

N-gram-Enhanced LSTM Architecture for Accurate Fake News Classification

Nusrath Begum Mohammad¹, Nanduri Shankar¹, Akila Gouni¹

¹Department of Computer Science and Engineering

¹Sree Dattha Group of Institutions, Sheriguda, Telangana.

ABSTRACT

The advent of the World Wide Web and the rapid proliferation of social media platforms, such as Facebook and Twitter, have revolutionized the landscape of information dissemination, allowing users unprecedented access to news and updates at their fingertips. While these platforms have empowered news outlets to deliver real-time information and fostered vibrant discussions on various societal issues, they have also become breeding grounds for misinformation and disinformation, commonly referred to as fake news. This troubling phenomenon has gained significant prominence, particularly evident during critical events such as the 2016 US elections, when misleading narratives spread rapidly across digital channels, resulting in detrimental effects on public opinion and behavior.

The impact of fake news extends beyond politics; it infiltrates various domains, including health, science, and finance, where inaccurate information can lead to severe consequences. For instance, the financial markets are particularly vulnerable to the ripple effects of unverified rumors, which can trigger market instability and influence investor behavior. Given the pervasive nature of fake news and its potential to shape societal perceptions, there is a pressing need for automated solutions to accurately classify news articles as real or fake, thereby empowering consumers to make informed decisions about the information they encounter.

This project aims to address the challenges posed by fake news through the development of an automated classification system. Initially, N-gram feature selection is employed to extract the most relevant features from the dataset, allowing the model to capture nuanced patterns and relationships within the textual data. Following this feature extraction, a Long Short-Term Memory (LSTM) model is utilized for the classification task. LSTM networks are particularly well-suited for processing sequential data, making them effective in understanding the context and dependencies present in news articles.

By integrating N-gram feature selection with LSTM-based classification, this work seeks to enhance the accuracy of fake news detection, thereby aiding individuals in identifying trustworthy news sources. Ultimately, the proposed solution aspires to contribute to the broader effort of mitigating the spread of misinformation and fostering a more informed society.

Keywords: social media, fake news, LSTM, N-gram feature selection.

1. INTRODUCTION

1.1 Research Motivation

The proliferation of fake news poses significant challenges to society, impacting various aspects of public discourse and decision-making. The motivation behind the proposed research lies in addressing the pressing need for effective tools to combat the spread of misinformation and disinformation.

One of the primary motivations is to safeguard the integrity of information and promote informed decision-making among the public. In an era where misinformation can spread rapidly and influence

public opinion, there is a critical need for reliable mechanisms to differentiate between genuine news sources and fabricated content.

Moreover, the potential consequences of fake news extend beyond individual perceptions to broader societal implications. Fake news can undermine trust in institutions, sow division within communities, and even pose risks to public safety and security. By developing robust fake news detection systems, researchers aim to mitigate these risks and foster a more informed and resilient society.

Furthermore, the proliferation of fake news has highlighted the limitations of traditional media literacy and fact-checking mechanisms in combating misinformation. In an increasingly digital and interconnected world, automated solutions leveraging advanced technologies offer a promising avenue for effectively identifying and combating fake news at scale.

1.2 Problem Statement

The proliferation of fake news poses a significant challenge to society, undermining the integrity of information and eroding trust in media sources. The primary problem addressed by the proposed research is the need for an automated solution to accurately classify news articles as genuine or fake.

Traditional approaches to fake news detection have relied on manual fact-checking and media literacy efforts, which are often time-consuming and resource-intensive. Moreover, the rapid spread of misinformation on digital platforms makes it challenging for humans alone to keep pace with the volume and velocity of fake news content.

The proposed research seeks to overcome these challenges by leveraging advanced machine learning techniques, specifically focusing on an N-gram feature selection-based LSTM model. The objective is to develop a robust and scalable solution capable of automatically identifying fake news articles with high accuracy.

By extracting optimal features from the dataset using N-gram feature selection and employing LSTM, a type of recurrent neural network well-suited for sequential data analysis, the research aims to enhance the effectiveness of fake news detection. The ultimate goal is to provide users with a reliable tool for discerning between genuine and fabricated news sources in real-time.

2. LITERATURE SURVEY

Saqib Hakak et. al [1] proposed a machine-learning based fake news detection model using a supervised approach. They used the ensemble approach for training and testing purposes consisting of decision tree, random forest, and extra tree classifiers. The aggregation of outputs was done using the bagging approach and compared to the state-of-the-art, our model achieved better results.

