Fake News Classification using BERT

Biswajit Paul

Ramakrishna Mission Vivekananda Educational and Research Institute

May 3, 2025

Abstract

In this work, we explore the use of Bidirectional Encoder Representations from Transformers (BERT) for the task of fake news classification. Using the WeLFake dataset, we construct a modular pipeline that includes data preprocessing, overlap mitigation, BERT-based fine-tuning, and comprehensive evaluation using accuracy, F1-score, ROC-AUC, and PCA-based visualization. Our results demonstrate strong classification performance and highlight the potential of transformer-based models in detecting misinformation.

Introduction

- The rise of fake news has posed significant challenges for digital media platforms, influencing public opinion and social discourse.
- Traditional machine learning techniques rely heavily on manual feature engineering, which may not capture semantic and contextual nuances of language.
- BERT, a transformer-based model, has demonstrated state-of-the-art performance in various NLP tasks due to its deep contextual understanding.
- This project aims to fine-tune a BERT model on the WeLFake dataset to detect fake news articles with high precision.

Dataset: WeLFake

- **WeLFake** is a comprehensive fake news dataset that combines data from multiple sources including Weibo and LIAR.
- It contains over **70,000+** news articles labeled as either `real (0)` or `fake (1)`.
- Each entry contains:
  - Title
  - Full text
  - Label (0 = Real, 1 = Fake)
- We split the dataset into training and test sets while handling overlapping entries to avoid data leakage.

Sample Records from WeLFake Dataset

ID	Title	Text	Label
0	LAW ENFORCEMENT ON HIGH ALERT Following Threat...	No comment is expected from Barack Obama Members of the House...	1
1	–	Did they post their votes for Hillary already?	1
2	UNBELIEVABLE! OBAMA'S ATTORNEY GENERAL SAYS MO...	Now, most of the demonstrators gathered last night...	1
3	Bobby Jindal, raised Hindu, uses story of Chri...	A dozen politically active pastors came here from...	0
4	SATAN 2: Russia unveils an image of its terrif...	The RS-28 Sarmat missile, dubbed Satan 2, will...	1

EDA: Label Distribution

Figure: Distribution of Fake vs Real News Labels

EDA: Text Length (Word Count)

