Fake News Detection: A Comparative Study of Logistic Regression, LSTM, and BERT

Antonio Leonetti[1]

[1]Department of Computer Science, University of Milan, Italy.

Contributing authors: antonio.leonetti@studenti.unimi.it;

Abstract

The rapid proliferation of online news sources and social media platforms has dramatically changed how information is disseminated and consumed. While these digital channels enable timely sharing of news, they also facilitate the spread of misinformation and *fake news*. Fake news can influence public opinion, manipulate elections, and erode trust in legitimate journalism. In this work, we build and evaluate three classifiers: Logistic Regression, Bidirectional LSTM networks, and fine-tuned BERT. We compare their performance on a balanced fake vs. real news dataset, analyze error modes, and discuss computational trade-offs.

Keywords: Fake News Detection, Text Classification, Logistic Regression, LSTM, BERT, Exploratory Data Analysis

1 Introduction

Automatic fake news detection remains a critical challenge for natural language processing and machine learning (5). Deceptive articles often mimic legitimate reporting yet embed subtle cues, such as exaggerated language or misinformation tactics, and require models that capture lexical and contextual signals. We compare:

1. **Logistic Regression:** A linear model combining TF–IDF, LDA topics, and numeric text features.
2. **Bidirectional LSTM:** A recurrent architecture that models sequential dependencies in token sequences.
3. **BERT:** A pre-trained transformer fine-tuned to leverage deep contextual embeddings.

Existing hybrid architectures (e.g., CSI) combine textual, user, and network signals to detect deception (4), but they will not be part of this study. We conduct experiments on the WELFake dataset, containing 72,134 labeled news articles. Subsequent sections detail our dataset exploration (Section 2), model implementations and results (Sections 3-4–5), and conclusions (Section 6).

2 Dataset and Exploratory Data Analysis

2.1 Dataset Cleaning and Split

The WELFake dataset contains 72,134 news articles. Each record includes:

- **ID:** Unique identifier.
- **Title:** Headline of the article.
- **Text:** Body text of the article (variable length, sometimes incomplete).
- **Label:** Binary veracity indicator (0 = false, 1 = real).

After removing ≈9,000 records with missing title/text or duplicates, $N_{\text{clean}} =$ 63,121 articles remained, preserving the original class proportion (55.12% Real, 44.88% Fake). We concatenated each article's `title` and `text` into `raw_text` and then randomly divided into Train (70%, 44,185), Validation (15%, 9,468) and Test (15%, 9,468), maintaining the balance of the class in each subset.

2.2 Text Preprocessing

We generated three text variants for feature extraction. First, `raw_text` concatenates the article's `title` and `text`. Next, to obtain `clean_text` we applied:

1. **Lowercasing & cleaning:** Remove punctuation, URLs, HTML.
2. **Tokenization:** NLTK's `word_tokenize`.
3. **Filtering:** Remove non-alphabetic tokens and stopwords (NLTK list).

Finally, we performed lemmatization to produce `lemmatized_text`.

2.3 EDA Summary

We followed standard NLP exploratory-data-analysis techniques (1) to compute length, sentiment, and readability features.

Feature	Mean	Median	IQR	Range
Word Count	558.64	415	[253, 689]	[2, 24 243]
Character Count	3 379.25	2 545	[1 538, 4 193]	[14, 143 035]
Average Word Length	6.12	6.08	[5.89, 6.28]	[3.53, 149.50]
Polarity (VADER)	0.06198	0.06133	[0.00750, 0.11267]	[–1.00, 1.00]
Subjectivity (TextBlob)	0.41005	0.41510	[0.35043, 0.47632]	[0.00, 1.00]
Flesch Reading Ease	46.65	47.22	[39.26, 54.73]	[–2 079.40, 119.19]
SMOG Index	13.15	13.60	[12.20, 14.90]	[0.00, 38.70]

