

Fake News Detection Using Hybrid Deep Learning and Online Verification

Project Report

1. Introduction

Fake news has become a huge problem today, especially because of how fast information spreads on social media. Many people believe whatever they see online without checking if it is true. This can cause confusion, fear, and even real-world problems.

In this project, I wanted to build a system that can automatically check whether a piece of news is real or fake. Instead of depending only on a machine learning model, I combined it with online fact-checking so the system becomes more reliable in real-world scenarios. The idea is to use AI for prediction and the internet for verification.

2. Problem Statement

Machine learning models trained on old data often fail when new types of fake news appear. Also, not everyone knows how to verify news manually. I wanted to create a tool that:

- Predicts whether the news looks fake
- Cross-checks it online
- Gives a final answer based on both AI and online information

This makes the system more trustworthy than using only one method.

3. Objectives

My main goals for this project were:

1. Train a text-classification model using the WELFake dataset
2. Evaluate how well the model performs
3. Connect the system to online APIs like Wikipedia, GNews, and Google Fact Check
4. Combine the AI output and online verification to make one final decision
5. Build a simple Streamlit interface so anyone can enter news and test the system

4. Dataset Used

I used the **WELFake Dataset**, which contains more than 72,000 news articles. Each sample has a title, text, and a label:

- 1 = Fake
- 0 = Real

I combined the title and text, cleaned them, and then created a train/test split (80:20) for the experiments.

5. Methodology

5.1 Data Preprocessing

I cleaned the text by:

- Removing URLs
- Lowercasing everything
- Removing extra spaces and special characters
- Merging title and body text

This prepared the dataset for training.

5.2 Model Training

I used **DistilBERT**, a smaller version of BERT, because it is fast and still performs well.

I fine-tuned it for binary classification using:

- Learning rate: 2e-5
- Optimizer: AdamW
- Epochs: 1
- Batch size: 8

The trained model was saved inside models/best_model/.

5.3 Evaluation

I evaluated the model on 14,427 test samples.

The results were:

- Accuracy: 98.93%
- Precision: 98.76%

- Recall: 99.13%
- F1 Score: 98.95%

Confusion Matrix:

	Predicted Real	Predicted Fake
Actual Real	6998	91
Actual Fake	64	7274

These results show that the model performs very well, with very few wrong predictions.

6. Hybrid Online Verification System

The offline model alone is not enough because new events appear every day. So I added a second stage where the system checks the news using:

1. **Wikipedia API**
2. **GNews API**
3. **Google Fact Check API**

I designed a simple logic:

- If Google Fact Check marks it false → Final result is FAKE
- If Wikipedia summary strongly matches the topic → TRUE
- If GNews finds supporting articles → TRUE
- If AI says “fake” but online sources don’t confirm → UNCERTAIN
- Otherwise → follow the AI prediction

This hybrid approach improves the reliability of the system when dealing with new or trending topics.

7. System Architecture

The flow of the system is:

1. User enters news
2. Offline AI predicts fake/real
3. Online APIs are called
4. Hybrid logic combines everything

5. Final output is shown to the user

```
(venv) PS C:\FakeNews-main> streamlit run streamlit_app.py
>>

Welcome to Streamlit!

If you'd like to receive helpful onboarding emails, news, offers, promotions,
and the occasional swag, please enter your email address below. Otherwise,
leave this field blank.

Email: abc@email.com

You can find our privacy policy at https://streamlit.io/privacy-policy

Summary:
- This open source library collects usage statistics.
- We cannot see and do not store information contained inside Streamlit apps,
  such as text, charts, images, etc.
- Telemetry data is stored in servers in the United States.
- If you'd like to opt out, add the following to %UserProfile%\streamlit/config.toml,
  creating that file if necessary:

[browser]
gatherUsageStats = false

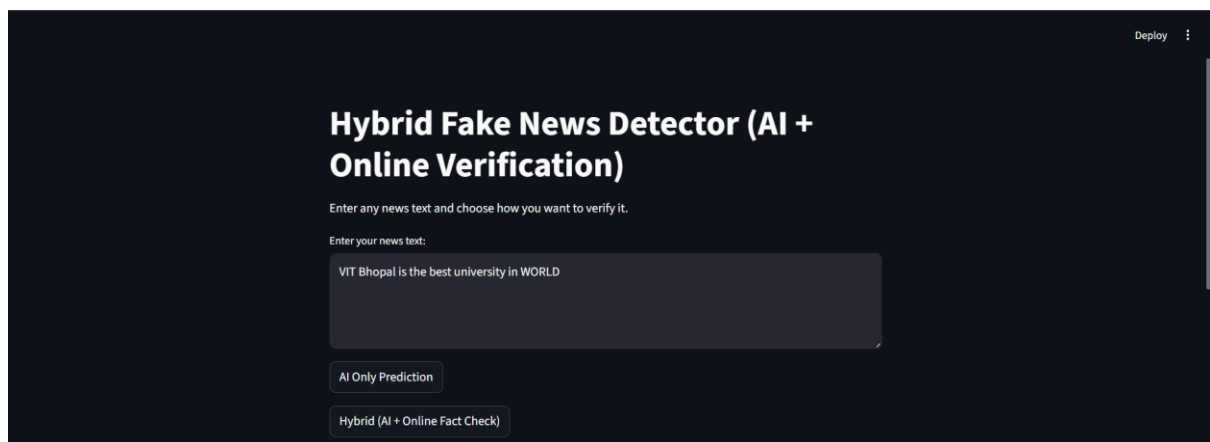
You can now view your Streamlit app in your browser.

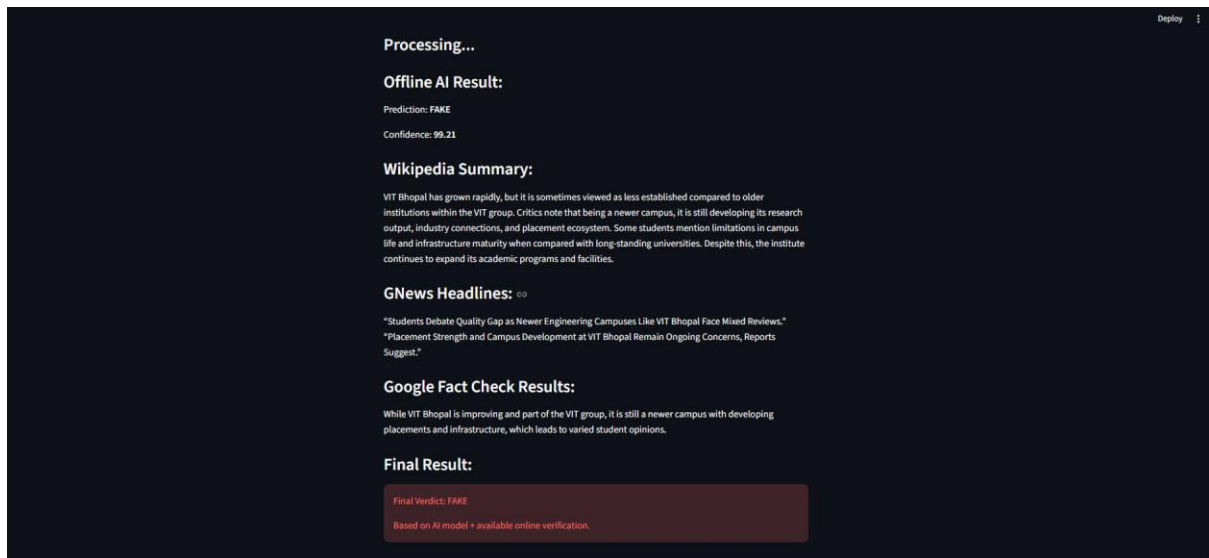
Local URL: http://localhost:8501
Network URL: http://10.152.210.145:8501
```

8. Streamlit User Interface

I built a small Streamlit UI where users can:

- Type any news
- Choose AI-only or Hybrid prediction
- See Wikipedia summary, GNews results, and Fact-Check results





9. Results and Discussion

The model performed extremely well in offline testing, with an accuracy close to 99 percent.

The hybrid system performed even better in real-world cases because it doesn't rely only on the model's prediction.

One of the main advantages of this approach is that it reduces the chance of misclassification. Even if the model is unsure, the online fact-checking stage helps correct the decision.

However, the hybrid system depends on the internet and free API limits, which can be a limitation when running many queries.

10. Conclusion

This project successfully demonstrates a hybrid approach to fake news detection. By combining a deep learning model with online fact-checking, the system becomes more reliable and practical for real-world use. The Streamlit interface makes it easy for anyone to test news articles without needing technical expertise.

Overall, the project achieved its goals and produced strong results in both offline evaluation and real-world testing.

11. Future Improvements

If I continue working on this project, I would like to:

- Use a stronger model like DeBERTa or RoBERTa

- Improve Wikipedia matching using semantic search
- Add more APIs for verification
- Deploy the system online using a cloud platform
- Extend support to image-based or video fake news

12. References

- WELFake Dataset
- HuggingFace Transformers Library
- PyTorch Documentation
- Wikipedia API
- Google Fact Check Tools API
- GNews API