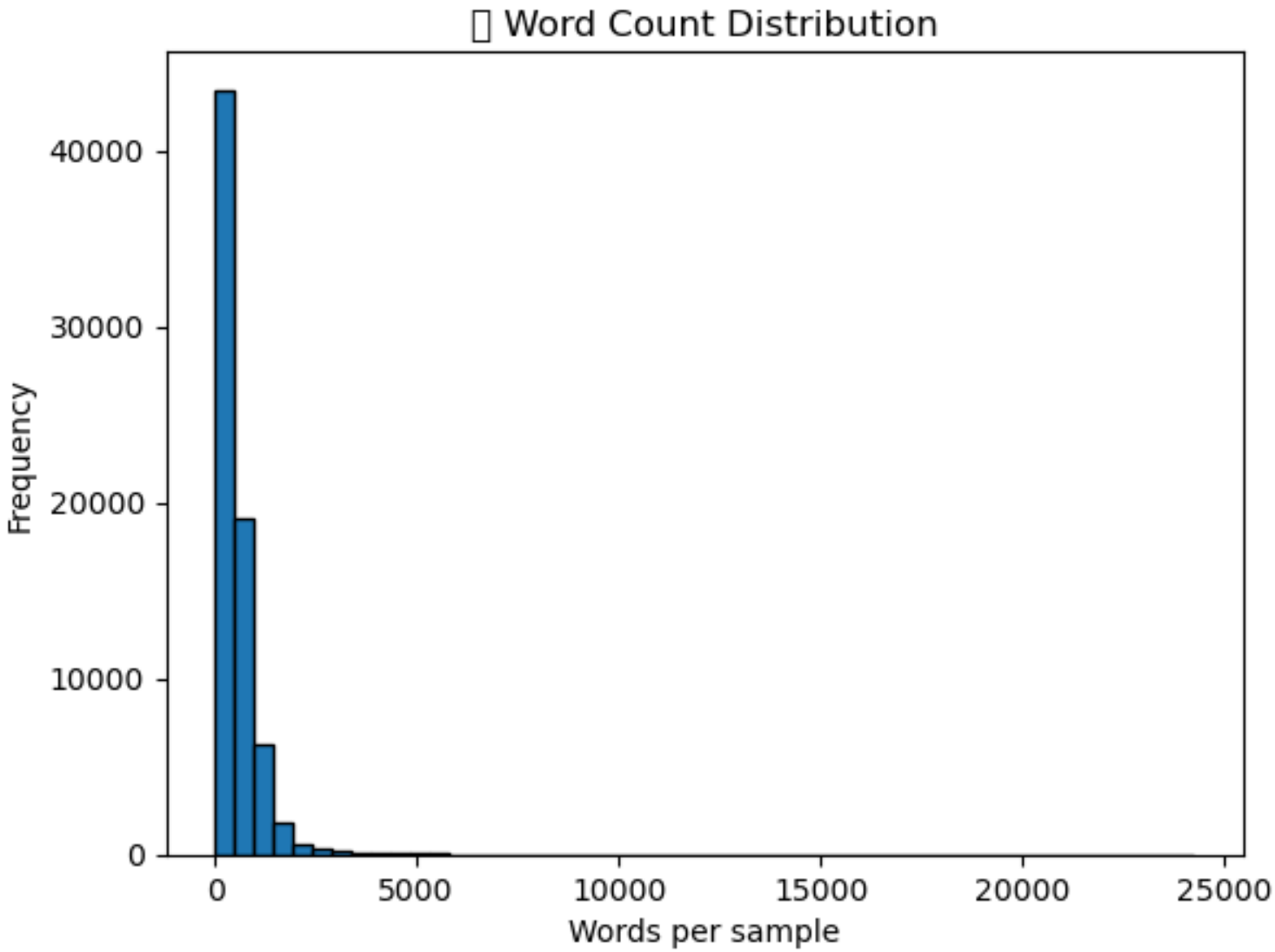


Figure: Histogram of Word Counts per Article

EDA: Token Length Distribution

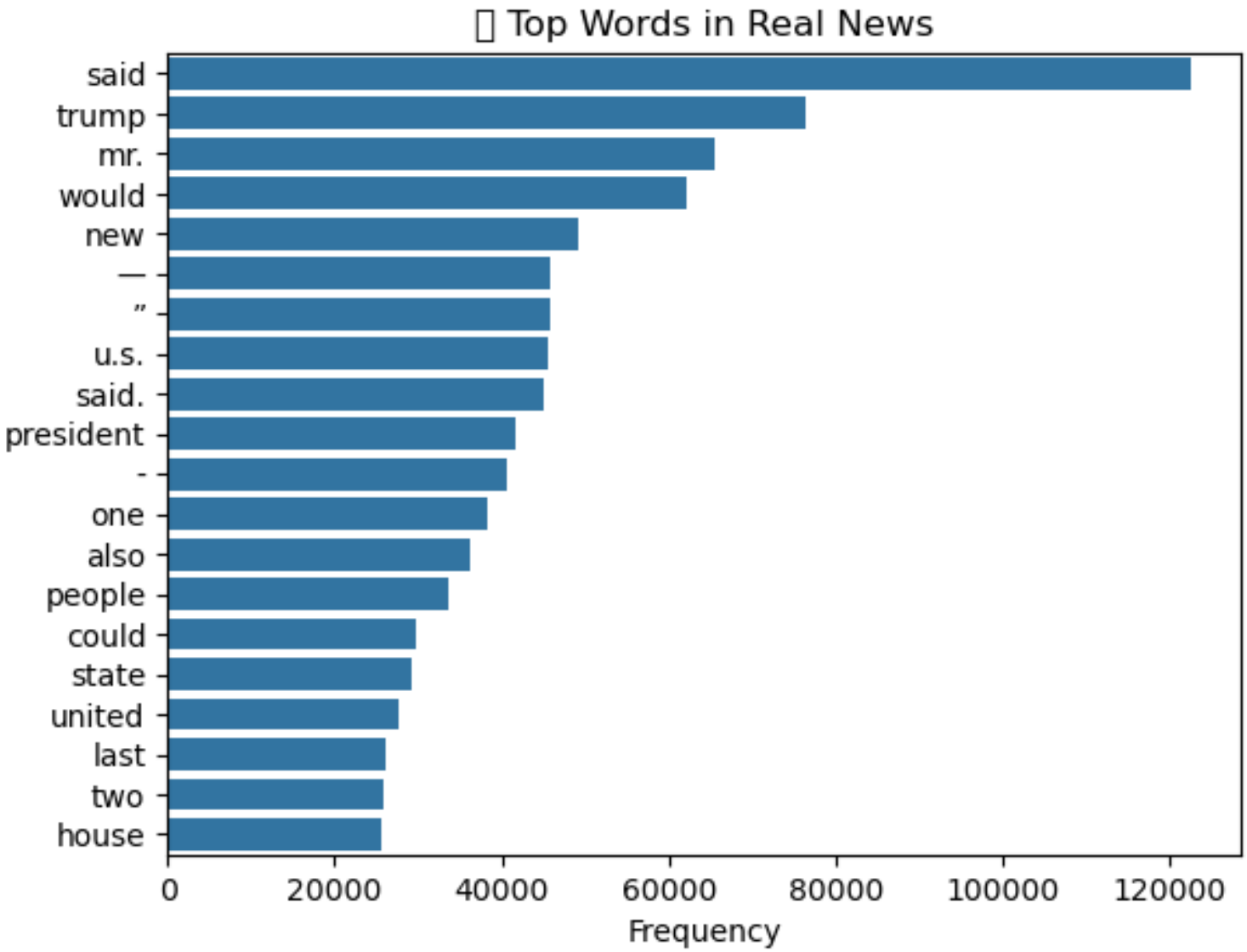


Figure: Distribution of Tokenized Input Lengths

EDA: Title Lengths

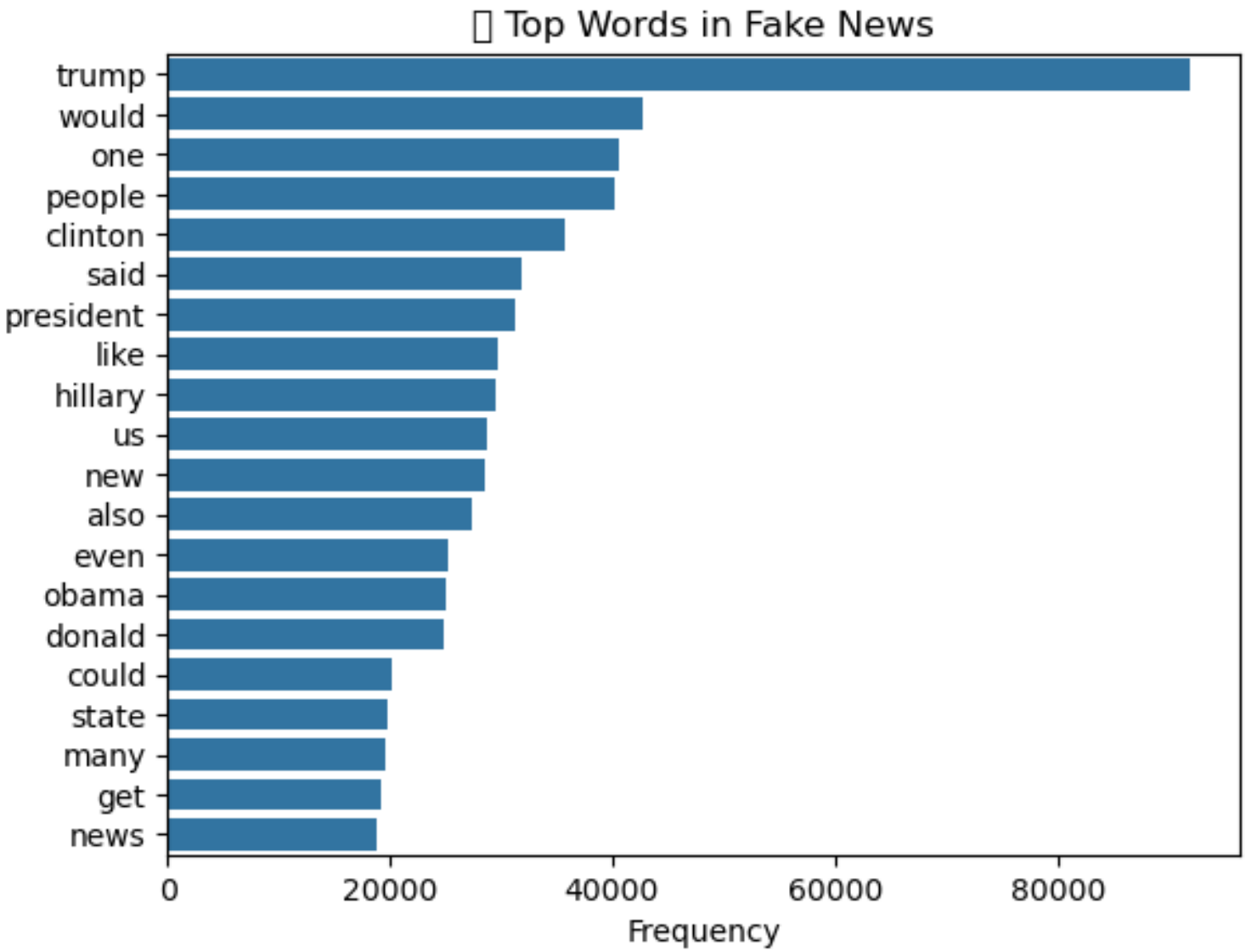


Figure: Histogram of Token Lengths in Titles

EDA: Text Body Lengths

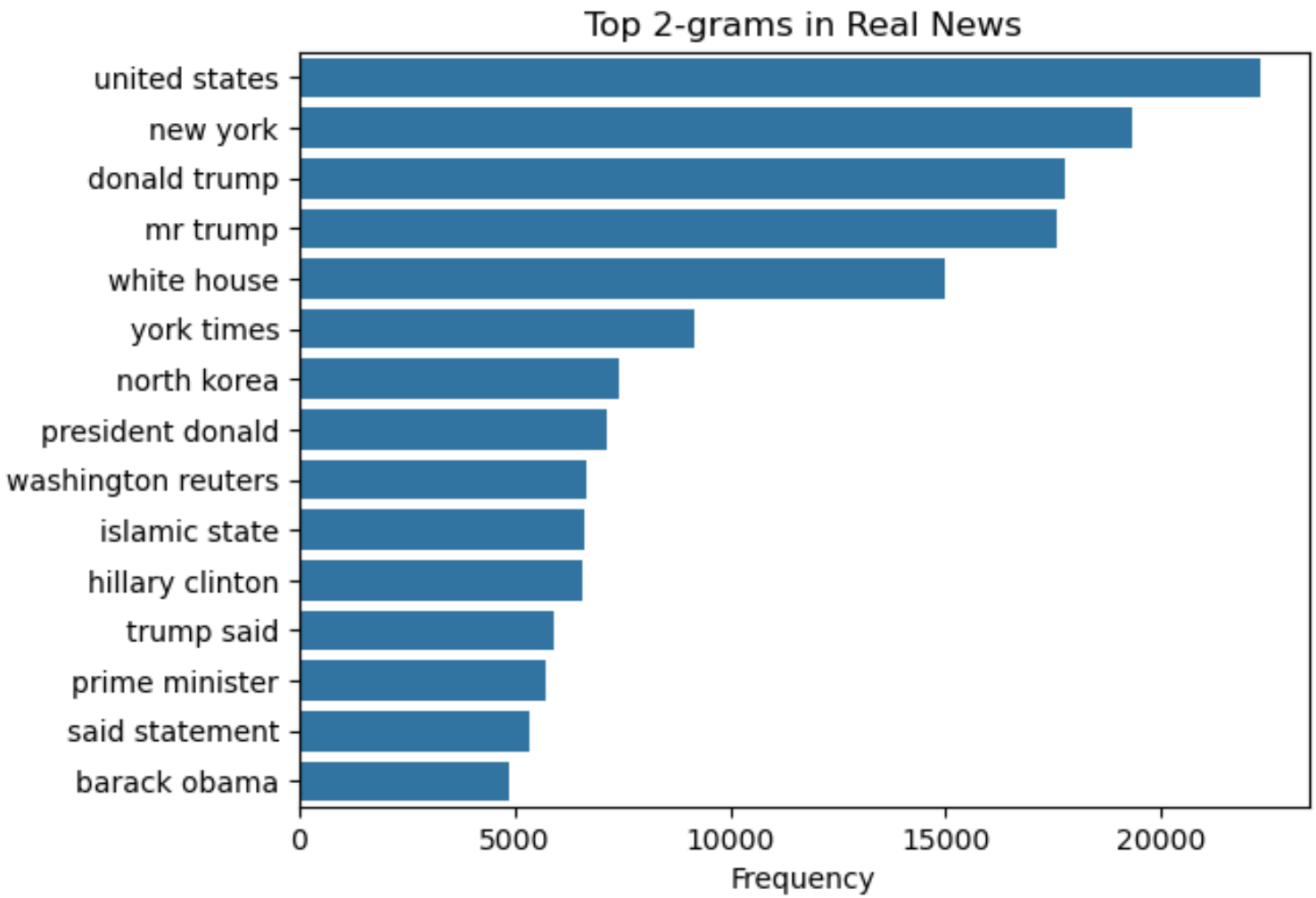


Figure: Histogram of Token Lengths in Text

EDA: Title vs Text Length (Combined)

