AntiHoaxID: A Machine Learning and Real-Time Web Verification Based Hoax News Detection System

Nezar Abdilah Prakasa
Master Student of Artificial Intelligence
Department of Computer Science and Electronics
Faculty of Mathematics and Natural Sciences
Universitas Gadjah Mada, Yogyakarta, Indonesia
nezarabdilahprakasa@mail.ugm.ac.id

Abstract: The spread of hoax news in Indonesia keeps rising alongside the growth of digital information consumption, particularly through social media and instant messaging platforms. This study proposes AntiHoaxID, a machine learning based hoax detection system in which the entire inference process runs client-side, without requiring a backend server, and is equipped with a dual-source real-time web verification mechanism and an Explainable AI (XAI) component. The classification model is built using Logistic Regression with TF-IDF feature representation (unigram-bigram, 10,000 features), trained on 4,231 Indonesian-language news samples (3,465 hoax, 766 valid), achieving 82% accuracy, 90% precision, and 88% recall for the hoax class. The trained model is exported from Python/scikit-learn's .pkl format to a 572 KB JSON format so it can be executed directly in the browser through a JavaScript reimplementation of TF-IDF and Logistic Regression. To improve prediction reliability, the system combines the model score with a dual real-time web verification score obtained in parallel from general Google Search and from TurnbackHoax.id, Indonesia's dedicated fact-checking platform (both queried through the Serper.dev API), using an equally weighted interpolation scheme (50% ML, 50% Web). The system also provides prediction explanations based on TF-IDF times model-coefficient feature contribution, and is delivered as a Progressive Web Application (PWA) that can be installed on a device and run offline. Test results show that combining the ML and dual-source Web scores improves detection reliability in ambiguous cases compared to using the ML model alone.

Index Terms: *hoax detection, TF-IDF, Logistic Regression, Explainable AI, Progressive Web Application, client-side machine learning.*

I. INTRODUCTION

A. Background

Indonesia is one of the countries with the highest internet penetration and social media usage rates in Southeast Asia. This condition makes information spread very quickly, but on the other hand it also opens a large gap for the spread of misinformation and disinformation, generally known as hoaxes. Various reports from monitoring institutions such as the Indonesian Anti-Slander Society (Mafindo) and the Ministry of Communication and Information show that the amount of identified hoax content keeps increasing every year, covering health, political, natural disaster, and religious issues.

Manual verification by fact-checkers has limitations in terms of speed and scalability, since the volume of circulating content far exceeds human verification capacity. Therefore, a machine learning based approach for automatic hoax content detection has become a relevant and continuously developing research topic. However, most existing hoax detection systems rely on a server-based architecture, which requires infrastructure cost, server maintenance, and creates dependency on backend service availability.

B. Problem Statement

Based on the background above, several problems become the focus of this research: (1) how to build a lightweight yet accurate Indonesian-language hoax classification model; (2) how the model can be executed without a backend server, so it can be accessed widely without infrastructure cost; (3) how to improve model prediction reliability through a real-time external verification mechanism; and (4) how to present the prediction result transparently so it can be understood and trusted by lay users.

C. Research Objective

This research aims to design and implement a web-based hoax detection system called AntiHoaxID, which combines a lightweight machine learning model (TF-IDF + Logistic Regression) executed entirely on the client side, a real-time web verification mechanism using a search engine, an Explainable AI component, and installability as a Progressive Web Application.

D. Research Contribution

The main contributions of this research are: (1) a full reimplementation of the TF-IDF and Logistic Regression

pipeline in JavaScript so the model can run directly in the browser, with the model compressed into a 572 KB JSON file; (2) a dual-source web verification mechanism that queries general Google Search and TurnbackHoax.id, Indonesia's dedicated fact-checking platform, in parallel, combined with the ML model score using an equal 0.50:0.50 weighting; (3) an Explainable AI mechanism based on feature contribution (TF-IDF times coefficient) presented visually; and (4) a system architecture delivered as a Progressive Web Application that can be installed, run offline, and requires no server hosting cost.

II. RELATED WORK

A. Hoax Detection with Machine Learning

Fake news detection research has been widely conducted using various approaches, ranging from classic methods based on linguistic features to deep learning models based on transformers [6]. Classic approaches such as Naive Bayes, Support Vector Machine (SVM), and Logistic Regression are generally combined with word-frequency based feature representations such as Bag-of-Words or TF-IDF. Although transformer-based approaches such as BERT and IndoBERT show higher performance on various benchmarks, the large model size (hundreds of MB to several GB) makes them impractical to run on the client side or on resource-constrained devices.

