FNU-BiCNN: Fake News and Fake URL Detection using Bi-CNN

R.Sandrilla[1]
Research Scholar, Department of Computer Science
Periyar University, Salem
Tamilnadu, India

M.Savitha Devi[2]
Assistant Professor, Department of Computer Science
Government Arts and Science College, Harur
Dharmapuri DT, Tamilnadu, India

***Abstract*—Fake news (FN) has become a big problem in today's world, recognition partly to the widespread use of social media. A wide variety of news organizations and news websites post their stories on social media. It is important to verify that the information posted is genuine and obtained from reputable sources. The intensity and sincerity of internet news cannot be quantified completely and remains a challenge. We present an FNU-BiCNN model for identifying FN and fake URLs in this study by analyzing the correctness of a report and predicting its validity. Stop words and stem words with NLTK characteristics were employed during data pre-processing. Following that, we compute the TF-IDF using LSTM, batch normalization, and dense. The WORDNET Lemmatizer is used to choose the features. Bi-LSTM with ARIMA and CNN are used to train the datasets, and various machine learning techniques are used to classify them. By deriving credibility ratings from textual data, this model develops an ensemble strategy for concurrently learning the depictions of news stories, authors, and titles. To achieve greater accuracy while using Voting ensemble classifier and compared with several machine learning algorithms such as SVM, DT, RF, KNN, and Naive Bayes were tried, and it was discovered that the voting ensemble classifier achieved the highest accuracy of 99.99%. Classifiers' accuracy, recall, and F1-Score were used to assess their performance and efficacy.**

Keywords*—Bi-LSTM; CNN; WORDNET; machine learning; fake news and URL; ARIMA

List of Abbreviations

FN-Fake news

FNU-BiCNN- Fake News and Fake URL Detection Using Bi-CNN

SVM- Support Vector Machine

DT-Decision Tree

RF-Random Forest

KNN- K-Nearest Neighbors

NB- Naive Bayes

CNN- Convolutional Neural Network

LSTM- long short-term memory network

ARIMA- Autoregressive Integrated Moving Average

URL-Uniform Resource Locators

MSSE- Minimum Sum of Squared Errors

BP- Back Propagation

TF- Term Frequency

IDF- Inverse Document Frequency

POS-Part of Speech

RNN- recurrent neural network

NLTK- Natural Language Toolkit

I. Introduction

Recent years have seen a rapid rise in the popularity of social networking sites due to greater media coverage. Rather than traditional media, social networking platforms are the preferred news source for many people [1]. Users of social networks may engage with individuals who share their interests and ideas. It's questionable, though, about the quality of the news. This media outlet disseminates false material in the style of news stories [2]. People and society might be adversely affected by its widespread use. FN is information that has been created solely to mislead the public. It is impossible to accurately measure the reliability of information posted on social media networks [3]. To overcome the problem above, a standardized solution is needed [4].

Numerous dangerous consequences might result from our culture's exponential growth of FN. First, FN alters how people view and respond to legitimate news. Second, the proliferation of FN [11] would consumers' faith in the media, make them distrustful, and jeopardize the news medium's legitimacy [7] [8]. Third, deliberate FN persuades people to accept skewed and manufactured tales [10].

There is lot of reasons to choosing the FN detection: First, true or untrue stories greatly influence a country's elections, such as in India, where 45% of voters believe fact-checking groups exist. [11] [12]. Most WhatsApp users in India prefer to accept transmitted information without checking it, which is the second-largest population in the world [13-17]. On the one hand, social media businesses are faced with the enormity of their enterprise as they consider the possible exploitation of its base. Examples of FN are photoshopped images, client-created content, or caricature accounts. Second, there is mounting evidence that customers have behaved bizarrely in response to news that was subsequently shown to be false. One recent example is the propagation of the new corona virus, which was propagated by false claims about the virus's origin, biology, and behavior. The situation deteriorated as more individuals became aware of the fabricated information online. Finding such news online is a difficult Endeavour.

Other types of FN include stories intended to appeal to a specific group or association and stories that offer a scientific or affordable explanation for an unresolved issue, leading to the spread of incorrect information. FN detection and Fake URL detection bring new and tough difficulties due to the aspects above [18-21].

For detecting malicious URL has merging a trust computational model with a collection of URL-based characteristics. And Malicious URL detection has used Bayesian learning and Dempster–Shafer theory to assess the credibility of tweets and it has only 95% accuracy rate [16].

Internet news items may automatically detect FN and URL information in internet news items using two new datasets. Data sets that have been pre-processed are utilized to distinguish between fake and legitimate news. ARIMA, Bi-LSTM, and Convolutional Neural Networks (CNN) deep learning algorithms were utilized for training the datasets. Our last step is using ensemble learning, in which we combine many classifiers to improve a model's ability to classify and approximate new data accurately. Base classifiers include Support Vector Machine (SVM), KNN, Naive Bayes [NB], Decision Tree (DT), and Random Forest (RF). Together, these two classification models provide a better estimate and a classifier that beats all others in terms of accuracy and predictability. By combining numerous models and then averaging them to generate a final model, this method helps limit the danger of a poor-performing classifier.

The following are the primary goals and accomplishments:

A strategy for spotting FN across several sources:

a) Extract top-level text features from actual and FN articles using TF-IDF and WORDNET.

b) URL characteristics may be extracted by looking at the domain name (domain)

c) The on-site URL feature may be used to estimate the multi-source trustworthiness score by combining text-based characteristics and multi-source credibility ratings to estimate news credibility.

A. Our Contributions

Multiple instances of text classification using both supervised and unsupervised learning methods have been seen in the present FN corpus. However, the majority of the research focuses on certain datasets or topics, most notably the realm of politics. As a consequence, the algorithm trained on a specific kind of content performs optimally when exposed to articles from various areas. Our research examines many textual features that may be utilized to detect false from genuine information. We use these qualities to train a mix of distinct machine learning algorithms utilizing voting ensemble approaches that have not been extensively investigated in the present literature. Voting ensemble learners have been shown to be beneficial in a broad number of applications due to their propensity to lower error rates. These strategies assist the effective and efficient training of various machine learning algorithms. Additionally, we performed extensive experiments on two publicly accessible real-world datasets as Fake news and URL datasets.

The rest of the article is arranged in the following manner: Section 2 deals with existing research methods for FN and URL detection. Section 3 explains how the FNU-BiCNN architecture works and how it is put into action. Section 4 presents the results of the research into the FNU-BiCNN framework. Conclusions and future research are discussed in Section 5.

II. Background Study

Agarwal, A., & Dixit, A. in [1] instead of studying a single approach, the author employed ensemble learning. The average accuracy score discovered was 85%, which is 15% better than the accuracy of the worst-performing KNN model. Additionally, the authors utilized just a percent of the information even with the supplied dataset and values. The remainder of the data was inadequate and did not give further distinguishing characteristics between fake and authentic news.

In Ahmed, A. A., & Abdullah, N. A. [2], the URLs of online pages may be used to identify phishing websites. The proposed method might distinguish between legitimate and counterfeit websites by looking at the Uniform Resource Locators (URLs) of suspicious web pages (URLs). URLs are examined for a variety of characteristics to identify phishing sites. The identified attacks are reported to the proper authorities to prevent such incidents.

In Birunda, S. S., & Devi, R. K. [3], a novel score-based framework for detecting FN from multiple sources has been developed. Using the TF-IDF approach, the top actual and false characteristics were retrieved from news articles. The Credibility Score of the sources was determined using the Site URL attributes provided from the source. To determine the news's dependability, the retrieved text-based characteristics and the multi-source Credibility Score were combined. The suggested framework's efficacy and practicality are assessed and compared to different classifiers.

Cheng, W. et al. in [5] for integrated forecasting models, presents a novel weighting approach for MSSE (Minimum Sum of Squared Errors) models that combine ARIMA (Autoregressive Integrated Moving Average) time-series models with BP (Back Propagation) neural networks.

