Fake News Detection Using Machine Learning and Deep Learning Models

Taha Sadikot
Naional Institute Of Technology
Kurukshetra
Haryana, India
19bmiit087@gmail.com

***Abstract*— this paper explores the implementation of fake news detection using both traditional machine learning algorithms and advanced deep learning architectures. The dataset consists of news articles labeled as real or fake. Our primary goal is to classify the news accurately based on text-based features. We apply different models, ranging from Logistic Regression, Random Forest, and Support Vector Machines (SVM) to more complex models like Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNN), and Bidirectional LSTMs. Through a comprehensive comparative analysis, we evaluate the performance of each model in terms of accuracy, precision, recall, and F1-score. The study highlights the advantages of deep learning models in capturing contextual and semantic information from news articles, while also showing the strength of traditional machine learning models in providing fast and interpretable results.**
Keywords—template, Scribbr, IEEE, format

1 Introduction

The rapid spread of misinformation and fake news poses a significant challenge to societies worldwide, influencing public opinion and decision-making. As social media and digital platforms have made it easier to disseminate fake news, it is crucial to develop automated systems that can efficiently identify and classify fake news. This study investigates the performance of various machine learning (ML) and deep learning (DL) models for fake news detection. The problem is treated as a binary classification task, where the objective is to classify news articles as either real or fake based on textual data.

The traditional machine learning models, including Logistic Regression, Random Forest, and SVM, are benchmarked against state-of-the-art deep learning architectures such as CNNs, LSTMs, and Bidirectional LSTMs. By comparing the performance of these models, this study aims to provide insights into the optimal approach for fake news detection.

2 Dataset and Preprocessing

2.1 Dataset Description

The dataset used in this study consists from between dates January 26, 2018 to February 23, 2028 of news articles, where each article is labeled as either fake or real. The dataset is split into training and testing sets, ensuring that the model can be trained and validated effectively. Data file consist of following shown in table below.

Column	Description
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

2.2 Preprocessing

Text pre-processing is a critical step in machine learning for natural language processing (NLP) tasks. The following steps were applied to the dataset:

- Handling Missing Data : Missing values in the author and title columns were replaced with empty strings (' ').
- Feature Engineering : The author and title columns were concatenated to form a new feature, content, which represents the combined text information from both fields.
- Text Cleaning : Regular expressions (re) were used to remove non-alphabetic characters from the text. All text was converted to lowercase.
- Stemming : The Porter Stemmer from the Natural Language Toolkit (NLTK) was used to reduce words to their base forms (e.g., "running" to "run"). Stemming helps normalize words with similar meanings.
- Stopword Removal : Common English words (stopwords) such as "the", "and", and "is" were removed to reduce noise in the dataset.

2.3 Text Vectorization:

To transform the text into numerical features, we applied TF-IDF Vectorization (Term Frequency-Inverse Document Frequency). This technique helps convert the text into feature vectors by assigning higher importance to words that are rare across the dataset but frequent in individual documents. It also reduces the impact of frequent but less informative words (such as “news” or “article”). This method was chosen over simpler approaches like CountVectorizer, as it takes into account word importance based on frequency.

3 MACHINE LEARNING MODELS

3.1 Model Selection

- Logistic Regression: A simple linear model that estimates the probability that a given input belongs to a particular class.
- Support Vector Machine (SVM): A powerful classifier that works by finding the hyper plane that best separates the classes in the feature space.
- Decision Tree: A tree-based model that splits the data into subsets based on feature values, making decisions based on those splits.
- K-Nearest Neighbors (KNN): A non-parametric classifier that predicts the label of a sample by looking at the k nearest data points in the training set.
- Naive Bayes (GaussianNB): A probabilistic model that assumes the independence of features.
- Random Forest: An ensemble learning method that combines multiple decision trees to improve prediction accuracy.
- Gradient Boosting: An ensemble method that builds trees sequentially to reduce errors from previous trees.
- AdaBoost: Another ensemble method, where each subsequent tree focuses on the misclassified samples of the previous trees.

In addition to these, XGBoost, LightGBM, and CatBoost were used for boosting-based models, which are known for their high performance in classification tasks.

3.2 Evaluation Metrics

The models were evaluated on the following metrics:

- Accuracy: The proportion of correctly classified instances.
- Precision: The proportion of true positives among all positive predictions.
- Recall: The proportion of true positives identified by the model out of all actual positives.
- F1-Score: The harmonic mean of precision and recall, balancing the two metrics.

The Accuracy, Precision, Recall and F1-score of different machine learning models are listed in below tables.

