SN Computer Science (2020) 1:143
https://doi.org/10.1007/s42979-020-00165-4

ORIGINAL RESEARCH

Fake News Detection Using a Blend of Neural Networks: An Application of Deep Learning

Aman Agarwal[1] · Mamta Mittal[2] · Akshat Pathak[1] · Lalit Mohan Goyal[3]

Received: 29 March 2020 / Accepted: 14 April 2020

Abstract
Fake news and its consequences carry the potential of impacting different aspects of different entities, ranging from a citizen's lifestyle to a country's global relations, there are many related works for collecting and determining fake news, but no reliable system is commercially available. This study aims to propose a deep learning model which predicts the nature of an article when given as an input. It solely uses text processing and is insensitive to history and credibility of the author or the source. In this paper, authors have discussed and experimented using word embedding (GloVe) for text pre-processing in order to construct a vector space of words and establish a lingual relationship. The proposed model which is the blend of convolutional neural network and recurrent neural networks architecture has achieved benchmark results in fake news prediction, with the utility of word embeddings complementing the model altogether. Further, to ensure the quality of prediction, various model parameters have been tuned and recorded for the best results possible. Among other variations, addition of dropout layer reduces overfitting in the model, hence generating significantly higher accuracy values. It can be a better solution than already existing ones, viz: gated recurrent units, recurrent neural networks or feed-forward networks for the given problem, which generates better precision values of 97.21% while considering more input features.

Keywords News headlines · Artificial intelligence · Convolutional neural networks (CNN) · Recurrent neural networks (RNN) · Word embedding

Introduction

Data, rising to become the wealthiest asset anyone can own, have to be transferred and shared and become even more important when data become information. One of the most common information sharing method is via news and articles available both in physical and digital form. With genuine information helping to make humans a more evolved species, fake news is ruining the whole purpose of it. The most visible consequence being the political one, fake news has led to manipulation of public ideologies and opinions about their democracies and governments. It leads to polarising the society during political events and elections, and hence breaking a nation further apart.

As rightly said, fake news is not new, with further implications and consequences, fake news can lead to breakdown and failure of biggest economies of the world, using mass manipulation and prove to be one of the most catastrophic "digital wildfire". Apart from political influences, this fake news can and have led to personal defamations, creating false perspectives and inciting the mass against several issues. As easily people accept and share this news, it is even easier for the sources to create it. Fake news is the greatest threat to our so-called freedom of media, apart from distortion and corrupting ideologies, it has also led to tangible consequences, like cybercrime, phishing, cyber-attacks and the list goes on. It is almost impossible to make public aware about such disasters, what does seem possible is eradicating the root cause [1, 2].

This article is part of the topical collection "Deep learning approaches for data analysis: A practical perspective" guest edited by D. Jude Hemanth, Lipo Wang and Anastasia Angelopoulou.

✉ Lalit Mohan Goyal
lalitgoyal78@gmail.com

1 Thapar Institute of Engineering and Technology, Patiala, Punjab 147004, India

2 Department of CSE, G. B. Pant Government Engineering College, New Delhi, India

3 Department of CE, J. C. Bose University of Science and Technology, YMCA, Faridabad, India

Published online: 25 April 2020

To address this issue, a system or process should be proposed and implemented, which labels or grades a given news article or piece on a defined scale and thus giving the reader an idea about the credibility of it. The labelling, if done manually, will be outdone by the number of articles and news published in an hour, thus generating a need of an automated and accurate labelling. In this paper, authors have proposed the labelling to be done into two categories, fake and genuine (reliable and unreliable). The problem statement comprises of taking a news article as input, which includes both title and text, with output as one of the two labels, fake or genuine. The proposed model contributes to the solution by providing a system which will make all able to identify the nature of the news, one is reading, with benchmark accuracy. Once this is achieved, it will ultimately lead to curbing the creation of fake news, as readership and reach of such news will decrease exponentially, leaving no motive for the sources, eradicating the root cause.

Deep learning, which has caught eyes and ears recently of almost everyone in this field, is the best suited concept to tackle the problem. This methodology has found its use in the field of recommender systems, image recognitions, biometrics, audio/visuals classification and aided the security and management of the industry in a less robust and more predictive manner. Using its nature of self-learning, the feature maps have given deep learning an exceptional upper hand in comparison with other methods of statistical modelling and learning.

