Multimedia Tools and Applications (2022) 81:17347–17368
https://doi.org/10.1007/s11042-022-12761-y

Rumor detection in social network based on user, content and lexical features

Sushila Shelke [1,2] · Vahida Attar [1]

Received: 21 May 2021 / Revised: 9 August 2021 / Accepted: 21 February 2022 /
Published online: 7 March 2022

Abstract
Emergence in the social network leads to the extensive and faster diffusion of news than conventional news channels. Verification of data is challenging due to massive information on a social network. Unverified information can be a rumor or fake news that causes damage to an individuals and organizations, revealing the harmful impact on humanity. Therefore, it is vital to combat rumor diffusion to minimize the adverse effects on society. Despite vigorous efforts to deal with this issue, researchers mainly focussed on temporal dynamics of posts and other features like a user, network, content-based, which demonstrate a moderate accuracy. The time series features are associated with an event that suppresses the other quality features related to each post. There is a scope for improvement in the accuracy, so this paper focuses on post-wise features such as user-based, content-based and lexical-based features along with post sequences. We proposed a framework that uses various essential features and combines two deep learning models. Word embedding is utilized with bidirectional long short-term memory (BiLSTM) and combined with post-wise features using a multilayer perceptron (MLP), which improves accuracy. The experiments on the real-world dataset of Twitter demonstrate a notable improvement in accuracy compared to state-of-the-art approaches.

Keywords Deep learning · Lexical features · Rumor · Rumor detection · Social network

✉ Sushila Shelke
sss17.comp@coep.ac.in; sushila.shelke@cumminscollege.in

Vahida Attar
vahida.comp@coep.ac.in

[1] Department of Computer Engineering and Information Technology, College of Engineering Pune, Savitribai Phule Pune University, Pune, India

[2] Department of Computer Engineering, Cummins College of Engineering for Women, Savitribai Phule Pune University, Pune, India

Abbreviations

BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
GAN	Generative Adversarial Network
GRU	Gated Recurrent Unit
LIWC	Linguistic Inquiry and Word Count
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
RNN	Recurrent Neural Network
TF-IDF	Term Frequency-Inverse Document Frequency
UCL	User-Content-Lexical features

1 Introduction

Exploding various social networking websites (e.g., Twitter, Facebook, Reddit) leads to a high impact on news and information propagation. There are 3.96 billion social media users around the world [36]. However, despite the internet access or the age of a person, 50% of the globe's population has a routine of using social networks. Stories or news related to politics, economy, social, scientific are uploaded continuously and spread rapidly in social networks, establishing a close real-time view of events or incidences worldwide. Using a social network, the public connected to the world has smooth and faster access to live updates and can help others. Even though it has many advantages, there are challenges verifying the truthfulness of posts [5, 7] and identifying the user who propagates a rumor or fake news on the social network [33]. Original posts are frequently altered by malicious users and disseminated quickly around the internet. Due to these successive moderations, the meaning of the initial post changes in the wrong way. In the rapid diffusion of such news, bots play an important role, where bots are the automatic programs that share fake news with a very high frequency than usual social media users [29]. Fake news and rumor are used alternatively in the literature, but there is a difference in the terms. Fake news is a verifiably false and intentionally misleading article [6]. However, a rumor is a story in circulation, which is unproven at the moment and may get proved as true, false, or unverified at a later stage [43]. The most circulating and misleading stories are based on an individual related to death or defamation, the organization's reputation, the quality of any product, etc.. Such stories are later proven as rumors if those are false. Intentionally spreading such wrong information misled the population and endorsed severe incidences such as violence [39]. Therefore, there is a need for a more accurate and automatic rumor detection system.

Recently, the whole world has been under the fear of coronavirus, which was started in December 2019 and continues to date. There are a few stories related to the COVID vaccine during this pandemic as below.

- COVID-19 Vaccine Cause Herpes [11]
- After the COVID-19 vaccine, blood or plasma donation is not allowed [1]
- COVID-19 killed fewer people than the flu [10]

Some effective strategies are required to fight the spread of such news, which builds fear and anxiety among society. Many specialty-based fact-checking websites such as Politifact, Snopes, FactCheck [13] work for debunking rumors or fake news. Also, there are crowdsourcing-based fact-checking sites like Twitter [40] and Facebook [12]. The rapid circulation of stories can create chaos within the society if not handled early. In the case of time-dependent events, the consequences can be frightful. News verification through manual efforts is time-consuming. Recognition of rumors and rumor sources [34] can control rumor dissemination.

Many researchers have put forth their views in a rumor detection survey. In [6], they categorize various features broadly into content-based (semantic and lexical features) and context-based (user and network-based) features. They depicted detection in terms of a classification problem by dividing it into four modules, detection, tracking, stance and veracity classification of rumor [43]. Also, reviewers classified the approaches based on machine learning-based (ML) and deep learning-based (DL) techniques. Many researchers have started with an ML-based method to solve the rumor detection problem [18]. However, the manual feature selection in ML approach is tedious and requires physical effort. Therefore, researchers moved on towards the DL-based approach to overcome the problems of ML classification. This research focused on deep learning-based strategies. The review on DL-based methods is depicted in [2], which shows a detailed analysis of the datasets utilized, various deep learning architectures and open challenges in rumor detection. The limitations explained in the existing review involve collecting or selecting benchmarked datasets, size or quality of data, and choice of DL architectures and relevant features.

This paper argues that the crucial focus of previous research was text and temporal features of posts using deep learning. Though few researchers combined time-dependent characteristics with other features such as user, content-based, they use aggregate or fraction values for such features. These aggregate values ignored many essential components associated with an individual post. This research utilizes the significant post-wise features from various categories such as user-based, content-based, lexical and post-based features using different deep learning models.

The contributions of this research are summarized below.

- We have collected a real-world dataset for rumor and non-rumor events from Twitter.
- Identified essential features from different categories such as user, content-based and lexical.
- We have proposed a hybrid deep learning model combining bidirectional LSTM (BiLSTM) and multilayer perceptron (MLP) models.
- We have comprehensively analyzed experimental results on real-world datasets and compared them with state-of-the-art deep learning-based rumor detection approaches.

The current paper has presented a literature review on rumor detection approaches using deep learning frameworks in section 2. The problem definition of rumor detection, data collection and methodology is explained in section 3. Section 4 discuss experimental results and, followed by a conclusion at the end of the paper.

2 Related work

The study in this research is a kind of classification problem used in many applications such as Email [30, 31], Sentiment classification [38] and Fake news detection [19]. The research work

in this paper focused on deep learning-based approaches. According to the different deep learning models used in the current work, we have divided the techniques based on recurrent neural network (RNN), convolutional neural network (CNN) and a combination of different models referred to as Hybrid models.

