International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

LEVERAGING EXPLAINABLE AI TO ENHANCE DEEP LEARNING IN FAKE NEWS DETECTION CAPTURING COMPLEX LINGUISTIC PATTERNS AND SEMANTIC INSIGHTS

Pundlik Dattatray Jadhav[1], Dr. Rajesh k Shukla[2]

[1]Research Scholar, Department of Computer Science and Engineering, Oriental University, Indore, Madhya Pradesh, India. pdjadhav17@gmail.com

[2] Supervisor, Oriental University, Department Computer Science and Engineering, Indore, Madhya Pradesh, India. rajeshshukla@oriental.ac.in

Abstract:

This study looks at how Explainable Artificial Intelligence (XAI) methods can be combined with Deep Learning (DL) models, namely Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), to make it easier to spot fake news. Fake news is a big problem for society, and regular ways of finding it are not always good at picking up on the complex language patterns and semantic details that are common in false information. Even though DL models, especially CNN and RNN, are very good at classifying text, it is hard to figure out how they make decisions because they are "black boxes." This could make them less reliable in high-stakes situations like finding fake news. To get around this problem, we investigate how XAI methods can make DL models more reliable and useful by making them clear and easy to understand. CNN and RNN models are used in the study to find fake news by looking at the text's complicated language structures and meaning trends. We use XAI techniques like SHAP (SHapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations) to show how these models understand the data they are given, showing important traits and trends that affect the discovery process. Our results show that combining XAI and DL makes the model better at finding fake news by bringing out language clues, rhetorical structures, and semantic errors that are often signs of fake content. The suggested method not only makes detection more accurate, but it also helps us understand the underlying language and semantic patterns better. This makes fake news detection systems more open and reliable. This study tells us a lot about how XAI can make DL models easier to understand. This makes it possible for more useful and trustworthy tools to be used in the fight against fake news.

Keywords: Explainable AI (XAI), Deep Learning (DL), Fake News Detection, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Linguistic Patterns Analysis

I. Introduction

In this digital age, fake news spreads very quickly and has become a major problem for society. It affects public opinion, government security, and trust in society. As social media platforms make it easier for more people to share information, it becomes more difficult and important to tell the difference between real and fake information. Because of this, there is a greater need for improved technologies that can correctly find false information and stop it

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

from spreading. Deep Learning (DL) methods, especially Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), have shown a lot of promise in automating the discovery of fake news by finding complex patterns in text data. But the fact that these DL models are unclear and "black boxes" makes things very hard because it's hard to figure out how they make decisions, which makes people worry about their dependability and trustworthiness [1]. Deep understanding models, such as CNNs and RNNs, are very good at understanding complicated language and semantic patterns. This makes them very good at finding fake news. CNNs [2] are very good at finding local patterns and hierarchical features in text, while RNNs can get information about the context and the order of events, which helps them spot minor signs of misleading content. Even though these models work very well, they are not always clear, which makes it hard for stakeholders to understand why they make the statements they do. In high-stakes tasks like finding fake news, where wrong classifications can have very bad results, it is important to have models that can be understood and give reasons for their choices.

Explainable Artificial Intelligence (XAI) methods have gotten a lot of attention to make DL models more open and reliable to solve the problem of interpretability. The goal of XAI methods is to connect the forecasting power of deep learning algorithms with the need for reasons that people can understand. XAI can help find the language patterns, meaning structures, and environmental cues that guide the classification of fake news by showing how DL models make decisions. Making this happen not only makes the recognition process more reliable, but it also helps people trust AI-driven systems more, which makes them more useful in real life. This study suggests a complete system that combines XAI methods with CNN and RNN models to make it easier to spot fake news. The goal of the merger is to catch the complicated language patterns and meaning insights that are typical of fake news while also making the decision-making process clear. We want to understand how CNN and RNN models work by using XAI techniques like SHAP (SHapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations). This will help us find the most important features and patterns that help classify fake news. This method helps us learn more about how certain words, phrases, and sentence patterns affect the model's results. This gives writers, experts, and fact-checkers useful information they can use [3].

The work is important because it could make models that find fake news more accurate and easier to understand. Previous study has mostly worked on making recognition better. This paper, on the other hand, stresses how important it is for AI systems to be able to be explained. Our method combines the best features of CNN and RNN models with XAI techniques to make a strong and clear system that can spot fake news very accurately and give clear reasons for its predictions [4]. The main problem this paper tries to solve is finding fake news. It does this by using DL models and XAI methods to make a solution that works and can be understood. The suggested approach not only improves the best way to find fake news, but it also adds to the growing body of study on explainable AI by showing how interpretability can make deep learning models more reliable and useful in the real world [5]. The point of this study is to learn more about how language complexity, semantic knowledge,

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

and explainability work together. This will help make AI-based solutions for fighting lies more reliable and effective.

II. Literature Review

In recent years, finding fake news has become an important area of study. This is because false information is spreading quickly and can hurt people. The speed with which fake news spreads on social media and other online platforms has made it even more important to have automatic ways to tell the difference between real and fake news. To solve this problem, researchers have investigated different machine learning and deep learning methods, focusing on understanding the complicated language patterns and hidden meanings in fake news. Support Vector Machines (SVM), Naïve Bayes, and Decision Trees were some of the first machine learning models that were used to find fake news. To find fake news, these models often used traits that were built by hand, like word frequency, mood analysis, and artistic trends. Some of these methods worked, but they were not perfect because they could not pick up on the complex and detailed language patterns that make up misleading material [6]. For example, studies showed that features that were made by hand didn't always work across different datasets. This showed the need for more advanced methods that can easily learn complex patterns.

Deep learning [7] has made a big difference in the field of finding fake news by letting models learn from raw text data without having to do feature engineering by hand. These days, Convolutional Neural Networks (CNNs) are often used for text classification jobs because they can find local word patterns and hierarchical features within lines. Researchers have shown that CNN-based models can very accurately spot fake news by finding small language clues and patterns that are often signs of lying. CNNs are very good at recognizing grammar structures, which helps them spot the false or sensationalist language that is often used in fake news stories. Recurrent Neural Networks (RNNs) and their variations, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), have also become popular in study that tries to spot fake news. CNNs focus on recording local features, but RNNs are good at modeling sequence relationships and surrounding information. This means they can understand the bigger meaning behind news stories. RNNs can find errors and gaps that may point to dishonesty because they can model how information flows between words [8].

