Journal of Statistics & Management Systems
ISSN 0972-0510 (Print), ISSN 2169-0014 (Online)
Vol. 27 (2024), No. 1, pp. 59–86
DOI : 10.47974/JSMS-1030

Deep learning based bi-polar sentiment classification of movie reviews in Hindi

Ankita Sharma *
Udayan Ghose †
University School of Information, Communication and Technology
Guru Gobind Singh Indraprastha University
Sector – 16C, Dwarka
New Delhi 110078
India

Abstract

With evolving technology, Hindi web content is growing considerably and catching popularity as the larger audience feels more connected and heard using their native language. A volcanic growth in online movie reviews (MRs) in Hindi has been observed lately; manually analyzing them is impossible. Hence, the research problem of automatic organization and classification of Hindi reviews is apparent as this can help viewers decide whether a movie is worth watching or not. This work focuses on developing a deep learning-based system for bi-polar sentiment classification of MRs for resource deficient language – Hindi. To this end, a primary Hindi movie review (MR) corpus is made and manually annotated with binary polarity class labels - positive or negative. The corpus is preprocessed using the preprocessing steps, and Random Word Embeddings (WEs) are utilized for feature extraction. This paper proposes an ensemble CNN_BiGRU, which is an integration of 1D CNN with BiGRU for the bipolar classification of Hindi MRs. To prove our ensemble's efficacy; other widely used mainstream deep learning models (DLMs) such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) based models – Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), are also applied and compared with proposed ensemble using average classification accuracy. Empirical results show the effectiveness of the proposed ensemble in achieving a reasonably good average accuracy of 89.366%. The results indicate that proposed CNN_BiGRU compares favorably to the state-of-the-art DLMs applied and hence gives an effective solution for sentence-level bi-polar classification of MRs in a resource deficient scenario.

Subject Classification: *Primary 68T50, Secondary 97R40.*

Keywords: *Deep learning, Ensemble, Hindi, Sentiment classification, Word embeddings.*

*E-mail: ankitasharma2711@gmail.com (Corresponding Author)
† E-mail: udayan@ipu.ac.in

1. Introduction

Text data is one of the primary forms of data available on the web. It's reliability and expression comprehensibility make it prominent. Therefore, text classification or categorization is of high social and economic value [1]. With the development of the Unicode (UTF-8) standard, there has been a plethora of Hindi textual content on the web. Therefore, there is a pressing need to understand and make sense of online Hindi data. Hence, the research problem of automatic organization and classification of Hindi data is apparent.

Decision-making is part and parcel of everyone's life. Our decisions are intrinsically susceptible and influenced by other people's decisions, thinking, and experiences [2]. It is known that watching a movie is the most favorite activity around the globe. Movie watchers watch a movie for different reasons; some watch for entertainment, some watch for stress management, and others watch movies to kill their time. But one thing common in all scenarios is that no one wants to waste their time and money on a movie that is not worth watching. Reading online movie reviews (MRs) can provide rescue to this problem. Reading reviews can help viewers to get an idea about the movie, like its storyline, positive and negative aspects, etc. By aggregating the crowd's wisdom, people can decide between watching a movie or not, and this can help the viewers save their time and money.

With evolving technology such as Hindi supporting keyboards and platforms, Hindi web content is growing considerably and catching popularity as the larger audience feels more connected and heard using their native language. A volcanic growth in online MRs in Hindi has been observed lately; manually analyzing them is impossible. In the present era, mining the emotional inclination of reviews is crucial for understanding the public emotion and sentiment trend regarding a movie. It can significantly help viewers or reviewers decide whether a movie is worth watching or not. Therefore, the research problem of the bi-polar sentiment classification (SC) of MRs data becomes evident. Bi-polar SC or binary sentiment analysis (SA), or movie review text orientation analysis categorizes written Hindi MRs into opposing pre-defined classes, i.e., positive, or negative. Manually going through all reviews is tiresome and far from human power [3]; it is a labor-intensive and expensive task.

SA or SC of Hindi textual data is relatively unexplored owing to limited resources available to deal with Hindi text like the gold standard annotated datasets with adequate quality and quantity, linguistic resources

like lexicons, parsers, part-of-speech taggers, etc. [4]. Conventionally SA was performed using Machine learning (ML), Lexicon-based, and hybrid [5]. But nowadays, deep learning (DL) for text SC is gaining popularity [6]. Lately, it has been cognized that DLMs have achieved remarkable outcomes for text SC tasks [7][8]. The most widely used DLMs for text classification tasks are LSTM, GRU, and CNN [9]. DL and ensemble learning are the remarkable words for the computer science community. There has been a notable amount of work done in resource-rich languages. While work done in Hindi is still emerging.

In this work, we focus on developing a DL based bi-polar SC system of MRs of a resource deficient language – Hindi. DLMs were chosen for the bi-polar classification of Hindi MRs because deep models by themselves focus on the right features with minute direction from the programmer. To this end, a primary Hindi review corpus comprising 6000 pure Hindi MRs is made, details of which is given in section 4. The corpus is preprocessed using various pre-processing steps and Random WEs is utilized for feature extraction. It is known that DLMs are increasingly developing in a mature direction for NLP tasks [8][10]. The textual SC using DLMs is becoming increasingly popular. Thereby, in this work we proposed an ensemble, which is an integration of CNN and BiGRU models for the bi-polar classification of Hindi MRs. To further prove our ensemble's efficacy; other widely used mainstream DLMs like CNN, and RNN-based models such as LSTM, and GRU are also applied and compared with our ensemble. The performance of the proposed ensemble and other DLMs that were applied were evaluated and analyzed using average classification accuracy.

The roadmap of the paper is as follows: After Introduction in Section 1, Section 2 deals with related work done in DL for Hindi. Section 3 presents the challenges while dealing with Hindi. Section 4 focus on movie review dataset used in this work. Section 5 describes the proposed methodology. Section 6 discusses the results obtained. In Section 7, we have concluded our work and stated future directions.

2. Literature Review

Bipolar SC in a resource-deficient language like Hindi is still in its inception phase. Few pieces of research have been conducted, some of which are stated below:

Authors in [11] attempt to perform SA on a tweet dataset provided by SAIL 2015 task organizers in Hindi, Tamil, and Bengali. Dataset consists of a total of 4,835 tweets. RNN was employed for SA task. The Tamil, Hindi,

and Bengali datasets have obtained 88.23%, 72.01%, and 65.16% accuracy respectively. Data cleaning, SentiWordNet to be employed for future work.

In [12], authors used the CNN approach for offensive tweet detection. A primary dataset named HEOT, consisting of 3,679 tweets was made. For classification, Transfer learning has been used. Accuracy of 83.9% was obtained and Gradient boosting to be employed for fine-tuning in the future.

An attempt has been made in [13] for Hate speech detection using the code-mixed social media text by employing the Hierarchical and sub-word level LSTM model having phonemic sub-words-based attention. Results have been compared with Random Forest and SVM classifier. Although SVM obtained the highest accuracy, Hierarchical LSTM got the highest Recall and F1 score. The future work would capture more semantic information.