- RNN based approach: RNN is a form of a feed-forward neural network used to process the sequential data with a variable-length, such as time-series data and is the first to apply for rumor detection by Ma et al. [25]. They extended the basic RNN model with memory unit models like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), where GRU performs well. The words related to rumor should get more attention, as proposed by Chen et al. [8] using multilayer LSTM and deep attention model. The model depicts the soft attention to the recurrence of distinct features with a specific focus and produces hidden representations in the posts. Guo et al. [17] proposed the same attention-based technique for the hierarchical network of word-post-subevent using bidirectional LSTM. They utilized propagation features such as an average of reposts and comments along with user and post-based features. Chen et al. [9] proposed an unsupervised learning model for rumor detection as an anomaly detection problem on Sina Weibo, a Chinese microblogging website. They utilized microblogs features (like Question mark count, Sentiment score, Pictures count) and posts (like length, count of likes, URL count). The experimental results show that their projected method could attain an accuracy of 92% and an F1 score of 89%. The self-learning semi-supervised deep learning model and the trust network layer are used in the FakeNewsNet dataset [35], which uses a bidirectional LSTM (BiLSTM) model with a trust layer and shows an F1 score of 88% [22].
- CNN-based approach: Yu et al. [42] discover that RNN models are not suitable for early detection of rumors with limited input data, so they propose a CNN-based approach for misinformation identification (CAMI). This model can extract essential features from the input sequence and perform well. Rumor identification based on only text features using the BiLSTM-CNN model is projected by Asghar et al. [3], where they proposed a web-based interface for rumor detection. Their method showed an accuracy of 86%. Lin et al. presented a recurrent CNN and combined bidirectional GRU with an attention network, which helps understand the vital information at the word level and learn the temporal features [23]. Also, they utilize the signal words from the text along with a fraction of user-based and content-based features.
- Hybrid Models: Ruchansky et al. [27] proposed a model for fake news detection based on the text of an article, the temporal activity of user response and source users propagating it. They put forward a hybrid model by integrating features from all three categories to get a more precise rumor classification. A recommender system determines a user's genuine interest based on user involvement [32]; therefore, user characteristics are vital. Song et al. [37] proposed a CNN-based model for credible early detection of rumor where they extract feature vectors in each interval using CNN and feed it to RNN. Also, they suggest that other features related to the user profile and propagation patterns can improve rumor detection. Liu et al. [24] proposed a model for early rumor detection, which combines RNN and CNN to capture temporal patterns of global and local features of users with the propagation paths. They utilized user-related characteristics such as user registration age, geo-enabled, verified users, etc. Data based on information campaign and promoting is used and proposed a Generative Adversarial Network network (GAN) for rumor detection by Ma et al. [26]. Kumar et al. proposed a model in which a sentiment analysis of social

network users utilized various deep learning models like CNN, a variation of RNN as long short-term memories (LSTM) with ensemble and attention mechanism [21]. They used Glove (global vector for word representation) for word embedding.

Other than the proposed categories of deep learning-based approaches, the researchers have used ensemble learning for rumor detection and transfer learning for fake news classification. In ensemble learning combination of RNN, GRU and LSTM models is used with various layers in the neural network by Kotteti et al. [20]. A transfer learning using BERT (Bidirectional Encoder Representations from Transformers) model referred to as FakeBERT uses a combination of deep Convolutional Neural Network (CNN) with different kernel sizes and filters by Kaliyar et al. in [19]. The experimental results of the FakeBERT model on the fake news dataset show an excellent accurateness of 98.90%.

In the literature, most of the researchers targeted temporal and text features. While considering temporal features, they think of aggregated features, which hides the importance of post-wise features. Table 1 shows a comparison of deep learning-based methods, where most of the research focuses on textual and temporal characteristics. For text-based features, the most common models are RNN and LSTM. Table 1 depicts the excellent utilization of factors from various categories, which contributes to refining the detection accuracy of rumor and non rumor events. Table 2 compares the performance of benchmarked methods concerning the accuracy and F1 score on a real-world Twitter dataset. This table also presents the text conversion method used in the literature. The most commonly used text representation method in previous work is the term frequency-inverse document frequency (TF-IDF), whereas recently, word2vec is utilized with glove vector. CNN-based methods are mainly used to detect rumors or fake news early. Table 2 reveals that the hybrid model shows excellent performance in terms of accuracy of 89% [27]. In Table 2, methods used in [8, 27, 37] consider either overall accuracy or cross-validation. From the literature, we can conclude that we have a scope for improvement in the preciseness of rumor detection.

Due to the advantage of the hybrid model from the literature, this research combines the BiLSTM model with Multilayer Perceptron (MLP) as a hybrid deep learning model. It explores different attributes from the user, content-based, lexical and text of a post. Features from each category are listed in section 3.

Table 1 Deep learning-based approaches with various features

Ref. no.	Approach	Models	Features type			
			User	Text	Propagation	Temporal
[25]	RNN	RNN, LSTM + GRU, Multilayer GRU		√		√
[8]	RNN	RNN+Multilayer LSTM		√		√
[17]	RNN	Bi-LSTM	√	√	√	
[9]	RNN	RNN+Autoencoder	√	√		√
[42]	CNN	CNN+Max Pooling		√		√
[3]	CNN	Bi-LSTM		√		
[23]	CNN	CNN+Bi-GRU	√	√		√
[27]	Hybrid	RNN+NN	√	√		√
[37]	Hybrid	RNN+CNN		√		√
[24]	Hybrid	GRU+CNN	√		√	√
[26]	Hybrid	GAN+CNN, GAN+GRU		√		√

Table 2 Comparative study of performance metrics

Ref. no.	Approach	Text representation	Model	Class	Accuracy	F1
[25]	RNN	tf-idf	GRU2	R	0.881	0.898
				NR		0.86
[8]	RNN	tf-idf	CallAtRumors-LSTM	–	–	0.87
[17]	RNN	tf-idf	HSA-BLSTM	R	0.844	0.825
				NR		0.863
[42]	CNN	Paragraph2vector	CAMI-CNN	R	0.777	0.793
				NR		0.758
[23]	CNN	word2vector	RCNN-FAN	R	0.799	0.792
				NR		0.805
[27]	Hybrid	doc2vec	CSI-LSTM+NN	–	0.892	0.894
[37]	Hybrid	tf-idf	CED-CNN+RNN	–	0.744	0.747
[24]	Hybrid	tf-idf	GAN-GRU	R	0.863	0.866
				NR		0.858

3 Methodology

This section presents problem definition, data collection and pre-processing, feature selection and methodology followed using a deep learning model.

3.1 Problem definition

The rumor detection in social networks formally presented as the event-wise sequence of posts given as input to the proposed model identifies whether the event is rumor or non-rumor. The event is any condition or incident that happened around us and informed through news, messages, like news-related bomb blasts, political statements, targeted organizations, etc. The input data contains a set of events $E = \{e_1, e_2, \ldots, e_N\}$, where N is the total number of events. Each event includes n posts, as $e_1 = \{p_1, p_2, \ldots, p_n\}$, where n is varying in size and p contains message or tweet related to the event. Along with the posts, other features are extracted, such as, $UserVect = \{u_1, u_2, \ldots, u_n\}$, where $u_1 \ldots u_n$ are user features like user registration age, friend count, etc., $ContVect = \{c_1, c_2, \ldots, c_n\}$, where $c_1 \ldots c_n$ are content-based features like number of question marks, number of URLs, etc. and $LexVect = \{l_1, l_2, \ldots, l_n\}$, where $l_1 \ldots l_n$ are lexical features. The proposed model's goal is to check whether the event is a rumor or not by considering various post-related features.

3.2 Data set

This research focused on the Twitter microblogging website. The data collection for rumors involves identifying rumor and non-rumor events from debunking sites, collecting data related to each event from Twitter and finally, cleaning data. This section presents the entire process of data curation.

3.2.1 Data collection

We have identified rumor and non-rumor events from debunking sites of www.snopes.com and www.politifact.com. These sites have the details of story circulation, evidence of news,

statuses such as True or False after investigation and the published date of the story. After verifying each tale with True or False, the events are categorized into Rumor and Non-rumor. Figure 1 shows the recognition of rumor events from snopes.com with a rating as False [16] and Fig. 2 shows the determination of non-rumor events from the Politifact site with news status as "Mostly True" [4]. Figures 1 and 2 present the example of event identification from various websites. The tweets were collected for each event from 1st March 2020 to 31st March 2020. Twitter's 30-day endpoint premium API paid scheme extracts tweets from the last 30 days during the above period. The data for each event is collected by writing different search queries. Figure 3 shows the data collected for news by altering the keywords in search queries highlighted in bold. The sample examples of finalized events related to rumor and non-rumor are listed in Tables 3 and 4. These tables give details of event headline, count of posts for each event and date of data collection.