Kaliyar et. al [2] propose a BERT-based (Bidirectional Encoder Representations from Transformers) deep learning approach (FakeBERT) by combining different parallel blocks of the single-layer deep Convolutional Neural Network (CNN) having different kernel sizes and filters with the BERT. Such a combination is useful to handle ambiguity, which is the greatest challenge to natural language understanding. Classification results demonstrate that our proposed model (FakeBERT) outperforms the existing models with an accuracy of 98.90%. Somya Ranjan Sahoo et. al [3] proposed a fake news detection approach for Facebook users using machine learning and deep learning classifiers in chrome environment. Our approach analyses both user profile and news content features. In this proposed work, they have developed a chrome extension that uses crawled data extracted by our crawler. Also, to boost

up the performance of chrome extension, they have used deep learning algorithm called Long Short-Term Memory.

Anshika Choudhary et. al [4] proposed a solution to fake news detection and classification. In the case of fake news, content is the prime entity that captures the human mind towards trust for specific news. Therefore, a linguistic model is proposed to find out the properties of content that will generate language-driven features. This linguistic model extracts syntactic, grammatical, sentimental, and readability features of particular news. Language driven model requires an approach to handle timeconsuming and handcrafted features problems in order to deal with the curse of dimensionality problem. Therefore, the neural-based sequential learning model is used to achieve superior results for fake news detection. The results are drawn to validate the importance of the linguistic model extracted features and finally combined linguistic feature-driven model is able to achieve the average accuracy of 86% for fake news detection and classification. The sequential neural model results are compared with machine learning based models and LSTM based word embedding based fake news detection model as well. Comparative results show that features based sequential model is able to achieve comparable evaluation performance in discernable less time.

Aphiwongsophon et. al [5] proposes the use of machine learning techniques to detect Fake news. Three popular methods are used in the experiments: Naive Bayes, Neural Network and Support Vector Machine. The normalization method is important step for cleaning data before using the machine learning method to classify data. The result show that Naive Bayes to detect Fake news has accuracy 96.08%. Two other more advance methods which are Neural Network and Support Vector Machine achieve the accuracy of 99.90%.

Georgios Gravanis et. al [6] proposed a model for fake news detection using content-based features and Machine Learning (ML) algorithms. To conclude in most accurate model, they evaluate several feature sets proposed for deception detection and word embeddings as well. Moreover, they test the most popular ML classifiers and investigate the possible improvement reached under ensemble ML methods such as AdaBoost and Bagging. An extensive set of earlier data sources has been used for experimentation and evaluation of both feature sets and ML classifiers. Moreover, they introduce a new text corpus, the “UNBiased” (UNB) dataset, which integrates various news sources and fulfills 142 several standards and rules to avoid biased results in classification task. Our experimental results show that the use of an enhanced linguistic feature set with word embeddings along with ensemble algorithms and Support Vector Machines (SVMs) is capable to classify fake news with high accuracy.

Umer et. al [7] proposed to employ the dimensionality reduction techniques to reduce the dimensionality of the feature vectors before passing them to the classifier. To develop the reasoning, this work acquired a dataset from the Fake News Challenges (FNC) website which has four types of stances: agree, disagree, discuss, and unrelated. The nonlinear features are fed to PCA and chi-square which provides more contextual features for fake news detection. The motivation of this research is to determine the relative stance of a news article towards its headline. The proposed model improves results by ~4% and ~20% in terms of Accuracy and F1-score. The experimental results show that PCA outperforms than Chi-square and state-of-the-art methods with 97.8% accuracy. Vasu Agarwal et. al [8] discusses the approach of natural language processing and machine learning in order to solve this problem. Use of bag-of-words, n-grams, count vectorizer has been made, TFIDF, and trained the data on five classifiers to investigate which of them works well for this specific dataset of labelled news statements. The precision, recall and f1 scores help us determine which model works best.

Junaed Younus Khan et. al [9] presented an overall performance analysis of 19 different machine learning approaches on three different datasets. Eight out of the 19 models are traditional learning models, six models are traditional deep learning models, and five models are advanced pre-trained language models like BERT. They find that BERT-based models have achieved better performance than all other models on all datasets. More importantly, we find that pre-trained BERT-based models are robust to the size of the dataset and can perform significantly better on very small sample size. They also find that Naive Bayes with n-gram can attain similar results to neural network-based models on a dataset when the dataset size is sufficient. The performance of LSTM-based models greatly depends on the length of the dataset as well as the information given in a news article. With adequate information provided in a news article, LSTM-based models have a higher probability of overcoming overfitting.

Reis et. al [10] presented a new set of features and measure the prediction performance of current approaches and features for automatic detection of fake news. Our results reveal interesting findings on the usefulness and importance of features for detecting false news. Finally, they discuss how fake news detection approaches can be used in the practice, highlighting challenges and opportunities.

Agarwal et. al [11] proposed 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.