B. TF-IDF and Logistic Regression

Term Frequency-Inverse Document Frequency (TF-IDF) [3] is a word-weighting method that represents the importance of a word in a document relative to a document collection (corpus):

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

$$IDF(t) = log\left(\frac{N}{1 + df(t)}\right) + 1$$

where TF(t,d) is the frequency of term t in document d, N is the total number of documents, and df(t) is the number of documents containing term t. This representation is then normalized using the L2 norm so the feature vector has unit length.

Logistic Regression [4] is a linear classification model that maps a linear combination of input features to a probability range [0,1] through a sigmoid function:

$$z = w \cdot x + b$$

$$P(y = 1|x) = \frac{1}{1 + e^{-z}}$$

The combination of TF-IDF and Logistic Regression was chosen in this research because of its lightweight nature, fast execution, ease of interpretation, and because it allows a full reimplementation in JavaScript without depending on large machine learning libraries.

C. Explainable AI (XAI) in Text Classification

Explainable AI is an important requirement in hoax detection systems because model decisions need to be accountable, especially when used by lay users. In linear models such as Logistic Regression with TF-IDF features, interpretation can be done directly through each feature's contribution to the final score. This approach is conceptually aligned with the feature attribution methods more commonly used in the XAI literature such as LIME [5], but with much lower computational complexity due to the linear nature of the model.

D. Progressive Web Application (PWA)

A Progressive Web Application is a web development approach that adopts native application characteristics, such as installability on devices, offline mode support through a service worker [8], and responsive performance. The main PWA components include a web app manifest [7] and a service worker. Applying PWA to a hoax detection system allows the application to be accessed without an app-store installation process, while still being usable under limited connectivity conditions.

III. METHODOLOGY

A. System Architecture

The AntiHoaxID system is designed with a serverless and client-heavy computation philosophy, where the entire ML inference process runs in the user's browser. Fig. 1 shows the overall system architecture.