Granik, M., & Mesyura, V. in [6] shows how to use a naive Bayes classifier to identify FN quickly. This strategy was developed and tested using a computerized system using Facebook data collection. For a quite simple model, the authors achieved a classification accuracy of around 74% on their test set.

In Jayasiriwardene, T. D., & Ganegoda, G. U. [9] to gather data to detect FN stories, this article demonstrates a method for extracting keywords from a particular tweet's body text. The proposed approach uses Stanford core NLP, POS tagging, and TF-IDF statistical techniques to identify keywords. The Wordnet lexical database was used to find synonyms, and Ginsim and word2vector may be used in combination to assess how related two words are. Using a bi-gram technique, keywords are created to increase news retrieval accuracy and efficiency. A list of the most relevant news tweets regarding the claimed tweet is compiled using the extracted keywords.

In Mansouri, R. et al. [13], use of semi-supervised linear discriminant analysis and CNNs are described in the following article to detect FN. With the addition of unlabeled datasets, it was necessary to increase training data. The proposed method alters LDA to provide a semi-supervised estimation of classes.

Qazi, M. et al. [15] discusses a novel approach for detecting fraudulent or legitimate news. For detecting purposes, liar data is employed. The literature review reveals that several machine learning-based detection approaches identify FN. However, these models lack performance. The authors attempt to increase performance by developing a transformer model based on the attention process.

In Rout, R. R., et al. [16], authors present a LA-MSBD method for detecting malicious social bots by merging a trust computational model with a collection of URL-based characteristics (MSBD). Additionally, the authors are used Bayesian learning and Dempster–Shafer theory to assess the credibility of tweets. MSBD has a 95% accuracy rate.

In Seo, Y., & Jeong, C.S. [19], the authors validated the proposed model using two distinct dataset types. To begin, when the sequence-to-sequence learning model is utilized, the parallel corpus dataset is used as an input for creating the sentences. Second, to modify CNN news articles provided by DeepMind and construct different sorts of propositions in order to test inference performance.

In Vogel, I., & Meghana, M. [20] as a first step in limiting the transmission of false news among online users, the authors presented three distinct ways for automatically detecting suspected FN spreaders on social media. The authors utilized the PAN 2020 author profile corpus and performed a variety of multilingual learning tests. To assess a variety of handmade and machine-learned qualities, the majority of which are language-independent? The characteristics were retrieved and their significance in the detection job was determined.

In Zhi, X. et al. [21], the authors present a unique model based on CNN-LSTM that incorporates news organizations, comments, sources, and market data in this study. The findings demonstrate that our strategy beats previous work that manually extracts features or criteria.

III. Proposed Framework

This section adds to the comprehensive approach of the proposed 'FNU-BiCNN' framework. The news data are first gathered, filtered, and pre-processed. Then, text-based elements are retrieved to determine the reliability of news items and URL information. Additionally, an ensemble technique is presented to use the source's credibility score and text-based elements to determine the authenticity of a news story and URLs. The FNU-BiCNN framework's pipeline is shown in Fig. 1.

A. Problem Statement

If a news article's content and its credibility score are both proved to be false (false) and genuine (true), otherwise, it is considered to be a fake.

Allowing for the fact that an infinite number of news stories originate from numerous sources, we designate this dataset as D = D_1, D_2..., D_{ns}. Y = "Real, Fake" is the dataset's classification code. The new unlabeled news article's degree of fakeness may be predicted using the dataset D, news articles from numerous sources (s), and the class labels Y.

B. Dataset

www.Kaggle.com's experimental dataset comprises 23,000 news stories that are either false or authentic. The data collection is 100 MB in size. This dataset contains information on the source of the news stories, the date of publication, the author, the title, and the text. True and fake datasets are used for training and testing purposes. 976 false and 598 actual news pieces are utilized for training purposes out of a total of 1574 articles. 241 fake samples and 152 authentic samples were found in 393 news stories. The difficulty of detecting FN articles is one of ML classification. There are 1217 FN pieces and 750 legitimate news articles shown in Fig. 2. More than 1500 URL links with descriptions were gathered from kaggle.com for the URL dataset. The dataset is 50 MB in size. The link source, title, and description are all included in this dataset. As a data source for training and testing, this dataset is divided into real URLs and fake URLs.

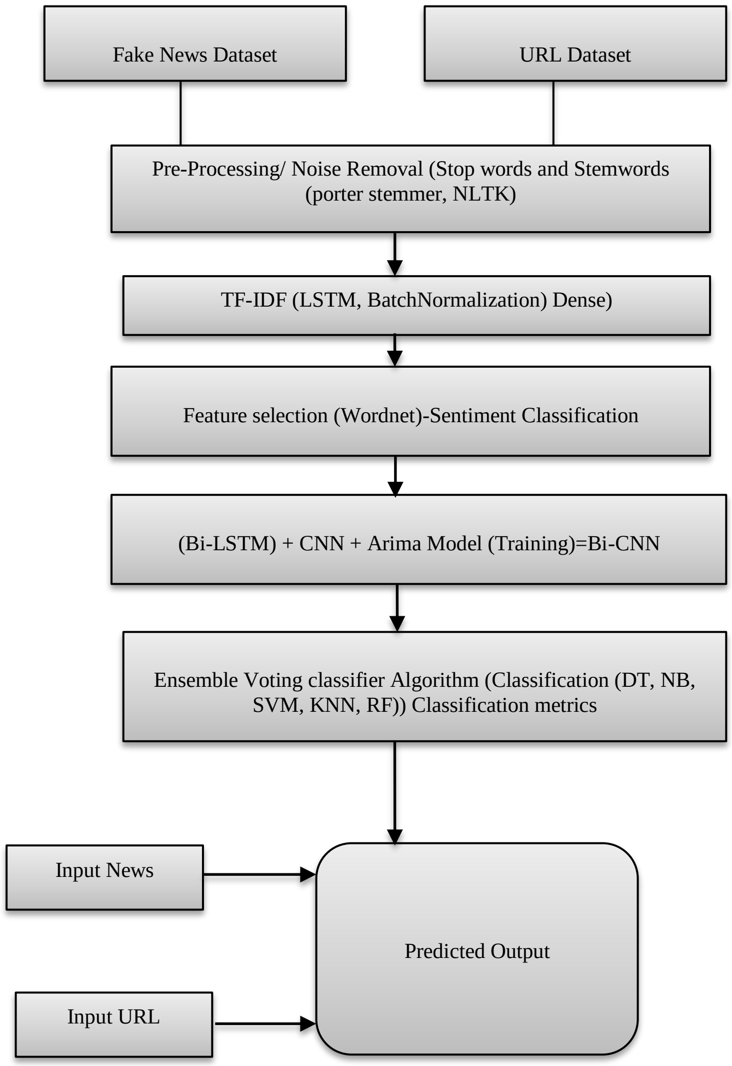

Fake News Dataset
URL Dataset
Pre-Processing/ Noise Removal (Stop words and Stemwords (porter stemmer, NLTK)
TF-IDF (LSTM, BatchNormalization) Dense)
Feature selection (Wordnet)-Sentiment Classification
(Bi-LSTM) + CNN + Arima Model (Training)=Bi-CNN
Ensemble Voting classifier Algorithm (Classification (DT, NB, SVM, KNN, RF)) Classification metrics
Input News
Input URL
Predicted Output

Fig. 1. FNU-BiCNN Model for Detection Fake News and URL.

C. Preprocessing

Processing raw data into a machine-readable format is known as pre-processing in the context of data mining. Many text pre-processing operations were carried on the news and URL datasets. While working toward these goals, the Keras library's algorithms were employed for character conversion to lowercase letters, stopword removal, stemming, and tokenization. Stopword, such as 'of,' 'the,' 'and,' 'an,' and similar words, are often found in texts used in this work. Eliminating stopword speeds up processing and frees up space formerly used by the useless words listed above. Words with similar meanings often occur in the text, such as games and sport. The strength of shortness is in the removal of unnecessary words. This is known as stemming, and it's done using the Porter stemmer technique from the NLTK open-source version.