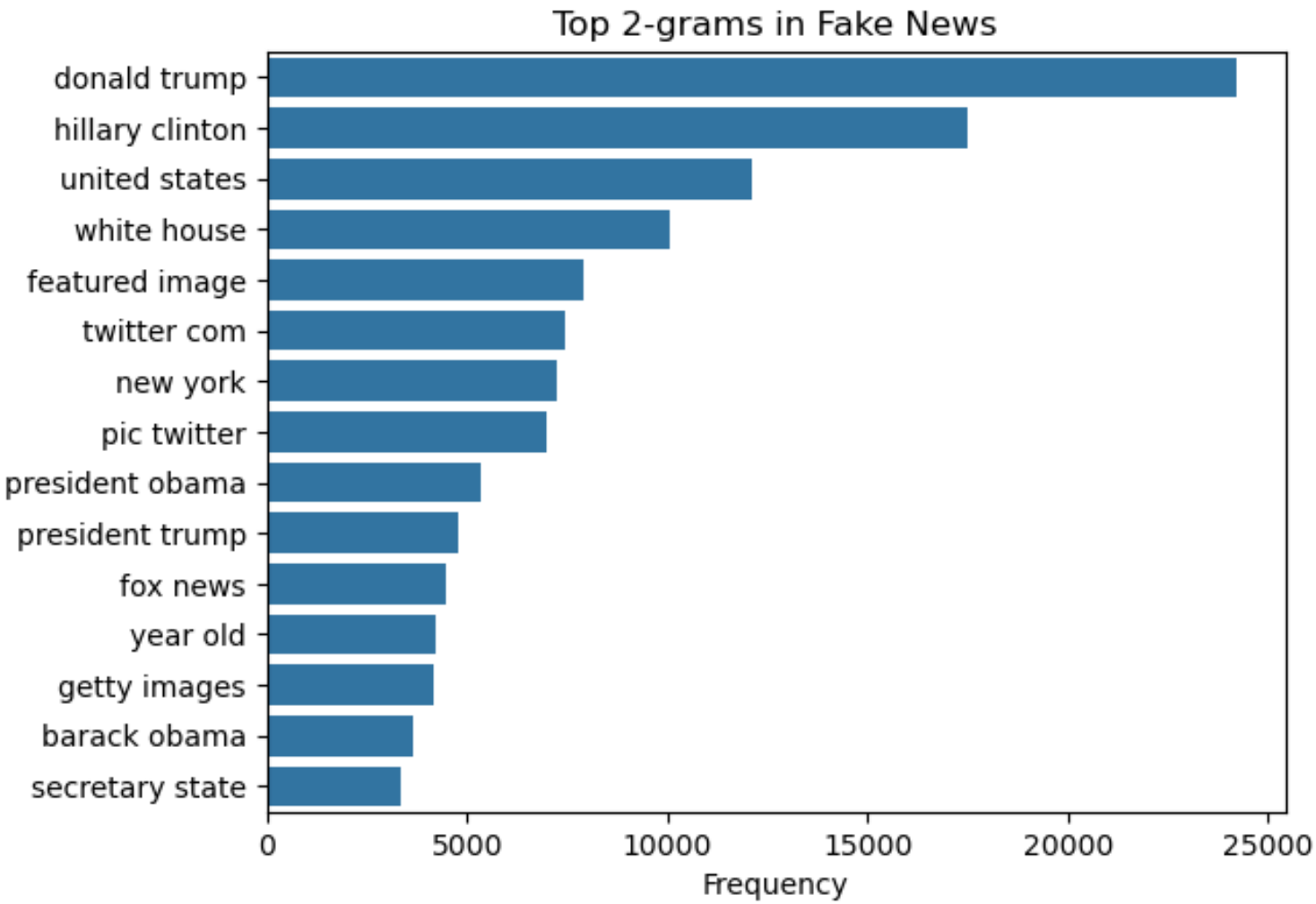
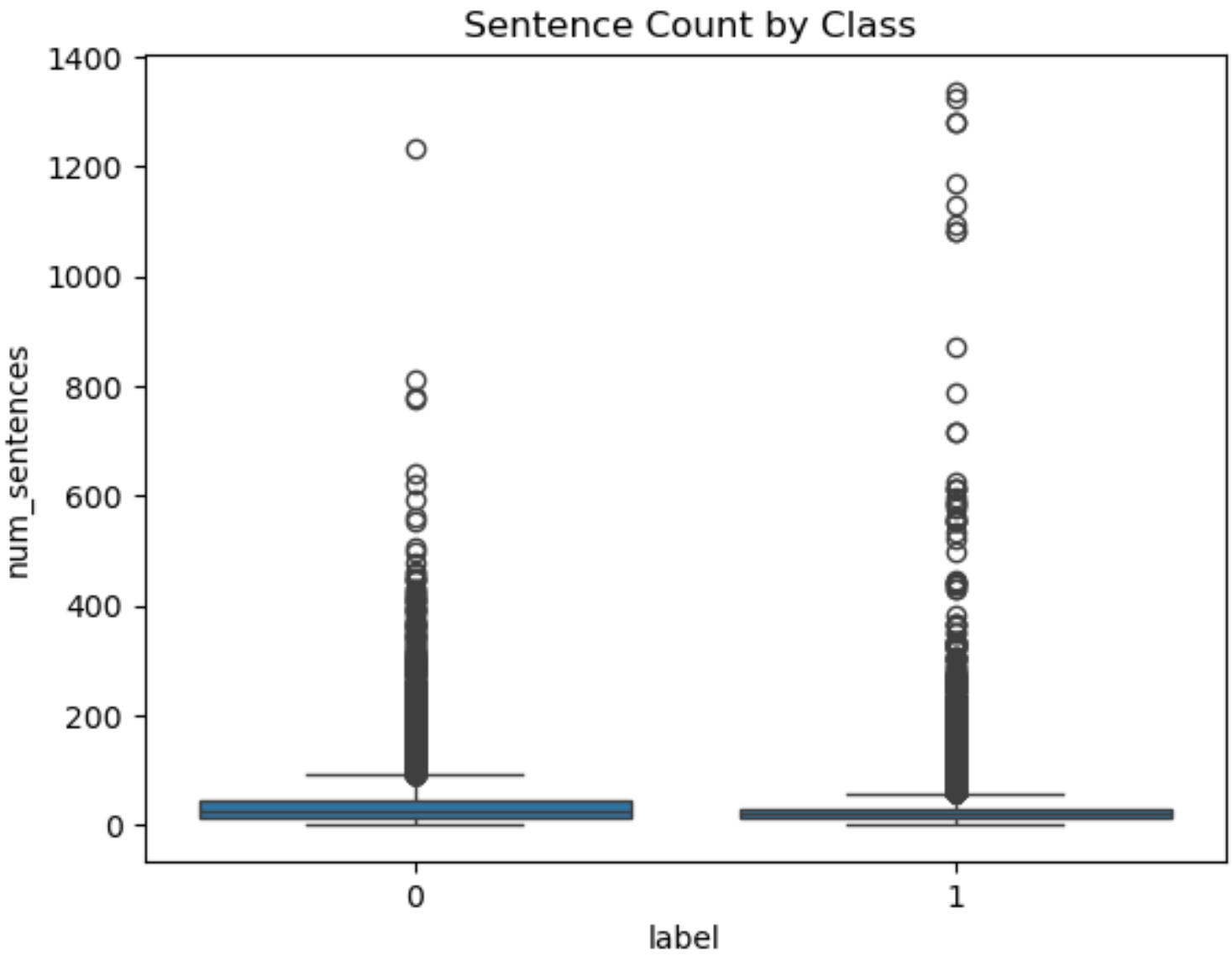


Figure: Overlay: Title Length vs Text Length

EDA: Frequent Words



EDA: Word Cloud of Common Terms

Figure: Word Cloud showing most frequent tokens in the news dataset

Text Cleaning and Preparation

- **What I did:**
  - Removed null titles and merged ‘title‘ and ‘text‘ fields.
  - Converted all text to lowercase and removed excess whitespace.
- **Why I did it:**
  - Titles alone are often short or missing; combining them ensures richer context.
  - Standardizing text improves tokenization and model stability.

Train-Test Split for TF-IDF Baseline

- **What I did:**
  - Applied an 80/20 split using `train_test_split` on the full dataset.
  - Ensured that both ‘title‘ and ‘text‘ were concatenated into a single input string.
- **Why I did it:**
  - Combining title and body improves the richness of textual features.
  - A clean split helps us evaluate how well simple statistical models perform before using deep learning.

