Headline Hunter: A Data Structure–Based Fake News Detection Approach

Nikita Miller, Anisha Katiyar, Aariz Faridi, Yatish Sikka
M.S. Applied Machine Learning, University of Maryland College Park, Maryland

***Abstract*—The widespread dissemination of fake news poses a significant challenge in the digital world, necessitating effective and scalable detection mechanisms. This project presents a comparative study of fake news detection methods based on classical data structures, TF-IDF vectors, and transformer-based models. We developed a lightweight, interpretable detection system leveraging HashMaps, Heaps, Sets, and other features using data structures and algorithms to classify news articles as real or fake. The system was benchmarked against traditional TF-IDF pipelines and a BERT model using the Kaggle Fake News Classification dataset. Results indicate that while BERT achieved the highest accuracy (97.95%), the data structure–based approach demonstrated superior runtime efficiency and minimal resource consumption, making it suitable for resource-constrained environments. TF-IDF offered a balance between performance and computational cost. However, it still did not offer the interpretability like the data structured approach did. The study highlights critical trade-offs between interpretability, scalability, and accuracy, providing practical insights for deploying fake news detection systems in real-world applications.**
The source code is publicly available at:
https://github.com/ani14kay/Headline-Hunter.

I. Introduction

The growth of misinformation, commonly known as fake news, has emerged as a critical challenge in the digital era. Misleading or fabricated news articles can manipulate public opinion, disrupt democratic processes, and spread harmful narratives with alarming speed. Consequently, developing effective and scalable methods for fake news detection has become a priority for researchers and practitioners alike.

While state-of-the-art deep learning models such as BERT have demonstrated high accuracy in text classification tasks, their deployment is often hindered by significant computational requirements and limited interpretability. In contrast, data structure–based algorithms offer lightweight, transparent alternatives that can operate efficiently in resource-constrained environments.

This project presents a comparative study on fake news detection using data structures and algorithmic techniques, specifically focusing on HashMaps, TF-IDF, Sets, Tries, and Heaps. We aim to evaluate the viability of these methods against traditional TF-IDF pipelines and advanced transformer-based models like BERT.

The primary objectives of this study are threefold: (1) to develop a fake news detection system leveraging data structures for feature extraction and classification, (2) to assess the runtime efficiency, scalability, and interpretability of these methods, and (3) to compare their performance with TF-IDF and BERT-based approaches on standard evaluation metrics.

II. Algorithm and Data Structure Review

This section presents the core data structures employed in our fake news detection system, highlighting their functionality, relevance to the task, and challenges in scalability.

A. Trie for Clickbait Phrase Detection

The Trie data structure, or prefix tree, is employed for efficient detection of clickbait phrases in news headlines. Tries enable rapid lookup of substrings by organizing characters in a tree-like hierarchy, allowing prefix-based searches in $O(m)$ time complexity, where m is the length of the query string.

In this project, a curated list of common clickbait phrases (e.g., “You won’t believe”, “Shocking truth”) is stored in a Trie. During analysis, each headline is tokenized, and phrases are checked against the Trie to flag potentially sensationalized content. The interpretability of Tries makes them suitable for rule-based detection, though maintaining extensive phrase lists can increase memory consumption as data scales.

B. Heap for Top-K Word Length Analysis

Heaps, specifically max-heaps, are utilized to identify the top-k longest words in news articles. This heuristic serves as a proxy for detecting complex or technical jargon, which may correlate with article credibility.

Given a word list extracted from an article, the heap efficiently maintains the top-k entries in $O(n \log k)$ time, where n is the number of unique words. The choice of k can be tuned based on dataset characteristics. While heaps are efficient for such ranking tasks, they offer limited semantic insights beyond word length statistics.

C. Set for Jaccard Similarity Computation

Sets are fundamental in calculating Jaccard similarity, a metric used to quantify the overlap between the title and body text of a news article. The Jaccard similarity is defined as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B represent the sets of unique tokens from the title and article body, respectively. Low Jaccard scores may indicate a mismatch between the headline and content, a common trait in fake news.