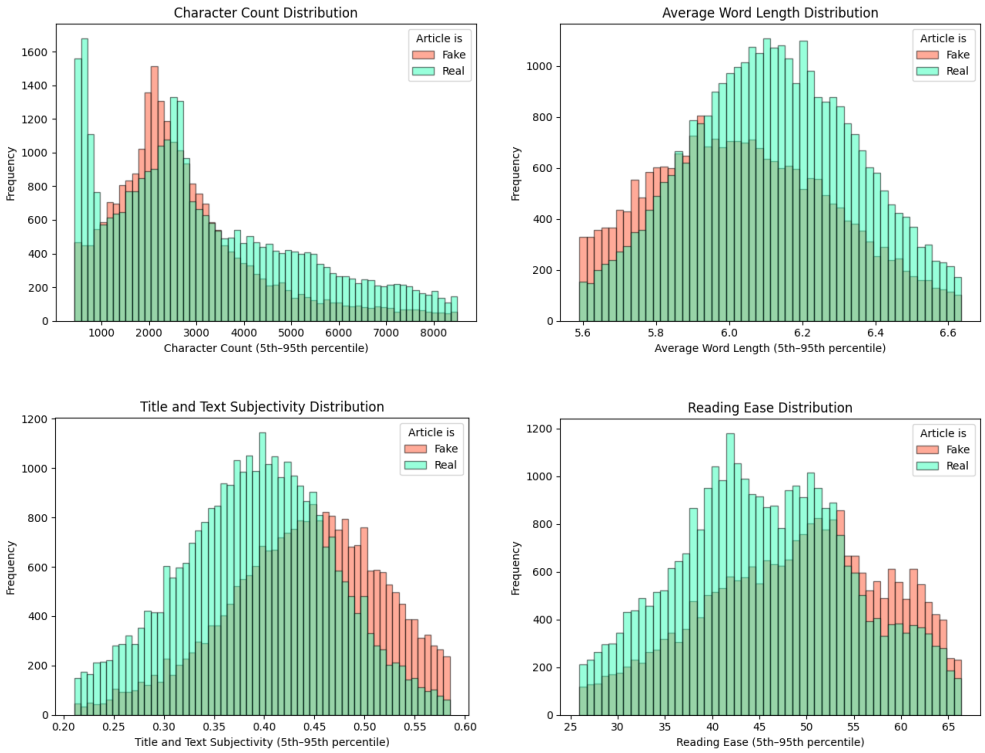


Fig. 1: Key EDA visualizations on $N = 63{,}121$.

These features show overlapping distributions between Real and Fake, with Fake articles slightly shorter and harder to read, while more polarized in sentiment. Figure 1 presents sample histograms.

3 Logistic Regression

3.1 Overview

Logistic Regression (LR) provides a baseline by modeling the log odds of *Fake* versus *Real* directly as a linear function of input features.

3.2 Features

Numeric Features: We extracted the seven text-level statistics we saw in the EDA

- `word_count`, `char_count`, `avg_word_length`
- `polarity`, `subjectivity` (TextBlob)
- `reading_ease` (Flesch), `smog_index`

After computing these on all 63 121 documents, we standard-scaled each feature (zero mean, unit variance) based on the Train+Validation split.

TF–IDF: Fitting on the 44 185 training documents produced a vocabulary of 5 000 n-grams. The resulting TF–IDF matrices have shape $(44\,185 \times 5\,000)$ for Train, $(9\,468 \times 5\,000)$ for Validation, and $(9\,468 \times 5\,000)$ for Test.
LDA Topics: We trained an LDA model with $K = 24$ topics on the bag-of-words corpus derived from the 44 185 training documents.

3.3 Model Formulation and Training

We used `scikit-learn`'s `LogisticRegression` with:

- `solver="saga"`, `penalty="l2"`, `class_weight="balanced"` to address the slight class imbalance.
- `max_iter=5000`, `tol=1e-5` to ensure convergence on high-dimensional data.
- Regularization parameter C (inverse of λ) selected via grid search over $\{0.01, 0.1, 1.0, 10.0\}$ using ROC-AUC on the 44 185/9 468 Train/Val split.

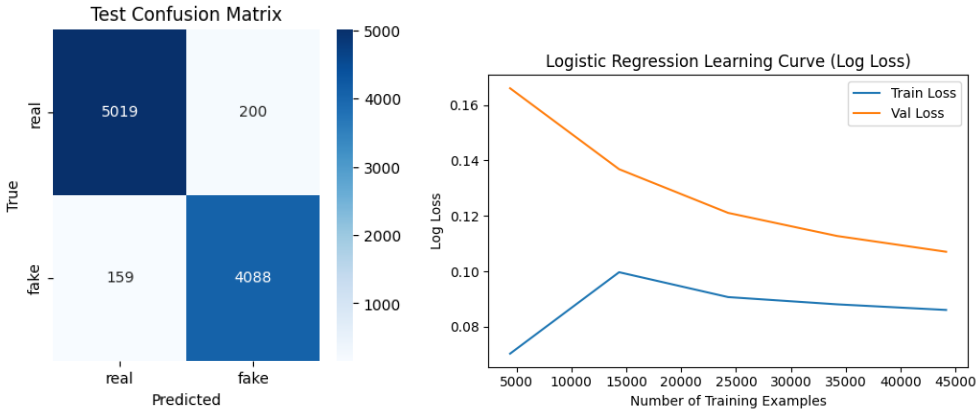
With $C = 10.0$, we retrained on the combined Train+Validation set ($N = 53\,653$) and evaluated on the held-out Test ($N = 9\,468$).