Authors in [14], attempt to perform SA is made on a Hindi-English code-mixed dataset using the sub-word LSTM. The primary dataset consists of tweets related to actor Salman Khan and PM Narendra Modi. The dataset has been manually annotated. Results indicated that the proposed performed better than the baselines. The sub-word-LSTM was used to obtain the highest accuracy of 69.7%. Future work is to investigate the scaling effect of RNN.

The study points in [15] were to borrow WEs and word polarities from English for binary multi-lingual SC. Hindi movie and tourism reviews were considered. Results implied that this borrowing approach outperformed baselines and achieved the highest accuracy of 80.2% in Hindi movies and 88.9% in tourism reviews. Multiclass is targeted for future work.

The point of study in [16] was to propose a hybrid of CNN and LSTM, to perform binary SA on Hindi, Bengali, and Tamil languages. A dataset of 2061 tweets was used. The Hindi dataset achieved 74.43% accuracy. Future work is to perform multiclass SA and emoticons handling.

Authors in [17] have proposed an ensemble of LSTM and Multinomial Naïve Bayes (MNB) for the Hindi-English mixed code SA. The highest accuracy of 70.8% is obtained with the ensemble. It has been observed that the MNB is suitable for analyzing rare keywords and slang, whereas the LSTM works well for longer sentences. Future work will extend the dataset and their work to other language pairs.

The work in this paper differs from the above-stated works as follows: most of the works have used code mixed datasets or translation datasets. Also, used pre-trained embeddings or embeddings from resource affluent

languages. However, none of the researchers have considered pure Hindi MRs corpus and used random word embeddings (WEs), and our dataset size is comparatively large compared to most works mentioned above. Whenever applying DLMs choosing the right hyperparameters is vital; in this work, this aspect has been considered for the binary classification of MRs in Hindi. Along with hyperparameters, 10-fold cross-validation was applied and average accuracy was used to evaluate the performance of the applied DLMs and proposed CNN_BiGRU.

3. Challenges while working with HINDI

Following are some complexities of the Hindi language, making Hindi SC a challenging task.

- Hindi is a resource-poor language. Therefore, it lacks the standard labeled dataset required for the supervised classification task. Getting the appropriate amount and quality of the Hindi dataset is challenging.
- There exists no capitalization in Hindi text in contrast to English, where the first letter of the word is usually capital suggesting the beginning of a sentence. Thus, identifying the start of a Hindi sentence, while reviewing it becomes challenging.
- People usually mix the English language while writing Hindi reviews. For instance: गैंग्स ऑफ वासेपुर का बैकग्राउंड म्युजिक, म्युजिक, सिनेमाटोग्राफी, एडिटिंग टॉप क्लास है। {The background music of Gangs of Wasseypur, music, cinematography, the editing is top class}. Getting pure Hindi corpora is difficult.
- Sarcasm Detection is tricky while dealing with Hindi reviews. For instance: इश्क इन पेरिस की सबसे अच्छी बात ये है कि फिल्म 96 मिनट में ही खत्म हो जाती है। {The best thing about Ishq in Paris is that the film ends in 96 minutes}.
- Some reviews in Hindi do not contain any sentiment words but still express sentiments. E.g., क्रीचर में ऐसा कुछ भी नहीं है जिसके लिए टिकट खरीदा जाए। {There is nothing in Creature for which to buy a ticket}.
- Negation handling is complex while dealing with Hindi review data. E.g., कुल मिलाकर 'तेरा सुरूर' का सुरूर नहीं चढ़ता। {Overall, 'Tera Suroor' does not start well}.
- In Hindi, exact words can have different spellings; therefore, including all variations of spellings in lexical resources or while

training classifiers becomes tedious. Hindi word ठंडा {ThaNDa} meaning cold or chill can also be written as ठण्डा.

- Hindi is of free order meaning thereby, the word order does not affect its meaning.

4. Dataset Description

This section describes the dataset used in this work.

4.1 *Description of Corpus*

Due to limited resources available to deal with Hindi, there have been accelerated linguistic challenges in coping with Hindi. Getting adequate Hindi corpora is a perpetual problem when working with Hindi. Movie review (MR) in Hindi are generally continuous Hindi sentences. This paper's Hindi review corpus is primary and collected from multiple MRs online websites[1,2]. These 6000 reviews were manually labeled as positive or negative. Distribution of sentiment labels is shown in Figure 1. Later, each review was manually checked by two native speakers to confirm the sentiment polarity label. The quality of annotation was measured with Cohen's Kappa which is provided by scikit-learn. The Cohen's Kappa came out to be ~84%, which is a standard agreement parameter. The corpus used in this work is a CSV file with two columns: review text and sentiment label. The sentiment label column has two values: 0 indicating negative and 1 indicating positive. A snapshot of our Hindi MR corpus is shown in Table 1 and summary of dataset used is given in Table 2. CV in Table 2, refers to cross-validation. Our Hindi MR corpus does not have a standard train-test split. Therefore, we have used 10-fold CV.

Table 1
Hindi MR Corpus

Review Text	Sentiment Label
अक्षय खन्ना इत्तेफाक फिल्म में चमकते हैं।	1
जिया और जिया फिल्म की कहानी बेहद कच्ची है।	0
भूमि फिल्म एक गंभीर मुद्दे का मज़ाक बना कर रख देती है।	0
फिल्लौरी फिल्म के धीमे संवाद फिल्म की कमजोड़ कड़ी हैं।	0

Contd...

[1] https://navbharattimes.indiatimes.com/movie-masti/movie-review
[2] https://www.aajtak.in/entertainment/film-review

चिर–परिचित फिल्म होने के बावजूद बद्रीनाथ की दुल्हनिया में ताजगी महसूस होती है।	1
जॉली एलएलबी 2 ऐसी फिल्म है जो पहले से आखिरी सीन तक दर्शकों का मनोरंजन करती है।	1

Table 2

Summary of Dataset used.

Domain	Classes	Corpus Size	Avg. Review length	Test Size
Movie	2	93663	15.61	10 CV

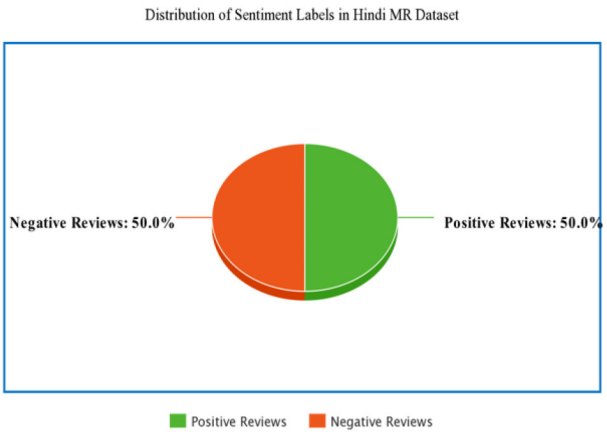

Figure 1

Statistics of Dataset Used.

5. Proposed Methodology

India has a diverse, rich culture and a multilingual society. With the advent of technology, Hindi content on the web is growing drastically in the form of tweets, blogs, movie reviews, product reviews, etc. [18][19] [30].

