Arabian Journal for Science and Engineering (2019) 44:3305–3314
https://doi.org/10.1007/s13369-018-3500-z

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Deep Learning Based Sentiment Analysis Using Convolution Neural Network

Sujata Rani[1] · Parteek Kumar[1]

Received: 12 July 2017 / Accepted: 30 July 2018 / Published online: 10 August 2018

Abstract

Sentiment analysis (SA) of natural language text is an important and challenging task for many applications of Natural Language Processing. Till now, researchers have used different types of SA techniques such as lexicon based and machine learning to perform SA for different languages such as English, Chinese. Inspired by the gain in popularity of deep learning models, we conducted experiments using different configuration settings of convolutional neural network (CNN) and performed SA of Hindi movie reviews collected from online newspapers and Web sites. The dataset has been manually annotated by three native speakers of Hindi to prepare it for training of the model. The experiments are conducted using different numbers of convolution layers with varying number and size of filters. The CNN models are trained on 50% of the dataset and tested on remaining 50% of the dataset. For the movie reviews dataset, the results given by our CNN model are compared with traditional ML algorithms and state-of-the-art results. It has been observed that our model is able to achieve better performance than traditional ML approaches and it has achieved an accuracy of 95%.

Keywords Deep learning · Sentiment analysis · Movie reviews · CNN

1 Introduction

Internet has become an essential part of people's life with the rapid growth of information technology. Mostly the people share their opinions about different entities such as products, services on different platforms like Web forums, blogs, social media sites, etc. These platforms consist of valuable information about a variety of domains, from commercial to political and social applications [1]. The analysis of this enormous amount of data is difficult to handle manually. In this case, sentiment analysis (SA) has been proved very useful. It is basically getting an idea of how people feel and what they are talking about. SA has become a hot research topic in the field of Natural Language Processing (NLP). It is the task of extracting automatically subjective information conveyed by a text. It covers extracting the polarity, or identifying the target (e.g., a product and a measure) or the holder of an opinion. SA can be performed at feature, sentence and, document level. SA is used in many other applications such as predicting customer trends, marketing strategies and stock market prediction, education to improve teaching and learning [2]. One of the important applications of SA is that it helps political parties or government to get an idea of what are the chances of their winning in coming elections and how much the public is satisfied with their policies as predicted during US elections 2016.

In the field of NLP, researchers are performing SA by experimenting with different tools and techniques such as rule based or machine learning (ML). ML is a term used to refer to computers learning from examples. There are many models of ML that support learning such as Naïve Bayes (NB), support vector machines (SVM), logistic regression, random forests. One area of ML is neural networks (NNs). NNs are a very rapidly growing field that further break out to various forms, such as convolutional neural network (CNN) and recurrent neural network (RNN). Deep learning refers to the number of layers that comprise the NN. Early NNs were defined with three layers: input, hidden and output. Adding several hidden layers makes the NN 'deep' and enables it to learn more subtle and complex relationships. The number of 'hidden layers' decides how deep the network is. Thus, 'deep learning' is a subfield of ML which uses deep neural

✉ Sujata Rani
sujata.singla@thapar.edu

Parteek Kumar
parteek.bhatia@thapar.edu

1 CSED, Thapar University, Patiala, India

networks. Recently, deep learning techniques have revealed great improvements over other existing ML techniques in all fields of NLP such as computer vision, entity recognition and speech recognition. The major reason behind the popularity of the deep learning models over the existing ML algorithms is improvement in accuracy. However, the accuracy of an ML algorithm is highly dependent on good feature representation of data. One main advantage of deep learning models over existing ML models is that they rely on extensive feature engineering. While traditional ML algorithms are based on handcrafted features, this is both time-consuming and leads to over-specified and incomplete features. However, the deep learning requires high computation power and storage due to increase in number of hidden layers as compared to traditional ML algorithms, but there is greater improvement in accuracy. This motivated us to perform SA for Hindi language using deep learning.

Hindi is the 4th most spoken language across the world and is widely used. This language has 422 million speakers which cover about 41% of total population of India [3]. From the last few years, with the introduction of Unicode standards, contents of Web pages in Hindi language have been increasing rapidly. Our work presented in this paper performs sentiment analysis for Hindi language content using CNN.

The main contributions of the paper can be summarized as follows. The paper presents the work on sentiment analysis for resource-poor Hindi language. The dataset of the Hindi movie reviews has been manually annotated at sentence level by three native speakers of Hindi to apply deep learning model. This dataset has been labeled into three sentiment classes, i.e., positive, negative and neutral. Then, sentiment analysis of movie reviews dataset has been performed using deep learning model, i.e., CNN. Several experiments of CNN have been conducted by setting different parameters such as number of convolution layers, number of hidden layers, number of filters, output dimension, regularizer, dropout and filter size to analyze the accuracy and training time of the models.

The rest of the paper is structured as follows. Section 2 discusses about the previous research work carried out by researchers on SA using ML and deep learning techniques on different languages. Section 3 describes the CNNs in brief, and the implementations of different CNN models are discussed in Sect. 4. Section 5 covers the experimental setup, and results given by the proposed system along with the comparison of results with existing works as well as traditional ML algorithms are discussed in Sect. 6. Finally, Sect. 7 concludes the paper and presents future implications.

