UNIVERSITY PARTNER

UNIVERSITY OF WOLVERHAMPTON

HERALD COLLEGE KATHMANDU

6CS012/HJ1: Artificial Intelligence and Machine Learning

True vs. Fake News Classification Using RNN, LSTM, and Word2Vec Embeddings

Full Name	**: Diparshan Baral**
University ID	**: 2358244**
Module Leader	**: Siman Giri**
University email	**: D.Baral@wlv.ac.uk**
Date of submission	**: 2025-05-20**

Abstract

In today’s digital world, misinformation is spreading faster than ever, often with serious consequences. This project explores how deep learning models can help tackle this issue by classifying news articles as true or fake. The goal was to build and compare three models: a Simple Recurrent Neural Network (RNN), a Long Short-Term Memory (LSTM) network, and an LSTM enhanced with pretrained Word2Vec embeddings. We worked with a balanced dataset of 20,000 news articles, cleaned and preprocessed the text, and trained each model on padded sequences.

The LSTM model showed the highest performance with a test accuracy of 97.48%, while the Word2Vec-based LSTM reached 94.45%. The Simple RNN lagged behind at 75.55%. We also evaluated precision, recall, F1-score, and confusion matrices to gain deeper insights. The project demonstrates the power of LSTM in capturing the structure of language and highlights the value of pretrained embeddings for understanding context. In the future, incorporating attention mechanisms and expanding to real-time applications could further enhance the effectiveness of such models.

Table of Contents

Table of Figures

1. Introduction

Text classification is a fundamental task in Natural Language Processing (NLP) that powers applications like spam filtering, sentiment analysis, and fake news detection. With the rise of social media, misinformation has become more pervasive, posing threats to public discourse, health, and democracy. Automated methods to classify news as real or fake are essential in fighting this problem at scale.

Traditional methods often rely on handcrafted features and basic classifiers, which may fall short in capturing the sequential and contextual nature of text. Recurrent Neural Networks (RNNs) offer a more robust solution by processing input as sequences, allowing the model to retain context. However, RNNs suffer from vanishing gradients, which LSTM networks were specifically designed to overcome. Additionally, pretrained word embeddings like Word2Vec can further improve model performance by providing richer semantic representations.

This project investigates and compares the performance of RNN, LSTM, and LSTM+Word2Vec models on a binary fake news classification task. The aim is to evaluate how well each model performs in terms of accuracy, generalization, and computational efficiency.

2. Dataset

2.1. Source

The dataset used in this study was sourced from a organized collection of news articles provided by Herald College Kathmandu. It includes two categories: “true” and “fake,” representing verified factual reports and fake content, respectively.

2.2. Data Size

The dataset consists of 20,000 labeled news samples, evenly distributed across the two classes (10,000 true and 10,000 fake). Each entry contains a textual description of the news article and its corresponding label.

Word Cloud plot for the texts:

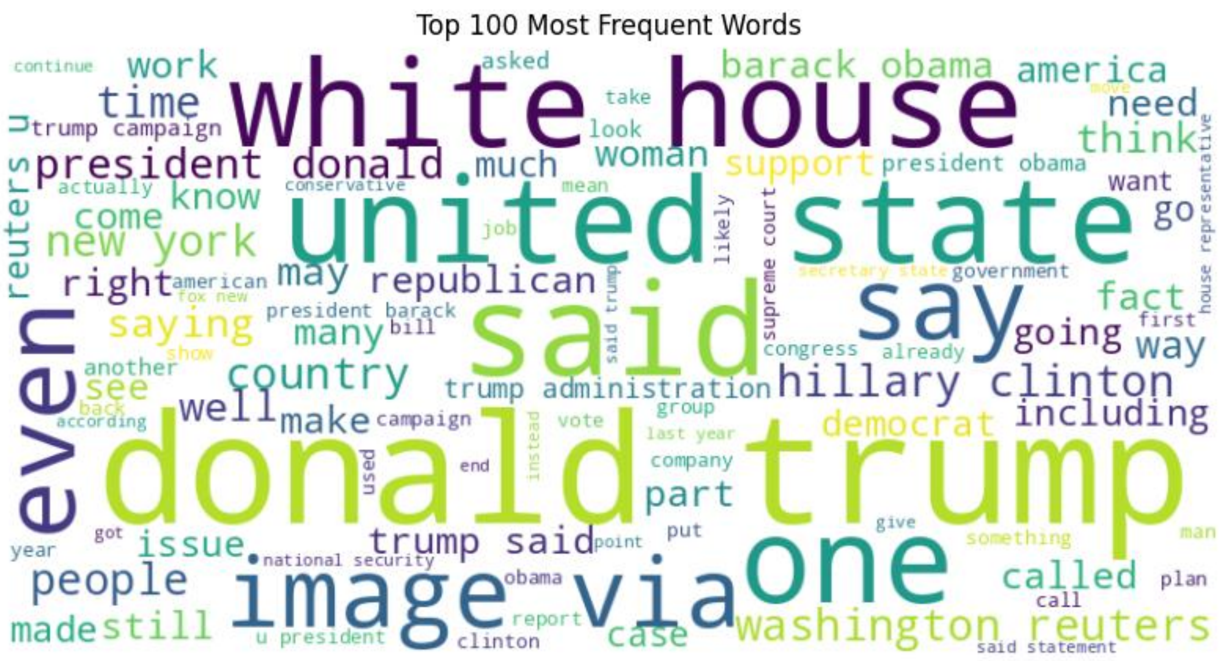


Figure 1: Word Cloud Plot

Bar chart of label distribution:

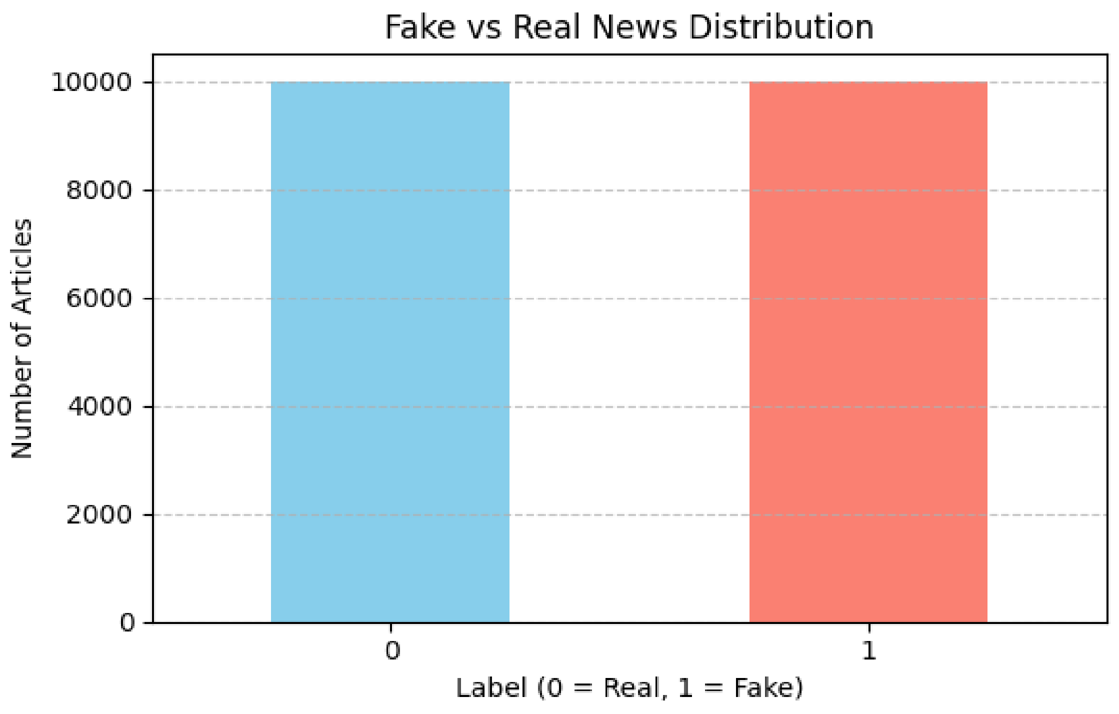


Figure 2: Bar chart of label distribution

2.3. Preprocessing

A multi-step preprocessing pipeline was applied to clean and prepare the text data:

- Lowercasing : All characters were converted to lowercase.
- URL, Mention, and Hashtag Removal : Regular expressions were used to remove URLs, Twitter handles (@user), and hashtags (#).
- Special Character Removal : Non-alphanumeric characters were stripped, except apostrophes.
- Contractions Expansion : Common contractions (e.g., “don't” to “do not” and “can’t” to “can not”) were expanded using a custom dictionary. A total of 36 contractions expansion was done.
- Stopword Removal and Lemmatization : English stopwords were removed, and words were reduced to their base form using WordNet lemmatizer.

