

FakeShield: An AI-Powered Multi-Signal Fake News Detection Platform Using TF-IDF and Logistic Regression

Naveera Sajid Dhanse

Department of Artificial Intelligence and Data Science

*Anjuman-1-Islam Abdul Razzak Kalsekar Polytechnic Campus,
New Panvel, Navi Mumbai, Maharashtra, India*

Guide: Prof. Ali Karim

Abstract – The proliferation of misinformation across digital media has emerged as a critical societal challenge, necessitating automated and scalable detection solutions. This paper presents FakeShield, a full-stack AI-powered fake news detection platform that combines machine learning-based text classification with a multi-signal credibility scoring engine, real-time news aggregation, and Wikipedia-assisted fact verification. The system employs a TF-IDF vectoriser paired with a Logistic Regression classifier trained on a large-scale labelled dataset (~116 MB of real and fabricated news articles), achieving competitive accuracy on binary fake/real classification tasks. Beyond raw model output, FakeShield computes a composite credibility score incorporating keyword sentiment analysis, title structure heuristics, and article length signals. The platform is built on a FastAPI backend, React + Vite frontend with a cyberpunk aesthetic, MySQL persistence layer, JWT-based authentication, and live news ingestion via the GNews API and RSS feeds. A comprehensive black-book audit identifies six critical issues—including non-functional biometric authentication, absence of HTTPS, and exposed environment secrets—and proposes a prioritised six-week remediation roadmap. Future work includes fine-tuning DistilBERT for improved semantic understanding, Redis caching, a browser extension, and a WhatsApp misinformation-detection bot targeting the Indian context. FakeShield demonstrates that a well-structured student project can evolve into a production-grade, portfolio-ready application with systematic engineering discipline.

Keywords — fake news detection; natural language processing; TF-IDF; logistic regression; FastAPI; React; credibility scoring; misinformation; DistilBERT; information integrity

1. INTRODUCTION

The rapid diffusion of misinformation through social media platforms, messaging applications, and online news aggregators constitutes one of the defining information-security challenges of the early twenty-first century. Studies indicate that false news spreads approximately six times faster

than factual reporting on platforms such as Twitter/X, and the phenomenon has been observed to influence electoral outcomes, public health behaviour, and financial markets. The Indian subcontinent presents a particularly acute instance of this problem: WhatsApp-mediated rumour propagation has been linked to communal violence and mob lynchings, underscoring the need for accessible, real-time verification tools that function in resource-constrained and multilingual environments.

Automated fake news detection is a non-trivial natural language processing (NLP) task. Classical approaches—term-frequency-based features combined with linear classifiers—offer computational efficiency and reasonable accuracy but fail to capture semantic context and are vulnerable to adversarial paraphrasing. Contemporary deep-learning methods, including transformer architectures such as BERT and RoBERTa, have achieved state-of-the-art performance on benchmark datasets but introduce significant inference latency and resource requirements that complicate deployment in web-scale applications.

This paper presents FakeShield, a full-stack web platform designed to bridge the gap between ML research prototypes and a usable product. FakeShield contributes the following: (i) a production-structured FastAPI + React architecture integrating ML inference, live news, and fact-checking in a single platform; (ii) a multi-signal credibility engine extending binary ML output with heuristic signals; (iii) a comprehensive security and quality audit providing a reproducible evaluation methodology for student ML projects; and (iv) a concrete upgrade roadmap from TF-IDF/LR to DistilBERT fine-tuning with RLHF feedback loops.

The remainder of this paper is organised as follows. Section II reviews related work. Section III describes the system architecture. Section IV details the ML pipeline. Section V presents the multi-signal credibility engine. Section VI discusses the security audit findings. Section VII outlines the improvement roadmap. Section VIII concludes with future directions.

2. RELATED WORK

A. Classical NLP Approaches

Early automated fake news detection systems relied heavily on bag-of-words representations and shallow classifiers. Pérez-Rosas et al. demonstrated that n-gram features combined with support vector machines (SVMs) achieved over 76% accuracy on a corpus of satirical versus legitimate news. TF-IDF representations have been extensively validated as strong baselines, particularly for datasets in which stylistic differences between fake and real news are pronounced—such as the use of sensationalist vocabulary and excessive capitalisation in fabricated content.