The headlines' keyword density was reduced to 372 due to the aforementioned pre-processing steps. Keras' library's tokenizer function divided each headline into a vector of words. The text is first transformed into vectors using word embedding (word2vec). Finally, a vocabulary for the 5,000 unigram words present in headlines and article bodies is developed. The maximum headline length is used for all headlines. Padding is not applied to headlines that are shorter than the maximum length.

D. Feature Extraction

In our FNU-BiCNN approach, we analyze the extracted characteristic from the news and link it to the news's speaker or author. In addition, the linked author is given a credibility score depending on the number of fake and real stories he publishes on his website. A good credit score assures accuracy and trustworthiness. This is why our system approaches the problem in two ways: first, by extracting news text's semantics, and second, by giving a credibility score to each author. As a result, a vector matrix is formed that links the word list to each source. We turned to the sci-kit learn python libraries for feature extraction and selection. After selecting features using the bag of words and n-grams technique, we used TF-IDF weighting (Term Frequency-Inverse Document Frequency). Additionally, we used word2vec and POS tagging to extract features. There is a short discussion of each feature extraction model:

Stop words will be labeled, and the labels will be used to eliminate them. WordNet is used to generate semantic phrases related to the subject terms. WordNet is an API that can generate synonyms for a given term. The TF-IDF approach is used to calculate the weight of these words.

- Bag-of-words: It is a basic method for representing text data and extracting textual characteristics. This section tokenizes and counts the words associated with each observation.

$$Weight = TF * IDF \tag{1}$$

For a term *i* in document *j:*

$$W_{ij} = tf_{ij} X log(\frac{N}{df_i}) \tag{2}$$

tf_{ij}=number of occurrences of *i* in *j*

df_i=number of documents containing *i*

N=total number of documents

$$w^x + b = 0 \tag{3}$$

Where w is the weight vector and b are the bias

$$L(w) = \sum_1 \max(0, 1 - y_i[w^t x_i + |b|]) + |w||^2 \tag{4}$$

- The term "n-gram" refers to an uninterrupted sequence of n tokens or words. To get the most out of your n-grams, it's better to use a bag of words rather than just words.
- Information may be retrieved using the TF-IDF algorithm. This measure provides an accurate value when the token is commonly used in the document and frequently used in the corpus. Each word is given a word frequency score, with the more interesting ones receiving the highest marks.
- Word2vec: This method embeds semantically similar words adjacently in numeric vectors called embeddings.
- POS-Tagging: POS tagging is described as the act of associating a word with certain portions of speech depending on its context and meaning. It may be employed to resolve grammatical ambiguity or disambiguate word senses to establish the news's legitimacy.

Proposed Algorithm 1: Bi-CNN

```
Algorithm 1: Bi-CNN:
Input: News articles and URL datasets
Output: Fake or Real Data

Step 1: Pre-processing using NLTK's porter stemmer algorithm
Step 2: Extract Top features using TF-IDF using LSTM, Batch
normalization, and dense
Step 3: Select Top features using TF-IDF and Bag of words
Step 4: If features are selected, then
Step 5: Using WORDNET sentiment analysis
Step 6: Else
Step 7: Repeat step 1
Step 8: Training data using ARIMA+BiLSTM+CNN
Step 9: Find the Training accuracy and loss
Step 10: Classification is done by ML Algorithms
Step 11: Find the Classification Metrics
Step 12: Create a pickle file
Step 13: Find the fake news and URLs
Step 14: End
```

E. Training Data

1) Arima model: The abbreviation ARIMA stands for Auto-Regressive Integrated Moving Average (ARIMA). When the lags of the stationarized series are included in the forecasting equation, it is referred to as an "autoregressive" component. When the lags of prediction errors are used, "moving average" is used. When a time series must be differentiated to be deemed inactive, it is referred to as an "integrated" form of a stationary series. ARIMA models include random-walk and random-trend models, autoregressive models, and exponential smoothing models [12].

The notation "ARIMA (p, d, q)" represents an ARIMA model that is not seasonal, with p signifying the number of autoregressive components and d and q denoting the number of dependent variables.

The number of nonseasonal deviations necessary for stationarity is given by d in the prediction equation, whereas the number of delayed forecast errors is denoted by q.

The forecasting equation is constructed in the following manner. Let's start by referring to y as the dth difference of Y, which equals eq (5,6,7):

$$if\ d = 0, y_t\ equals\ Y_t \tag{5}$$

$$if\ d = 1, y_t = Y_t - Y_t - 1 \tag{6}$$

$$if\ d\ is\ equal\ to\ 2, then\ Y_t = (Y_t - Y_t)2Y_t - 2 = Y_t - 2Y_t - 1 + Y_t - 2 = (Y_t - 1Y_t - 2) \tag{7}$$

a) FN Detection in Time Series: Because of the nature of the data, we cannot employ traditional FN detection algorithms to detect anomalies in time series data. The challenge of time series FN identification must be addressed separately from the other jobs.

The following are the actions that must be followed to detect FN in time-series data:

- Make a note of whether the FN data is moving or not. Make the FN data motionless by changing it too stationary if it isn't already.
- The study's findings fit a time series model to the pre-processed FN data.
- Calculate the observation's Squared Error for every observation in the data.
- Calculate your data's error threshold.
- We can label an observation as FN if the number of mistakes surpasses a specific threshold.

As previously established, time-series FN data is strictly sequential and prone to autocorrelation in distribution. Time-series models would be trained and utilized to discover the general behavior of the fictional data, and they would attempt to forecast the actual data using the fictitious data. If an observation is normal, the forecast will be as close to the true value as feasible; but, if an observation is an FN, the forecast will be as far from the true value as possible; hence, by studying the forecast errors, we may detect the FN in the data [5].

2) Bi-LSTM: A recurrent neural network (RNN) is the go-to choose when dealing with sequential input. It stores and uses just the most significant parts of the incoming data to predict the future output. Memory cells in the RNN keep track of the most important information from earlier inputs. Long-term dependence isn't taken into consideration either. The consequence is the creation of a particular RNN to deal with long-term dependency. It is known as the long short-term memory network (LSTM). Input (IG), output (OG), and forget gates are three of the gating principles used to do this (FG). (8)– (11) explain how the information flow (read, write, and reset) in the gradient is regulated by these gates in conjunction with the candidate hidden state (CHS), current state (CS), and hidden sequence (HS) (10). (12). Left-to-right or right-to-left are the only two input directions that the LSTM network accepts for processing. Consequently, gathering new information will be more difficult in the future. Since the original input sequence is being learned in both directions, Eq. employs a bidirectional LSTM in this case (13).

$$IG_t = \sigma(w_{IG}x_t + R_{IG}h_{t-1} + b_{IG}) \tag{8}$$

$$OG_t = \sigma(w_{OG}x_t + R_{OG}h_{t-1} + b_{OG}) \tag{9}$$

$$FG_t = \sigma(w_{FG}x_t + R_{FG}h_{t-1} + b_{FG}) \tag{10}$$

$$CHS_t = tanh(w_{CHS}x_t + R_{CHS}h_{t-1} + b_{CHS}) \tag{11}$$

$$CS_t = FG_t \text{ X } CS_{t-1} + IG_t \tag{12}$$

$$Y_t = V(H\,s_t{:}\,H\,S_t) \tag{13}$$

Where w_{IG}, w_{OG}, w_{FG}, and w_{CHS} are referring to the weight matrices of the current input x_t. R_{IG}, R_{OG}, R_{FG}, and R_{CHS} are referred to as the weight matrices of the previous state h_{t-1}. R_{IG}, R_{OG}, R_{FG}, and R_{CHS} are denoted as the bias value. Y_t represents the output of the forward LSTM and backward LSTM units.

