JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

"COMPARATIVE ANALYSIS OF VARIOUS FEATURE EXTRACTION TECHNIQUES USING LEARNING ALGORITHMS"

Indu Bala [1],
[1]*Research Scholar, Department of Computer Science and Application, Arni University, Kathgarh, Indora, Himachal Pradesh*
India
indubala91087@gmail.com

Dr. Sunita Mahajan [2]
[2]*Assistant Professor, Department of Computer Science and Application, Arni University, Kathgarh, Indora, Himachal Pradesh*
India
sunitamahajan2603@gmail.com

Dr. Ikvinderpal Singh [3]
[3] *Assistant Professor, PG Department of Computer Science and Applications, Trai Shatabdi GGS Khalsa College, Amritsar (Punjab).*
India
ips_sikand@yahoo.com

Abstract -- The surge of fake news across digital platforms has emerged as a pressing concern in contemporary times, jeopardizing the authenticity of public conversations and societal unity. We present an extensive examination and comparative analysis of feature extraction techniques One Hot, Bag of Words, Term Frequency-Inverse Document Frequency, and Word2Vec using two popular learning models: Multinomial Naive Bayes and LSTM. Multinomial Naive Bayes (Multinomial NB) is a variant of the Naive Bayes classifier, particularly well-suited for text classification tasks, including fake news detection, while LSTM is a type of recurrent neural network (RNN) capable of capturing sequential dependencies in textual data. We evaluate the performance of these models across different feature extraction methods using standard evaluation metrics. This paper provides valuable insights into the effectiveness of various feature extraction techniques for fake news detection, offering practical guidance for researchers and practitioners in the field.

Keywords: ***LSTM (Long short-term memory); MultinominalNB (*Multinomial Naive Bayes classifier*); Kaggle; RNN (Recurrent neural network); one hot; TFIDF(Frequency-inverse document frequency); word2vec; BoW (Bag of Words)***

I. INTRODUCTION

Feature extraction is a crucial step in machine learning and deep learning pipelines, playing a pivotal role in transforming raw data into a format suitable for learning algorithms. With the advancements in technology, various feature extraction techniques have emerged, each offering unique advantages and trade-offs. This research paper aims to provide a comprehensive comparative analysis of different feature extraction methods, including traditional machine learning techniques and deep learning-based approaches.

In today's digital era fake news refers to false or misleading information presented as genuine news. It can take various forms, including fabricated stories, manipulated images or videos, misleading headlines, and biased reporting. Fake news can be spread through traditional media channels like newspapers and television, as well as through digital platforms such as social media, websites, and messaging apps. The spread of fake news can have serious consequences, including influencing public opinion, shaping political discourse, and undermining trust in credible sources of information. It can also contribute to social polarization, incite violence, and damage reputations. Detecting fake news is a challenging task due to the proliferation of misinformation online and the evolving tactics used by purveyors of fake news. It requires a combination of human judgment, critical thinking skills, and technological tools. Some common approaches to detecting fake news include fact-checking by reputable organizations, analyzing the credibility of sources, examining the consistency of information across multiple sources, and using computational methods such as natural language processing and machine learning to identify patterns of misinformation. Efforts to combat fake news involve a multi-faceted approach that includes media literacy education, promoting critical thinking skills, improving algorithms for identifying and flagging fake news content, and fostering collaboration between technology companies, policymakers, journalists, and civil society organizations.

This paper evaluates the effectiveness of various feature extraction methods across different datasets and tasks, shedding light on their strengths, weaknesses, and applicability. In this study, we give an extensive examination and comparative analysis of feature extraction techniques using two learning models.

II. LITERATURE REVIEW

Various approaches have been explored for detecting fake news, as evidenced by research conducted by Ahmed et al. [1], which utilizes N-gram and TF–IDF methods for feature extraction, along with classifiers such as Stochastic Gradient Descent (SGD), Support Vector Machine (SVM), Linear Support Vector Machine (Linear SVM), K-Nearest Neighbor (KNN), and Decision Tree (DT). Using the ISOT Fake News dataset, Ahmed et al. achieved an accuracy of 92% by employing the Linear SVM classifier. In contrast, Ozbay and Alatas [2] employed TF–IDF exclusively for feature extraction, albeit with a broader array of classifiers including ZeroR, CV Parameter Selection (CVPS), Weighted Instances Handler Wrapper (WIHW), and DT, among others. Their approach surpassed the results reported in [1], achieving an accuracy of 96.8% and high precision, recall, and F1-scores.