Even though CNNs and RNNs are very popular, they are often attacked for being "black boxes" because it is hard to understand how they make decisions. This lack of openness is a big problem, especially in areas like finding fake news where faith and being able to explain things are very important. Explainable Artificial Intelligence (XAI) has come up as a potential way to solve this problem. It offers ways to make the way deep learning models make decisions easier to understand. Researchers have used methods like SHAP (SHapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations) to figure out the traits and trends that affect model forecasts [9]. This helps us understand how CNNs and RNNs tell the difference between real and fake news. Recent studies have started to combine XAI methods with deep learning models to make it easier to understand when

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

fake news is being spread. For instance, study used SHAP to find the most important words and phrases for classifying fake news. This gave researchers a better idea of the language and semantic patterns that affect model choices. Similarly, other studies used LIME to come up with reasons for RNN predictions. These studies [10] showed how certain story patterns and artistic elements can help find fake news. The results of these studies show that mixing deep learning with XAI could lead to more open and reliable methods for finding fake news.

Recently published study has also shown how important it is to use meaningful knowledge to spot fake news. Beyond just looking at patterns at the word level, semantic analysis tries to figure out what words, phrases, and sentences mean and how they relate to each other. Language models that have already been taught, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT-3, are very good at catching meaning details and surrounding information. This makes them useful tools for finding fake news. Researchers found that BERT-based models could do better than standard CNN and RNN methods [11] by using contextual embeddings to understand what text really means, which made it easier to spot fake news. Although BERT and other transformer-based models have come a long way in terms of accuracy, they are still hard to understand. Researchers are still looking into how to combine these advanced models with XAI techniques. For example, attention weights and gradient-based methods can help us understand how transformer models make decisions. There is a clear progression from basic machine learning methods to advanced deep learning methods that use CNNs, RNNs, and transformer models in the literature on finding fake news. Deep learning has made it easier to record complicated language patterns and meaningful information, but it is still hard to figure out what it all means. Adding Explainable AI methods is a potential way to make fake news detection models more open and reliable. This will help us learn more about the language and semantic clues that tell the difference between real and fake content. This mix of deep learning and XAI is a big step toward making tools that fight the spread of false information more reliable and effective in the digital age.

Table 1: Summary of related work

Approach	Methods	Finding	Focused Parameters	Limitation	Advantage
Traditional Machine Learning [12]	SVM, Naive Bayes, Decision Trees	Moderate accuracy, struggles with complex patterns	Word Frequency, Stylistic Patterns	Limited generalization across diverse datasets	Easy implementation, interpretable features
CNN-based Deep Learning [13]	Convolutional Neural Network (CNN)	High accuracy, effective in capturing local features	Local Word Patterns, Syntactic Features	Limited interpretability, black-box nature	High accuracy in pattern recognition
RNN-based	Recurrent Neural	Good at capturing sequential	Contextual Information,	Difficulty in capturing non-	Captures sequential

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

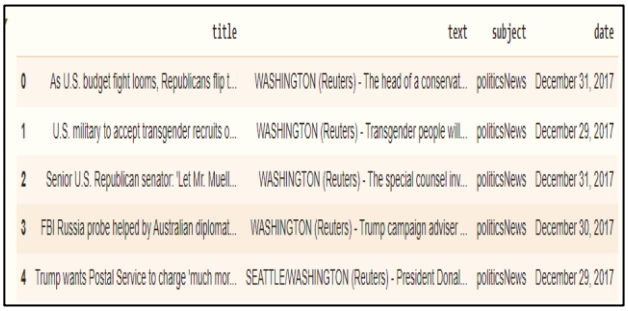
Deep Learning [14]	Network (RNN)	dependencies	Sequential Dependencies	sequential features	information effectively
Hybrid CNN-RNN Model [15]	Combination of CNN and RNN	Enhanced performance in combining local and sequential features	Combination of Local and Sequential Patterns	Complexity in model training and implementation	Combines advantages of CNN and RNN
Transformer Models (BERT) [16]	Transformer-based BERT Model	Superior performance in capturing semantic nuances	Semantic Context, Phrase-Level Analysis	High computational resources required	Handles complex semantic relationships
LSTM with Attention Mechanism [17]	LSTM with Attention Layer	Improved detection through focused attention on key text segments	Attention on Key Phrases and Words	Prone to overfitting with small datasets	Enhanced attention to important text segments
GAN for Fake News Detection [18]	Generative Adversarial Networks (GAN)	Captures deceptive patterns via generative approach	Pattern Generation and Detection	Challenges in training and generating accurate samples	Innovative approach to identifying fake news
Ensemble Learning Models	Random Forest, AdaBoost, XGBoost	Combines strengths of multiple models for higher accuracy	Combination of Model Strengths	Risk of overfitting and complexity	Improved accuracy through model integration
Hierarchical Attention Networks [19]	Hierarchical Attention Networks	Effectively identifies hierarchical structures in text	Hierarchical Text Structures	Computationally intensive, requires large datasets	Effectively captures multi-level text structures
Bi-Directional LSTM [20]	Bi-Directional LSTM (BiLSTM)	Better understanding of contextual information	Contextual and Sequential Dependencies	Difficulty in training, requires extensive data	Better representation of context and semantics
Graph Neural Networks [21]	Graph Convolutional Networks (GCN)	Leverages graph-based relationships in text	Graph-Based Relationships and Dependencies	High computational demand, complex structure	Captures relational information in data
Contextual	Word2Vec, GloVeEmb	Captures deep semantic	Word Meaning,	Limited interpretability	Deep semantic

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Embedding Models	eddings	meanings in word usage	Semantic Context	, requires pre-trained models	understanding of text
Explainable AI Integration	SHAP, LIME with Deep Learning Models	Provides transparency and interpretability in decisions	Interpretability, Feature Importance	Interpretability techniques may not capture all nuances	Adds explainability to deep learning models

III. Dataset Description

The Kaggle Fake News Detection dataset [22] is a complete set of data that can be used to build and test machine learning models that can spot fake news. This dataset has many news stories that have been marked as either fake or real. This makes it a good choice for training and testing classification algorithms. A news story's title and text are two of its most important parts. The title gives a short overview of the story, and the text is the whole story. It is very important to look for these things in order to find the language patterns and meaning frameworks that separate fake news from real news. Another important feature is the theme, which sorts articles into groups like politics, science, and entertainment. This gives recognition models more context, which can make them more accurate. The release date is also given, which lets experts look at how fake news spreads over time. Finally, the label property tells you whether the story is real or fake. This is usually done with binary values, with 0 meaning "fake" and 1 meaning "real."Because this dataset is so diverse and large, it's a great way to test out different ways to find fake news, such as deep learning methods like CNNs, RNNs, and transformer models. It gives researchers a lot of chances to look into how different parts of text and language clues can be used to make systems that can spot fake news accurately and reliably. In the figure 1, Fake News Detection dataset that is shown: title, text, subject, and date. Each row is a different news story, and the "title" column shows the article's headline, which is often very important for knowing what the story is about. The main content for studying and spotting fake news is the "text" column, which has a short or full body of the story. The "subject" column, which looks like "politicsNews," sorts the pieces by topic or field, and the "date" column shows when they were published. By looking at word patterns, the bigger picture, and current events, this dataset format can help machine learning models find fake news.



	title	text	subject	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017

Figure 1. Dataset Description

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The bar chart illustrates in figure 2 the distribution of fake and real news articles in the dataset. There are slightly more fake news articles compared to real ones, indicating a balanced dataset. This balance is essential for training machine learning models effectively, as it ensures that the model learns to distinguish between both classes accurately.

