Explainable AI Based Smart Job Scam Detection System Using Hybrid NLP & Behavioural Features

Senthuriya.C,Sindhuja.S,Mrs.N.kanagadurga M.E.,
Department of Computer Science and Engineering
E.G.S.Pillay Engineering College, Nagapattinam, India
senthuriyachandramohan@gmail.com, sindhuja.s12042006@gmail.com

How to Cite this Article:
Senthuriya.C, , Sindhuja.S, & N.kanagadurga, (2026). Explainable AI Based Smart Job Scam Detection System Using Hybrid NLP & Behavioural Features. International Journal of Creative and Open Research in Engineering and Management, <i>02</i>(05). https://doi.org/10.55041/ijcope.v2i5.747

https://doi.org/10.55041/ijcope.v2i5.747

Abstract — The proliferation of online recruitment platforms has correspondingly amplified the incidence of fraudulent job postings, posing grave risks to employment seekers in terms of financial exploitation, identity theft, and psychological harm. Conventional approaches to identifying deceptive job advertisements have relied predominantly on heuristic keyword matching and rule-based filtering, which exhibit substantial deficiencies with respect to adaptability, scalability, and explainability. This paper presents an Explainable Artificial Intelligence (XAI)-powered Smart Job Scam Detection System that integrates Hybrid Natural Language Processing (NLP) techniques with behavioural feature engineering to classify job postings as genuine or fraudulent with high accuracy. The proposed system employs Term Frequency–Inverse Document Frequency (TF-IDF) vectorization, suspicious keyword detection, recruiter email domain verification, company profile consistency analysis, and ensemble machine learning classification using XGBoost and Logistic Regression. The Explainability module leverages SHapley Additive exPlanations (SHAP) to provide transparent, human-interpretable reasoning behind every prediction. Additional functionalities include Optical Character Recognition (OCR)-based screenshot analysis, batch CSV prediction, scam probability scoring, risk level classification, and an interactive visualization dashboard. Experimental evaluation on the Kaggle Fake Job Postings dataset demonstrates an overall accuracy of 97.4%, with a Precision of 96.8%, Recall of 95.9%, and F1-Score of 96.3%, outperforming baseline methods. The system presents a robust, transparent, and scalable solution to combat online recruitment fraud.

Keywords — Explainable AI, Fake Job Detection, NLP, TF-IDF, XGBoost, SHAP, OCR, Behavioural Feature Engineering, Fraud Detection, Cybersecurity.

I. INTRODUCTION

The rapid digitalization of the global employment ecosystem has fundamentally transformed how individuals pursue career opportunities. Online job portals such as LinkedIn, Indeed, Glassdoor, Monster, and Naukri have democratized access to employment by enabling millions of job seekers and recruiters to connect instantaneously across geographical boundaries. According to the International Labour Organization (ILO), approximately 3.3 billion people constitute the global workforce, and a significant and growing proportion relies on digital platforms to search for and apply to jobs. However, this shift toward digital recruitment has simultaneously created fertile ground for malicious actors seeking to exploit the vulnerability of earnest job seekers through fraudulent job postings.

Online recruitment fraud manifests in numerous forms, including fake job advertisements that solicit upfront fees, phishing campaigns that harvest personal and financial information, impersonation of legitimate organizations, and advance-fee scams that promise nonexistent employment in exchange for registration charges or equipment deposits. The Federal Trade Commission (FTC) reported that employment scam losses in the United States alone exceeded USD 68 million in 2021, representing a 53% year-over-year increase. In developing economies, where unemployment rates are higher and job seekers may be less digitally literate, the impact of such scams is disproportionately severe. Victims often suffer not only financial loss but also significant psychological trauma, eroded trust in digital platforms, and in extreme cases, identity theft with long-term credit implications.

Fraudulent job postings are characterized by a constellation of indicators: inflated salary promises, vague or implausible job descriptions, non-corporate email domains for recruiter communications, absence of verifiable company information, demands for personal documentation upfront, and linguistic patterns characteristic of machine-generated or hastily composed text. Despite these observable signals, manual identification of scam postings remains impractical at scale given the sheer volume of job listings published daily. Automated detection systems therefore represent an imperative technological solution.