Ahmad et al. [3] conducted similar research to [1], [2], comparing individual learning algorithms with ensemble learning algorithms. They evaluated Logistic Regression (LR), LSVM, Multilayer Perceptron (MLP), and KNN individually, then compared them with ensemble learning methods such as Random Forest (RF), Voting Classifier, Bagging Classifier, and Boosting Classifier across multiple datasets including ISOT Fake News Dataset [1], Fake News Dataset [4], Fake News Detection Dataset [5], and a combined dataset. Results from testing on the first dataset outperformed those in [2], with the RF algorithm achieving accuracy, precision, recall, and F1 scores of 99%, 99%, 100%, and 99%, respectively. The RF algorithm also performed well on the third and fourth datasets, with 95% and 91% accuracy, respectively.

In contrast, Kaliyar et al. [6] proposed a distinct approach to fake news detection, opting for a pre-trained word embedding (Glove) combined with a Convolutional Neural Network (CNN), rather than TF–IDF and traditional machine learning algorithms. Using the Fake News Dataset [4], their method surpassed Ahmad et al.'s study [3], achieving higher accuracy, precision, recall, and F1-score.

Bahad et al. [7] also investigated fake news detection using the Fake News Detection Dataset [5], employing GloVe pre-trained word embeddings combined with various deep learning architectures including CNN, Recurrent Neural Network (RNN), Unidirectional Long Short-Term Memory (LSTM), and Bidirectional LSTM. Results varied, with one architecture outperforming Ahmad et al.'s [3] study with 98.75% accuracy using Bidirectional LSTM. Additionally, Bahad et al. tested the Fake or Real News Dataset [8], achieving 91.48% accuracy using Unidirectional LSTM.

A year later, Deepak & Chitturi [9] conducted a similar study using the Fake or Real News Dataset [8]. They incorporated secondary features such as news domains, writers, and headlines, along with word embeddings like Bag of Words (BoW), Word2Vec, and GloVe, combined with a Feed-forward Neural Network (FNN) and LSTM. Their results showed that including secondary features positively impacted performance, with LSTM accuracy increasing by 7.6% when such features were integrated. However, despite the notable improvement, the results did not surpass those in [10].

Hadeer Ahmed et al. [11] devised a fake news detection model that integrated machine learning methods alongside n-gram analysis. Their study entailed a thorough exploration and comparison of two distinct feature extraction techniques and six machine learning classification techniques. The experimental assessments revealed that the most optimal combination involved utilizing TF-IDF for feature extraction paired with LSVM as the classifier. This particular approach yielded an impressive accuracy rate of 92%.

Uma Sharma et al. [12] introduced an approach to identify counterfeit news utilizing machine learning algorithms. Their method comprises feature extraction, data preprocessing, and classification employing a range of machine learning algorithms. The researchers observed that the SVM classifier achieved an accuracy of 94.5%.

Mykhailo Granik et al. put forward a straightforward technique for categorizing fake news, employing the Naïve Bayes algorithm [13]. Farzana Islam et al. also utilized the Naïve Bayes classifier algorithm for fake news classification, employing two feature extraction methods (TF-IDF vectorization and count vectorization) on the two fake news datasets submitted by the Kaggle community.

III. METHODOLOGY

The dataset available on Kaggle, focusing on fake news, serves as a valuable resource for detecting deceptive news content. It comprises a collection of news articles accompanied by labels indicating their authenticity. By leveraging this dataset, machine learning models can be trained to discern patterns within the text and predict the genuineness of news articles. Compiled from reputable sources like Politico, NPR, CNN, and Reuters, the dataset is meticulously curated, ensuring reliability. Additionally, it encompasses a diverse range of news articles spanning different categories such as politics, business, entertainment, and more. Below, you'll find a methodology detailing the implementation of various feature extraction and model training techniques.