Reading online MRs in Hindi is an apt method for understanding emotions and opinions towards a particular movie [20]. Bi-polar classification of MRs in Hindi refers to classifying the Hindi reviews into positive or negative polarity labels, which in turn provide us information regarding the sentiment present in them, that is, the viewers or reviewer's emotional color and tendency.

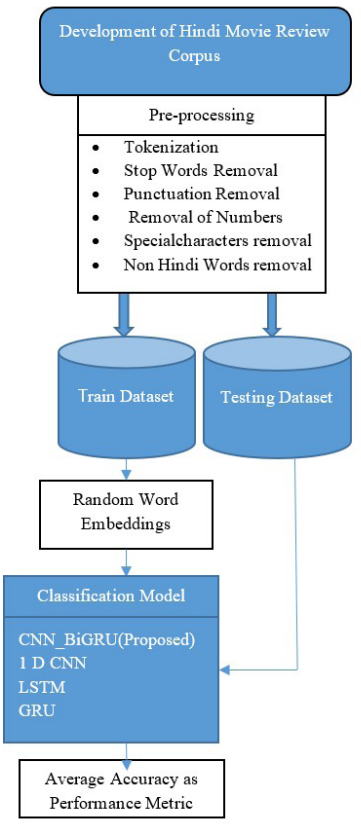

Figure 2

Block diagram depicting the bi-polar classification of Hindi MRs.

Figure 2. Presents the block diagram for bipolar classification of Hindi reviews. It consists of three major modules, pre-processing, feature extraction using Random word embedding, DLMs, and a proposed ensemble for bipolar categorizing movie reviews into positive or negative. Firstly, the Hindi MRs corpus was developed using reviews from online sources. Secondly, the Hindi MRs were passed through the pre-processing unit as the reviews were collected from online sources containing irrelevant information that needed cleaning. Afterward, random word embeddings are used for feature extraction. Then an ensemble model CNN_BiGRU is proposed and applied to the Hindi MR corpus for doing SC on Hindi MRs. Also, some state-of-the-art DLMs such as 1D CNN, LSTM, and GRU are applied and compared with the proposed ensemble. At last, the

performance of all the applied DLMs is evaluated using average accuracy. The details of each step are given beneath.

5.1 *Pre-processing of Hindi Review corpus*

Pre-processing of the corpus is the essential part of any problem statement [21]. Removing the irrelevant information from text is required, resulting in better classification performance. The pre-processing of MRs has three stages. Firstly, tokenization of Hindi reviews is done. Tokenizer class of Tensor Flow was employed for the same. Secondly, Numbers, special characters, non-Hindi words, and punctuation removal are performed. Thirdly, Hindi stop words are removed through the "spacy.lang.hi" library. The output corpus after pre-processing is shown in Figure 3.

	Review Text
0	अक्षय खन्ना इत्तेफाक फिल्म चमकते
1	जिया जिया फिल्म कहानी बेहद कच्ची
2	भूमि फिल्म एक गंभीर मुद्दे मज़ाक बना कर रख देती
3	फिल्लौरी फिल्म धीमे संवाद फिल्म कमजोड़ कड़ी
4	चिर-परिचित फिल्म होने बावजूद बद्रीनाथ दुल्हनि...
5	जॉली एलएलबी ऐसी फिल्म पहले आखिरी सीन तक दर...

Figure 3

Preprocessed Corpus.

5.2 *Word Embeddings (WEs)*

Interpreting textual data for humans is easy, but understanding and analyzing humongous textual data becomes tedious for machines. It is known that textual Hindi MRs cannot be fed directly into DLMs; therefore, it is required to convert the review data into numeric form. Random WEs were used for feature extraction as mentioned in [22] and conventional Term frequency- Inverse document frequency (Tf-idf) was not used. In Tf-idf, rare words are given more importance than commonly occurring words. While in MRs, sentiment words are commonly occurring words like "अच्छी" {good}, "मनोरंजक" {amusing}, "उबाऊ" {boring}, "रोमांचक" {thriller}, "बेहतरीन" {best}, "गजब" {Amazing}, "कमज़ोर" {weak}, "जोरदार" {vigorous}, "बोझिल" {cumbersome} etc. frequently appear and are assigned lower weights by Tf-idf; Also, a relation among words is not preserved

when employing Tf-idf. All these problems were catered using random WEs. WEs are the numerical representation of movie review vectors that map every word in the review corpus to a vector space. Based on the usage of words in reviews, syntactic, semantic, and contextual information is captured, which was missing in Tf-idf. In classical ML, feature extraction takes place in a supervised manner, while embeddings are obtained unsupervised [23]. For Indian languages, Wikipedia dumps are used for pre-trained model's generation, which is insufficient. In this work, random WEs mentioned in [22] was applied instead of using pre-trained models. In random WEs, all the word vectors are initialized randomly and improved while training.

5.3 *A brief description of DLMs employed for the bi-polar classification of Hindi MRs is given:*

These days the most widely used DLMs are RNN and CNN in the field of NLP [8][10]. Both works well for classification and regression problems. CNN performs well for image data, while RNN is suitable for textual and speech data [29][32]. This work applies both the mainstream DLMs to the Hindi MRs corpus.

5.3.1 Convolutional Neural Networks (CNN)

CNN, known as ConvNet, is a type of Feed-Forward Neural Network (FFNN) consisting of multiple convolutions, and convolution operations are learned at each layer. In ConvNet, a neuron at each layer have been connected to only a small portion of the layer before it., rather than being connected in a fully connected manner resulting in a smaller number of weights to be handled. CNN mainly consists of layers segregated for two functions: convolution, Rectified Linear Unit (ReLU), pooling mainly used for feature learning, while fully-connected (FC) layer is used for classification purposes.

1D CNN is deployed in this work for the binary classification of Hindi MRs as shown in Figure 4. In this work, each input in the input layer is a textual MR in Hindi. However, this input is a review after tokenization and pre-processing. Also, the input is the word vector of each word in the review. Suppose that the review is pre-processed into n words, each word is converted into a vector through random WEs; which is mapped into an m-dimensional vector, and the sequence of words in review is entwined and mapped into n x m dimensional matrix [10][22].

$$\begin{bmatrix} W11 & W12\cdots & W1n \\ \vdots & ..... & \vdots \\ Wn1 & Wn2\cdots & Wnm \end{bmatrix} \quad (1)$$

The convolutional layer's job is extracting features from input review; it performs the convolution calculation on input review using the convolution kernel to produce feature map.

Assuming that the input review r is composed of n word vectors W_1, W_2, …….W_n, the operation of the convolution layer can be expressed as [10] :

$$Ev = f(Wgt.W_{i:1-1} + bi) \quad (2)$$

Where $W_{i:i+h-1}$ refers to combination of vector W_i, W_{i+1}, …….$W_{i+h-1,}$ l refers to dimension of convolution kernel, f(*) is non-linear function, in this work ReLU is considered. Wgt refers to weight matrix and bi refers to bias.