143 Jiang et. al [12] evaluated the performance of five machine learning models and three deep learning models on two fake and real news datasets of different size with hold out cross validation. They also used term frequency, term frequency-inverse document frequency and embedding techniques to obtain text representation for machine learning and deep learning models respectively. To evaluate models' performance, they used accuracy, precision, recall and F1-score as the evaluation metrics and a corrected version of McNemar's test to determine if models' performance is significantly different. Then, they proposed our novel stacking model which achieved testing accuracy of 99.94% and 96.05 % respectively on the ISOT dataset and KDnugget dataset. Furthermore, the performance of our proposed method is high as compared to baseline methods. Thus, they highly recommend it for fake news detection.

3. PROPOSED SYSTEM

3.1 Overview

This project implements a Graphical User Interface (GUI) application for detecting fake news using machine learning techniques, specifically utilizing a combination of LSTM (Long Short-Term Memory) neural networks and Random Forest classifiers. Here's an overview of the project:

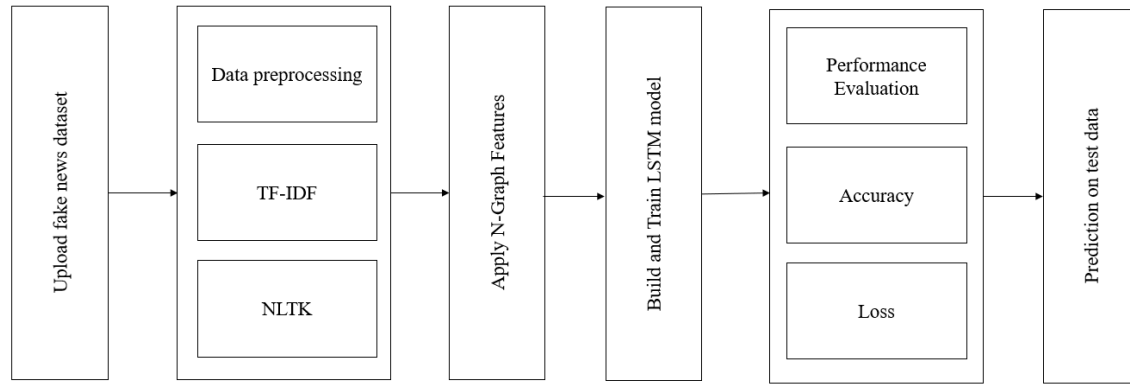


Fig. 1: Block diagram of proposed system.

3.2 TF-IDF Feature extraction

TF-IDF which stands for Term Frequency – Inverse Document Frequency. It is one of the most important techniques used for information retrieval to represent how important a specific word or phrase is to a given document. We have a string or Bag of Words (BOW) and we have to extract information from it, then we can use this approach. The tf-idf value increases in proportion to the number of times a word appears in the document but is often offset by the frequency of the word in the corpus, which helps to adjust with respect to the fact that some words appear more frequently in general. TF-IDF uses two statistical methods, first is Term Frequency and the other is Inverse Document Frequency. Term frequency refers to the total number of times a given term t appears in the document doc against (per) the total number of all words in the document and The inverse document frequency measure of how much information the word provides. It measures the weight of a given word in the entire document. IDF shows how common or rare a given word is across all documents.

TF-IDF can be computed as $tf * idf$

TF-IDF does not convert directly raw data into useful features. Firstly, it converts raw strings or dataset into vectors and each word has its own vector. Then we'll use a particular technique for retrieving the feature like Cosine Similarity which works on vectors, etc.

Terminology

t — term (word)

d — document (set of words)

N — count of corpus

corpus — the total document set

Step 1: Term Frequency (TF): Suppose we have a set of English text documents and wish to rank which document is most relevant to the query, "Data Science is awesome!" A simple way to start out is by eliminating documents that do not contain all three words "Data" is, "Science", and "awesome", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document; the number of times a term occurs in a document is called its term frequency. The weight of a term that occurs in a document is simply proportional to the term frequency.

Step 2: Document Frequency: This measures the importance of document in whole set of corpora, this is very similar to TF. The only difference is that TF is frequency counter for a term t in document d , whereas DF is the count of occurrences of term t in the document set N . In other words, DF is the number of documents in which the word is present. We consider one occurrence if the term consists in the document at least once, we do not need to know the number of times the term is present.

Step 3: Inverse Document Frequency (IDF): While computing TF, all terms are considered equally important. However, it is known that certain terms, such as “is”, “of”, and “that”, may appear a lot of times but have little importance. Thus, we need to weigh down the frequent terms while scale up the rare ones, by computing IDF, an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. The IDF is the inverse of the document frequency which measures the informativeness of term t . When we calculate IDF, it will be very low for the most occurring words such as stop words (because stop words such as “is” is present in almost all of the documents, and N/df will give a very low value to that word). This finally gives what we want, a relative weightage. Now there are few other problems with the IDF, in case of a large corpus, say 100,000,000, the IDF value explodes, to avoid the effect we take the log of idf . During the query time, when a word which is not in vocab occurs, the df will be 0. As we cannot divide by 0, we smoothen the value by adding 1 to the denominator. The TF-IDF now is at the right measure to evaluate how important a word is to a document in a collection or corpus. Here are many different variations of TF-IDF but for now let us concentrate on this basic version.