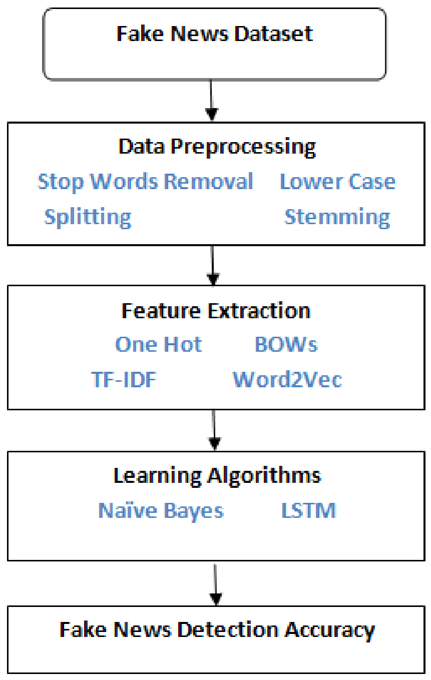

Figure 1: Steps to Detect Fake News

A. DATASET

The dataset "train.csv" available on Kaggle is frequently employed in the realm of discerning between genuine and deceptive news articles. It comprises a collection of news pieces along with corresponding labels denoting their authenticity. Leveraging this dataset, machine learning algorithms can be trained to discern patterns within the textual content, thereby enabling predictions regarding the veracity of news articles

Dataset Description

train.csv: A full training dataset with the following attributes:

id: unique ID for a news article

title: the title of a news article

author: author of the news article

text: the text of the article; could be incomplete

label: a label that marks the article as potentially unreliable

1: unreliable

0: Reliable

B. PRE-PROCESSING

In the preprocessing of a fake news dataset, we can import libraries such as PorterStemmer and stopwords from NLTK (Natural Language Toolkit) in Python. Additionally, regular expressions are utilized to efficiently manipulate and filter the textual content.

Here's a brief outline of the preprocessing steps:

Importing Libraries: Begin by importing necessary libraries such as PorterStemmer and stopwords from NLTK. These libraries provide tools for stemming and removing common words (stopwords) that may not contribute much to the analysis.

```
import re

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer
```

Loading the Dataset: Load the fake news dataset into your Python environment using pandas or a similar library.

Text Cleaning: Use regular expressions to remove any special characters, URLs, or other non-alphanumeric characters from the text data. This step ensures that only meaningful words and phrases remain for analysis.

Tokenization: Split the text into individual words or tokens. This step is crucial for further analysis as it breaks down the text into manageable units.

Stopword Removal: Utilize the stopwords list from NLTK to filter out common words like "the," "is," and "and." These words typically do not provide much information and can be safely removed.

Stemming: Apply PorterStemmer to reduce words to their root or base form. For example, "running," "runs," and "ran" would all be stemmed to "run." This step helps in reducing the dimensionality of the dataset and improves the efficiency of subsequent analyses.

```
for i in range (0, len(messages)):

   rm = re.sub('[^a-zA-Z]',' ', messages['title'][i])

   rm = review.lower()

   rm = review.split()

   rm = [ps.stem(word) for word in review if not word in stopwords.words('english')]
```

Normalization: Finally, consider additional normalization techniques such as converting all text to lowercase to ensure consistency in the dataset.

By implementing these preprocessing steps, the fake news dataset can be effectively cleaned and prepared for further analysis, ultimately enhancing the performance of machine learning models and other analytical techniques applied to the data.

C. FEATURE EXTRACTION TECHNIQUES

Feature extraction techniques like One Hot Encoding, Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec are used in fake news detection.

The choice between One Hot Encoding, Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec depends on the specific requirements of your fake news detection task and the characteristics of your dataset. Each technique has its strengths and weaknesses, and the best choice often depends on the nature of the text data and the goals of the analysis.

Let's compare these techniques:

(i) One Hot Encoding: This one-hot encoding captures the word frequency patterns in the fake news articles. It simplifies the text data for machine learning models by converting textual information into numerical vectors, allowing for tasks like text classification, sentiment analysis, and information retrieval.

Strengths:

1. Simple and computationally efficient.

2. Captures word frequency information, similar to BoW.

Weaknesses:

1. Ignores word order and semantics, similar to BoW.

2. May lead to high-dimensional and sparse feature vectors, particularly when dealing with a large vocabulary.

(ii) Bag of Words (BoW): BoW represents a document as an unordered set of words and their frequencies. It creates a feature vector with the count of each word in the document.

Use Case: BoW is simple and effective for capturing word frequency patterns in fake news articles.

Application: BoW is used for tasks like text classification, sentiment analysis, and information retrieval. It simplifies text data for machine learning models by converting textual information into numerical vectors.

Strengths:

1. Simple and computationally efficient.

2. Captures word frequency information.

Weaknesses:

1. Ignores word order and semantics.

2. May lead to high-dimensional and sparse feature vectors.

(iii) Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF measures the importance of a word in a document relative to its frequency across the entire dataset. It combines term frequency (TF) and inverse document frequency (IDF).