The statistics for real-world data collected from Twitter are given in Table 5. 78% of events are identified from politifact.com and 22% are from snopes.com. Besides the dataset we formed, we also utilize the publicly available Twitter dataset [28], constructed by [25] for rumor detection. Due to Twitter's policy, only tweet ids are given in a dataset for rumor and non-rumor events. Therefore, we have extracted tweets for each tweet id in JSON format. This research focuses on English posts only. Twitter does not provide data for a few events for reasons such as user does not exist, user suspended. After data pre-processing, the events having a single post and non-English language are removed from the dataset. Therefore, the final count for the events is 986, whereas the total posts become 267,708. The detailed steps of data pre-processing are explained further.

3.2.2 Data pre-processing

The data extracted from Twitter contains URLs, hashtags, mentions, emoticons, special characters, etc., which need to be preprocessed to use the cleaned text data as input to our model. The text data is prepared by removing URLs, hashtags, mentions, emoticons, punctuations, and non-ASCII characters using the python regex library represented by *re*. The duplicate posts are removed from the dataset. We have expanded the contractions

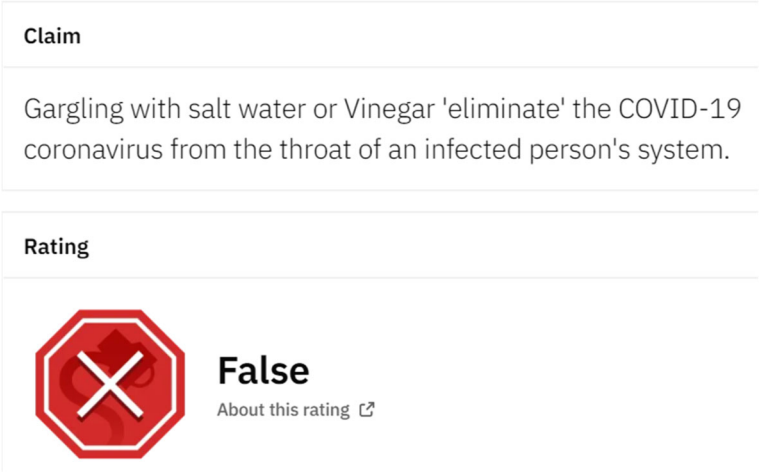


Fig. 1 Identification of event as rumor from snopes.com

Fig. 2 Identification of event as non-rumor from Politifact.com

present in the tweet, such as can't to "cannot", don't to "do not". Finally, non-ASCII characters are removed and text is converted to lower string. The detailed function of text preprocessing is shown in Fig. 4. These clean tweets are used to pull out lexical features and word embedding. The content-based components are extracted from the original tweets.

Data encoding in the numeric form must be needed for the text data to input a deep neural network. Besides the approximate average length of all posts, we consider the maximum lengths for the sequence as 100. The *Tokenizer* separates the post into different tokens and used as a word dictionary. This word dictionary is used to convert the message into a sequence of integers using the *text_to_sequence* function. Padding after the post is performed to make all sequences equal to maximum length. The embedding layer is used to understand the meaning of words in a post, transforming each word into an n-dimensional word embedding vector by taking a sequence of posts as an input. The output of this embedding layer passed as an input to the BiLSTM model.

3.3 Feature selection

Text and time series data are the main attraction of recent work in rumor detection. But along with text, we have targeted features like a user, content-based, and lexical features from posts. User-based features include registration age, description length, follower count, etc. Lee and Bosch explored lexical and content-based features for identifying the variety in languages [41]. The content-based features are grammar-level features, including part of speech (POS)

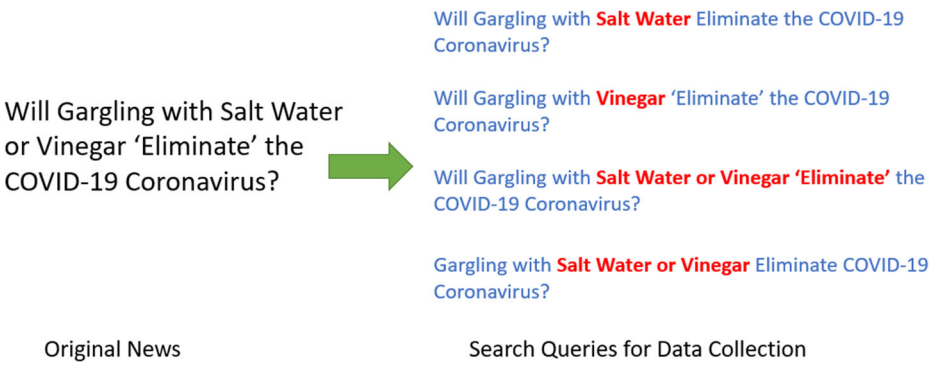


Fig. 3 Example of search queries for data collection of events

Table 3 Sample list for rumor events

Sr. no.	Date	Event story	Post_count
1	09-03-2020	A homemade hand sanitizer made with Tito's Vodka can be used to fight the new coronavirus.	2186
2	11-03-2020	Will an Asteroid Hit Earth in April 2020?	37
3	21-03-2020	Says drinking a bleach solution will prevent you from getting the coronavirus	123
4	21-03-2020	Don't hold your breath. This isn't a credible way to test for coronavirus	996
5	23-03-2020	Will Eating Bananas Prevent Coronavirus Infection?	60
6	23-03-2020	Did Nostradamus Predict the COVID-19 Pandemic?	125
7	26-03-2020	Can You Get a Free Coronavirus Test by Donating Blood?	143
8	26-03-2020	Will Gargling with Salt Water or Vinegar 'Eliminate' the COVID-19 Coronavirus?	144
9	26-03-2020	Will Sipping Water Every 15 Minutes Prevent a Coronavirus Infection?	17
10	28-03-2020	Beware of rumors of robbers posing as COVID testers	110
11	29-03-2020	Does 'Every Election Year' Have a Coinciding Disease?	4971

tagging, count of question marks, exclamatory marks, word count, etc. However, lexical features from each post are identified using the Empath open-source library [14] to analyze the text into lexical features, similar to Linguistic Inquiry and Word Count (LIWC). It provides a total of 190 lexical features. Table 6 shows the identified features from each category. We have extracted 8 features from the user category, 12 from content-based, and 190 from the lexical category. Principal component analysis (PCA) was applied as a dimension reduction technique on 190 features from the lexical category. The optimum number of components is determined using the cumulative explained variance graph in Fig. 5 and concluded with 125 as principal components. These features are normalized using standard scalar and feed to a multilayer perceptron (MLP), one of the deep neural networks.

Features are represented through boxplot and heatmap in feature selection from the user and content-based category. Figure 6 shows the boxplot for value distribution of user registration

Table 4 Sample list for non-rumor events

Sr. no.	Date	Event story	Post_count
1	24-03-2020	Was 'Coronavirus' Replaced with 'Chinese Virus' in Trump's Notes?	10,304
2	27-03-2020	Bill Gates told us about the coronavirus in 2015	5000
3	27-03-2020	Was COVID-19 Discovered in the US and South Korea on the Same Day?	52
4	27-03-2020	Are Most Cruise Ships Registered Under Foreign Flags	1474
5	27-03-2020	Did Video Show Italian Army Trucks Transporting Coffins Amid COVID-19 Pandemic?	92
6	27-03-2020	Is This a Photo of an American Revolutionary War Vet?	865
7	28-03-2020	Spectrum will provide free internet to students during coronavirus school closures	2850
8	28-03-2020	Does 'Triscuit' Mean 'Electric Biscuit'?	68
9	01-04-2020	Did Empire State Building Display 'Siren' Lights During COVID-19 Pandemic?	9994
10	01-04-2020	Did Cities Close Schools, Businesses During the 1918 Pandemic?	36
11	01-04-2020	Did the Trump Administration Send 18 Tons of PPE to China in Early 2020?	1092

Table 5 Details of real-world and benchmarked dataset

Name	Real-World	Benchmarked		
	English Only	Existed	Extracted	English Only
Total Events	70	992	990	986
Total Rumor Events	51	498	498	498
Total Non-Rumor Events	18	494	492	489
Total Posts	85,560	340,176	274,530	267,708
Total Rumored Posts	47,209	132,470	105,256	104,920
Total Non-Rumored Posts	38,351	207,706	169,274	162,788
Minimum Posts Per Event	2	10	2	2
Maximum Posts Per Event	10,304	3029	2838	2702

age and it can be noticed that the user sending genuine posts is much older on Twitter than the user sending fake posts. Figure 7 shows the difference between the correlation matrix of features from user and content-based groups related to rumor and non-rumor posts. It can be observed that few features (retweet count and follower count, verified user and follower count) in non-rumor data are highly correlated.