B. Deep Learning Methods

The introduction of transformer-based language models fundamentally advanced fake news detection. BERT-based classifiers fine-tuned on the LIAR dataset achieved macro-F1 scores exceeding 0.68 on a six-class veracity labelling task. More recent work has explored cross-lingual models such as XLM-R for multilingual misinformation detection, and graph neural networks for propagation-aware fake news identification that leverages social network structure in addition to article content.

C. Multi-Signal and Knowledge-Grounded Systems

Researchers have increasingly moved beyond unimodal text classification toward multi-signal systems. Popat et al. proposed CredEye, which combines source credibility, article stance, and entity-level fact verification from knowledge bases. The integration of Wikipedia as a lightweight, open-access knowledge graph has been validated as an effective fact-checking signal. FakeShield adopts this multi-signal philosophy, combining ML probability, keyword heuristics, title analysis, and Wikipedia search in a unified credibility score.

D. Deployment and Production Systems

Despite substantial academic progress, there remains a significant gap between research prototypes and deployable applications. Full-stack platforms integrating live news ingestion, user authentication, persistent history, and rate-limited APIs are underrepresented in the literature. This work addresses that gap by presenting a complete system design including security considerations, database schema, API structure, and frontend UX alongside the ML components.

3. SYSTEM ARCHITECTURE

A. Overview

FakeShield follows a three-tier client-server architecture comprising a React + Vite single-page application (SPA) frontend, a FastAPI asynchronous backend, and a MySQL relational database. Table I summarises the technology stack.

Table 1. Technology Stack

Layer	Technology	Notes
Backend	FastAPI 0.111.0	Async, OpenAPI auto-docs
Database ORM	SQLAlchemy 2.0.30	Declarative models
Database	MySQL via PyMySQL	Users, analyses, feedback
ML Engine	scikit-learn	Logistic Regression + TF-IDF
Auth	python-jose + passlib	JWT tokens, bcrypt hashing
Frontend	React 18 + Vite	Tailwind CSS, Framer Motion
State Mgmt	Zustand	Token persistence
Scraping	BeautifulSoup4 + httpx	URL article extraction
News Feed	feedparser + GNews API	RSS + REST live news
Rate Limiting	slowapi	Request throttling

B. Data Flow

The canonical request lifecycle proceeds as follows. A user submits an article text or URL through the React frontend. The Axios HTTP client attaches a JWT Bearer token and dispatches a POST request to the FastAPI /analyze endpoint. The router validates the token via the `get_current_user` dependency, passes the text to the ML predictor service, which returns a binary prediction and confidence probability. The credibility service then computes a composite score from multiple signals. The result is persisted to the MySQL analyses table and returned to the frontend, where the ResultCard and CredibilityPanel components render the verdict. If a URL is submitted, the BeautifulSoup4 scraper first extracts article body text before the ML pipeline is invoked.

C. Module Organisation

The backend adopts a clean separation of concerns: routers/ handles HTTP routing and request validation; services/ encapsulates business logic including ML inference, credibility computation, and Wikipedia queries; models/ defines SQLAlchemy ORM entities; and schemas/ declares Pydantic request/response types. The ML predictor is instantiated as a singleton at startup, loading model.pkl (~40 KB) and vectorizer.pkl (~180 KB) once into memory to minimise per-request overhead.

4. MACHINE LEARNING PIPELINE

A. Dataset

The model was trained on a combined dataset comprising two CSV files—Fake.csv and True.csv—totalling approximately 116 MB of labelled news articles. This dataset is representative of the widely used ISOT Fake News Dataset format, containing news articles across political, world news, and governmental domains. The binary labelling (fake / real) enables supervised classification training.

B. Feature Engineering — TF-IDF

Text is represented using Term Frequency-Inverse Document Frequency (TF-IDF) vectorisation. For a term t in document d across corpus D , the TF-IDF weight is defined as:

$$\text{tfidf}(t,d,D) = \text{tf}(t,d) \times \log(|D| / \text{df}(t,D))$$

where $\text{tf}(t,d)$ is the raw term count normalised by document length, and $\text{df}(t,D)$ is the number of documents containing term t . This representation captures lexical patterns characteristic of fabricated content—including inflammatory vocabulary, excessive capitalisation, and domain-specific trigger words—while suppressing common function words via the IDF weighting.