2 Previous Research

A comprehensive research has been performed on SA for Hindi language using machine learning as well as lexicon-based techniques, some of which are summarized as follows. Joshi et al. [4] first attempted to work on SA for Indian languages by developing their own lexical resource Hindi SentiWordNet (HSWN) and achieved an accuracy of 78.14 %. After this, Mittal et al. [5] performed SA of movie reviews using HSWN and achieved an accuracy of 80.21%. Arora [6] performed SA on a corpus of Hindi reviews and blogs related to products and movies using subjective lexicon and N-gram approaches. Sharma et al. [7] proposed SA system using an unsupervised dictionary-based approach. Sharma and Bhattacharya [8] proposed a bootstrap approach to extend the HSWN using existing Hindi WordNet and used this lexicon to validate the SA system by achieving an accuracy of 87%. Pandey and Govilkar [9] and Sharma et al. [10] used unsupervised lexicon-based approach to perform SA. Recently, Akhtar et al. [11] performed aspect-based SA of product reviews by creating their own annotated dataset for Hindi and achieved an accuracy of 54.05%.

The researchers have experimented using deep learning models to analyze sentiments since deep learning has got more popularity in recent years because of its accuracy and automatic feature engineering. Some of the research works using deep learning techniques are summarized as follows. Shirani-Mehr [12] compared different deep learning algorithms like RNN, CNN and CNN with Naïve Bayes (NB) on a movie reviews dataset. Santos and Gattti [13] proposed a deep CNN to perform SA of movie reviews and Twitter messages and achieved an accuracy of 85.7 and 86.4%, respectively. Stojanovski et al. [14] analyzed the performance of CNN for SA of Twitter and achieved an F1-score of 64.85% on Twitter 2015 dataset which was comparable to the existing traditional approaches to SA. Ouyang et al. [15] proposed a framework for CNN using Word2Vec to perform SA. They used the parametric rectified linear unit (PReLU), normalization and dropout technology to improve the accuracy of the model. Singhal et al. [16] compared different variations of deep learning models such as basic neural networks, CNN, RNN and long short-term memory (LSTM) with different ML algorithms and concluded deep learning to be more effective than ML. Zhang and Chen [17] performed SA on Chinese corpus using CNN and analyzed that CNNs perform better than traditional ML techniques. Hassan and Mahmood [18] proposed ConvLstm neural network architecture for SA of short texts and achieved an accuracy of 88.3% for binary classification and 47.5% for five-class classification problem. Zhang et al. [19] proposed a lexicon-based sentiment CNN model for SA of Chinese text. They concluded that their model performs better with word embedding features only.

From the literature review, it has been observed that researchers have experimented with different versions of neural networks such as CNN, RNN and LSTM using different input representations, i.e., word embedding or sentiment features using lexicon. Also, the majority of research work has been performed on resource-rich English and Chinese languages. There exists a little research work on resource-poor languages such as Hindi language due to the lack of sufficient resources such as polarity lexicons, lexical tools and annotated corpora. Till now, mainly the researchers have used lexicon-based and other traditional ML algorithms to perform SA for Hindi. In this work, a sentiment analysis system has been developed for Hindi language. Due to the unavailability of polarity labeled dataset for Hindi, a corpus of Hindi movie reviews has been constructed by annotating it manually for experimentation.

In this work, SA of Hindi text has been performed using CNN. The reason behind the use of CNN is that CNNs have shown greater improvements over image classification and NLP recently. As in case of CNN, we just need to label the whole sentences artificially, which avoids the huge work in comparison with other deep learning models such as RNN. CNN can extract an area of features from global information, with the convolution operation, a piece of data information can be extracted together as the features and it is able to consider the relationship among these features. Therefore, the same operation can be applied on text data to make the input features to the model that can be trained in another effective way for sentiment analysis. For experimentation, 12 CNN models have been built by varying the parameters such as number of convolution layers, number of filters and filter size to analyze the performance of model. The performance of the proposed CNN model has also analyzed over existing ML techniques. In the next section, a brief description about the neural network and CNN is discussed.

3 Neural Network

A neural network is a cascade of neuron layers with output of one layer fed as input to the next successive layer. Each layer passes on the modified version of data to the next layer to promote more informative features further. Neural networks cannot process direct words, but they work on word embedding or more specifically feature vectors representing those words [15]. As neural networks learn features from the task in hand, they can adapt to any domain.

3.1 Convolutional Neural Network (CNN)

CNNs are almost similar to the ordinary neural networks. Like ordinary neural networks, neurons in CNNs take some input, process it and propagate it further. The difference is that CNNs explicitly assume input as images. This is the reason they are explicitly used for analyzing image data. Regular neural networks do not scale well to full images [12]. For small dimensions, these are manageable, but as the dimensions grow, more neurons and parameters are required leading to the problem of overfitting. A CNN is a feed-forward neural network which consists of four layers. First is the input layer that represents the sentences over nxk dimension, second is convolutional layer, then global max pooling layer and finally fully connected layer producing output results as shown in Fig. 1.

Convolutional layer is the main building block of a CNN as most of the computations are done at this layer. It is a feature extraction layer which extracts the local features through the filters and generates feature map computed by convolution kernel function and then outputs it to pooling layer.