Step 4: Implementing TF-IDF: To make TF-IDF from scratch in python, let's imagine those two sentences from different document:

first_sentence: “Data Science is the sexiest job of the 21st century”.

second_sentence: “machine learning is the key for data science”.

3.3 Natural Language Toolkit

NLTK is a toolkit build for working with NLP in Python. It provides us various text processing libraries with a lot of test datasets. A variety of tasks can be performed using NLTK such as tokenization, lower case conversion, Stop Words removal, stemming, and lemmatization.

Tokenization

The breaking down of text into smaller units is called tokens. tokens are a small part of that text. If we have a sentence, the idea is to separate each word and build a vocabulary such that we can represent all words uniquely in a list. Numbers, words, etc. all fall under tokens.

Lower case conversion

We want our model to not get confused by seeing the same word with different cases like one starting with capital and one without and interpret both differently. So we convert all words into the lower case to avoid redundancy in the token list.

Stop Words removal

When we use the features from a text to model, we will encounter a lot of noise. These are the stop words like the, he, her, etc... which don't help us and just be removed before processing for cleaner processing inside the model. With NLTK we can see all the stop words available in the English language.

Stemming

In our text we may find many words like playing, played, playfully, etc... which have a root word, play all of these convey the same meaning. So we can just extract the root word and remove the rest. Here the root word formed is called 'stem' and it is not necessarily that stem needs to exist and have a meaning. Just by committing the suffix and prefix, we generate the stems.

Lemmatization

We want to extract the base form of the word here. The word extracted here is called Lemma and it is available in the dictionary. We have the WordNet corpus and the lemma generated will be available in this corpus. NLTK provides us with the WordNet Lemmatizer that makes use of the WordNet Database to lookup lemmas of words.

3.4 LSTM Network

Deep learning is a new research direction in the field of artificial intelligence. It is developed on the basis of shallow neural networks with the improvement of computer hardware levels and the explosive growth of the current data volume. Deep learning and shallow neural network structure are both layered. Each layer will process the data input to the model and combine low-level features into potential high-level features by learning data rules. Compared with shallow models, deep learning can express complex high dimensionality such as high-variable functions and find the true relationships within the original data better. In the 1980s, artificial neural network back propagation algorithm was born. This method can automatically learn data rules from a large amount of training data without manual intervention. At present, deep learning is the most concerned research direction in the field of artificial intelligence, which completely subverts the shallow model in traditional machine, proposes a deep learning network model, and elevates it to a new height from theory to application. CNN (convolutional neural network) and RNN are two types of classical deep learning network structures now.

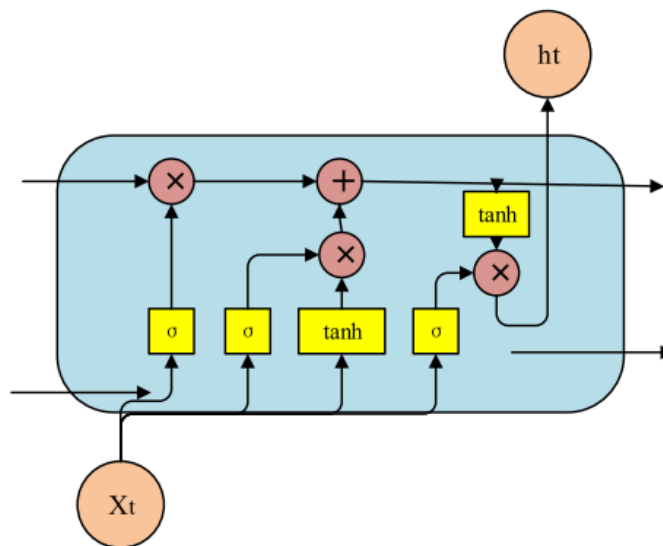


Fig. 4.2: LSTM model structure.

Because there are connections between neurons in the RNN layer, the network can learn the change law of sequence data before and after, and the internal sequence rules of data is easy to be mined. Thus RNN

is widely used in the field of sequence data processing such as speech recognition and machine translation. However, this structure also has some problems. When data is transmitted backward, the problem of gradient disappearance or gradient explosion is unavoidable, which limits its processing of long-term dependencies. The LSTM network changes the way of gradient transmission during backpropagation by adding multiple special computing nodes in the hidden layer of RNN, which effectively slows the problem of gradient disappearance or gradient explosion. Its model structure is shown in figure 4.