3.4 Results and Discussion

On the Test set, the final LR classifier achieved:

- **Accuracy:** 96.0%
- **Real class:** $P = 0.97,\ R = 0.96,\ F_1 = 0.97$ (support = 5 219)
- **Fake class:** $P = 0.95,\ R = 0.96,\ F_1 = 0.96$ (support = 4 249)

LR effectively leverages lexical, topical, and numeric signals, achieving 96% accuracy. Its interpretability (feature weights) and low training cost (≈5 min CPU) make it a strong baseline, though it lacks context modeling.



4 Bidirectional LSTM

4.1 Overview

RNN, particularly Long Short-Term Memory (LSTM) units, can capture sequential dependencies and compositional semantics not accessible to bag-of-words models.

4.2 Data Preparation and Embeddings

Using Keras's `Tokenizer` with `num_words=5000`, we fit on the 44 170 training documents (`train.csv` after cleaning), building a vocabulary of the top 5 000 most frequent tokens (post-stopword removal and lemmatization). Each document's concatenated was converted to a sequence truncate/pad to $L_{\max} = 200$.

4.3 Training Procedure

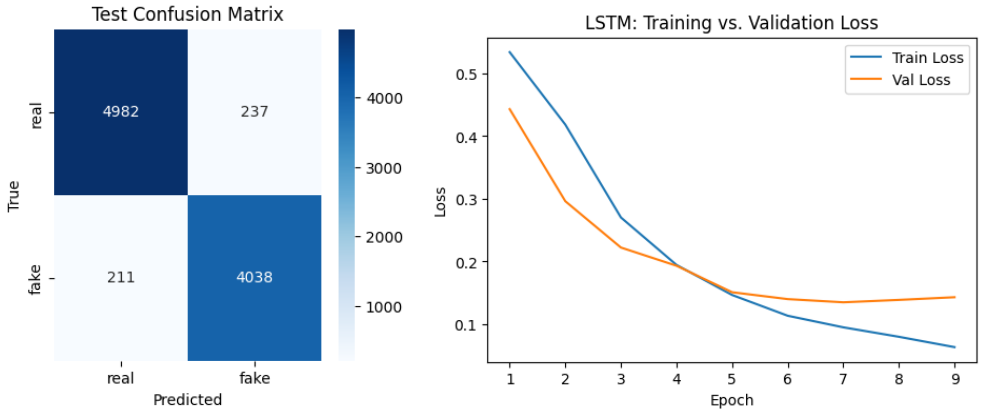
The training was performed using binary cross-entropy loss optimized with Adam. We used batches of 64 examples and trained for up to 10 epochs, employing early stopping on validation loss with patience of two epochs.

4.4 Results

On the held-out Test set ($N_{\text{test}} = 9\,468$), the final BiLSTM achieved:

- **Accuracy:** 95.0%
- **Real class:** $P = 0.96,\ R = 0.95,\ F_1 = 0.96$ (support $= 5\ 219$)
- **Fake class:** $P = 0.94,\ R = 0.95,\ F_1 = 0.95$ (support $= 4\ 249$)

BiLSTM captures sequential context—negation and multi-word patterns—yielding robust performance (95%). Training is moderate (≈25 min GPU), and inference is ≈8 ms/article.



5 BERT Fine-Tuning

5.1 Overview

BERT (Bidirectional Encoder Representations from Transformers) learns deep contextual embeddings by pretraining on massive corpora, and then fine-tuning on downstream tasks. We fine-tuned `bert-base-uncased` on the fake news classification task, adapting the final pooled [CLS] embedding to a binary classification head.

5.2 Fine-Tuning Procedure

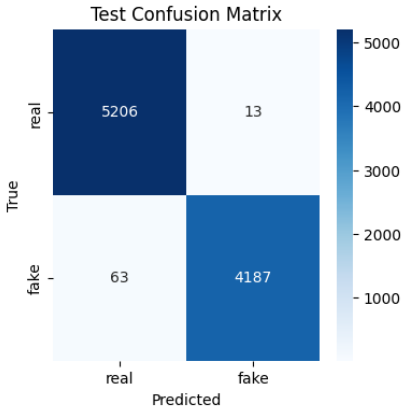
For BERT-specific preprocessing steps, we adapted methods from (2). BERT was fine-tuned using the AdamW optimizer (learning rate 5×10^{-5}) with a linear warm-up over the first 10% of steps. We enabled automatic mixed-precision to speed up training, processed examples in batches of 16, and trained for up to three epochs—stopping early if validation loss did not improve. The training was run on an NVIDIA RTX 4070 SUPER GPU, taking about 6 minutes per epoch.

