multinomial Na ïve Bayes-unidirectional	83.55
	Multinomial Na ïve Bayes – bidirectional	86.59
	Support Vector Machine-unidirectional	86.95
	Support Vector Machine with Na ïve Bayes Feature – unidirectional	88.29
	Support Vector Machine-bigram	89.2
	Support Vector Machine with Na ïve Bayes Feature – bidirectional	91.2
	Multinomial Na ïve Bayes-unidirectional	83.55
	Multinomial Na ïve Bayes – bidirectional	86.59
[29]	One hot bidirectional-Long Short-Term Memory	91.86
[30]	F-Dropout	91.1
[31]	Long Short-Term Memory	88.5
	N Bag of Words	80.7
	Long Short-Term Memory + Recurrent Neural Network + attention	90.6
	Convolutional-Recurrent Attention Network random	92.0
	Convolutional-Recurrent Attention Network pretrain	92.1
[32]	Multi-Task (Liu)	91.3
[33]	Paragraph vector	92.7
[34]	Virtual Adversarial	94.09
[35]	Averaged Paragraph Vector	88.3
	Long Short-Term Memory	89.1
	Paragraph Vector (Logistic Regression)	94.4
	Paragraph Vector (2 Layer MultiLayer Perceptron)	**94.5**
This paper	Proposed Methodology	90.01

Table 7 shows a comparison of results on dataset 1 (IMDB dataset). IMDB dataset is used by many researchers for text classification with Na ïve Bayes, Support Vector Machine, Long Short-Term Memory etc. In many cases, deep learning algorithms are found better than other classification algorithms namely Na ïve Bayes and support vector machine. CNN shows good performance with 90.01% accuracy which is better than many others. In literature, it is reported that hybrid/combined methods provide more accuracy than individual methods. Results show that CNN gives better accuracy than hybrid models of Na ïve Bayes and support vector machine. The performance of LSTM in paper [25-26,29], multi-layer perceptron in [35], paragraph vector in [33] is found better than CNN. The hybrid models of LSTM show better accuracy than CNN. The proposed convolutional neural network model for text classification is on word level. Paragraph vector techniques presented in [33, 35] shows better results than the proposed CNN.

Table 8 shows the accuracy of the proposed methodology and different machine learning methods in literature for dataset 4 (Yelp Review Polarity dataset). Paper [38] investigated linear support vector classification, stochastic gradient descent, logistic regression, na ïve Bayes, logistic regression, linear support vector classification with different processing steps. It shows a small variation in accuracy. The proposed methodology is found better with 93.54% accuracy than support vector machine, na ïve Bayes, logistic regression, Stochastic Gradient Descent, long short-term memory and hybrid models etc.

Table 8. Comparison of results for dataset 4 (Yelp Review Polarity dataset)