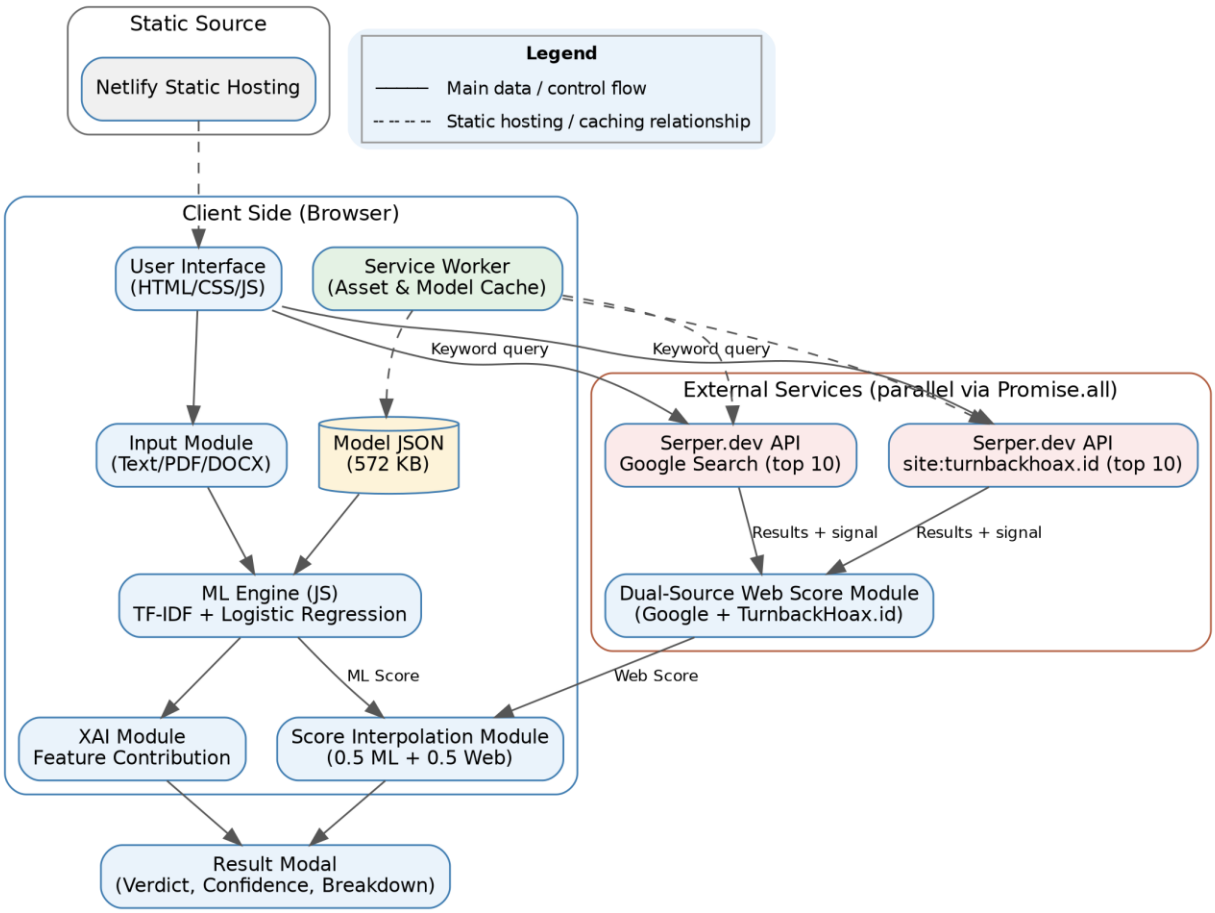

Fig. 1. System architecture of AntiHoaxID, including the dual-source web verification path (Google Search and TurnbackHoax.id).

This architecture consists of four main layers: (1) the user interface and multi-format input module; (2) the JavaScript-based ML engine that performs inference using the model exported to JSON; (3) the dual-source web verification module, which queries general Google Search and TurnbackHoax.id in parallel through the Serper.dev API; and (4) the score interpolation module that combines the ML score and the combined Web score into a single final decision.

B. Analysis Workflow

The process flow from user input to the final result is shown in Fig. 2.

