

UCLA

UCLA Electronic Theses and Dissertations

Title

A Comparative Study of Machine Learning Techniques for Fake News Detection

Permalink

<https://escholarship.org/uc/item/6rs5z9qh>

ISBN

9798190929560

Author

Shih, Cindy

Publication Date

2026-08-31

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

A Comparative Study of Machine Learning Techniques
for Fake News Detection

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Applied Statistics and Data Science

by

Cindy Shih

2026

© Copyright by

Cindy Shih

2026

ABSTRACT OF THE THESIS

A Comparative Study of Machine Learning Techniques for Fake News Detection

by

Cindy Shih

Master of Applied Statistics and Data Science

University of California, Los Angeles, 2026

Professor Yingnian Wu, Chair

This thesis examines fake news detection as a supervised text classification problem and compares several machine learning techniques using the WELFake dataset. The analysis includes both classical machine learning models and recurrent neural network models in order to evaluate their relative strengths for distinguishing fake news from real news using article text. The models considered in this study are Logistic Regression, Linear SVM, vanilla RNN, LSTM, GRU, and BiLSTM. After preprocessing, all models were trained and evaluated using the same training, validation, and test split to ensure a fair comparison. Performance was assessed using accuracy, precision, recall, F1-score, and training time. The results show that BiLSTM achieved the strongest overall performance, while GRU and Linear SVM also performed very well. At the same time, the classical baseline models remained

highly competitive while requiring substantially less training time. These findings highlight the tradeoff between predictive performance, computational efficiency, and model complexity, and show that strong baseline methods remain effective for text-based fake news detection.

The thesis of Cindy Shih is approved.

Nicolas Christou

George Michailidis

Yingnian Wu, Committee Chair

University of California, Los Angeles

2026

Contents

1	Introduction	1
1.1	Background and Motivation	1
1.2	Research Questions	3
2	Data Description	4
2.1	Dataset Source	4
2.2	Dataset Information	4
2.3	Data Preprocessing	6
2.4	Text Representation	7
3	Exploratory Data Analysis	11
3.1	Class Distribution	11
3.2	Text Length by Class	12
3.2.1	Title Length by Class	12
3.2.2	Content Length by Class	13
3.3	Common Word Patterns	14
3.4	EDA Summary	16
4	Modeling Approaches	17
4.1	Model Comparison Framework	17
4.2	Classical Machine Learning Models	18
4.2.1	Logistic Regression	18
4.2.2	Linear SVM	18

4.3	Recurrent Neural Network Models	19
4.3.1	Shared Neural Architecture	19
4.3.2	Vanilla RNN	20
4.3.3	LSTM	21
4.3.4	GRU	22
4.3.5	BiLSTM	22
4.4	Training and Evaluation	23
5	Results and Discussion	25
5.1	Performance Comparison of Models	25
5.2	Discussion of Results	26
5.3	Limitations	28
6	Conclusion	29
6.1	Concluding Remarks	29
6.2	Future Work	30

List of Figures

3.1	Class Distribution of Fake and Real News	12
3.2	Distribution of Title Word Count by Class	13
3.3	Distribution of Content Word Count by Class	14
3.4	Top Words in Fake News	15
3.5	Top Words in Real News	16

List of Tables

2.1	Data Description	5
5.1	Performance Comparison of All Models	26

Chapter 1

Introduction

1.1 Background and Motivation

Fake news has become a serious issue in the modern digital information environment because false or misleading stories can spread rapidly through online platforms while still appearing to be legitimate journalism. This issue is important not only because it can mislead the public, but also because it can influence political decision-making, weaken trust in institutions, and create financial incentives for producing sensational but inaccurate content (Lazer et al., 2018; Allcott and Gentzkow, 2017). During the 2016 United States presidential election, for example, false stories favoring Donald Trump were shared on Facebook approximately 30 million times, compared with about 8 million shares for false stories favoring Hillary Clinton, illustrating the large scale at which misinformation can circulate in political settings (Allcott and Gentzkow, 2017). Research has also shown that false news spreads significantly farther, faster, deeper, and more broadly than true news online, which makes misinformation especially difficult to contain once it begins to diffuse through social networks (Vosoughi et al., 2018).

At the same time, distinguishing real news from fake news is often challenging. False

stories are frequently designed to imitate the style and appearance of legitimate reporting, which means that readers may not be able to identify them based on surface features alone. In online environments, individuals also do not always apply the critical thinking skills needed to carefully evaluate the credibility of the information they encounter, which further increases the difficulty of the problem (Machete and Turpin, 2020; Lazer et al., 2018). Recent advances in artificial intelligence have made this issue even more urgent. Generative AI systems are now capable of producing fluent and convincing text at scale, making it easier to generate misleading content quickly and at low cost (Cassar, 2023; Campante et al., 2025). Public awareness of this issue increased substantially after the release of ChatGPT on November 30, 2022, which helped bring large language model technology into broad public use (OpenAI, 2022). Consequently, fake news is no longer limited to traditionally produced misinformation, but is increasingly linked to AI-assisted content generation, creating new challenges for trust in news and for the governance of digital information environments (Raman et al., 2024).

Because of these challenges, automatic fake news detection has become an important application of machine learning. However, fake news detection is not a simple keyword-matching problem. Fake and real news often discuss the same public figures, institutions, and political events, and they may share overlapping vocabulary. As a result, successful detection requires methods that can learn more meaningful patterns from text, including differences in wording, framing, and structure, rather than relying only on a small set of obvious terms (Lazer et al., 2018; Aïmeur et al., 2023).

This study focuses specifically on text-based fake news detection. More precisely, it investigates whether machine learning models can distinguish fake news articles from real news articles using the textual content of headlines and article bodies. In this study, fake news detection is treated as a supervised text classification problem, in which each article is represented by its text and assigned a binary label indicating whether it is fake or real. This narrower focus allows the classification task to be examined in a clear and controlled setting

while still addressing an important real-world problem.