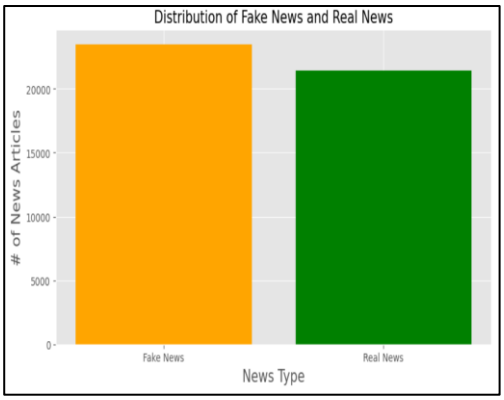


Figure 2. News Distribution

IV. Methodology

A. Data Pre-processing:

1. Check Missing Values:

The first step in pre-processing data is to look for datasets that are missing values. Values that are missing can be caused by poor data entry, communication mistakes, or other things. It is very important to find and deal with these missing numbers because they can change the research results and make machine learning models work less well. Data security can be maintained by doing things like getting rid of rows with missing values or filling them in with the right values (mean, median, or mode).

Algorithm:

Let D be the dataset with n rows and m columns.

Define M(i,j) as an indicator function where:

$$M(i,j) = \begin{cases} 1 \; if \; the \; value \; at \; position \; (i,j) \; is \; missing \\ 0 \; otherwise \end{cases}$$

Calculate the total number of missing values T in the dataset:

$$T = \Sigma \, (i = 1 \; to \; n) \, \Sigma \, (j = 1 \; to \; m) \, M(i,j)$$

Calculate the missing value ratio R for each column j:

$$R_j = \frac{\sum_{i=1}^{n} M(i,j)}{n}$$

2. Remove Noisy Words from Text:

These are words that do not belong there and don't add to your understanding of the text. URLs, HTML tags, numbers, letter marks, and special symbols are all examples. These parts can make things more complicated than they need to be, which makes it harder for the model to learn important things from the data. Getting rid of words that don't add anything to the sense of the text helps make it more focused and meaningful, which makes machine learning models more accurate.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Algorithm:
1. Let S be the set of all words in the text, and N be the set of noisy words (e.g., punctuation, numbers, special characters).
2. Define a filtered word set S':
S' = S - N
3. For each word w in S, check:
w in N -> remove w
3. Remove Stopwords:
Get rid of stopwords, which are common words like "the," "is," "in," and "and" that show up a lot in text but don't add much to the analysis. You don't need to know these words to understand the context or sense of the text, and they can slow down computers. Getting rid of stopwords makes the dataset shorter, which lets the model focus on words that are more useful for finding fake news.
Algorithm:
1. Let S be the set of all words in the text, and W be the predefined stopwords list.
2. Define a clean word set S' as:

$$S' = S - W$$

3. For each word w in S:

$$w\ in\ W -> \ remove\ w$$

4. Stemming and Lemmatization:
Stuttering and lemmatization are ways to break down words into their root forms. Lemmatization takes words back to their base or dictionary form, stemming gets rid of word endings to get to the base form. "Running" changes to "run," and "better" changes to "good." These methods help to normalize text data, lower its dimensionality, and make sure that words that are similar are treated the same, which makes the model work better.
1. Given a word w in the text, define stem(w) as the stemmed version and lemma(w) as the lemmatized version.
2. Apply stemming:
w -> stem(w)
3. Apply lemmatization:
w -> lemma(w)
5. Generating Word Count:
To make a word count, you must figure out how often each word appears in a text or across the dataset. This step helps you understand the most popular words and find important trends in the text. Word count is an important part of machine learning models because it helps them figure out how important and common certain words are, which can help them tell the difference between fake and real news, as shown in figure 3. This step helps organize the data more clearly so that it can be analysed more effectively.
Algorithm:
1. Let D = {d1, d2, ...,dn} be the set of documents, and W = {w1, w2, ..., wm} be the vocabulary of unique words.
2. Define the frequency f(w_i, d_j) as the count of word w_i in document d_j:

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$f(w_i, d_j) = \Sigma\,(k = 1\ to\ |d_j|)1(w_k = w_i)$$

where $1(w\_k = w\_i)$ is an indicator function that equals 1 if w_k = w_i and 0 otherwise.

3. Create the word count vector F_j for document d_j:

$$F_j = [f(w_1, d_j), f(w_2, d_j), \ldots, f(w_m, d_j)]$$

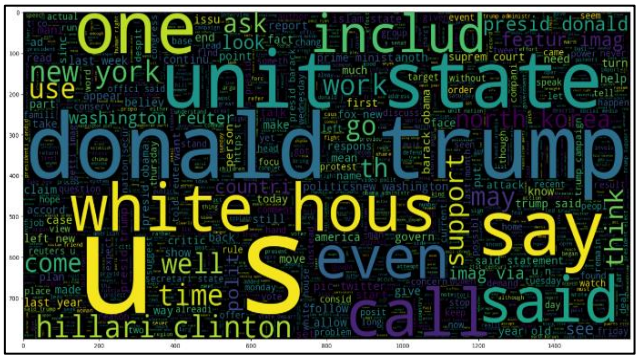


Figure 3. Word Count

6. Punctuation Removal

Getting rid of punctuation marks, such as commas, periods, exclamation points, and question marks, is an important part of preparing text. These symbols do not add to the sense of the text, but they can make natural language processing tasks noisier, which can make models work better and lower the number of dimensions.

Step-wise Mathematical Model:

1. Define Text Set:

Let T = {w1, w2, ...,wn} be the set of all words in the text, where each wi may include punctuation.

2. Define Punctuation Set:

Let P be the set of punctuation marks: P = {.,!,?,;,:,',",-,—,()}.

3. Filter Words:

For each word wi in T, check:

$$w'\_i = wi - P$$

where w'_i represents the word without punctuation.

