Fake News Detection Using Machine Learning and Deep Learning Methods

張祐嘉
thsg314@gmail.com

劉益鳴
morris46202@gmail.com

朱冠宇
wuten770623@gmail.com

陳柏瑜
41047054s@gapps.ntnu.edu.tw

張鐙友
ivanty2chang@gmail.com

***Abstract*—In recent years, the proliferation of fake news has emerged as a significant challenge, impacting public opinion, democratic processes, and social stability. Fake news detection has thus become a critical area of research, leveraging advances in natural language processing (NLP), machine learning, and artificial intelligence. This paper explores various methodologies employed in the automatic detection of fake news, including supervised and unsupervised learning techniques, feature extraction methods, and the integration of deep learning models such as Convolutional Neural Networks (CNNs), this paper also validates the results by conducting cross-comparison using two different datasets. Through a comprehensive review of existing literature and the implementation of novel detection frameworks, this study aims to contribute to the development of robust and scalable solutions for mitigating the spread of misinformation.**

I. Introduction

With the advancement of technology, opportunities for human beings to access information have greatly increased. Tools such as desktop computers, laptops, mobile phones, and tablets, combined with the rapid development of the internet, allow people to learn about each other's current status in a very short time even separated by miles far. The evolution of information technology and the internet has shortened the spatial and temporal distances in contemporary society. Everyone can be both a receiver and a provider of information, the ease of accessing information today has led to the proliferation of fake news, which can closely resemble the truth. Coupled with the advancement of social media, fake news often appears identical to real news. Therefore, detecting and preventing fake news has become an important issue.

The word fake news becomes a trend happened in 2017, which is the year after America president election. At that time, various rumors said one of the candidates had used fake news to spread false information and or incorrect news. From that year on, fake news and how to detect fake news has become a war continues nowadays. [1] [2]

Medias can be classified into many types. From traditional media like TV or radio to bloggers or KOL (Key Opinion Leader). Therefore, a person can have a variety of ways to receive information more than decades ago, including fake news. An FND is hard to deployed to every kind of it, and so is ours. Hence, we focus on more like online newspaper, which contains title, context and so on that we usually see on the Internet.

The rest of paper is organized as follows. Section II presents the methodologies used on FND. Our data source and searching method are provided in section II. Section IV provides the experiment result and corresponding analysis. Finally, section V concludes the paper.

II. Methodology on FND

In this section, we present some methodologies adapted in FND, including none coding method and coding method.

A. None Coding Method

To effectively tackle the challenge of fake news, it is crucial to enhance individual critical intuition and logical reasoning skills. Creating intuitive conflicts can activate the brain's reasoning process, helping people to approach information more cautiously and critically.

B. Coding Methods

To effectively tackle the challenge of fake news by using artificial intelligence, we should preprocess the news or articles to training data. Then, train the model with the data.

1) Word Embedding
Word embedding is a type of mapping. It takes a word from the text space and maps it into another numerical vector space under certain method. The input of word embedding is a set of non-repetitive vocabularies while the output are corresponding vectors for each word in the input. For instance, if the initial string is "an apple on an apple tree". The repeated words will first be removed to get [an, apple, on, tree]. This produces an output that indicates the position of each word in the dictionary. For example, "an" would be [1,0,0,0] while tree would be [0,0,0,1]. Through this method, each word can be converted into a numerical vector and be applied for subsequent machine learning steps. The embedding method is divided into two categories, either embedding based on frequency or on prediction. Count vector and term frequency-inverse document frequency (TF-IDF) are method of word embedding based on frequency while Word2Vec, Doc2Vec, and GloVe are based on prediction. We will give a brief explanation of GloVe in the subsequent section.

2) Global Vector for Word Representation (GloVe) [3]
GloVe is a word representation tool based on global word frequency statistics. It represents a word as a vector composed of real numbers capturing semantic characteristics, such as similarity and analogy, between words. By performing operations like Euclidean distance or cosine similarity on these vectors, we can calculate the semantic similarity between two words.

