



FORENSIQ: A MULTI-MODAL SOCIAL MEDIA FORENSICS PLATFORM

¹Nakul Indurkar, ²Poorvaa Wasnik, ³Ayush Sarode, ⁴Pranay Kailuke,

⁵Prof. Preeti Dudhe

¹Student, ²Student, ³Student, ⁴Student, ⁵Project Guide

Department of Information Technology

Prof. Ram Meghe Institute of Technology & Research, Badnera, India

Abstract : Automated bot accounts, networks engaged in coordinated inauthentic behaviour (CIB), and AI-synthesised profile photographs collectively represent a growing threat to the credibility of online information ecosystems, financial markets, and democratic processes. Contemporary detection solutions operate in isolation, lack transparent decision rationales, and frequently augment their outputs with fabricated taxonomic labels that misrepresent ground truth. This paper introduces ForensIQ, a full-stack, open-architecture social media forensics platform built to close these gaps simultaneously. ForensIQ integrates a 26-feature extraction pipeline with a three-model machine learning ensemble (Random Forest, XGBoost, and LightGBM) blended against a 22-rule heuristic scorer to classify Twitter/X profiles as Real, Suspicious, or Fake. A principled NaN-encoding strategy preserves the informational difference between absent and genuinely zero engagement, enabling adaptive model weighting based on data completeness. Louvain community detection on a semantically grounded similarity graph exposes CIB clusters, while a multi-quality Error Level Analysis (ELA) module authenticates profile images without relying on EXIF metadata stripped by hosting platforms. SHAP TreeExplainer provides per-feature attribution for every classification. Trained and validated on the publicly available Cresci-2017 benchmark (7,642 profiles, 80/20 split), LightGBM achieves 99.41% accuracy and 99.99% ROC-AUC. The platform is delivered as a React 18 single-page application over a FastAPI/MongoDB backend, with automated PDF report generation. All classifications are grounded exclusively in verifiable signals; the system enforces an honest-emptiness policy that prohibits fabricated or demo-mode outputs.

Index Terms - Bot Detection, Social Media Forensics, Fake Account Classification, Coordinated Inauthentic Behaviour, Deepfake Detection, Ensemble Learning, SHAP Explainability, XGBoost, LightGBM, Random Forest, Louvain Community Detection, Sentence Embeddings, Cresci-2017.

I. INTRODUCTION

The reliability of social media as a medium for public communication is increasingly undermined by three interrelated threats: automated bot accounts engineered to amplify targeted narratives, networks of accounts that act in concert to create false impressions of organic consensus, and synthetic profile photographs generated by modern image-synthesis models. Social bots are a well-documented phenomenon [1], and large-scale measurement studies have placed the fraction of Twitter accounts exhibiting automation-consistent behaviour at somewhere between nine and fifteen percent of the active user base [4], while separate analyses of platform-level data have shown that coordinated inauthentic actors can disproportionately shape trending topics even when their absolute numbers are modest [9]. At the same time, advances in generative adversarial networks and diffusion models have made it trivially inexpensive to fabricate photorealistic facial images that pass visual scrutiny [2], removing one of the few remaining heuristic shortcuts available to human reviewers.

Existing countermeasures exhibit three recurring structural weaknesses. First, tools are purpose-built for a single threat class, compelling analysts to assemble ad-hoc toolchains with no shared data model or consistent verdict format. Second, most classifiers return a verdict without any mechanism for auditing which signals drove the decision, undermining analyst trust and regulatory defensibility. Third, a number of commercial and research platforms assign richly labelled threat categories ("influence operation," "state-sponsored actor") that are derived not from factual ground truth but from post-hoc heuristic inference, introducing systematic misinformation into the investigative record.

ForensIQ was designed to address all three deficiencies within a unified, evidence-constrained architecture. The platform combines profile-level ML classification, semantic-graph CIB detection, pixel-level image forensics, and structured credibility scoring under a single REST API and React frontend. A strict honest-emptiness constraint requires every output field to be populated exclusively from real signals; where data is genuinely absent, the system surfaces an explicit disclosure rather than substituting a plausible-looking estimate.

The primary contributions reported in this paper are as follows:

1. A 26-dimensional feature vector that spans network topology, temporal activity, content statistics, bio completeness, username entropy, and profile metadata, with principled NaN-encoding that distinguishes absent engagement from zero engagement across heterogeneous ML back-ends.