4. Create Clean Text:

The resulting clean text T' is:

$$T' = \{w'\_1, w'\_2, \ldots, w'\_n\}$$

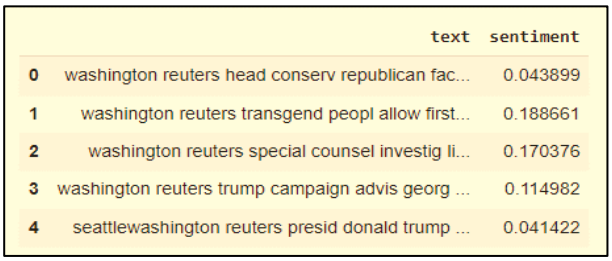
7. Sentiment Analysis (FEATURE)

The data shown is the outcome of analyzing how people felt about a group of news stories. Each row has a cleaned piece of text and an emotion score that goes with it. The "text" column shows the articles' processed text, which is free of capitalization, stopwords, and

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

other unnecessary words so it can be analyzed. The numbers in the "sentiment" column range from 0 to 1, which show how the text makes you feel. Values that are closer to 0 show negative sentiment, while values that are closer to 1 show good sentiment. This research helps us understand the emotional tone of the stories, which is a key part of spotting fake news and figuring out the general tone or bias of the news.



	text	sentiment
0	washington reuters head conserv republican fac...	0.043899
1	washington reuters transgend peopl allow first...	0.188661
2	washington reuters special counsel investig li...	0.170376
3	washington reuters trump campaign advis georg ...	0.114982
4	seattlewashington reuters presid donald trump ...	0.041422

Figure 4. Sentiment Analysis

B. Feature Extraction

A very important step in getting text data ready for machine learning models is feature extraction. To do this, written information must be turned into number representations that capture the meaning and structure of the text. This makes it possible for machine learning systems to read the words correctly. Text-2 Vector Conversion and Word2Vec are two feature extraction methods that are used a lot.

1. Text-2 Vector Conversion

Text-2 Vector Conversion is a way to turn text data into number vectors so that machine learning models can easily handle text data. One popular method is the Bag-of-Words (BoW) model, which shows each document as a list of words that are used a lot. The Term Frequency-Inverse Document Frequency (TF-IDF) method is another famous one. It finds out how important a word is in a document compared to a group of documents.

"Bag-of-Words" (BoW): This method makes a list of all the unique words in the dataset. Once that is done, each text is shown as a vector, with each piece showing how often a word from the dictionary appears. For instance, if the word "fake" shows up three times in a text, its vector number will be three.TF-IDF: This method gives words weights based on how often they appear in a document and how rarely they appear in all documents. This is how to figure out TF-IDF:

$$TF - IDF(t,d) = TF(t,d) \times \log(DF(t)N)$$

Where t is the term, d is the document, N is the total number of documents, and DF(t) is the number of documents containing the term t. This method captures the importance of words, making it more informative than simple word counts.

2. Word2VEC

Word2Vec is a sophisticated feature extraction method that turns words into dense, continuous vector representations that show how they relate to each other semantically. Word2Vec, unlike BoW or TF-IDF, makes word embeddings that show words in a three-

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

dimensional space, with similar words being closer together. Neural networks are used in this method to learn how to describe words based on the context in which they appear.

There are two main architectures used in Word2Vec:

Continuous Bag of Words (CBOW): This method has the model guess a target word by looking at the words that are around it. CBOW learns to guess the main word from a group of words that make up the background. This helps figure out what words mean by looking at the words around them.

This model, called skip-gram, works the other way around from CBOW. It looks at the main word and guesses the things that go with it. Some word connections are hard to find, but skip-gram is great at finding them, and it is often used when datasets are not very big.

Word2Vec embeddings give a full picture of words, including subtle differences in grammar and meaning. This makes them very useful for jobs like finding fake news, figuring out how people feel about something, and sorting text into groups. This set of word vectors can be fed into deep learning models to help them learn more complex links in written data.

C. DL Model

1. CNN

Convolutional Neural Networks (CNNs) are strong deep learning models that are commonly used to pull out features in text-based tasks like finding fake news. CNNs are usually used to process images, but they are also great at handling natural language (NLP) tasks because they can find local patterns and n-grams (word sequences) in text. When CNNs try to find fake news, they use convolutional filters on word embeddings (like Word2Vec) to find important patterns and traits that separate fake news from real news. The convolution layers pull out hierarchical features, and the pooling layers lower the number of dimensions so that they can focus on the most important data. CNNs are good at finding important language cues, context, and meaning structures. This lets them correctly label text as either fake news or real news.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 98, 128)	512
max_pooling1d (MaxPooling1D)	(None, 49, 128)	0
conv1d_1 (Conv1D)	(None, 47, 64)	24,640
max_pooling1d_1 (MaxPooling1D)	(None, 23, 64)	0
flatten (Flatten)	(None, 1472)	0
dense_5 (Dense)	(None, 128)	188,544
dropout (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 64)	8,256
dense_7 (Dense)	(None, 1)	65

Total params: 222,017 (867.25 KB)
Trainable params: 222,017 (867.25 KB)
Non-trainable params: 0 (0.00 B)

Figure 5: Representation of CNN sequence on Dataset

The figure 5shows how a sequence CNN model for finding fake news is put together. It is made up of several layers, including two 1D convolutional layers (conv1d), max-pooling layers, a flatten layer, and fully linked (dense) layers. The model has a dropout layer for

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

regularization that keeps it from fitting too well. It's possible to train 222,017 factors, and the final form of each layer is very specific. This parameter count shows how complicated the model is, matching its ability to learn and its ability to run quickly. The last layer of output gives a single number that can be used for binary classification to show whether the news is real or fake.

Step wise Process:

1. Word Embedding Layer

The input text is converted into a word embedding matrix using a pre-trained model like Word2Vec. Let the input sequence of words be x = [x1, x2, ...,xn], where n is the number of words in the text. Each word xi is represented as a vector of dimension d.

The word embedding matrix E for the entire input sequence is:

E = [[e11, e12, ..., e1d][e21, e22, ..., e2d]...[en1, en2, ..., end]]

where E is of dimension n x d.

2. Convolution Operation

A convolution operation is performed using a filter W of size h x d. This filter slides over the embedding matrix E, capturing local features. For a given position i, the convolution operation produces:

$$ci = f(\Sigma\ (j = 1\ to\ h)\ \Sigma\ (k = 1\ to\ d)\ Wjk * e(i + j - 1, k) + b)$$

where W is the filter matrix, b is the bias term, and f is an activation function (e.g., ReLU: f(x) = max(0, x)). The result is a feature map C = [c1, c2, ..., c(n-h+1)].

3. Multiple Filters Application

Multiple filters of different sizes h are applied to capture various n-grams and patterns in the text. For m filters, the feature maps for each filter are represented as:

F = [C1C2...Cm]

where F is of dimension m x (n-h+1).

4. Max Pooling Layer

A max-pooling operation is applied to reduce the dimensionality of each feature map and retain the most important features. For each feature map Ci, the pooled value pi is:

$$pi = max(Ci) = max([ci1, ci2, \ldots, ci(n - h + 1)])$$

The pooled feature vector P becomes:

$$P = [p1, p2, \ldots, pm]$$

where P is of dimension m.