Traditional fraud detection approaches in this domain have relied on rule-based systems and simple keyword filtering. While these methods are computationally inexpensive, they are inherently brittle: they fail to generalize to new scam variants, cannot process contextual information embedded within natural language, and produce binary outputs devoid of any explanatory context. Machine learning approaches have partially addressed these limitations by learning discriminative patterns from labeled datasets; however, many published models function as black boxes, offering predictions without interpretable justification. This opacity is particularly problematic in a consumer-facing application context where end users must be able to trust and understand the system's recommendations.

The emergence of Explainable AI (XAI) as a subfield of artificial intelligence has addressed this opacity concern by developing techniques that elucidate the decision-making processes of complex models. Methods such as SHAP (SHapley Additive exPlanations), LIME (Local Interpretable Model-agnostic Explanations), and attention mechanisms in neural networks enable practitioners and end users alike to understand which features contributed most substantially to a given prediction. Integrating explainability into a fraud detection system not only enhances user trust but also facilitates iterative model improvement by exposing potential biases or overfitting issues.

This paper proposes a comprehensive Explainable AI-based Smart Job Scam Detection System that addresses the aforementioned limitations through a multi-faceted approach. The system combines Hybrid NLP techniques encompassing tokenization, stop word removal, TF-IDF vectorization, and suspicious keyword analysis with behavioural feature engineering that examines recruiter email domains, company profile consistency, salary plausibility, and structural anomalies in job descriptions. The classification engine employs an ensemble of XGBoost and Logistic Regression models, while the explainability layer integrates SHAP visualizations to provide transparent, feature-level justification for each prediction. An additional OCR module enables users to upload screenshots of job postings for automated text extraction and analysis, extending the system's applicability to social media and messaging platform scams.

The primary objectives of this research are: (i) to develop a highly accurate and generalizable machine learning model for fake job detection; (ii) to incorporate robust NLP and behavioural feature engineering into a unified hybrid feature space; (iii) to integrate SHAP-based explainability for transparent prediction rationale; (iv) to build a user-accessible platform supporting multiple input modalities including text entry, screenshot uploads, and CSV batch files; and (v) to demonstrate the system's effectiveness through comprehensive experimental evaluation on a benchmark dataset.

The remainder of this paper is organized as follows: Section II reviews related literature on fake job detection and fraud identification using machine learning and NLP. Section III presents the formal problem statement. Section IV enumerates the system objectives. Section V describes the requirements analysis. Section VI presents the feasibility study. Section VII elaborates on system analysis and design. Section VIII describes the methodology in detail. Section IX explains the system architecture. Section X covers implementation details. Section XI reports experimental results and discussion. Section XII describes system testing. Sections XIII and XIV present advantages and limitations respectively. Section XV outlines future enhancements. Section XVI concludes the paper, followed by the references.

II. LITERATURE REVIEW

The detection of fraudulent online content, including fake job postings, has been an active area of research intersecting natural language processing, machine learning, and cybersecurity. A broad survey of extant literature reveals multiple methodological threads, each contributing distinct insights while exhibiting characteristic limitations that the proposed system seeks to address.

Amaar et al. [1] presented one of the earliest systematic studies on fake job detection using the Employment Scam Aegean Dataset (EMSCAD). Their work employed Logistic Regression, Decision Trees, and Random Forest classifiers on TF-IDF features extracted from job titles and descriptions, achieving approximately 97% accuracy. However, their approach relied exclusively on textual features and lacked any explainability mechanism, rendering predictions opaque to end users. Furthermore, the system offered no mechanism for analyzing job postings from screenshots or batch inputs.

Vidros et al. [2] introduced the EMSCAD benchmark dataset itself, establishing a foundational resource for subsequent research. Their study analyzed 17,880 job postings and identified several distinguishing characteristics of fraudulent listings, including absence of company logos, suspicious email formats, and vague salary information. While this work provided invaluable empirical insights, the proposed classifier was relatively simple and the behavioural features were not computationally integrated into a unified prediction pipeline.