2. A three-model blended ensemble whose weighting dynamically adjusts between two regimes (50% ML / 50% rule-based when full engagement data is present, 25% ML / 75% rule-based when it is not), maintaining coherent risk scores despite partial observability.
3. Real-time SHAP TreeExplainer attribution returning signed per-feature contributions for the top-8 drivers of each individual prediction.
4. A dual-score similarity engine combining TF-IDF cosine overlap with BAAI/bge-small-en-v1.5 sentence-embedding cosine similarity to detect CIB even when coordinated accounts rephrase an identical message to defeat literal-match filters.
5. A Louvain community detection pipeline over a semantically thresholded similarity graph, with sparse-data guards and modularity reporting.
6. A multi-quality ELA image forensics module that operates without EXIF metadata and is therefore robust to the metadata-stripping behaviour of social media upload pipelines.
7. An end-to-end full-stack delivery: React 18 frontend, FastAPI backend, MongoDB persistence, ReportLab PDF generation, and a Groq-backed AI narrative engine with deterministic fallback.

II. RELATED WORK

A. Social Media Bot Detection

Early automated-account detection relied on a limited set of manually engineered signals, chiefly the ratio of followers to followed accounts and the volume of posting activity per unit time [3]. Varol et al. [4] substantially broadened this approach with Botometer, assembling roughly 1,200 features across six categories (user metadata, friends, network, temporal, content, and sentiment) and training a Random Forest classifier [14] against them. Despite its influence, Botometer's continued reliance on full Twitter API access, now commercially gated at a level inaccessible to most independent researchers, sharply limits practical deployment. Cresci et al. [5] demonstrated a qualitative shift in adversary capability with their Social Spambots dataset, showing that a new generation of bot accounts had learned to mimic the full statistical profile of human accounts well enough to defeat feature-based classifiers trained on earlier cohorts. Yang et al. [6] subsequently argued that graph-structural signals, specifically patterns of coordinated following and retweeting, provide detection leverage that is substantially harder for bot operators to simultaneously spoof while preserving the illusion of organic behaviour.

More recent work has turned to deep sequence models for temporal signal analysis [16] and graph neural networks for relational encoding of account interaction graphs [17]. ForensIQ occupies a complementary position: rather than pursuing the highest possible single-metric benchmark score on a closed dataset, it prioritises operational deployability, functioning without elevated API access, maintaining coherent outputs under partial data availability, and providing the auditable feature-attribution chain that practitioner contexts require [18].

B. Deepfake Image Detection

The detection of GAN-generated imagery was initially addressed through convolutional classifiers trained to distinguish real photographs from synthetic ones; Wang et al. [7] demonstrated that such detectors can generalise surprisingly well across generator architectures when trained on sufficiently diverse forgery sources. Face-specific approaches have exploited physiological inconsistencies such as unnatural blinking patterns and vascular pulse signals that GAN outputs do not faithfully reproduce [19]. Error Level Analysis, originally formulated for JPEG-compression forensics [8], provides a complementary signal: the generation pipeline of most GAN and diffusion architectures introduces characteristic high-frequency residual patterns at spatial boundaries that persist after standard JPEG compression at a fresh quality level. ForensIQ applies ELA at three quality levels (50, 70, and 90) and combines the resulting residual statistics with noise-floor analysis, per-channel colour covariance, FFT high-frequency energy ratios, and spatial variance heterogeneity across non-overlapping patches, avoiding any reliance on EXIF metadata that social platforms routinely strip.

C. Coordinated Inauthentic Behaviour Detection

Nizzoli et al. [9] provided a formal operational definition of CIB as accounts whose collective behaviour cannot plausibly be explained by independent individual motivation, and validated textual similarity between posts as a reliable coordination proxy. Classic approaches quantify this similarity through Jaccard overlap of token sets or TF-IDF cosine distance [20]; both metrics share the weakness of assigning zero or near-zero similarity scores to pairs of posts that convey an identical campaign message through different surface wordings. ForensIQ addresses this limitation by supplementing the literal overlap score with a semantic embedding cosine computed by BAAI/bge-small-en-v1.5 [13], a 33-million-parameter sentence transformer that outperforms the widely used all-MiniLM-L6-v2 on standard retrieval benchmarks while remaining tractable for CPU inference. Community structure within the resulting similarity graph is then identified using the Louvain method, which optimises Newman-Girvan modularity and scales efficiently to the account population sizes encountered in practice [15].