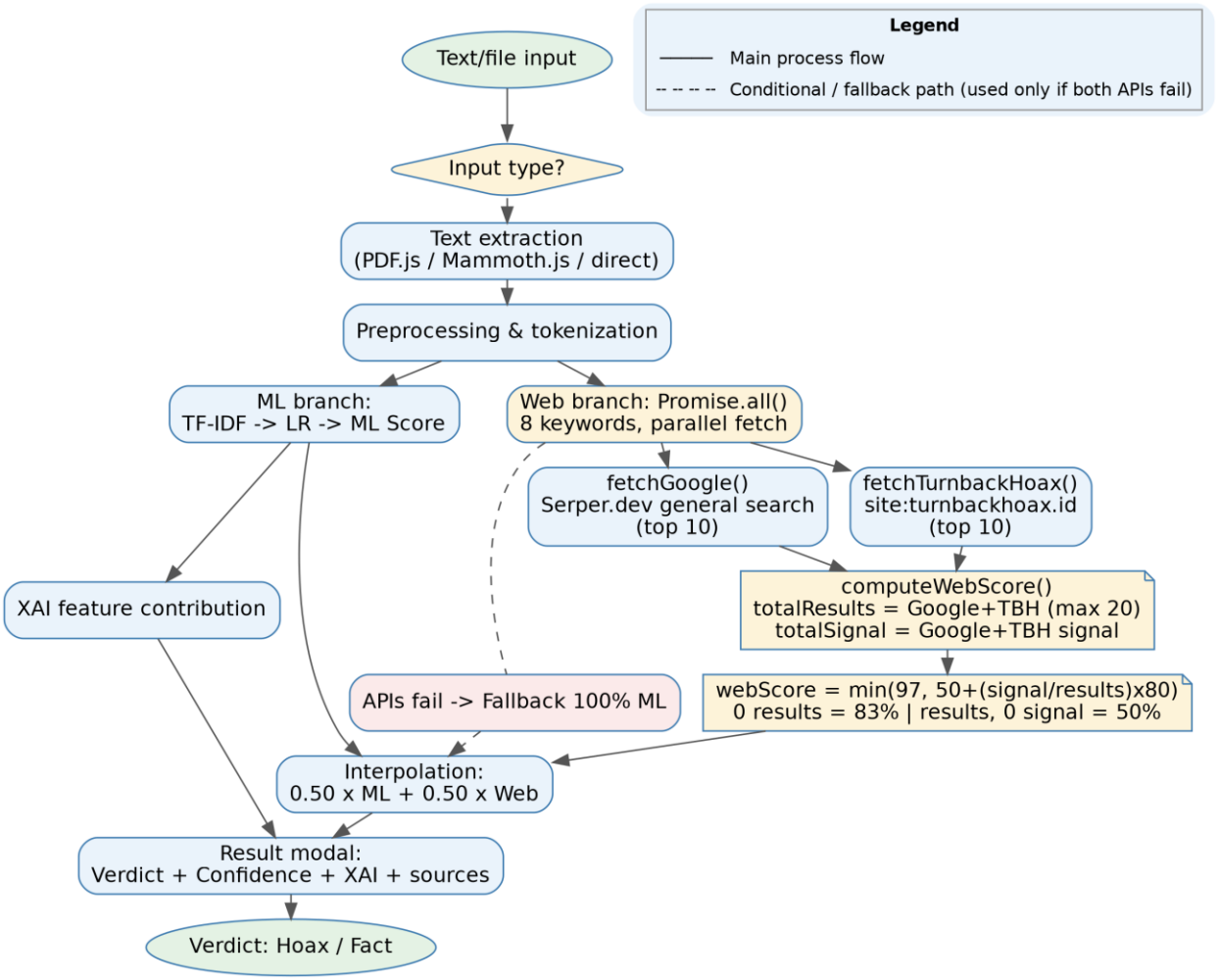

Fig. 2. Flowchart of the AntiHoaxID analysis process, showing the parallel dual-source web verification (Google Search and TurnbackHoax.id) and the combination of ML and Web scores.

C. Preprocessing and Feature Extraction

The preprocessing stage includes removal of non-alphanumeric characters, case-folding, and word tokenization. Feature extraction uses TF-IDF with the configuration in Table 1.

Table 1. TF-IDF vectorizer configuration.

Parameter	Value
analyzer	word
ngram_range	(1, 2)
max_features	10,000
sublinear_tf	False
norm	L2

The use of a unigram-bigram combination aims to capture short phrase context (e.g. “fake news”, “viral hoax”) that often serves as a linguistic marker of hoax content, in addition to single words.

D. Model Training and Export

The model was trained using the scikit-learn library [2] on the “Indonesia False News (Hoax) Dataset” [1], consisting of 4,231 labeled rows (3,465 hoax, 766 valid). The fitted LogisticRegression model and TfidfVectorizer were then saved in .pkl format.

Since Python/scikit-learn cannot be executed directly in the browser, the model components (TF-IDF vocabulary, per-term IDF values, and the Logistic Regression coefficients and intercept) were exported into a single JSON file of 572 KB, enabling inference entirely on the client side without a Python runtime or a model API server.

E. Client-Side ML Implementation

The reimplementation of model inference in JavaScript covers three main computation stages, illustrated in Fig. 3.

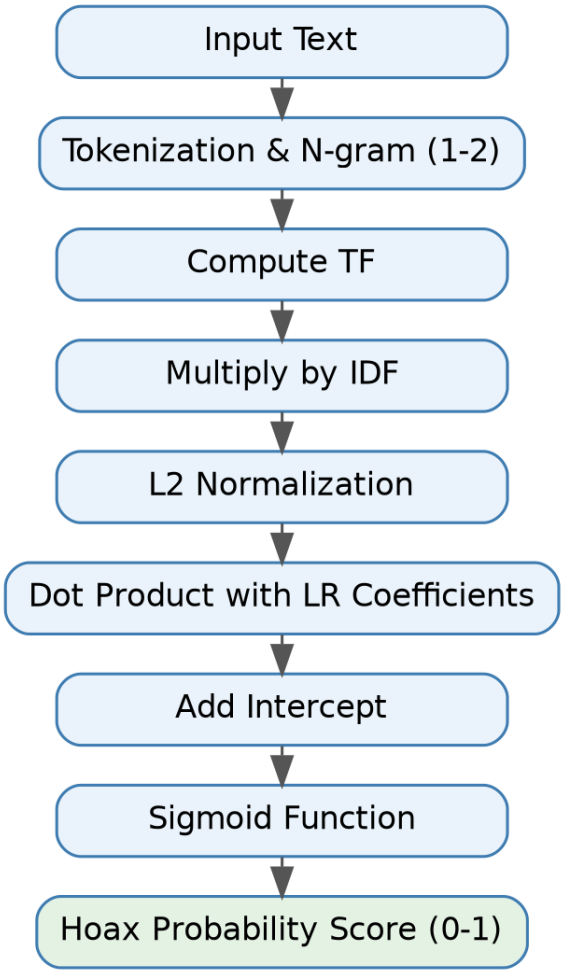

Fig. 3. Client-side ML computation pipeline.