The temporal aspects of news article were considered and worked upon by multiple researchers, and as discussed in the following section, it is implemented and accomplished using RNN architecture. A major drawback faced due to RNN is the cost of computation, which in this case increased dramatically due to large text datasets. To deal with the above-branched problem, the authors used CNN for extracting features from text and then processed the generated features with RNN way of learning. This way, lower computation cost is achieved, with extraction of better feature maps that were absent in solely RNN-based models. The datasets were studied and looked upon and selected from a competitive online research Kaggle,[1] due to factors like size and credibility. Abundance of plug-ins are already available to rate a news article as fake or real, making both datasets and labels easily available. The authors have used a dataset from the aforementioned competition that consisted of 20,800 news articles (both title and body) for training purpose and 5200 news articles for testing purpose. The following paper focuses on already existing solutions for the given problems, followed by the methodology and architecture of the proposed model and surmised with their comparison.

[1] https://www.kaggle.com/c/fake-news/data.

Related Works

There has been a lot of studies, research and implementation for prediction and detection of fake news, all over the globe. The authors of this paper are inspired and more inclined by the following remarkable conclusions made by the related work available. In [3], the authors have classified fake news using multiple models and techniques, namely, logistic regression, feed-forward network, RNN(Vanilla), gated recurrent units (GRUs), long short-term memory (LSTMs), bidirectional LSTM, CNN with max pooling and CNN with both max pooling and Attention. They compared all these models on parameters such as precision, recall and F1 score, with results indicating best recall and F1 score by the GRUs, while the best precision using attention-based CNN. This validates the experiment to blend features of both RNN and CNN for the task in hand, considering the importance of context and previous knowledge.

In [4], the authors have used different types of data, including the ones that include article context, the ones that include social context and the ones that includes both. Comparing the results of the three, using both article and social context was found most useful. Furthermore, in [4], the dataset is used on three machine learning techniques, that include, support vector machine (SVM), logistic regression (LR) and Naïve Bayes classification (GNB), which concluded in SVMs and LRs having similar and better results than GNB technique.

In [5], the subject and creator of a news article is taken in consideration too, with a model of their own, referred as deep diffusive network model, which is based on RNN and GRUs as well, complemented by regularisation techniques, it further validates the advantages of RNN.

The authors of [6] made a special change with dataset collection and usage. The dataset was crowd sourced using mainstream sources under six domains, and then fake news was created using Amazon Mechanical Turk (AMT). The second database related to celebrity news was made available from internet. Furthermore, five linguistic features were extracted (*N*-grams, Punctuation, Psycholinguistic features, Readability and Syntax), and different permutations were experimented using linear SVM classifier. In [7], the incorporation of three aspects, text, response the article gets and the source is regarded important. Following this a CSI model (collect, score and integration), based again on RNN is trained and compared against the traditional ones, with allowance to separate the prediction on users and articles. The results suggested an integrated use of linguistic, syntactic and semantic features. Mittal et al. [8–11] have presented the role of machine learning in predicting Crime rates, air quality as well as different data mining techniques exists and how they can be used in different domains. Shastri et al. [12] have presented artificial intelligence technique for prediction of stock market.

Major observations were drawn by the patent issued by Volkova [13], in which several permutations of CNN, number of epochs, type of input dataset, along with observations recorded by using RNN separately. The data used in some observations also consisted images of so-called suspicious tweets (clickbait), and hence is more general and not specific to fake news. In another patent by Lee and Son [14], determinations of fake news are based on collective intelligence, which uses a suspicion index of a social network service (SNS) which is the source, manipulated by the suspicious users of the given SNS, which when beyond a specific threshold value, determines the article as suspicious.

Materials and Methods

The technique implemented by the authors of this paper tackles the problem of fake news from a purely Natural Language Processing[2] perspective. The proposed work is on the classification of articles into fake or real, not taking into consideration their sources. The used dataset (Kaggle fake news dataset) comprises of the columns: author, text and title among others. A combination of title and text is used in the model because of the absence of background on the authors. A possible improvement in practical applications may be to verify the news source itself, scraping data about the sites and finding out which sources are more likely to spread fake news.