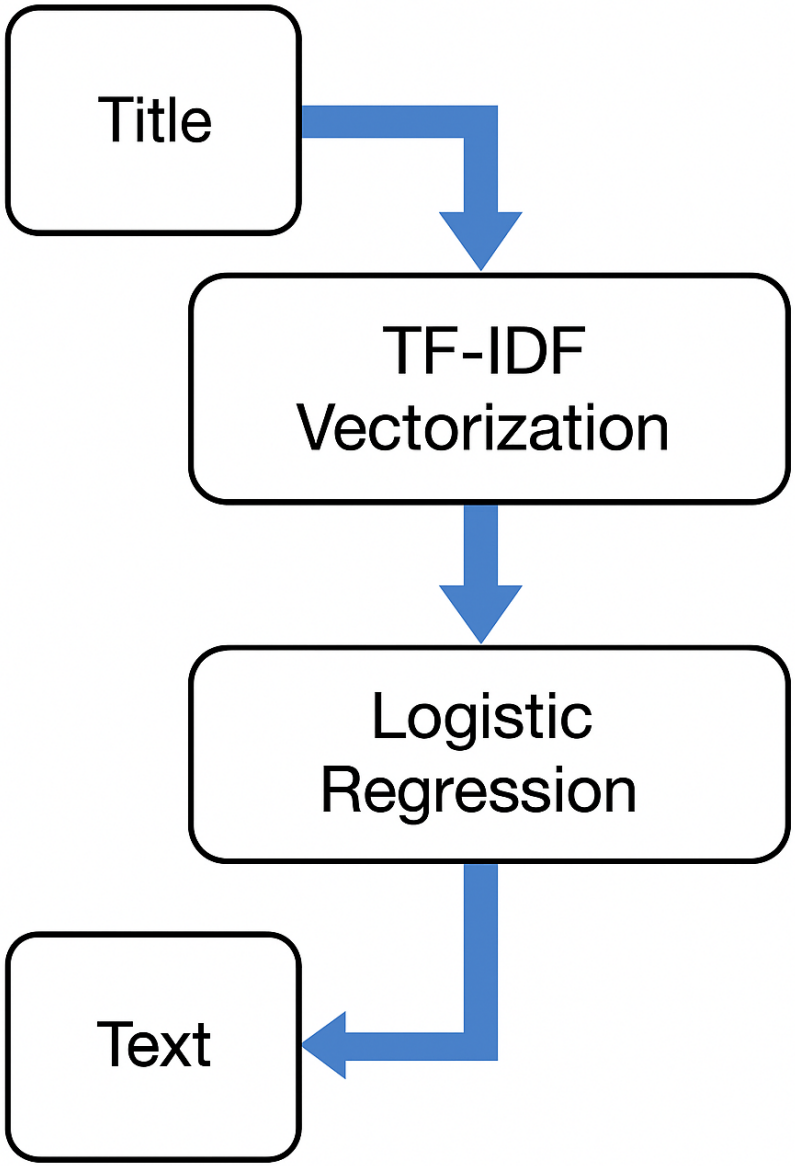
TF-IDF Vectorization

- **What I did:**
  - Applied TF-IDF vectorization with unigrams and bigrams.
  - Removed English stopwords and limited features using `max_features`.
- **Why I did it:**
  - TF-IDF captures term importance without considering word order.
  - Restricting to the top features reduces dimensionality and overfitting.

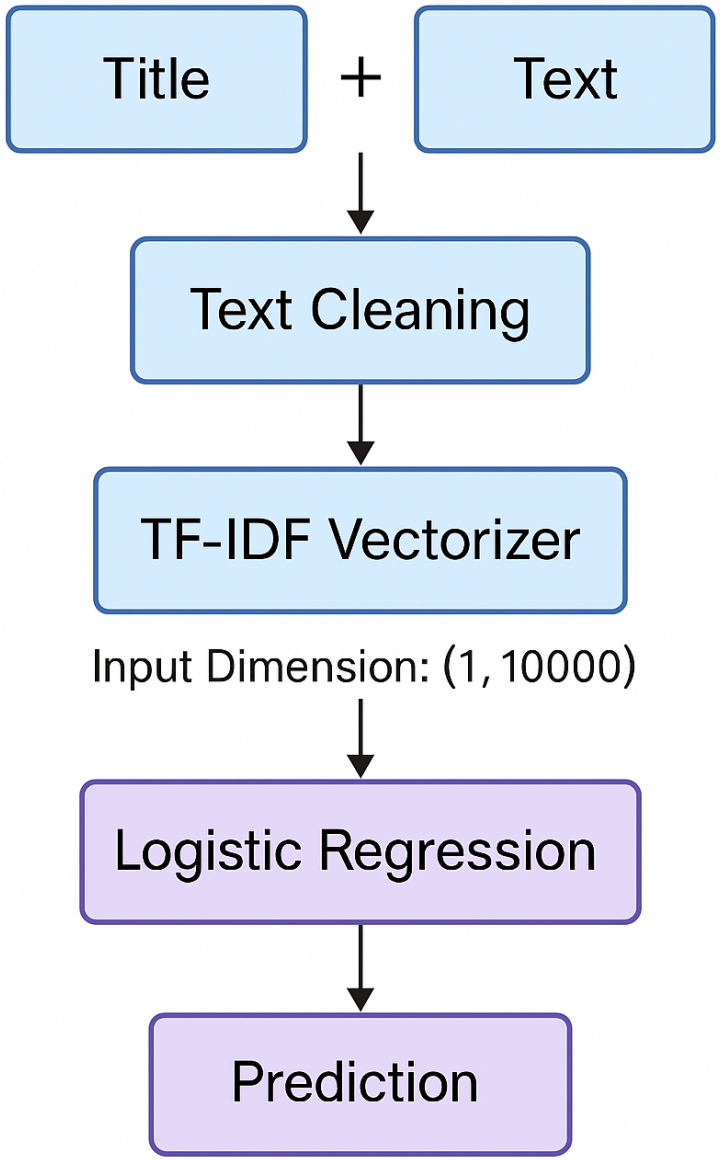
Logistic Regression for Fake News

- **What I did:**
  - Trained a scikit-learn Logistic Regression classifier on TF-IDF features.
  - Evaluated it using accuracy, F1-score, and confusion matrix.
- **Why I did it:**
  - Logistic Regression is a strong, interpretable baseline for binary classification tasks.
  - It provides a benchmark for comparing deep learning models like BERT.

TF-IDF + Logistic Regression Architecture



TF-IDF + Logistic Regression Architecture



How the Model Works

- **Input:** Title and text of news articles are concatenated into a single string.
- **Vectorization:** Each article is converted into a numerical feature vector using TF-IDF.
- **Classification:** A Logistic Regression model is trained on these vectors to predict whether the news is real (0) or fake (1).
- **Pipeline:** Preprocessing $\rightarrow$ TF-IDF Transformation $\rightarrow$ Model Training $\rightarrow$ Evaluation.
- The model learns weights for each word/phrase that correlate with fake or real news based on historical data.

Why I Implemented This Baseline

- **Interpretability:** The weights learned by Logistic Regression are human-readable and help identify influential terms.
- **Speed:** The pipeline is fast to train and test, ideal for comparison with deep models.
- **Benchmarking:** Provides a strong and reliable baseline to evaluate how much improvement BERT or LSTM models provide.
- **Simplicity:** Easy to deploy and doesn't require GPUs or complex dependencies.
- **Debugging:** Helps in debugging preprocessing and understanding data characteristics early on.

Explaining Key Terms

- **TF (Term Frequency):** How often a word appears in a document.
- **IDF (Inverse Document Frequency):** Penalizes common words across many documents.
- **TF-IDF Weight:** TF $\times$ IDF — highlights important, rare terms in context.
- **Feature Vector:** A high-dimensional sparse vector representing each article.
- **Coefficient (Weight):** In Logistic Regression, indicates the influence of a word on the prediction.
- **Bias Term:** A constant offset added to help the model shift the decision boundary.

TF-IDF + Logistic Regression Configuration

- **TF-IDF Vectorizer Settings:**
  - `stop_words='english'` — removes common stopwords to reduce noise.
  - `ngram_range=(1,2)` — captures unigrams and bigrams for richer representation.
  - `max_features=10000` — limits dimensionality to top 10k features by frequency.
- **Logistic Regression Settings:**
  - `class_weight='balanced'` — adjusts weights inversely to class frequencies to handle imbalance.
  - `solver='liblinear'` — good for small datasets and L1/L2 regularization.
  - `max_iter=1000` — ensures convergence during training.
- **Why these settings?**
  - Helps improve generalization, handle noisy labels, and reduce overfitting on sparse TF-IDF vectors.
  - The balanced setting improves recall for underrepresented classes (e.g., fake news samples).

Evaluation: Classification Metrics

- **Accuracy:** Measures the percentage of correctly predicted samples.
- **Precision:** Proportion of predicted fake news that are actually fake.
- **Recall:** Proportion of actual fake news that were correctly identified.
- **F1-score:** Harmonic mean of precision and recall, balancing false positives and false negatives.
- **Why these matter:**
  - In fake news detection, high recall is critical to catch as many fake articles as possible.
  - High precision ensures we don't incorrectly flag real news.