3) Convolutional neural network: CNN: To summarize, CNN is a deep learning model that does very well in image categorization and automated natural language processing. When it comes to identifying higher-level traits, CNN has a distinctive structure. The primary processing unit of CNN is the convolutional layer, which uses matrix coefficients to identify features. There are several kernels or filters in this layer. These filters use an activation function called ReLU to help process a portion of the input sequence throughout the whole input data set (Rectified Linear Unit). As an extension of the corrected linear unit, the ReLU aims to remove negative values from the activation map by setting it to zero in most cases. It is, therefore, more effective than the sigmoid and the Tanh activation functions for solving the problem of the unseen gradient.

As a result of the convolutional approach, a fixed-sized word-level embedding may be obtained by first aggregating the local features generated by the neural network around each word in the neighboring word. As a result, CNN is used to model latent textual properties to identify FN. Let the jth word in the news *i* denote as $x_{i,j} \in R^k$, which is a k-dimensional word embedding vector. Believe the maximum length of the news is n, s.t., the news has less than n words can be padded as a series with length n. Hence, the overall news can be written as

$$X_{i,l:n}^{T1} = x_{i,1} \oplus x_{i,2} \oplus x_{i,3} \dots \oplus x_{i,n} \tag{14}$$

This means that the news $X_{i,l:n}^{T1}$ is concatenated by every word. In this case, each news can be presented as a matrix. Then we use convolutional filters $\omega \in R^k$ to construct the new features. The feature c_i as follows:

$$c_i = f\left(w.X_{i,j:j+h-1}^{T1} + b\right) \tag{15}$$

Where, the b$\in R^k$. A max-pooling layer is applied to take the maximum feature map c. The maximum value is denoted as c=max{c}. Convolutional discoveries may be saved for FN detection in the max-pooling layer, increasing the model's durability by putting the pooling results into a fully connected layer.

4) Ensemble model for classification: SVM, DT, RF, KNN, and NB classification algorithms combine and use voting ensemble classifier algorithms. All of the code was written in Python. Python libraries are used in our source code: Keras, NLTK, NumPy, Pandas, Sklearn, and scikit. Algorithms were judged on their accuracy, precision, recall, and F-score, among other metrics.

a) Naïve Bayes is a machine learning method used to solve text categorization issues. Apart from that, it is quite simple to apply and extremely effective.

b) SVM (Support Vector Machine): When it comes to regression-based classification, an SVM is a common tool. Despite this, it is a common tool for resolving categorization issues. A point on an N-dimensional space, where N is the number of spaces, is often used to represent each piece of data. In addition, each element has its value, which indicates a certain location. Finally, we arrive at the hyperplane on which the data are grouped. Outliers may be excluded using the SVM algorithm while calculating the hyperplane.

c) KNN (K-Nearest Neighbors): It is one of the simplest machine learning models that adhere to the principles of supervised learning. It forecasts the similarities between the data for which the class is predicted and the data for which the class is already predicted. It classifies the new situation as a class, which is quite similar to a record or data. It is also applicable to regression and classification. It is, however, mostly utilized in categorization.

d) Random Forest Algorithm (RF): This is a well-known technique for supervised machine learning. Classification and regression are two possible applications. It uses ensemble learning, which combines the results of several classifiers to improve overall accuracy.

e) DT (Decision Tree): In classification, the DT algorithm is one of the most often used methods. C4.5 algorithm, which needs all data to be quantitative or categorical, is used. Continuous data will not be analyzed as a consequence of this decision. Pruning in DT may be accomplished in two ways. For instance, one method, termed "subtree replacement," proposes replacing nodes in a decision tree to minimize the number of tests in the convinced path. Typically, subtree raising has a negligible effect on decision tree models. Typically, there is no precise method to anticipate the utility of choice. However, it may be prudent to disable it if the induction method takes longer than expected due to the computational complexity of raising the subtree.

IV. Results and Discussion

CUDA is a technology developed by NVIDIA. Python offers a driver and runtime API for existing toolkits and libraries, simplifying GPU-accelerated computation for optimum performance while maintaining simplicity. The data set of FN material, which can be accessed here, and the news URLs, which can be viewed here, were pre-processed using the Porter stemmer algorithm and NLTK. The Natural Language Toolkit (NLTK) is a critical framework for developing programs that interact with data derived from human language in Python programming. Additionally, it includes over 50 corpora and lexical resources, such as WordNet, and a collection of text processing libraries for classification, tokenization, stemming, labeling, parsing, semantic reasoning, and wrappers for wrappers industrial-strength natural language processing (NLP) libraries. Porter stemming (or Porter stemmer) is a method for removing morphological and in flexional ends from English words. When setting information retrieval systems, the normalizing approach is most often used, and it is most commonly referred to as a "normalization process." In this work, the Porter stemming technique was used to extract morphological and in flexional ends from news articles, allowing for sentence normalization and mistake removal.

The TF-IDF technique is used to mitigate the effect of tokens that often occur in a dataset and are empirically less significant than features in a tiny fraction of the training dataset. The number of negative and positive words has been tallied in this study, with 0 positive and one negative word shown in Fig. 2.

The year-wise news and its positive and negative words have been measured in this work with Fig. 2 and X-axis denotes the fake news Subjects and Y-axis denotes the number of news counts.

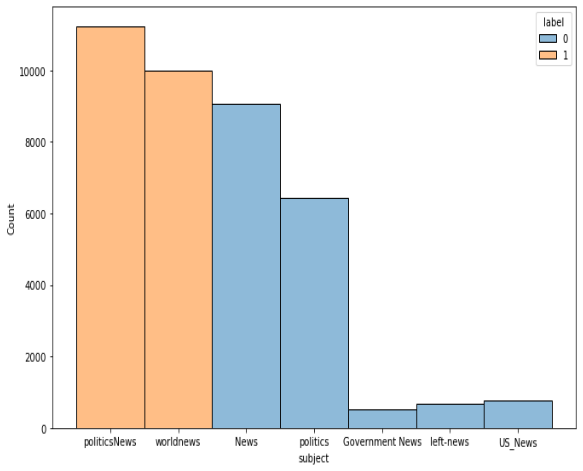


Fig. 2. Total Word Count.

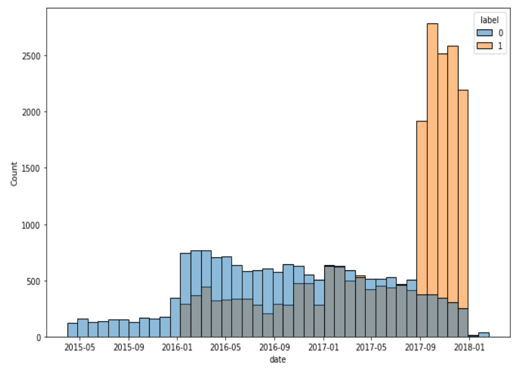


Fig. 3. Total Word Count on Date Wise.

Fig. 3 described that the greater number of FN had been measured in 2017 and 2019 and X-axis denotes the news feeding year and date and Y-axis denotes the number of news counts.

This study's word count and the TF-IDF calculated the average character length. The length of the false words character on the news content dataset was much longer than the length of the real words character due to the findings, as shown in Fig. 4 and X-axis denotes the Label like True or False and Y-axis denotes the Average character length.

Articles' propositions have been measured by counting the number of characters on each true text line and the number of characters on each erroneous text line in this work. The findings indicated that the fake character had used more articles from the news content dataset than the true character. The examination of propositions of articles on true and FN was depicted in Fig. 5, which represented the analysis results and X-axis denotes the number of characters and Y-axis denotes the proportion of articles.

In this work, the proposition of articles has also been measured in terms of the number of words per article on both the actual text and the false text, which has been considered. The findings revealed that the fake character had exploited more articles in the news content collection than the true character. The examination of propositions of articles on the number of words per piece of true and FN was depicted in Fig. 6, which showed the study results and X-axis denotes the number of words per article and Y-axis denotes the proportion of articles.