Sets provide constant-time lookup operations, making them well-suited for this task. However, set-based methods lack sensitivity to word order and semantic similarity, limiting their effectiveness in nuanced cases.

D. Hash Map for Word Frequency Ratio

To identify repetitive patterns in fake news, we compute the ratio of the most frequent word to the total word count for both the title and article body. This is implemented using a hash map (`dict`) to store word frequencies.

Each word is inserted into the dictionary with its count incremented. The final ratio is obtained by dividing the maximum frequency by the total number of words in the text.

Time complexity: $O(n)$, where n is the number of tokens. **Space complexity**: $O(m)$, where m is the number of unique tokens.

This feature is useful for identifying articles that overuse certain terms, a tactic often associated with low-effort or spammy content.

E. Insertion Sort for Punctuation Emphasis

To capture sensational tone in fake news headlines, we compute the length of the longest contiguous run of punctuation marks (e.g., "!!!", "??", "...") in the article title.

Punctuation runs are extracted using the regular expression pattern, which matches sequences of two or more exclamation marks, question marks, or periods. These sequences are then ranked by length using an explicit implementation of **insertion sort**.

Insertion sort is chosen here for its simplicity and suitability for small input sizes. Titles rarely contain more than a few punctuation runs, so the algorithm performs efficiently despite its $O(n^2)$ worst-case complexity.

Time complexity: $O(n^2)$, where n is the number of punctuation runs. **Space complexity**: $O(n)$.

This feature adds interpretability to the model, quantifying exaggerated punctuation often associated with emotionally charged or misleading headlines.

F. Linear Scan for Capital Word Detection

This feature extracts the longest fully capitalized word in the title. It is computed using a single-pass linear scan:

- Each token is checked using `isupper()`.
- A max-length tracker is used to record the longest matching word.

Time complexity: $O(n)$, where n is the number of words in the title. **Space complexity**: $O(1)$.

This metric helps flag headlines that use visual emphasis (e.g., "EXCLUSIVE REPORT") — a common trait in misleading news.

G. Summary

The features implemented collectively span a range of foundational data structures: Tries, Heaps, Sets, Hash Maps, and linear scans. This structure-driven approach offers multiple advantages:

- **High interpretability**: Each feature has a clearly defined logic and intuitive meaning.
- **Efficient computation**: All features can be extracted in linear or near-linear time.
- **Modularity**: Features can be added or removed independently, allowing for fast ablation testing.

Unlike TF-IDF or embedding-based approaches, these handcrafted features are suited for environments with constrained resources (e.g., edge devices, real-time inference).

H. Scalability and Real-World Relevance

The selected data structures offer favorable trade-offs in terms of runtime efficiency, interpretability, and ease of implementation. Hash maps and sets offer constant-time operations on average, while Tries and Heaps enable targeted pattern detection and ranking with low overhead.

However, scaling this system to web-scale datasets introduces limitations. Trie memory consumption increases linearly with phrase count and average phrase length ($O(N \cdot L)$), and hash maps grow with vocabulary diversity. Additionally, Jaccard computations require explicit set construction and can become bottlenecks at scale.

Despite these challenges, the system's explainability and performance on mid-sized datasets make it well-suited for use in constrained environments such as browser extensions, mobile apps, and low-power fact-checking tools.

III. Implementation

This section outlines the complete pipeline used in our fake news detection system, including dataset preparation, preprocessing, feature extraction using data structures, and the training of three model variants.

A. Dataset

We used the Fake News Classification dataset from Kaggle [1], comprising two CSV files: `train.csv` and `test.csv`. These were merged for consistent preprocessing and evaluation. Each entry includes a `title`, `text`, and a binary `label` (0 = fake, 1 = real).