Reference	Techniques other than CNN	Accuracy in %
[36]	Multi-Layer Perceptron Naive Bayes	73.6
	Kneser–Ney Bayes	81.8
	Na ïve Bayes	86.0
	Generative Long Short-Term Memory–shared a comp.	88.2
	Generative Long Short-Term Memory–independent comp.	90.0
	Discriminative Long Short-Term Memory	92.6
[37]	Doc2vec	64.84
	Doc2vec +ARI + Sentiment	65.01
	Activity + Rating	74.68
	Activity + Rating + Elite + Check-in	79.43
	Unigram + Bigram	73.63
	Consistency	76.5
	$Activity\ Model^{-}$	80.24
	$Activity\ Model^{+}$	86.35
	N-gram + Consistency	79.72
	N-gram + $Activity^{-}$	82.84
	N-gram + $Activity^{+}$	88.44
	N-gram + Consistency + $Activity^{-}$	86.58
	N-gram + Consistency + $Activity^{+}$	91.09
	M_{yelp}	89.87
[38]	Linear Support Vector Classification + Unigrams + Part-of speech + Word sense disambiguation	78.6
	Stochastic Gradient Descent + Unigrams + Part-of speech + Word sense disambiguation	78.6
	Logistic Regression + Unigrams + Part-of speech + Word sense disambiguation	78.7
	Na ïve Bayes + Unigrams + Part-of speech + Word sense disambiguation	78.7
	Na ïve Bayes + unigram + Stop words + without removing punctuations + without Handling Negations	88.1
	Na ïve Bayes + unigram + Stop words + without removing punctuations + without Handling Negation	88.3
	Na ïve Bayes + unigram + Stop words + with removing punctuations + with Handling Negations	89.7
	Logistic Regression + unigram + Stop words + without removing punctuations + without Handling Negations	89.3
	Logistic Regression + unigram + Stop words + without removing punctuations + without Handling Negation	89.8
	Logistic Regression + unigram + Stop words + with removing punctuations + with Handling Negations	90.0
	Linear Support Vector Classification + unigram + Stop words + without removing punctuations + without Handling Negations	91.7
	Linear Support Vector Classification + unigram + Stop words + without removing punctuations + without Handling Negation	91.9
	Stochastic Gradient Descent + unigram + Stop words + without removing punctuations + without Handling Negations	91.4
	Stochastic Gradient Descent + unigram + Stop words + without removing punctuations + without Handling Negation	91.1
	Linear Support Vector Classification + unigram + Stop words + with removing punctuations + with Handling Negations	92.3
	Stochastic Gradient Descent + unigram + Stop words + with removing punctuations + with Handling Negations	92.6
This paper	**Proposed Methodology**	**93.54**

Table 9 shows the results comparison of proposed methodology with the literature on dataset 3 (Amazon Review Polarity dataset). Performance of proposed methodology is better than papers in the literature reported in Table 9. The proposed CNN shows better accuracy than Recurrent Neural Network, LSTM, Support Vector Machine, Random Forest, Logistic Regression, Linear Support Vector Machine, Stochastic Gradient Descent Classifier etc. The accuracy obtained using CNN is 94.28%.

Dataset 1, 4 and 3 are benchmarks datasets with two classes. CNN provides better results with more than 90% accuracy. Except dataset 1, CNN gives better results than all other classifiers including hybrid models of deep learning algorithms.

Table 10 shows the results comparison of proposed methodology with the literature on dataset 5, 6 and 7. Performance of LSTM in the paper [36] is better than the proposed CNN methodology. Dataset 5, 6 and 7 are benchmark datasets with 4, 10 and 14 classes. Long short-term memory with generative model and discriminative model works better than CNN.

Table 9. Comparison of results for dataset 3 (Amazon Review Polarity dataset)

Reference	Techniques other than CNN	Accuracy in %
[39]	Bigram bag-of-words	58.2
	Unigram bag-of-words with random forest	61.9
	One Layer Recurrent Neural Network	65
	two-layers LSTM	65
[40]	Support Vector Machine	57.7
	Random Forest	75.4
[41]	Multinomial Model	66.02
	Stemming -+ Stop words removed	78.11
	Bigrams + Trigrams	79.66
	Logistic Regression	54.65
[42]	Multinomial Na ïve Bayes	84
	Logistic Regression	92
	Stochastic Gradient Descent Classifier	91
	Linear Support Vector Machine	93
This paper	**Proposed Methodology**	**94.28**

Table 10. Comparison of results (accuracy values are in %)

Reference	Technique	Dataset 5	Dataset 6	Dataset 7
[36]	Na ïve Bayes	90	68.7	96.0
	Kneser-ney Bayes	89.3	69.3	95.4
	Machine learning program na ïve Bayes	89.9	60.6	87.2
	Discriminative long short-term memory	**92.1**	**73.7**	**98.7**
	Generative long short-term memory independent component	90.7	70.5	94.8
	Generative long short-term memory shared component	90.6	69.3	95.4
This paper	Proposed Methodology	88.16	72.77	96.67

Table 11. Comparison of results for dataset 3 to 7 (accuracy values are in %)