The convolution is linear; to introduce non-linearity, ReLU is employed. Non-linearity comes through an activation function. ReLU is the most popular activation function that brings non-linearity. Input is the weighted sum to the activation function, generating the output.

The eigenvector ev obtained after, convolution kernel extraction is given below

$$ev = \{ev_1, ev_2, ev_3 \ldots\ldots\ldots\ldots ev_{n-1+1}\} \quad (3)$$

After the convolution operation, next, it goes to the pooling layer, which reduces the dimensions of the formed feature matrix. Dominant features are extracted from the constricted neighborhood. A pooling layer is added to reduce the dimensional complexity, still keeping sufficient information for convolutions. When Maximum pooling is used, and the maximum value amongst all the values in the pooling region are considered. The resultant matrix has lesser dimensions and main features.

$$ev = \max(ev_i). \quad (4)$$

Lastly comes the FC layer; as the name suggests, every node in each layer is connected to every other node in other layers. The FC layer is used to flatten the results. In the FC layer, classification, for instance, of MRs into positive or negative polarity classes is done. The output obtained from the previous layer is fed to FFNN, and in every training, backpropagation is applied. While training, weights and bias vary every time we feed them,

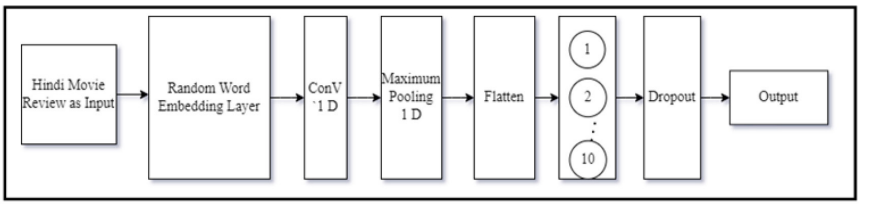

Figure 4
1D CNN model

multiple iterations occur, and every single iteration is called an epoch. After a few epochs, CNN can differentiate between the dominant and low-level features.

5.3.2 Recurrent Neural Network (RNN)

RNN is a generalization of FFNN, and they are good at handling variable-length sequence data. They are recurrent as they carry out an identical task for each sequence data element, with output dependent on prior computation; produced output is copied and returned to the recurrent network. For decision making, present input and output from previously learned input are considered and it is very vital for NLP tasks [27]. They make use of memory to process the sequences of data. It assumes that all inputs are interrelated. RNNs were the first of their kind as they operate over the sequence of vectors and contain memory. RNN makes use of prior information sequence to produce present resultant information.

In simple words, RNN contains two inputs, one is present input, and the other is the recent past. It beautifully captures the connection among words in the text. It is relevant as sequential data has vital information about upcoming data. Hidden layers play a significant role in RNN as it remembers some sequence details. Sequence-based learning is dealt with RNN [32]. They are good at relating previous information to present information., but as the sequence grows, RNN becomes inefficient in connecting previous information to present one and inefficient in discriminating among significant and insignificant information. This issue is known as long-term dependencies, and the cause is the vanishing gradient problem [1]. The most widely used solution to this problem is special kinds of RNNs, namely LSTM and GRU.

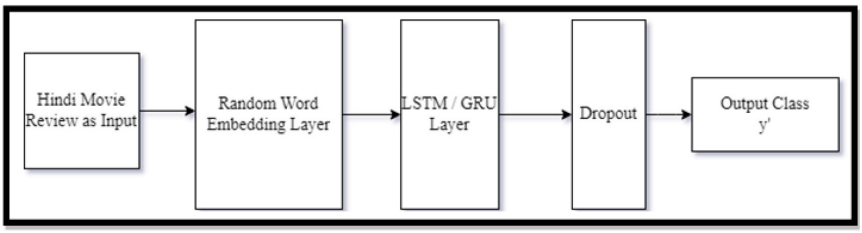

Figure 5

LSTM/GRU model

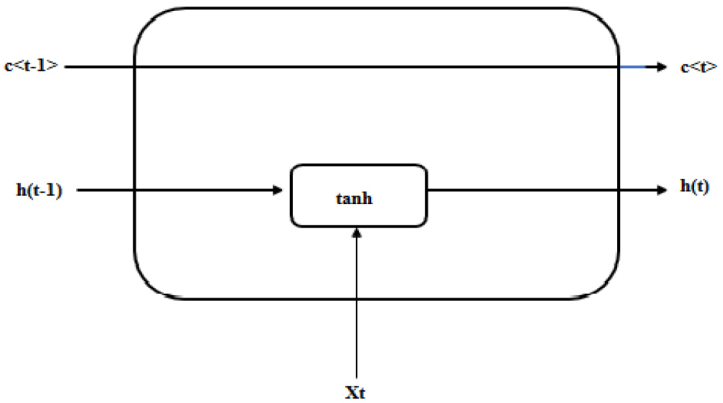

Figure 6

Basic LSTM block

5.3.2.1 Long Short-Term Memory (LSTM)

Conventionally, processing used to take place independently when dealing with textual data. Due to the unavailability of past information, there was no relation between the past and present information. It is known that CNN efficiently captures the local contextual information while failing to capture long-term dependencies among words [29]. However, LSTM solves this problem using a memory cell to capture long-term dependencies among words. LSTM is known to remember information for a longer time and gives accurate results for longer sequences [26]. They are mainly employed in SA and text classification tasks [25]. The basic LSTM model and box is shown in Figure 5 and Figure 6 respectively.

Input is MRs in Hindi that goes into the WEs layer. Next comes the LSTM layer, which contains LSTM units, consisting of input, output, and forget gates. The input gate, also known as the update gate, does critical

information analysis. The Update gate decides how much vital information will be added to the current state. The sigmoid activation function $\sigma(*)$ acts as a filter and decides determines what information is to be kept in the memory cell.

The input gate (i) calculation simplifies to:

$$i = \sigma(W`Xt + W``h\langle t-1\rangle + W```c\langle t-1\rangle + bi) \quad (5)$$

Where W`is connecting weight among Xt and i. W`` is connecting weight among h<t-1> and i. W``` is connecting weight among c<t-1> & i, and bi is bias term.

The forgot gate decides how much of the past information is to be remembered. Also, it decides how much information is forgotten from the memory cell at a particular time. The previous and present state is examined, and a value between 0 and 1 is given in the memory cell state. Zero signifies that information is irrelevant hence to forget or remove. While one states that the information is relevant and needs to be stored.

The forget gate (fo) calculation simplifies to [26][32]:

$$fo = \sigma(Wg`Xt + W``gh\langle t-1\rangle + Wg```c\langle t-1\rangle + bi) \quad (6)$$

x(t) and h(t-1) are inputs of LSTM unit.Wg` is connecting weight between Xt and fo. Wg`` is connecting weight between h<t-1> and ho. Wg``` is connecting weight among c<t-1> and fo, bi refers to bias.

The output gate selects essential information using present input and previous output and decides what information to be shown as output.