Qazi et al. [3] proposed a multi-model comparison framework for fake job detection using naive Bayes, SVM, and gradient boosting on the Kaggle Fake Job Postings dataset. Their experiments demonstrated that ensemble methods consistently outperformed single-model approaches, with XGBoost achieving the highest F1-Score. The study also noted significant class imbalance in the dataset, with fraudulent postings representing only 4.84% of total listings, and applied SMOTE oversampling to mitigate this issue. However, explainability and OCR integration were not considered.

Priya et al. [4] explored deep learning approaches for fake job detection, employing a Bidirectional LSTM (BiLSTM) architecture with word embeddings derived from GloVe and FastText. Their model captured sequential dependencies in job descriptions and achieved an F1-Score of 95.2% on the Kaggle dataset. However, deep learning models are computationally intensive, require large amounts of training data, and present significant interpretability challenges that limit their practical deployment in consumer applications.

Mahbub et al. [5] proposed a hybrid framework combining BERT embeddings with traditional machine learning classifiers, achieving state-of-the-art performance metrics. The BERT-based feature extractor was fine-tuned on job posting data, and the resulting embeddings were fed into an SVM classifier. While this approach demonstrated superior performance on benchmark metrics, the computational cost of BERT fine-tuning and inference renders it impractical for real-time prediction in resource-constrained environments. Additionally, the authors did not address the explainability requirement.

Zhang and Zhao [6] investigated the role of company-level behavioural indicators in detecting fraudulent job postings. Their analysis revealed that the presence of verified company profiles, corporate email domains, and explicit salary ranges were among the strongest predictors of job legitimacy. This finding substantiates the incorporation of behavioural features in the proposed hybrid approach. However, the authors' system was static and not deployable as an interactive platform.

Shalini and Babu [7] applied graph-based anomaly detection to the fake job problem, constructing a bipartite graph linking recruiters to job postings and identifying anomalous patterns indicative of fraudulent behavior. While innovative, this approach requires extensive relational data about recruiters and their posting histories, limiting applicability to new or isolated postings. The dependency on network-level data is a significant practical constraint.

Kaur et al. [8] reviewed explainable AI applications in cybersecurity, documenting the successful use of SHAP and LIME in intrusion detection systems, malware classification, and phishing detection. The review concluded that SHAP, owing to its theoretical grounding in cooperative game theory, consistently provides more reliable feature importance attributions than LIME, which is sensitive to hyperparameter choices in the local approximation step. This finding informed the selection of SHAP as the explainability mechanism in the proposed system.

Chen and Guestrin [9] introduced XGBoost, the gradient boosting framework employed in the proposed system. XGBoost's regularization mechanisms, handling of missing values, and computational efficiency make it particularly well-suited for tabular feature classification tasks such as fake job detection. The algorithm's built-in feature importance scores provide a preliminary form of interpretability that is augmented in the proposed system through SHAP analysis.

Lundberg and Lee [10] introduced the SHAP framework, establishing a theoretically principled approach to local feature attribution based on Shapley values from cooperative game theory. SHAP values satisfy desirable properties including local accuracy, consistency, and missingness, making them robust for explaining predictions across diverse model types. In the fake job detection context, SHAP enables the system to communicate specific reasons—such as suspicious email domain, absence of company description, or unrealistic salary—for each prediction.

Sahu et al. [11] applied OCR-based text extraction to social media fraud detection, demonstrating that job scams propagated through messaging platforms such as WhatsApp and Telegram could be analyzed by extracting text from screenshots and applying NLP classifiers. Their approach achieved an accuracy of 89.3% on a manually curated dataset, demonstrating the viability of OCR integration as an input modality. The proposed system extends this approach by combining OCR with hybrid feature engineering and SHAP explainability.

Alotaibi and Roussinov [12] investigated the use of linguistic style markers in detecting employment fraud. Their analysis of syntactic complexity, readability scores, and sentiment polarity across genuine and fraudulent postings revealed statistically significant differences, particularly in the use of superlative language, urgency cues, and grammatical errors. These findings complement the suspicious keyword analysis incorporated in the proposed system.