Reference	Technique	Dataset 3	Dataset 4	Dataset 5	Dataset 6	Dataset 7
[43]	Word context region embedding with CNN	95.1	**96.4**	92.8	**73.7**	**98.9**
	Context word region embedding with CNN	**95.3**	96.2	**92.8**	73.4	**98.9**
[44]	FastText, h=10 with CNN	91.2	93.8	91.5	72.0	98.1
	FastText, h=10, bigram with CNN	94.6	95.7	92.5	72.3	98.6
[45]	Character level CNN	94.1	94.5	91.4	71.7	98.57
[46]	Large Word2Vec ConvNets	94.12	95.4	90.08	68.03	98.58
	Small Word2Vec ConvNets	94	94.44	88.65	68.5	98.29
	Large Word2Vec ConvNets thesaurus	94.2	95.37	90.09	68.77	98.36
	Small Word2Vec ConvNets thesaurus	94.34	94.64	89.12	70.14	98.47
	Large lookup table ConvNets	94.16	95.11	91.45	70.94	98.28
	Large lookup table ConvNets	94.15	94.46	89.13	69.8	98.50
	Large lookup table ConvNets thesaurus	94.48	94.97	91.06	71.16	98.42
	Large lookup table ConvNets thesaurus	94.49	94.63	90.88	71.08	98.23
	Large full ConvNets	94.22	94.75	90.15	70.1	98.34
	Small full ConvNets	94.22	94.33	88.41	69.99	98.11
	Large full ConvNets thesaurus	94.49	95.12	90.49	70.42	98.45
	Small full ConvNets thesaurus	94.34	94.58	89.11	70.1	98.31
	Large ConvNets	94.49	94.11	87.18	70.45	98.27
	Small ConvNets	94.5	93.47	84.35	70.16	98.02
	Large ConvNets thesaurus	95.07	94.18	86.61	71.2	98.40
	Small ConvNets thesaurus	94.35	93.51	85.2	70.16	98.15
This paper	Proposed Methodology	94.28	93.54	88.16	72.77	96.67

Table 12. Comparison of results for dataset 1 (IMDB dataset) with other CNN models

Parameters	[50]	[48]	[47]	[49]	[10]	This paper
Accuracy (%)	76.67	77.5	84.87	88.74	89.16	**90.01**
Word embedding technique	NA	Word2Vec, Glove, Fast text	Word2Vec, Glove, Fast text	Word2Vec	NA	Word2Vec
Window size	NA	3,4,5	3,4,5	NA	NA	3,4,5
Dropout	NA	0.5	0.5	NA	0.4	0.2
Batch size	NA	64	64	100	NA	100
Learning rate	NA	NA	0.001	0.003	NA	0.001
Vector size	NA	100	100	300	NA	300
L2	NA	0.1	0.1	NA	NA	0.1

NOTE: NA indicates not available

D. Comparison of results with CNN from literature

The methodology of this paper is compared with other CNN methodology. Table 11 summarized the accuracy of CNN approaches used in the different paper for text classification on benchmark datasets 3 to 7. The performance is CNN variations reported in these papers is similar. The difference in percentage accuracy is in the range of 2-4 percentages. Results from table 11 show that different CNN models proposed in literature give better accuracy for dataset 3 to 7. The methodology proposed in [43], CNN with word context region embedding and context word region embedding is better than other CNN variations. For dataset 3 and 4, results from table 8, 9 and 11 shows that accuracy of proposed CNN model is better than other techniques and close the best CNN model reported in [43].

The accuracy of all the classification techniques is less for dataset 2 and 6. For the remaining dataset the accuracy is more than 90%. There is scope to improve the performance of classification techniques.

Comparison of various parameters of convolutional neural network for dataset 1 (IMDB dataset) is summarized in Table 12. Results show that tuning of hyperparameters namely dropout rate, batch size, learning rate, vector size has the effect of performance of CNN.