Evaluation Summary: TF-IDF + Logistic Regression

Metric	Train Score	Test Score
Accuracy	0.967	0.956
Precision	0.969	0.954
Recall	0.959	0.958
F1-score	0.963	0.956

Table: Performance metrics for the TF-IDF + Logistic Regression model

Evaluation: Confusion Matrix (Visualization)

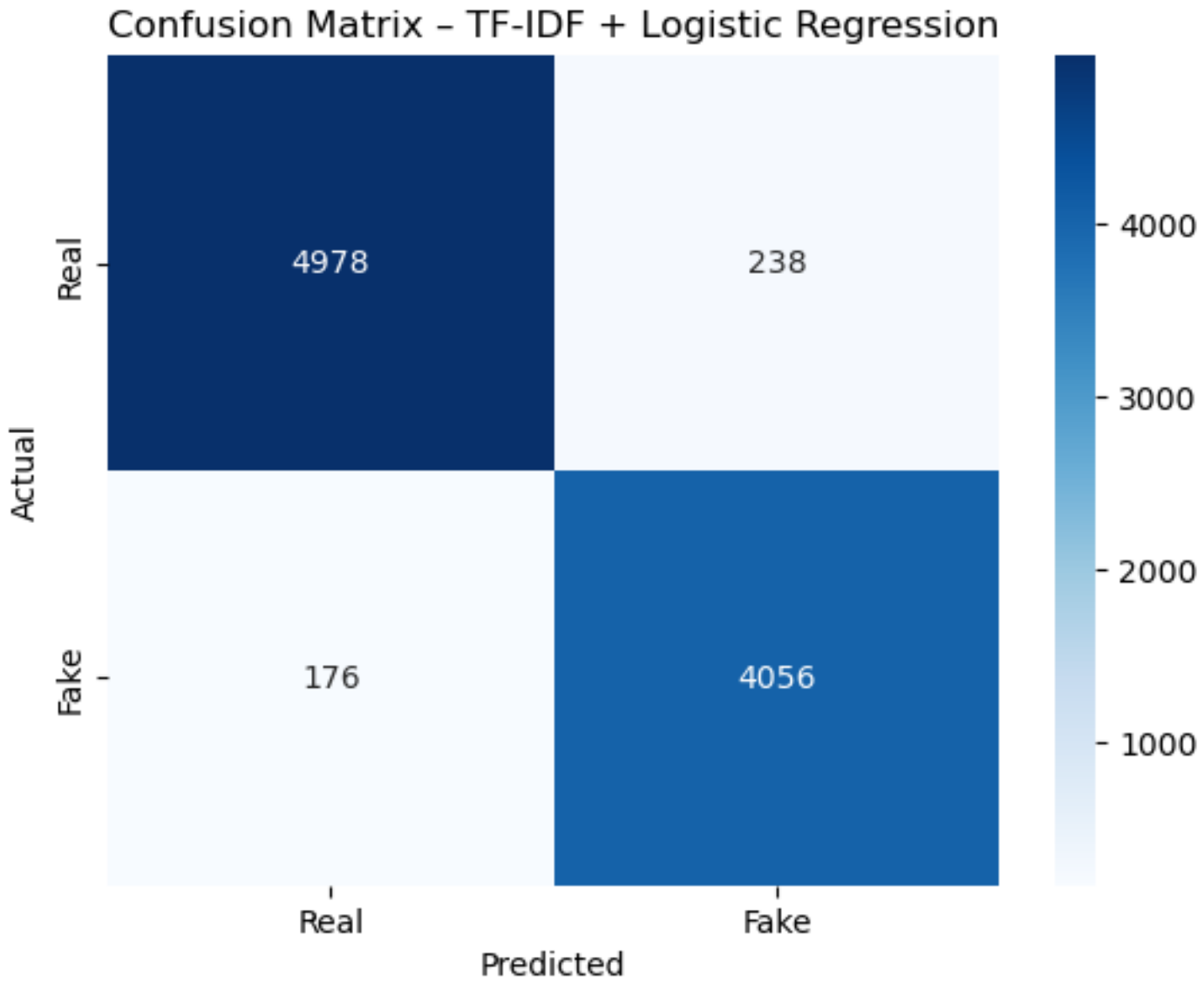


Figure: Confusion matrix showing true vs predicted labels

Evaluation: Confusion Matrix (Interpretation)

- **True Positives (Bottom-right):** Fake news correctly identified.
- **True Negatives (Top-left):** Real news correctly identified.
- **False Positives (Top-right):** Real news mistakenly predicted as fake.
- **False Negatives (Bottom-left):** Fake news missed and labeled as real.

Why this matters:

- Helps assess not just how accurate the model is, but how it's failing.
- Especially useful when real and fake classes are imbalanced.

Evaluation: ROC-AUC Curve

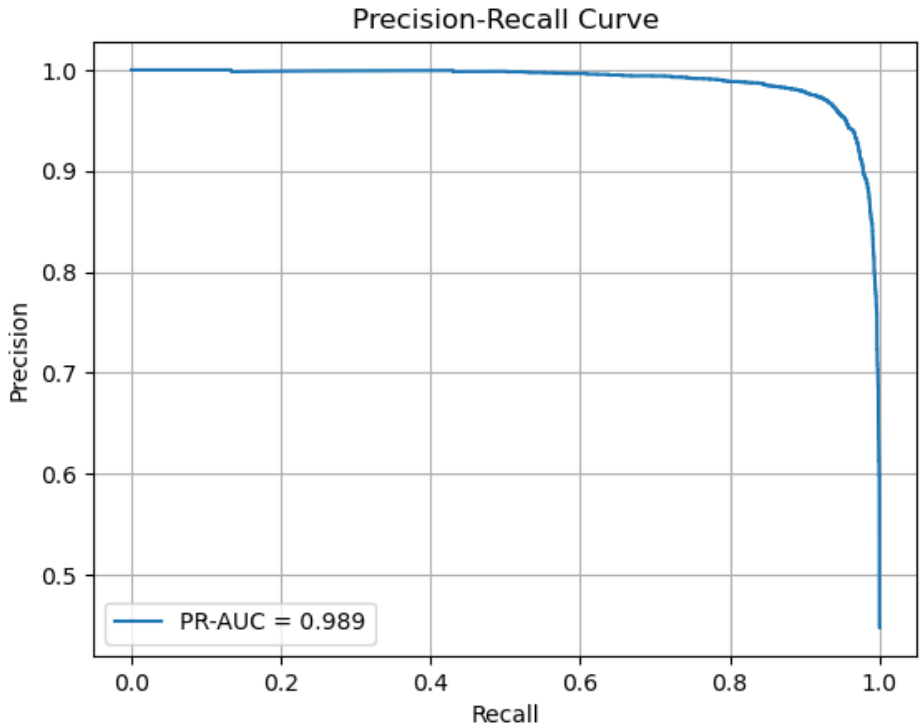


Figure: ROC Curve: Trade-off between true positive rate and false positive rate

AUC (Area Under Curve) closer to 1.0 indicates a strong classifier.

Train-Test Learning Curve (Plot)