In the absence of these source checks, however, it is determined that the most reliable way to determine fake news is to look at the common linguistic features across the source's stories, including sentiment, complexity and structure. For example, fake-news outlets were found to be more likely to use language that is hyperbolic, subjective and emotional. The text is pre-processed and made into an embedding layer using embedding matrix from pre-trained GloVe embedding [15]. GloVe stands for "Global Vectors for Word Representation". The sequential model used in the proposed methodology consists of the following:

- Two Convolutional1D layers
- Max pooling layers
- LSTM layers.

The model thus built is trained and used to predict the nature of the news, observing and analysing its linguistic features.

[2] https://en.wikipedia.org/wiki/Natural_language_processing.

Data Pre-processing Using NLTK and Tokenizer

Data pre-processing is an important step here, as in most NLP applications. The text and the title of the article are concatenated, followed by removal of stop words, tokenization and lemmatization of text. While tokenizing, a maximum of 50,000 words is considered, which is the vocabulary size. All textual sequences are then converted to numerical sequences and padded or trimmed to a maximum sequence length of 300.

Using Word Embedding

Word embeddings[3] are a group of natural language processing techniques that aim to map semantic meaning and relationships into a geometric space. For this, numeric vectors are associated with all words in the dictionary, such that the distance (e.g. cosine distance or L2 distance) between any two vectors would provide us an idea of the semantic relationship between the two words. Embedding space refers to the space formed by the collection of these word vectors. For example, words like "mango" and "zebra" are very different semantically, therefore their associated vectors should be very far apart in any reasonable embedding. But "bed" and "sleep" are related words, so they should be found close by in the embedding space.

Figure 1 depicts the visualisation of GloVe embedding and shows the distance map between words with substring 'ap', with the distance between the words proportional to lingual relation.

In an embedding, word representation is done using dense vectors. These vectors signify the projection of the word into a continuous, high dimensional vector space. This comes as an improvement over the earlier used Bag of Words[4] model wherein large sparse vectors of vocabulary size were used as word vectors. These large vectors also gave no information about how two words were interrelated or any other useful information. The words surrounding any word in the text grant is its position within the vector space. Glove embedding is used in the model, along with Keras embedding layer, which is used for training neural networks on text data. This is a flexible layer, used here to load pre-trained GloVe embedding of 100 dimensions. This is a type of transfer learning. The embedding layer is initialised with weights from this GloVe embedding. Since the learned word weights in this model are not to be updated, therefore the trainable attribute for this model is set to be false.

[3] https://en.wikipedia.org/wiki/Word_embedding.

[4] https://en.wikipedia.org/wiki/Bag-of-words_model.

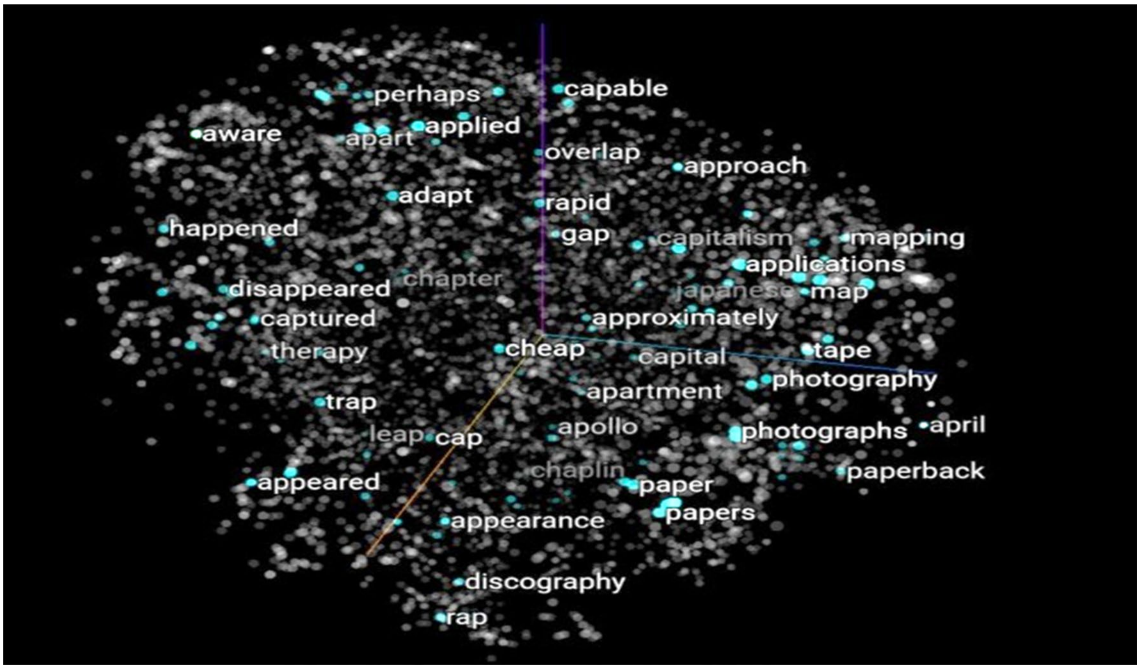

Fig. 1 Visualisation of all the words having substring 'ap' in GloVe embedding using t-SNE and Tensor board