Where h_{t-1} represents the output of the previous cell, and x_t represents the input of the current cell. σ represents the sigmoid function. The difference between LSTM and RNN is that it adds a “processor” to the algorithm to determine the usefulness of the information. The structure of this processor is called a cell. Three gates are placed in a cell, which are called *Input gate*, *Forget gate*, and *Output gate*. A piece of information enters the LSTM network, and it can be judged whether it is useful according to the rules. Only the information that meets the algorithm authentication will be left, and the non-conforming information will be forgotten through the *Forget gate*.

FORGET GATE

The first step for data entering the LSTM is to decide what information should be lost and what retained. This decision is made by the Forget gate, which reads h and x and outputs a value between 0 and 1, where 1 means “complete reserved”, 0 means “completely discarded”. Forget gate is calculated as:

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

In the formula, f_t is the calculation result of the Forget gate which is mainly used to control the retention of the information transmitted from the unit state at the previous moment to the unit state at the current moment. $[]$ indicates that the two vectors are spliced, h_{t-1} is the output of the unit at the previous moment, and are the weight and bias of Forget gate, W_f and b_f are Sigmoid activation functions.

INPUT GATE

Input gate determines the addition of new information, and its operation process includes sigmoid layer and tanh layer. The sigmoid layer determines the information that needs to be updated. The calculation formula is:

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

In the formula, i_t is the calculation result of the input gate, and the input gate also has independent weight and bias. The role of the tanh layer is to generate a vector of candidate update information. Its calculation formula is:

$$\tilde{C}_t = \tanh (W_c * [h_{t-1}, x_t] + b_c)$$

\tilde{C}_t is the unit state of the current input, the unit state of the current moment is C_t , and its calculation formula is:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

OUTPUT GATE

Output gate is roughly the same as the Input gate, and its operation flow includes sigmoid layer and tanh layer. The sigmoid layer determines the output part of the information, and the calculation formula is:

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

Finally get the output of the current moment h_t :

$$h_t = o_t * \tanh (c_t)$$

The forward propagation of LSTM calculates the cell state C_t and h_t the output of the current moment and completes the forward propagation calculation of the network. The backpropagation of LSTM is like the back-propagation principle of RNN. Finally, the weights and biases of all parts of the network are updated to complete the model training.

4 RESULTS AND DESCRIPTION

Figure 1: Figure exhibits the graphical user interface (GUI) built using the tkinter library, facilitating interaction with the fake news detection system. Users can utilize this intuitive interface to access various functionalities and perform tasks related to fake news detection, enhancing user experience and accessibility. Figure 2: Shows the process of uploading the fake news dataset into the system. Through this interface feature, users can seamlessly import relevant datasets into the application, providing the necessary data for subsequent analysis and model training.

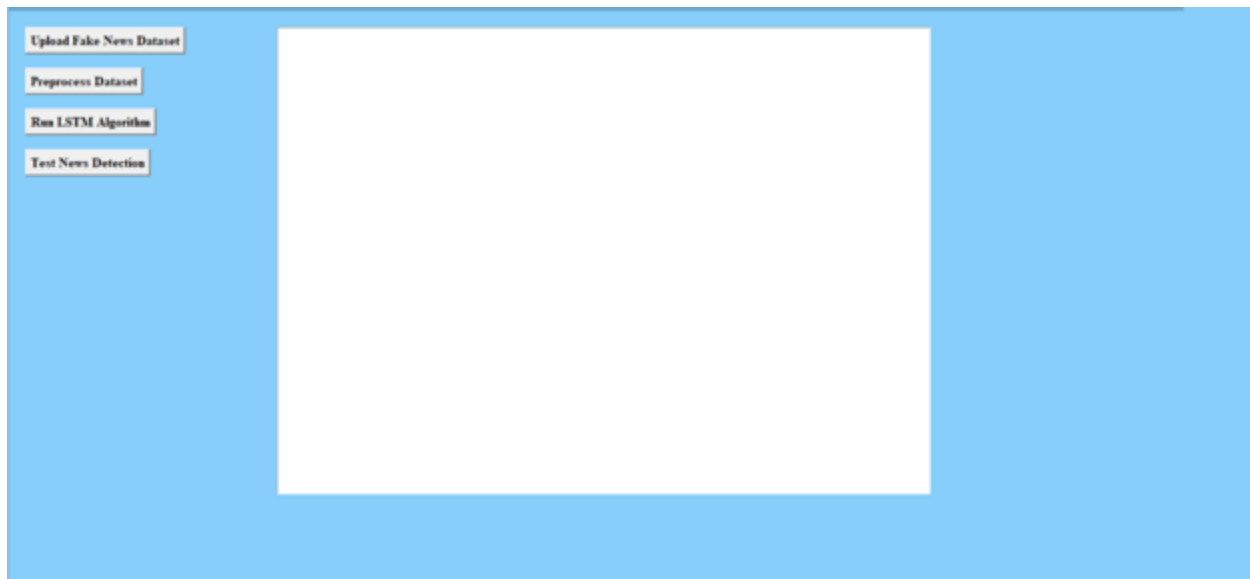


Figure 1: Displays the GUI interface of fake news detection.