3.4 Proposed model

This section presents the details of various deep learning models implemented in this research, including existing models such as words Embedding, BiLSTM and MLP, and newly proposed models, which are explained below:

3.4.1 Word embedding

The dataset contains several tweets *twt* and every tweet *twt* is encompassed of an order of n words, i.e.t_1, t_2,….t_n . Each word t_i transform into an embedding vector $w_i \in E^m$, called as word embedding. The Keras embedding layer is utilized in this research. The input to the embedding layer contains an input matrix of two dimensional, also known as word embedding matrix represented by $E^{l \times m}$, where l is the tweet's length, and m is the dimension of word

```
def clean_text(post):
        clean_pst = re.sub('http\S+\s*', '', x)  # remove URLs
        clean_pst = re.sub('RT', ' ', clean_pst)  # remove RT
        clean_pst = re.sub('#\S+', '', clean_pst)  # remove hashtags
        clean_pst = re.sub('@\S+', '', clean_pst)  # remove mentions
        clean_pst = Expand contractions
        clean_pst = Remove punctuation symbols
        clean_pst = Remove emoticons, symbols and pictographs
        clean_pst = Remove non asscii characters
        clean_pst = clean_pst.lower() #Convert text to lower case
        clean_pst = clean_pst.strip()
        return clean_pst
```

Fig. 4 Pseudocode for text preprocessing

Table 6 Identified features from user, content and lexical category

Category	Features
User	User_Registration_Age, Is_Verified_user?, User_Description_Length, Follower_count, Friends_count, Favourite_count, Status_count, User_Location_present
Content based	#Hashtags, #URLs, # Question_Marks, #Exlamatory, #Mentions Retweet_count, Word_count, Sentiment_score, Is_Media_present, Tweet_favorite_count, Tweet_geo_location_present, Tweet_reply_count
Lexical	alcohol, ancient, anger, animal, anonymity, anticipation, appearance, art, attractive, banking, beach, beauty, blue_collar_job, body, breaking, business, car, celebration, cheerfulness, childish, children, cleaning, clothing, cold, college, communication, competing, computer, confusion, contentment, cooking, crime, dance, death, deception, disappointment, disgust, dispute, divine, domestic_work, dominant_heirarchical, dominant_personality, driving, eating, economics, emotional, envy, exasperation, exercise, exotic, fabric, family, farming, fashion, fear, feminine, fight, fire, friends, fun, furniture, gain, giving, government, hate, healing, health, hearing, help, heroic, hiking, hipster, home, horror, hygiene, independence, injury, internet, irritability, journalism, joy, kill, law, leader, legend, leisure, liquid, listen, love, lust, magic, masculine, medical_emergency, medieval, meeting, messaging, military, money, monster, morning, movement, music, musical, negative_emotion, neglect, negotiate, nervousness, night, noise, occupation, ocean, office, optimism, order, pain, party, payment, pet, philosophy, phone, plant, play, politeness, politics, poor, positive_emotion, power, pride, prison, programming, rage, reading, real_estate, religion, restaurant, ridicule, royalty, rural, sadness, sailing, school, science, sexual, shame, shape_and_size, ship, shopping, sleep, smell, social_media, sound, speaking, sports, stealing, strength, suffering, superhero, surprise, swearing_terms, swimming, sympathy, technology, terrorism, timidity, tool, torment, tourism, toy, traveling, trust, ugliness, urban, vacation, valuable, vehicle, violence, war, warmth, water, weakness, wealthy, weapon, weather, wedding, white_collar_job, work, worship, writing, youth, zest

embedding. The parameters used in the Embedding layer are the size of embedding dimension (m) =32, vocabulary size =1000 and length of the sequence (l) = 100.

3.4.2 Bidirectional LSTM (BiLSTM)

We have used a variant of RNN as a bidirectional LSTM model (BiLSTM), which involves forward LSTM and backward LSTM. LSTM specifically fails to remember part of the historical data through three entryways (input door, forget door and output door), adds the

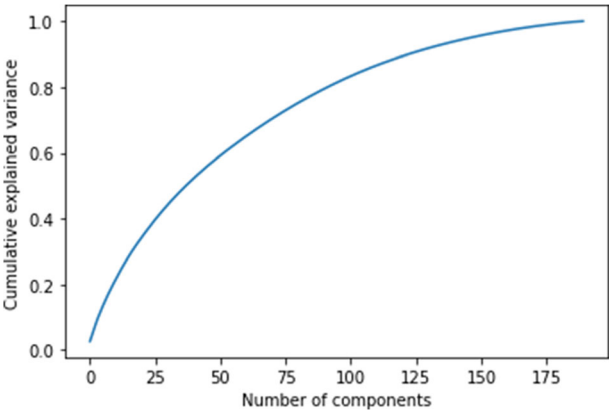


Fig. 5 Cumulative explained variance graph for PCA components

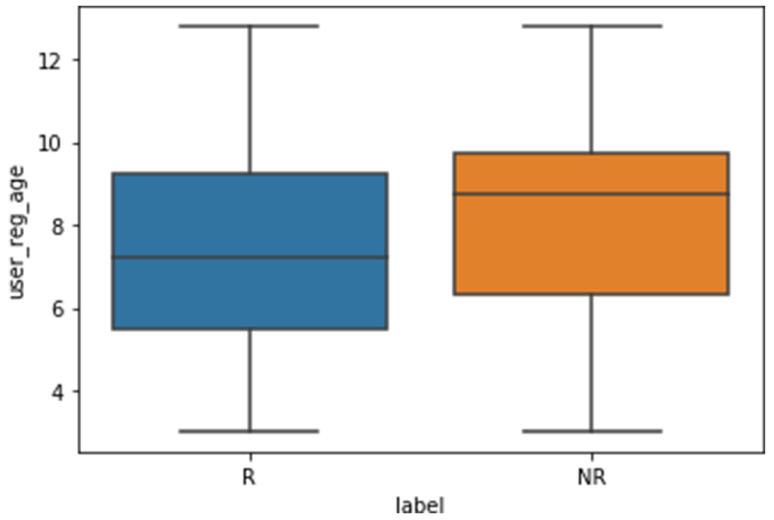


Fig. 6 Distribution of user registration age over rumor and non-rumor posts

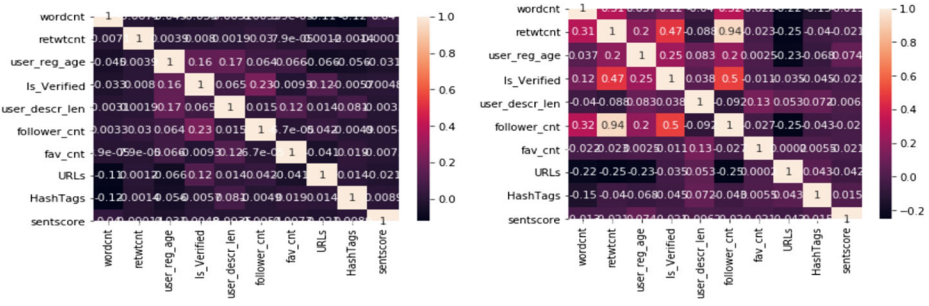
amount of the current information data, and incorporates it into the present status to create the output state. The word embedding vector of posts is given as input to BiLSTM and represented by the *BiLSTM_Embed* model.