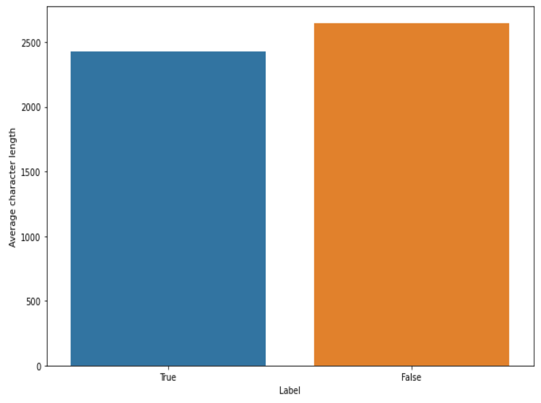


Fig. 4. Average Character Length.

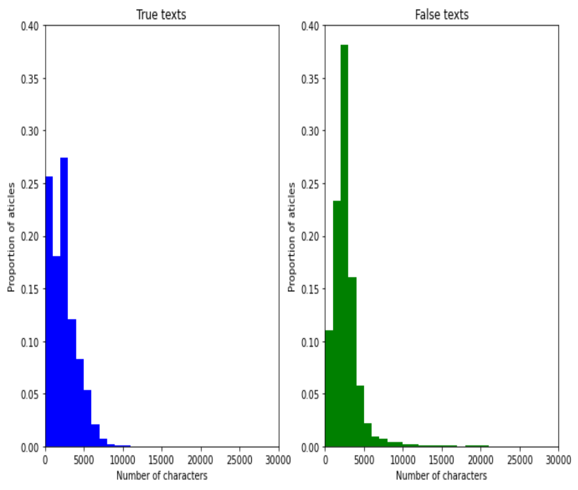


Fig. 5. Number of Character with Proposition of Article.

The work used a Fake URL data set to identify the most often encountered fake URL. The difference between fake and true news has been found in this effort to boost the learning rate of CNN. The sources of FN and authentic news were depicted in Fig. 7 and 8, respectively. With the source data set, 21 typical false news URLs were discovered, and the right-wing news website was shown to have distributed the fakest news based on the URL score and X-axis denotes the news count feed and Y-axis denotes the source of data.

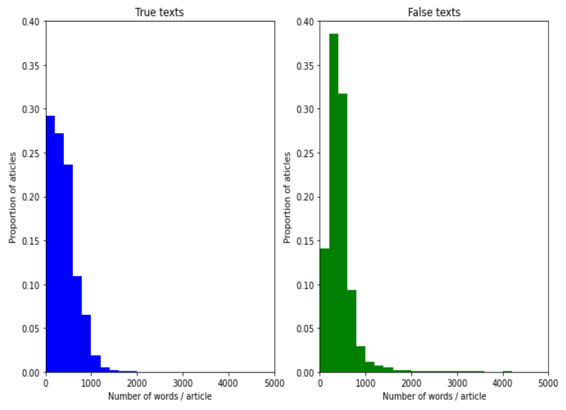


Fig. 6. Number of Character / Articles with Proposition of Article.

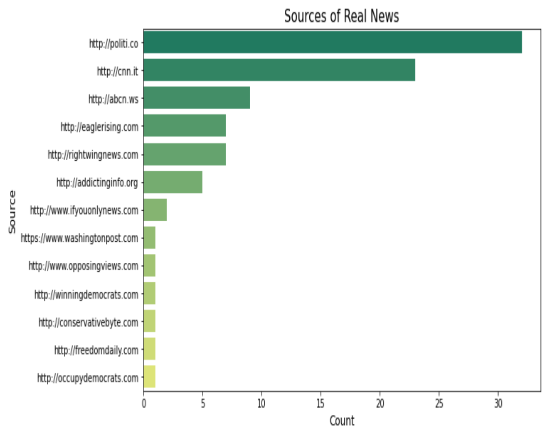


Fig. 7. Source of Real News.

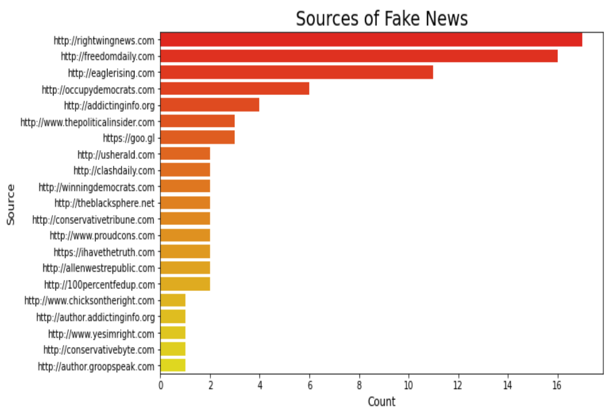


Fig. 8. Source of Fake News.

With this effort, Fig. 9 discovered a common source of actual and FN information and a common source of both. It has been determined that the seven most prevalent URLs are associated with this work and X-axis denotes the news count and Y-axis denotes the source of data.

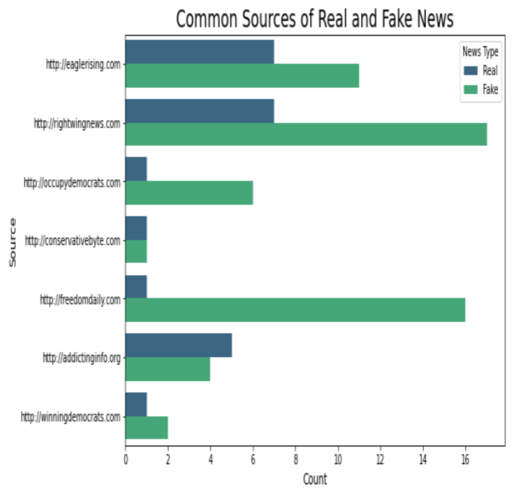


Fig. 9. Source of Common Real and Fake News.

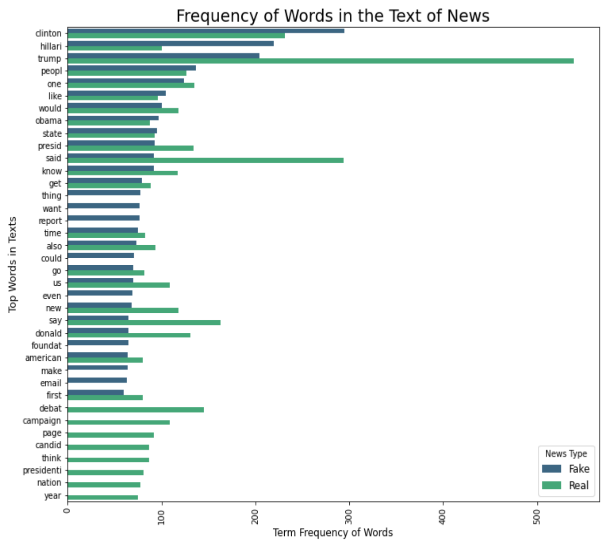


Fig. 10. Frequency Words in the Text of News.

A study was conducted to determine the frequency of word count in the news title and the news content to determine whether or not the news is fake. The chart depicted the frequency of news title appearances, whereas Fig. 10 indicated the frequency of text news appearances and X-axis denotes the Term Frequency of words and Y-axis denotes the Top words list.

Fig. 11 represents that the density of word distribution has been identified based on the frequency of words in the title and their distribution in the content by using the density parameter on the title. Among the findings was that the authentic news series has the largest dispersion density than the FN series. Using the graphic as an example, the outcome demonstrated the observations in the fictitious data are autocorrelated, which indicates that they are closely related to one another and the observations that came before them in the data set. The nature of the data makes it impossible to utilize typical FN detection algorithms to detect abnormalities in time series data due to the way the data is structured. To be successful, the task of time series false news identification must be handled independently of the other jobs. And X-axis denotes the title length and Y-axis denotes the density values.