The purpose of this study is to compare several machine learning techniques for fake news detection, including both classical machine learning models and recurrent neural network models. The classical models serve as strong and interpretable baseline methods, while the recurrent models make it possible to examine whether sequential neural architectures can better capture meaningful textual patterns in news articles. In this sense, the study is not only concerned with identifying the single best-performing model. It also seeks to examine the tradeoff between predictive performance, training efficiency, and model complexity. This is important because a more complex model may produce only a modest improvement in performance, and that improvement may not always justify the additional computational cost and reduced interpretability.

Overall, this study aims to evaluate whether more advanced neural models provide meaningful improvement over strong baseline methods and to better understand which types of models are most effective for text-based fake news detection.

1.2 Research Questions

This study is guided by the following research questions:

1. How do different machine learning techniques compare in performance for fake news detection?
2. Do recurrent neural network models provide meaningful improvement over strong classical baseline models for distinguishing fake news from real news?
3. Which model offers the best balance between predictive performance, training efficiency, and model complexity?

Chapter 2

Data Description

2.1 Dataset Source

The dataset used in this study is the WELFake dataset, which was introduced by Verma, Agrawal, Amorim, and Prodan in their paper *WELFake: Word Embedding Over Linguistic Features for Fake News Detection*. In that study, the authors proposed WELFake as a larger and more general dataset for fake news detection in order to support machine learning research on text-based misinformation classification. The dataset was constructed by combining multiple existing news sources so that the resulting corpus would be more diverse and less likely to favor a narrowly biased classification outcome (Verma et al., 2021).

2.2 Dataset Information

The WELFake dataset contains 72,134 news articles, including 35,028 real news articles and 37,106 fake news articles. It was created by merging four existing datasets: Kaggle, McIntire, Reuters, and BuzzFeed Political. The purpose of combining these sources was to produce a larger and more general benchmark dataset for fake news detection using textual information

(Verma et al., 2021).

The raw dataset includes four main columns: a serial number column, a title column, a text column, and a label column. The title column contains the news headline, the text column contains the article body, and the label column indicates whether the article is fake or real. In this study, the serial number column was not used because it did not provide meaningful predictive information for classification. Instead, the analysis focused on the title, text, and label fields. During preprocessing, a new variable called `content` was created by combining the title and text fields into a single text input.

After preprocessing and cleaning, the final dataset used in this study contained 63,676 observations. Among these, 34,790 articles were labeled as fake and 28,886 articles were labeled as real. Although the cleaned dataset is somewhat imbalanced, the two classes remain reasonably comparable in size, making the dataset suitable for supervised text classification. Table 2.1 provides an overview of the variables used in this study.

Table 2.1: Data Description

Variable Name	Description
<code>title</code>	Headline of the news article
<code>text</code>	Body text of the news article
<code>content</code>	Combined text field formed by concatenating title and text during preprocessing
<code>label</code>	Target variable indicating whether the article is fake or real (0 = fake, 1 = real)

2.3 Data Preprocessing

Before model training, several preprocessing steps were applied to improve data quality and prepare the dataset for text classification. The goal of these steps was to reduce noise, standardize the textual input, and ensure that all models were trained and evaluated on the same cleaned corpus.

First, the index-like serial number column was removed because it did not contain meaningful predictive information for fake news detection. The title and text columns were then checked for missing values. Since some observations had missing headlines or missing article bodies, those missing values were replaced with empty strings so that partially complete observations could still be processed consistently.

After handling missing values, the title and text fields were combined into a single text field, denoted as **content**. This step was used to represent the full textual information of each article in one place. Combining the headline and article body allowed the models to use both short-form and longer-form textual cues during classification, rather than relying on only one of the two fields.

Rows with empty combined content were removed from the dataset because they did not contain usable textual information. In addition, duplicate articles were dropped based on the **content** field so that repeated observations would not affect model training or artificially inflate performance during evaluation. This was an important step because duplicated text could otherwise make the classification task easier if nearly identical articles appeared in both the training and test sets.

A basic text-cleaning procedure was then applied to standardize the content field. More specifically, all text was converted to lowercase, URLs were removed, non-letter characters were removed, and repeated whitespace was reduced to a single space. Lowercasing ensured that words with different capitalization patterns were treated consistently. Removing URLs

and non-letter characters reduced noise that was unlikely to contribute meaningful semantic information to the classification task. Standardizing whitespace further improved consistency in the cleaned text representation.

After preprocessing, the cleaned dataset was divided into training, validation, and test sets using stratified sampling in order to preserve class proportions across the three subsets. Stratification was used so that the label distribution remained comparable across all subsets, making model evaluation more reliable. The final split used in this study consisted of 44,573 observations in the training set, 9,551 observations in the validation set, and 9,552 observations in the test set.

Using a consistent preprocessing pipeline for all models was important for fairness in comparison. Although the classical machine learning models and recurrent neural network models used different text representations, they were trained on the same cleaned dataset and evaluated using the same train, validation, and test split. This reduces unnecessary variation and allows the comparison to focus more directly on differences in modeling approach rather than differences in data preparation.

2.4 Text Representation

Two different text representation approaches were used in this study because the classical machine learning models and the recurrent neural network models require different input formats. However, in order to make the comparison between models more consistent, both approaches were designed using the same maximum vocabulary size of 20,000 terms.

For the classical machine learning models, the cleaned article content was transformed using Term Frequency–Inverse Document Frequency (TF–IDF). TF–IDF is a numerical representation of text that measures how important a word is to a document relative to the entire corpus. The main idea is that a word should receive a higher weight if it appears fre-

quently in one document but not too frequently across all documents. In this way, TF-IDF emphasizes terms that are more informative for distinguishing documents while reducing the influence of very common words that appear in many articles.

The TF-IDF weight of term t in document d is defined as

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t),$$

where the term frequency is

$$\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}},$$

and the inverse document frequency is

$$\text{IDF}(t) = \log \left(\frac{N}{df_t} \right).$$

In these expressions, $f_{t,d}$ denotes the number of times term t appears in document d , N denotes the total number of documents in the corpus, and df_t denotes the number of documents that contain term t . As a result, terms that occur often in a particular article but rarely across the full dataset receive larger TF-IDF weights. In practice, the implementation used in this study follows the TF-IDF transformation provided by the scikit-learn library.