3.4.3 Multilayer perceptron (MLP)

MLP is one of the deep neural network (DNN) used to learn the post-wise features from the user, lexical and content-based features. A feature vector of 200 is given as input to MLP. A feature vector from MLP and BiLSTM is combined and provided as input to a densely connected layer in a hybrid model. lex_PCA and UCL_PCA models are executed explicitly on MLP.

3.4.4 Overview of proposed models

Based on the above-mentioned existing models, we have implemented BiLSTM_Embed, Lex_PCA, UCL_PCA and BiLSTM_UCL models.



(a) Rumor (b) Non-Rumor

Fig. 7 Correlation matrix for rumor and non-rumor posts

- BiLSTM_Embed - Word embedding vector generated in the word embedding layer is feed into the BiLSTM model having 2 dense layers.
- Lex_PCA - 190 features from the lexical category are reduced to 125 using the PCA technique. To understand the significance of PCA, the Lex_PCA model is implemented and feeds with 125 principal components of lexical attributes as input. Lex_PCA model is implemented with 3 dense layers.
- UCL_PCA - model combines 20 features from user and content-based categories and 125 components from lexical category. Altogether, 145 features are normalized using standard scalar and passed as input to MLP.
- BiLSTM_UCL – Combines the features vector from BiLSTM_embed and UCL_PCA. Figure 8 shows the block diagram for the BiLSTM_UCL model's input and output.

The final proposed BiLSTM_UCL is a combination of BiLSTM_Embed and UCL_PCA, depicted as proposed deep learning architecture in Fig. 9. The variance of RNN and extension of basic LSTM as Bidirectional LSTM (BiLSTM) is used for the sequence of posts. BiLSTM processes the text forward and backward, providing additional context to the network with a better learning on the problem. After applying the post sequence to BiLSTM, we get a feature vector *text_feat* and feed as input to this model's fully connected layer fc_1. We extracted different feature vectors for each post, such as user-based features as *UserVect* having user registration age, friend count, etc. Content-based features as *ContVect* have features like a count of Question Marks and the number of URLs, etc. and lexical features after applying PCA as *LexVect*. The features of each post are combined into a different vector such as FeatVect = {UserVect, ContVect, LexVect}} and they are given as input to fully connected layer fc_3 to get the second feature vector. The feature vectors from both the networks and dense layers fc_1 and fc_4 are merged and input to dense layer fc_5. The activation function used are relu in the

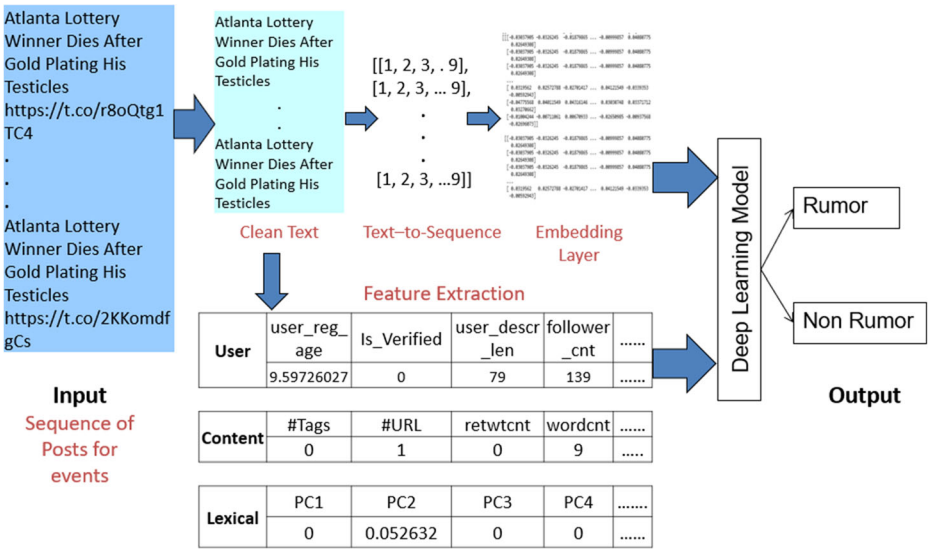


Fig. 8 Block diagram for input and output of BiLSTM_UCL model

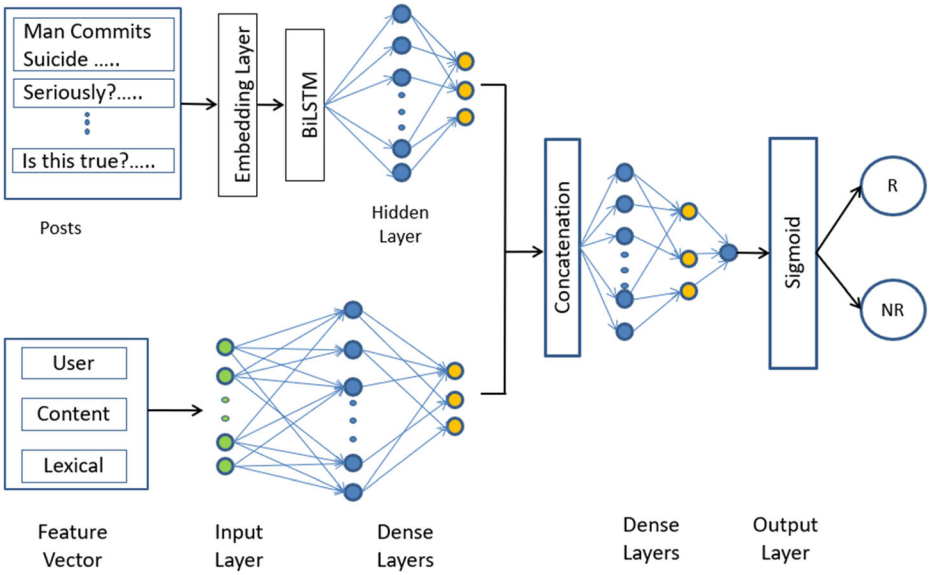


Fig. 9 Proposed architecture of BiLSTM_UCL model

dense layer and the sigmoid in the output layer. Figure 10 represents a summary of the BiLSTM_UCL model.

4 Experimental results

This section presents the dataset used, baseline approaches, experimental environment, evaluation metrics and experimental analysis on various deep learning models.

4.1 Dataset

The proposed method is evaluated on a real-world and benchmarked dataset of Twitter. Earlier, Table 7 explains the statistics of collected data and benchmarked datasets. The actual data collected from Twitter is significantly less as compared to the benchmarked dataset. Therefore, we have combined benchmarked and real-world data to get an extended dataset. Table 7 shows the details of benchmarked and extended real-world datasets. Thus, original benchmarked data get grown event-wise by 7% and post-wise by 32%. Table 7 shows the actual data size used in the experimental evaluation. Data availability after extraction varies in different papers because few posts are not available at the time of data extraction from Twitter. We have split the dataset as 70% for training, 20% for testing and 10% for validation.

4.2 Baseline approaches

Following baseline, algorithms are identified to compare and evaluate the proposed deep learning models.