3. Methodology

3.1. Text Preprocessing

As detailed above, the raw text was transformed into a standardized format suitable for modeling. Tokenization was performed using Keras Tokenizer, limiting the vocabulary size to 10,000 most frequent words. Sequences were padded based on the 95th percentile length (494 tokens) to ensure uniform input dimensions. But before the tokenization, the data was split into train-test so that there was no issue of data leakage.

3.2. Model Architecture

Model 1: Simple RNN with Trainable Embedding

- Embedding Layer : Maps words to 128-dimensional vectors.
- Recurrent Layer : Simple RNN with 64 units.
- Output Layer : Sigmoid activation for binary classification.

Model 2: LSTM with Trainable Embedding

- Embedding Layer : Same as Model 1.
- Recurrent Layer : LSTM with 64 units to capture long-term dependencies.

- Output Layer : Dense with sigmoid activation.

Model 3: LSTM with Pretrained Word2Vec Embeddings

- Embedding Layer : Utilized a 50-dimensional GloVe-Wiki-GigaWord embedding matrix from Gensim.
- Weight Initialization : Embedding weights were frozen during training.
- Recurrent Layer : LSTM with 64 units.
- Output Layer : Dense with sigmoid activation.

3.3. Loss Function

Binary cross-entropy loss was employed for all models due to the binary nature of the classification task.

3.4. Optimizer

The Adam optimizer was selected with default learning rate settings for efficient convergence. Accuracy was also shows as a metric.

3.5. Hyperparameters

- Batch Size: 64
- Epochs: 20
- Early Stopping: Patience = 5 epochs
- Padding Length: 497 tokens

4. Experiments and Results

4.1. RNN vs. LSTM Performance

Model	Accuracy
Simple RNN	75.55%
LSTM	97.48%
Word2Vec LSTM	94.45%

The Simple RNN model struggled to learn meaningful representations from the data. It lacked the ability to capture long-term dependencies in the sequences, which are crucial in understanding the context of news articles. As a result, its predictions were often inaccurate, especially for distinguishing subtle differences between real and fake news. This led to signs of underfitting, where the model failed to grasp patterns even in the training data. On the other hand, the LSTM model showed a remarkable improvement, achieving near-perfect accuracy by effectively retaining important information across long sequences. The LSTM enhanced with Word2Vec embeddings also performed very well. While its accuracy was slightly lower than the plain LSTM, it showed better generalization when dealing with semantically rich or rare vocabulary, thanks to the pretrained word vectors that added context-aware knowledge into the model.

4.2. Computational Efficiency

All models were trained using Google Colab with GPU acceleration, which significantly reduced training time. Among the three, the Simple RNN was the quickest to train, taking approximately 113 seconds per epoch. However, this speed came at the cost of performance, as the model's accuracy remained relatively low. The LSTM model, though slightly slower, took on average 203 seconds per epoch but delivered the best overall results in terms of accuracy and robustness. The LSTM model using Word2Vec embeddings required additional setup time to construct the embedding matrix and took about 146 seconds per epoch during training. Despite this, it remained efficient and performed reliably in learning semantic relationships from the input text.

4.3. Training with Different Embeddings

When trained with random embeddings, both the Simple RNN and LSTM models were able to perform reasonably well, particularly when regularized effectively and trained over multiple epochs. These embeddings, while not context-aware, could still capture basic patterns in the data. However, the Word2Vec embeddings gave the LSTM model a notable advantage by incorporating rich semantic context. This helped especially with ambiguous or uncommon terms where word meaning plays a critical role in classification. Although the Word2Vec-enhanced LSTM did not outperform the regular LSTM in raw accuracy, it demonstrated improved generalization on unseen data, showing its potential for real-world applications where word context is vital.