The inputs to CNN are sentences or documents represented as a matrix. Each row of the matrix corresponds to one token, typically a word, but it could be a character. That is, each row is vector that represents a word. Typically, these vectors are word embeddings (low-dimensional representations) like *word2vec* or *GloVe*, but they could also be one-hot vectors that index the word into a vocabulary. Convolutional neural networks are fantastic for visual recognition tasks. A number of popular computer vision networks such as VGG, LeNet, AlexNet, Inception, ResNet are available. Such neural networks are built generally for image recognition. These networks contain high number of neural networks that extract abstract features from the image. For this, these networks require huge memory and computation requirements while training and also require more computationally intensive network to produce more accuracy depending upon the layers used. For example, AlexNet is composed of five convolution layers and three fully connected layers. LeNet-5 consists of seven layers; GoogleNet (Inception) consists of 22-layer deep CNN; VGG 16 and VGG 19 consist of 16 and 19 weight layers, respectively. The training of these deep networks is challenging as these require a large memory and computation. However, in case of our proposed CNN model, the experiments have been performed with two or three convolution layers and analyzed that by increasing number of convolutional layers, there is not much improvement in accuracy but the training time increases because of increase in computations.

4 Implementation

4.1 Tools Used

We have used Jupyter Notebook (an open-source Web application) [20] as development environment for performing experiments. It helps in creating and sharing documents consisting of code, text, equations and visualizations. It also includes machine learning, data cleaning, transformation and statistical modeling, etc. The CNN models have been developed using Python package TFLearn (a deep learning library)

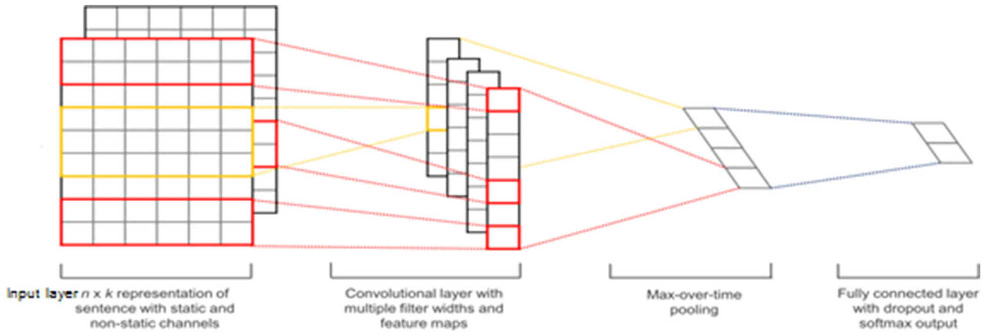

Fig. 1 Architecture of Convolutional Neural Network [15]

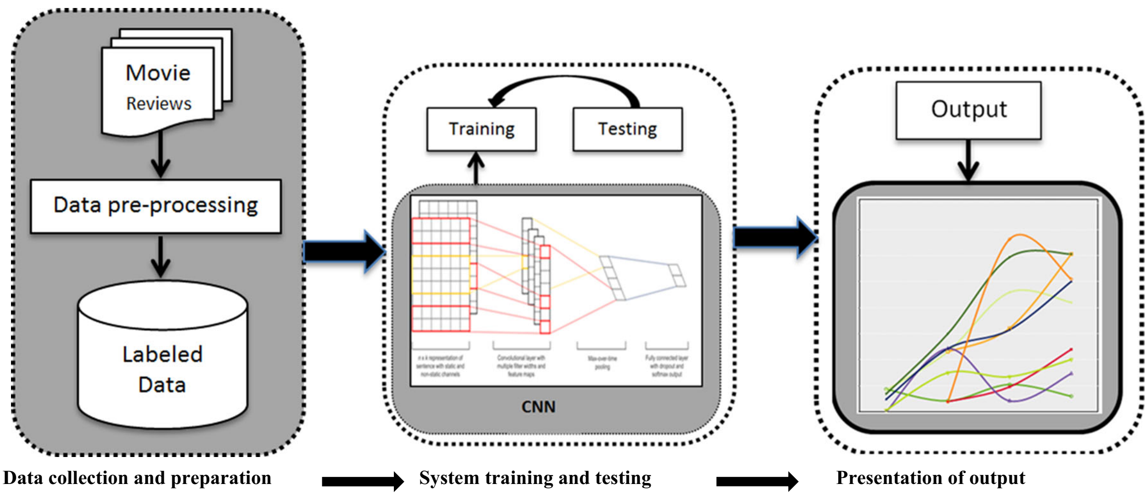

Fig. 2 Workflow of our system

[21] built on the top of the TensorFlow (an open-source software library for numerical computations) [22] that speeds up the development of deep learning models. All strings in the sentences are transformed to list of sequences using vocabulary processor of TFLearn as neural networks do not handle strings.

4.2 Working of Our System

The workflow of our proposed system is shown in Fig. 2, and it works through three phases such as data collection and preparation; system training and testing; and presentation of output.

The brief description about its phases is given as follows.