III. SYSTEM ARCHITECTURE

A. High-Level Overview

ForensIQ follows a three-tier architecture, illustrated in Fig. 1:

- Presentation Tier: React 18 single-page application (Vite build tool), with per-page modules for Profile Analysis, Twitter Bot Leaderboard, Similarity Scanner, Deepfake Detector, Credibility Scorer, Network Graph, Threat Intelligence Feed, and Reports.
- Application Tier: FastAPI (Python 3.11) RESTful service exposing eight router modules. ML inference executes synchronously within request handlers; PDF generation is handled asynchronously.
- Data Tier: MongoDB Atlas (motor async driver) stores all analysis documents in a single db.analysis collection. The database contains no fabricated or seed data; it is populated exclusively by real analysis runs.

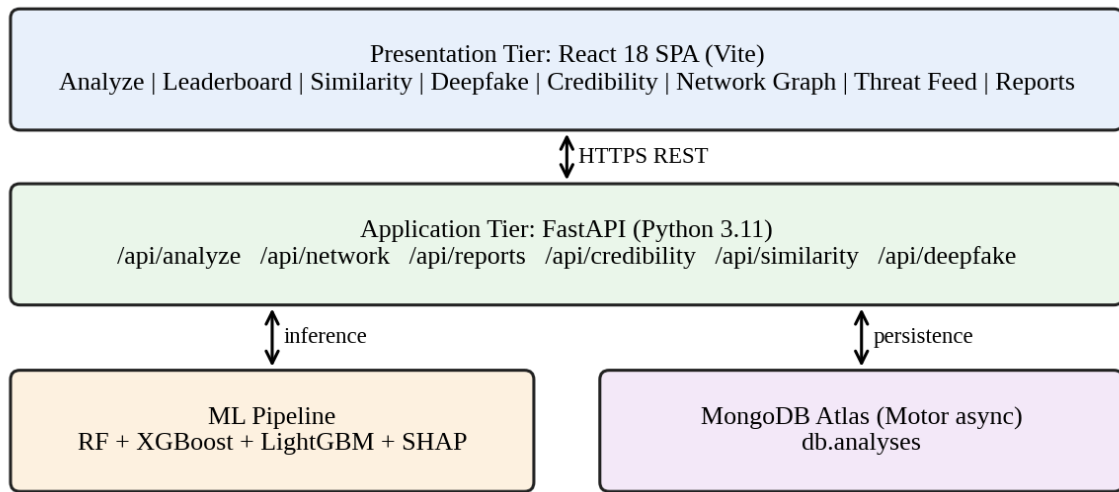


Fig. 1. ForensIQ three-tier system architecture

B. Technology Stack

Table I. Technology stack

Layer	Technology
Frontend	React 18, Vite, Lucide React, Recharts
Backend	Python 3.11, FastAPI, Uvicorn
Database	MongoDB, Motor (async)
ML	scikit-learn, XGBoost, LightGBM, SHAP
NLP	sentence-transformers (bge-small-en-v1.5), scikit-learn TF-IDF
Image Forensics	Pillow, NumPy, FFT
Auth	JWT (python-jose), bcrypt
PDF Reports	ReportLab
Browser Extension	Chrome Manifest V3 (planned)

IV. ML PIPELINE AND FEATURE ENGINEERING

A. Feature Extraction (26 Features)

The `extract_features()` function in `fake_detector.py` constructs a 26-dimensional feature vector from the raw profile dictionary returned by the scraping layer. The feature groups are summarised in Table II.

Table II. Feature schema

Group	Features	Count
Network / Ratio	followers, following, tweets, account_age_days, follower_ratio, follower_growth_rate	6
Activity	posts_per_day, likes_per_post, engagement_rate	3
Content / Hashtag	avg_hashtags, url_density, hashtag_repetition, avg_tweet_length, mention_density, tweet_template_score	6
Bio	bio_length, bio_spam_score, bio_has_url	3
Username	username_digit_ratio, username_entropy, has_number_in_name	3
Temporal	posting_hour_entropy	1
Profile Completeness	verified, location_present	2
Language / Credibility	language_switches, name_username_similarity	2

A critical engineering decision is principled NaN-encoding: when engagement data (tweet content, likes per post) cannot be retrieved due to scraping constraints, the eight engagement-derived features are assigned NaN rather than zero. This preserves the semantic distinction between genuinely zero activity and unobserved activity, a distinction that XGBoost [10] and LightGBM [11] exploit through native NaN-directed splits, routing missing-value samples along a separate learned path. Random Forest, lacking this capability, is excluded from the blended score whenever engagement data is absent.