In this study, the TF-IDF vectorizer was limited to a maximum of 20,000 features, with English stop-word removal and both unigram and bigram features included. Using unigrams and bigrams allowed the models to capture not only individual words but also short word combinations that may carry useful contextual information. This TF-IDF representation was used as the input for the Logistic Regression and Linear Support Vector Machine models.

For the recurrent neural network models, the cleaned article content was converted into sequences of integer tokens using a tokenizer fitted on the training data. Tokenization is the process of mapping each word in the corpus to a unique integer index, which allows the text

to be represented numerically and used as input for neural network models. In this study, the tokenizer vocabulary was limited to the 20,000 most frequent words. This restriction helped reduce computational cost while retaining the most informative and frequently occurring terms in the corpus. An out-of-vocabulary token was also used so that words not included in the vocabulary could still be represented consistently during model training and evaluation.

After tokenization, each article was transformed into a sequence of integer indices corresponding to the words that appeared in the text. However, because news articles vary substantially in length, the resulting sequences did not all have the same size. Since neural networks require fixed-length inputs within each batch, the tokenized sequences were padded or truncated to a common maximum length of 300 tokens. Padding was applied to shorter sequences by adding zeros to the end of the sequence, while longer sequences were truncated beyond the 300-token limit. This step made it possible to train the recurrent neural network models efficiently while keeping the input format consistent across all observations.

The sequence representation was then passed through an embedding layer. The embedding layer transforms each token index into a dense vector representation in a lower-dimensional continuous space. Unlike TF-IDF, which represents text as sparse weighted counts, embeddings allow semantically or contextually related words to have similar numerical representations. In this study, the embedding dimension was set to 128, meaning that each token was mapped to a 128-dimensional vector before being processed by the recurrent layer.

This sequence-based representation was used for the vanilla RNN, LSTM, GRU, and BiLSTM models. Using the same tokenizer vocabulary size and the same maximum sequence length across all neural models helped ensure that the comparison focused on differences in recurrent architecture rather than differences in text input preparation. In addition, using the same maximum vocabulary size of 20,000 for both TF-IDF and tokenized sequences made the comparison between classical machine learning models and recurrent neural net-

work models more interpretable. Although the two representations are still fundamentally different, keeping the vocabulary size consistent reduced one source of variation and allowed the comparison to focus more directly on differences in model structure and learning behavior.

Chapter 3

Exploratory Data Analysis

3.1 Class Distribution

Figure 3.1 shows the class distribution of the cleaned dataset. After preprocessing, fake news articles account for 54.64% of the dataset, while real news articles account for 45.36%. This indicates a moderate class imbalance, with fake news appearing somewhat more frequently than real news. However, the difference is not severe, so the two classes remain reasonably comparable in size for supervised classification.

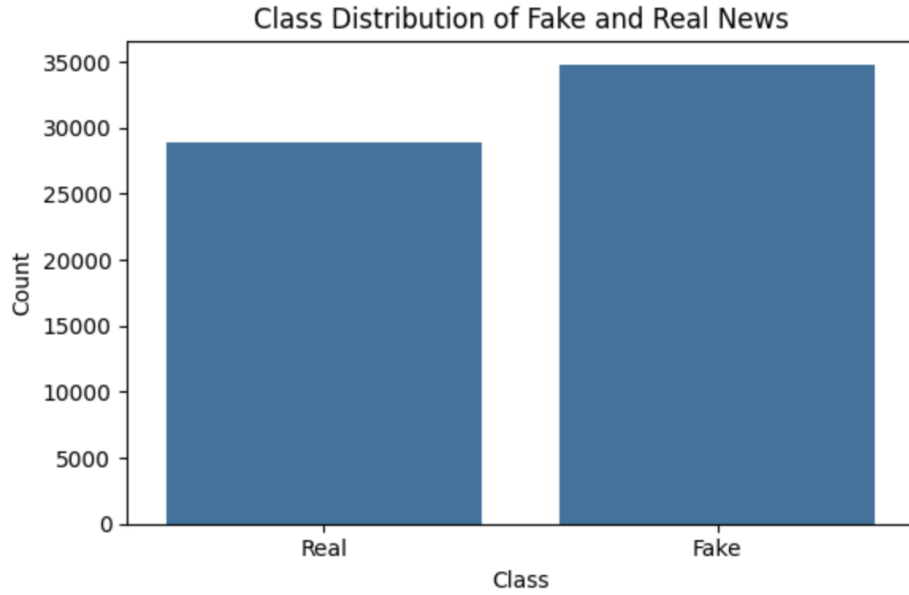


Figure 3.1: Class Distribution of Fake and Real News

3.2 Text Length by Class

To better understand the structure of the text data, word counts were calculated for both article titles and the combined article content. Examining these length distributions helps show whether fake and real news differ in simple structural features before applying more advanced machine learning models.

3.2.1 Title Length by Class

Figure 3.2 shows the distribution of title word count by class. Compared with fake news titles, real news titles appear to be slightly longer on average and show a somewhat wider spread. In contrast, fake news titles are more concentrated around shorter lengths, with the highest density occurring around 9 to 11 words. Although the two classes still overlap substantially, this pattern suggests that title length differs somewhat between fake and real

news and may provide a small amount of additional discriminatory information.