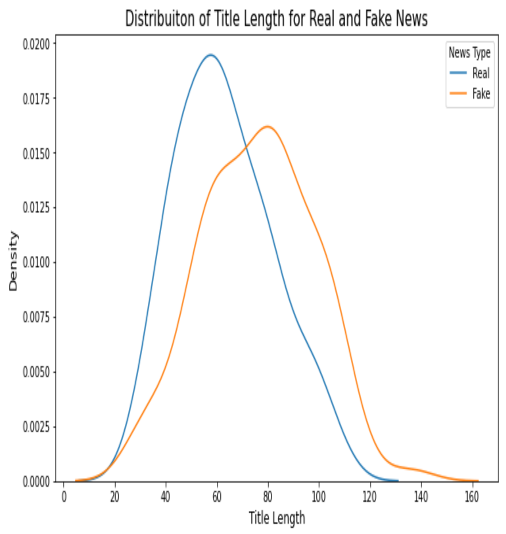


Fig. 11. Density Distribution Title Length for Real and Fake News.

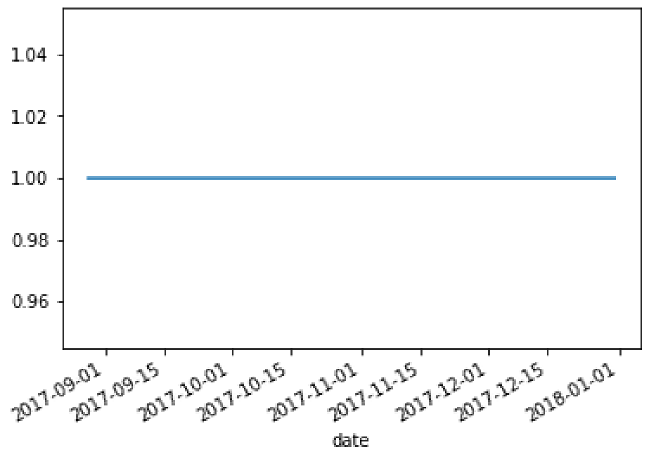


Fig. 12. Fake News Data Content Feed.

The count of news material has been autocorrelated with the time serious model developed by Arima in this research. Table II describes the autocorrelation on the FN data content, which depends on the feed's date, as depicted in Fig. 12 and X-axis denotes the date of fake news feed and Y-axis denotes the number of data. The result demonstrated that the news has been average with the date parameter. The autocorrelation of the lag that has been measured with this work has been signified in Fig. 13 and X-axis denotes the Lag Level and Y-axis denotes the Autocorrelation values. The density has also been calculated and exhibited in Fig. 14 and X-axis denotes the news content feed and Y-axis denotes the density percentage.

Table I represents the training and testing accuracy of Arima +Bi LSTM. The results show that the proposed Arima +Bi LSTM achieve a high accuracy of 0.9993. The performance has been evaluated in Fig. 15 and X-axis denotes the Epoch number and Y-axis denotes the accuracy percentage.

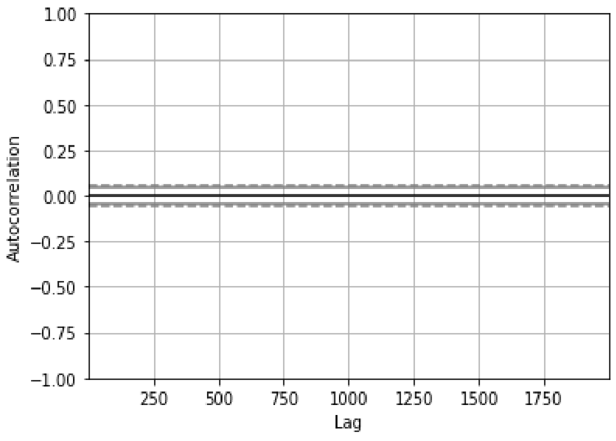


Fig. 13. Fake News Data Content Feed with Autocorrelation on Date.

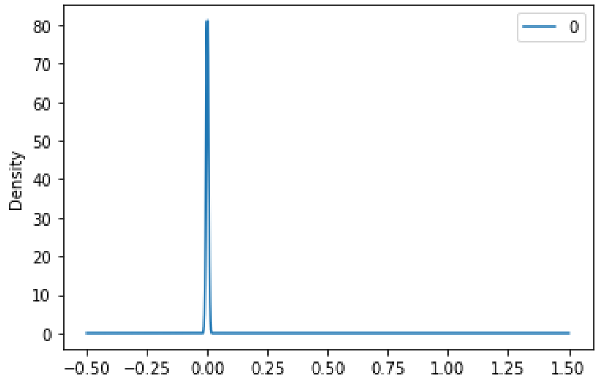


Fig. 14. Fake News Data Content Feed with Density Analysis.

TABLE I. ACCURACY ANALYSIS OF ARIMA+Bi LSTM

Epoch	**Training _Accuracy**	**Testing _Accuracy**
1	0.8964	0.9937
2	0.9993	1.0000

Table II represents the training and testing loss of Arima +Bi LSTM. The results show that the proposed Arima +Bi LSTM achieve the minimal loss of 0.0032. The performance has been evaluated in Fig. 16 and X-axis denotes the Epoch number and Y-axis denotes the loss percentage.

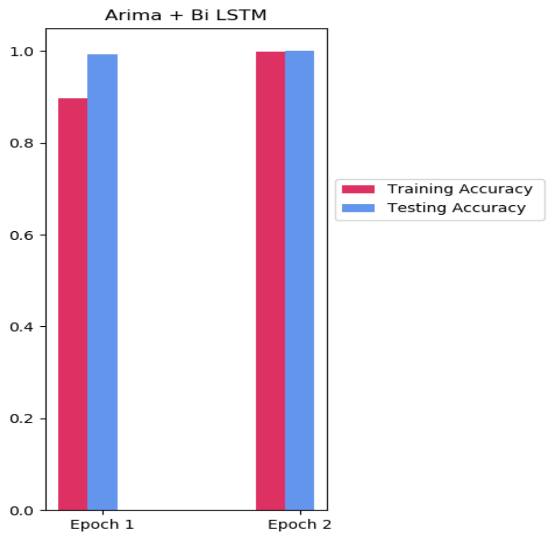


Fig. 15. Accuracy Analysis of Arima +Bi LSTM.

TABLE II. LOSS ANALYSIS OF ARIMA+Bi LSTM

Epoch	**Training _loss**	**Testing _Loss**
1	0.2872	0.0163
2	0.0032	0.0014

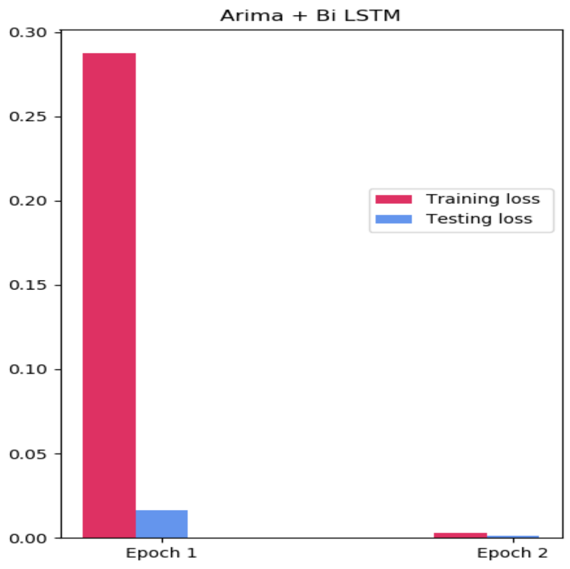


Fig. 16. Loss Analyses of Arima +Bi-LSTM.

Table III represents the training and testing accuracy of CNN. The results show that the CNN achieves an accuracy rate of 0.9676. The performance has been evaluated in Fig. 17 and X-axis denotes the Epoch numbers and Y-axis denotes the accuracy percentage.