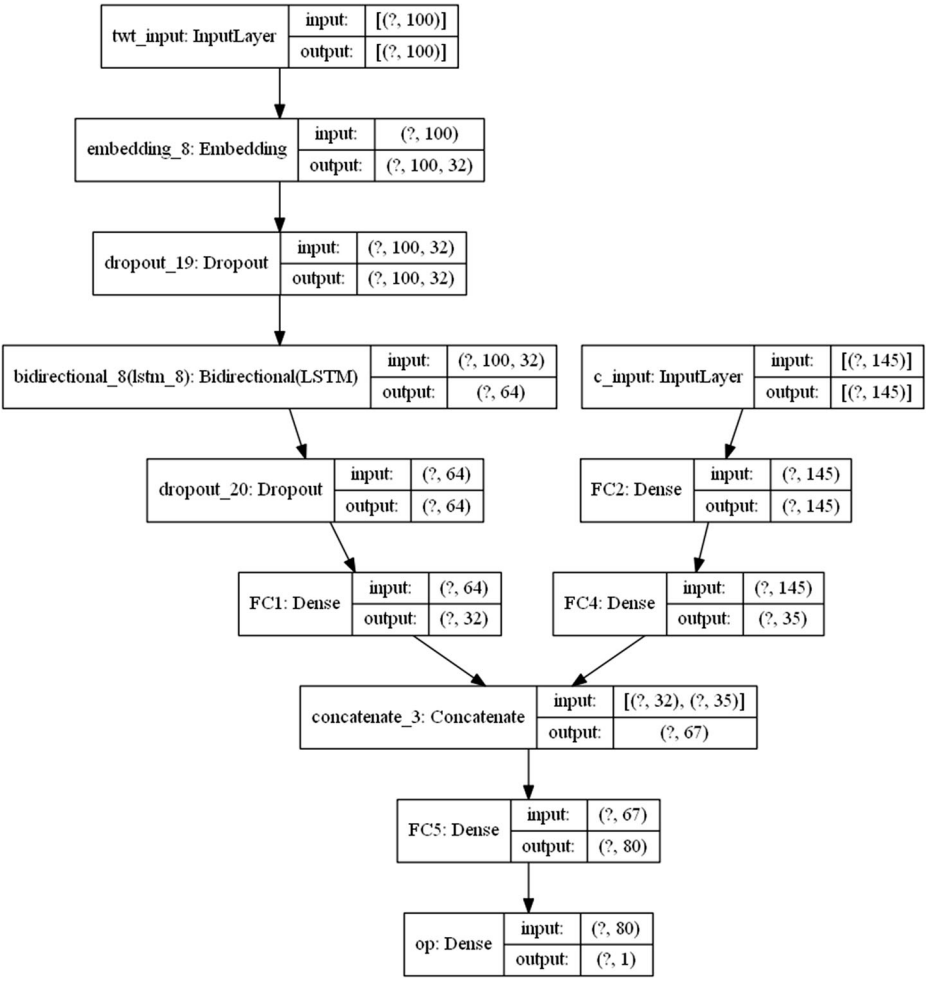


Fig. 10 BiLSTM_UCL model summary

GRU-II [25] experimented with different variants of RNN on time-series posts and revealed that GRU with 2-layers presents good performance.

Table 7 Details of benchmarked and extended real-world dataset

Name	Benchmarked	Real-world-extended
Total Events	986	1056
Total Rumor Events	498	549
Total Non-Rumor Events	489	507
Total Posts	267,708	353,268
Total Rumored Posts	104,920	152,129
Total Non-Rumored Posts	162,788	201,139
Minimum Posts Per Event	2	2
Maximum Posts Per Event	2702	10,304

HAS-BLSTM [17] used a hierarchical attention model for social information with three hierarchy levels as word, post and subevent and utilized the BiLSTM model for rumor detection.

CAMI [42] discover that the CNN model can extract the significant features from post sequence and such models are suitable for early detection of rumors.

CSI [27] proposed a model for fake news classification using features based on the text of an article, the temporal characteristics of user reply and origin users broadcasting it. They combined the RNN model with a deep neural network (DNN) by integrating features from all three categories to get a more accurate rumor classification.

4.3 Experimental setup and evaluation metrics

The environmental setup used for implementation includes a scientific python development environment as Spyder-anaconda, tweepy library to access Twitter data and Keras with Tensorflow for deep learning. For the GPU environment, we have used the Google Colab cloud service. We have used tweepy API for collecting real-world data from Twitter.

For evaluation metrics, we adopted Accuracy, Precision, Recall and F1 scores for a comprehensive evaluation are defined in Eqs. (1), (2), (3), (4). The confusion matrix summarizes predicated results over actual results, as shown in Fig. 11, where R stands for rumor and NR for nonrumor. Accuracy is a fraction of correct predictions overall predictions. The quantity of accurate positive results divided by the quantity of positive results predicted by the classifier is called Precision. The recall is the quantity of correct positive results divided by the amount of all relevant samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

		Predicted Class	
		Positive (R)	Negative (NR)
Actual Class	Positive (R)	**True Positive (TP)**	**False Negative (FN)**
	Negative (NR)	**False Positive (FP)**	**True Negative (TN)**

Fig. 11 Confusion matrix for rumor detection

$$Recall = \frac{TP}{(TP + FN)} \tag{3}$$

$$F1 = \frac{2*Recall*Precision}{Recall + Precision} \tag{4}$$

4.4 Experimental analysis

This section explains the optimal hyperparameters used to set up different models in the research and experimental analysis. Table 8 shows the optimal hyperparameters used in the experiment, which involves the parameters used in various deep learning models, activation function, loss function.

The performance of various models is evaluated to finalize the proposed hybrid model. We have used binary_crossentropy as the loss function and Adagrad as the optimizer. The models are trained with a batch size of 32. However, the hyperparameters used for Adagrad are learning rate as 1e-1 and epsilon as 1e-07. Early stopping with the patience of 3 and drop out of 0.5 is used to avoid the overfitting of the model. Due to the early stopping number of epochs varies from 10 to 50, whereas the models are evaluated for 100 epochs. The accuracy of a model is verified with the learning curve of accuracy and loss to training and validation data. Figure 12 shows the learning curve of accuracy and loss for the BiLSTM_UCL model.

Initially, PCA applied on lexical features and given as input of 125 principal components to MLP called as Lex_PCA model shows the accuracy of 91%. UCL_PCA is an MLP model that takes 145 features as an input, where 12 features from content-based, 8 from the user group and 125 features from the lexical category after applying PCA. The model trained with UCL features where UCL stands for User-Content-Lexical, which shows an accuracy of 93%. The BiLSTM model considers posts with word embedding as input to the bidirectional LSTM model and offers 95% accuracy. This model has 5 dense layers in MLP. The third model, BiLSTM_USL, combines the output of the previous two models and shows an accuracy of 97%.

Table 9 compares experimental results on various deep learning models for real-world and benchmarked datasets. Here, the actual data collected from Twitter is significantly less than the

Table 8 Details of optimal hyperparameters

Parameter name	Value of parameter
Vocabulary size	1000
Sequence length	100
Dropout	0.5
Adagrad Learning rate	0.001
Epoch	100
Number of Dense layers	5
Batch Size	32
Loss function	binary_crossentropy
Activation Function	Relu
Activation Function in Output Layer	Sigmoid
Loss Function	Adagrad

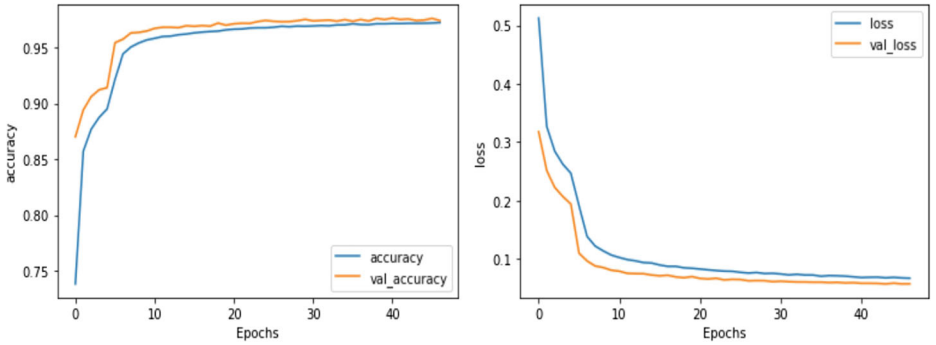


Fig. 12 The learning curve of accuracy and loss for the BiLSTM_UCL model

benchmarked dataset. Therefore the proposed method is evaluated on the extended dataset. Table 9 expresses a significant improvement in precision and recall value from 0.90 to 0.96 for rumor events. Also, it can be observed that results are slightly similar on a benchmarked and extended dataset.