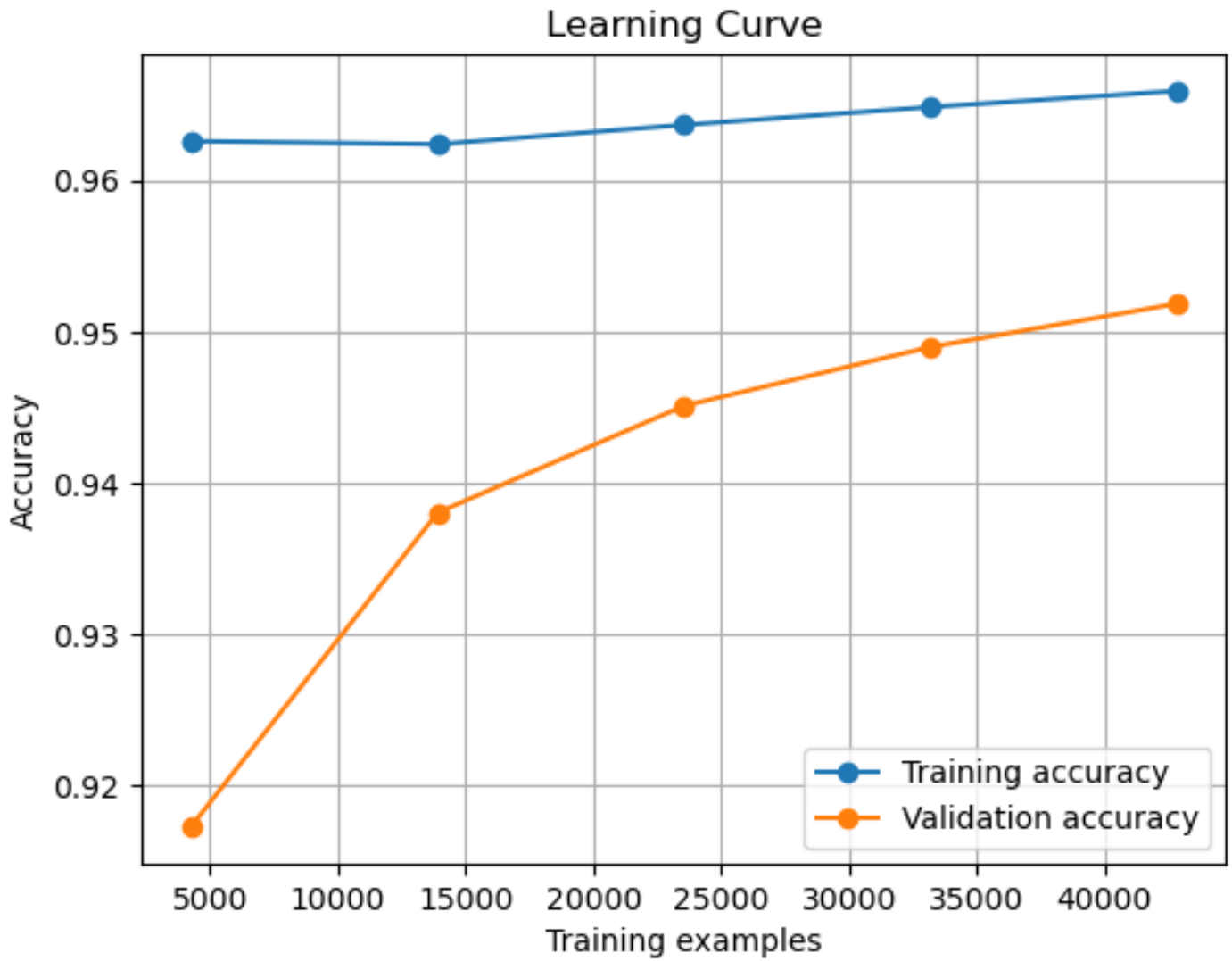


Figure: Training vs Validation Performance over Time

Learning Curve (Interpretation)

- **Training Curve:**
  - Reflects how well the model fits the training data.
  - Steep upward trends may indicate overfitting.
- **Validation/Test Curve:**
  - Shows how well the model generalizes to unseen data.
  - A flat or declining curve may indicate underfitting or overfitting.
- **Ideal Behavior:**
  - Both curves should converge to a high score.
  - A narrow gap between them suggests strong generalization.

PCA Projection of TF-IDF Features

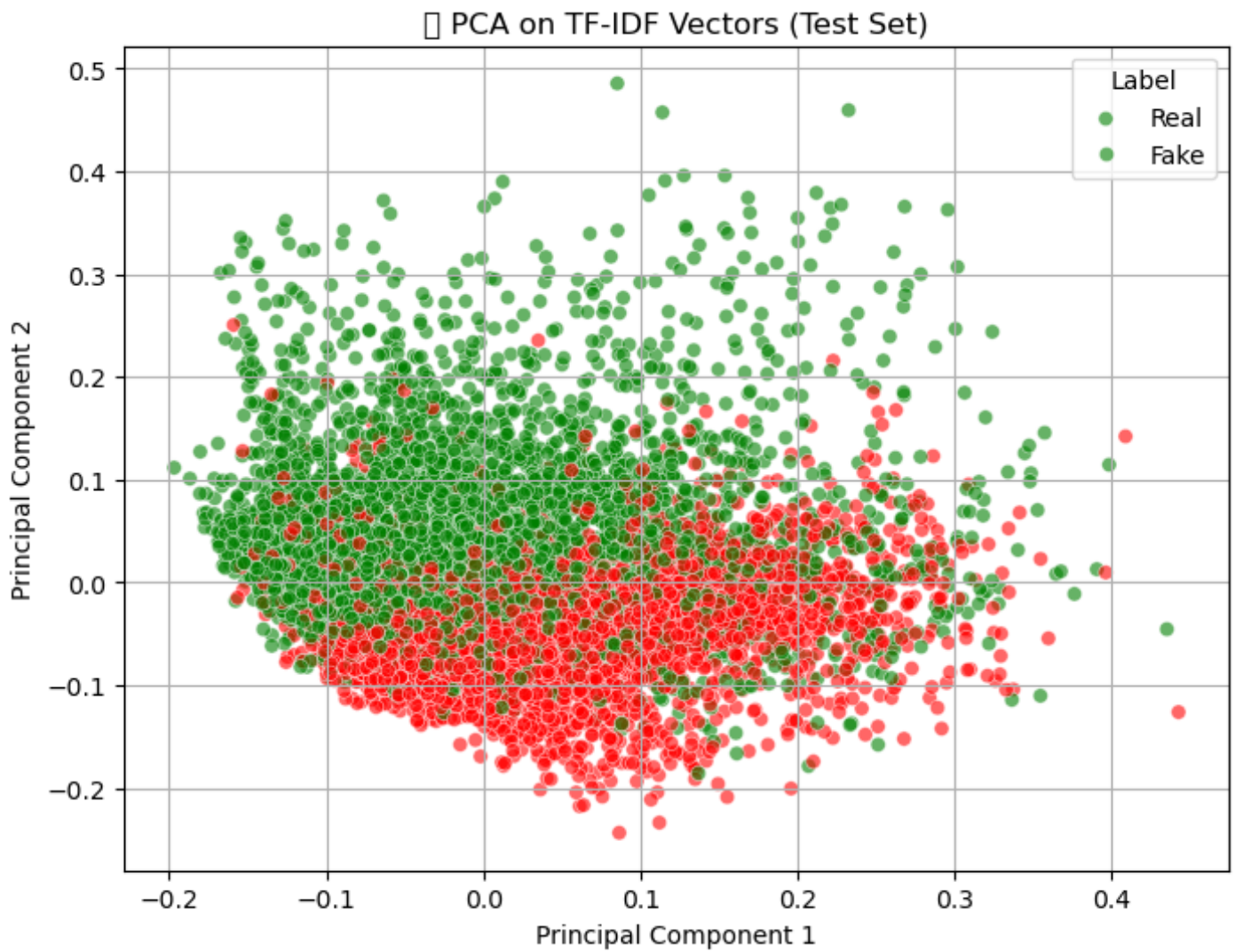


Figure: 2D PCA projection of TF-IDF feature vectors colored by class

PCA Interpretation

- **Dimensionality Reduction:**
  - PCA compresses high-dimensional TF-IDF features into two dimensions.
- **Cluster Analysis:**
  - Densely grouped points suggest class coherence in feature space.
  - Overlapping regions imply potential classification challenges.
- **Usefulness:**
  - Helpful for visually diagnosing model separation capacity.
  - Assists in understanding why misclassifications might occur.

Error Analysis: What I Did and Why

- **What I did:**
  - Identified samples where the TF-IDF + Logistic Regression model made incorrect predictions.
  - Focused on both:
    - False Positives (real news predicted as fake)
    - False Negatives (fake news predicted as real)
- **Why I did it:**
  - Understand misclassification patterns and failure points.
  - Reveal text traits that confuse the model (e.g., emotional tone, misleading cues).

Error Examples from TF-IDF + Logistic Regression

- **False Positive Example:**
  - *"President announces major breakthrough in COVID-19 vaccine research"*
  - Predicted: Fake — Actual: Real
  - **Why it failed:** The model likely flagged the urgent tone as sensational.
- **False Negative Example:**
  - *"BREAKING: Obama admits secret Muslim past!"*
  - Predicted: Real — Actual: Fake
  - **Why it failed:** Lacks direct fake cues; model missed implicit bias wording.

TF-IDF + Logistic Regression: Final Takeaways

- A strong and interpretable baseline with solid metrics (accuracy: 95.6%).
- Handles fake vs real separation reasonably well using term frequency features.
- Limitations:
  - Lacks contextual understanding of language.
  - Fails to detect nuanced fakeness or sarcasm.
- Motivation for using BERT:
  - To capture deeper semantic and contextual information.
  - To improve generalization on harder-to-detect fake news.