The calculation method for output gate Ot is as follows:

$$Ot = \sigma(W`oXt + W``oh(t-1) + W```oc\langle t-1\rangle + bi) \quad (7)$$

W`o is connecting weight of Xt and Ot. W``o is connecting weight between h<t-1> and Ot. Bias is bi and W```o is connecting weight among c<t-1> and Ot.

The drop-out layer is added at the end to increase model generalization and decelerate the overfitting. LSTM's gates help deal with vanishing gradient and exploding problems of RNN. The gating mechanism in LSTM controls the information flow, hence giving controlled exposure to the stored contents.

5.3.2.2 Gated Recurrent Unit (GRU)

GRU is a successor and a lightweight variant of LSTM with few parameters, faster convergence, and better learning performance [8][10]. It is generally employed for faster operations over smaller datasets [25].

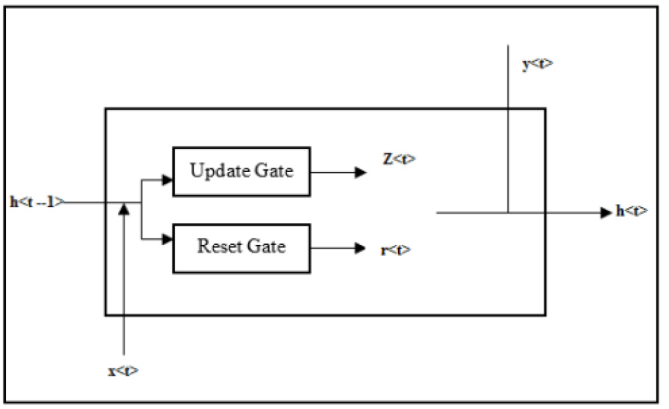

Figure 7

The Basic GRU box

GRU comprises two gates, namely the update gate and resets gate [26]. The Update gate determines how much past information is retained in the present step. GRU combines input and the forget gate into a single gate called the update gate and merges the hidden and cell state at the same time. While the reset state determines what information to remove before going to the next step, meaning how much past information to forget. The larger the value of updating the gate, the greater the degree of influence. Similarly, the larger the value of the resetting gate, the lesser information is ignored [26]. The good thing about GRU and LSTM is that they keep existing content intact and add new content [27]. The basic GRU box is shown in Figure 7.

As shown above, x<t> is the current state, h<t-1> is hidden state, z<t> and r<t> refers to update and the reset gate respectively. Z<t> and r<t> jointly control the calculation from h<t-1> to h<t>.

$$Z\langle t\rangle = \sigma(Wz.[h\langle t-1\rangle, x\langle t\rangle]) \tag{8}$$

$$r\langle t\rangle = \sigma(Wr.[h\langle t-1\rangle, x\langle t\rangle]) \tag{9}$$

σ refers to sigmoid function, tanh implies hyperbolic tangent function. The candidate activation function h`<t> at present is combinedly determined by r<t>, h<t-1> that is output hidden layer neuron at the previous point.

$$h`\langle t\rangle = \tanh(W.[r\langle t\rangle * h\langle t-1\rangle, x\langle t\rangle]) \tag{10}$$

With the calculation results from (8), (9) and (10), h<t> which is the current output can be obtained from (11).

$$h\langle t\rangle = (1-z) * h\langle t-1\rangle + Z\langle t\rangle * h`\langle t\rangle \quad (11)$$

5.4 *Proposed ensemble CNN_BiGRU*

Ensembling is the process in which multiple models are applied to a dataset and their results are aggregated to form a better -performing model [28]. We proposed an ensemble by integrating 1D CNN with a single BiGRU as shown in Figure 8. Various experiments were performed with different combinations to choose the suitable base classifiers, and CNN and GRU were chosen as base classifiers. Usually, ensemble-based model makes better predictions than the independently trained models. In this work, 1D CNN and GRU are good models to be merged for the Hindi SC task.

The proposed ensemble is comprised of the following elements: Input layer consists of Hindi MRs and it defines the length of review

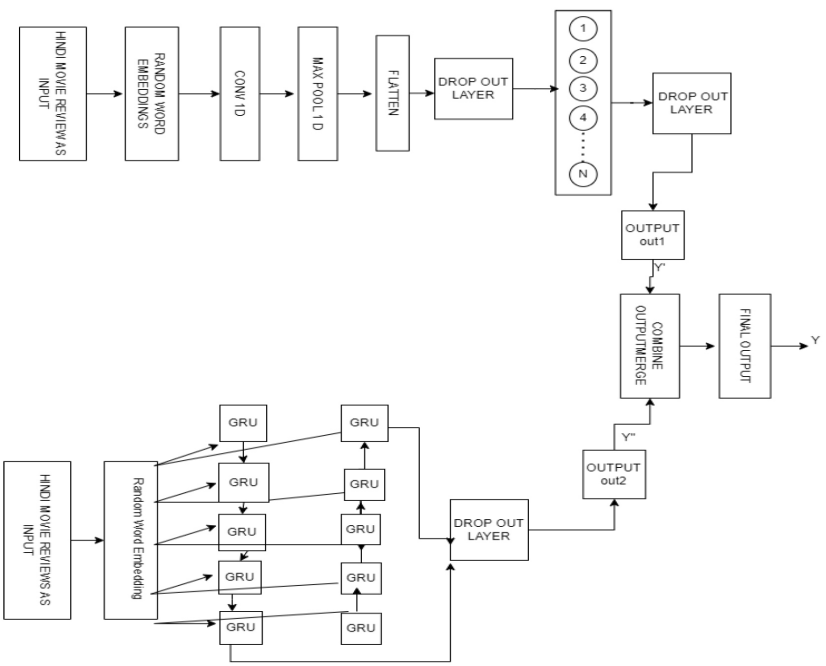

Figure 8

The proposed framework for CNN_BiGRU

sequences. The maximum sequence length is chosen as the length of the longest review in training set. Zero post padding is used. Embedding layer is set to vocabulary size and it improves while training. ConV 1D applied consist of 32 filters. The size of kernel is set to the number of words considered at once, we experimented with different kernel sizes such as (1,2,3,4,5 &6). To consolidate the output from ConV layer max pooling layer is used. This layer extract main features and compress the text features obtained by ConV layer. Flatten layer is used for dimensionality reduction. In stage 1, 1D CNN is applied using the elements mentioned above and Y` is the output (out1) generated by 1D CNN. Thereafter, in stage 2 the BiGRU is applied on reviews as we are dealing with Hindi textual reviews, therefore both preceding and following words have effect on the present word. After initial steps same as in stage 1, BiGRU unit is applied on the reviews, after the WEs layer two GRU layers are there to consider earlier and later information in the sequence. In this way forward and backward information processing used to take place. Next, it goes to the drop out layer which is employed to deal with the overfitting problem. In drop out layer, we are basically activating some features and deactivating some other features. This way Y`` that is the output (out2) generated by stage 2. Both the outputs are combined using concatenate to get the final output Y. This way reviews are classified as either positive or negative.It is known that despite of little parameter tuning, 1D CNN has proved to perform well for classification task. While BiGRU, which is a sequencing processing model with two GRU's performs efficiently on temporal data by considering both earlier and later information in the sequence. CNN is very poor in dealing with sequential information, while GRU is not good enough at extracting local contextual features. By amalgamating both together into one, as, this proposed ensemble balances out and compensates for each other's weaknesses, and the resulting model will maximize their respective strengths. Local static features from Hindi textual reviews are extracted using CNN, and sequence information is extracted through GRU. This ensemble will further enhance the bi-polar classification strategy. The hyperparameters used in proposed CNN_ BiGRU and scratch algorithm is given in Table 3 beneath.