Sequential Model

The built model begins with the embedding layer, followed by convolutional and max pooling layers and then an LSTM layer, i.e. combining CNN and LSTM layers to form a new kind of architecture to benefit from the advantages of both LSTM and CNN. CNN provides the model its ability to learn localised response from time and space related data, while LSTM which specialises in dealing with sequential data can benefit from getting data transformed to a higher level from the convolutional layer. The CNN layer uses pre-trained word vectors from the embedding matrix for learning higher order representations (n-grams). Thereafter, for learning sequential correlations from the higher order sequence representations which were obtained, LSTM was used. The input to this LSTM layer is the feature maps of convolutional layer organised as a series of sequence of window features as shown in Fig. 2. This enabled the construction of the LSTM from sentences transformed into successive window (n-gram) features enabling to remove aspects of variations from the sentences. If LSTM was directly used, it would have operated directly on the sentences.

N-gram Feature Extraction Through Convolution

Any convolution network starts with creating an input layer which is fed to the designed network. The following text depicts feature extraction using convolution, which is then used as input for the next set of neural network.

From [16], let $x_i \in R^d$ be the word vector for ith word in a given sentence of dimension d. Let $x \in R^{L \times d}$ represent the input sentence with length L. Taking k as the length of the filter, and a vector $m \in R^{k \times d}$ is a filter for the convolution function. For each position, say j in the given sentence, a window vector w_j consisting of k consecutive word vectors are formulated, which is represented as:

$$w_j = \left[x_j, x_{j+1}, \ldots, x_{j+k-1}\right] \tag{1}$$

Here, the commas represent row vector concatenation. A filter m combines with the window vectors (k-grams) at each and every position in a way to build a feature map $c \in R^{L-k+1}$; each element c_j of the feature map for window vector w_j is produced as follows:

$$c_j = f\left(w_j \circ m + b\right) \tag{2}$$

where $\circ$ signifies element-wise multiplication, $b \in R$ is a bias term and f is a nonlinear transformation function with possible types like hyperbolic tangent, sigmoid, softmax, leaky ReLU (rectified linear unit), linear, etc. In the given case, ReLU is used. A number of filters are used to produce multiple feature maps. For n filters of equal size, the produced n feature maps can be reordered as feature representations for each window w_j,