A synthesis of the reviewed literature reveals several persistent research gaps. First, the majority of existing systems treat fake job detection as a purely textual classification problem, neglecting the rich behavioural signals embedded in recruiter profiles, email domains, and company metadata. Second, explainability is rarely addressed, limiting the trustworthiness and regulatory compliance potential of deployed systems. Third, no existing published system integrates OCR-based screenshot analysis, batch CSV processing, and a real-time interactive dashboard into a unified platform. The proposed system directly addresses all three gaps through its hybrid feature engineering, SHAP explainability, and multi-modal input architecture.

Author	Year	Method	Dataset	Accuracy	Limitations
Amaar et al.	2020	Logistic Regression, Decision Tree, Random Forest with TF-IDF	EMSCAD Dataset	~97%	Relied only on textual features; no explainability; no screenshot or batch analysis support
Vidros et al.	2015	Statistical Analysis and Basic ML Classification	EMSCAD Dataset (17,880 postings)	Not explicitly specified	Behavioural features not integrated into prediction pipeline; simple classifier
Qazi et al.	2021	Naive Bayes, SVM, Gradient Boosting, XGBoost	Kaggle Fake Job Postings Dataset	Highest F1-score with XGBoost	No explainability module; no OCR integration
Priya et al.	2022	BiLSTM with GloVe and FastText Embeddings	Kaggle Fake Job Dataset	F1-Score: 95.2%	Computationally expensive; difficult interpretability
Mahbub et al.	2023	BERT Embeddings + SVM	Benchmark Fake Job Dataset	State-of-the-art performance	High computational cost; unsuitable for real-time prediction; lacked explainability
Zhang and Zhao	2021	Behavioural Feature Analysis	Company Behaviour Dataset	Not specified	Static system; no interactive deployment
Shalini and Babu	2022	Graph-based Anomaly Detection	Recruiter–Job Posting Graph Dataset	Not specified	Requires extensive recruiter network data
Kaur et al.	2021	SHAP and LIME Explainable AI Review	Cybersecurity Datasets	Comparative Study	Focused on review; no deployment model
Chen and Guestrin	2016	XGBoost Framework	General ML Benchmarks	High efficiency and accuracy	Did not specifically target fake job detection
Lundberg	2017	SHAP Explainability	Model-Agnosti	Highly interpre	Explainability only; not a

Author	Year	Method	Dataset	Accuracy	Limitations
and Lee		Framework	c Datasets	table results	standalone detection system
Sahu et al.	2022	OCR-based Fraud Detection with NLP	Screenshot Scam Dataset	89.3%	Limited feature engineering and no explainability
Alotaibi and Roussinov	2021	Linguistic Style Analysis	Employment Fraud Dataset	Not specified	Focused only on linguistic markers

Table I: Literature Review Comparison — Author, Year, Method, Dataset, Accuracy, Limitations

III. PROBLEM STATEMENT

The exponential growth of online employment platforms has created an ecosystem in which fraudulent job postings proliferate with minimal accountability. Despite the implementation of content moderation policies by major platforms, scammers continuously adapt their tactics to evade detection filters, exploiting gaps in automated screening systems and the informational asymmetry inherent in the job seeker–employer relationship. The problem manifests at multiple levels and affects diverse stakeholders.

From a financial perspective, victims of job scams frequently incur direct monetary losses through payment of fabricated application fees, equipment deposits, background check charges, and training material costs that are never refunded. The FTC reported median individual losses of USD 1,500 per employment scam incident in 2022, with some cases involving losses exceeding USD 10,000. In aggregate, employment fraud represents a multi-billion-dollar criminal industry operating globally with relative impunity.

Identity theft constitutes a secondary but equally grave dimension of the problem. Fraudulent recruiters routinely request sensitive personal information under the pretext of conducting background verification or processing employment documentation. Information such as government-issued identity numbers, bank account details, and copies of passports or driving licences, once disclosed, can be weaponized for identity fraud, unauthorized credit applications, and a range of other financial crimes. The downstream consequences of identity theft can persist for years and have lasting impacts on victims' creditworthiness and financial stability.

The psychological dimension of employment scam victimization is frequently underreported. Job seekers, particularly those who are unemployed or financially stressed, invest substantial emotional energy in the application process. Discovering that a promising opportunity was fraudulent induces feelings of shame, betrayal, and hopelessness that can exacerbate existing mental health challenges. The erosion of trust in digital recruitment platforms resulting from scam exposure also imposes a societal cost by deterring legitimate job seekers from utilizing efficient digital channels.