Table 3
Hyperparameter Configuration for our CNN_BiGRU Ensemble.

Name of Parameter	Parameter Value
Optimizer	Adam
epochs	15
Batch size	60
Loss	Binary_cross entropy
Weight regularization (L2)	3
Padding type	Post zero padding
Activation	ReLU (Hidden Layer), Sigmoid (Output layer)
Drop out	0.5

Algorithm: CNN_BiGRU

Input: Hindi MRs Corpus
1: Procedure PRE-PROCESSING
2: while movie reviews in clean_reviews do
3: {
4: Tokenization
4: Stop Words Removal
5: Punctuation Removal
6: Numeric Removal
7: Specialcharacters removal
8: Non-Hindi Words removal
9: }
10: end While
11: end Procedure
12: Procedure FEATURE SELECTION
13: while movie reviews in directory do
14: word_vectors randomly_initalized
15: improved during training
16: end while
17: end Procedure
18: Procedure DIVIDE DATA
19: 10-fold Cross-validation
20: end Procedure
21: Procedure MAKE ENSEMBLE CNN_BiGRU

```
    while layers do
        model ← model.layers
    end while
        model ← model.compile
end Procedure
Procedure COMBINE RESULTS
    CNN ← out1
    BiGRU ← out2
    merged = concatenate ([out1, out2])
end Procedure
```

5.5 *Evaluation Metric*

The Hindi MR corpus achieves balanced distribution of both the polarity classes and avoids divergence of experimental results. Since, the corpus is almost balanced, therefore average accuracy is used as the performance evaluation metric in this work. Confusion matrix summarizes the classification performance [31]. It consists of four parameters True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). The description of confusion matrix for bi-polar classification is given given in Table 4. Using confusion matrix, we can calculate accuracy as given in eq. 12.

Table 4
Confusion matrix

Class1	Class 2	
TP	FP	Positive
FN	TN	Negative

Accuracy refers to accurately predicted MRs classes out of total predictions made [24][31].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{12}$$

Where, TP refers to the number of positive MRs correctly predicted as positive.

FP refers to the number of positive MRs incorrectly predicted as negative.

FN refers to the number of negative MRs incorrectly predicted as positive.

TN refers to the number of negative MRs correctly predicted as negative.

6. Results and Discussions

This paper's primary focus was to develop a deep learning-based system for bi-polar SC of MRs for a resource deficient language – Hindi. In order to achieve this objective, an ensemble CNN_BiGRU is proposed to take advantage of the competence of DLMs and the robustness of the ensemble for performing bipolar categorization of Hindi MRs. Due to the unavailability of a standard public Hindi dataset, we have used our corpus, comprising 6K Hindi MRs divided into positive and negative polarity classes collected from online sources. The present work is confined to coarse-grained sentience level SC of Hindi MRs.

Our corpus achieves the balanced distribution of both the polarity classes and avoids divergence of experimental results. Since the corpus is almost balanced, average accuracy is used as the performance evaluation metric. The present work is implemented using Intel i5 processor, 8GB RAM, GTX1080Ti in Python using Jupyter notebook IDE.

In order to obtain reliable results, the corpus was 10- fold cross-validated as there is no standard train/test split for our corpus. After preprocessing and feature extraction, some mainstream DLMs like 1D CNN, LSTM, and GRU are initially applied to MRs corpus. Firstly, 1D CNN was applied to MR corpus; it had a 1D convolution and pooling layer. Hyperparameters are used as given in Table 5, and the 1D CNN model employed in this work is shown in Figure 4.

It is known that CNN has the capacity for feature extraction using filters. Therefore, 100 filters have been used. Also, a feature map is built and outputted to the pooling layer. This layer aggregates the data provided by the convolution layer and reduces its representation. CNN takes Hindi MRs as input; the importance is assigned to features present in reviews, and this way, CNN distinguishes one review from another. Here less preprocessing is required in comparison to other classification algorithms applied. Since we are working with textual Hindi reviews, which are sequential, therefore is a one-dimensional matrix. A WEs layer and 1D convnet are needed to perform bipolar classification using CNN.

In the convolutional layer, the input review is multiplied by the kernel to get the convolved layer. In training and testing, different kernels were utilized, with a kernel size of 1 to 6 and epochs of 15. The 10-fold-cross-validation was applied for every kernel. The highest average accuracy of

88.699% was obtained with kernel size 6, while the second highest accuracy was 88.416% with kernel size 3 taking batch size 32 and 15 epochs. Table 6. Lists all the average accuracies obtained from CNN. The kernel size-based comparison of average accuracy is given in Figure 9.

Table 5

Hyperparameters for 1 D CNN.

No. of filters	Size of kernel	Activation Function	Constraint Max Norm	Drop out
100	1,2,3,4,5 & 6	ReLU	3	0.5

Table 6

The average accuracy obtained by 1D CNN for SC of Hindi MRs for kernel 1 to 6 respectively.

A_{cc}	Kernel 1	Kernel 2	Kernel 3	Kernel 4	Kernel 5	Kernel 6
A_1	87.166	87.333	87.166	86.333	89.666	89.333
A_2	90.499	88.999	88.999	87.999	86.666	88.499
A_3	90.333	88.333	89.333	87.833	85.166	88.666
A_4	85.833	88.166	89.166	88.499	88.833	90.333
A_5	87.333	87.999	85.833	87.833	89.166	88.833
A_6	86.833	90.833	88.999	88.666	86.000	87.666
A_7	88.666	87.500	88.333	89.333	87.666	90.499
A_8	90.166	88.499	88.333	89.333	88.333	87.666
A_9	87.166	88.833	89.833	89.499	87.666	88.833
A_{10}	87.333	86.000	88.166	87.166	88.833	86.666
$\mathbf{A_{vg(Acc)}}$	88.132	88.249	88.416	88.249	87.799	**88.699**

To experiment further, LSTM and GRU were both applied separately for bipolar categorization of Hindi reviews. Based on evidence obtained from the state-of-the-art studies, it is implied that LSTM and GRU were both excellent for short text SC [10]. LSTM and GRU were both applied separately for bi-polar categorization of Hindi reviews. While experimenting, the batch size 32, sigmoid activation function, and dropout rate of 0.5 was used. The number of epochs used was 15; if less than 15 are taken, model overfitting will occur.