$$W = \left[c_1; c_2; \ldots; c_n\right] \tag{3}$$

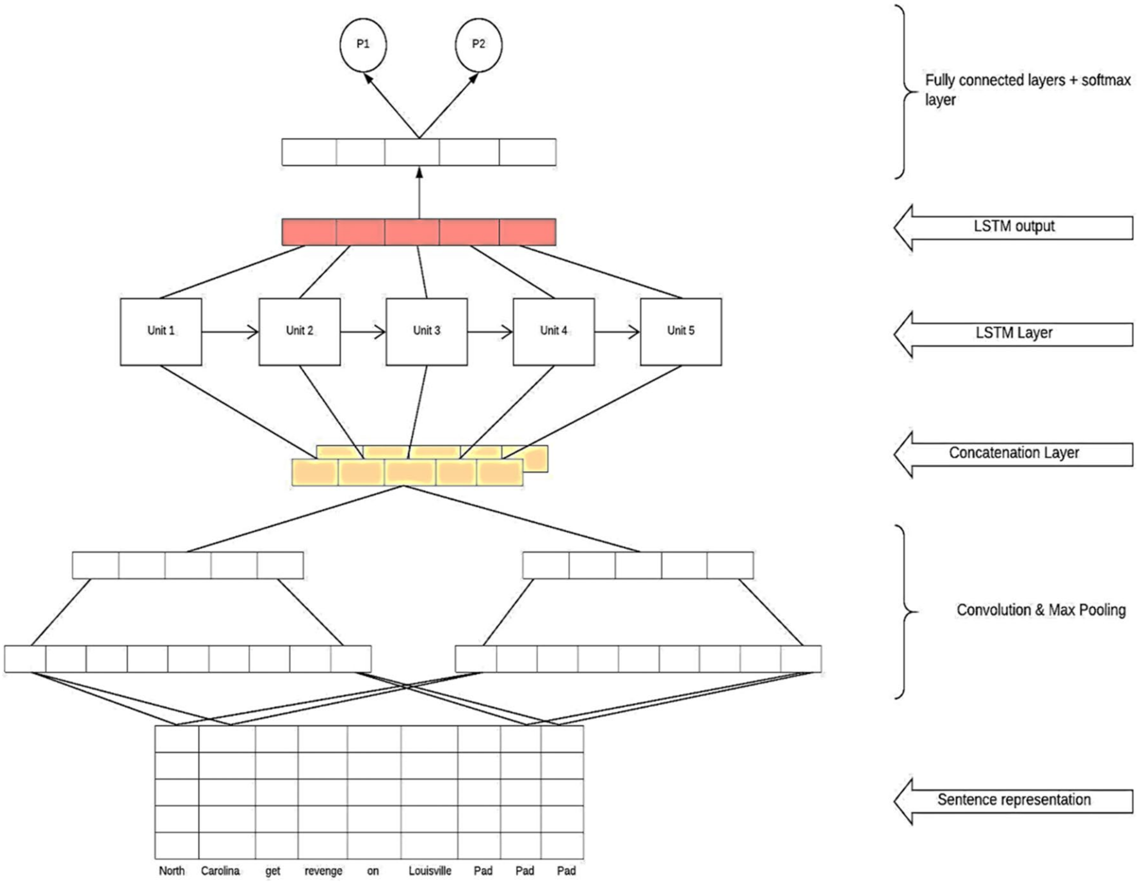

Fig. 2 Representation of the proposed model

Semicolons here signify column vector concatenation and c_i denote the feature map produced by the ith filter. Each row W_j of $W \in R^{(L-k+1)\times n}$ is the new feature representation produced from n features for the window vector at position given by j. These so generated, higher order window representations, are now given as input to the LSTM. For the first convolutional layer, the number of filters is chosen as 32 and filter length as 5. A filter length of 5 means that the filter has dimensions 5 * 100. This is because a 100-dimensional embedding is used. Padding is set as 'same', which means that the output of the convolution will have width same as that of the input, i.e. 100 here. The activation function is set to 'ReLU'. All of these parameters have been decided using hyper-parameter tuning. The next layer is a max-pooling layer of pool size = 2. Similarly, parameters for the second convolutional and max-pooling layers were set.

Long Short-Term Memory Networks

The output from the convolution neural network hence obtained is fed to an LSTM unit for obtaining the latent temporal features of the text, whose architecture and functioning is shown in the following text.

Like all standard RNNs, an LSTM unit has a set of repeat modules for every time step. For each and every time step, the output given by the module is manipulated by a range of gates in R^d as a function controlled by the old hidden state h_{t-1} and by the input at the current time step x_t: the forget gate f_t, the input gate i_t and the output gate o_t, these gates, altogether give a decision on how to update the current memory cell c_t and the current hidden state h_t. d is used to represent the memory dimension in the LSTM and every vector in the given architecture companionate equal dimensions. The expressions for LSTM transition functions, defined in [16], are given by:

$$i_t = \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right) \tag{4}$$

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \tag{5}$$

$$q_t = \tanh\left(W_q\left[h_{t-1}, x_t\right] + b_q\right) \tag{6}$$

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{7}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot q_t \tag{8}$$