4.2.1 Data Collection and Preparation

In this phase, dataset of Hindi Movie reviews has been extracted from फिल्म-समीक्षा philm-sameeksha '*film-review*' section of *aajtak* (http://aajtak.intoday.in/film-review.html) and *jagran* (http://www.jagran.com/entertainment/reviews-news-hindi.html) online newspapers. The collected data are preprocessed by removing irrelevant data in order to generate a rich set of features. Transliteration of romanized text into Hindi text is performed using Google API. The punctuations, URLs, etc., have been removed using regular expressions of Python. To handle the issue of Hinglish words, a mapping dictionary of 250 frequently used movie reviews-related words has been prepared containing a mapping of Hinglish to Hindi words. This dataset has been manually annotated by three native speakers of Hindi into three classes such as positive, negative and neutral. This dataset consists of 7354 movie reviews which include 2341 positive, 2037 negative and 2976 neutral sentences. Some sample sentences of the dataset are given in Table 1.

4.2.2 System Training and Testing

The system is trained using CNN model which consists of four layers, i.e., input layer, convolution layer, global max pool layer and fully connected layer. These layers are briefly described as follows.

Table 1 Example sentences of dataset

Sr. no.	Some sample sentences of the dataset
1.	Negative Hindi Sentence: कहानी को ठीक से समेटा नहीं गया है । Transliteration: kahaanee ko theek se sameta nahin gaya hai. English Translation: The story has not been properly compiled.
2.	Positive Hindi Sentence: आमिर बशीर स्टेज के बेहतरीन कलाकार हैं । Transliteration: aamir basheer stej ke behatareen kalaakaar hain. English Translation: Amir is the best actor of Bashir Stage.
3.	Neutral Hindi Sentence: उन्हें रास्ते में सैमुअल जैक्सन मिलते हैं । Transliteration: unhen raaste mein saimual jaiksan milate hain. English Translation: They get Samuel Jackson on the way.

a) Input Layer

A neural network requires word embedding as an input to the CNN, i.e., a vector representation of each word or sentence. Reviews are considered as a sequence of words where each word is signified by a vector $v \in \mathbb{R}^{1\times d}$, known as word embedding. Here, d is the dimension and $d \ll |V|$, the vocabulary size V. In this case, *word2vec* tool has been used which is able to capture the semantic properties of words in the dataset. The model has been trained on 50% of the dataset, and this trained model is used for mapping a word onto its respective vector representation. The high-dimensional vectors are calculated for every word by calculating softmax probability for every word. The dimension of vector corresponds to the number of neurons in the hidden layer. The vector dimension of a word has been set to 100. Each sentence is padded with zero vectors in order to make its length uniform throughout the dataset. All the vectors are subsets of the word embedding matrix M consisting of all words in V. These words are mapped into indices $1 \ldots |V|$ to quickly look up the vector of the word in M. Then, for each review x, a sentence vector $X = \{w_1, w_{2,\ldots,} w_i, \ldots, w_{|x|}\}$ has been built and $X \in \mathbb{R}^{d\times|x|}$, where w_i represents the word embedding at the corresponding position i in a sentence. Then, X is fed to the convolutional neural network.

b) Convolution Layer

In this layer, a set of m filters are applied to a sliding window of length h over each sentence. These filters are applied to every possible window of words in the sentence, and a feature c_i is generated. Each filter has its own separate bias. These m filters working in parallel generate multiple feature maps.

c) Global Max Pooling Layer

The pooling layer samples the feature map generated by the convolution layer and the local optimum features. This layer aggregates the information and reduces the representation.

d) Fully Connected Layer

The fully connected layer computes the transformation using the equation given in (1) where α is the rectified linear unit (ReLU) activation function, $W \in \mathbb{R}^{m\times m}$ is the weight matrix, $b \in \mathbb{R}^m$ is the bias and C_{pool} is the feature map matrix generated by pooling layer.

$$x = \alpha(W * C_{\text{pool}} + b) \tag{1}$$

The output vector of this layer corresponds to the sentence embedding for each review. Finally, the output of the previous layer is passed to a fully connected softmax layer. It returns the class K with the largest probability. The '*categorical_crossentropy*' loss function has been used in softmax layer as there are three classes to measure the error probability between the network prediction and real output label. The softmax layer returns the classification result; then, the model parameters are updated by the back-propagation algorithm according to the actual classification label of the training data. Finally, each sentence gets three labels with values where one value represents the real label. For example, 'positive' = [0, 0, 1], 'negative' = [1, 0, 0] and 'neutral' = [0, 1, 0].

The above process is operated iteratively to train the model. The CNN models are trained on the 50% of the data and then tested on the remaining 50% of validation set to calculate the accuracy of analyzing sentiment. The different CNN models are built by experimenting with varying parameter settings such as number of convolution layers, filter size and number of filters.

4.2.3 Presentation of Output

The output results are presented by drawing line charts, and these charts help in performing the comparative analysis between different models of CNN. The different parameter settings for conducting experimentation are given in the next section.