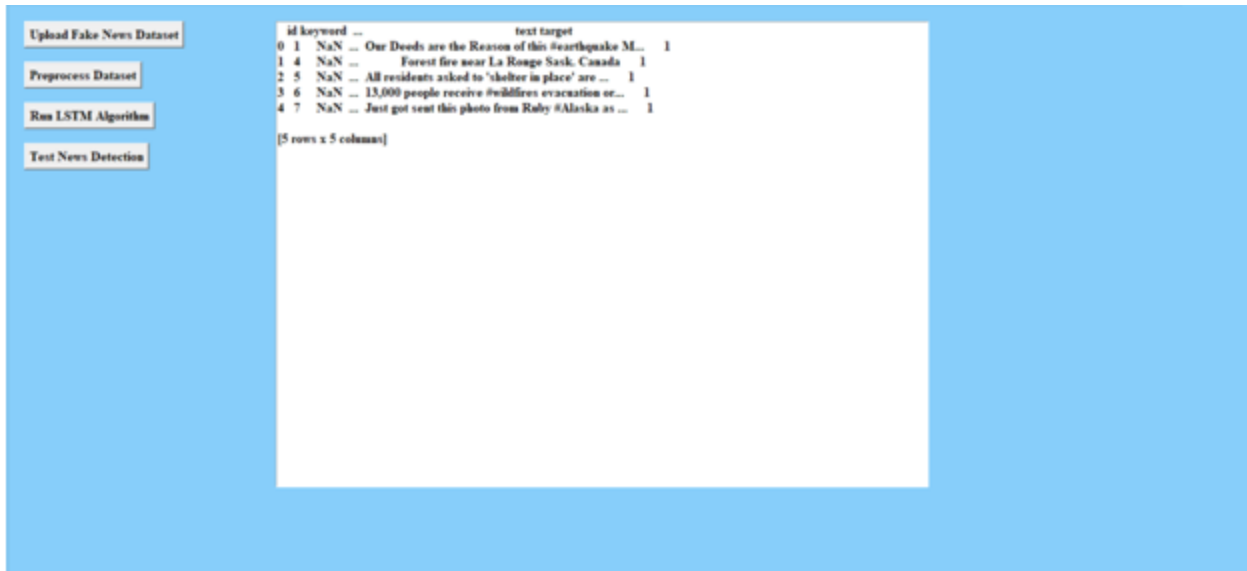


Figure 2: Displays the Upload of Fake News Dataset.

Figure 3 Illustrates the preprocessing steps applied to the dataset, including data cleaning, transformation, and splitting into training (6090 samples) and testing (1523 samples) subsets. This preparatory phase is essential for ensuring data quality and model performance, laying the foundation for accurate fake news detection. Figure 4 Visualization presents the confusion matrix generated by evaluating the LSTM (Long Short-Term Memory) model on the test dataset. The confusion matrix offers insights into the performance of the model by displaying the counts of true positive, true negative, false positive, and false negative predictions, enabling assessment of the model's accuracy and effectiveness.