3) Machine learning methods
Machine learning (ML), a subset of artificial intelligence (AI), enables computers to perform tasks without explicit programming by learning from data. It encompasses various algorithms and models, such as linear regression for predicting continuous variables, logistic regression for binary classification, decision trees for interpretable decision-making, random forests for enhanced accuracy, support vector machines (SVM) for optimal classification, K-means clustering for grouping data, and neural networks for complex tasks like image and speech recognition. Key processes include training on labeled data and testing on new data to evaluate performance. ML types include supervised learning, which uses labeled data for tasks like spam detection and house price prediction; unsupervised learning, which finds patterns in unlabeled data for applications like customer segmentation and market basket analysis; semi-supervised learning, combining small labeled datasets with large unlabeled ones; and reinforcement learning, where models learn through rewards and penalties, often used in gaming and robotics. ML is applied in natural language processing (NLP) for chatbots and sentiment analysis, computer vision for facial recognition and autonomous vehicles, recommendation systems for personalized content, fraud detection in finance, and healthcare for disease prediction and personalized treatment. However, challenges like data quality and quantity, overfitting, computational demands, and interpretability must be addressed to maximize ML's potential.
4) Deep learning methods
Deep learning, a specialized subset of machine learning, uses multi-layer neural networks to mimic the structure and function of the human brain, enabling computers to learn from large datasets and perform complex tasks with high accuracy. Neural networks consist of input, hidden, and output layers, utilizing activation functions to introduce non-linearity, and are trained through forward propagation and backpropagation, with optimization algorithms like stochastic gradient descent and Adam adjusting weights. Types of deep learning models include Convolutional Neural Networks (CNNs) for image data, Recurrent Neural Networks (RNNs) for sequential data, Long Short-Term Memory networks (LSTMs) for long-term dependencies, Generative Adversarial Networks (GANs) for generating new data, and autoencoders for data compression and noise reduction. Applications span computer vision, natural language processing, autonomous driving, healthcare, and recommendation systems. However, deep learning faces challenges such as high data and computational resource requirements, poor model interpretability, overfitting, and ethical concerns. Despite these challenges, deep learning plays a crucial role in advancing artificial intelligence, offering innovative and efficient solutions across various fields.

III. Data Source and Search Method

In this section, we provide the data set sources and searching methods.

A. Dataset

In this paper, experiments were conducted using two sets of data set from Kaggle. The first data set [4] consists of two files, *fake.csv* and *true.csv*, with mostly polotical news. *fake.csv* is the file that holds all the required information (title/text/subject/date/label) along with a label of 1 while *true.csv* has the same columns as *fake.csv*, but holds the information with label 0. The second data set [5] is from a Kaggle competition and only consists of one file *train.csv*, containing the information of both true and fake data (id/title/author/text/label) with much more various subjects compared with the first dataset. We will use both datasets as training dataset and testing dataset or mix them together and analysis each case in section IV.

B. Preprocess

Before applying the methods to our data, the data will first go through a series of preprocessing as follows:

1) Data clearing
Since we want to focus on the 'text' task, both data set will only be remaining of columns (text/label) after data clearing, and while dropping the unused columns, we would also remove the data consisting of wrong values or having invalid format.
2) Text clearing
When doing text clearing, we would first ensure the paragraph would be in lower case, then remove stop words to lower the effect of them. Last, we would turn the paragraph into single word tokens in order to apply word vectorizing.
3) Word vectorizing
In this part, a paragraph of word tokens would be transferred into word vectors based on preloaded GloVe word vector dictionary (*glove.6B.100d*), and the output array would have a dimension of 100.

C. Model

1) K-Nearest-Neighbors (KNN)
The KNN model is a nonparametric classification algorithm. It is known for its simplicity and effectiveness. When training, a labeled training dataset is provided where the data points are categorized into various

classes, then the distance between the data points will be used to classify the new input data.We set the numbers of estimators = number of features = 100.