Model Name	**Accuracy**	**Precision**	**Recall**	**F1-Score**
Logistic Regression	0.9752	0.9605	0.9914	0.9757
Support Vector Machine	0.9856	0.9778	0.9938	0.9857
Decision Tree	0.9904	0.9904	0.9904	0.9904
K-Nearest Neighbors	0.5245	0.5129	1.0000	0.6781
Naive Bayes	0.8050	0.9051	0.6822	0.7780
Gradient Boosting	0.9637	0.9355	0.9962	0.9649
AdaBoost	0.9772	0.9675	0.9875	0.9774
Random Forest Classification	0.9906	0.9867	0.9947	0.9907
XGBClassifier	0.9873	0.9788	0.9962	0.9874
LGBMClassifier	0.9877	0.9824	0.9933	0.9878
CatBoostClassifier	0.9901	0.9848	0.9957	0.9902

4 DEEP LEARNING MODELS

4.1 Neural Networks

Deep learning models are particularly well-suited for NLP tasks due to their ability to capture complex patterns and dependencies within the data. For fake news detection, we experimented with several architectures:

4.1.1 Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network (RNN) that is particularly good at capturing sequential patterns in data. This makes it ideal for understanding the context in fake news detection.

4.1.2 Convolutional Neural Network (CNN): CNNs are commonly used for image data but have also proven effective for text classification by capturing local word patterns (n-grams) in text.

4.1.3 Bidirectional LSTM (Bi-LSTM): A variant of LSTM that processes the input sequence from both directions (forward and backward), allowing the model to understand both past and future context.

4.2 Model Configuration

For each model, we used the following layers and parameters

- Embedding Layer: Converts the input text into dense vector representations (word embeddings).
- LSTM Layer: The core recurrent layer for sequence modelling in LSTM and Bi-LSTM models.
- Conv1D Layer: Used in CNN models to capture local n-gram features from text.
- Dense Layer: The final fully connected layer with a sigmoid activation function to produce the output (real or fake).

4.3 Results of Deep Learning Models

Model Name	Accuracy	Precision	Recall	F1-Score
LSTM	0.9865	0.9856	0.9875	0.9866
CNN	0.9894	0.9876	0.9914	0.9895
Bidirectional LSTM	0.9829	0.9893	0.9765	0.9828

Among the deep learning models, the Bidirectional LSTM performed the best, achieving the highest accuracy.

5 COMPARATIVE ANALYSIS:

Model Type	Model Name	Accuracy	Precision	Recall	F1-Score
Traditional Ml	Random Forest Classification	0.9906	0.9867	0.9947	0.9907
Deep Learning	CNN	0.9894	0.9876	0.9914	0.9895

- Random Forest emerged as the best-performing traditional machine learning model with an accuracy of 99.6%. The model's ability to combine multiple decision trees helped improve its predictive performance by reducing overfitting and enhancing generalization on unseen data. Its ensemble-based approach made it highly reliable for binary classification, such as fake news detection. IN deep learning, the

- Convolutional Neural Network (CNN) outperformed other neural network architectures with an accuracy of 98.94%. CNN's ability to capture local features (like word n-grams) from text through convolutional layers makes it well-suited for text classification tasks. While slightly outperforming Random Forest, CNN's ability to model spatial dependencies in the text allows it to detect nuanced patterns that traditional ML methods may miss.

6 CONCLUSION

In conclusion, while traditional machine learning models like Random Forest offer strong, interpretable results with lower computational cost, deep learning models like CNN are better suited for capturing deeper semantic patterns in text, offering superior flexibility and adaptability for more complex datasets. Depending on the application needs—such as speed, interpretability, or accuracy—either approach can be highly effective for fake news detection.

7 FUTURE WORK

- Transformer-based Models: Future work could include experiments with transformers like BERT and GPT, which have demonstrated state-of-the-art results in NLP tasks.

- Explainability: Enhancing model interpretability, especially in deep learning models, using techniques such as SHAP or LIME to explain predictions.

8 REFERENCES

[1] Random Forest: Scikit-learn Random Forest

[2] Logistic Regression: Scikit-learn Logistic Regression

[3] Support Vector Machines (SVM): Scikit-learn SVM

[4] Gradient Boosting: Scikit-learn Gradient Boosting

[5] Porter Stemming: NLTK Stemming

[6] TF-IDF Vectorizer: Scikit-learn TF-IDF

[7] CNN for Text Classification: Towards Data Science - CNN

[8] LSTM Networks: Colah's Blog - LSTM

[9] Fake News Detection: Medium - Fake News Detection

[10] Keras Documentation: Keras

[11] TensorFlow Documentation: TensorFlow