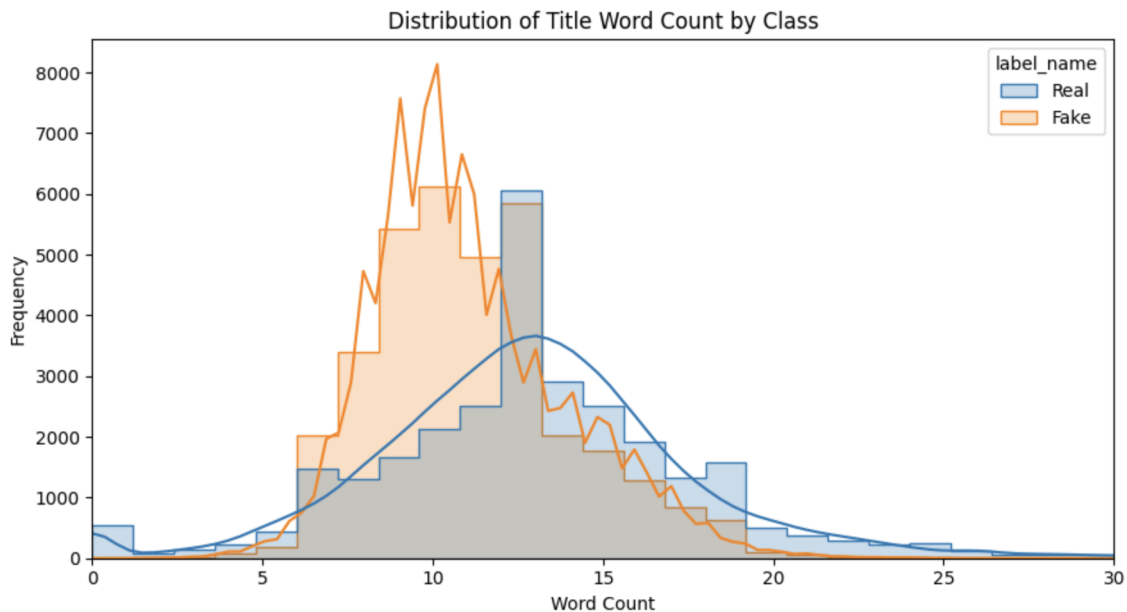


Figure 3.2: Distribution of Title Word Count by Class

3.2.2 Content Length by Class

Word counts were also calculated for the combined article content. The summary statistics show that the average content length is 554.19 words, while the median content length is 414 words. This difference suggests that the distribution of article length is right-skewed, with some articles substantially longer than the typical observation.

Figure 3.3 presents the distribution of content word count by class. The results suggest that the two classes have different length patterns, although they still overlap substantially. Real news articles appear somewhat more concentrated around moderate content lengths, while fake news articles show a broader spread across shorter and mid-range lengths together with a visible right tail. This indicates that content length may provide some useful structural information, but it is not sufficient on its own to distinguish fake news from real news.

These findings are also consistent with the percentile analysis of content length. The

50th percentile is 414 words, the 75th percentile is 685 words, the 90th percentile is 1,115 words, and the 95th percentile is 1,413 words. This substantial variation in article length supports the use of truncation and padding in the recurrent neural network models.

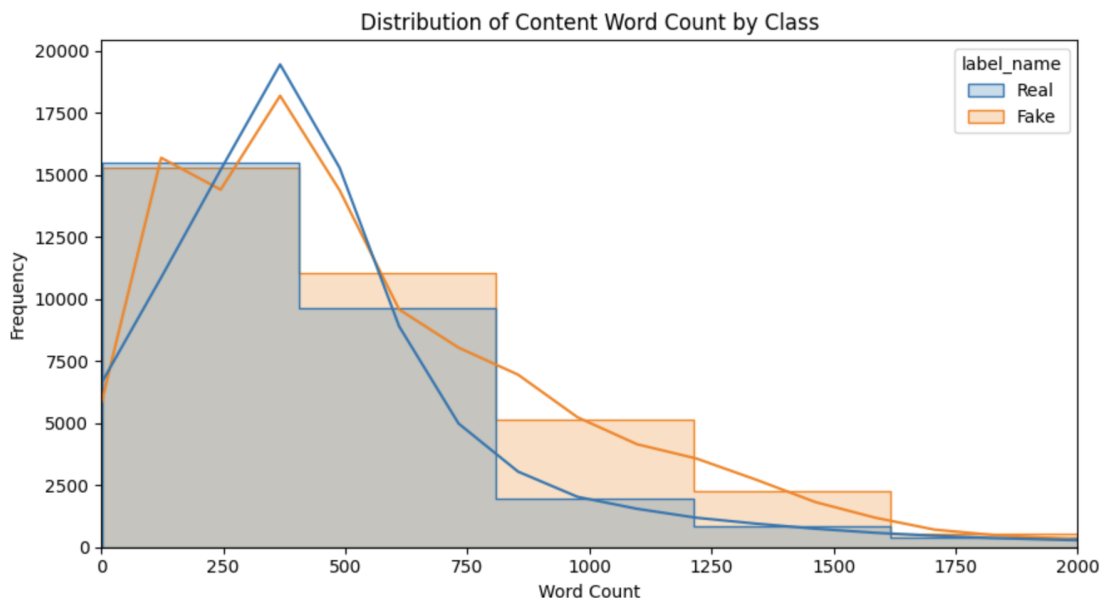


Figure 3.3: Distribution of Content Word Count by Class

3.3 Common Word Patterns

Figures 3.4 and 3.5 present the most frequent words in fake news and real news articles. Many of the most common terms in both classes are related to politics, public figures, and general news reporting. Examples include *trump*, *president*, *clinton*, *election*, and *said*. This suggests that the dataset is strongly centered on political and current-event topics. The substantial overlap in these common words also supports the point raised earlier in Chapter 1 that fake and real news can be difficult to distinguish, since both classes often rely on similar topical vocabulary.

At the same time, the rankings are not identical across the two classes. Some words

appear prominently in both groups, while others are more noticeable in only one of them. For example, *obama* appears among the most frequent words in real news but not in fake news. This difference may reflect the time period or source composition represented in the dataset, although it should be interpreted cautiously. Overall, these patterns suggest that the classification problem is not determined by topic alone. Instead, successful fake news detection likely depends on broader lexical patterns, contextual cues, and writing structure rather than only a few obvious keywords.