5. Flattening Layer

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The pooled feature vector P is flattened into a single column vector to form the input to the fully connected layer:

F_flattened = [p1p2...pm]

Where F_flattened is of dimension m x 1.

6. Fully Connected Layer

The flattened vector is passed through a fully connected layer to learn complex combinations of features. The output y from the fully connected layer is calculated as:

$$y = f(Wf * F\_flattened + bf)$$

Where Wf is the weight matrix, bf is the bias vector, and f is the activation function (e.g., sigmoid or softmax for binary/multi-class classification).

7. Output Layer and Loss Function

The final output from the fully connected layer is passed to an output layer that predicts the class label (fake or real). For binary classification, the output probability y_hat is computed using the sigmoid function:

$$y_{hat} = \frac{1}{1 + \exp(-y)}$$

The loss function used for training is the binary cross-entropy loss:

$$L = -\left(\frac{1}{N}\right)\sum_{i=1}^{N}\left[yi * \log\left(y_{hat_i}\right) + (1 - yi) * \log\left(1 - y_{hat_i}\right)\right]$$

where N is the number of training samples, yi is the true label, and y_hat_i is the predicted probability.

2. RNN

Recurrent Neural Networks (RNNs) are great at finding fake news because they are very good at processing linear data and gathering environmental information. RNNs are different from standard models because they can handle word sequences in articles while keeping the order of events and knowing how words relate to each other. Because of this, RNNs are very good at finding the patterns, subtleties, and contexts in language that separate fake news from real news. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are two variations that make it easier for RNNs to keep track of long-term relationships, which solves the problem of disappearing slopes. RNNs are very good at accurately identifying fake news because they learn from patterns, which lets them follow the flow of information, spot errors, and spot misleading language.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Model: "sequential_1"

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 256, 128)	1,280,000
bidirectional_2 (Bidirectional)	(None, 256, 128)	98,816
bidirectional_3 (Bidirectional)	(None, 32)	18,560
dense_2 (Dense)	(None, 64)	2,112
dropout_1 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 1)	65

Total params: 1,399,553 (5.34 MB)
Trainable params: 1,399,553 (5.34 MB)
Non-trainable params: 0 (0.00 B)

Figure 6: Representation of RNN Architecture

Figure 6 shows the structure of an RNN model for finding fake news, with a focus on layers that can go both ways. The model starts with an embedding layer with 1,280,000 parameters that turns words into vectors. The next part has two bidirectional layers (Bidirectional LSTM/GRU) that handle text in both forward and backward ways. This lets the model understand how sets of events are related to each other in context. There are 117,376 trainable factors in these levels as a whole. After a dropout layer to stop overfitting, a fully linked dense layer with 64 units is used to learn more features. With a total of 1,399,553 trainable parameters, the last thick layer gives out a single number for binary classification.

3. CNN + RNN

The CNN+RNN combined method takes the best parts of both Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) and uses them together to find fake news more effectively. This combination makes it easier for the model to understand both short-term word-level traits and long-term environmental relationships. This makes it better at looking at complex textual data. Using convolutional filters, the CNN part pulls out important features like n-grams, word patterns, and grammar structures. The RNN (usually LSTM or GRU) part handles the text's sequence connections and context. In real life, the text that is being entered is first turned into word embeddings and then sent through the CNN layers, which look for important local patterns. The RNN layers then take these traits and use them to understand how time and context move. Together, these elements help the model better spot minor language cues, errors, and lies in stories. The mixed CNN+RNN method is very good at finding fake news because it uses CNNs' ability to find important word patterns and RNNs' ability to understand how information is put together. This leads to better classification and a better understanding of fake news material.

CNN + RNN: Hybrid Approach for Fake News Detection - Mathematical Model

Step 1: Word Embedding Layer

The input text is converted into a word embedding matrix using a pre-trained model like Word2Vec or GloVe. Let the input sequence of words be x = [x_1, x_2, ...,x_n], where n is the number of words, and each word x_i is represented as a vector of dimension d.

The embedding matrix E for the sequence is:

$$E = [e_{11}\, e_{12} \ \dots \ e_{1d}][\, e_{21}\, e_{22} \ \dots \ e_{2d}]$$

$$\dots [e\_n1 \ \ e\_n2 \ \dots \ e\_nd]$$

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$where\ E\ \in\ \mathbb{R}\hat{}(n\ \times\ d).$$

Step 2: Convolution Operation (CNN)

A filter W of size h × d slides over the embedding matrix E to perform the convolution operation. For a given position i, the convolution output c_i is computed as:

$$c_i = f\left(\Sigma_{j=1}^{h}\Sigma_{k=1}^{d}W_{jk} * e_{i+j-1,k} + b\right)$$

where f is an activation function (e.g., ReLU), and b is a bias term. This operation generates a feature map C = [c_1, c_2, ..., c_(n-h+1)].

Step 3: Max Pooling Layer (CNN)

A max-pooling operation is applied to the feature map C to reduce its dimensionality and retain the most important features:

$$p = max(C) = max([c\_1, c\_2, \ldots, c\_(n-h+1)])$$

The output is a pooled feature vector P that summarizes the important features extracted by the CNN.

Step 4: Sequence Formation for RNN

The pooled feature vectors P from multiple filters are concatenated to form a sequence input S for the RNN:

$$S = [p\_1, p\_2, \ldots, p\_m]$$

where m represents the number of filters used in the CNN layer.

Step 5: Recurrent Operation (RNN)

The sequence S is fed into an RNN (e.g., LSTM or GRU) layer, which processes the sequence step-by-step to capture temporal dependencies. For each time step t, the hidden state h_t is updated as:

$$h_t = g(W_h * S_t + U_h * h_{t-1} + b_h)$$

where g is the activation function, W_h, U_h are weight matrices, and b_h is the bias term.

Step 6: Output and Classification Layer

The final hidden state h_T (where T is the last time step) is passed to a fully connected layer for classification. The output y is computed using:

$$y = \sigma(W_y * h_T + b_y)$$

Where W_y is the weight matrix, b_y is the bias term, and σ is the sigmoid (for binary classification) or softmax (for multi-class classification) activation function.

4. LIME

Local Interpretable Model-agnostic Explanations, or LIME, are a good way to make fake news spotting models more open. Because CNNs and RNNs are "black boxes," LIME makes

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

them easier to understand by coming up with local reasons for each guess. To do this, it changes versions of the input text, watches how the model's predictions change, and then fits an understandable linear model to closely match how the original model behaved in that case. This method shows readers which words or phrases have the most weight in the model's choice, which helps them understand why a story is labeled as real or fake. The fact that LIME is easy to understand builds trust and lets people make better decisions about how to spot fake news.

Step 1: Model Prediction

Given an input instance x, we have a complex machine learning model f that makes predictions. The goal of LIME is to explain the prediction f(x). For a fake news detection model, f(x) might be the probability that a news article is classified as fake.

$$f(x) = \hat{y}$$

where$\hat{y}$ is the predicted output.