V. CONCLUSIONS

In literature, different machine learning algorithms have experimented for text classification. To test the performance of CNN models, seven benchmark datasets with varying classes from 2 to 14 are used. Performance of CNN models are compared with other techniques. As compared to other text classifiers, CNN models give the best accuracy for all seven datasets except dataset 1.

This paper presents an investigation of convolutional neural network with variations of hyperparameters namely activation function, batch size, epochs, dropout rates, feature map values for text classification. Results on seven-benchmark dataset show that CNN with word embedding technique gives good results. Comparison of obtained results with literature shows that CNN gives best accuracy for two datasets and close to the best for the remaining five datasets. The result shows that the performance of text classifiers varies with a number of classes, training and testing samples. Best obtained batch size and epochs are different for tested datasets. ReLU activation function is found better for four datasets. Results show that the accuracy is increased with increase in epochs and dropout rate to a certain number. Subsequently, it shows a reduction in accuracy. CNN performance increases with a number of feature maps. But, it takes a longer time to train the model.

REFERENCES

[1] M. Ikonomakis, S. Kotsiantis and V. Tampakas, “Text Classification Using Machine Learning Techniques”, *WSEAS transactions on computers*, vol. 4, no. 8, pp. 966-974, 2005

[2] R. Talib, M. Kashif Hanif, S. Ayesha and F. Fatima, “Text Mining: Techniques, Applications and Issues”, *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, 2016. http://dx.doi.org/10.14569/IJACSA.2016.071153

[3] A. Abbasi, H. Chen and A. Salem, “Sentiment analysis in multiple languages: Feature selection for opinion classification in Web forums”, *ACM Transactions on Information Systems (TOIS)*, vol. 26, no. 3, p. 12, 2008. http://dx.doi.org/10.1145/1361684.1361685

[4] S. Günal, S. Ergin, M. Gülmezoğlu and Ö. Gerek,“On Feature Extraction for Spam E-Mail Detection“, International Workshop on Multimedia Content Representation, Classification and Security, 2006, pp. 635--642. http://dx.doi.org/10.1007/11848035_84

[5] A. Harb, M. Planti é, G. Dray, M. Roche, F. Trousset and P. Poncelet, “Web Opinion Mining: How to extract opinions from blogs?”, *Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology*, pp. 211--217, 2008. http://dx.doi.org/10.1145/1456223.1456269

[6] X. Zhu, J. Huang, Z. Zhou and Y. Han, “Chinese Article Classification Oriented to Social Network Based on Convolutional Neural Networks”, *Data Science in Cyberspace (DSC), IEEE International Conference on*, pp. 33--36, 2016. http://dx.doi.org/10.1109/DSC.2016.28

[7] S. Liaoa, J. Wangb, R. Yu, K. Sato and Z. Cheng, “CNN for situations understanding based on sentiment analysis of twitter data”, *Procedia computer science*, vol. 111, pp. 376--381, 2017. http://dx.doi.org/10.1016/j.procs.2017.06.037

[8] M. Hughes, I. Li, S. Kotoulas and T. Suzumura, “Medical text classification using convolutional neural networks”, *Stud Health Technol Inform*, vol. 235, pp. 246--50, 2017.

[9] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. Trippe,

J. Gutierrez and K. Kochut, "A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques*", arXiv preprint arXiv:1707.02919*, 2017.

[10] C. Liu, S. Zhao and M. Volkovs, "Learning Document Embeddings With CNNs." *arXiv preprint arXiv:1711.04168*, 2017.

[11] R. Duda, P. Hart and D. Stork, "Pattern Classification", 2012.

[12] Y. Yang and X. Liu, "A re-examination of text categorization methods", *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 42--49, 1999. http://dx.doi.org/10.1145/312624.312647

[13] P. Bednár, "Active learning of SVM and decision tree classifiers for text categorization", Fourth Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence, Herlany, Slovakia, 2006.