These stages are mathematically identical to the TfidfVectorizer and LogisticRegression.predict_proba() pipeline in scikit-learn, so the inference result in JavaScript is consistent with the result in the Python training environment, as long as the vocabulary order and IDF values are replicated precisely from the original model.

F. Explainable AI Mechanism

The contribution of each feature to the prediction is computed with the formula:

$$contribution(i) = normalized\_tfidf(i) \times coef(i)$$

A negative contribution value indicates that the feature pushes the prediction toward the hoax class, while a positive value pushes it toward the valid class. Features with the highest absolute contribution value are displayed in two separate columns, as illustrated in Fig. 4.

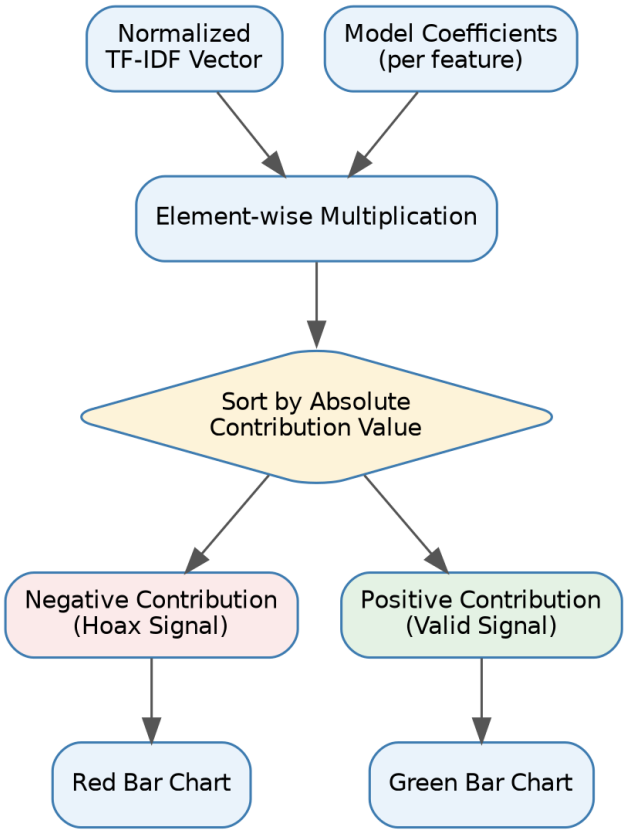

Fig. 4. Explainable AI mechanism based on feature contribution.

G. Dual-Source Web Verification Scoring Scheme

The web verification score was originally computed from a single source, general Google Search (via the Serper.dev API, top 10 results). To improve coverage and reliability for Indonesian-language content, a second, dedicated source was added: TurnbackHoax.id, Indonesia's official fact-checking platform. The TurnbackHoax.id source is implemented as a Serper.dev query restricted with a site operator (site:turnbackhoax.id followed by the extracted keywords), also retrieving the top 10 results. Because every result returned from TurnbackHoax.id already originates from a dedicated debunking platform, all of its results are automatically classified as a strong hoax signal (+2 points per result), unlike Google results, which are scored condition by condition. Both searches are executed in parallel using Promise.all to minimize latency. The scoring scheme is shown in Table 2.

Table 2. Dual-source web verification scoring scheme (Google Search and TurnbackHoax.id).

Condition	Effect on Score
Google result: fact-checker domain (turnbackhoax, cekfakta, mafindo)	+2 per result
Google result: debunking word in title/snippet	+1 per result
Google result: mainstream media, no debunking indication	Neutral

Condition	Effect on Score
TurnbackHoax.id result (any result from this source)	+2 per result
Both sources return 0 results combined	Penalty to 83%
Combined results found, but 0 total debunking signal	Neutral to 50%

The keywords used for both searches are the eight most significant words resulting from feature extraction of the input text. The combined score is computed by function computeWebScore(), which aggregates the total number of results and the total signal points from both sources:

$$totalResults = Google\ results + TurnbackHoax\ results\ \ (max.20)$$

$$totalSignal = Google\ signal + TurnbackHoax\ signal$$

$$webScore = min\left(97,\quad 50 + \frac{totalSignal}{totalResults} \times 80\right)$$

If both sources return zero search results, the system applies a penalty score of 83% (a stronger indication of hoax than the previous single-source scheme, since the complete absence of any indexed discussion from two independent sources is treated as more suspicious). If combined results exist but carry zero debunking signal, the score remains neutral at 50%.