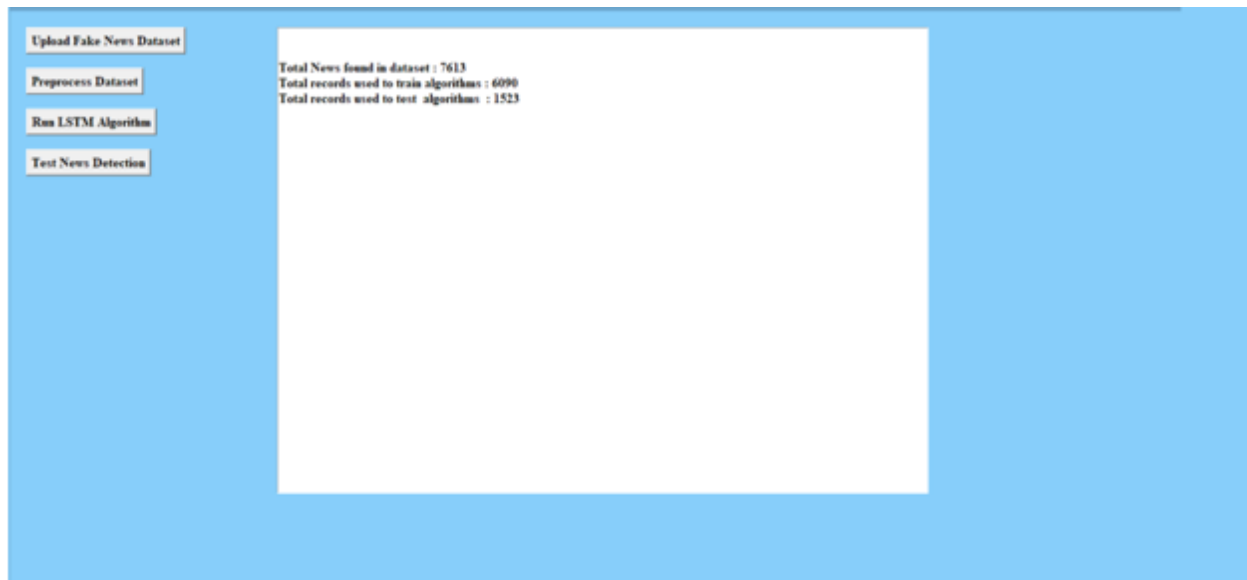


Figure 3: Presents the preprocessing and splitting of data to train 6090 and test 1523.

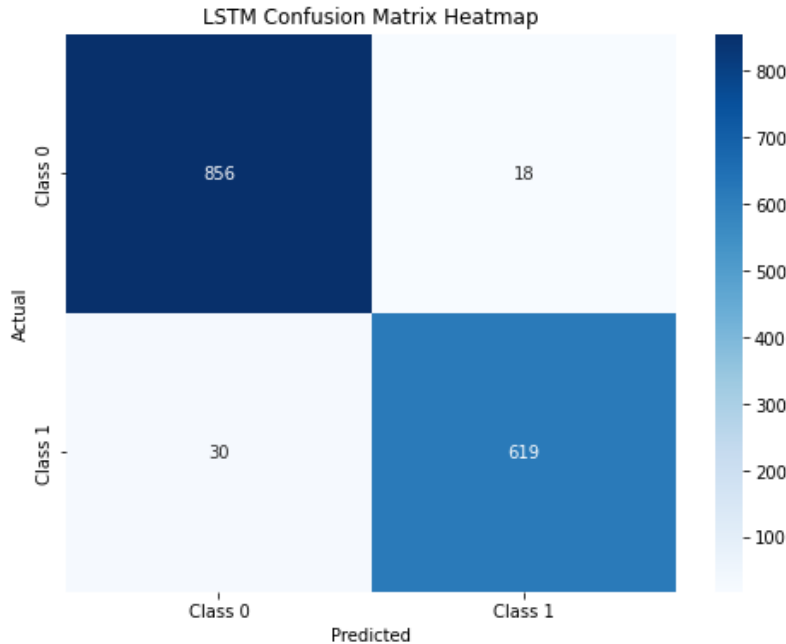


Figure 4: Presents the confusion matrix of LSTM model.

Figure 5 showcases the model's predictions on the test data, illustrating how the LSTM model classifies each instance as either fake or genuine news. By displaying the model's outputs, users can evaluate its performance and gauge its ability to accurately identify fake news, ultimately assessing the system's effectiveness in fulfilling its intended purpose.