2) Random Forest (RF) [6]
Randomforest is also a machine learning method that combines multiple decision trees to achieve the given task. In each tree, a random subset of feature values will go through the tree, where each node represents a feature, and each branch represents a decision rule. When reaching a leaf node, the tree then makes decision based on the label or the value. In this paper, we set the numbers of estimators = number of features = 100.

3) Convolutional Neural Networks (CNN)
The CNNs are a class of deep neural networks, mainly built of Convolutional Layer, Pooling Layer, and Fully Connected Layer. In this paper, we will be using the CNN structure we built as shown in tableI, and tableII also shows the hyperparameter of our CNN model.

TABLE I
STRUCTURE FOR CNN

Layers	Number of parameters
conv1d(Conv1D)	256
max_pooling1d (MaxPooling1D)	0
flatten (Flatten)	0
dense (Dense)	401536
dense_1 (Dense)	129

TABLE II
HYPERPARAMETERS FOR CNN

Hyperparameters	Description or value
No. of convolution layers	1
No. of max pooling layers	1
No. of dense layers	2
Loss function	Binary_crossentropy
Activation function	Relu
Optimizer	Adam
Output layer	Sigmoid
Number of epochs	50
Batch size	128

4) Long Short-Term Memory (LSTM) [7]
LSTM layers are a type of recurrent neural network (RNN) layer designed to address some of the limitations of traditional RNNs.It consists a mamory cell to store and retrieve information over long sequeces , which also controls what information is allowed to enter/leave a cell.

5) FNDNET [8]
Since we want to compare our model with deep CNN, we also implemented FNDNET to try to find out if the performance is good enough when using the same dataset. The activation function we chose is ”sigmoid” since it suits better compared to ”softmax” in binary classification tasks.

TABLE III
STUCTURE FOR FNDNET

Layers	Number of parameters
conv1d_1(Conv1D)	38528
conv1d_2(Conv1D)	51328
conv1d_3(Conv1D)	64128
max_pooling1d_1 (MaxPooling1D)	0
max_pooling1d_2 (MaxPooling1D)	0
max_pooling1d_3 (MaxPooling1D)	0
concatenate_1 (Concatenate)	0
conv1d_4(Conv1D)	82048
max_pooling1d_4 (MaxPooling1D)	0
conv1d_5(Conv1D)	82048
max_pooling1d_5 (MaxPooling1D)	0
flatten_1 (Flatten)	0
dense_2 (Dense)	49280
dense_3 (Dense)	258

TABLE IV
HYPERPARAMETERS FOR FNDNET

Hyperparameters	Description or value
No. of convolution layers	5
No. of max pooling layers	5
No. of dense layers	2
Loss function	Binary_crossentropy
Activation function	Relu
Optimizer	Adam
Output layer	Sigmoid
Number of epochs	3
Batch size	128

D. Analysis

In order to compare the performance of the models we mentioned previously, we will be using some commonly used classification metrics, such as confusion matrix, F_1-score and accuracy.

1) Confusion matrix
Confusion matrix consists of four elements, true positive (TP), false positive (FP), true negative (TN) and false negative (FN). Through these elements, we can see how precisely our model can predict the fake news.
   a) True Positives (TP) are the instances where the model correctly predicts the positive class. High TP values indicate that the model is effective at identifying positive cases. In our scenario, it means that the model is able to identify the fake news correctly.
   b) False Positives (FP) are the instances where the model incorrectly predicts the positive class. In our scenario, it means that the model is not able to identify the fake news correctly.
   c) True Negatives (TN) are the instances where the model correctly predicts the negative class. High TN values reflect the model's accuracy in recognizing negative cases. In our scenario, it means that the model is able to identify the real news correctly.
   d) False Negatives (FN) are the instances where the model incorrectly predicts the negative class. This happens when the model fails to identify fake

news and predicts them as authentic ones. In our scenario, it means that the model is not able to identify the real news correctly.