5 Experiment Setup

In this work, different CNN models have been built using several parameters for each layer. The parameter settings used for CNN model such as vocabulary size, vector size, number

Table 2 Parameters settings of the proposed CNN

Parameter	Value
Vocabulary size	13, 398
Input vector size	100
Number of convolutional layers	2, 3
Number of hidden layers	6, 7
Activation function	ReLU
Number of filters	10, 50, 60, 100, 128, 256
Filter size	3, 4, 5, 7
Number of fully connected layers	1
Output dimension	128
Regularizer	L2
Dropout	0.5
Number of epochs	5
Batch size	64

of convolutional layers, hidden layers, fully connected layers, number of filters, filter size, activation function, regularizer, dropout, number of epochs and batch size are specified in Table 2. For the experimentation, the number of convolution layers has been taken either 2 or 3 and number of filters has been varied from 10 to 256. Also, the experiments have been conducted by varying filter sizes such as 3×3, 4×4, 5×5 and 7×7. The values of these parameters have been set by analyzing the studies conducted by other authors in this area [23]. The other parameters such as output dimension, regularizer, dropout, number of epochs and batch size have been fixed as change in these parameters has not shown any improvement in the accuracy of the model.

In all models, the number of convolutional layers has been varied along with other parameters such as number of filters and size of filters. The configuration settings of all the 12 CNN models are described in Table 3.

The results given by our model after experimentation with different parameter settings of CNN are discussed in the next section.

6 Results and Discussions

The validation accuracy and loss score of all CNN models are listed in Table 4 along with its training time (in seconds).

After performing several experiments with different parameters, it has been observed that CNN model with two convolution layers and filter sizes 3, 4 performs better and achieved an accuracy of 95.4%. In case of CNN model with three convolutional layers, maximum achieved accuracy is 93.44%. It has been analyzed that by increasing the number of convolutional layers and filters, the training time of the model is increased. Figure 3 shows the average validation accuracy and loss score of all CNN models. The X-axis specifies the number of training iterations, and Y-axis specifies the percentage of accuracy and loss score in Fig. 3a, b, respectively.

The average learning curve in Fig. 3a shows that there is gradual increase in the accuracy percentage with the increase in training; it means that models are learning from data. The loss score is the total number of errors that the model predicted. Figure 3b shows that at the start, there were a lot of errors and as the number of training steps increased, the errors decreased. As most of the models had a good nonlinear learning curve, dropping rapidly in the beginning and mostly have reached below 0.6.

The other performance parameters such as precision, recall, F-measure, Kappa score, mean absolute error (MAE) and root-mean-squared error (RMSE) for each of the CNN model are listed in Table 5.

Figure 4 represents the comparison of precision, recall and F-measure of all 12 CNN models. The X-axis of bar

Table 3 Parameters settings for CNN

MName	Convolution layers	Hidden layers	No. of filters	Filter size
CNN1	2	6	10	3.4
CNN2	2	7	10	3.5
CNN3	2	6	50	3.4
CNN4	2	7	50	3.5
CNN5	2	6	60	3.4
CNN6	2	7	60	3.5
CNN7	3	6	100	3.4.5
CNN8	3	7	100	7.4.3
CNN9	3	6	128	3.4.5
CNN10	3	7	128	7.4.3
CNN11	3	6	256	3.4.5
CNN12	3	7	256	7.4.3

Table 4 Accuracy and loss of CNN models

MName	Validation accuracy	Validation loss	Training time (s)
CNN1	0.926	0.221	16.73
CNN2	0.953	0.142	16.87
CNN3	0.954	0.155	24.02
CNN4	0.884	0.298	25.07
CNN5	0.938	0.165	25.53
CNN6	0.941	0.183	26.12
CNN7	0.737	0.673	46.4
CNN8	0.859	0.331	53.43
CNN9	0.92	0.235	61.22
CNN10	0.934	0.165	67.11
CNN11	0.927	0.235	166.84
CNN12	0.934	0.205	169.83

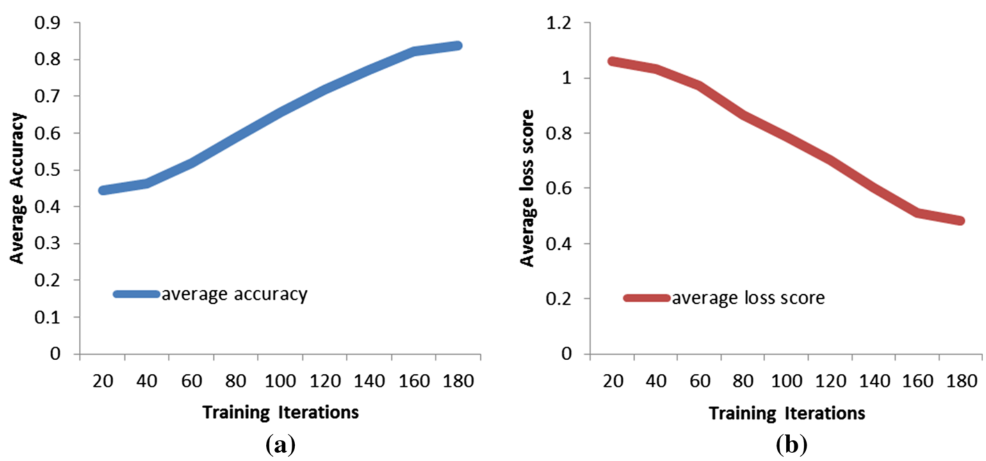

Fig. 3 Average learning curve of **a** accuracy, **b** loss score for all CNN models