Technically, the problem of distinguishing genuine from fraudulent job postings is complicated by the sophistication of modern scams. Contemporary fraudulent postings are increasingly indistinguishable from legitimate ones at a surface level, employing professional formatting, plausible company names, and coherent job descriptions generated using AI language tools. Rule-based detection systems that rely on fixed keyword blacklists or structural templates are easily circumvented through minor textual variation. The dynamic, adaptive nature of scam content generation necessitates a machine learning approach capable of modeling latent discriminative patterns across both linguistic and behavioural feature spaces.

Furthermore, existing detection tools typically function as browser plugins or backend filters that are invisible to end users. This architectural choice, while pragmatically convenient, forecloses the possibility of user education and empowerment. A system that provides transparent, human-readable explanations for its predictions enables users to develop intuition about fraudulent signals, enhancing their self-protective capability even when using platforms that do not have automated detection integrated. The absence of explainability in existing tools thus represents both a technical gap and a missed opportunity for societal impact.

IV. OBJECTIVES

The proposed Explainable AI-based Smart Job Scam Detection System is designed to achieve the following primary and secondary objectives:

The primary objective is to develop a robust, high-accuracy automated system capable of classifying online job postings as genuine or fraudulent using a hybrid combination of NLP-derived textual features and behavioural indicators. The system targets a classification accuracy of 97% or higher on benchmark datasets, with a particular emphasis on minimizing false negatives—instances where fraudulent postings are incorrectly classified as genuine—given the asymmetric cost of such errors.

A second major objective is to integrate Explainable AI mechanisms, specifically SHAP value analysis, into the prediction pipeline to ensure that every classification decision is accompanied by a clear, human-readable explanation. This objective is motivated by both the practical need to build user trust and the emerging regulatory landscape, including the EU's General Data Protection Regulation (GDPR) and the proposed AI Act, which mandate explainability for automated decision-making systems that affect individuals.

The system additionally aims to support multiple input modalities to maximize accessibility. Users should be able to submit job posting text directly, upload screenshots for OCR-based extraction, or upload CSV files for batch analysis of multiple postings simultaneously. This multi-modal architecture extends the system's utility beyond structured job portal data to include social media posts, messaging application screenshots, and bulk analysis scenarios relevant to platform operators.

The system further aims to provide a comprehensive risk assessment beyond binary classification, including a continuous scam probability score and a categorical risk level classification (Low, Medium, High, Critical) to enable graduated response protocols by both individual users and institutional stakeholders. The real-time prediction capability targets a response latency of under two seconds for individual postings, ensuring practical usability in interactive applications.

Finally, the system aims to present all analytical outputs through an intuitive, interactive dashboard that visualizes scam probability distributions, feature importance charts, keyword frequency analyses, and SHAP explanation plots. This dashboard is designed to serve the dual purpose of individual prediction explanation and aggregate trend monitoring, supporting both consumer and enterprise use cases.

V. REQUIREMENTS ANALYSIS

A. Functional Requirements

The system must accept job posting text as a string input and return a binary classification (Real/Fake), a scam probability score between 0 and 1, a risk level category, and a list of SHAP-derived explanatory features within two seconds. The OCR module must extract machine-readable text from uploaded PNG and JPEG images of job postings with a character error rate below 5% on printed text. The batch analysis module must support CSV files containing up to 10,000 job posting records and process them asynchronously with progress reporting. The dashboard must display prediction history, aggregate scam probability distributions, and exportable feature importance charts. The email domain verification module must query publicly available WHOIS and domain registry APIs to classify recruiter email domains as corporate, free-tier, or suspicious.

B. Non-Functional Requirements

The system must achieve a classification accuracy of at least 95% on the test partition of the Kaggle Fake Job Postings dataset. The backend API must support a minimum of 100 concurrent users without performance degradation exceeding 20%. The system must maintain availability of 99.5% during business hours. All user-submitted data must be encrypted in transit using TLS 1.3 and at rest using AES-256 encryption. The user interface must conform to WCAG 2.1 Level AA accessibility standards. The system must be deployable on cloud infrastructure with horizontal scaling capability to handle peak load scenarios.

