Model Comparison on Fake News Detection

Chen-Yang Yu
Department of AI Robotics
National Cheng Kung University

Shang-Yuan Chuang
Department of Computer Science
National Cheng Kung University

Chiu-Chang Cheng
Department of Intelligent Computing
National Cheng Kung University

Ya-Chi Chan
Department of AI Robotics
National Cheng Kung University

***Abstract*—We compared the application and performance of different methods in the task of fake news detection using a purely content-based dataset. We also experimented with integrating NLP methods (such as BERT and TF-IDF), traditional machine learning algorithms, and Graph Neural Networks (GNNs) at various stages of upstream and downstream tasks.**

Our project aims to inspect the model performance on fake news detection. https://github.com/LittleFish-Coder/fake-news-detection-model-comparison

I. Introduction

The detection of false news is critically important due to its capacity to spread swiftly and pervasively across social networks, impacting public opinion, influencing political outcomes, and causing economic fluctuations. Quantitative evidence underscores the scale and speed of this phenomenon. According to a study of Soroush Vosoughi, Deb Roy and Sinan Aral, false news cascades reached between 1,000 and 100,000 people within the top 1% of such cascades, while the truth rarely reached beyond 1,000 individuals. Additionally, false news reached 1,500 people six times faster than the truth and achieved significant network depths—19 hops deep—almost ten times quicker than true stories reached 10 hops.

These findings highlight the substantial challenge false news poses, as it outpaces the truth in both reach and velocity, underscoring the urgent need for effective detection mechanisms. Tools and strategies for combating false news are essential not only for safeguarding public discourse but also for maintaining the integrity of democratic processes and economic stability. The deployment of sophisticated algorithms to analyze and flag false content, combined with public education initiatives about media literacy, are vital components in tackling this issue.

II. Dataset

The Kaggle Fake News KDD 2020 competition focuses on developing machine learning algorithms to detect fake news. The dataset provided for this challenge consists of news articles labeled as real or fake, which participants use to train their models. The goal is for participants to create a system that can effectively identify fake news.

A. train.csv

This dataset is used to train machine learning models and contains the following attributes:

1) text: The complete text of the news article.
2) label: This is a binary label indicating the reliability of the article. The labels are defined as follows:
    a) 1: Indicates the article is fake (potentially unreliable).
    b) 0: Indicates the article is true (reliable).

B. test.csv

This dataset is used to test the performance of the models trained on the train.csv data.

1) id: A unique identifier for each article, used to track submissions.
2) text: The text content of the article, identical in form to the text in the training set but without labels, as the purpose here is to predict these labels based on the trained model.

C. submission_sample.csv

This file is an example of how participants should format their predictions for the test dataset.

1) id: The unique identifier for each article in the test dataset. This matches the id field in test.csv.
2) label: The predicted label of the news article, where:
    a) 1: Indicates the article is fake (potentially unreliable).
    b) 0: Indicates the article is true (reliable).

D. train validation split

Since we don't have the ground truth answer of the test data, we extract 20% of data from the train.csv as our validation set.

- training set: 80% of train.csv
- validation set: 20% of train.csv

E. Evaluation Method

The evaluation of the models was carried out using accuracy, which is defined as follows:

$$\text{accuracy} = \frac{\text{correct predictions}}{\text{correct predictions} + \text{incorrect predictions}} \quad (1)$$

III. Related Work

A. Word2Vec

Word2Vec is a popular technique for natural language processing (NLP) that aims to represent words in a continuous vector space. Developed by a team at Google led by Tomas Mikolov, Word2Vec transforms words into dense vectors of real numbers in such a way that words sharing similar contexts in a large corpus have similar vector representations. This process is known as word embedding. Word2Vec models can capture semantic relationships between words, enabling various NLP applications such as language modeling, machine translation, and text classification.

There are two main architectures for implementing Word2Vec: Continuous Bag of Words (CBOW) and Skip-gram, and in this implementation, we use Skip-gram.