H. Final Score Interpolation

The final score is produced through a weighted interpolation between the ML score and the combined dual-source Web score:

$$Final\ Hoax\ Score = 0.50 \times ML\_Score + 0.50 \times Web\_Score$$

The weighting was revised from the original 0.65:0.35 (ML-dominant) scheme to an equal 0.50:0.50 scheme. This change is justified by the improved quality and coverage of the web signal following the introduction of dual-source verification: TurnbackHoax.id supplies highly specific and reliable signals for documented Indonesian hoax content, so the web evidence is no longer considered secondary to the statistical ML model. An equal weighting reflects comparable trust between the statistical approach (ML) and the evidence-based approach (web verification), and reduces the influence of ML bias toward patterns learned from historical training data, which may not cover newly emerging hoaxes. If both the Google and TurnbackHoax.id requests fail, the system automatically uses the ML score fully (100% weight) as a fallback mechanism.

IV. IMPLEMENTATION

A. User Interface

The AntiHoaxID interface is designed with a minimalist approach using a sky-blue color palette and Inter typography (Google Fonts). The main interface consists of an input area (PDF/DOCX upload or direct text paste), an analysis button, and a fullscreen result modal that appears once the inference process is complete, as summarized in Table 3.

Table 3. Interface components and their functions.

Component	Function
Upload/Text Input Area	Accepts PDF, DOCX, or direct text input (max. 50,000 characters, max. file size 10 MB)
Progress Indicator	Shows text extraction and inference status
Result Verdict Modal	Shows Hoax/Fact label with an animated confidence bar
Score Breakdown Panel	Shows the ML score, Web score, and final score breakdown
XAI Panel	Shows hoax-triggering vs valid-indicating words/phrases with a bar chart
Search Result Panel	Shows two sections: top 10 Google Search results and top 10 TurnbackHoax.id results, each with signal badges
PWA Install Banner	Encourages the user to install the app on the device

B. Approach Comparison

Table 4 compares AntiHoaxID with other commonly used hoax detection approaches.

Table 4. Comparison of AntiHoaxID with other common approaches.

Aspect	Deep Learning (BERT/IndoBERT)	Classic Server-Based	AntiHoaxID (Client-Side)
Model size	100+ MB to several GB	Varies, on server	572 KB (JSON)
Server requirement	Yes (GPU recommended)	Yes	No
Hosting cost	High	Medium-High	Low (static hosting)
Offline capability	No	No	Yes (Service Worker)
Interpretability	Low (needs extra XAI)	Varies	High (linear, native)
Real-time external verification	Rarely integrated	Varies	Integrated (dual-source: Google +

Aspect	Deep Learning (BERT/IndoBERT)	Classic Server-Based	AntiHoaxID (Client-Side)
			TurnbackHoax.id via Serper.dev)
Inference latency	Depends on server & network	Depends on server & network	Instant (local browser)

C. PWA Architecture

Fig. 5 shows the Progressive Web Application architecture of the system, covering the Web App Manifest, Service Worker, and the install experience flow.