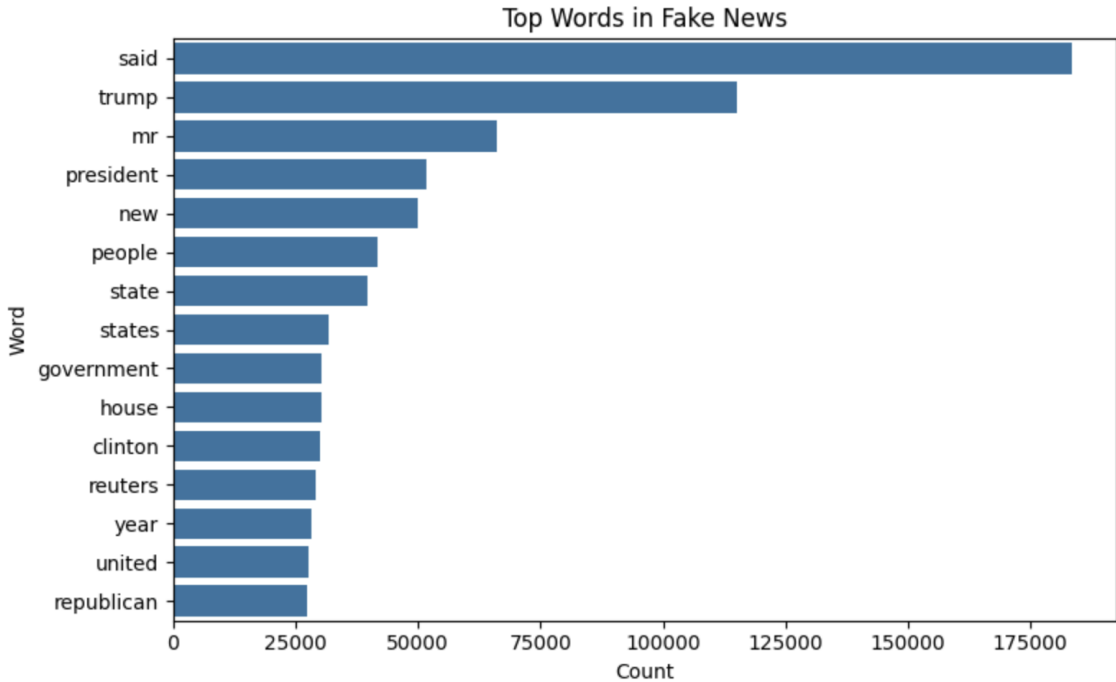


Figure 3.4: Top Words in Fake News

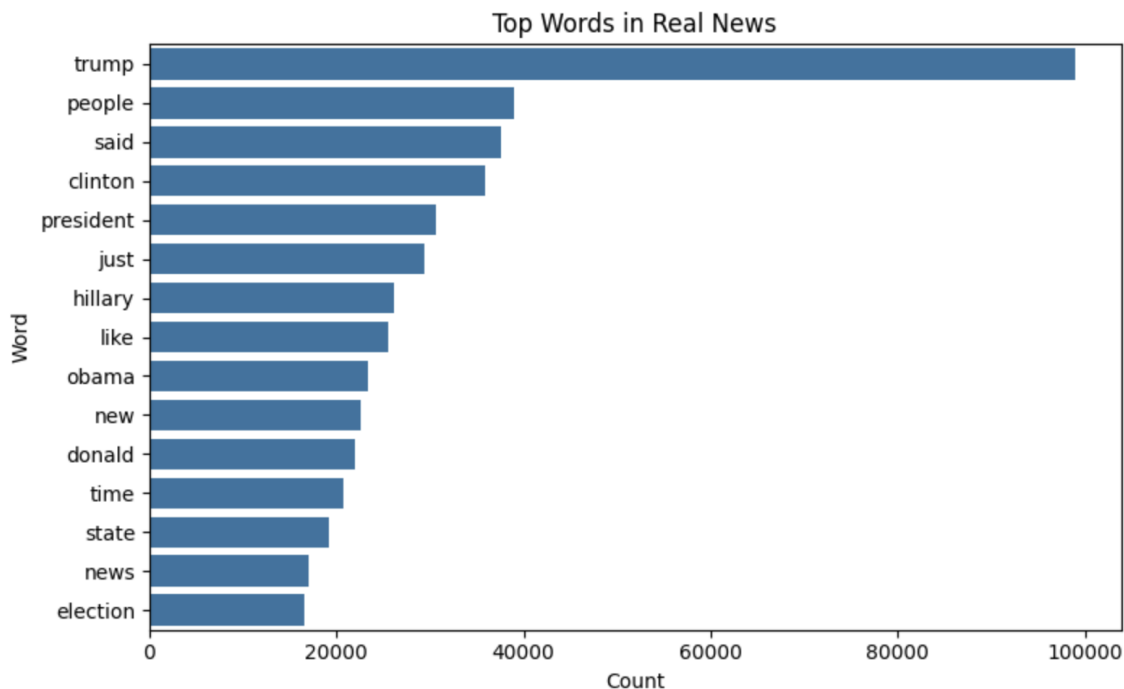


Figure 3.5: Top Words in Real News

3.4 EDA Summary

Overall, the exploratory data analysis suggests that the WELFake dataset is suitable for text-based fake news classification. The cleaned dataset is only moderately imbalanced, so both classes are adequately represented. In addition, title length and content length show some differences across classes, although neither feature alone is sufficient for classification. Fake news titles tend to be somewhat shorter and more concentrated, while content length shows different distributional patterns across the two classes together with substantial overlap. The word frequency analysis further shows that fake and real news share many important political and current-event terms, which suggests that the classification task cannot be solved through simple keyword matching alone. Taken together, these findings motivate the comparative modeling framework presented in the next chapter.

Chapter 4

Modeling Approaches

4.1 Model Comparison Framework

This study compares six models: Logistic Regression, Linear SVM, vanilla RNN, LSTM, GRU, and BiLSTM. These models represent two broad approaches to text classification. The first group consists of classical machine learning models trained on TF-IDF features, whereas the second group consists of recurrent neural network models trained on tokenized text sequences. Including both groups allows the study to compare strong and interpretable baseline methods with more complex neural architectures designed to capture sequential structure in text.