B. Preprocessing

We performed standard NLP preprocessing on the `title` and `text` fields:

- Conversion to lowercase
- Removal of URLs, HTML tags, citations, and punctuation using regular expressions
- Whitespace normalization

Processed outputs were stored as `clean_title` and `clean_text`, which were used for some handcrafted features and combined into a single `combined` field for TF-IDF and BERT pipelines.

C. Feature Extraction Using Data Structures

We engineered eight features using classical data structures and algorithms, applied to either the cleaned or raw text fields depending on the nature of the feature:

- **Clickbait Score (Trie)**: Applied to `clean_title`. A prefix tree was used to match known clickbait phrases.

- **Top-K Word Length (Heap)**: Applied to `clean_title` and `clean_text`. A max-heap extracted the average length of the five longest words.
- **Max Word Frequency Ratio (Hash Map)**: Applied to `clean_title` and `clean_text`. A frequency map computed the ratio of the most common word to total word count.
- **Jaccard Similarity (Set)**: Applied to `clean_title` and `clean_text`. Token sets were compared using Jaccard index to assess lexical overlap.
- **Punctuation Emphasis (Regex + Insertion Sort)**: Applied to the raw `title`. A regex extracted repeated punctuation runs (e.g., "!!!", "..."), which were sorted by length using insertion sort.
- **Longest Capitalized Word (Linear Scan)**: Applied to the raw `title`. The longest fully capitalized token was selected using a single-pass scan.

Features dependent on formatting (punctuation and capitalization) intentionally bypassed preprocessing to preserve signal. In contrast, lexical features relied on cleaned text for consistent tokenization.

This hybrid approach ensured both surface-level stylistic and content-level lexical traits were captured efficiently.

D. Model 1: DSA + Random Forest

We trained a Random Forest classifier using the eight DSA-based features. The model used 200 trees and a maximum depth of 10. Data was split into 80% training and 20% testing.

E. Model 2: TF-IDF + Random Forest

For comparison, we trained a second Random Forest model using TF-IDF vectors generated from the `combined` field. We limited the vectorizer to the top 5000 terms. The same training, evaluation, and metric logging procedures were applied.

F. Model 3: BERT Fine-Tuning

We fine-tuned a `bert-base-uncased` model using the HuggingFace Transformers library for binary classification. The pipeline included:

1) Tokenization of `combined` with padding and truncation
2) Conversion to PyTorch datasets
3) Training for 2 epochs with a batch size of 8
4) Evaluation using the HuggingFace `Trainer`

Metrics collected included accuracy, F1 score, inference time per sample, model size (in MB), and parameter count.

G. Summary of Features

Table I summarizes the engineered features utilized in the DSA model.

TABLE I
ENGINEERED FEATURES FOR DSA-BASED DETECTION

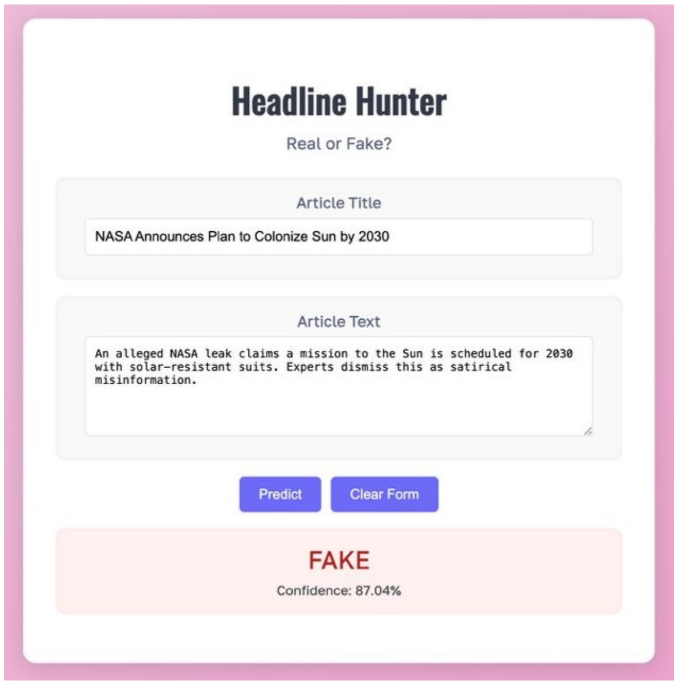
Feature	Description
Clickbait Score	Phrase-based heuristic (conceptual Trie)
Title Top-K Word Length	Heap-based average of longest title words
Text Top-K Word Length	Heap-based average of longest text words
Title Max Word Ratio	HashMap frequency ratio in title
Text Max Word Ratio	HashMap frequency ratio in text
Jaccard Title-Text Similarity	Set-based lexical overlap
Title Punctuation Emphasis	Proportion of ! and ? characters
Title Longest Capital Word	Length of longest uppercase word