Table 5 Other performance measures

MName	Precision	Recall	F-measure	Kappa	MAE	RMSE
CNN1	0.931	0.926	0.926	0.889	0.08	0.304
CNN2	0.954	0.953	0.953	0.93	0.05	0.239
CNN3	0.954	0.954	0.954	0.93	0.06	0.297
CNN4	0.885	0.884	0.883	0.826	0.117	0.347
CNN5	0.939	0.938	0.938	0.907	0.076	0.322
CNN6	0.942	0.941	0.941	0.911	0.065	0.277
CNN7	0.753	0.736	0.716	0.604	0.365	0.754
CNN8	0.863	0.859	0.855	0.788	0.143	0.383
CNN9	0.921	0.92	0.92	0.881	0.11	0.413
CNN10	0.934	0.934	0.934	0.901	0.073	0.298
CNN11	0.93	0.927	0.927	0.89	0.076	0.286
CNN12	0.935	0.934	0.934	0.901	0.068	0.265

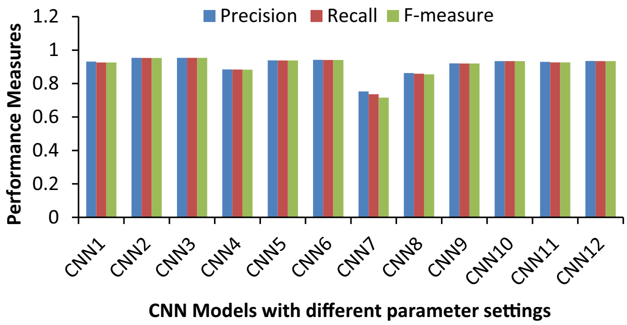

Fig. 4 Comparison of precision, recall and F-measure for all CNN models

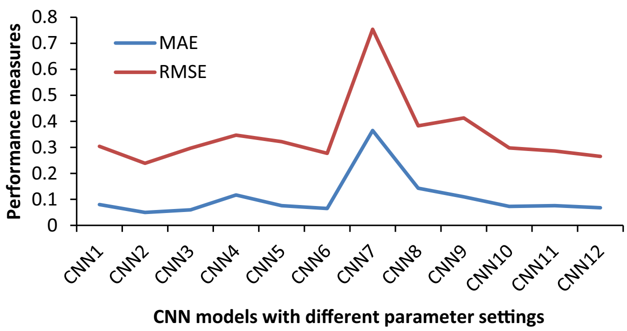

Fig. 5 Comparison of error rates of all CNN models

chart specifies the CNN models with different parameter settings, and Y-axis specifies the value of precision, recall and F-measure for each CNN model. Figure 5 compares the error rates in terms of MAE and RMSE given by CNN models.

From Figs. 4 and 5, it has been observed that CNN3 is the best model and CNN7 model performs worst out of all models as its accuracy is less than 80%. The confusion matrix of the best model CNN3 is given as follows.

	Confusion matrix		
	Predicted		
	A	B	C
Actual			
A = Positive	948	20	28
B = Negative	25	1505	24
C = Neutral	7	16	1104

Figure 6a, b represents the learning curve of accuracy and loss score for the best performing model CNN3.

From Fig. 6a, it has been observed that the accuracy of the model increases with the number of training iterations. But after 130 iterations, the accuracy gets stabled. Figure 6b represents that the learning curve of loss score for CNN model CNN3 drops rapidly and reaches below 0.2, which means that this model has least number of errors.

Similarly, Fig. 7a, b represents the learning curve of accuracy and loss score for the worst performing model CNN7.

6.1 Comparison

In this section, results given by our system are compared with baseline ML algorithms to analyze the improvement in accuracy. The baseline ML algorithms such as NB, k-nearest neighbor (k-NN), maximum entropy (ME) and support vector machines (SVM) have been applied on the same dataset. The experimentation with these ML algorithms has been performed using WEKA (an open-source machine learning software). After preprocessing, the dataset is prepared for further processing by converting it into '*.arff*' format and by setting its file encoding parameter to '*UTF-8*' in WEKA. The input is Hindi sentences with string data type to the WEKA. The string data are converted into numerical matrices by applying '*StringToWordVector*' filter using its properties such as *IDFTransform*, *TFTransform* and *tokenizer*. The system learns on the basis of bag-of-words unigram feature model using different classifiers such as NB, k-NN, ME and SVM. The system uses 50% of the dataset for training and 50% of the dataset for testing purpose. The accuracy achieved using ML approaches on the tested dataset is compared with our CNN model and is represented in Fig. 8.

After performing experiments with different parameters variations of CNN models, it has been analyzed that nine out of 12 CNN models are able to achieve an accuracy above than 90%. This accuracy is comparatively better than baseline ML algorithms. Out of baseline ML algorithms, maximum accuracy (i.e., 90%) is achieved by NB.

Also, the experiment has been performed on the proposed approach using HSWN. HSWN is a polarity lexicon consisting of sentiment words along with pre-computed polarity score. In this case, a word is represented using n-dimensional vector, that is $x \in \mathbb{R}^n$ where n is the feature number and the word vector is built using sentiment attribute feature of the word itself. For this, three word sentiment attribute features have been used, i.e., whether the word is positive, negative or neutral. The value of each dimension is represented by 1 or 0, where 1 indicates that feature is present and 0 indicates that the feature is absent. For a given sentence, it contains k words; then, the features matrix of the sentence is $X \in \mathbb{R}^{k \times n}$ and is used as input to the CNN. Since the HSWN consists of the general sentiment words and also the limited coverage of sentiment words, the results given by the proposed CNN model are not encouraging on the movie reviews dataset.