B. TF-IDF

TF-IDF, which stands for Term Frequency-Inverse Document Frequency, is a widely used statistical measure in natural language processing (NLP) and information retrieval. It evaluates how important a word is to a document within a collection or corpus. The core idea behind TF-IDF is to balance the frequency of a word in a specific document against its frequency across all documents in the corpus.

C. Bert Encoder

BERT (Bidirectional Encoder Representations from Transformers) is a groundbreaking technique for natural language processing (NLP) introduced by Google. Unlike traditional models that process text sequentially (left-to-right or right-to-left), BERT processes text bidirectionally, allowing it to capture context from both directions simultaneously. This bidirectional approach enables BERT to develop a deeper understanding of language nuances, making it highly effective for various NLP tasks.

BERT utilizes a transformer-based architecture, specifically designed to handle large-scale pre-training on a vast corpus of text. During pre-training, BERT learns to predict missing words in a sentence (masked language modeling) and to determine if two sentences are consecutive (next sentence prediction). These tasks help BERT grasp the intricacies of language, including syntax and semantics.

One of the key innovations of BERT is its ability to produce contextualized word embeddings. Unlike traditional word embeddings such as Word2Vec, which generate a single static vector for each word, BERT generates dynamic embeddings that consider the context in which a word appears. This allows BERT to distinguish between different meanings of a word based on its usage in a sentence.

BERT has been successfully applied to numerous NLP applications, including question answering, sentiment analysis, named entity recognition, and text classification. Its versatility and state-of-the-art performance have made it a cornerstone in modern NLP research and applications.

D. SVM

Several studies employ Support Vector Machine (SVM) for detecting fake news, each highlighting its effectiveness through quantifiable results. Jain et al. (2019) used SVM in a binary classification model to differentiate real from fake articles, achieving an accuracy of 93.50% in conjunction with Naive Bayes and NLP techniques. This model incorporated an aggregator, authenticator, and a suggestion system to refine its accuracy. Another study by Aphiwongsophon et al. (2018) reported the highest accuracy of 99.90% using a combination of SVM, Naive Bayes, and Neural Network, applied to a massive dataset of 948,373 Twitter messages, showcasing SVM's robust performance when used alongside other algorithms. Abdullah et al. (2019) achieved an accuracy of 89.34% with SVM, part of a suite of techniques including Naive Bayes, Logistic Regression, LSTM, and RNN, tested against a dataset from the 2010 Chile earthquake. Additionally, Reis et al. (2019) reported an 89% accuracy rate using SVM, which was part of a model that also included KNN and Naive Bayes, further confirming SVM's utility in diverse fake news detection scenarios. These findings illustrate the adaptability and efficacy of SVM in various configurations for the purpose of fake news detection, demonstrating significant success rates across different datasets and methodologies.

E. Logistic Regression

Tiwari et al. (2020) specifically identify Logistic Regression as the top-performing algorithm when paired with the term frequency-inverse document frequency (TF-IDF) feature extraction method, achieving an accuracy of 71%. This indicates that the success of Logistic Regression can vary significantly depending on the feature extraction method employed. Additionally, Logistic Regression is frequently mentioned alongside other algorithms in comparative studies aimed at detecting fake news, suggesting its common use in the field although specific accuracy figures for these broader applications are not detailed. This reflects the algorithm's recognized utility and effectiveness in classifying and analyzing fake news content across different research studies.