Use Case: TF-IDF is useful for highlighting distinctive terms that may indicate the uniqueness of content.

Application: TF-IDF is widely used in information retrieval, document classification, and text mining. It helps in capturing the importance of words in a document relative to the entire corpus.

Strengths:

1. Considers both local and global importance.

2. Helps identify significant terms by down weighting common words.

Weaknesses:

1. Ignores word order and semantics.

2. May not capture the semantic relationships between words.

(iv) Word2Vec: Word2Vec represents words as dense vectors in a continuous vector space, capturing semantic relationships. Word embeddings represent words as dense vectors in a continuous vector space. These vectors capture semantic relationships between words.

Use Case: Word embeddings capture context and semantic meaning, which can help in understanding the subtle nuances of language and detecting deceptive content.

Application: Word2Vec is commonly used in natural language processing tasks such as language translation, sentiment analysis, and document clustering. It provides dense vector representations that can preserve semantic relationships.

Strengths:

1. Captures semantic information and word context.

2. Can capture complex relationships and analogies.

Weaknesses:

1. Requires large amounts of data for effective training.

2. May struggle with out-of-vocabulary words.

D. LEARNING ALGORITHMS

(i) Multinomial Naive Bayes Classifier

Multinomial Naive Bayes (MNB) is a variation of the Naive Bayes classifier specifically designed for text classification tasks where the features (input variables) represent word counts or term frequencies. It assumes that the features are generated from a multinomial distribution.

Here's how the Multinomial Naive Bayes classifier works:

1. *Feature Representation***:** In text classification tasks, documents are represented as feature vectors, where each feature represents the frequency of occurrence of a term (word) in the document. The vocabulary of terms is typically constructed from the entire corpus of documents in the dataset.
2. *Probability Estimation***:** MNB applies Bayes' theorem to estimate the probability of each class given the observed features. The classifier calculates the conditional probability of each feature given the class and the prior probability of each class.
3. *Parameter Estimation:* MNB estimates the parameters of the model from the training data. This involves calculating the probabilities of each term occurring in each class. For each term in the vocabulary, the classifier estimates the probability of observing that term given each class.
4. *Smoothing:* To handle the issue of zero probabilities (i.e., when a term is not present in the training data of a particular class), MNB typically employs smoothing techniques such as Laplace smoothing or add-one smoothing. Smoothing helps to avoid assigning zero probability to unseen terms in the test data.
5. *Classification Decision:* Given a new document, the Multinomial Naive Bayes classifier calculates the posterior probability of each class given the observed feature vector. It selects the class with the highest posterior probability as the predicted class for the document. Multinomial Naive Bayes is a simple yet effective probabilistic classifier for text classification tasks. Despite its "naive" assumption of feature independence, it often performs well in practice and serves as a strong baseline model for many text classification problems.

(ii) Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture that is designed to overcome the limitations of traditional RNNs in capturing long-term dependencies in sequential data. LSTM is a powerful and versatile architecture for modeling sequential data, capable of capturing long-term dependencies and achieving state-of-the-art performance in a wide range of applications.

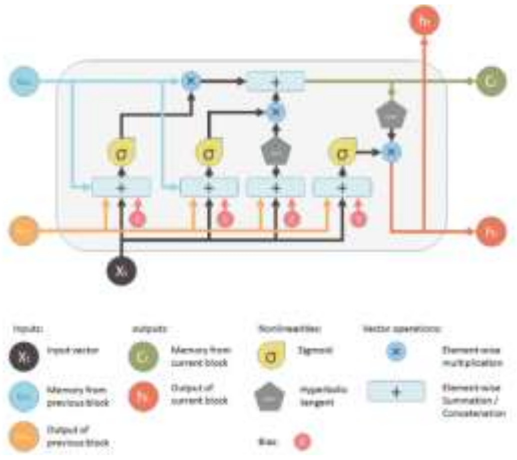

Image Source: https://blog.mlreview.com/

Figure 2: LSTM Model

The Long Short-Term Memory (LSTM) network is a type of recurrent neural network (RNN) architecture designed to address the vanishing gradient problem and capture long-term dependencies in sequential data.