Step 2: Generate Perturbations

LIME generates a set of perturbed instances around the original instance x to understand how the model behaves locally.

$Let\ Z = \{z\_1, z\_2, \ldots, z\_n\}$ represent these perturbed instances created by adding noise or modifying features in x.

For each perturbed instance z_i:

$$z\_i = x + \varepsilon\_i$$

Where ε_i represents the perturbation applied to the original instance.

Step 3: Model Predictions on Perturbations

The complex model f predicts the output for each perturbed instance z_i. Thus, we have a set of predictions:

$$f(z\_i) = \hat{y}\_i\ for\ all\ z\_i\ in\ Z$$

These predictions help understand the model's behaviour around the original instance x.

Step 4: Weighting Perturbed Instances

LIME assigns weights to each perturbed instance based on its similarity to the original instance x. A similarity kernel π(x, z) is defined, typically using an exponential kernel function:

$$\pi(x, z) = \exp\left(-\frac{D(x, z)^2}{\sigma^2}\right)$$

where D(x, z) is the distance between x and z, and σ is a scaling factor. Instances closer to x receive higher weights, indicating their greater relevance.

Step 5: Train an Interpretable Model

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

LIME fits an interpretable linear model g (e.g., linear regression) to approximate f around x using the weighted perturbations. The objective is to minimize the weighted loss function:

$$L(g) = \Sigma_{i=1}^{n} \pi(x, z_i) * \left(f(z_i) - g(z_i)\right)^2$$

where g(z_i) is the prediction of the linear model for instance z_i. The goal is to find g that best approximates f locally.

Step 6: Generate Explanation

The resulting linear model g provides coefficients β that indicate the contribution of each feature to the prediction:

$$g(z) = \beta_0 + \Sigma_{j=1}^{m} \beta_j z_j$$

where β_j represents the importance of feature j in the local explanation around x.

These coefficients β_j form the interpretable explanation, showing which features influenced the model's prediction for fake news detection.

V. Result Analysis and Discussion

Figure 7 shows the CNN model's training and validation accuracy and loss over 10 iterations for finding fake news. The accuracy graph (left) shows that both the training and evaluation accuracy slowly rise until they hit over 98%. This shows that the model is learning well. The validation accuracy is very close to the training accuracy, which means that there is good adaptation without a lot of over fitting. The loss line on the right shows that both training and validation loss are going down. This means that as training goes on, the model's mistake goes down. The validation loss stays close to the training loss, which means that the model keeps doing the same job on data it hasn't seen before. In general, the model does a great job of learning and shows almost no signs of overfitting.

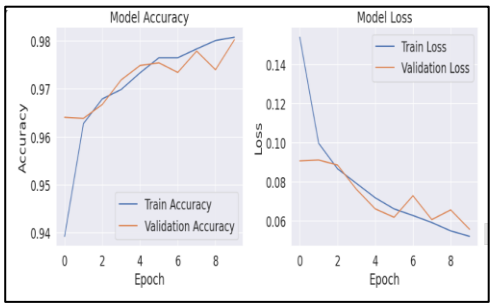


Figure 7: CNN model accuracy and loss comparison

Over 10 epochs, the plots show how the CNN model learned with Word2Vec embeddings changed over time in terms of accuracy and loss. The accuracy graph (left) shows that both training and validation accuracy keep getting better until they hit about 98%. This shows that the model learns and applies what it has learned from the Word2Vec representations. The fact that training and confirmation accuracy are very close to each other means that there is not much over fitting. The loss graph as illustrate in figure 8, on the right shows that both the training loss and the validation loss are steadily going down. This means that the model is

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

getting better with each epoch. The validation loss stays close to the training loss, which shows that the model still works well on data it hasn't seen before. In general, adding Word2Vec embeddings has made the model better at finding fake news that is correct and can be used in other situations.

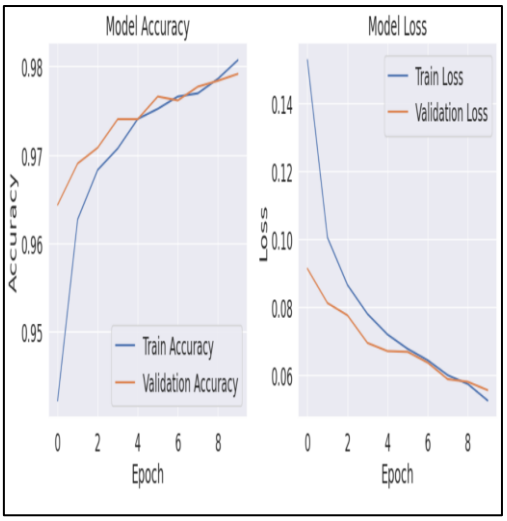


Figure 8: CNN model accuracy and loss comparison using Word2VEC Embedding's

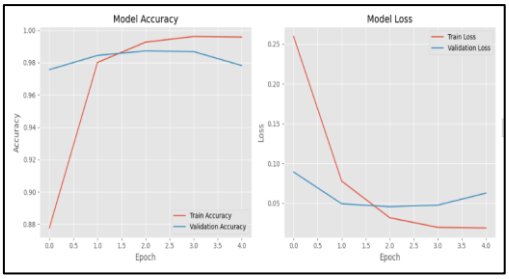


Figure 9: RNN model accuracy and loss comparison

The training accuracy goes up quickly in the accuracy graph (left), as shown in figure 9, hitting almost 100% by the third epoch. This shows that the model learns the training data well. The confirmation accuracy, on the other hand, stays around 98% and starts to go down after the third epoch, which could mean that the model is too good. The difference in accuracy between training and validation data shows that the model does well on training data, but it might not be as good at generalizing to data it hasn't seen before. The training loss drops rapidly in the loss graph (right), as shown in figure 9, getting close to zero by the fourth epoch. This shows that the model is learning well from the training data. The validation loss, on the other hand, goes down at first but starts to rise after the third epoch. This shows that the model is overfitting and starting to remember things instead of generalizing them. The RNN model does well on training data, but its validation loss is going up and its validation accuracy is staying the same. This suggests that it needs regularization methods, like dropout, to make it more general and stop it from overfitting when it comes to finding fake news.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