F. GCN

Fake news detection relies not only on the content of the text itself but also on the complex relationships between texts, such as similarities between different articles, citation relationships, and connections between authors and sources. GCN can effectively capture and utilize these graph-structured information, thereby improving detection accuracy. Utilizing Multiple Features:

In fake news detection, besides the text content, other features can be considered, such as diffusion patterns in social networks, user behavior, publication time of articles, and sources. These features can be constructed into a graph structure, and GCN can simultaneously utilize these multi-modal data for learning and inference. Enhancing Contextual Information:

Traditional text classification methods like RNN and CNN mainly focus on local information, whereas GCN can enhance each node's feature representation through information from neighboring nodes, thus capturing broader contextual information, which is crucial in fake news detection. Dynamic Updating and Adaptability:

GCN can dynamically update with the addition of new data, which is very useful for tasks like fake news detection that require continuous updating and adaptation to new patterns of real and fake news. Potential of Graph-Based Methods in Fake News Detection Research indicates that graph-based methods hold significant potential in the field of fake news detection.

According to "Fake News Detection through Graph-based Neural Networks: A Survey (2023)": https://arxiv.org/pdf/2307.12639

Many methods have been proposed to detect fake news, including many deep learning and graph-based approaches. In recent years, graph-based methods have yielded strong results, as they can closely model the social context and propagation process of online news. In this paper, we present a systematic review of fake news detection studies based on graph-based and deep learning-based techniques. We classify existing graph-based methods into knowledge-driven methods, propagation-based methods, and heterogeneous social context-based methods, depending on how a graph structure is constructed to model news-related information flows. We further discuss the challenges and open problems in graph-based fake news detection and identify future research directions.

IV. Model Pipelines

We proposed 5 pipelines to evaluate the model performance.

1) Content → TF-IDF → Vector → ML Classifier(SVM/Logistic Regression)
2) Content → Encoder(BERT) → Embedding → ML Classifier(SVM/Logistic Regression)
3) Content → Encoder(BERT) → Embedding → Encoder(Fintuned) + Classifier
4) Content → TF-IDF+PMI → Vector → Graph Construction → GCN Classifier
5) Content → Encoder(BERT) → Embedding → Graph Construction → GCN Classifier

V. Model Details

A. Word2Vec + ML (Logistic Regression / SVM)

In this study, we investigate the application of Word2Vec combined with two machine learning algorithms, Logistic Regression and Support Vector Machine (SVM), for a natural language processing task. Word2Vec is utilized to transform textual data into continuous vector representations, capturing semantic relationships between words. These word embeddings are then used as features for classification models.

Logistic Regression: The word embeddings generated by Word2Vec were used as input features for a Logistic Regression model. Logistic Regression is a straightforward yet effective classification algorithm that predicts the probability of a binary outcome. In this study, it achieved a classification accuracy of 73%.

Support Vector Machine (SVM): Similarly, we used the Word2Vec embeddings as features for an SVM model. SVM is a powerful classification algorithm that finds the optimal hyperplane to separate different classes in the feature space. This model achieved a classification accuracy of 72%.

B. Embedding(Bert) + ML (Logistic Regression)

In the development of a machine learning pipeline for detecting fake news on the Kaggle Fake News KDD 2020 dataset, the BERT encoder plays a pivotal role in the preprocessing phase. This pipeline begins by employing BERT (Bidirectional Encoder Representations from Transformers) to generate word embeddings for each token in the dataset's text entries. These embeddings serve as the input for a logistic regression model. This model achieves an accuracy of 0.69, offering a solid baseline while also allowing for straightforward interpretation and analysis of feature influence, which is valuable for understanding model decisions.

C. Embedding(Bert) + ML (SVM)

In a parallel approach, the same dataset and BERT-generated embeddings are utilized to train a Support Vector Machine (SVM) model for fake news detection. By creating a hyperplane that best divides the data points of the different classes, SVM effectively differentiates between 'fake' and 'real' news with a higher degree of accuracy, achieving a score of 0.73. This indicates a stronger performance in terms of precision and reliability compared to the logistic regression model.

D. Embedding(Bert) + MLP(huggingface)

In this study, we also explore the combination of Bidirectional Encoder Representations from Transformers (BERT) with a Multi-Layer Perceptron (MLP) for the task of fake news detection. BERT, a state-of-the-art pre-trained language model developed by Google, is utilized to generate contextual embeddings of textual data. These embeddings are then used as input features for a neural network model implemented using the Huggingface library.