[14] W. Wang, D. Do and X. Lin,"Term graph model for text classification", *International Conference on Advanced Data Mining and Applications*, pp. 19--30, 2005. http://dx.doi.org/10.1007/11527503_5

[15] V. Korde and C. Mahender, "Text classification and classifiers: A survey", *International Journal of Artificial Intelligence \& Applications*, vol. 3, no. 2, p. 85, 2012.

[16] S. Kamruzzaman, F. Haider and A. Hasan, "Text classification using data mining", *arXiv preprint arXiv:1009.4987*, 2010.

[17] B. Pang and L. Lee,"Opinion mining and sentiment analysis", *Foundations and Trends in Information Retrival*, vol. 2, No. 1-2, pp. 1--135,2008. http://dx.doi.org/10.1561/1500000011

[18] V. Gupta, "Recent trends in text classification techniques", *International Journal of Computer* Applications, vol. 35, no. 6, 2011.

[19] Y. Kim, "Convolutional neural networks for sentence classification*", arXiv preprint arXiv:1408.5882*, 2014. http://dx.doi.org/10.3115/v1/D14-1181

[20] A. Hassan and A. Mahmood,"Deep learning for sentence classification.", Systems, Applications and Technology Conference (LISAT), 2017 IEEE Long Island, 2017, pp. 1--5. http://dx.doi.org/10.1109/LISAT.2017.8001979

[21] C. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts", *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 69--78, 2014.

[22] Y. He and D. Zhou, "Self-training from labeled features for sentiment analysis", *Information Processing \& Management*, vol. 47, no. 4, pp. 606--616, 2011. http://dx.doi.org/10.1016/j.ipm.2010.11.003

[23] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts", Proceedings of the 42nd annual meeting on Association for Computational Linguistics, 2004, p. 271, http://dx.doi.org/10.3115/1218955.1218990

[24] A. Maas, R. Daly, P. Pham, D. Huang, A. Ng, and C. Potts, "Learning word vectors for sentiment analysis", Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1, 2011, pp. 142--150.

[25] Y. Zhang, M. Er, R. Venkatesan, N. Wang and M. Pratama, "Sentiment classification using comprehensive attention recurrent models", *Neural Networks (IJCNN), 2016 International Joint Conference on*, pp. 1562--1569, 2016. http://dx.doi.org/10.1109/IJCNN.2016.7727384

[26] Y. Long, Q. Lu, R. Xiang, M. Li and C. Huang, "A cognition based attention model for sentiment analysis", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 462--471, 2017.

[27] Y. Wen, W. Zhang, R. Luo and J. Wang, "Learning text representation using recurrent convolutional neural network with highway layers", *arXiv preprint arXiv:1606.06905*, 2016.

[28] S. Wang and C. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification", Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2, 2012, pp. 90--94.

[29] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using LSTM for region embeddings", *arXiv preprint arXiv:1602.02373,* 2016.

[30] S. Wang and C. Manning, "Fast dropout training". international conference on machine learning, 2013, pp. 118--126.

[31] J. Du, L. Gui, R. Xu and Y. He, "A Convolutional Attention Model for Text Classification", *National CCF Conference on Natural Language Processing and Chinese Computing,* pp. 183--195, 2017. http://dx.doi.org/10.1007/978-3-319-73618-1_16

[32] P. Liu, X. Qiu and X. Huang, "Recurrent neural network for text classification with multi-task learning", *arXiv preprint arXiv:1605.05101,* 2016.

[33] Q. Le and T. Mikolov, "Distributed representations of sentences and documents", *International Conference on Machine Learning*, pp. 1188--1196, 2014.

[34] T. Miyato, A. Dai and I. Goodfellow, "Adversarial training methods for semi-supervised text classification", *arXiv preprint arXiv:1605.07725*, 2016.

[35] J. Hong and M. Fang, "Sentiment analysis with deeply learned distributed representations of variable length texts", Technical report, Stanford University, 2015.

[36] D. Yogatama, C. Dyer, W. Ling, and P. Blunsom, "Generative and discriminative text classification with recurrent neural networks", *arXiv preprint arXiv:1703.01898*, 2017.