H. User Interface Demo

To showcase the practical application of our fake news detection system, we developed a simple web interface titled **Headline Hunter**. This interface allows users to input an article's title and body text, upon which the system predicts whether the content is real or fake, along with a confidence score.

The interface utilizes the trained DSA-based model for its lightweight and fast inference capabilities, ensuring near-instantaneous feedback. This highlights the viability of deploying such models in real-time scenarios like browser extensions or mobile applications.

The source code for the user interface and model integration is publicly available at:
https://github.com/ani14kay/Headline-Hunter.


Headline Hunter

Real or Fake?

Article Title

NASA Announces Plan to Colonize Sun by 2030

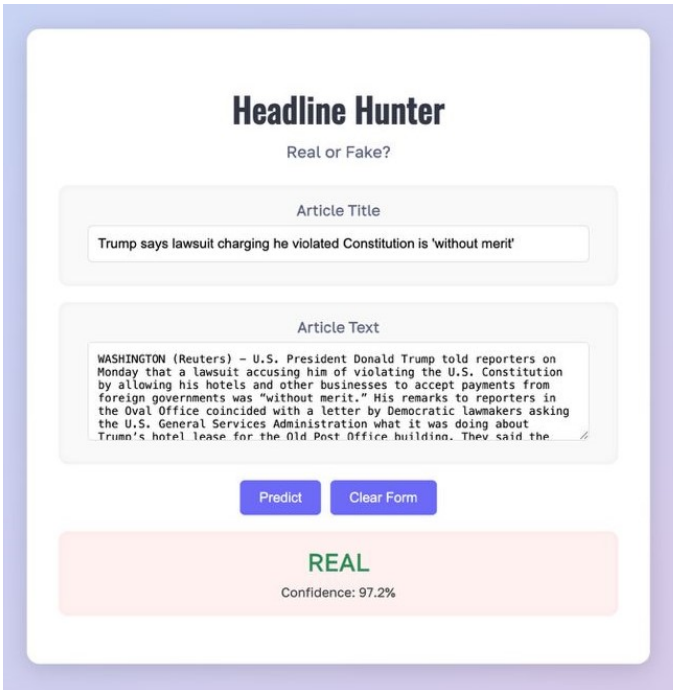
Article Text

An alleged NASA leak claims a mission to the Sun is scheduled for 2030 with solar-resistant suits. Experts dismiss this as satirical misinformation.

FAKE

Confidence: 87.04%

Fig. 1. Headline Hunter: Fake News Detection Interface with Example Prediction (Fake detected)



Trump says lawsuit charging he violated Constitution is 'without merit'

WASHINGTON (Reuters) – U.S. President Donald Trump told reporters on Monday that a lawsuit accusing him of violating the U.S. Constitution by allowing his hotels and other businesses to accept payments from foreign governments was "without merit." His remarks to reporters in the Oval Office coincided with a letter by Democratic lawmakers asking the U.S. General Services Administration what it was doing about Trump's hotel lease for the Old Post Office building. They said the

REAL
Confidence: 97.2%

Fig. 2. Headline Hunter: Fake News Detection Interface with Example Prediction (Real detected)

IV. Analysis and Results

This section evaluates the performance of the three fake news detection approaches implemented: Data Structure–Based (DSA), TF-IDF, and fine-tuned BERT. Key metrics such as accuracy, F1 score, training time, inference time, and model size are reported, followed by an analysis of limitations and trade-offs.