BERT Embeddings: BERT is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. This means that BERT captures the context of a word based on the entire sentence, providing a richer and more nuanced understanding of text compared to traditional embeddings like Word2Vec. The BERT embeddings are leveraged to capture intricate semantic and syntactic information from the text, which are crucial for accurately identifying fake news.

Multi-Layer Perceptron (MLP): The contextual embeddings generated by BERT are fed into an MLP, a type of feedforward artificial neural network, for classification. The MLP consists of multiple layers of neurons, including an input layer, one or more hidden layers, and an output layer. Each neuron in the hidden layers applies a nonlinear activation function to its inputs, enabling the network to capture complex patterns and

relationships in the data. The final layer of the MLP outputs the probability of the text being classified as fake news.

Implementation with Huggingface: The Huggingface library provides a robust and user-friendly platform for implementing and fine-tuning BERT and other transformer-based models. By using Huggingface, we are able to efficiently integrate BERT embeddings with an MLP and leverage pre-trained weights for improved performance. The integration also facilitates experimentation with different hyperparameters and network architectures.

In our experiments, the BERT + MLP model achieved a classification accuracy of 81%, demonstrating the effectiveness of leveraging contextual embeddings and deep learning for fake news detection.

E. TF-IDF + TextGCN

In this method, we construct a heterogeneous graph from all the texts in the dataset, which includes two types of nodes: document nodes and vocabulary nodes. This heterogeneous graph has two types of edges: document-vocabulary edges and vocabulary-vocabulary edges.

The threshold for establishing document-vocabulary edges is based on TF-IDF values. TF-IDF assigns higher weighted scores to more important words/documents from a piece of text or a corpus. The importance of a word increases with its frequency within a text, but decreases with its frequency across different documents.

The threshold for establishing vocabulary-vocabulary edges is based on Pointwise Mutual Information (PMI), which measures the co-occurrence of vocabulary pairs. Due to the long length of the texts, we use a fixed-size sliding window to count the occurrences of vocabulary pairs within the same window.

After constructing the graph, we use the graph's adjacency matrix as input data for training the GCN model. The train_mask and test_mask are used to apply masking effects to the graph nodes during training.

In this method, we construct a heterogeneous graph from all the texts in the dataset, which includes two types of nodes: document nodes and vocabulary nodes. This heterogeneous graph has two types of edges: document-vocabulary edges and vocabulary-vocabulary edges.

The threshold for establishing document-vocabulary edges is based on TF-IDF values. TF-IDF assigns higher weighted scores to more important words/documents from a piece of text or a corpus. The importance of a word increases with its frequency within a text, but decreases with its frequency across different documents.

The threshold for establishing vocabulary-vocabulary edges is based on Pointwise Mutual Information (PMI), which measures the co-occurrence of vocabulary pairs. Due to the long length of the texts, we use a fixed-size sliding window to count the occurrences of vocabulary pairs within the same window.

After constructing the graph, we use the graph's adjacency matrix as input data for training the GCN model. The train_mask and test_mask are used to apply masking effects to the graph nodes during training.

F. Embedding + GCN (cos_sim or KNN)

First, we use BertTokenizer to tokenize the text, generating corresponding encoding ID vectors. These vectors are then fed into the BertModel to produce the corresponding embeddings. These embeddings are subsequently used to construct graphs using different graph construction strategies before being fed into the GCN for training.

The graph construction strategies include using cosine similarity as the threshold for edge creation and using KNN (k-nearest neighbors) as the threshold for edge creation.

In the first strategy, we calculate the cosine similarity between embeddings. Due to the large number of samples, using a standard cosine similarity algorithm would result in a similarity matrix too large for RAM. Therefore, we store these cosine similarity values as a sparse matrix and set a threshold of 0.5, resulting in a graph density of approximately 0.4.