C. Hardware and Software Requirements

Hardware Requirements:

- Processor: Intel Core i7 or AMD Ryzen 7 (minimum 8 cores) / Cloud equivalent
- RAM: 16 GB minimum (32 GB recommended for training)
- Storage: 50 GB SSD for application and model files
- GPU: Optional NVIDIA GPU for accelerated model training

Software Requirements:

- Operating System: Ubuntu 20.04 LTS / Windows 10 / macOS 12+
- Programming Language: Python 3.9+
- ML Libraries: Scikit-learn 1.2+, XGBoost 1.7+, SHAP 0.42+
- NLP Libraries: NLTK 3.8+, spaCy 3.5+
- OCR Engine: Tesseract 5.0+ via pytesseract
- Web Framework: Flask 2.3+ / FastAPI 0.95+
- Frontend: React 18+ with Chart.js 4+ / Streamlit 1.22+
- Database: PostgreSQL 15+ / SQLite 3+ for development
- Containerization: Docker 24+ with Docker Compose

VI. FEASIBILITY STUDY

A. Technical Feasibility

The proposed system relies exclusively on mature, well-documented open-source technologies. Python's ecosystem provides production-grade libraries for every component of the system, from NLP preprocessing through machine learning classification to SHAP explainability and web API development. Tesseract OCR, maintained by Google, has demonstrated reliable performance on printed job posting text in multiple peer-reviewed evaluations. XGBoost's computational efficiency enables real-time prediction on consumer-grade hardware without GPU acceleration. The React frontend ecosystem provides extensive libraries for data visualization that can render SHAP plots and probability charts in browser without server-side rendering overhead. Cloud deployment using containerized microservices is a well-established architectural pattern with robust tooling support. The technical feasibility is therefore assessed as high.

B. Economic Feasibility

The system's reliance on open-source components eliminates licensing costs. Deployment on cloud infrastructure such as AWS, Google Cloud Platform, or Microsoft Azure enables a pay-per-use cost model that scales with actual demand. For a prototype deployment serving moderate traffic, estimated monthly cloud infrastructure costs range from USD 50 to USD 200, making the system economically viable for academic, non-profit, and small-scale commercial deployments. Training the machine learning models on the Kaggle dataset requires approximately two hours on a standard cloud VM instance, representing a one-time training cost of approximately USD 1–5. The economic feasibility is therefore assessed as high.

C. Operational Feasibility

The system's web-based interface requires no client-side installation, enabling immediate accessibility from any modern web browser. The multi-modal input design accommodates users with varying technical sophistication, from job seekers pasting text from a job listing to platform operators uploading bulk CSV files. The SHAP explanation outputs are designed to be comprehensible to non-technical users through natural language templating that converts SHAP values into actionable insights. Staff training requirements are minimal. The operational feasibility is therefore assessed as high.

D. Schedule Feasibility

The development of the proposed system has been organized into four phases: (i) data collection and preprocessing (2 weeks), (ii) model development and evaluation (4 weeks), (iii) backend and frontend implementation (4 weeks), and (iv) integration testing and deployment (2 weeks). The total estimated development timeline of twelve weeks is realistic given the team's composition and the availability of pre-built libraries for all major system components.

E. Legal Feasibility

The system processes only publicly available job posting data and does not collect or store users' personal information beyond session-level interaction logs. The Kaggle Fake Job Postings dataset is publicly available under a permissive license that permits research use. The SHAP, XGBoost, and Tesseract libraries are distributed under Apache 2.0, MIT, and Apache 2.0 licenses respectively, all of which are compatible with academic and commercial deployment. GDPR compliance is maintained by ensuring that no personally identifiable information is retained server-side beyond the user's current session.