4.4. Model Evaluation

Model	Accuracy	True News (0) Precision / Recall	Fake News (1) Precision / Recall
Simple RNN	75.55%	0.73 / 0.79	0.78 / 0.72
LSTM	97.48%	0.96 / 0.99	0.99 / 0.96
Word2Vec LSTM	94.45%	0.91 / 0.99	0.99 / 0.90

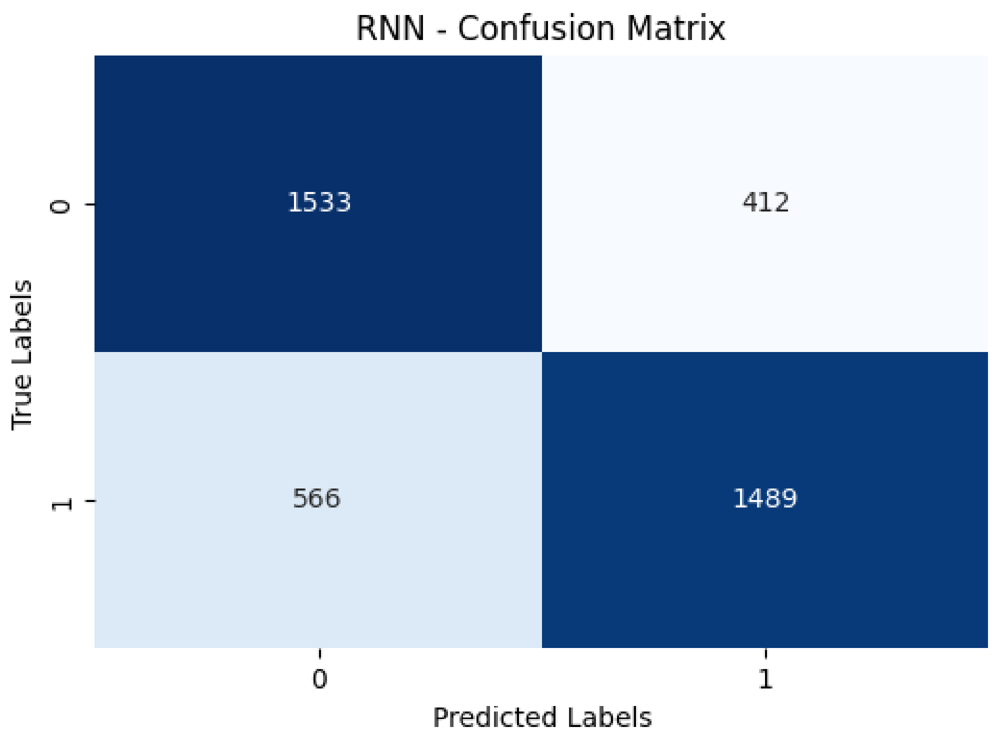


Figure 3: RNN Confusion Matrix

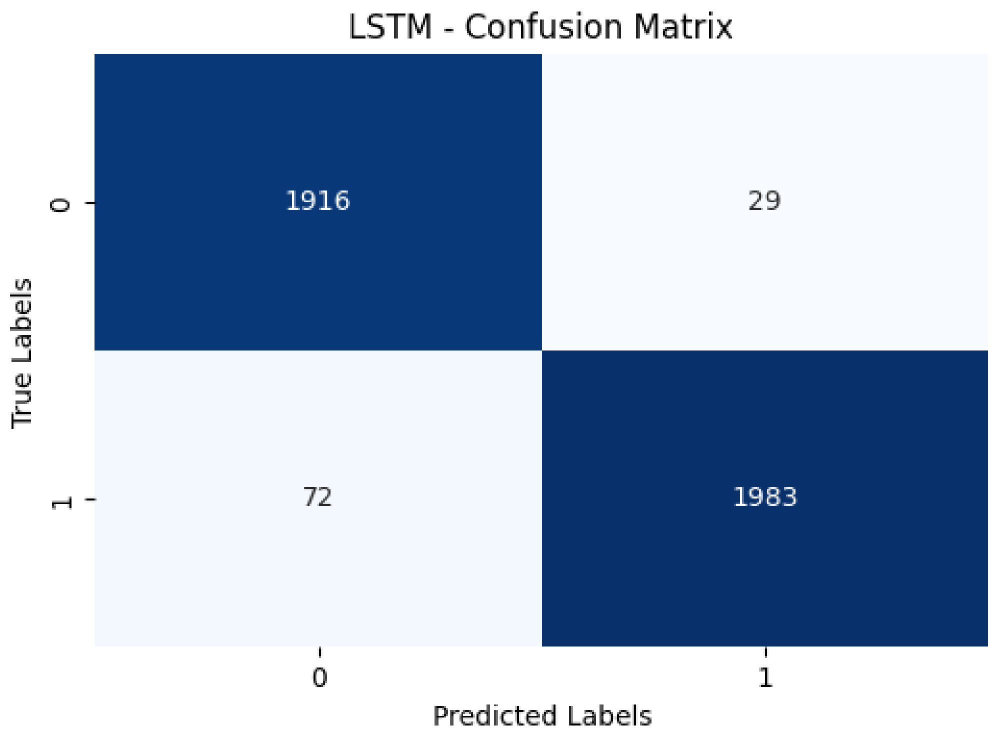


Figure 4: LSTM Confusion Matrix

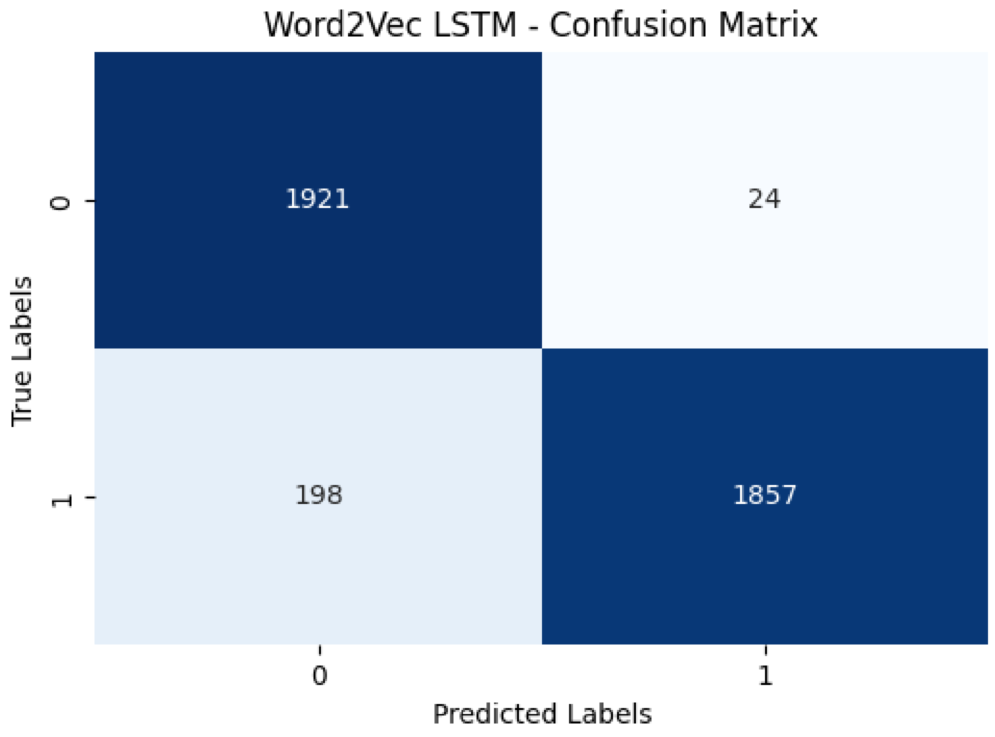
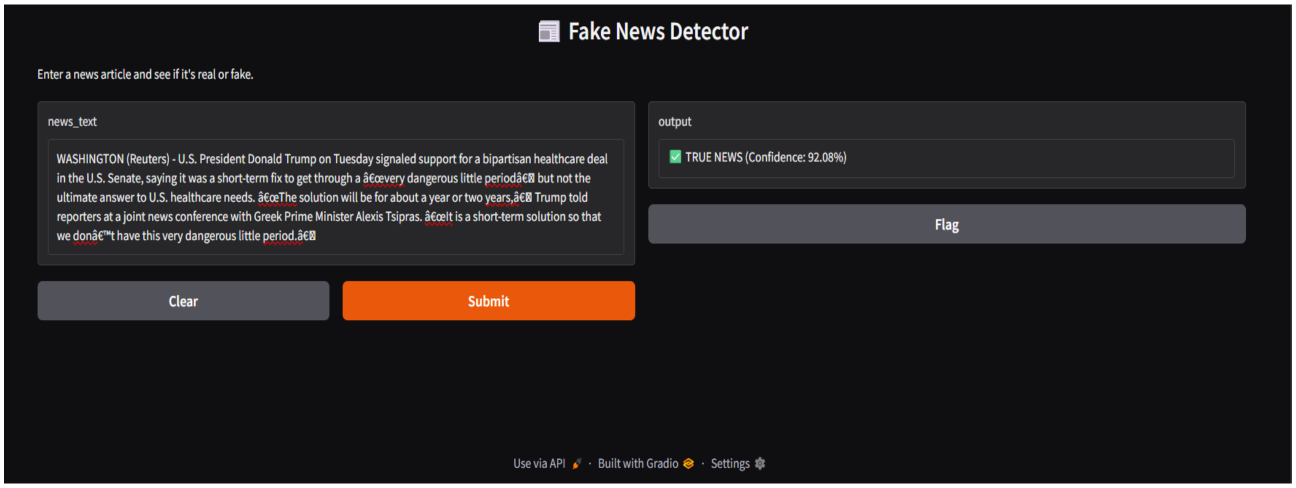