A. Performance Metrics

Table II summarizes the comparative results.

TABLE II
Performance Comparison of Fake News Detection Methods

Model	Accuracy	F1 Score	Train Time	Inference Time	Model Size
DSA	0.9102	0.9211	1.94 sec	0.000008 sec	10.32 MB
TF-IDF	0.9597	0.9632	5.06 sec	0.000016 sec	5.01 MB
BERT	0.9795	0.9809	3.5 hours	0.1171 sec/sample	437.96 MB

The BERT model achieved the highest accuracy and F1 score due to its superior contextual understanding. However, this came at a significant cost in terms of computational resources and inference speed. The TF-IDF model offered a balanced trade-off between performance and efficiency. The DSA-based model, while slightly lower in accuracy, demonstrated exceptional runtime efficiency and minimal resource consumption, making it well-suited for deployment in resource-constrained environments.

B. Limitations and Bottlenecks

Each method presents unique limitations:

- **DSA**: The sparse, surface-level features engineered from data structures such as HashMaps, Heaps, and Sets struggle to capture nuanced semantic relationships. As a result, the model may fail to detect sophisticated forms of misinformation.
- **TF-IDF**: Although it improves accuracy over the DSA approach, TF-IDF suffers from large vector dimensionality, leading to increased memory usage and reduced interpretability.
- **BERT**: While BERT excels in accuracy, its high resource demand, substantial model size (438 MB), and slow inference time (0.1171 sec per sample) limit its applicability in real-time or edge deployments.

C. Real-World Exceptions Encountered

In developing and testing the system, several practical challenges arose that reflect real deployment scenarios:

- **Missing or malformed values**: Some rows in the dataset contained 'NaN', non-string types, or entirely empty fields, requiring explicit handling before tokenization or feature extraction.
- **Encoding inconsistencies**: Certain articles contained non-ASCII characters or Unicode anomalies, which led to parsing issues during cleaning and model input preparation.
- **Feature dependency on format**: Features like punctuation runs and capitalized word length failed if applied to already-cleaned text. This required selective bypass of preprocessing for specific feature extractors.
- **Dataset imbalance by topic**: While labels were balanced, topical bias in the dataset (e.g., clustering around certain political topics) may affect generalization in real-world deployment.
- **Resource usage with BERT**: Even with only two training epochs, BERT required substantial memory and time. This made experimentation slower compared to DSA and TF-IDF models.

These exceptions informed our design decisions and emphasize the need for robustness and fallback logic in production-grade fake news detection systems.

D. Trade-offs and Practical Considerations

The choice of detection method depends on the deployment context:

- For applications prioritizing interpretability, low latency, and minimal resource usage (e.g., browser plugins, mobile apps), the DSA-based approach is recommended.
- For scenarios requiring higher accuracy and where computational resources are not a constraint (e.g., server-side batch processing), BERT is the preferred solution.
- TF-IDF serves as a middle ground, offering moderate performance with reasonable efficiency but lacks deep semantic understanding.

V. Conclusion

This study conducted a comparative analysis of fake news detection methods leveraging classical data structures, TF-IDF vectors, and transformer-based models. The data structure–based approach demonstrated notable advantages in runtime efficiency and interpretability, making it suitable for lightweight, real-time applications. TF-IDF offered a balanced trade-off between accuracy and computational cost, while BERT achieved superior performance at the expense of significant resource consumption.

Key findings include:

- DSA models are highly efficient and interpretable but struggle with nuanced linguistic patterns.
- TF-IDF enhances accuracy but suffers from scalability and lacks semantic depth.
- BERT delivers state-of-the-art performance but requires substantial computational resources.

Future work may focus on hybrid approaches that combine DSA interpretability with contextual embeddings from models like BERT.