BERT + Classifier: Architecture Overview

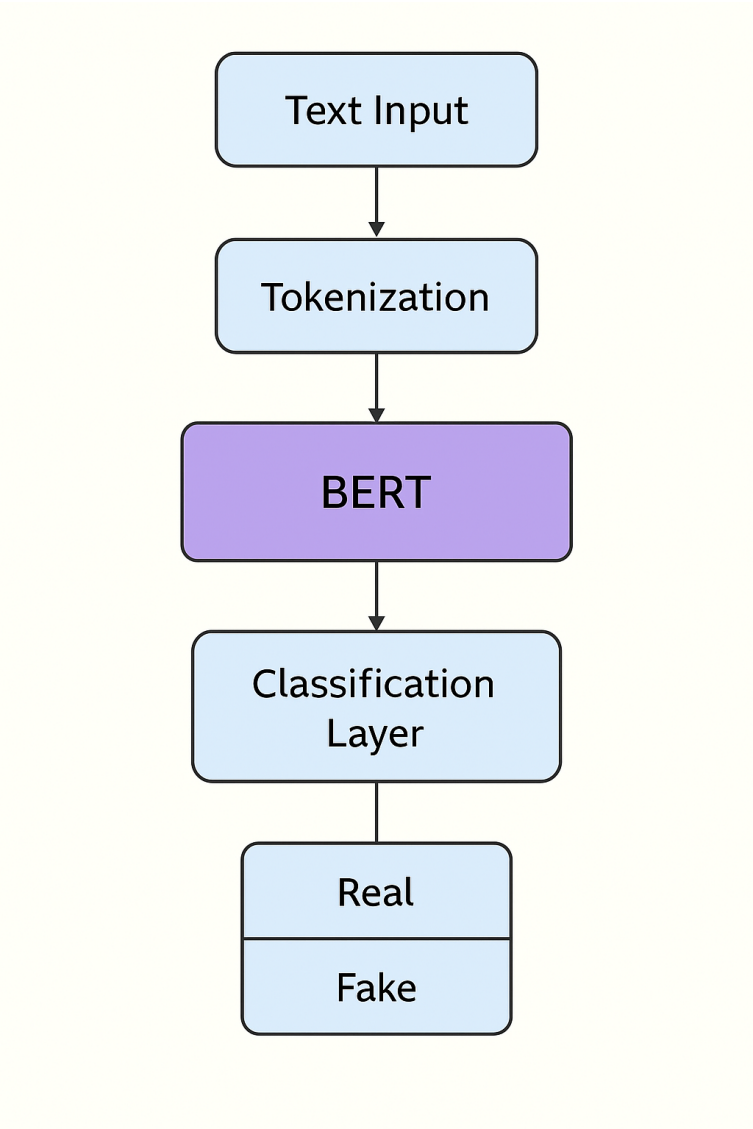


Figure: End-to-End Architecture for Fake News Classification using BERT

BERT Architecture Diagram

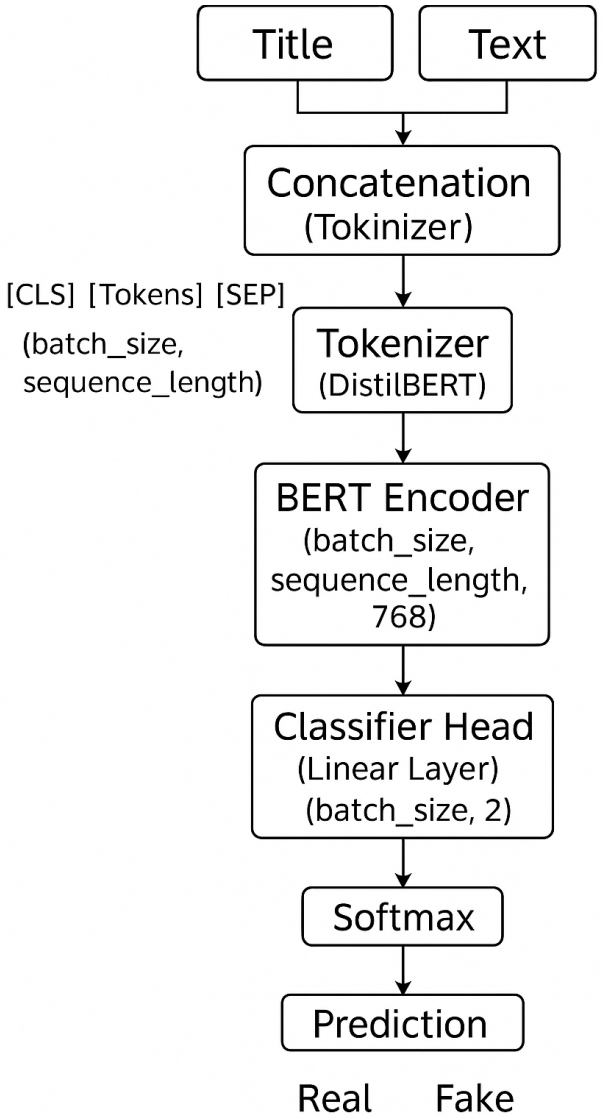


Figure: End-to-end architecture for fake news classification using DistilBERT

Designing the BERT-Based Classifier

- **Model:** `distilbert-base-uncased` — a lighter version of BERT optimized for speed and accuracy.
- **Tokenizer:** HuggingFace's pretrained tokenizer used to handle subwords, padding, truncation.
- **Max Length:** 256 tokens per sample to ensure memory efficiency.
- **Classifier Head:** Linear layer with 2 output nodes (real/fake), softmax-activated.
- **Loss:** CrossEntropyLoss — standard for multi-class classification.
- **Optimizer:** AdamW with `lr=5e-5` for stable convergence.

Training Strategy

- **Epochs:** 3 full passes over the training data.
- **Batch Size:** 8 (CUDA-safe on 6GB VRAM, GTX 1660Ti).
- **Training Loop:**
  - Load batch $\rightarrow$ forward pass through BERT $\rightarrow$ compute loss
  - Backpropagate gradients $\rightarrow$ update weights with optimizer
- **Evaluation:**
  - Performed on a stratified test split (15
  - Metrics include: Accuracy, F1, ROC-AUC, Confusion Matrix, PR Curve
- **Safety:** Manual GPU cleanup between epochs to prevent memory overflow.

BERT Training – Step-by-Step with Vector Dimensions

- **Input:**
  - Tokenized input IDs: shape `(batch_size, max_len)` → `(8, 256)`
  - Attention mask: same shape
- **Forward Pass:**
  - BERT outputs hidden states → shape: `(batch_size, seq_len, hidden_size)` → `(8, 256, 768)`
  - CLS token embedding extracted → shape: `(8, 768)`
- **Classification Head:**
  - Fully connected layer → shape: `(8, 768)` → `(8, 2)`
  - Outputs logits for 2 classes (real, fake)
- **Loss Computation:**
  - CrossEntropyLoss compares logits with true labels → shape `(8,)`.
- **Backpropagation:**
  - Gradients computed and weights updated via AdamW optimizer.

Training Loss Over Epochs

- **Epoch 1:** 6753 batches — Final loss: **0.0817**
- **Epoch 2:** 6753 batches-Final loss: **0.0375**
- **Epoch 3:** 6753 batches- Final loss: **0.0234**

Interpretation:

- The steady decline in training loss across epochs indicates that the model is learning effectively.
- Low final loss suggests the model has captured the patterns in the training data well.
- No signs of instability or overfitting during training — clean convergence.
- Further epochs may bring marginal gains but risk overfitting without validation loss tracking.

BERT Evaluation Metrics Summary

Label	Precision	Recall	F1-score	Support
Real (0)	0.996	0.953	0.974	5216
Fake (1)	0.946	0.995	0.970	4318
Accuracy	0.972 (Overall)			
ROC AUC	0.9987			
Macro Avg	0.971	0.974	0.972	9534
Weighted Avg	0.973	0.972	0.972	9534

Table: Performance metrics for BERT on test set

Precision-Recall Curve (Plot Only)