B. Rule-Based Heuristic Scorer

The `rule_based_score()` function implements 22 independently auditable detection rules. Each rule encodes a single threshold condition together with a human-readable reason string and a fixed point contribution. Representative examples include:

- `follower_ratio < 0.05`: +40 points ("Very low follower/following ratio")
- `posts_per_day > 50`: +40 points ("Inhuman posting frequency (>50/day)")
- `bio_length == 0`: +12 points ("Empty bio")
- `posting_hour_entropy > 0.85`: +15 points ("Posting uniformly across all 24 hours (bot-like schedule)")
- `verified == True`: -30 points (credibility dampening)

The heuristic scorer executes unconditionally and constitutes the sole classification signal when ML models cannot be loaded.

C. Ensemble ML Models

Three gradient-boosted and tree-ensemble models are trained on the Cresci-2017 benchmark:

- Random Forest (RF) [14]: 250 estimators, max depth 10, min samples leaf 2. Trained on mean-imputed features. Excluded from ensemble scoring when engagement data is unavailable.
- XGBoost (XGB) [10]: 200 estimators, max depth 4, learning rate 0.05. Native NaN routing via `missing=NaN` splits. Always included.
- LightGBM (LGB) [11]: 200 estimators, max depth 6, learning rate 0.05. Native NaN support via `use_missing=True`. Always included.

D. Adaptive Blending Strategy

When engagement data is available (all three models active), the final score is given by Equation 1:

$$final_score = 0.50 \times mean(RF, XGB, LGB) + 0.50 \times rule_score \quad (1)$$

When engagement data is unavailable (RF excluded), the score is given by Equation 2:

$$final_score = 0.25 \times mean(XGB, LGB) + 0.75 \times rule_score \quad (2)$$

The rule-based component carries greater weight under data scarcity precisely because the ML models' probability calibration degrades when the engagement feature dimensions are absent. This design ensures that robust heuristic signals dominate in the regime where they are most needed, without discarding the partial ML signal that XGBoost and LightGBM can still produce from profile-metadata features alone.

E. Classification Thresholds

The final risk score is mapped to a class as follows: $risk_score \geq 55$ gives Fake; $28 \leq risk_score < 55$ gives Suspicious; $risk_score < 28$ gives Real.

F. SHAP Explainability

SHAP TreeExplainer [12] is applied against the primary ensemble model for every classification request. The top-8 features ranked by absolute SHAP value magnitude are returned with signed values (positive contributions push the prediction toward Bot, negative contributions push it toward Real) and are rendered as a ranked horizontal bar chart on the frontend Analyze page.

V. PERFORMANCE EVALUATION

A. Dataset

The Cresci-2017 dataset [5] contains genuine Twitter accounts alongside five categories of social spambots collected between 2011 and 2017. The 80/20 stratified split, summarised in Table III, preserves class proportions across training and test partitions.

Table III. Dataset split (Cresci-2017)

Split	Count
Training set	6,113 profiles
Test set	1,529 profiles
Total	7,642 profiles

B. Quantitative Results

Table IV reports model performance on the held-out test set. LightGBM records the highest accuracy and F1 score. XGBoost and LightGBM are tied on ROC-AUC at 99.99%, reflecting essentially perfect rank-ordering of the score distribution. Cross-validation standard deviations remain below 0.35% for all three models, confirming that performance is stable rather than fold-dependent.