The purpose of this comparison is not only to identify the highest-performing model, but also to evaluate whether more complex architectures provide enough improvement to justify their additional training cost and reduced interpretability. For this reason, the selected models span a range of complexity, from classical linear baselines to gated and bidirectional recurrent networks.

In this study, the classical machine learning models were implemented using scikit-learn, whereas the recurrent neural network models were implemented using TensorFlow/Keras.

4.2 Classical Machine Learning Models

4.2.1 Logistic Regression

Model Configuration:

- Input representation: TF-IDF features from cleaned article content
- Maximum number of features: 20,000
- Stop-word removal: English stop words removed
- N-gram range: unigrams and bigrams, that is, `ngram_range=(1,2)`
- Model setting: `LogisticRegression(max_iter=1000)`

Logistic Regression is a standard method for binary classification and is widely used as a baseline in text classification research. In this study, each article was represented as a sparse TF-IDF vector, and the model learned a linear relationship between those features and the binary class label.

Logistic Regression was included because it provides a simple, interpretable, and computationally efficient benchmark. Since the model assigns coefficients directly to the input features, it is easier to interpret than more complex neural models. Although Logistic Regression does not explicitly model word order, it often performs strongly on text classification tasks when paired with TF-IDF features. For this reason, it serves as an important baseline against which the more complex models can be evaluated.

4.2.2 Linear SVM

Model Configuration:

- Input representation: TF-IDF features from cleaned article content

- Maximum number of features: 20,000
- Stop-word removal: English stop words removed
- N-gram range: unigrams and bigrams, that is, `ngram_range=(1,2)`
- Model setting: `LinearSVC()` with default implementation settings

Linear SVM was also trained on TF-IDF features. Like Logistic Regression, it operates on a sparse vector representation of the text. However, instead of directly estimating class probabilities, Linear SVM seeks a separating hyperplane that maximizes the margin between the two classes in the feature space.

Linear SVM was included because it is one of the strongest traditional methods for high-dimensional document classification. In text mining applications, it often performs well when the feature space is large and sparse. Since fake news detection with TF-IDF features fits this setting, Linear SVM provides an important classical benchmark for comparison with more complex neural architectures.

4.3 Recurrent Neural Network Models

4.3.1 Shared Neural Architecture

Model Configuration:

- Tokenizer: `Tokenizer(num_words=20000, oov_token="<OOV>")`
- Maximum vocabulary size: 20,000
- Maximum sequence length: 300
- Padding and truncation: post-padding and post-truncation

- Embedding dimension: 128
- Dropout rate: 0.3
- Output layer: `Dense(1, activation="sigmoid")`
- Loss function: binary cross-entropy
- Optimizer: Adam

All recurrent neural network models in this study were built on the same general pipeline so that the comparison would focus on differences in the recurrent layer rather than on unrelated design changes. The input to each model was a tokenized and padded sequence of article text. Each token was first mapped into a dense vector representation through an embedding layer, allowing the models to learn distributed word representations during training.

The embedding layer was followed by a recurrent layer, which differed depending on the model. After the recurrent layer, a dropout layer was applied to reduce overfitting, and a final dense layer with sigmoid activation produced the predicted probability for binary classification. Keeping this shared pipeline consistent across neural models makes the comparison more interpretable, since differences in performance can be attributed more directly to the recurrent architecture itself.

4.3.2 Vanilla RNN

Model Configuration:

- Architecture: `Embedding(20000, 128) → SimpleRNN(64) → Dropout(0.3) → Dense(1, activation="sigmoid")`

The vanilla RNN serves as the simplest sequential baseline in this study. It processes the input sequence from left to right and updates a hidden state at each time step. In principle, this allows the model to use earlier words when interpreting later words in the sequence.

The vanilla RNN was included to provide a basic reference point for sequential modeling. Unlike the classical machine learning models, it directly incorporates word order and contextual accumulation across the sequence. However, standard recurrent networks are known to have difficulty preserving useful information across long input sequences. The vanilla RNN therefore serves as a lower-complexity sequential baseline for evaluating whether more advanced recurrent architectures are necessary for article-level fake news detection.

4.3.3 LSTM

Model Configuration:

- Architecture: `Embedding(20000, 128) → LSTM(64) → Dropout(0.3) → Dense(1, activation="sigmoid")`

Long Short-Term Memory, or LSTM, was included because it addresses an important limitation of standard recurrent networks. In a vanilla RNN, the hidden state may gradually lose relevant information as the sequence becomes longer. LSTM mitigates this problem by introducing a memory cell together with gating mechanisms that regulate how information is stored, updated, and forgotten over time.

More specifically, the LSTM architecture uses input, forget, and output gates. The input gate controls how much new information enters the memory cell, the forget gate determines how much previous information is retained, and the output gate controls how much of the stored information is passed to the next layer. This structure allows the model to preserve more useful information across longer sequences, which is especially relevant for news articles where important contextual cues may appear far apart in the text.

4.3.4 GRU

Model Configuration:

- Architecture: `Embedding(20000, 128) → GRU(64) → Dropout(0.3) → Dense(1, activation="sigmoid")`

The Gated Recurrent Unit, or GRU, was included as another gated recurrent architecture. GRU is similar in motivation to LSTM but uses a simpler internal structure. Instead of maintaining a separate memory cell and three gates, GRU typically uses two gates, commonly referred to as the update gate and the reset gate, to control how information is incorporated and retained across the sequence.

GRU was selected because it often provides performance comparable to LSTM while using a more compact architecture. This makes it useful for evaluating whether a simpler gated architecture can achieve results similar to or better than LSTM for fake news detection.

4.3.5 BiLSTM

Model Configuration:

- Architecture: `Embedding(20000, 128) → Bidirectional(LSTM(64)) → Dropout(0.3) → Dense(1, activation="sigmoid")`

The Bidirectional Long Short-Term Memory model, or BiLSTM, extends the LSTM architecture by processing the input sequence in both forward and backward directions. Instead of relying only on past context, the model can incorporate information from both earlier and later words when constructing the sequence representation.