Here's how LSTM works:

1. *Sequential Data Processing:* Like traditional RNNs, LSTM is designed to process sequences of data inputs. Each input in the sequence is processed one at a time, and the network maintains a hidden state that captures information from previous inputs.
2. *Memory Cells:* The key innovation of LSTM is the introduction of memory cells, which allow the network to selectively store and access information over long periods. These memory cells are composed of multiple layers of neural network units (or cells) that interact with each other in a controlled manner.
3. *Gates***:** LSTM uses three types of gates to control the flow of information: input gate, forget gate, and output gate. These gates are implemented as special types of neural network layers that take input from the current input, the previous hidden state, and the current memory cell state, and produce output signals that regulate the flow of information.
4. *Input Gate:* Controls how much new information is added to the memory cell.
5. *Forget Gate***:** Controls how much of the existing memory cell state is retained or forgotten.
6. *Output Gate:* Determines how much of the memory cell state is used to compute the output of the LSTM unit.
7. *Cell State:* In addition to the hidden state, LSTM maintains a separate memory cell state that carries information over multiple time steps. This cell state can be updated or modified by the gates based on the input and previous states.
8. *Training:* During the training process, the parameters (weights and biases) of the LSTM network are learned using backpropagation through time (BPTT), which is a variant of backpropagation tailored for sequential data. The network learns to adjust its parameters to minimize the difference between its predicted outputs and the true outputs.
9. *Long-Term Dependencies:* One of the main advantages of LSTM is its ability to capture long-term dependencies in sequential data. This is achieved through the use of memory cells and gated mechanisms, which allow the network to selectively store and access information over multiple time steps.

LSTM is a powerful and versatile architecture for modeling sequential data, capable of capturing long-term dependencies and achieving excellent performance in a wide range of applications.

IV. RESULTS AND DISCUSSION

We are evaluating the performance of Multinomial Naive Bayes and LSTM models across different feature extraction techniques. Assess metrics such as accuracy to measure the effectiveness of each combination.

Choosing between Multinomial Naïve Bayes Classifier and Long Short-Term Memory Networks (LSTMs) for a specific task depends on the nature of the data and the task requirements. The Multinomial Naïve Bayes classifier algorithm plays a significant role in fake news detection due to its simplicity, efficiency, and effectiveness, especially in scenarios with limited computational resources and large datasets. LSTMs excel in handling sequential dependencies, making them powerful for tasks involving time series or natural language processing. If your data has temporal dependencies or involves sequences, LSTMs might be more appropriate.

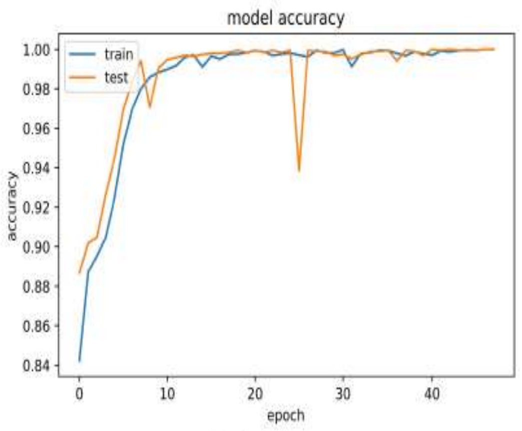

Figure 3: Word2Vec +LSTM

When we are using the word embedding technique Word2Vec and LSTM it gives high accuracy to detect fake news.

Performance Evaluation Metrics

To evaluate our best model, we use accuracy, precision, recall, and F1-score as evaluation metrics.

Precision = True Positive/ (True positive + False Positive)

Recall = True Positive/ (True Positive + False Negative)

$$\text{F1-Score} = \frac{2*(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

As we implement each feature extraction technique

Table 1: Evaluation table

Model	***Accuracy***	***Precision***	***Recall***	***F1-Score***
Word2Vec+ LSTM	**97%**	**96.6%**	**97.4%**	**97%**

and train the Multinomial Naive Bayes classifier and LSTM models separately. Utilize appropriate evaluation techniques such as cross-validation or train-test splits to assess model performance. Compare results across feature extraction methods and models to identify the most suitable approach for the specific task or dataset. By conducting such a comparative analysis, researchers and practitioners can gain insights into the strengths and weaknesses of different feature extraction techniques and machine learning models, aiding in informed decision-making for text analysis tasks.