$$h_t = o_t \odot \tanh(c_t) \tag{9}$$

σ is the logistic sigmoid function with an output value in the range [0, 1], tan$h$ represents the hyperbolic tangent function with an output value in the range [−1, 1], and $\odot$ signifies the element-wise multiplication. LSTM specialises in dealing with long-term dependencies (no vanishing gradients problem), and thus, it has been chosen to follow the convolutional layer. For the LSTM layer used, the number of units = 100, dropout = 0.2, recurrent dropout = 0.2. Next, after the LSTM layer is a batch normalisation layer. Batch normalisation reduces the amount by which the hidden unit values shift around (covariance shift). Also, batch normalisation allows each layer of a network to learn by itself a little bit more independently of other layers. It also reduces overfitting; therefore, if batch normalisation is used, less dropout is used, which is good as not losing too much information. Next, four fully connected layers are added to the model with suitable number of neurons and activation function as 'ReLU' for all dense layers except the last one, which has 2 neurons and a softmax activation function. Two scenarios of including a dropout with a probability of 20% to prevent overfitting, and excluding it to exploit the dataset, is considered. The entire model is trained by minimising the binary cross-entropy error. Given a training sample x_i and its true label $y_i \in [0, 1]$ and the estimated probabilities $p(y_i) \in [0, 1]$, the error is defined as:

$$H_p(q) = \sum_{i=1}^{N} y_i \cdot \log\left(p(y_i)\right) + \left(1 - y_i\right) \cdot \log(1 - \left(p(y_i)\right))$$

Stochastic gradient descent is employed to learn the model parameters and the optimizer "adam" with metrics set as accuracy is used. Surmising, the text after being tokenized, is projected into a vector space. With onset of training, the features are extracted as a window vector which is passed on to the LSTM unit for its sequential analysis and learning. Thus, model is trained and is tested against the said split of data.

Results and Discussion

In this study and experiment, the dataset was collected from a Kaggle competition and is divided into two sets: 1/5th being testing set and 4/5th being the training set. The model trained consisted of two one-dimensional convolutional layers, a max pooling layer and an LSTM layer. Authors have considered two cases of testing the model, one using a dropout layer and another without it. Setting the dropout as 0.2, the accuracy of the model saw a significant increase of approximately 2%, which evidently justifies the presence of overfitting in the data.

Moreover, the data were pre-processed using a 100-dimensional GloVe embedding, creating a dense vector space for each word instead of traditional Bag of Words, and hence weighing in the relation between similar words. While running both the cases for 15 epochs, and a batch size of 128, accuracy and data loss changes are quite visible, depicted in Fig. 3.

The loss shown is the mean squared error (MSE), which is initially 44.68% due to random weights of the neurons of the hidden layers and gives a minute value of 1.24% by the last epoch. Unlike the other experiments of regression, the proposed model is solely based for classification, having a user defined variable of threshold, which can be further be manipulated by other attributes such as source and author, in the future works. Summary of the proposed model has been presented in Table 1.

The unexpected spike in validation loss in Fig. 4 is a mere cause of 'unlucky' data in a particular batch and epoch and can suggest overfitting, in the corresponding batch, which also resulted in the minima in Fig. 4. Table 2 shows the hyper-tuning of different parameters of a neural network, such as number of dense layers, number of filters, inclusion and value of dropout layer, type of optimiser used. The different combinations were observed and recorded, leading to selection of the best, epochs of which are shown in Fig. 3.

Table 3 shows the confusion matrix of the testing done, with 4160 instances. Given the results of confusion matrix, precision is calculated to be 97.21%, with sensitivity and specificity to be 91.89% and 97.44%, respectively. The model training accuracy is a well-high of 99.54% in the last epoch, competing leader board of the competition from which the dataset was collected.

Surmising the model proposed which includes of blend of two aspects of deep learning, convolutional and recurrent proved to be a better solution than the case when each of them is used separately. It also shows to outdo the other methods such as SVM and GRUs, with the results clearly shown in Table 3 and the plots. With implementing it in multiple scenarios, including and excluding the dropout layer, the model's accuracy had an advantage of ~ 5% using the dropout layer, hence curbing the overfitting present in the dataset. More parameters like number of filters in the Conv1D layer, number of dense layers was varied and recorded given by the blend of CNN and RNN type of learning. For future work, the inclusion of features like source or the author of the article, user response, along with the model proposed, can lead the way towards a state-of-the-art solution to this potentially hazardous "digital wildfire".

```
Epoch 5/15
16640/16640 [==============================] - 29s 2ms/step - loss: 0.0822 - acc: 0.9686 - val_loss:
0.1542 - val_acc: 0.9507
Epoch 6/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0600 - acc: 0.9767 - val_loss:
0.1823 - val_acc: 0.9380
Epoch 7/15
16640/16640 [==============================] - 29s 2ms/step - loss: 0.0420 - acc: 0.9841 - val_loss:
0.8472 - val_acc: 0.8123
Epoch 8/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0338 - acc: 0.9879 - val_loss:
0.2520 - val_acc: 0.9296
Epoch 9/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0221 - acc: 0.9924 - val_loss:
0.2201 - val_acc: 0.9543
Epoch 10/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0255 - acc: 0.9912 - val_loss:
0.2272 - val_acc: 0.9377
Epoch 11/15
16640/16640 [==============================] - 29s 2ms/step - loss: 0.0169 - acc: 0.9945 - val_loss:
0.2700 - val_acc: 0.9478
Epoch 12/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0178 - acc: 0.9938 - val_loss:
0.2055 - val_acc: 0.9563
Epoch 13/15
16640/16640 [==============================] - 29s 2ms/step - loss: 0.0145 - acc: 0.9951 - val_loss:
0.2225 - val_acc: 0.9488
Epoch 14/15
16640/16640 [==============================] - 30s 2ms/step - loss: 0.0186 - acc: 0.9928 - val_loss:
0.3229 - val_acc: 0.9466
Epoch 15/15
16640/16640 [==============================] - 34s 2ms/step - loss: 0.0124 - acc: 0.9954 - val_loss:
0.2428 - val_acc: 0.9498
```