From Table 10, it can be observed that combining *BiLSTM_Embed* and *UCL_PCA* model improves the accuracy, which shows the accuracy of 97% for the *BiLSTM_UCL* model. CSI [27] shows the highest accuracy of 89% from the previous work, which the Lex_PCA model shows. The experimental results shown in Table 10 are the values taken from the results mentioned in the related research paper for similar datasets and methodology. The experimental results of the proposed method are tested on a benchmarked dataset. Figure 13 presents the comparison of the proposed model with existing models concerning accuracy. Figure 14 shows the improvement in all implemented models where accuracy improves from 89% to 97%. The results from overall experiments conclude that the proposed BiLSTM_UCL model shows an excellent enhancement in the accuracy of rumor detection.

4.5 Discussion

Table 9 presents the performance of proposed models on two datasets of different sizes and shows similar performance in terms of precision, recall and F1 score. Deep learning models are most suitable on large dataset and experiments in this research demonstrates that proposed method is scalable.

Table 9 Performance of proposed models on benchmarked and real-world dataset

Sr. no.	Models	Class	Benchmarked dataset			Real-world extended dataset		
			Precision	Recall	F1	Precision	Recall	F1
1.	Lex_PCA	NR	0.88	0.94	0.91	0.91	0.94	0.92
		R	0.90	0.80	0.85	0.91	0.87	0.89
2.	UCL_PCA	NR	0.92	0.97	0.95	0.92	0.97	0.95
		R	0.95	0.87	0.91	0.96	0.89	0.93
3.	BiLSTM_Embed	NR	0.95	0.97	0.96	0.96	0.96	0.96
		R	0.95	0.93	0.94	0.95	0.94	0.95
4.	BiLSTM_UCL	NR	0.97	0.97	0.98	0.97	0.98	0.98
		R	0.96	0.98	0.97	0.96	0.97	0.97

Table 10 Experimental results

Sr. no.	Models	Class	Accuracy	Precision	Recall	F1
1	GRU-II [25]	R	0.881	0.851	0.95	0.898
		NR		0.93	0.8	0.86
2	HSA-BLSTM [17]	R	0.844	0.87	0.67	0.757
		NR		0.73	0.899	0.805
3	CAMI [42]	R	0.777	0.744	0.848	0.793
		NR		0.82	0.705	0.758
4	CSI [27]	–	0.892	–	–	0.83
5	Lex_PCA	R	0.89	0.90	0.80	0.85
		NR		0.88	0.94	0.91
6	UCL_PCA	R	0.93	0.95	0.87	0.91
		NR		0.92	0.97	0.95
7	BiLSTM_Embed	R	0.95	0.95	0.93	0.94
		NR		0.95	0.97	0.96
8	BiLSTM_UCL	R	**0.97**	0.96	0.98	0.97
		NR		0.97	0.97	0.98

The computational complexity of NN models are analyzed in terms of a multiplication per recovered output by Freire et al. in [15]. Parameters considered for MLP are [batch, n_s, n_i], where the *batch* is the batch size, n_s is memory size and is n_i features count. Considering the Lex_PCA, an MLP based model having 125 features and 3 dense layers with the number of neurons in each dense layer as nd_1, nd_2 and nd_3, then the computational complexity (CC) of the MLP model can be given in Eq. (5) as:

$$CC_{MLP} = \underbrace{n_s n_i n d_1}_{a} + \underbrace{n d_1 n d_2 + n d_2 n d_3}_{b} + \underbrace{n d_3 n_o}_{c} \tag{5}$$

Where a, b and c represent contributions from the input, hidden and an output layer of MLP.

The limitation of this research is the real-world data collected is relatively less; therefore, we have extended the benchmarked dataset by combining collected real-world data with the existing dataset. Although the results are evaluated on the benchmarked dataset and baseline algorithms, the dataset is not entirely available due to Twitter's policy. In the previous work, methods are evaluated on Sina Weibo and Twitter dataset. However, this research assessed only the Twitter dataset and focused on only English posts.

5 Conclusions

The diffusion of rumors and their impact on society is a massive problem in current social networks. To combat this, we have come up with rumor detection using post-wise essential features. Compared to the previous work, where more importance was given to text and temporal features and showed a moderate accuracy, this paper focused on text, user, content-based, and lexical category features. The BiLSTM with word embedding and MLP model with various features improves the accuracy. The experimental results compared with the state-of-the-art approaches and show a good improvement in the accuracy. This research also fetched

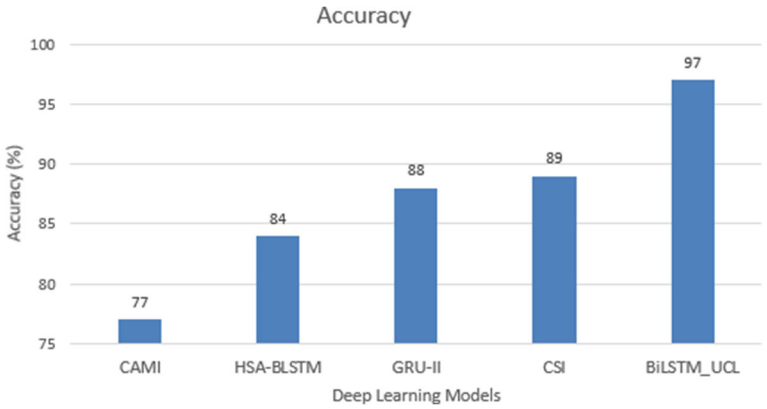


Fig. 13 Comparison of the proposed model with existing models

real-world data from Twitter and evaluated the experiment on both real-world and benchmarked datasets. Lexical features with PCA components show an accuracy of 89%. The continuous improvements in the proposed models help finalize the combined model of BiLSTM_UCL with significant features from selected categories, demonstrating accuracy of 97%.

In the future, we are planning to implement the same aspect with temporal features and attention models. The attention model can be utilized to identify the significant attributes from lexical features and will help to replace the feature selection method. In temporal characteristics, the word count of each post can be used to convert variable-length posts into fixed length posts. Also, future research may utilize multimedia-based features (such as image count, multimedia content present, is_real_image in the post? and video link present) to check the real news.

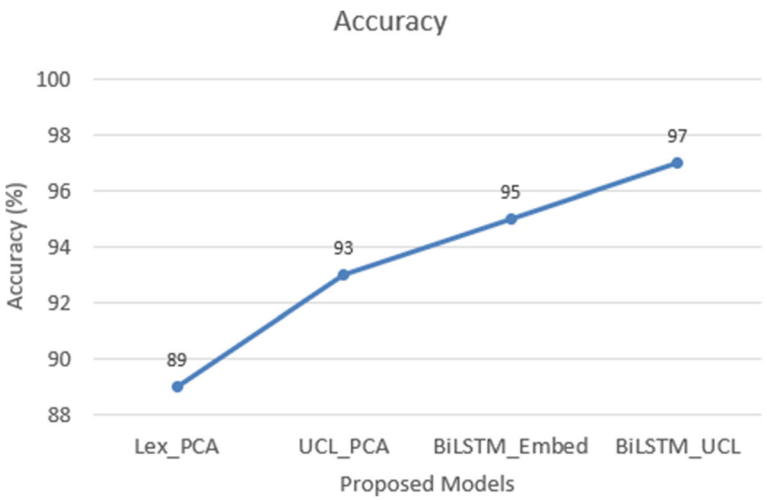


Fig. 14 Improvement in accuracy throughout the proposed models

Declarations

Conflict of interest The research work presented here has not been submitted to, nor under review, at another journal or other publishing venue. All authors have participated in conception and design, analysis and interpretation of the data, drafting the article or revising it critically for important intellectual content, and approval of the final version.