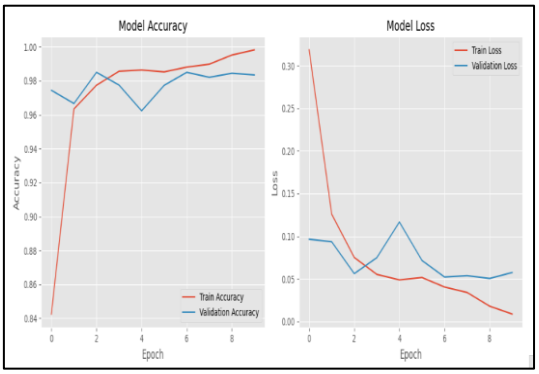


Figure 10: RNN model accuracy and loss comparison using Word2VEC Embedding's

Figure 10 shows plots that show the accuracy and loss of an RNN model using Word2Vec embeddings during training and evaluation over 10 "epochs."The left accuracy curve shows that the training accuracy quickly rises to almost 100%, which means that the model learns well from the training data. The confirmation accuracy goes up to about 98% by the third epoch and stays that way, which means it works well with data it has not seen before. The model is using the Word2Vec embeddings to effectively describe features, as shown by the steady rise in accuracy between training and validation. The training loss steadily goes down in the loss graph (right) until it gets close to zero. This means that the model fits the training data well. But the validation loss changes over time, especially after the fourth phase, which could mean that the model is too good. The confirmation loss is still low compared to the training loss, even with these changes. The using Word2Vec embeddings makes it easier for the RNN model to find environmental links in fake news spotting. However, regularization methods may be needed to make generalization even better.

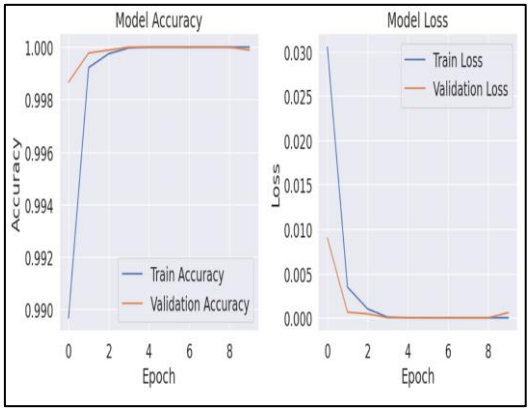


Figure 11: CNN+RNN model accuracy and loss comparison using Word2VEC Embedding's

In the accuracy graph (left), both training and validation accuracy rapidly reach almost 100% within the first few epochs, indicating that the model is highly effective in learning and generalizing the patterns in the data. The validation accuracy closely follows the training accuracy, suggesting excellent performance on unseen data without over fitting. In the loss graph (right), both training and validation loss decrease sharply and converge to nearly zero by the 3rd epoch. This minimal loss indicates that the model has learned the data exceptionally well, achieving high precision, confusion matrix illustrates in figure 12.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

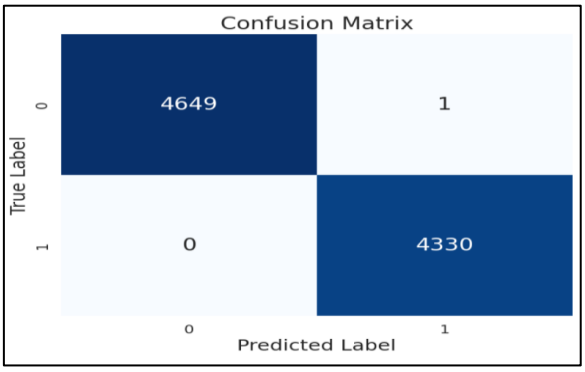


Figure 12: Confusion Matrix for CNN + RNN

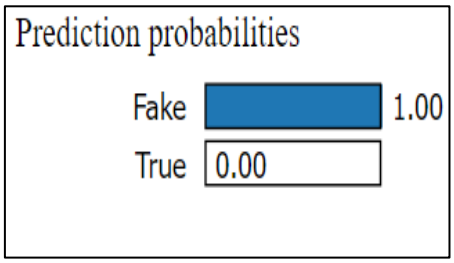


Figure 13: XAI

The image shows in figure 13, which features are most important for a model that uses AI methods that can be understood, such as LIME or SHAP, to find fake news. Both the "Fake" and "True" sides of the graph show the most important factors in deciding whether news is fake or true. Each feature (like feature_64 and feature_186) has a baseline value that tells you the range of values that have a big effect on the ranking. The feature inputs are shown with bars that show how important they are, with larger bars showing more value. This image helps you see which traits are most important to the model's decision-making process. It also shows you how the model tells the difference between fake and real news.

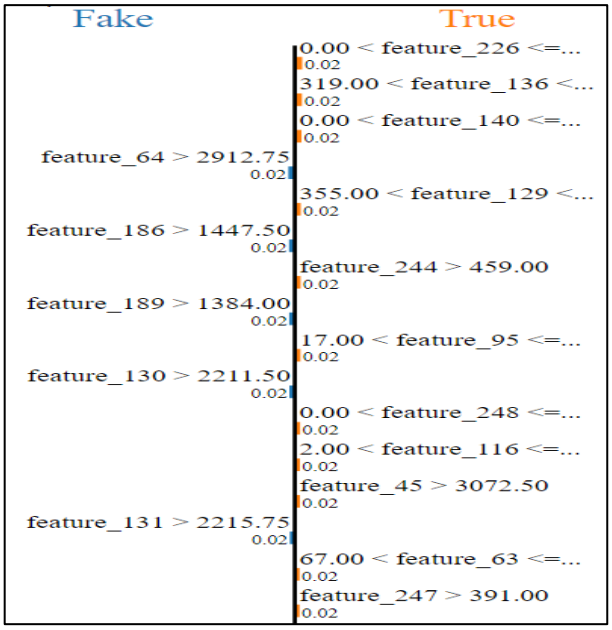


Figure 14: Fake new detection using XAI

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

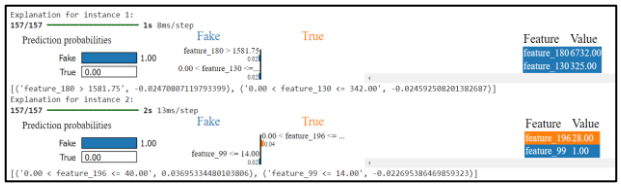


Figure 15 Prediction probability by XAI

Figure 14 shows in detail how two examples were categorized by a model for finding fake news using an interpretability tool such as LIME or SHAP. This helps us understand how the model makes its decisions. In Case 1, the model says there is a 100% chance that the story is "Fake." With a number greater than 1581.75, feature_180 is the most important factor and has a big impact on the ranking. feature_130, which is between 0 and 342, is another factor that plays a role, though it has a slightly smaller effect. The numbers shown for features show that feature_180 has a high value of 6732, which is very close to the fake name. In Instance 2, the model says again that it is 100% sure that the story is "Fake." Feature_196, which has a value less than 40, is very important here, followed by feature_99, which has a value less than 14. The feature values show that feature_196 has a significant value of 19628, which supports the false classification. These answers make the model's findings clear by showing which traits and their values are most important for classifying news as fake. This ability to be understood is important for gaining trust in the model, especially in high-stakes situations where knowing why predictions are made is just as important as the predictions themselves.