TABLE III. ACCURACY ANALYSIS OF CNN

Epoch	Training _Accuracy	Testing _Accuracy
1	0.8502	0.9559
2	0.9319	0.9621
3	0.9450	0.9579
4	0.9592	0.9672
5	0.9676	0.9875

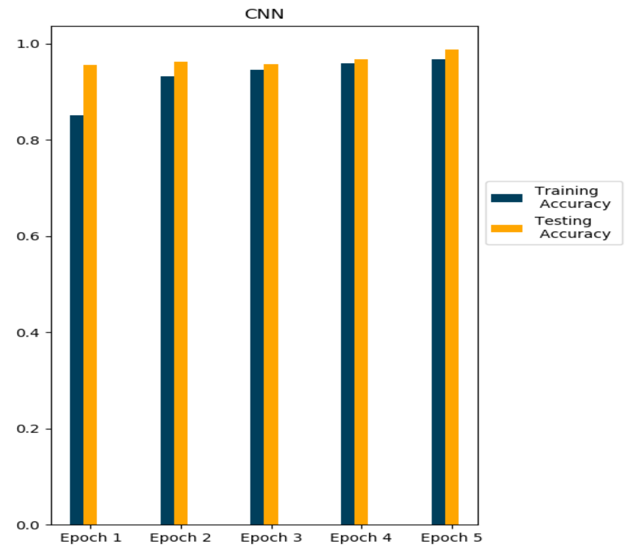


Fig. 17. Accuracy Analysis of CNN.

Table IV represents the training and testing Loss of CNN. The results show that the CNN achieves the loss rate of 0.1991. The performance has been evaluated in Fig. 18 and X-axis denotes the Epoch numbers and Y-axis denotes the loss values.

TABLE IV. LOSS ANALYSIS OF CNN

Epoch	Training _loss	Testing _Loss
1	2.8687	0.4365
2	0.3846	0.2573
3	0.2780	0.2928
4	0.2119	0.2012
5	0.1991	0.0630

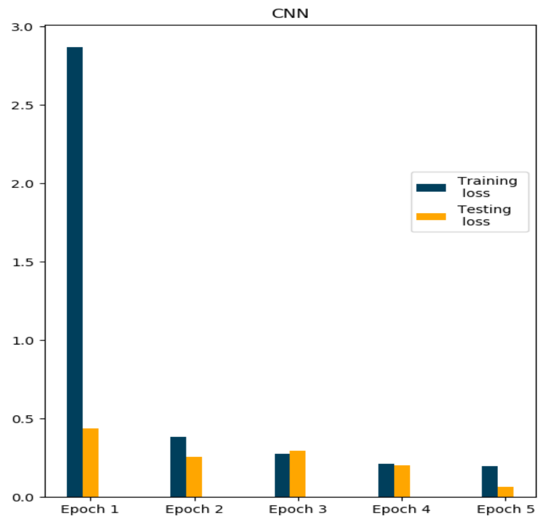


Fig. 18. Loss Analysis of CNN.

A. Evaluation Metrics

We employed many indicators to assess the performance of algorithms. The confusion matrix serves as the foundation for the majority of them. The confusion matrix is a tabular representation of the performance of a classification model on the test set, consisting of four parameters: true positive, false positive, true negative, and false negative.

1) Accuracy: Accuracy is a popular statistic representing the proportion of accurately anticipated observations, whether true or incorrect. The following equation may be used to determine the accuracy of model performance,

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (16)$$

In most circumstances, a model with a high accuracy value is a good model. Still, since we are training a classification model in this situation, an item that was predicted as true but was false (false positive) might have negative implications; similarly, an article forecasted as false but included factual data can generate trust concerns. Previously, we employed three different measures that considered the erroneously categorized observation, namely precision, recall, and F1-score.

2) Recall: Recall is a metric that indicates the total number of positive classifications outside the true class. Our example illustrates the proportion of articles expected to be true to the overall number of true articles and URLs.

$$Recall = \frac{TP}{TP+FN} \quad (17)$$

3) Precision: On the other hand, the accuracy score quantifies the ratio of true positives to all real occurrences anticipated. In our situation, precision refers to the proportion of articles classified as true among all positively predicted (true) articles,

$$Precision = \frac{TP}{TP+FP} \quad (18)$$

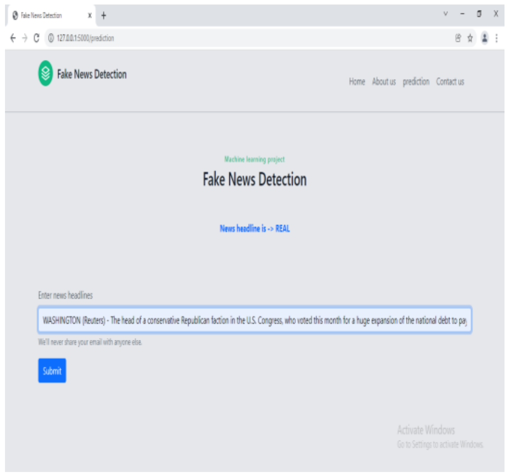
4) *F1-Score*: The F1-score is a trade-off between accuracy and recall. It computes the harmonic mean of the two. thus, it considers both false positive and false negative observations. The following formula may be used to determine the F1-score,

$$F1 - Score = 2x\frac{Precision\ X\ Recall}{Precision+Recall} \quad (19)$$

Table V represents the performance metrics of Ensemble algorithms. The results show that the classification metrics are shown in Fig. 19. The DT, RF, SVM, and NB classifiers achieve 100% accuracy. X-axis denotes the Algorithms and Y-axis denotes the accuracy percentage.

Fig. 20 represents the FN detection using any new news content and Fig. 21 represent the FN detection with the URLs on the FLASK platform. Games, presentations, animations, visualizations, webpage components, and other interactive applications can all be made with Flask. The Python GUI platform was used for this project. A flashing indicator in Flask gives a very simple way to provide feedback to a user. This method allows you to record a message after each request and only access it on subsequent requests. This is often used in conjunction with a layout template that performs the same thing.

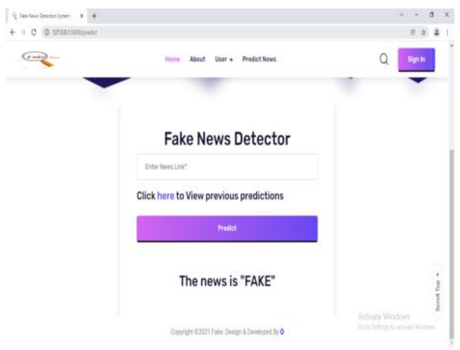
In Table VI is presented the Accuracy achieved in the existing authors and methods. And Fig. 22 represents the comparative accuracy chart for the existing author's method and proposed FNU-BiCNN method. In X-axis denotes the methods and Y-axis denotes accuracy percentage.



WASHINGTON (Reuters) - The head of a conservative Republican faction in the U.S. Congress, who voted this month for a huge expansion of the national debt to pay

We'll never share your email with anyone else.

Fig. 20. Fake News Detection with FLASK Platform.



Click here to View previous predictions

The news is "FAKE"

Copyright ©2021 Fake. Design & Developed By

Fig. 21. Fake News Detection using URL Link.

TABLE V. PERFORMANCE ANALYSIS OF ENSEMBLE MODEL

Methods	Accuracy	Precision	Recall	F-measure
DT	100	100	100	100
NB	100	100	100	100
RF	100	100	100	100
SVM	100	100	100	100
KNN	89	91	89	89
Ensemble (voting classifier)	100	100	100	100

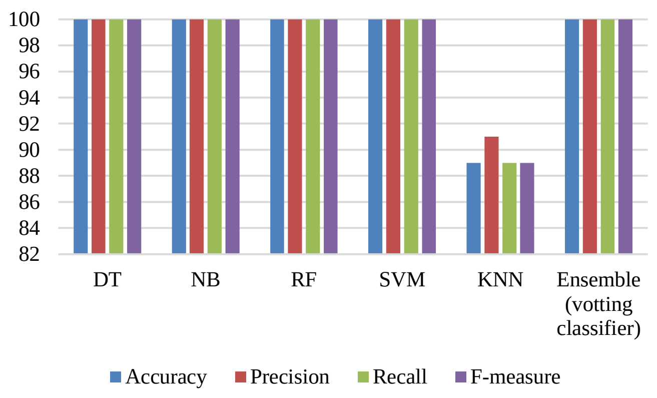


Fig. 19. Performance Analysis of Ensemble Model.