This bidirectional structure is particularly relevant for article-level text, where the meaning of a phrase may depend not only on the words that come before it but also on the words that follow. In news writing, contextual interpretation often depends on relationships across

phrases, clauses, and sentences. BiLSTM was therefore included to test whether access to both forward and backward contextual information improves classification performance relative to one-directional recurrent models.

4.4 Training and Evaluation

Model Configuration:

- Data split: 70% training, 15% validation, 15% test
- Split method: stratified `train_test_split` with `random_state=42`
- Early stopping: monitor = validation loss, patience = 2, restore best weights = True
- Maximum epochs: 5
- Batch size: 64
- Evaluation metrics: accuracy, precision, recall, F1-score, and training time

All models were trained and evaluated using the same train, validation, and test split so that the comparison would be consistent across methods. This design reduces variation caused by different data partitions and allows the model comparison to focus more directly on differences in representation and learning strategy. The same cleaned text data were used throughout, with TF-IDF features used for the classical models and tokenized sequences used for the neural models.

For the neural models, tokenization was fitted on the training set, and the resulting sequences were padded or truncated to a fixed length of 300 tokens using post-padding and post-truncation. Training used early stopping based on validation loss, with a patience value of 2 and restoration of the best model weights. Each neural model was trained for up to 5 epochs with a batch size of 64.

Model performance was assessed using accuracy, precision, recall, and F1-score. Accuracy provides an overall measure of classification correctness, while precision and recall help evaluate performance more carefully by accounting for the balance between false positives and false negatives. F1-score is especially useful because it combines precision and recall into a single summary measure and therefore provides a more informative basis for comparison than accuracy alone. In addition, training time was recorded in order to compare computational efficiency across models.

Because the goal of this study is model comparison, the focus is not only on which model achieved the highest predictive performance, but also on whether any performance gain was large enough to justify additional model complexity and training cost. This is especially relevant in fake news detection, where a slightly stronger model may not always be the most practical choice if it is much slower to train or more difficult to interpret.

Chapter 5

Results and Discussion

5.1 Performance Comparison of Models

Table 5.1 presents the test-set performance of all six models. Among the models considered in this study, BiLSTM achieved the highest overall accuracy and F1-score. GRU also performed very strongly and produced results that were close to BiLSTM across all evaluation metrics. Among the classical machine learning models, Linear SVM remained highly competitive, with performance that was nearly identical to that of the strongest neural models while requiring substantially less training time. Logistic Regression also produced strong results and served as a reliable baseline.

The recurrent neural network models did not perform equally well. While BiLSTM and GRU achieved strong results, the vanilla RNN performed substantially worse than all other models. LSTM improved considerably over the vanilla RNN, but it still performed below GRU and BiLSTM in this study. These results suggest that sequential modeling alone is not sufficient for strong performance. Instead, the ability to preserve and use relevant contextual information across longer sequences appears to be particularly important for article-level fake news detection.

Table 5.1: Performance Comparison of All Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Train Time (s)
BiLSTM	95.76	93.34	97.62	95.43	69.70
GRU	95.64	94.66	95.80	95.23	50.23
Linear SVM	95.63	95.22	95.15	95.19	1.99
Logistic Regression	94.80	94.11	94.44	94.27	4.92
LSTM	87.32	83.85	89.25	86.46	51.47
Vanilla RNN	69.12	74.36	48.72	58.87	87.23

5.2 Discussion of Results

The results of this study provide clear answers to the research questions introduced in Chapter 1. With respect to the first research question, the six machine learning techniques did not perform equally well for fake news detection. BiLSTM achieved the strongest overall performance, followed closely by GRU and Linear SVM. Logistic Regression also performed well, whereas LSTM and especially the vanilla RNN produced weaker results. These findings show that model choice has a substantial effect on predictive performance, even when all models are trained and evaluated on the same cleaned dataset.

With respect to the second research question, the results indicate that recurrent neural network models can provide improvement over strong classical baseline models, but the improvement is limited rather than dramatic. BiLSTM and GRU both slightly outperformed Linear SVM, which suggests that more advanced sequential architectures can provide a modest advantage in this task. However, the performance gap between the best neural models and Linear SVM was small. This means that the results do not support the idea that neural models automatically dominate classical methods in fake news detection. Instead, they suggest that strong baseline models remain highly competitive.

With respect to the third research question, the results suggest that the best balance between predictive performance, efficiency, and model complexity depends on the practical goal of the analysis. If predictive performance alone is the primary consideration, then BiLSTM is the strongest overall model. However, if computational efficiency and simplicity are also important, Linear SVM may provide the most practical balance. Its performance was very close to the top-performing neural models, while its training time was substantially lower. GRU also represents a strong compromise, since it achieved near-best performance with lower training time than BiLSTM.

One particularly important finding is the poor performance of the vanilla RNN. Because this model served as the simplest sequential baseline, its weaker results help show that not all recurrent architectures are equally effective for long text classification. The relatively low recall and F1-score suggest that the vanilla RNN had difficulty preserving useful information across longer article sequences. In contrast, the substantially stronger results of LSTM, GRU, and BiLSTM suggest that gated mechanisms and, in the case of BiLSTM, bidirectional context are important for capturing relevant information in article-level fake news detection.

Taken together, the results indicate that model complexity can improve classification performance, but only when that complexity is paired with an architecture that is well suited to longer text sequences. The best neural models clearly outperformed the weakest recurrent baseline and produced slight improvements over the strongest classical baselines. At the same time, the strong performance of Linear SVM shows that simpler models remain highly effective. This reinforces the importance of comparing advanced methods against strong baselines rather than assuming that greater complexity will necessarily lead to substantially better results.

5.3 Limitations

This study has several limitations. First, although WELFake combines multiple source datasets and is therefore broader than a single-source benchmark, it is still one unified corpus. As a result, the findings of this study should be interpreted as performance on the WELFake dataset rather than as universal performance across all fake news detection settings. Different datasets may vary in writing style, topic distribution, source composition, label quality, and time period, which means that model performance may change under different conditions.