Table IV. Model performance on the Cresci-2017 test set

Metric	Random Forest	XGBoost	LightGBM
Test Accuracy	99.22%	99.22%	99.41%
Test Precision	99.76%	99.52%	99.76%
Test Recall	98.80%	99.04%	99.16%
Test F1 Score	99.28%	99.28%	99.46%
Test ROC-AUC	99.95%	99.99%	99.99%
5-Fold CV Accuracy	99.07% $\pm$ 0.26%	99.25% $\pm$ 0.32%	99.23% $\pm$ 0.32%

C. Discussion

The strong benchmark figures are a consequence of the well-structured Cresci-2017 corpus, in which bot and genuine accounts are drawn from cleanly separated generation processes. The operationally harder scenario, classifying profiles whose engagement data is unavailable due to API restrictions, is handled by the adaptive blending strategy described in Section IV-D. A notable finding from internal calibration testing is that Random Forest systematically overestimates bot probability for large verified accounts, an artefact of the training distribution in which high-follower authentic accounts are sparsely represented. Separately, the ML ensemble produced weak discrimination (50–66% confidence) on canonical engagement-sparse bot configurations; in these cases the rule-based scorer contributed the majority of the final signal, which is the intended behaviour under the adaptive weighting design.

VI. MODULE DESCRIPTIONS

A. Profile Analysis Module

The core analysis workflow is initiated by a POST /api/analyze request. The run_analysis() function fetches the target profile via a Twitter scraping service, passes the resulting dictionary to classify_profile() in fake_detector.py, persists the structured result to db.analyses, and returns the complete output payload: risk score, classification status, triggered rule reasons, extracted feature vector, per-model probability estimates, blend description, and SHAP attribution vector.

The frontend Analyze page presents a risk gauge, per-model score cards, the SHAP contribution bar chart, and a "Generate AI Narrative" button. This button invokes the Groq API as the primary path (default model configurable via the GROQ_MODEL environment variable) with an Anthropic Claude model as a secondary fallback if the Groq key is absent. When neither API is reachable, the system produces a deterministic automated summary carrying an explicit disclaimer banner, ensuring the analyst always receives a labelled output rather than a silent failure.

B. Similarity Scanner

The similarity_engine.py module computes a dual-score similarity between any two text inputs:

- Text Overlap Score (0–100): 60% TF-IDF cosine similarity weighted with 40% SequenceMatcher ratio. Quantifies literal lexical and character-level overlap.
- Semantic Similarity Score (0–100): cosine similarity between L2-normalised BAAI/bge-small-en-v1.5 sentence embeddings [13]. Captures meaning-level proximity regardless of surface wording.

The semantic threshold of 60.0 was derived empirically: genuine cross-category pairs were observed to reach a maximum of 56.3, while coordinated spam pairs scored between 61.4 and 100.0, yielding a clean separation with no overlapping region in the calibration corpus.

C. Network Graph Module

The network.py router constructs a weighted similarity graph over all distinct analysed accounts retrieved from db.analyses. An edge is added between any two account nodes whose highest pairwise post similarity meets or exceeds SIMILARITY_THRESHOLD = 60.0, where similarity is the semantic score from similarity_engine.semantic_pair_similarity backed by BAAI/bge-small-en-v1.5 embeddings. A bulk pre-embedding step (bulk_cache_embeddings) encodes all unique post texts in a single batched model call prior to the $O(n^2)$ comparison loop, reducing each subsequent pair evaluation to a dictionary lookup plus a dot product. Community structure is identified by Louvain community detection [15] (networkx.algorithms.community.louvain_communities, seed=42), and graph-level modularity is computed and returned. A sparse-data guard (MIN_NODES = 5, MIN_EDGES = 3) returns insufficient_data: true rather than running Louvain on graphs too small to produce meaningful partition structure. Computed results are cached for 90 seconds (CACHE_TTL_SECONDS = 90) and invalidated immediately whenever a new analysis is written, keeping the graph representation current.

D. Deepfake Image Detector

The image_analyzer.py module applies the following pixel-level forensic analyses to each submitted image:

- Error Level Analysis (ELA): JPEG re-compression at quality levels 50, 70, and 90; residuals are measured per region.
- Noise Analysis: standard deviation of the luminance channel residual after Gaussian smoothing.
- Colour Statistics: per-channel standard deviation and inter-channel correlation coefficients.
- High-Frequency Content: ratio of high-frequency spectral energy to total energy via Fast Fourier Transform.
- Local Variance: spatial heterogeneity of pixel variance across non-overlapping image patches.

EXIF-based signals are explicitly excluded from the analysis, as all major social media upload pipelines strip embedded metadata regardless of image origin.