VI. Bonus

A. Ethical Considerations

Automated fake news detection systems raise several ethical concerns:

- **False Positives**: Mislabeling legitimate news as fake can harm the credibility of reputable outlets and infringe upon freedom of expression.
- **Training Data Bias**: Datasets may reflect biases in topic coverage and source reliability, leading to skewed detection outcomes.
- **Censorship Risks**: Deploying such systems without transparency could facilitate censorship, especially in politically sensitive contexts.

Ethical deployment requires transparent models, continuous dataset auditing, and clear delineation between fact-checking and content suppression.

B. Social Impact

The societal implications of scalable fake news detection are far-reaching:

- **Potential Integrations**: Browser plugins, AI-powered news aggregators, and digital assistants can leverage lightweight models for real-time misinformation alerts.
- **Real-World Examples**: Applications include combating COVID-19 misinformation, monitoring election-related disinformation, and curbing viral hoaxes.
- **Long-Term Implications**: Widespread deployment can enhance public awareness but also necessitates ethical safeguards to prevent misuse.

C. BERT Comparison

BERT serves as a pretrained baseline, offering unmatched contextual understanding of language. However, its deployment involves trade-offs:

- **Speed**: Inference is orders of magnitude slower compared to traditional methods.
- **Size**: The model's 438 MB footprint limits usage on edge devices.
- **Accuracy**: Despite costs, BERT's superior accuracy justifies its use in high-stakes applications like misinformation monitoring by regulatory bodies or fact-checking agencies.

D. Proposed Enhancements

To improve detection accuracy while retaining interpretability, the following enhancements are proposed:

- **Stylometric Features**: Incorporating metrics such as capitalization percentage, punctuation frequency, and sentence length variability to capture writing style anomalies.
- **Named Entity Overlap**: Comparing named entities in titles and bodies to detect discrepancies indicative of clickbait or misinformation.
- **Semantic Drift Outlier Detection**: Identifying articles with significant semantic divergence from mainstream reporting using lightweight embedding techniques.

VII. Contributions and Reflection

A. Individual Contributions

This project was a collaborative effort by all group members, with responsibilities distributed as follows:

- **Yatish Sikka**: Led the implementation of data structure–based feature engineering, including HashMap frequency computations and Heap-based top-k word analysis. Also contributed to the initial preprocessing pipeline and dataset cleaning functions.
- **Nikita Miller**: Focused on developing and fine-tuning the TF-IDF vectorizer, managing vector dimensionality constraints, and training the Random Forest classifier. Additionally handled Jaccard similarity computations and custom stylometric feature design.
- **Aariz Faridi**: Implemented the end-to-end BERT fine-tuning pipeline using HuggingFace Transformers, including dataset tokenization, model training, and evaluation metrics. Also consolidated performance benchmarks across models.
- **Anisha Katiyar**: Coordinated the literature review on fake news detection methods, drafted the ethical considerations and social impact sections, and designed result visualizations. Supported evaluation and runtime performance analysis.

B. Reflection

Throughout this project, the team gained practical experience in applying fundamental data structures to real-world NLP problems. The exercise reinforced the importance of efficient algorithm design, especially in resource-constrained environments.

Working with large-scale models like BERT highlighted the trade-offs between accuracy and computational demands, prompting discussions on scalability and deployment feasibility. Moreover, addressing ethical implications deepened our understanding of the societal responsibilities associated with automated misinformation detection.

Despite facing challenges in balancing model complexity with interpretability, the team successfully demonstrated that lightweight, data structure–based methods can serve as viable alternatives for certain applications. The project underscored the value of hybrid approaches that blend classical techniques with modern machine learning to achieve practical and responsible AI solutions.

References

[1] A. Singh, “Fake News Classification Dataset,” Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/aadyasingh55/fake-news-classification

[2] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] T. Wolf *et al.*, “Transformers: State-of-the-Art Natural Language Processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[5] H. Lhoest *et al.*, “Datasets: A Community Library for Natural Language Processing,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 175–184, 2021.