Fig. 3 Variation of loss and accuracy of model while training (model described in Fig. 2)

Table 1 Model Summary

Layer (type)	Output shape	No. of parameters
embedding_1 (Embedding)	(None, 300, 100)	5,000,100
conv1d_1 (Conv1D)	(None, 300, 32)	16,032
max_pooling1d_1	(None, 150, 32)	0
conv1d_2 (Conv1D)	(None, 150, 64)	6208
max_pooling1d_2 (Max pooling)	(None, 75, 64)	0
lstm_1 (LSTM)	(None, 100)	66,000
batch_normalisation_1 (Batch Normalisation)	(None, 100)	400
dense_1 (Dense)	(None, 256)	25,856
dense_2 (Dense)	(None, 128)	32,896
dropout_1 (dropout)	(None, 300, 100)	0
dense_3 (Dense)	(None, 64)	8256
dense_4 (Dense)	(None, 2)	130

Conclusions and Future Scope

In this paper, authors have concentrated solely on the labelled dataset, pre-processed using NLTK library of python and weighed in the relation between words using pre-trained GloVe embedding. GloVe embedding was found to be extremely useful as it provided each word a vector projection which was manipulated by its relation, similarities, dissimilarities with other words in the vocabulary, hence complementing the training process in a much more significant way than what a traditional method of Bag of Words would have done. Furthermore, the fake news prediction and detection is based on feature extraction using both convolutional and recurrent type of neural networks, with former providing better feature extraction in less time and latter aiding the model with its sequential and memory-holding characteristics.

The ongoing works and studies related to fake news detection and prediction include many hackathons and online competitions, in order to make a commercially available and feasible model, which ensures a