C. Classifier — Logistic Regression

A Logistic Regression classifier is trained on the TF-IDF feature matrix. The model learns a weight vector w such that the posterior probability of the 'fake' class is:

$$P(y=\text{fake} \mid x) = \sigma(w^T \cdot \text{tfidf}(x)) = 1 / (1 + e^{-(w^T \cdot \text{tfidf}(x))})$$

The sigmoid output provides a calibrated probability estimate used directly as the confidence score displayed to users. Logistic Regression was selected for its interpretability, fast inference (sub-millisecond per prediction), and compact serialised form (~40 KB model.pkl), making it suitable for deployment in memory-constrained environments.

D. Keyword Extraction

FakeShield reuses the trained TF-IDF vectoriser to extract top- N keywords from submitted text. The inference-

time TF-IDF vector for the input document is computed, and the N terms with the highest weights are returned as the keyword set. This approach is computationally free given that the vectoriser transform is already required for classification, and produces contextually relevant terms that align with the model's decision features.

E. Limitations

The TF-IDF + LR pipeline has three principal limitations. First, TF-IDF is a bag-of-words model that ignores word order and semantic relationships; consequently, 'not fake' and 'fake' yield similar representations. Second, Logistic Regression is a linear model incapable of capturing the non-linear linguistic patterns present in satire and nuanced political commentary. Third, the training distribution is skewed toward US political news, which may degrade performance on Indian regional news, scientific misinformation, and AI-generated synthetic text.

5. MULTI-SIGNAL CREDIBILITY ENGINE

A key contribution of FakeShield is its credibility engine, which extends the binary ML classification with a composite score derived from multiple independent signals. This design acknowledges that no single classifier is infallible and that orthogonal signals can compensate for individual model weaknesses.

The credibility score C is computed as a weighted linear combination of: (i) ML confidence probability P_{ml} ; (ii) a keyword sentiment penalty K derived from a lexicon of high-risk inflammatory terms; (iii) a title structure score T that penalises all-caps headlines and excessive punctuation; and (iv) an article length signal L that discounts very short texts as potentially incomplete or decontextualised. The composite score is normalised to $[0, 100]$ and presented to users alongside a qualitative label (High / Medium / Low credibility).

Additionally, the fact-checker module queries the Wikipedia API with extracted claim entities. Wikipedia snippet relevance is scored using TF-IDF cosine similarity between the user's claim and retrieved article summaries, providing a knowledge-grounded verification signal distinct from the ML classifier.

6. IMPROVEMENT ROADMAP

A. Short-Term (Weeks 1–3): Security and Reliability

The immediate priority is remediation of all critical and high-severity security findings as enumerated in Section VI. Concurrently, a pytest-based automated test suite should be established covering the ML predictor, authentication service, credibility engine, and all API endpoints. A GitHub Actions CI pipeline should execute this suite on every pull request and block merges on test failure. Rate limiting on the /analyze and

/factcheck endpoints should be confirmed active at 20 requests per minute per authenticated user.

B. Medium-Term (Weeks 4–5): ML Upgrade and UX Polish

The ML pipeline upgrade follows a phased approach. In Phase 1, the current model's precision, recall, and F1 scores on a held-out test split are evaluated and published on the platform's About page to establish a transparency baseline. In Phase 2, a DistilBERT model [11] is fine-tuned on the same training data. DistilBERT retains approximately 97% of BERT's accuracy at 40% fewer parameters, making it more suitable for web deployment than full BERT. A/B testing between the LR and DistilBERT models will quantify the accuracy-latency trade-off.

Table 2. Platform completeness Assessment

Dimension	Rating	Key Gap
Overall Completeness	68%	Polish, security, scalability
ML Model Quality	Good baseline	Deep-learning upgrade required
Security Posture	Medium	HTTPS, secrets, SSRF, face auth
Code Quality	Good	factcheck.py needs splitting
Frontend Quality	Strong visual	Responsiveness, accessibility
Test Coverage	Very low	No automated test suite
Documentation	Minimal	README, API docs, setup guide

Frontend improvements include mobile responsiveness auditing at 375 px viewport width, skeleton loading states replacing single spinners, empty-state illustrations, and an analysis permalink enabling users to share results via a short URL.