Table 2: Comparison Table

Feature Extraction Techniques	***Accuracy (Multinomial Naïve Bayes Classifier)***	***Accuracy (Long Short-Term Memory)***
One Hot	0.71	0.85
BOW	0.9	0.92

TF-IDF	0.88	0.9
Word2Vec	0.92	0.97

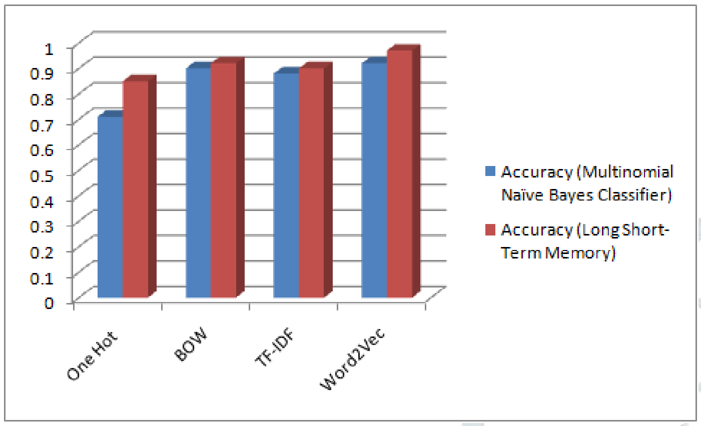

Figure 4: Comparison chart of MultinomialNB and LSTM

V. CONCLUSION AND FUTURE WORK

In this paper, we can use four feature extraction techniques one hot, Bag of words, TF-IDF, and Word2vec, and two models MultinomialNB and LSTM on a large dataset from Kaggle. After the final result, we conclude that the performance of the word2vec features extraction technique works very well with the LSTM-trained model. It gives an accuracy of 0.97 to detect fake news which is a better value. For the future, we are prepared to use a combination of two or more different deep learning models to detect fake news on large as well as small datasets. Already work done on this topic is enormous but with the rise in social media activities on political issues it is a hot topic to work on further for better results.

CONFLICT OF INTEREST

The authors declare no conflict of interest exists.

REFERENCES

[1] Ahmed H., Traore I., Saad S. Detection of Online Fake News using N-Gram Analysis and Machine Learning Techniques, Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) (2017), pp. 127-138, https://doi.org/10.1007/978-3-319-69155-8_9

[2] Ozbay F.A., Alatas B. Fake news detection within online social media using supervised artificial intelligence algorithms Phys. A Stat. Mech. Appl., 540 (2020), Article 123174, https://doi.org/10.1016/j.physa.2019.123174

[3] Ahmad I., Yousaf M., Yousaf S., Ahmad M.O. Fake news detection using machine learning ensemble methods Complexity, 2020 (2020), pp. 1-11, https://doi.org/10.1155/2020/8885861

[4] UTK Machine Learning Club Fake news (2018) https://www.kaggle.com/c/fake-news/overview (accessed June 3, 2020)

[5] Jruvika Fake news detection (2017) https://www.kaggle.com/jruvika/fake-news-detection/version/1

(accessed June 3, 2020)

[6] Kaliyar R.K., Goswami A., Narang P., Sinha S. FNDNet – A deep convolutional neural network for fake news detection Cogn. Syst. Res., 61 (2020), pp. 32-44, https://doi.org/10.1016/j.cogsys.2019.12.005

[7] Bahad P., Saxena P., Kamal R. Fake news detection using bi-directional LSTM-recurrent neural network Procedia Comput. Sci., 165 (2019), pp. 74-82, https://doi.org/10.1016/j.procs.2020.01.072

[8] McIntire G. Fake or real news (2017) https://github.com/joolsa/fake_real_news_dataset (accessed June 3, 2020)

[9] Deepak S., Chitturi B. Deep neural approach to fake-news identification Procedia Comput. Sci., 167 (2020), pp. 2236-2243, https://doi.org/10.1016/j.procs.2020.03.276 This article is free to access.

[10] De Paor S., Heravi B. Information literacy and fake news: How the field of librarianship can help combat the epidemic of fake news J. Acad. Librariansh., 46 (2020), Article 102218, https://doi.org/10.1016/j.acalib.2020.102218

[11] Issa Traore, Hadeer Ahmed and Sherif Saad, "Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques" International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments.

[12] Sidarth Saran, Uma Sharma, and Shankar M. Patil. "Fake News Detection Using Machine Learning Algorithms".International Journal of creative research thoughts(IJCRT) 2020.

[13] M.Granik, M ykhailo, and V. Mesyura. "Fake news detection using naive Bayes classifier." 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). IEEE, 2017.