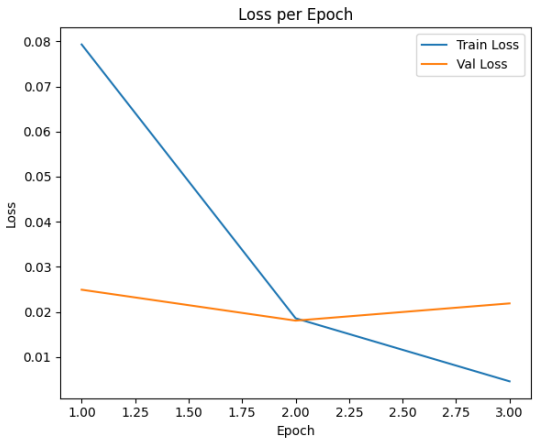
5.3 Results and Discussion

On the Test set ($N_{\text{test}} = 9\,469$), fine-tuned BERT delivered:

- **Accuracy:** 99.20%
- **Real class:** $P = 0.9969,\ R = 0.9852,\ F_1 = 0.9910$ (support = 5 219)
- **Fake class:** $P = 0.9945,\ R = 0.9906,\ F_1 = 0.9925$ (support = 4 250)

BERT's contextual embeddings yield 99.2% accuracy, excelling at subtle semantic distinctions. Training cost (≈18 min) and inference latency (≈15 ms/article) are higher.





6 Conclusion

6.1 Summary of Findings

- **Logistic Regression (LR):** Combining TF–IDF, LDA topics, and numeric text features yielded 96.0 % accuracy (macro-F_1 = 0.96) on WELFake. Training and validation losses remained closely aligned, indicating minimal overfitting.
- **Bidirectional LSTM (BiLSTM):** Modeling 200-token sequences with 100-dimensional embeddings produced 95.0 % accuracy (macro-F_1 = 0.95). Validation loss began to plateau after epoch 5 while training loss continued to decrease, signaling mild overfitting that was mitigated by early stopping.
- **BERT Fine-Tuning:** Leveraging deep contextual embeddings achieved 99.2 % accuracy (macro-F_1 = 0.9918). Training/validation loss curves declined in tandem, showing that careful regularization prevented overfitting despite BERT's high capacity.

6.2 Overfitting Concerns and External Validation

Although all three models performed exceptionally on WELFake, overfitting remains a concern, particularly for neural architectures:

- **LR:** The near-overlap of train/validation losses (Figure) confirms that the regularized linear model generalizes well.
- **BiLSTM:** A widening gap between training and validation loss after the fifth epoch (Figure) indicates that the model began to memorize training details; early stopping curtailed further overfitting.
- **BERT:** Loss curves (Figure) decrease in parallel, suggesting robust generalization under our hyperparameters and dataset size.

To assess generalizability, we evaluated all three models on the ISOT Fake News Dataset(3). Each maintained strong performance—with only modest accuracy declines relative to WELFake—demonstrating that our models capture broadly applicable signals rather than dataset-specific artifacts.

6.3 Future Directions

- **Domain Adaptation:** Fine-tune or pre-train models on specialized corpora (e.g., medical, political) to improve detection in niche contexts.
- **Explainability:** Incorporate model-agnostic explanation methods (e.g., SHAP, attention visualization) to reveal decision rationales and uncover biases in BiLSTM and BERT.
- **Boosting:** Explore ensemble techniques—such as combining LR's interpretable features with BERT's deep embeddings or using gradient boosting on learned representations—to further improve robustness and accuracy.

All code, experiments, and additional materials are available on GitHub: github.com/Jaegon99/fake-news-detection.

References

[1] Neptune.ai. (2021). Exploratory Data Analysis for Natural Language Processing: Tools and Techniques. Retrieved from https://neptune.ai/blog/exploratory-data-analysis-natural-language-processing-tools.

[2] mexwell. (2021). BERT for Binary Classification. Kaggle Notebook. Retrieved from https://www.kaggle.com/code/mexwell/bert-for-binary-classification/notebook.

[3] emineyetm. (2021). Fake News Detection Datasets. Kaggle Dataset. Retrieved from https://www.kaggle.com/datasets/emineyetm/fake-news-detection-datasets.

[4] Ruchansky, N., Seo, S., & Liu, Y. (2017). CSI: A Hybrid Deep Model for Fake News Detection. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 797–806.

[5] Zhou, X., & Zafarani, R. (2020). A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. *ACM Computing Surveys*, 53(5), 1–40.