As discussed earlier, there exists a little work on SA of Hindi language due to non-availability of annotated corpora. And, majority of research work has been performed on SA for English and Chinese languages using CNN. Table 6 represents the comparative analysis of existing research works on

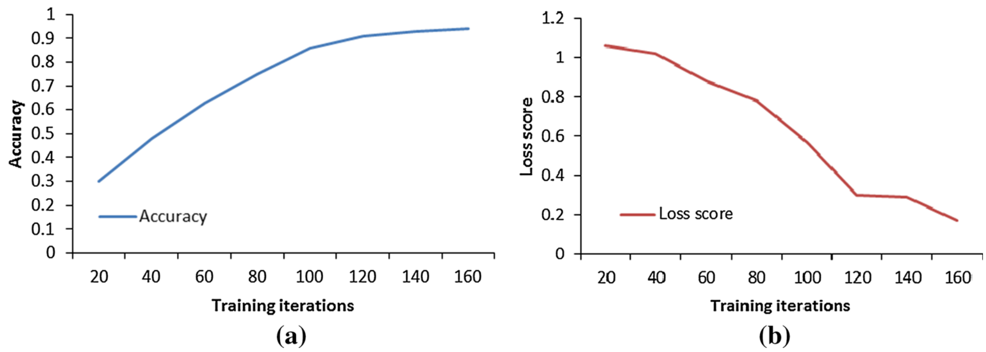

Fig. 6 Learning curve of **a** accuracy, **b** loss score for model CNN3

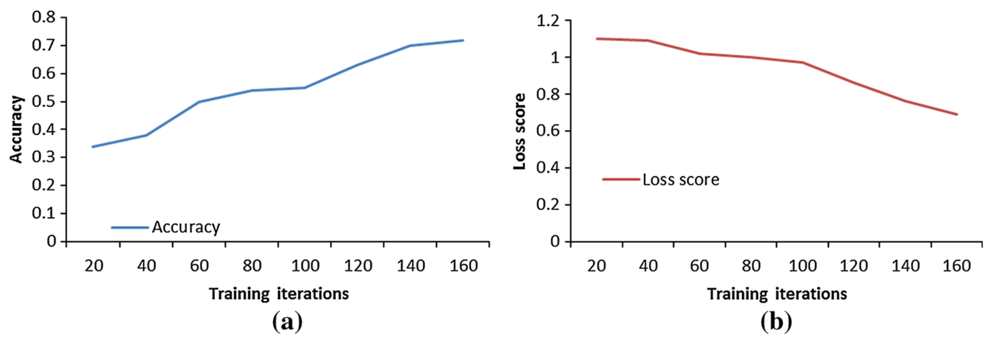

Fig. 7 Learning curve of **a** accuracy, **b** loss score for model CNN7

SA with proposed work on the basis of different parameters such as domain, languages and corpus size.

In order to perform the analysis on the basis of performance, the results given by the proposed approach have been compared with the existing works using CNN on the basis of accuracy measure. The accuracies shown in Fig. 9 for the existing works have been achieved by researchers on their own datasets, domains and languages. From the figure, it has been observed that our CNN model is able to achieve better results on the newly constructed dataset in comparison with the existing works on SA that used their own datasets.

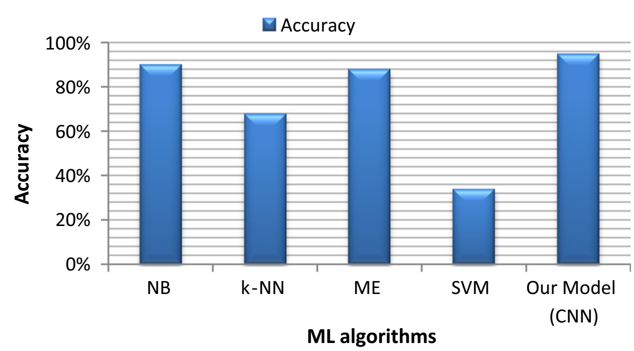

Fig. 8 Comparative analysis of accuracy with baseline ML algorithms with CNN

7 Conclusions and Future Scope

In this paper, we applied CNN to perform the SA of Hindi movie reviews. The experimental results suggest that properly trained CNNs can outperform the baseline ML algorithms for sentiment classification. In our model, the sentences of reviews are labeled into three classes such as positive, negative and neutral. All the experiments are performed using different parameter settings for all CNN models, and it has been observed that CNN model having two convolutional layers with filter sizes 3 and 4 performs the best all over models and is able to achieve an accuracy of 95%.

Future work includes experimentation with other deep learning models along with extension of the dataset for other domains such as products, restaurant reviews, social media analysis and political analysis to build the system in general so that it can be used in other applications too for the benefits of the society.