2) F_1-Score

F_1-score consists of another two elements, precision and recall.

a) Precision (Positive Predictive Value) indicates the proportion of positive identifications that are actually correct. Equation 1 shows how to calculate the precision.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

b) Recall2 (Sensitivity or True Positive Rate) measures the proportion of actual positives that are correctly identified by the model. Equation 2 indicates how to calculate the recall.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

F_1-score, showed in equation 3, is the harmonic mean of precision and recall, providing a single metric that balances both the false positives and false negatives. It's particularly useful when dealing with imbalanced classes, as it gives a more nuanced view of the model's performance than accuracy alone.

$$F_1 = \frac{2 \times (Precision \times Recall)}{(Precision + Recall)} \tag{3}$$

3) Accuracy

The accuracy is an intuitive metric that represents the proportion of correctly classified instances out of the total instances. We will be using this metric to show how good a model is at identifying fake news. Equation 4 indicated how to calculate the accuracy.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \tag{4}$$

IV. Result and Analysis

Below are the respective performance results of the different machine learning and deep learning methods we proposed in section II. First, we have the corresponding confusion matrices here that will help us yield the other metrics.

For all the methods we implemented here, we utilized the pre-trained word-embedding method GloVe, since it was showed in previous papers that it performs better than other word embedding models. We will be comparing the results of machine learning and deep learning method here, also testing the model with different datasets that differs from the training dataset. Table V - IX shows the precision, recall, and F_1-score with different dataset combinations under five training models. Table V uses the ISOT dataset as training and testing dataset. Table VI uses the ISOT dataset as training dataset while Kaggle as testing dataset. Table VII uses the Kaggle dataset as training dataset while ISOT as testing dataset. Table VIII uses the Kaggle dataset as both training and testing dataset. Finally, Table IX shows the mixed dataset as training and testing dataset. Table X - XIV shows the accuracy with different dataset combinations under five training models. Table X uses the ISOT dataset as training and testing dataset. Table XI uses the ISOT dataset as training dataset while Kaggle as testing dataset. Table XII uses the Kaggle dataset as training dataset while ISOT as testing dataset. Table XIII uses the Kaggle dataset as both training and testing dataset. Finally, Table XIV shows the mixed dataset as training and testing dataset.

For machine learning part, we implemented the KNN and random forest models. Using KNN as a machine learning method, we initially expected it to not perform well, but it still achieved 92% and 84% accuracy when testing on the same dataset of ISOT and kaggle respectively. As for random forest, the model performed slightly better than KNN, with a result of 95% and 86% respectively.

For deep learning part, we implemented CNN, LSTM, and FNDNET. When training and testing with the same datasets, we found out that among all the models, the FNDNET model performs the best, with a testing accuracy of 100% and 97% with 3 epochs. CNN performs also rather good with a 96.2% and 87% in testing accuracy with 50 epochs. Using LSTM as the model, we obtained a 93.5% and 86% testing accuracy also with 50 epochs.

To evaluate the models' ability to predict result on different dataset, we prepared two datasets with different dataset. ISOT is a larger dataset that mostly consisted of political news, while the Kaggle one is a smaller one that contains global news of all sort. We trained the model with one and tested the other, and we found the results in tables XI and XII.

In table XI and XII, we can see the decrease in accuracy when we switched the training and testing dataset. Even though the Kaggle one is much smaller than the ISOT one, the accuracy of testing Kaggle on ISOT is higher than testing ISOT on Kaggle. We think the reason behind this is because the Kaggle dataset has a broader range of categories including the political ones, while the ISOT doesn't.

One thing we noticed is that the FNDNET performs poorly, which contradicts what the original paper suggested. The author of the paper suggested that when the value of gradients diminishes by adding new layers, it should be better at preventing over-fitting problems.