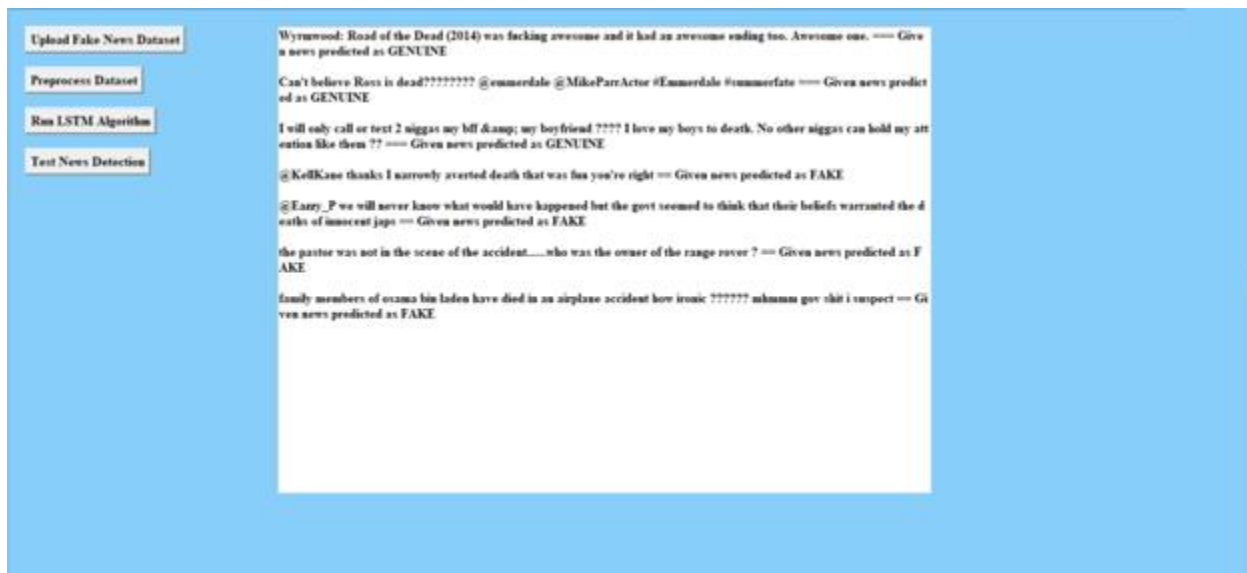


Figure 5: Displays the model prediction on test data.

5 CONCLUSION

We exploit the latent community structure in the global news network to improve the prediction of the viral cascades of news about events. The cascades which have early adopters in different communities have advantages in disseminating the contagion to these communities in parallel and therefore are more likely to result in the viral infections within a limited time period. Our model captures such property by inferring the community structure using the response times of nodes. Thus, we avoid using the explicit network topology which is often not known because the references to propagation sources are usually missing in the real data sets. Due to the size of the relevant data sets, we successfully parallelized the inference algorithm for distributed memory machines and tested this parallelization on the RPI AMOS achieving orders of magnitude speedup.

REFERENCES

- [1] A. Nematzadeh, E. Ferrara, A. Flammini, and Y. Y. Ahn, “Optimal network modularity for information diffusion,” *Phys. Rev. Lett.*, vol. 113, no. 8, p. 088701, 2014.
- [2] J. Firmstone and S. Coleman, “The changing role of the local news media in enabling citizens to engage in local democracies,” *Journalism Pract.*, vol. 8, no. 5, pp. 596–606, 2014.
- [3] R. A. Hackett, “Decline of a paradigm? Bias and objectivity in news media studies,” *Crit. Stud. Media Commun.*, vol. 1, no. 3, pp. 229–259, 1984.
- [4] S. Della Vigna and E. Kaplan, “The Fox News effect: Media bias and voting,” *Quart. J. Econ.*, vol. 122, no. 3, pp. 1187–1234, 2007.
- [5] M. Gentzkow and J. M. Shapiro, “Media bias and reputation,” *J. Political Economy*, vol. 114, no. 2, pp. 280–316, 2006.
- [6] M. Karsai et al., “Small but slow world: How network topology and burstiness slow down spreading,” *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 83, no. 2, p. 025102, 2011.
- [7] F. D. Gilliam, Jr., and S. Iyengar, “Prime suspects: The influence of local television news on the viewing public,” *Amer. J. Political Sci.*, vol. 44, no. 3, pp. 560–573, 2000.
- [8] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Apr. 2002.
- [9] J. Yang and J. Leskovec, “Overlapping community detection at scale: A nonnegative matrix factorization approach,” in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 587–596.
- [10] S. Ji, N. Satish, S. Li, and P. Dubey. (2016). “Parallelizing word2vec in shared and distributed memory.” [Online]. Available: <https://arxiv.org/abs/1604.04661?context=stat.ML>
- [11] T. Mikolov, I. C. K. Sutskever, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
- [12] A. Mnih and Y. W. Teh. (2012). “A fast and simple algorithm for training neural probabilistic language models.” [Online]. Available: <https://arxiv.org/abs/1206.6426>
- [13] B. Recht, C. Re, S. Wright, and F. Niu, “HOGWILD: A lock-free approach to parallelizing stochastic gradient descent,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 693–701.
- [14] R. G. Miller, Jr., *Survival Analysis*, vol. 66, Hoboken, NJ, USA: Wiley, 2011.

- [15] A.-L. Barabási. (2005). “The origin of bursts and heavy tails in human dynamics.” [Online]. Available: <https://arxiv.org/abs/cond-mat/0505371>
- [16] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2014, pp. 701–710.
- [17] D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2003, pp. 137–146.
- [18] E. Mossel, J. Neeman, and A. Sly. (2012). “Stochastic block models and reconstruction.” [Online]. Available: <https://arxiv.org/abs/1202.1499>
- [19] R. M. Gomez, J. Leskovec, and B. Schölkopf, “Structure and dynamics of information pathways in online media,” in Proc. 6th ACM Int. Conf. Web Search Data Mining, 2013, pp. 23–32.
- [20] N. X. Vinh, J. Epps, and J. Bailey, “Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance,” J. Mach. Learn. Res., vol. 11, pp. 2837–2854, Jan. 2010.