Table 6 Comparative analysis of existing works with proposed CNN model

Author	Domain	Language	Corpus size
Shirani-Mehr [12]	Movie reviews	English	11,855
Ouyang et al. [15]	Movie reviews	English	10,662
Zhang and Chen [17]	Microblogs	Chinese	5000
Our work (2017)	Movie reviews	Hindi	7354

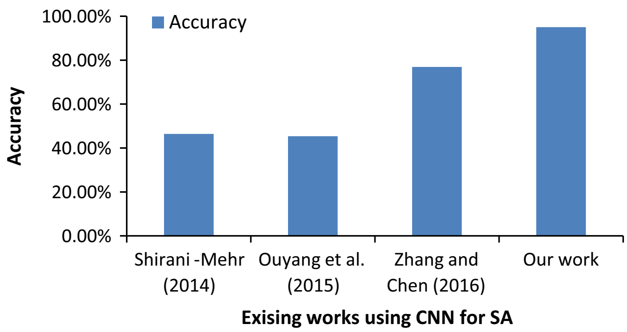

Fig. 9 Comparison of results of our model with existing works

Acknowledgements This Publication is an outcome of the R&D work undertaken in the project under the Visvesvaraya PhD Scheme of Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

1. Almatrafi, O.; Parack, S.; Chavan, B.: Application of location-based sentiment analysis using Twitter for identifying trends towards Indian general elections 2014. In: Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication. ACM, Bali, Indonesia, pp. 1–5 (2015)
2. Rani, S.; Kumar, P.: A sentiment analysis system to improve teaching and learning. Computer **50**(5), 36–43 (2017)
3. List of languages by number of native speakers in India [Online]. https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers_in_India. Accessed 4 June 2017
4. Joshi, A.; Balamurali, A.R.; Bhattacharyya, P.: A fall-back strategy for sentiment analysis in Hindi: a case study. In: Proceedings of the 8th International Conference on Natural Language Processing, pp. 1–6 (2010)
5. Mittal, N.; Agarwal, B.; Chouhan, G.; Bania, N.; Pareek, P.: Sentiment analysis of Hindi review based on negation and discourse relation. In: proceedings of International Joint Conference on Natural Language Processing, pp. 45–50 (2013)
6. Arora, P.: Sentiment Analysis for Hindi Language. MS Thesis. International Institute of Information Technology, Hyderabad, India (2013)
7. Sharma, R.; Nigam, S.; Jain, R.: Polarity detection movie reviews in Hindi language. arXiv preprint arXiv:1409.3942 (2014)
8. Sharma, R.; Bhattacharyya, P.: A sentiment analyzer for Hindi Using Hindi Senti Lexicon. In: 11th International Conference on Natural Language Processing, pp. 1–6 (2014)
9. Pandey, P.; Govilkar, S.: A framework for sentiment analysis in Hindi using HSWN. Int. J. Comput. Appl. **119**(19), 23–26 (2015)
10. Sharma, Y.; Mangat, V.; Kaur, M.: A practical approach to sentiment analysis of Hindi tweets. In: Proceedings of 1st International Conference on Next Generation Computing Technologies, pp. 677–680 (2015)
11. Akhtar, M.S.; Ekbal, A.; Bhattacharyya, P.: Aspect based sentiment analysis in Hindi: resource creation and evaluation. In: International Conference on Language Resources and Evaluation, pp. 1–7 (2016)
12. Shirani-Mehr, H.: Applications of deep learning to sentiment analysis of movie reviews. Technical report, Stanford University (2014)
13. Dos Santos, C.N.; Gatti, M.: Deep Convolutional Neural Networks for sentiment analysis of short texts. In: Proceedings of 25th International Conference on Computational Linguistics, pp. 69–78 (2014)
14. Stojanovski, D.; Strezoski, G.; Madjarov, G.; Dimitrovski, I.: Twitter sentiment analysis using Deep Convolutional Neural Network. In: International Conference on Hybrid Artificial Intelligence Systems, pp. 726–737 (2015)
15. Ouyang, X.; Zhou, P.; Li, C.H.; Liu, L.: Sentiment analysis using Convolutional Neural Network. In: International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), pp. 2359–2364 (2015)
16. Singhal, P.; Bhattacharyya, P.: Sentiment Analysis and Deep Learning: A Survey [Online]. http://www.cfilt.iitb.ac.in/resources/surveys/sentiment-deeplearning-2016-prerna.pdf (2016)
17. Zhang, L.; Chen, C.: Sentiment classification with Convolutional Neural Networks: an experimental study on a large-scale Chinese conversation corpus. In: Proceedings of 12th International Conference on Computational Intelligence and Security (CIS), pp. 165–169 (2016)
18. Hassan, A.; Mahmood, A.: Deep learning approach for sentiment analysis of short texts. In: 3rd International Conference on Control, Automation and Robotics (ICCAR), pp. 705–710 (2017)
19. Zhang, Y.; Chen, M.; Liu, L.; Wang, Y.: An effective Convolutional Neural Network model for Chinese sentiment analysis. In: AIP Conference Proceedings, vol. 1836, number 1, pp. 1–6 (2017)
20. Jupyter [Online]. http://jupyter.org/. Accessed 2 June 2017
21. TFLearn: Deep learning library featuring a higher-level API for TensorFlow [Online]. http://tflearn.org/. Accessed 5 June 2017
22. Tensorflow [Online]. https://www.tensorflow.org/. Accessed 3 June 2017
23. Svensson, K.: Sentiment analysis with convolutional neural networks: classifying sentiment in Swedish reviews. Bachelor Dissertation, Linnaeus University, Sweden (2017)