TABLE V
F1-SCORE OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *ISOT* AND TESTING DATASET *ISOT*

Classification model	Precision	Recall	F1-score
KNN	92	92	92
RF	95	95	95
CNN	97	97	97
LSTM	94	94	94
FNDNET	100	100	100

V. Conclusion

In conclusion, when we test the results with the same datasets, FNDNET performs the best, but other machine

TABLE VI
F1-SCORE OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *ISOT* AND TESTING DATASET *Kaggle*

Classification model	**Precision**	**Recall**	**F1-score**
KNN	71	59	61
RF	72	57	60
CNN	63	59	55
LSTM	62	59	57
FNDNET	57	52	41

TABLE VII
F1-SCORE OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *Kaggle* AND TESTING DATASET *ISOT*

Classification model	**Precision**	**Recall**	**F1-score**
KNN	79	62	65
RF	70	66	66
CNN	70	70	70
LSTM	69	68	68
FNDNET	61	60	58

TABLE VIII
F1-SCORE OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *Kaggle* AND TESTING DATASET *Kaggle*

Classification model	**Precision**	**Recall**	**F1-score**
KNN	84	82	82
RF	86	86	86
CNN	88	87	87
LSTM	86	86	86
FNDNET	97	97	97

TABLE IX
F1-SCORE OF FIVE DIFFERENT MODELS WITH MIXED TRAINING DATASET AND MIXED TESTING DATASET

Classification model	**Precision**	**Recall**	**F1-score**
KNN	86	84	84
RF	89	88	88
CNN	91	91	91
LSTM	88	88	88
FNDNET	98	98	98

TABLE X
ACCURACY OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *ISOT* AND TESTING DATASET *ISOT*

Classification model	**Accuracy**
KNN	92
RF	95
CNN	97
LSTM	94
FNDNET	100

TABLE XI
ACCURACY OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *ISOT* AND TESTING DATASET *Kaggle*

Classification model	**Accuracy**
KNN	59
RF	57
CNN	59
LSTM	59
FNDNET	52

TABLE XII
ACCURACY OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *Kaggle* AND TESTING DATASET *ISOT*

Classification model	**Accuracy**
KNN	62
RF	66
CNN	70
LSTM	68
FNDNET	60

learning and deep learning methods did not performed as bad as we initially thought. However, when we change the test data to another dataset, the accuracy of all models drops to around 50-60%. This indicates that our model may have encountered some batch effect (or in other words, lack of robustness). Therefore, when this model encounters fake news that doesn't exist in the training data, it cannot precisely distinguish the fake news. This situation can be clearly found when comparing XI and XII ,since the latter has a smaller amount of data when training. Which implies there is room for improvement when using multi-datasets.

REFERENCES

[1] I. Ali, M. N. B. Ayub, P. Shivakumara, N. F. B. M. Noor *et al.*, "Fake news detection techniques on social media: A survey," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

TABLE XIII
ACCURACY OF FIVE DIFFERENT MODELS WITH TRAINING DATASET *Kaggle* AND TESTING DATASET *Kaggle*

Classification model	Accuracy
KNN	82
RF	86
CNN	87
LSTM	86
FNDNET	97

TABLE XIV
ACCURACY OF FIVE DIFFERENT MODELS WITH MIXED TRAINING DATASET AND MIXED TESTING DATASET

Classification model	Accuracy
KNN	84
RF	88
CNN	91
LSTM	88
FNDNET	98

[2] M. F. Mridha, A. J. Keya, M. A. Hamid, M. M. Monowar, and M. S. Rahman, “A comprehensive review on fake news detection with deep learning,” *IEEE access*, vol. 9, pp. 156 151–156 170, 2021.
[3] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
[4] E. BOZKUŞ, “Fake news detection datasets,” 2023. [Online]. Available: https://www.kaggle.com/datasets/emineyetm/fake-news-detection-datasets/data
[5] W. Lifferth, “Fake news,” 2018. [Online]. Available: https://kaggle.com/competitions/fake-news
[6] L. Breiman, “Random forests,” *Machine learning*, vol. 45, pp. 5–32, 2001.
[7] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[8] R. Kaliyar, A. Goswami, P. Narang, and S. Sinha, “Fndnet- a deep convolutional neural network for fake news detection,” *Cognitive Systems Research*, vol. 61, 06 2020.