Adam optimizer, known to be the best optimizer, is used. The Kernel size used is 3. To use maximum data for training, the model was cross-

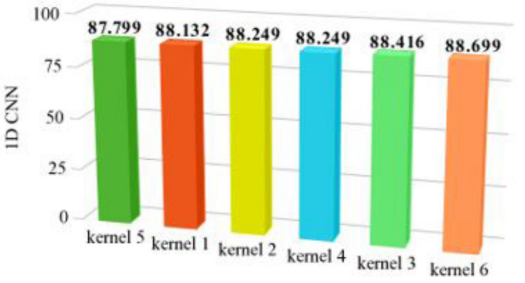

Figure 9

The average accuracy-based comparison of 1D CNN based on the size of kernel.

validated in each epoch. The hyper-parameters used for LSTM and GRU employed are mentioned in Table 7. The average accuracy obtained from GRU and LSTM is shown in Tables 8 and 9, respectively. Experimental results showed that RNN variants – LSTM and GRU outperformed the 1D CNN and achieved an average accuracy of 89.230 % and 89.049%, respectively at kernel size 3. Among these two sequence data benchmark models, LSTM performs better than GRU, as LSTM generally avoids language dependency; therefore, it is widely used for SA.

Table 7

Hyperparameters for the LSTM and GRU model.

RNN Models	Activation Function	No. of Units	Drop out
LSTM	ReLU (hidden layer), Sigmoid (output layer)	64	0.5
GRU	ReLU (hidden layer), Sigmoid (output layer)	64	0.5

Table 8

The average accuracy obtained by LSTM for bi-polar SC of Hindi MRs.

A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	$A_{vg(Acc)}$
90.166	88.333	89.499	89.833	87.5	89.660	87.5	90.160	89.990	89.660	**89.230**

Table 9

The average accuracy obtained by GRU for bi-polar SC of Hindi MRs.

A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	$\mathbf{A_{vg(Acc)}}$
90.833	90.166	86.833	88.999	88.833	89.333	89.499	87.166	89.499	89.333	**89.049**

It has been observed from the experiments performed above that DLMs are very competent for the bi-polar classification of Hindi MRs, and it motivated us to use ensemble for performing bi-polar Hindi review classification. The proposed ensemble as mentioned in section 5.4 is applied to MRs corpus.

For the ensemble, the parameter configuration is as follows: The activation function used to ReLU and the optimizer employed is 'Adam.' The dropout probability is set to 0.5 when fine-tuning. The batch size used was set to 60. The number of iterations or epochs is set to 15. It is known that learning rate (lr) is vital for weight and offsets optimization. Therefore, lr of 0.001 was employed. Binary-cross entropy is used as a loss function.

Different kernel size is utilized to test the proposed model, as the different size of the kernel will affect the training and accuracy results. Six kernel size has been utilized as shown in Table 10. The average accuracy-based comparison is shown in Figure 10. The experimental results show that the ensemble performed optimally at kernel one and achieved the highest average accuracy of 89.366%, which is better and the highest among the base DLMs such as CNN, LSTM, and GRU, which are applied in this work. While the average accuracy at kernel size 2, 5, and 6 is higher than the highest accuracy obtained by CNN.

The best average accuracies obtained by the DLMs deployed in this work is shown in Figure 11, from the results it is implied that the proposed CNN_BiGRU has achieved the highest accuracy and is more efficient in finding classification in patterns in training Hindi MRs. Consequently, it is concluded that CNN_BiGRU has better bi-polar classification performance than other DLMs. By employing our proposed ensemble to classify HMRs, we can help viewers/ reviewers decide whether to watch a movie or not in time. And could help save their precious time. It is known that DLMs need massive data as support; in present work, the proposed model with 6k corpus size is able to perform exceptionally well. Thereby, in the future, we will enhance the size of the dataset to further improve classification results. Also, other combinations of potent DLMs can be combined in ensemble to investigate if they can challenge state-of-the-art models in case of resource poor scenario. Further, to improve the generalization capability of Hindi MRs SC, more delicate granularities/sentiments and multilabel aspects can be considered.

Table 10

Average accuracy obtained by (Ours) CNN_BiGRU for SC of Hindi MRs for kernel 1 to 6 respectively.

A_{cc}	Kernel 1	Kernel 2	Kernel 3	Kernel 4	Kernel 5	Kernel 6
A_1	89.166	89.833	86.333	86.333	88.166	88.999
A_2	89.666	88.166	89.999	87.666	89.499	87.999
A_3	88.333	90.333	88.999	88.499	88.666	90.499
A_4	90.499	88.333	88.333	87.999	88.666	87.666
A_5	89.999	88.499	87.500	87.333	89.499	87.833
A_6	89.166	86.166	87.999	89.166	88.333	88.666
A_7	88.999	89.666	89.833	88.499	90.333	90.166
A_8	90.833	88.999	87.833	89.166	89.833	88.833
A_9	88.666	88.499	89.666	88.666	88.666	88.499
A_{10}	88.333	88.999	87.999	87.999	87.833	89.666
$\mathbf{A_{vg(Acc)}}$	**89.366**	88.749	88.449	88.132	88.949	88.882

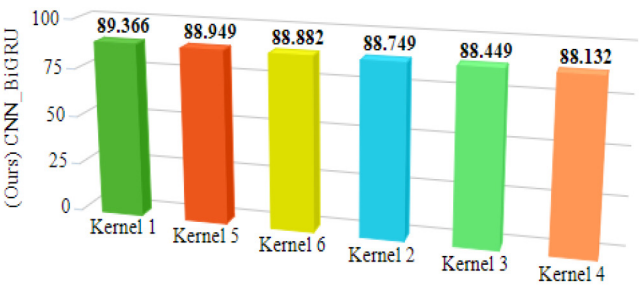

Figure 10

Average accuracy-based comparison of (Ours) CNN_BiGRU based on size of kernel.

7. Conclusion and Future Work

Intending at bi-polar SC of MRs for a resource deficient language - Hindi, MRs Corpus is made, and a deep learning-based ensemble CNN_BiGRU is proposed, which integrates CNN and BiGRU models. CNN_BiGRU is applied to the Hindi MRs corpus for classifying reviews into positive or negative classes. To further prove the efficacy of our proposed CNN_BiGRU, other mainstream DLMs like 1D CNN, LSTM, and GRU are also applied to MR corpus and compared with the proposed ensemble.

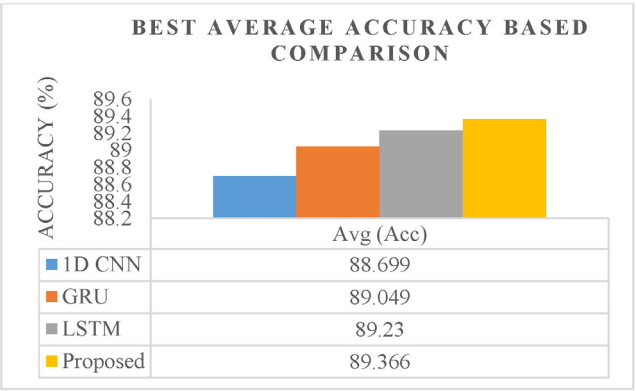

Figure 11

The best average accuracy obtained based comparison of the proposed CNN_BiGRU with state-of-the-art models.