References

1. After COVID-19 vaccine, blood or plasma donation not allowed. [Online]. Available: https://www.politifact.com/factchecks/2021/may/04/tiktok-posts/no-red-cross-isnt-warning-vaccinated-people-not-do/. Accessed 5 April 2021
2. Al-Sarem M, Boulila W, Al-Harby M, Qadir J, Alsaeedi A (2019) Deep learning-based rumor detection on microblogging platforms: a systematic review. IEEE Access 7:152788–152812
3. Asghar MZ, Habib A, Habib A, Khan A, Ali R, Khattak A (2019) Exploring deep neural networks for rumor detection. Journal of Ambient Intelligence and Humanized Computing:1–19
4. Bill Gates told us about the coronavirus in 2015 (n.d.) [Online]. Available: PolitiFact | Bill Gates warned in 2015 that we were unprepared for an infectious virus. Accessed 5 April 2021
5. Boididou C, Middleton SE, Jin Z, Papadopoulos S, Dang-Nguyen DT, Boato G, Kompatsiaris Y (2018) Verifying information with multimedia content on twitter: a comparative study of automated approaches. Multimed Tools Appl 77(12):15545–15571. https://doi.org/10.1007/s11042-017-5132-9
6. Bondielli A, Marcelloni F (2019) A survey on fake news and rumour detection techniques. Inf Sci 497:38–55
7. Castillo C, Mendoza M, Poblete B (2011) Information credibility on twitter. In: Proceedings of the 20th international conference on world wide web - WWW '11, p 675. https://doi.org/10.1145/1963405.1963500
8. Chen T, Li X, Yin H, Zhang J (2018) Call attention to rumors: deep attention based recurrent neural networks for early rumor detection. In: Pacific-Asia conference on knowledge discovery and data mining, pp 40–52
9. Chen W, Zhang Y, Yeo CK, Lau CT, Lee BS (2018) Unsupervised rumor detection based on users' behaviors using neural networks. Pattern Recogn Lett 105:226–233. https://doi.org/10.1016/j.patrec.2017.10.014
10. COVID-19 killed fewer people than the flu. [Online]. Available: https://www.politifact.com/factchecks/2021/apr/27/facebook-posts/no-covid-19-hasnt-killed-fewer-people-flu/. Accessed 5 April 2021
11. COVID-19 Vaccine Cause Herpes (2021) [Online]. Available: https://www.snopes.com/fact-check/covid-19-vaccine-herpes/. Accessed 5 April 2021
12. Facebook Social Network (n.d.) [Online]. Available: https://facebook.com
13. Fact Checking website (n.d.) [Online]. Available: https://www.factcheck.org/
14. Fast E, Chen B, Bernstein MS (2016) Empath: understanding topic signals in large-scale text. In: Proceedings of the 2016 CHI conference on human factors in computing systems, pp 4647–4657
15. Freiredecarvalhosouza PJ, Osadchuk Y, Spinnler B, Napoli A, Schairer W, da Costa NMS, Prilepsky J, Turitsyn SK (2021) Performance versus complexity study of neural network equalizers in coherent optical systems. J Lightwave Technol 39:6085–6096
16. Gargling with salt water or Vinegar 'eliminate' the COVID-19 coronavirus from the throat (n.d.) [Online]. Available: Will Gargling with Salt Water or Vinegar 'Eliminate' the COVID-19 Coronavirus? | Snopes.com
17. Guo H, Cao J, Zhang Y, Guo J, Li J (2018) Rumor detection with hierarchical social attention network. In: Proceedings of the 27th ACM international conference on information and knowledge management, pp 943–951
18. Jogalekar NS, Attar V, Palshikar GK (2020) Rumor detection on social networks: a sociological approach. In: 2020 IEEE international conference on big data (big data), pp 3877–3884. https://doi.org/10.1109/BigData50022.2020.9378149
19. Kaliyar RK, Goswami A, Narang P (2021) FakeBERT: fake news detection in social media with a BERT-based deep learning approach. Multimed Tools Appl 80(8):11765–11788
20. Kotteti CMM, Dong X, Qian L (2020) Ensemble deep learning on time-series representation of tweets for rumor detection in social media. Appl Sci 10(21):7541
21. Kumar S, Asthana R, Upadhyay S, Upreti N, Akbar M (2019) Fake news detection using deep learning models: A novel approach. Transactions on Emerging Telecommunications Technologies:e3767
22. Li X, Lu P, Hu L, Wang X, Lu L (2021) A novel self-learning semi-supervised deep learning network to detect fake news on social media. Multimedia Tools and Applications:1–9

23. Lin X, Liao X, Xu T, Pian W, Wong K-F (2019) Rumor detection with hierarchical recurrent convolutional neural network. In: CCF international conference on natural language processing and Chinese computing, pp 338–348
24. Liu Y, Wu YB (2018) Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. AAAI
25. Ma J, Gao W, Mitra P, Kwon S, Jansen BJ (2016) Detecting rumors from microblogs with recurrent neural networks detecting rumors from microblogs with recurrent neural networks. In: Proceedings of the 25th international joint conference on artificial intelligence (IJCAI 2016), July, pp 3818–3824
26. Ma J, Gao W, Wong K-F (2019) Detect rumors on twitter by promoting information campaigns with generative adversarial learning. In: The World Wide Web Conference, pp 3049–3055
27. Ruchansky N, Seo S, Liu Y (2017) Csi: a hybrid deep model for fake news detection. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp 797–806
28. Rumor Detection Public Dataset (n.d.) [Online]. Available-http://alt.qcri.org/~wgao/data/rumdect.zip
29. Shao C, Ciampaglia GL, Varol O, Yang K-C, Flammini A, Menczer F (2018) The spread of low-credibility content by social bots. Nat Commun 9(1):1–9
30. Sharaff A, Nagwani NK (2020) ML-EC2: an algorithm for multi-label email classification using clustering. International Journal of Web-Based Learning and Teaching Technologies (IJWLTT) 15(2):19–33
31. Sharaff A, Nagwani NK, Dhadse A (2016) Comparative study of classification algorithms for spam email detection. In: Emerging research in computing, information, communication and applications. Springer, pp 237–244
32. Sharaff A, Khurana S, Cheepurupalli K, Sahu T (2020) Personalized recommendation system with user interaction based on LMF and popularity model. In: 2020 international conference on system, computation, automation and networking (ICSCAN), pp 1–6
33. Shelke S, Attar V (2019) Source detection of rumor in social network – a review. Online Social Networks and Media 9:30–42. https://doi.org/10.1016/J.OSNEM.2018.12.001
34. Shelke S, Attar V (2020) Origin identification of a rumor in social network. In: Cybernetics, cognition and machine learning applications. Springer, pp 89–96
35. Shu K, Mahudeswaran D, Wang S, Lee D, Liu H (2020) Fakenewsnet: a data repository with news content, social context, and spatiotemporal information for studying fake news on social media. Big Data 8(3):171–188
36. Social network statistics. [Online]. Available: https://backlinko.com/social-media-users. Accessed 5 April 2021
37. Song C, Yang C, Chen H, Tu C, Liu Z, Sun M (2019) CED: credible early detection of social media rumors. IEEE Trans Knowl Data Eng
38. Srinivasarao U, Sharaff A (2021) Email sentiment classification using lexicon-based opinion labeling. In: Intelligent computing and communication systems. Springer, pp 211–218
39. Tchakounté F, Calvin KA, Ari AAA, Mbogne DJF (2020) A smart contract logic to reduce hoax propagation across social media. Journal of King Saud University-Computer and Information Sciences
40. Twitter Social Network (n.d.) [Online]. Available: https://twitter.com
41. van der Lee C, van den Bosch A (2017) Exploring lexical and syntactic features for language variety identification. Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial), pp 190–199
42. Yu F, Liu Q, Wu S, Wang L, Tan T (2017) A convolutional approach for misinformation identification. In: Proceedings of the 26th international joint conference on artificial intelligence, pp 3901–3907
43. Zubiaga A, Aker A, Bontcheva K, Liakata M, Procter R (2018) Detection and resolution of rumours in social media: a survey. ACM Computing Surveys (CSUR) 51(2):32