[37] S. Mukherjee, S. Dutta, and G. Weikum, "Credible Review Detection with Limited Information using Consistency Analysis", *arXiv preprint arXiv:1705.02668*, 2017.

[38] A. Salinca, "Business reviews classification using sentiment analysis", 2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2015, pp. 247--250. http://dx.doi.org/10.1109/SYNASC.2015.46

[39] B. Nguy, "Evaluate Helpfulness in Amazon Reviews Using Deep Learning."

[40] K. Jónsson and D. Platt, "CSE 255 Assignment 1: Helpfulness in Amazon Reviews."

[41] M. Agarwal and M. Bhand, "Classification of Amazon Reviews.", 2009.

[42] J. Vijayabhaskar, R. Sridhar and P. Vijayaragavan, "User Review Sentiment Classification and Aspect Extraction", *Imperial Journal of Interdisciplinary Research*, vol. 3, no. 4, 2017.

[43] C. Qiao, B. Huang, G. Niu, D. Li, D. Dong, W. He, D. Yu and H. Wu, "A new method of region embedding for text classification.", *ICLR,* 2018

[44] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov, "Bag of tricks for efficient text classification." *arXiv preprint arXiv:1607.01759*, 2016. http://dx.doi.org/10.18653/v1/E17-2068

[45] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent

layers." *arXiv preprint arXiv:1602.00367*, 2016.
[46] X. Zhang, J. Zhao and Y. LeCun, "Character-level convolutional networks for text classification", Advances in neural information processing systems, 2015, pp. 649--657.
[47] G. Lee, J. Jeong, S. Sao, C. Kim and P. Kang, "Sentiment classification with word localization based on weakly supervised learning with a convolutional neural network." *Knowledge-Based Systems*, vol. 152, pp.70-82, 2018. http://dx.doi.org/10.1016/j.knosys.2018.04.006
[48] G. Lee, J. Jeong, S. Seo, C. Kim and P. Kang1, "Sentiment Classification with Word Attention based on Weakly Supervised Leaning with a Convolutional Neural Network", *arXiv preprint arXiv:1709.09885*, 2017.
[49] H. Bedi and N. Cheke, "Sentiment Analysis of Movie Reviews ", 2012.
[50] S. Chintala, "Sentiment Analysis using neural architectures." *New York University*, 2012.
[51] A. Mandelbaum and A. Shalev, "Word embeddings and their use in sentence classification tasks", *arXiv preprint arXiv:1610.08229*, 2016.
[52] G. Chen, D. Ye, Z. Xing, J. Chen and E. Cambria, "Ensemble application of convolutional and recurrent neural networks for multi-label text categorization", *Neural Networks (IJCNN), 2017 International Joint Conference on*, pp. 2377--2383, 2017. http://dx.doi.org/10.1109/IJCNN.2017.7966144
[53] https://drive.google.com/drive/u/0/folders/0Bz8a_Dbh9Qhbfll6bVpmNUtUcFdjYmF2SEpmZUZUcVNiMUw1TWN6RDV3a0JHT3kxLVhVR2M

Authors' Profiles

Dr. Amol C. Adamuthe received Ph.D. from Mumbai University and Master of Technology in Computer Engineering from Dr. B. A. Technological University, Lonere, MS, India. He is currently an Assistant Professor at Rajarambapu Institute of Technology, Rajaramnagar, MS, India. His areas of interest are technology forecasting, optimization, soft computing and cloud computing.

Sneha N. Jagtap completed M. Tech. in Computer Science and Engineering from Rajarambapu Institute of Technology, Rajaramnagar in year the 2018. Her areas of interest are text classification, machine learning and web development.

How to cite this paper: Amol C. Adamuthe, Sneha Jagtap, "Comparative Study of Convolutional Neural Network with Word Embedding Technique for Text Classification", International Journal of Intelligent Systems and Applications(IJISA), Vol.11, No.8, pp.56-67, 2019. DOI: 10.5815/ijisa.2019.08.06