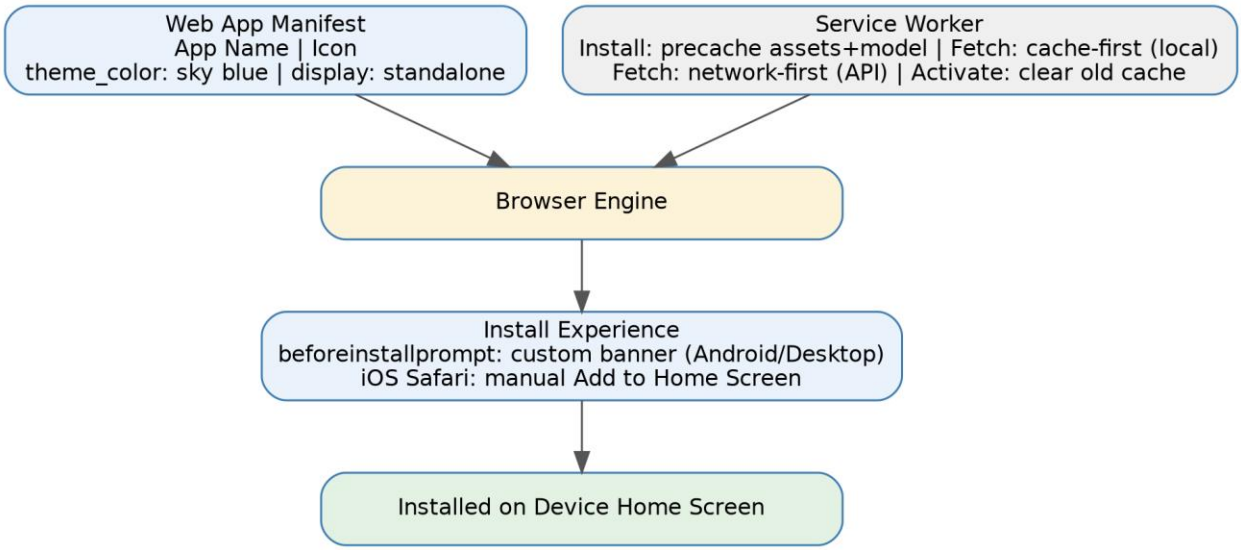

Fig. 5. Progressive Web Application architecture of AntiHoaxID.

The caching strategy applied differentiates the treatment between local (static) assets and requests to the external API: local assets (model JSON, HTML, CSS, JavaScript, PDF.js, Mammoth.js) use a cache-first strategy so the application can still load the interface and run ML inference even offline. In contrast, requests to the Serper.dev API use a network-first strategy, with an automatic fallback to the full ML score if the network is unavailable.

D. Dual Web Verification Implementation

The dual-source web verification is implemented through two asynchronous functions, fetchGoogle() and fetchTurnbackHoax(), both calling the Serper.dev API with the same set of eight extracted keywords. fetchGoogle() issues a general web search request, while fetchTurnbackHoax() appends a site restriction operator (site:turnbackhoax.id) to the query so that only results from Indonesia's dedicated fact-checking platform are returned. Both requests are dispatched together using Promise.all(), so the two searches run concurrently rather than sequentially, keeping the added TurnbackHoax.id lookup from increasing the system's overall response latency.

Once both requests resolve, the function computeWebScore() aggregates their outputs: the total result count (totalResults, capped at 20) and the total signal points (totalSignal) are summed across both sources, following the scoring rules in Table 2, and passed through the formula introduced in Section III-G to produce the final Web score. If either request fails, the module falls back to whichever source is available; if both fail, the system falls back to the ML score alone, as described in Section III-H.

V. RESULTS AND EVALUATION

A. Model Evaluation Metrics

Table 5. Logistic Regression model evaluation results.

Metric	Value
Overall accuracy	82%
Precision (Hoax class)	90%
Recall (Hoax class)	88%

The high precision value for the hoax class (90%) indicates that when the model predicts a text as a hoax, that prediction is very likely correct, thereby minimizing the risk of false alarms

against valid news. A recall of 88% shows that the model captures most of the actual hoax cases, although the class distribution imbalance in the training data (3,465 hoax versus 766 valid) needs attention when interpreting the model's generalization.

B. Test Case Analysis

Table 6 summarizes several test-case scenarios that illustrate how the ML score and the Web score interact in the final interpolation.

Table 6. Illustration of test-case scenarios (values are illustrative to explain the score interpolation mechanism).

Scenario	ML Score	Web Signal	Web Score	Final Score	Verdict
Strong hoax linguistic pattern + debunking matches from Google and TurnbackHoax.id	High	Fact-checker + TurnbackHoax (+2 each)	High	High	Hoax
Neutral/factual text + no relevant results from either source	Low-Medium	0 results (both sources)	83% (penalty)	Medium	Depends on threshold
Ambiguous text + results from mainstream media without debunking	Medium	Neutral	50%	Medium	Depends on threshold
Weak hoax pattern + no debunking signal	Low	Neutral	50%	Low-Medium	Fact/needs further verification

C. Impact of Dual-Source Verification and Weight Rebalancing

The addition of TurnbackHoax.id as a second, dedicated verification source is expected to improve detection reliability in two complementary ways. First, it increases signal coverage for hoax content that has already circulated and been documented in Indonesia: because TurnbackHoax.id is a curated fact-checking database, any match returned from it is treated as a high-confidence hoax indicator (+2 per result), whereas general Google Search may surface a mix of mainstream reporting, discussion forums, or unrelated pages for the same keywords. Second, running the Google and TurnbackHoax.id queries in parallel through Promise.all() means the additional source does not introduce extra sequential latency, so the coverage gain is obtained without a proportional cost to response time.