The results show that our proposed CNN_BiGRU utilizing the random WEs has surpassed the DLMs applied in this work and has achieved a reasonably good average accuracy of 89.366% at kernel size 1, batch size 60 and 15 epochs. From the empirical results obtained, we can conclude that CNN_BiGRU has achieved state-of-the-art results and can be applied for bi-polar SC in a resource-deficient scenario, and it will help viewers to decide whether to watch a movie or not. The future aspect of this study is to consider other finer granularities and multi-label aspects to enhance the generalization capability of the SC system; also, the size of the dataset would be enhanced. Further, the results encourage us to experiment by combining other potent DLMs with existing good DLMs to see if they can challenge the state-of-the-art models even more.

Acknowledgment

This work is supported by the Guru Gobind Singh Indraprastha University through Indraprastha Research Fellowship (IPRF), award letter number GGSIPU/DRC/2022/1247.

References

[1] D. S Kulkarni and S.S Rodd, "Sentiment Analysis in Hindi—A Survey on the State-of-the-art Techniques," *In Transactions on Asian and Low-Resource Language Information Processing*, 21(1), pp. 1-46 (2021).

[2] A. Jain and V. Jain, "Sentiment classification of twitter data belonging to renewable energy using machine learning," *in Journal of Information and Optimization Sciences*, 40:2, pp. 521-533 (2019).

[3] A. Sharma and U. Ghose, "Sentimental analysis of twitter data with respect to general elections in India," *in Procedia Computer Science*, 173, pp.325-334 (2020).

[4] T. Young, D. Hazarika, S. Poria, and E. Cambria," Recent trends in deep learning based natural language processing," *in IEEE Computational Intelligence Magazine*, 13(3), pp. 55-75 (2018).

[5] A. Sharma and U. Ghose, "Lexicon a linguistic approach for sentiment classification," *In IEEE 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 887-893 (2021).

[6] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *In Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4) (2018).

[7] A. Kumar and V. H. C. Albuquerque, "Sentiment Analysis Using XLM-R Transformer and Zero-shot Transfer Learning on Resource-poor Indian Language," *in Transactions on Asian and Low-Resource Language Information Processing*, 20(5), pp. 1-13 (2021).

[8] A.R. Pathak, M. Pandey and S. Rautaray, "Empirical evaluation of deep learning models for sentiment analysis", *in Journal of Statistics and Management Systems*, 22:4, pp. 741-752 (2019).

[9] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *in Journal of Artificial Intelligence Research*, 61, pp. 65-170 (2018).

[10] H. Qiu, C. Fan, J. Yao, and X. Ye, "Chinese Microblog Sentiment Detection Based on CNN-BiGRU and Multihead Attention Mechanism," *in Scientific Programming* (2020).

[11] S. Seshadri, A.K. Madasamy, and S.K Padannayil, "Analyzing sentiment in Indian languages micro text using recurrent neural network", *in IIOAB Journal*, 7, pp. 313-318 (2016).

[12] P. Mathur, R. Shah, R. Sawhney, and D. Mahata, "Detecting offensive tweets in hindi-english code-switched language," *in Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pp. 18-26 (2018).

[13] T. Y. S. S. Santosh and K. V. S. Aravind, "Hate speech detection in hindi-english code-mixed social media text," *in Proceedings of the ACM India joint international conference on data science and management of data*, pp. 310-313 (2019).

[14] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma, "Towards subword level compositions for sentiment analysis of Hindi English code-mixed text," *in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 2482-2491 (2016).

[15] P. Singhal and P. Bhattacharyya, "Borrow a little from your rich cousin: Using embeddings and polarities of English words for multilingual sentiment classification," *in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp.* 3053-3062 (2016).

[16] R. Bhargava, S. Arora, and Y. Sharma, "Neural network-based architecture for sentiment analysis in Indian languages," *in Journal of Intelligent Systems*, 28(3), pp. 361-375 (2019).

[17] M. G. Jhanwar and A. Das, "An ensemble model for sentiment analysis of Hindi-English code-mixed data," arXiv preprint arXiv:1806.04450. https://doi.org/10.48550/arXiv.1806.04450 (2018).

[18] C. Nanda, M. Dua, and G. Nanda, "Sentiment analysis of movie reviews in hindi language using machine learning," *in IEEE International Conference on Communication and Signal Processing (ICCSP)*, pp. 1069-1072 (2018).

[19] S. Ghosh, A. Paul, M. Sharma and S. Basu, "Decision-making under Risk: Evidence from the Hindi Movie Industry," *in Journal of Operations and Strategic Planning*, 1(1), pp.15-33 (2018).

[20] V. Singh, P. Saxena, S. Singh and S. Rajendran, "Opinion mining and analysis of movie reviews," *in Indian Journal of Science and Technology,* 10(19), pp.1-6 (2017).

[21] S. Rani and P. Kumar, "A journey of Indian languages over Sentiment Analysis: A Systematic Review", *in Artificial Intelligence Review 52.2,* pp.1415-1462 (2019).

[22] Yoon Kim, "Convolutional Neural Networks for Sentence Classification," *in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746-1751 (2014).

[23] K. Saurav, K. Saunack, D. Kanojia, and P Bhattacharyya," A Passage to India": Pre-trained Word Embeddings for Indian Languages, arXiv preprint arXiv:2112.13800 (2021).

[24] A. Hafiz and A. Hafiz, "Novel Opinion mining System for Movie Reviews", *in International Journal of Intelligent Systems and Applications in Engineering 8.2*, pp.94-101 (2020).

[25] S. Murali and T. R Swapna, "An empirical evaluation of temporal convolutional network for offensive text classification", *in International Journal of Innovative Technology and Exploring Engineering*, vol. 8, pp. 2179-2184 (2019).

[26] J. Chung, C. Gulcehre, K. H Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555 (2014).

[27] X. Chen, C. Wang, D. Li, and X. Sun, "A New Early Rumor Detection Model Based on BiGRU Neural Network," *In Discrete Dynamics in Nature and Society* (2021).

[28] I. Nazeer, M. Rashid, S. K Gupta and A. Kumar, "Use of Novel Ensemble Machine Learning Approach for Social Media Sentiment Analysis," *in Analyzing Global Social Media Consumption, IGI Global,* pp. 16-28 (2021).

[29] X Wang and H. C Kim, "Text categorization with improved deep learning methods," *in Journal of information and communication convergence engineering,* 16(2), pp.106-113 (2018).

[30] S. R. Shah and A. Kaushik, "Sentiment Analysis on Indian Indigenous Languages: A Review on Multilingual Opinion Mining", arXiv:1911.12848 (2019).

[31] Md S. Akhtar, A. Kumar, A. Ekbal, and P. Bhattacharyya, "A Hybrid Deep Learning Architecture for Sentiment Analysis", *in proceedings of the 26th International Conference on Computational Linguistics (COLING)*, pp.482–493 (2016).

[32] G. Xu, Y. Meng, X. Qiu, Z. Yu and Xu Wu "Sentiment Analysis of Comment Texts Based on BiLSTM," *in IEEE Access, Special Section on Artificial Intelligence and cognitive computing for communication and network,* pp. 51522-51531 (2019).

Received August, 2022
Revised December, 2022