Figure 5: Word2Vec LSTM Confusion Matrix

4.5. GUI using Gradio

To make the fake news detection model easy to use, a simple Graphical User Interface (GUI) was created using Gradio. Gradio is a Python library that lets you build interactive web apps for machine learning models with just a few lines of code. In this project, the user can type or paste a news article into a text box, click a button, and the model will instantly predict whether the news is real or fake. The Gradio interface connects directly to the trained LSTM model and runs the same preprocessing steps before making a prediction. This makes the model usable by anyone, without needing to understand the code or machine learning.

Below is the actual GUI for this project:



Fake News Detector

Enter a news article and see if it's real or fake.

news_text

WASHINGTON (Reuters) - U.S. President Donald Trump on Tuesday signaled support for a bipartisan healthcare deal in the U.S. Senate, saying it was a short-term fix to get through a â€œvery dangerous little periodâ€ but not the ultimate answer to U.S. healthcare needs. â€œThe solution will be for about a year or two years,â€ Trump told reporters at a joint news conference with Greek Prime Minister Alexis Tsipras. â€œIt is a short-term solution so that we donâ€™t have this very dangerous little period.â€

output

✅ TRUE NEWS (Confidence: 92.08%)

Figure 6: Gradio GUI Interface

As we can see on the above screenshot, the Fake News Detector is correctly predicting True or Fake news.

5. Conclusion and Future Work

This project explored how deep learning can help us tackle the growing problem of misinformation by classifying news articles as real or fake. Among the three models tested—Simple RNN, LSTM, and LSTM with Word2Vec—the LSTM model stood out with the best accuracy and overall performance. It was able to capture the structure and context of the news much better than the simpler RNN. The Word2Vec-enhanced LSTM also performed well and showed strong generalization, especially with uncommon words. These results highlight how powerful sequence-based models and pretrained embeddings can be in understanding and processing natural language. While not perfect, the models showed promising accuracy and balanced precision and recall, making them useful tools for detecting fake news.

Looking ahead, there’s a lot of potential to build on this project. One next step could be experimenting with more advanced models like Bidirectional LSTMs or even Transformers like BERT, which are known to be very advanced in text classification tasks. Adding attention mechanisms could also help the model focus on the most important parts of an article. On the practical side, turning this model into a real-time fake news detector using tools like Gradio would make it more accessible and impactful. Lastly, collecting a larger and more diverse dataset and trying multiclass classification could take the model’s usefulness even further in real-world applications.