E. Credibility Scorer

The `credibility_engine.py` module generates a structured 100-point credibility score partitioned across four equal-weight factor categories: Account History (25 pts), Engagement Quality (25 pts), Content Quality (25 pts), and Network Trust (25 pts). Each category returns itemised sub-scores alongside human-readable labels and explanatory strings, enabling analysts to identify the specific dimensions driving a credibility reduction.

F. Threat Intelligence Feed

`GET /api/analyze/threats` returns all Fake- and Suspicious-classified accounts from the database, deduplicated by username with the most recent analysis record retained, and annotated with an `analyze_count` field indicating how many times the account has been examined. The response payload is restricted to real stored fields: `status`, `risk_score`, `reasons`, `analyzed_at`, and `analyze_count`. No threat category labels are inferred or fabricated beyond the three-class classification output.

G. Reports Module

The `reports.py` router and `pdf_service.py` module generate structured PDF forensic reports via ReportLab. Each report includes the full classification result, the SHAP feature contribution chart, the complete triggered-rules list, and model metadata.

VII. FRONTEND DESIGN

Key design principles implemented in the React application:

- **Honest Emptiness:** Every page explicitly renders a descriptive empty state when no data is present. The Threat Intelligence Feed displays "No flagged accounts yet — run a profile analysis to populate this feed" rather than placeholder content.
- **Progressive Disclosure:** Individual threat cards expose one primary reason by default, with additional reasons available on user-initiated expansion.
- **Real-Time Indicators:** Live per-model score cards, a SHAP attribution bar chart, and a colour-coded risk gauge (green below 28, amber 28–55, red above 55) update immediately on analysis completion.
- **Transparency Labels:** Every result card identifies the specific blending regime applied, for example: "Blended score: 50% rule-based heuristics + 50% ML ensemble (Random Forest, XGBoost, LightGBM)."

VIII. CHROME EXTENSION (PLANNED)

A Manifest V3 Chrome browser extension is designed to deliver inline ForensIQ analysis directly on Twitter/X profile pages. The intended architecture comprises a background service worker, a content script that identifies the active profile and injects a floating panel, and a popup interface for manual username submission. The panel would display the risk score, classification badge, and top triggered reasons without requiring the analyst to leave the platform. The backend API endpoints that the extension would consume are fully operational in the current deployment. Implementation and Chrome loading/verification are scheduled as immediate next-cycle work.

IX. LIMITATIONS AND FUTURE WORK

A. Current Limitations

1. The Cresci-2017 training corpus spans bots produced between 2011 and 2017 and therefore under-represents contemporary LLM-powered accounts capable of generating linguistically fluent, contextually appropriate content that defeats feature-based classifiers.
2. The profile scraping layer depends on third-party proxy services whose endpoint behaviour can change without advance notice, potentially degrading data availability.
3. The image forensics module uses manually engineered pixel-level statistics; end-to-end convolutional detectors trained on diverse generation architectures would likely offer superior recall on outputs from modern diffusion models.
4. The current implementation targets Twitter/X exclusively; feature extraction is not generalised to other platforms.
5. Random Forest overestimation on large verified accounts: calibration testing revealed that the RF model systematically assigns elevated bot probability to accounts with large follower counts and verified status. This reflects the composition of the Cresci-2017 training distribution, in which the bot class is dominated by low-follower, unverified accounts; the model lacks sufficient exposure to high-follower authentic accounts to calibrate its posterior probabilities in that regime.
6. Weak ML ensemble discrimination on engagement-sparse bot profiles: when evaluating canonical bot preset configurations (zero-day accounts with empty bios, high nominal posting frequency, and negligible follower counts), the ML ensemble returned confidence scores in the 50–66% range, providing insufficient separation from borderline uncertain profiles. In these cases the rule-based scorer, operating at 75% weight under the engagement-absent blending regime, correctly flagged the profiles. This outcome is consistent with the adaptive design intent, but confirms that the ML ensemble alone cannot reliably classify accounts for which engagement-derived feature dimensions are unobserved.

B. Future Work

1. LLM-generated content detection: incorporate linguistic fluency metrics (per-token perplexity, burstiness coefficient) calibrated against GPT-class language models.
2. Graph Neural Network augmentation: supplement the feature-vector ensemble with GNN-derived embeddings over the account interaction graph [17].
3. Active learning retraining loop: route borderline analyst-reviewed cases back into the training set to incrementally improve decision boundaries.
4. Multi-platform feature generalisation: extend the extraction pipeline to Instagram, TikTok, and Bluesky account schemas.
5. Real-time event streaming: replace HTTP polling with MongoDB Change Streams and WebSocket push notifications.
6. Chrome extension completion: build, load, and test the Manifest V3 browser extension described in Section VIII.