TABLE VI. COMPARATIVE ANALYSIS

Paper number	Method	Accuracy
[1]	LSTM	97%
[4]	NB and RF	81%
[6]	NB	74%
[10]	CNN+LSTM	98.3%
[12]	DT	95%
[16]	LA-MSBD	95%
Proposed	FNU-BiCNN	100%

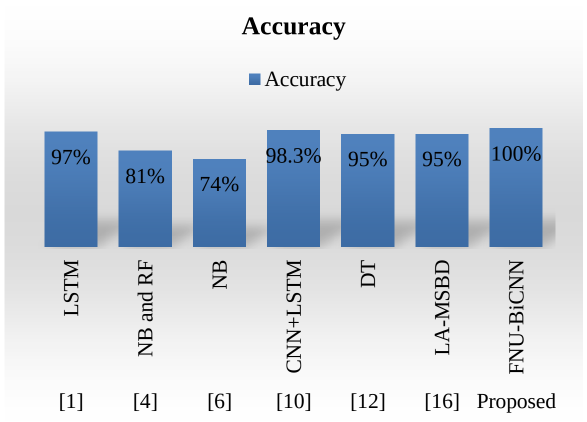


Fig. 22. Comparison Analysis for Various Authors and Proposed Method.

V. Conclusion

We attempted to evaluate fake and legitimate news in our study by thoroughly comprehending and fact-checking it. It outlines a broad methodology and the numerous aspects that affect the news's believability. Rather than studying a single strategy, we used an ensemble learning approach. The average accuracy score was 89% and 11% improvement over our worst-performing model, KNN. Additionally, we only utilized a portion of the information even with the supplied dataset and values. The FNU-BiCNN model is proposed in this article; Bi-LSTM was merged with an ARIMA model and CNN to reach the desired results, yielding a high accuracy rate of 0.9993 when combined with the other models. The proposed approach was evaluated using a variety of performance parameters, indicating that it yields an extraordinarily high accuracy rate of 100%. As the last step, our research creates an algorithm for identifying FN in the Kaggle dataset. Many messages and spam forwards confuse social network users by delivering inaccurate information. In our tests, the voting ensemble classifier came out on top regarding accuracy and precision above all other classification approaches.

Numerous outstanding difficulties with FN detection deserve the attention of researchers. For example, identifying key aspects involved in the propagation of FN is a critical first step toward reducing its spread. Similarly, real-time FN recognition in videos may be another future direction.

References

[1] Agarwal, A., & Dixit, A. (2020). Fake News Detection: An Ensemble Learning Approach. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). doi:10.1109/iciccs48265.2020.9121

[2] Ahmed, A. A., & Abdullah, N. A. (2016). Real-time detection of phishing websites. 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). doi:10.1109/iemcon.2016.7746247.

[3] Birunda, S. S., & Devi, R. K. (2021). A Novel Score-Based Multi-Source Fake News Detection using Gradient Boosting Algorithm. 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS). doi:10.1109/icais50930.2021.93958.

[4] Bhutani, B., Rastogi, N., Sehgal, P., & Purwar, A. (2019). Fake News Detection Using Sentiment Analysis. 2019 Twelfth International Conference on Contemporary Computing (IC3). doi:10.1109/ic3.2019.8844880.

[5] Cheng, W., Zhou, Y., Guo, Y., Hui, Z., & Cheng, W. (2019). Research on prediction method based on ARIMA-BP combination model. 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE). doi:10.1109/eitce47263.2019.90947.

[6] Granik, M., & Mesyura, V. (2017). Fake news detection using naive Bayes classifier. 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). doi:10.1109/ukrcon.2017.8100379.

[7] Hiramath, C. K., & Deshpande, G. C. (2019). Fake News Detection Using Deep Learning Techniques. 2019 1st International Conference on Advances in Information Technology (ICAIT). doi:10.1109/icait47043.2019.89872.

[8] Jiang, T., Li, J. P., Haq, A. U., & Saboor, A. (2020). Fake News Detection using Deep Recurrent Neural Networks. 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP). doi:10.1109/iccwamtip51612.2020.9.

[9] Jayasiriwardene, T. D., & Ganegoda, G. U. (2020). Keyword extraction from Tweets using NLP tools for collecting relevant news. 2020 International Research Conference on Smart Computing and Systems Engineering (SCSE). doi:10.1109/scse49731.2020.931302.

[10] Kaliyar, R. K. (2018). Fake News Detection Using A Deep Neural Network. 2018 4th International Conference on Computing Communication and Automation (ICCCA). doi:10.1109/ccaa.2018.8777343.

[11] Konkobo, P. M., Zhang, R., Huang, S., Minoungou, T. T., Ouedraogo, J. A., & Li, L. (2020). A Deep Learning Model for Early Detection of Fake News on Social Media*. 2020 7th International Conference on Behavioural and Social Computing (BESC). doi:10.1109/besc51023.2020.934831.

[12] Lyu, S., & Lo, D. C.-T. (2020). Fake News Detection by Decision Tree. 2020 SoutheastCon. doi:10.1109/southeastcon44009.202.

[13] Mansouri, R., Naderan-Tahan, M., & Rashti, M. J. (2020). A Semi-supervised Learning Method for Fake News Detection in Social Media. 2020 28th Iranian Conference on Electrical Engineering (ICEE). doi:10.1109/icee50131.2020.9261053.

[14] Masood, F., Ammad, G., Almogren, A., Abbas, A., Khattak, H. A., Din, I. U., … Zuair, M. (2019). Spammer Detection and Fake User Identification on Social Networks. IEEE Access, 1–1. doi:10.1109/access.2019.2918196.

[15] Qazi, M., Khan, M. U. S., & Ali, M. (2020). Detection of Fake News Using Transformer Model. 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). doi:10.1109/icomet48670.2020.9074.

[16] Rout, R. R., Lingam, G., & Somayajulu, D. V. L. N. (2020). Detection of Malicious Social Bots Using Learning Automata With URL Features in Twitter Network. IEEE Transactions on Computational Social Systems, 1–15. doi:10.1109/tcss.2020.2992223.

[17] Shantanu, Janet, B., & Joshua Arul Kumar, R. (2021). Malicious URL Detection: A Comparative Study. 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS). doi:10.1109/icais50930.2021.9396.

[18] Singh, L. (2020). Fake News Detection: a comparison between available Deep Learning techniques in vector space. 2020 IEEE 4th Conference on Information & Communication Technology (CICT). doi:10.1109/cict51604.2020.931209.

[19] Seo, Y., & Jeong, C.-S. (2018). FaGoN: Fake News Detection model using Grammatic Transformation on Neural Network. 2018 Thirteenth International Conference on Knowledge, Information and Creativity Support Systems (KICSS). doi:10.1109/kicss45055.2018.89505.

[20] Vogel, I., & Meghana, M. (2020). Detecting Fake News Spreaders on Twitter from a Multilingual Perspective. 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA). doi:10.1109/dsaa49011.2020.00084.

[21] Zhi, X., Xue, L., Zhi, W., Li, Z., Zhao, B., Wang, Y., & Shen, Z. (2021). Financial Fake News Detection with Multi fact CNN-LSTM Model. 2021 IEEE 4th International Conference on Electronics Technology (ICET). doi:10.1109/icet51757.2021.945092.