Second, the study focuses only on text-based classification and does not incorporate user behavior, propagation patterns, images, source metadata, or other multimodal information that may also be useful for identifying misinformation. In practice, fake news detection may benefit from combining textual content with these additional signals.

Third, model performance may depend on preprocessing choices such as sequence length, vocabulary size, and cleaning rules. Although consistent settings were used in order to make the comparison fair, alternative parameter choices or more extensive tuning may lead to different results. In addition, the neural models in this study were intentionally kept relatively simple so that the comparison would remain clear and manageable. More extensive hyperparameter tuning or alternative architectures may improve performance beyond the results reported here.

Chapter 6

Conclusion

6.1 Concluding Remarks

This study examined fake news detection as a supervised text classification problem and compared several machine learning techniques using the WELFake dataset. By evaluating both classical machine learning models and recurrent neural network models, the study aimed to better understand how predictive performance, training efficiency, and model complexity interact in text-based fake news detection.

Overall, the results show that more advanced neural architectures can improve classification performance, but the improvement over strong classical baseline methods is limited rather than dramatic. BiLSTM achieved the strongest overall results, while GRU also performed very well. At the same time, Linear SVM remained highly competitive and offered a strong balance between predictive performance and computational efficiency. These findings suggest that model selection for fake news detection should not be based on accuracy alone, but should also consider practicality, interpretability, and training cost.

This study also provides useful insight into the broader modeling tradeoff in fake news detection. More complex architectures may offer some predictive benefit, but that benefit

is not always large enough to justify the additional computational cost. In this sense, the results support a balanced approach to model selection, in which performance should be evaluated together with efficiency and model complexity rather than in isolation.

In addition, the exploratory data analysis suggested that the dataset is strongly associated with political and current-event topics. The substantial overlap in high-frequency words between fake and real news further showed that the classification task is not driven by a few simple keywords. Instead, successful fake news detection appears to depend on broader lexical patterns, contextual structure, and representation quality. This makes the task both practically important and methodologically challenging.

More broadly, this study contributes to the comparative evaluation of text classification models for fake news detection by showing that strong baseline methods remain effective even when compared with more complex recurrent models. The findings therefore support a more careful and evidence-based view of model choice, rather than assuming that greater complexity will automatically produce the most useful solution. Taken together, the results suggest that effective fake news detection depends not simply on using the most advanced model, but on selecting a model whose performance gains are meaningful in relation to its computational cost and interpretability.

6.2 Future Work

Several directions for future work follow naturally from this study. One possible extension would be to evaluate the same models on additional fake news datasets in order to examine whether the results generalize beyond WELFake. Another useful direction would be to incorporate pretrained word embeddings or transformer-based models and compare them against both the recurrent models and the classical baselines used here. Because transformer architectures are designed to capture broader contextual relationships in text, they may

provide further improvement for article-level fake news detection.

Future work could also explore multimodal fake news detection by combining article text with metadata, source information, social propagation patterns, or visual content. In addition, further research could investigate cross-dataset generalization, since a model that performs strongly on one benchmark may not perform equally well on data collected from different sources or time periods. Such extensions would provide a more complete understanding of how different modeling approaches perform in more realistic misinformation detection settings.

Bibliography

- [1] Aïmeur, E., Amri, S., & Brassard, G. (2023). Fake news, disinformation and misinformation in social media: A review. *Social Network Analysis and Mining*, 13(1), Article 30. <https://doi.org/10.1007/s13278-023-01028-5>

- [2] Allcott, H., & Gentzkow, M. (2017). Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2), 211–236. <https://doi.org/10.1257/jep.31.2.211>

- [3] Campante, F. R., Durante, R., Hagemeister, F., & Sen, A. (2025). GenAI misinformation, trust, and news consumption: Evidence from a field experiment. *NBER Working Paper No. 34100*. <https://www.nber.org/papers/w34100>

- [4] Cassar, D. (2023). The misinformation threat: A techno-governance approach for curbing the fake news of tomorrow. *Digital Government: Research and Practice*, 4(4), Article 24, 1–28. <https://doi.org/10.1145/3631615>

- [5] Lazer, D. M. J., Baum, M. A., Benkler, Y., Berinsky, A. J., Greenhill, K. M., Menczer, F., Metzger, M. J., Nyhan, B., Pennycook, G., Rothschild, D., Schudson, M., Sloman, S.

- A., Sunstein, C. R., Thorson, E. A., Watts, D. J., & Zittrain, J. L. (2018). The science of fake news. *Science*, 359(6380), 1094–1096. <https://doi.org/10.1126/science.aao2998>
- [6] Machete, P., & Turpin, M. (2020). The use of critical thinking to identify fake news: A systematic literature review. In M. Hattingh, M. Matthee, H. Smuts, I. Pappas, Y. K. Dwivedi, & M. Mäntymäki (Eds.), *Responsible Design, Implementation and Use of Information and Communication Technology* (Lecture Notes in Computer Science, Vol. 12067, pp. 235–246). Springer, Cham. https://doi.org/10.1007/978-3-030-45002-1_20
- [7] OpenAI. (2022, November 30). Introducing ChatGPT. <https://openai.com/index/chatgpt/>
- [8] Raman, R., Nair, V. K., Nedungadi, P., & Sahu, A. K. (2024). Fake news research trends, linkages to generative artificial intelligence and sustainable development goals. *Heliyon*, 10(3), e24727. <https://doi.org/10.1016/j.heliyon.2024.e24727>
- [9] Verma, P. K., Agrawal, P., Amorim, I., & Prodan, R. (2021). WELFake: Word embedding over linguistic features for fake news detection. *IEEE Transactions on Computational Social Systems*. <https://doi.org/10.1109/TCSS.2021.3068519>
- [10] Vosoughi, S., Roy, D., & Aral, S. (2018). The spread of true and false news online. *Science*, 359(6380), 1146–1151. <https://doi.org/10.1126/science.aap9559>