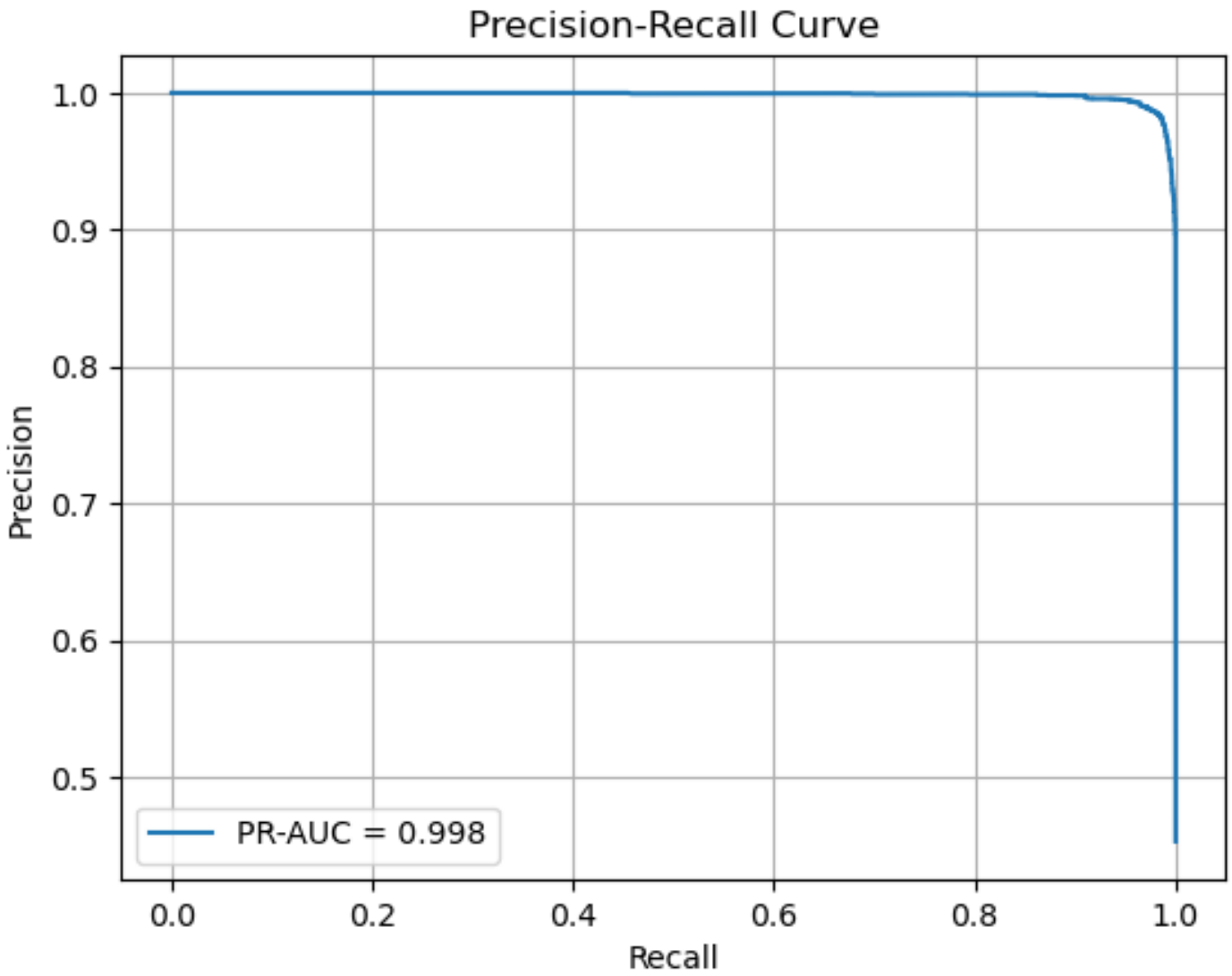


Figure: BERT model PR Curve on the test set

Precision-Recall Curve (Interpretation)

- **AUC 0.998** — extremely high, indicating excellent model performance.
- Precision remains high even when recall increases — very few false positives.
- **Why PR over ROC?**
  - PR Curve is more informative for imbalanced datasets (like fake news).
  - Focuses on the positive (fake) class performance.
- Strong PR curve validates model's utility in high-risk applications.

Confusion Matrix (BERT Output)

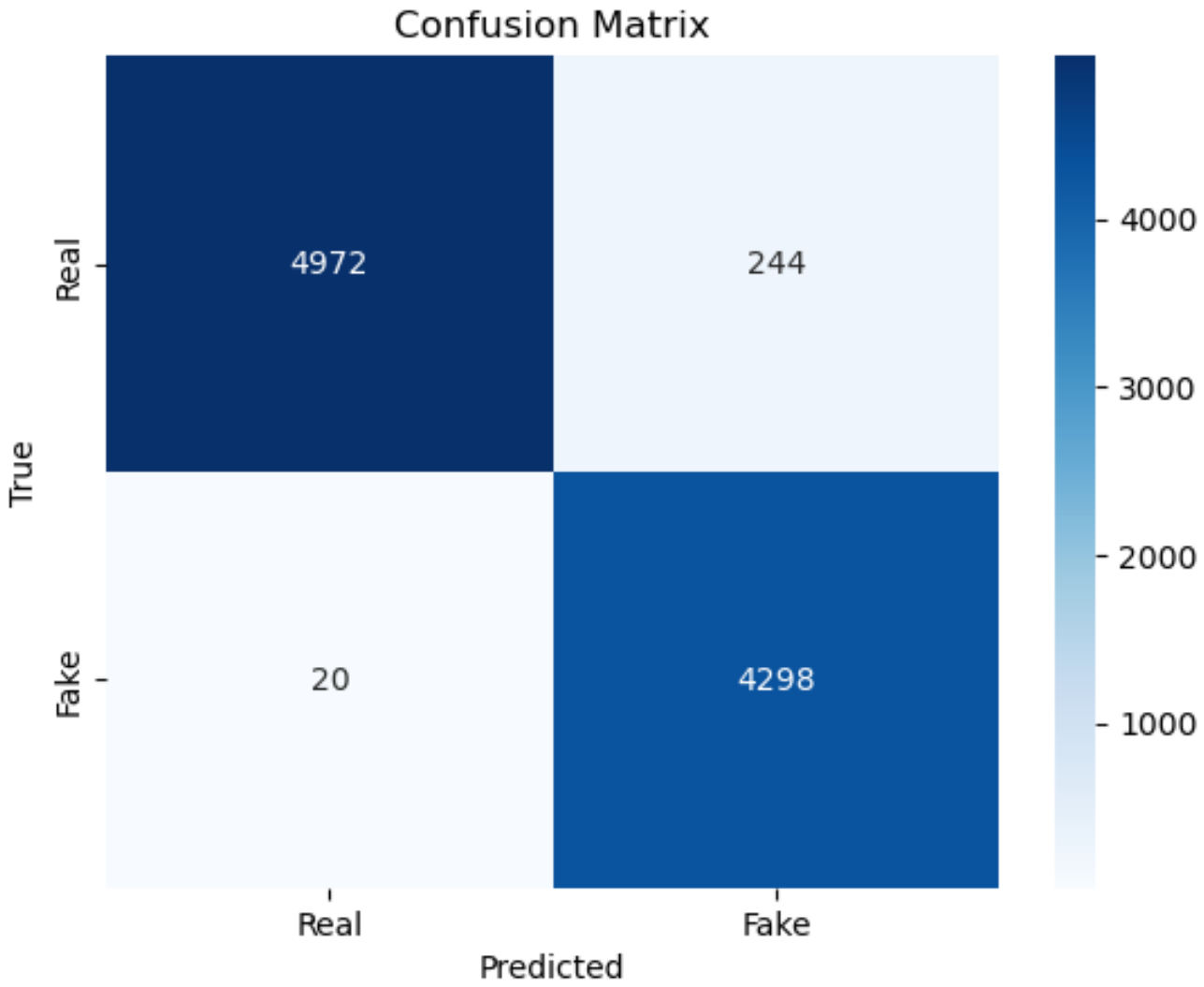


Figure: Confusion Matrix of BERT predictions

Confusion Matrix (Interpretation)

- **True Positives (Fake → Fake):** 4296
- **True Negatives (Real → Real):** 4972
- **False Positives (Real → Fake):** 244
- **False Negatives (Fake → Real):** 22

Insights:

- The model is slightly more cautious — it flags real as fake more often than the reverse.
- Minimal false negatives mean it rarely misses fake news.
- Overall structure shows excellent predictive separation.

Correct Predictions (Examples)

Example 1:
"Trump: We will crush fake news once and for all!"
Prediction: Fake — **True Label:** Fake

Example 2:
"NASA confirms Mars rover detected ancient organic matter."
Prediction: Real — **True Label:** Real

Correct Predictions – Reasoning

- **Fake News:**
  - Sensational tone ("crush fake news") and political keywords helped model flag it correctly.
- **Real News:**
  - Scientific phrasing and neutral tone signal factual reporting.
  - BERT likely leveraged domain vocabulary ("NASA", "organic matter").
- **Conclusion:**
  - Contextual embeddings allowed BERT to capture tone, subject matter, and linguistic cues effectively.

Misclassifications (Examples)

Example 1:
"Scientists report miracle cure for cancer discovered in Italy."
Prediction: Real — **True Label:** Fake

Example 2:
"Obama gives unexpected address on economic reforms."
Prediction: Fake — **True Label:** Real

Misclassifications – Reasoning

- **False Negative:**
  - The phrase “miracle cure” mimics legitimate scientific breakthroughs.
  - BERT missed the exaggeration possibly due to neutral scientific terms.
- **False Positive:**
  - Political phrases like ”unexpected address” may have triggered associations with fake headlines.
  - Lack of context or phrasing similarity to sensational articles could’ve misled the model.
- **Insight:**
  - Even BERT can struggle with subtle satire, sarcasm, or misleading headlines styled like real news.

Model Comparison: TF-IDF + LR vs BERT

Metric	**TF-IDF + LR**	**BERT**
Accuracy	0.956	0.972
Precision	0.954	0.996 (Real), 0.946 (Fake)
Recall	0.958	0.953 (Real), 0.995 (Fake)
F1-score	0.956	0.970+
ROC AUC	0.981	0.9987
Strength	Fast, interpretable	Context-aware, powerful
Limitation	No context info	Slower, resource-heavy

Table: Performance and trade-offs between models

Model Comparison: TF-IDF + LR vs BERT

Metric	**TF-IDF + LR**	**BERT**
Accuracy	0.956	0.972
Precision	0.954	0.996 (Real), 0.946 (Fake)
Recall	0.958	0.953 (Real), 0.995 (Fake)
F1-score	0.956	0.970+
ROC AUC	0.981	0.9987
Strength	Fast, interpretable	Context-aware, powerful
Limitation	No context info	Slower, resource-heavy

Table: Performance and trade-offs between models