VI. Conclusion

This research shows that using Explainable AI (XAI) methods along with deep learning models, especially CNNs and RNNs, can help find fake news more easily. By using XAI techniques, we were able to not only find fake news very accurately, but we also learned a lot about the complicated language patterns and semantic structures that set fake news apart from real news. Traditional deep learning models are often "black boxes," which makes it hard to figure out how they make decisions. Using XAI methods like LIME and SHAP, on the other hand, helped us understand the model's forecasts and find important traits and trends that affect the classification, making it clearer and more reliable. Our mixed method, which combines the best features of CNNs for finding local word patterns and RNNs for figuring out how events depend on each other, worked very well for dealing with the complicated language used in fake news. Using XAI proved that the model could pick up on minor language cues, rhetorical structures, and errors that are often signs of false information. This complete understanding not only makes systems that look for fake news more reliable, but it also gives people who have a stake in the issue, like writers, fact-checkers, and lawmakers, information that they can understand and use. Combining AI methods that can be explained with deep learning provides a strong and clear way to deal with the growing problem of fake news. It makes sure that the model's choices are correct and easy to understand, which leads to more trust and use in real-world situations. Researchers can look into how XAI can be used with other advanced neural designs in the future. This could help find false information even better across a wider range of subjects and languages.

Received: August 05, 2025

Volume 38 No. 7s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

References

[1] M. Park and S. Chai, "Constructing a User-Centered Fake News Detection Model by Using Classification Algorithms in Machine Learning Techniques," in IEEE Access, vol. 11, pp. 71517-71527, 2023, doi: 10.1109/ACCESS.2023.3294613.

[2] A. Sormeily, S. Dadkhah, X. Zhang and A. A. Ghorbani, "MEFaND: A Multimodel Framework for Early Fake News Detection," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 5337-5353, Aug. 2024, doi: 10.1109/TCSS.2024.3355300.

[3] P. Wei, F. Wu, Y. Sun, H. Zhou and X. -Y. Jing, "Modality and Event Adversarial Networks for Multi-Modal Fake News Detection," in IEEE Signal Processing Letters, vol. 29, pp. 1382-1386, 2022, doi: 10.1109/LSP.2022.3181893.

[4] B. Xie, X. Ma, J. Wu, J. Yang and H. Fan, "Knowledge Graph Enhanced Heterogeneous Graph Neural Network for Fake News Detection," in IEEE Transactions on Consumer Electronics, vol. 70, no. 1, pp. 2826-2837, Feb. 2024, doi: 10.1109/TCE.2023.3324661.

[5] R. Kozik, A. Pawlicka, M. Pawlicki, M. Choraś, W. Mazurczyk and K. Cabaj, "A Meta-Analysis of State-of-the-Art Automated Fake News Detection Methods," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 5219-5229, Aug. 2024, doi: 10.1109/TCSS.2023.3296627.

[6] C. Xu and M. -T. Kechadi, "An Enhanced Fake News Detection System with Fuzzy Deep Learning," in IEEE Access, vol. 12, pp. 88006-88021, 2024, doi: 10.1109/ACCESS.2024.3418340.

[7] W. Shahid et al., "Detecting and Mitigating the Dissemination of Fake News: Challenges and Future Research Opportunities," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 4649-4662, Aug. 2024, doi: 10.1109/TCSS.2022.3177359.

[8] A. Gupta et al., "Combating Fake News: Stakeholder Interventions and Potential Solutions," in IEEE Access, vol. 10, pp. 78268-78289, 2022, doi: 10.1109/ACCESS.2022.3193670.

[9] B. Jamshidi, S. Hakak and R. Lu, "A Self-Attention Mechanism-Based Model for Early Detection of Fake News," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 5241-5252, Aug. 2024, doi: 10.1109/TCSS.2023.3322160.

[10] A. H. J. Almarashy, M. -R. Feizi-Derakhshi and P. Salehpour, "Enhancing Fake News Detection by Multi-Feature Classification," in IEEE Access, vol. 11, pp. 139601-139613, 2023, doi: 10.1109/ACCESS.2023.3339621.

[11] X. Gao, X. Wang, Z. Chen, W. Zhou and S. C. H. Hoi, "Knowledge Enhanced Vision and Language Model for Multi-Modal Fake News Detection," in IEEE Transactions on Multimedia, vol. 26, pp. 8312-8322, 2024, doi: 10.1109/TMM.2023.3330296.

[12] F. Khan, R. Alturki, G. Srivastava, F. Gazzawe, S. T. U. Shah and S. Mastorakis, "Explainable Detection of Fake News on Social Media Using Pyramidal Co-Attention Network," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 4574-4583, Aug. 2024, doi: 10.1109/TCSS.2022.3207993.

[13] A. Silva, L. Luo, S. Karunasekera and C. Leckie, "Embracing domain differences in fake news: Cross-domain fake news detection using multi-modal data", Proc. AAAI Conf. Artif. Intell., vol. 35, no. 1, pp. 557-565, 2021.

Received: August 05, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[14] Y.-J. Lu and C.-T. Li, "GCAN: Graph-aware co-attention networks for explainable fake news detection on social media", 2020.
[15] K. Shu, D. Mahudeswaran, S. Wang and H. Liu, "Hierarchical propagation networks for fake news detection: Investigation and exploitation", Proc. Int. AAAI Conf. Web Social Media, vol. 14, pp. 626-637, 2020.
[16] M. Davoudi, M. R. Moosavi and M. H. Sadreddini, "DSS: A hybrid deep model for fake news detection using propagation tree and stance network", Expert Syst. Appl., vol. 198, 2022.
[17] S. Raza and C. Ding, "Fake news detection based on news content and social contexts: A transformer-based approach", Int. J. Data Sci. Analytics, vol. 13, no. 4, pp. 335-362, 2022.
[18] Z. Jin, J. Cao, Y.-G. Jiang and Y. Zhang, "News credibility evaluation on microblog with a hierarchical propagation model", Proc. IEEE Int. Conf. Data Mining, pp. 230-239, 2014.
[19] X. Ma et al., "A comprehensive survey on graph anomaly detection with deep learning", IEEE Trans. Knowl. Data Eng., Oct. 2021.
[20] G. Zhang et al., "efraudcom: An e-commerce fraud detection system via competitive graph neural networks", ACM Trans. Inf. Syst., vol. 40, no. 3, pp. 1-29, 2022.
[21] B. Xie, X. Ma, J. Wu, J. Yang, S. Xue and H. Fan, "Heterogeneous graph neural network via knowledge relations for fake news detection", Proc. SSDBM, pp. 15, 2023.
[22] DatasetUsed: https://www.kaggle.com/datasets/emineyetm/fake-news-detection-datasets

Received: August 05, 2025