In the second strategy, we calculate the distances between embeddings and select the top K nearest neighbor embeddings to create edges.

VI. Result

We compare the performance of models trained using different methods, using validation accuracy as the metric to evaluate model performance.

Model	Validation Accuracy
TF-IDF + LR	0.71
TF-IDF + SVM	0.72
Embeddings + LR	0.69
Embeddings + SVM	0.73
Embeddings + BERT + MLP	*0.81
TF-IDF + TextGCN	0.66
Embeddings + GCN (cos_sim)	0.59
Embeddings+GCN (KNN)	0.61

TABLE I
Model Comparison

Conclusion

- We compared the application and performance of different methods in the task of fake news detection using a purely content-based dataset. We also experimented with integrating NLP methods (such as BERT and TF-IDF), traditional machine learning algorithms, and Graph Neural Networks (GNNs) at various stages of upstream and downstream tasks.
- The method using embeddings and BERT performed the best. We speculate that this is because the dataset for this task is purely text-based, so using NLP-based methods consistently for both upstream and downstream tasks results in better performance.
- Traditional ML classifiers generally performed better than GCN. We speculate that this is because the dataset consists only of pure text and lacks more complex interaction relationships that the GCN model could leverage.
- The GCN method using TF-IDF and PMI performed better than the GCN method using word embeddings with cosine similarity or KNN. We speculate that this is

because TF-IDF and PMI better capture the relationships between texts and words, as well as between words, compared to simply using cosine similarity and KNN.

Future Work

- We can improve the GNN model. In this project, the GNN did not perform better than other models, possibly because our GNN model structure and design strategy were relatively simple, and did not fully utilize and exploit the high-level connectivity in the data.During the edge construction phase, we can compare the performance of different cosine similarity thresholds and adjust different K values in KNN.Moreover, we can try a learning-based approach for edge construction [7], or enhance the use of anomalous node information by individually learning and generating node graph filters for each node [8]. Additionally, we can adopt different GNN models, such as GAT.
- We can try incorporating social network information. Since this dataset only includes news text and labels, adding additional information such as the context of news publication and user interactions may provide more features to utilize. This could better leverage the GNN-based model's ability to capture complex interaction relationships, leading to improved model performance.

References

[1] Vosoughi S, Roy D, Aral S. The spread of true and false news online. Science. 2018;359(6380):1146–1151. doi: 10.1126/science.aap9559.

[2] Jain, Anjali, et al. "A smart System for Fake News Detection Using Machine Learning." 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT). Vol. 1. IEEE, 2019.

[3] Aphiwongsophon, Supanya, and Prabhas Chongstitvatana. "Detecting fake news with machine learning methods." 2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications & Information Technology (ECTI-CON). IEEE, 2018.

[4] Abdullah-All-Tanvir, Ehesas Mia Mahir, Saima Akhter, and Mohammad Rezwanul Huq "Detecting Fake News using Machine Learning and Deep Learning Algorithms" 2019 7th International Conference on Smart Computing and Communications (ICSCC)

[5] Reis, Julio CS, et al. "Supervised learning for fake news detection." IEEE Intelligent Systems 34.2 (2019): 76-81.

[6] Tiwari, V., Lennon, R.G. and Dowling, T. (2020), "Not everything you read is true! Fake news detection using machine learning algorithms", 2020 31st Irish Signals and Systems Conference, ISSC 2020, doi: 10.1109/ISSC49989.2020.9180206

[7] Yixin Liu, Yu Zheng, Daokun Zhang, Hongxu Chen, Hao Peng, Shirui Pan, Towards Unsupervised Deep Graph Structure Learning. Web Conf 2022

[8] Wei Zhuo, Zemin Liu, Bryan Hooi, Bingsheng He, Guang Tan, Rizal Fathony, Jia Chen, Partitioning Message Passing for Graph Fraud Detection. ICLR 2024