The rebalancing of the interpolation weight from 0.65:0.35 to an equal 0.50:0.50 (ML:Web) is a direct consequence of this improved web-signal quality. Under the previous scheme, the ML score was weighted more heavily because the single Google-based web signal was considered a supplementary, less domain-specific context. With TurnbackHoax.id contributing highly specific and reliable signals for Indonesian hoax content, the combined web evidence is no longer secondary to the statistical model; an equal weighting lets a strong dual-source debunking signal shift an ambiguous ML prediction more decisively, while also reducing the system's reliance on ML patterns learned only from historical training data, which may not reflect newly emerging hoax narratives.

VI. DISCUSSION

A. System Advantages

Some of the main advantages of AntiHoaxID compared to conventional approaches include: (1) no server infrastructure cost, since all ML inference runs on the client and the application is hosted as static files on Netlify; (2) offline usability thanks to the Service Worker caching strategy; (3) decision transparency through the linear feature-contribution based XAI component; and (4) improved reliability through real-time external contextual verification.

B. System Limitations

Several limitations should be noted: (1) the class distribution imbalance in the training dataset (hoax to valid ratio of about 4.5:1) may cause a model bias toward the majority class prediction; (2) the dataset size (4,231 samples) is relatively limited compared to large-scale hoax detection benchmarks; (3) dependency on third-party services (Serper.dev for Google Search, and the continued availability and update frequency of the TurnbackHoax.id database) for the web verification component; (4) the TF-IDF and Logistic Regression based model has limitations in capturing deep semantic context compared to transformer-based models; and (5) the current dual-source web signal scoring scheme, including the 83% penalty threshold and the 0.50:0.50 interpolation weight, is determined heuristically and has not been optimized through large-scale empirical validation.

C. Practical Implications

The proposed client-side architecture has the potential to become an alternative hoax detection solution that can be accessed widely, especially by individuals or organizations with limited resources to manage server infrastructure. This approach is also relevant for digital literacy education scenarios, where the transparency of the XAI mechanism can be used as a learning aid regarding the linguistic patterns of hoax content.

VII. CONCLUSION AND FUTURE WORK

A. Conclusion

This research successfully designed and implemented AntiHoaxID, an Indonesian-language hoax detection system that combines a TF-IDF based Logistic Regression model with dual-source real-time web verification, combining general Google Search and TurnbackHoax.id, Indonesia's dedicated fact-checking platform, queried in parallel and interpolated with the ML score using an equal 0.50:0.50 weighting. The system is accompanied by an Explainable AI component, and is delivered as a Progressive Web Application that runs entirely on the client side. The model trained on 4,231 data samples achieved 82% accuracy, 90% precision, and 88% recall for the hoax class. The reimplementation of the ML pipeline in JavaScript allows the system to run without a backend server, with very low infrastructure cost through static hosting.

B. Suggestions for Future Development

For future development, it is suggested to: (1) perform thorough quantitative validation of the Web scoring scheme using a larger and more representative labeled test dataset; (2) address the class imbalance through resampling or class-weighting techniques; (3) explore adding datasets from other sources; (4) consider alternative models that remain compact yet capture semantic context better without sacrificing client-side execution capability; and (5) conduct usability testing of the XAI component to ensure the explanation provided genuinely improves lay users' understanding and trust.

REFERENCES

[1] M. Ghazi Muharam, "Indonesia False News (Hoax) Dataset," Kaggle, [Online dataset].

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

[3] G. Salton and C. Buckley, "Term-weighting Approaches in Automatic Text Retrieval," Information Processing & Management, vol. 24, no. 5, pp. 513-523, 1988.

[4] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, Applied Logistic Regression, 3rd ed. Hoboken, NJ, USA: Wiley, 2013.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You? Explaining the Predictions of Any Classifier," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135-1144.

[6] X. Zhou and R. Zafarani, "A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities," ACM Computing Surveys, vol. 53, no. 5, pp. 1-40, 2020.

[7] W3C, "Web App Manifest," W3C Working Draft.

[8] Google Developers, "Service Workers: An Introduction," Chrome Developers Documentation.

[9] Indonesian Anti-Slander Society (Mafindo), Fact-Checking Report, [Online resource].

[10] TurnbackHoax.id, Mafindo Fact-Checking Database, [Online resource].

[11] Serper.dev, "Serper API Documentation."