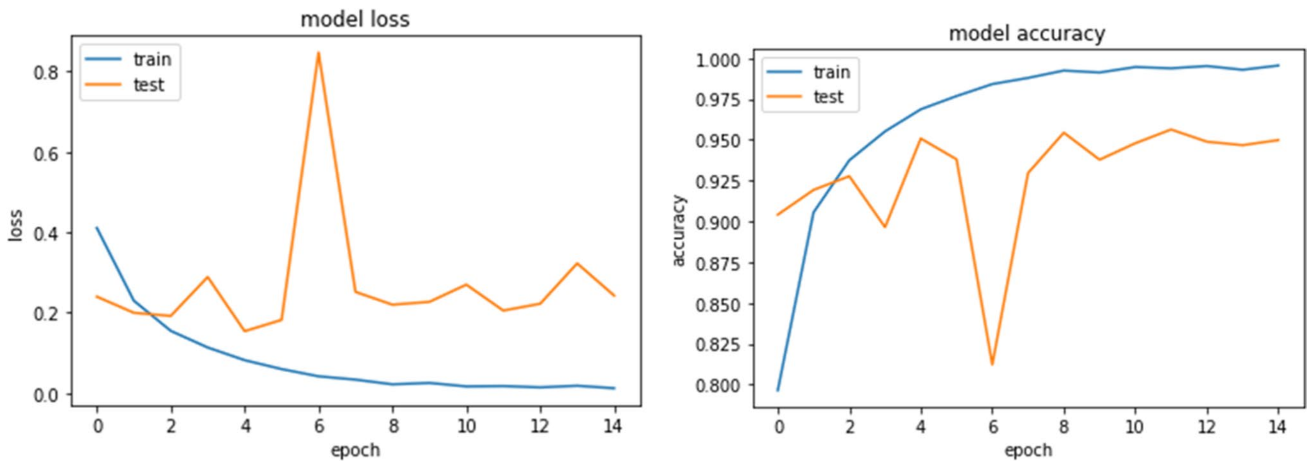

Fig. 4 Data loss and accuracy during training and testing

Table 2 Observation of hyper-tuning of parameters

S. no.	No. of dense layers	No. of filters in Conv1d	Dropout	Train accuracy (%)	Test accuracy (%)	Optimiser
1	6	64,128	0.20	98.34	91.56	SGD with momentum
2	4	32,64	0.20	98.84	92.63	Adam
3	4	64,128	0.20	99.54	94.71	Adam
4	4	64,128	0.25	99.10	93.95	Adam
5	4	64,128	0.00	98.56	88.23	Adam

Table 3 Confusion matrix for training

	Predicted as genuine news	Predicted as fake news
Actually, genuine news	2057	54
Actually, fake news	166	1883

state-of-the-art prediction, as a plug-in or extension in an internet browser. The currently available tool allows you to mark news as authentic or suspicious, but does not give a rating or label beforehand, to depict the credibility, which is required to eradicate the real issue. With enough developments and inclusion of more input attributes in the model, a future tool can look like this, where people can find a score for articles, as well as label an article and rewarded in some form of credits for the same.

Compliance with Ethical Standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Shu K, Sliva A, Wang S, Tang J, Liu H. Fake news detection on social media: a data mining perspective. ACM SIGKDD Explor Newsl. 2017;19(1):22–36.
2. Ajao O, Bhowmik D, Zargari S. Fake news identification on twitter with hybrid cnn and rnn models. In: Proceedings of the 9th international conference on social media and society. 2018. pp. 226–30.
3. Bajaj S. The pope has a new baby! Fake news detection using deep learning. 2018.
4. Shu K, Mahudeswaran D, Liu H. Fake news tracker: a tool for fake news collection, detection, and visualization. Comput Math Organ Theory. 2019;25(1):60–71.
5. Zhang J, Cui L, Fu Y, Gouza FB. Fake news detection with deep diffusive network model. Preprint arXiv:1805.08751. Accessed 22 May 2018.
6. Pérez-Rosas V, Kleinberg B, Lefevre A, Mihalcea R. Automatic detection of fake news. Preprint arXiv:1708.07104. Accessed 23 Aug 2017.
7. Ruchansky N, Seo S, Liu Y. Csi: A hybrid deep model for fake news detection. In: Proceedings of the 2017 ACM on conference on information and knowledge management. 2017. pp. 797–806.
8. Mittal M, Goyal LM, Sethi JK, Hemanth DJ. Monitoring the impact of economic crisis on crime in India using machine learning. Comput Econ. 2019;53(4):1467–85.
9. Mittal M, Goyal LM, Hemanth DJ, Sethi JK. Clustering approaches for high-dimensional databases: a review. Wiley Interdiscip Rev Data Min Knowl Discov. 2019;9(3):e1300.

10. Sethi JK, Mittal M. A new feature selection method based on machine learning technique for air quality dataset. J Stat Manag Syst. 2019;22(4):697–705.
11. Sethi J, Mittal M. Ambient air quality estimation using supervised learning techniques. EAI Endorsed Trans Scalable Inf Syst. 2019;6(22).
12. Shastri M, Roy S, Mittal M. Stock price prediction using artificial neural model: an application of big data. EAI Endorsed Trans Scalable Inf Syst. 2019;6(20).
13. Volkova S, inventor; Battelle Memorial Institute Inc, assignee. Prediction of social media postings as trusted news or as types of suspicious news. United States patent application US 15/886,079. 2018 Dec 20.
14. Lee KM, Son HS, Method and apparatus for spotting fake news using collective intelligence, South Korea Patent KR101869815B1, 2017.
15. Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014. pp. 1532–43.
16. Zhou C, Sun C, Liu Z, Lau F. A C-LSTM neural network for text classification. Preprint arXiv:1511.08630. Accessed 27 Nov 2015.