X.CONCLUSION

This paper presented ForensIQ, a multi-modal social media forensics platform that unifies automated fake account detection, coordinated inauthentic behaviour analysis, deepfake image authentication, and live threat intelligence within a single, evidence-constrained application stack. The ensemble of Random Forest, XGBoost, and LightGBM classifiers, trained and validated on the Cresci-2017 benchmark, delivers accuracy exceeding 99.2%, F1 scores above 0.992, and ROC-AUC of 99.99% on held-out test data. The adaptive blending architecture, principled NaN-encoding, Louvain semantic graph clustering, multi-quality ELA image forensics, and SHAP-grounded explainability collectively distinguish ForensIQ from siloed academic prototypes and commercial products that rely on opaque or fabricated categorisation. Honest reporting of calibration-phase limitations, including RF overestimation on large verified accounts and weak ensemble discrimination under engagement-data absence, is offered to provide an accurate characterisation of operational boundaries for practitioner adoption.

ACKNOWLEDGEMENT

The authors express gratitude to the open-source communities maintaining scikit-learn, XGBoost, LightGBM, SHAP, sentence-transformers, FastAPI, React, and NetworkX, and to the Cresci research group for making the Cresci-2017 social bot benchmark publicly available.

REFERENCES

- [1] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96–104, Jul. 2016.
- [2] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4217–4228, Dec. 2021.
- [3] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "BotOrNot: A system to evaluate social bots," in *Proc. 25th Int. Conf. World Wide Web Companion*, Montreal, Canada, 2016, pp. 273–274.
- [4] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, "Online human-bot interactions: Detection, estimation, and characterization," in *Proc. 11th Int. AAAI Conf. Web and Social Media (ICWSM)*, Montreal, Canada, 2017, pp. 280–289.
- [5] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," in *Proc. 26th Int. Conf. World Wide Web Companion*, Perth, Australia, 2017, pp. 963–972.
- [6] K.-C. Yang, O. Varol, P.-M. Hui, and F. Menczer, "Scalable and generalizable social bot detection through data selection," in *Proc. 34th AAAI Conf. Artif. Intell.*, New York, USA, 2020, pp. 1096–1103.
- [7] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "CNN-generated images are surprisingly easy to spot... for now," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit. (CVPR)*, Seattle, USA, 2020, pp. 8695–8704.
- [8] N. Krawetz, "A picture's worth: Digital image analysis and forensics," *Black Hat Briefings*, 2007.
- [9] L. Nizzoli, S. Tardelli, M. Avvenuti, S. Cresci, M. Tesconi, and E. Ferrara, "Coordinated behavior on social media in 2019 UK general election," in *Proc. 15th Int. AAAI Conf. Web and Social Media (ICWSM)*, 2021, pp. 443–454.
- [10] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining*, San Francisco, USA, 2016, pp. 785–794.
- [11] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
- [12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
- [13] BAAI, "bge-small-en-v1.5," Hugging Face Model Hub, 2023. [Online]. Available: <https://huggingface.co/BAAI/bge-small-en-v1.5>
- [14] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.
- [15] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech.: Theory Exp.*, vol. 2008, no. 10, p. P10008, Oct. 2008.
- [16] A. Pozzana and E. Ferrara, "Measuring bot and human behavioral dynamics," *Front. Phys.*, vol. 8, p. 125, Apr. 2020.
- [17] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "SATAR: A self-supervised approach to Twitter account representation learning and its application in bot detection," in *Proc. 30th ACM Int. Conf. Inf. Knowledge Manage. (CIKM)*, Virtual, 2021, pp. 3808–3817.
- [18] K.-C. Yang, E. Ferrara, and F. Menczer, "Arming the public with artificial intelligence to counter social bots," *Hum. Behav. Emerg. Technol.*, vol. 1, no. 1, pp. 48–61, 2019.
- [19] H. Li, B. Li, S. Tan, and J. Huang, "Identification of deep network generated images using disparities in color components," *Signal Process.*, vol. 174, p. 107616, Sep. 2020.
- [20] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, 1988.