C. Long-Term (Week 6+): Deployment and Scalability

The full stack will be containerised using Docker Compose, enabling one-command local setup and reproducible deployment. The production deployment targets Railway (backend), Vercel (frontend), PlanetScale (MySQL), and Upstash Redis (caching layer). Redis caching of identical text/URL analysis results with a one-hour TTL will significantly reduce ML inference load and database write frequency. Alembic database migrations will replace manual schema management.

D. Future Features

Three future features present significant societal impact potential for the Indian context: (i) a WhatsApp/Telegram bot enabling users to forward suspicious messages for ML analysis, directly addressing WhatsApp-mediated misinformation propagation; (ii) a Chrome/Firefox browser extension overlaying FakeShield verdict badges on news articles as users browse; and (iii) a multilingual model based on XLM-R supporting Hindi, Bengali, and other Indian languages underserved by English-centric detection systems.

7. RESULTS AND DISCUSSION

The platform's core differentiator relative to comparable student projects is the combination of multi-signal credibility scoring and live news integration within a polished, branded UI. Most academic fake news detection implementations present a bare ML inference endpoint; FakeShield provides a complete product experience including user authentication, persistent history, dashboard analytics, Wikipedia fact-checking, and a cyberpunk visual identity that enhances presentation value at academic and industry showcases.

The primary technical debt is the absence of automated testing, which creates regression risk for any ML or backend changes. The six-week remediation roadmap presented in Section VII addresses this systematically, prioritising security hardening before feature expansion—a discipline that distinguishes engineering-mature projects from purely feature-driven prototypes.

8. CONCLUSION

This paper has presented FakeShield, a full-stack AI-powered fake news detection platform demonstrating that principled software engineering can elevate a student ML project to production-grade quality. The system integrates TF-IDF + Logistic Regression classification, multi-signal credibility scoring, live news aggregation, Wikipedia fact-verification, JWT-authenticated user accounts, and persistent analysis history within a cohesive FastAPI + React architecture.

A systematic black-book audit identified six security findings requiring immediate remediation, with particular urgency around exposed environment credentials, absent HTTPS, and a non-functional biometric authentication feature that creates false user trust. The proposed six-week roadmap provides a concrete path from the current 68% completeness rating to a deployment-ready, portfolio-grade product.

Future work will focus on fine-tuning DistilBERT to address the semantic limitations of the current linear classifier, implementing a Redis caching layer to improve scalability, and extending the platform to support Indian-language content through XLM-R multilingual models and a WhatsApp bot

interface—addressing a high-impact real-world misinformation vector in the Indian context.

REFERENCES

- [1] V. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.
- [2] S. Sharma, "Misinformation and WhatsApp in India: Implications and policy responses," Observer Research Foundation, Tech. Rep., 2019.
- [3] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
- [4] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," in *Proc. 27th Int. Conf. Comput. Linguistics (COLING)*, 2018, pp. 3391–3401.
- [5] F. Ahmed and M. TrabelsiEl, "Detecting opinion spams and fake news using text classification," *Security Commun. Netw.*, vol. 2017, Article ID 9573262, 2017.
- [6] W. Yang, Z. Shi, and X. Luo, "Fake news detection with BERT pre-training," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2019, pp. 1–8.
- [7] A. Conneau et al., "Unsupervised cross-lingual representation learning at scale," in *Proc. ACL*, 2020, pp. 8440–8451.
- [8] Y. Bian et al., "Rumor detection on social media with bi-directional graph convolutional networks," in *Proc. AAAI*, vol. 34, 2020, pp. 549–556.
- [9] K. Popat, S. Mukherjee, A. Yates, and G. Weikum, "CredEye: A credibility lens for analyzing and explaining misinformation," in *Proc. WWW Companion*, 2018, pp. 155–158.
- [10] J. Thorne et al., "FEVER: A large-scale dataset for fact extraction and VERification," in *Proc. NAACL*, 2018, pp. 809–819.
- [11] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.