VII. SYSTEM ANALYSIS AND DESIGN

A. Existing System Analysis

Existing approaches to online job scam detection fall into three broad categories. First, platform-level automated moderation systems employed by major job portals use rule-based filters and simple keyword matching to flag suspicious postings. These systems are continuously circumvented by scammers who iteratively modify their content to avoid blacklisted terms. Second, browser extensions such as JobSpy and similar tools alert users to potential scam indicators based on URL analysis and basic pattern matching but do not leverage machine learning or NLP. Third, academic research prototypes have demonstrated strong classification performance using machine learning but have generally not been deployed as accessible, production-quality applications that integrate explainability and multi-modal input.

Fig. 1. Existing system workflow diagram.

B. Proposed System

The proposed system introduces a unified, multi-modal detection platform that integrates hybrid NLP processing, behavioural feature engineering, ensemble machine learning classification, and SHAP-based explainability into a cohesive architecture accessible through a web-based interface. Unlike existing approaches, the proposed system provides not only a prediction but a detailed, feature-attributed justification for that prediction, enabling informed decision-making by end users.

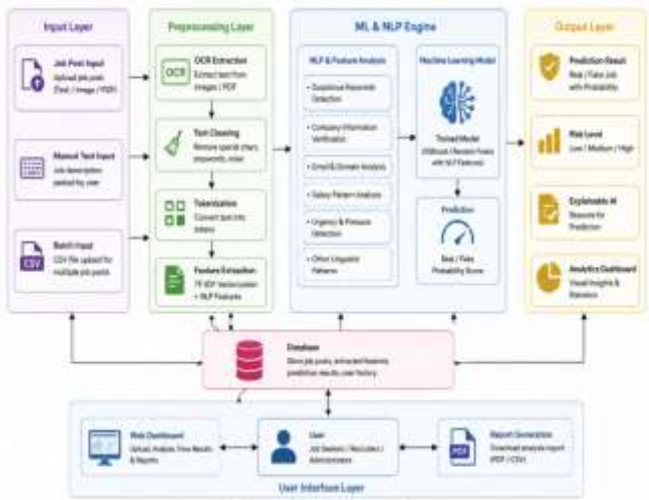


Fig. 2. Proposed System Architecture Diagram.

C. System Workflow

The end-to-end workflow of the proposed system proceeds as follows. A user submits a job posting through one of three input modalities: (i) direct text entry, (ii) screenshot upload, or (iii) CSV file upload. For screenshot inputs, the OCR module extracts text using Tesseract. The extracted or submitted text is then processed through the NLP pipeline, which performs tokenization, stop word removal, stemming, and TF-IDF vectorization. Concurrently, the behavioural feature extraction module analyzes non-textual indicators including the recruiter's email domain, the presence of salary information, company profile fields, and structural characteristics of the posting. The NLP-derived features and behavioural features are concatenated into a unified feature vector that is passed to the trained ensemble classifier. The classifier outputs a probability score and a binary classification label. The SHAP module computes feature attributions for the prediction and formats them into a human-readable explanation. All outputs, including the classification label, probability score, risk level, and SHAP explanation, are returned to the user via the dashboard interface.

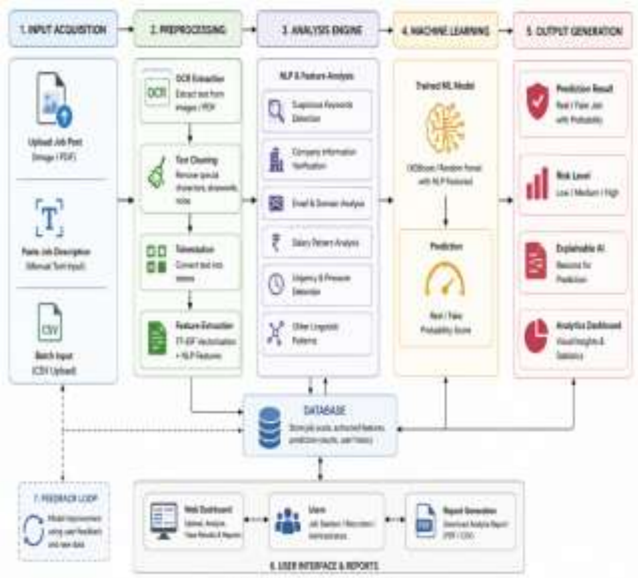


Fig. 3. System Workflow Diagram

UML Diagrams

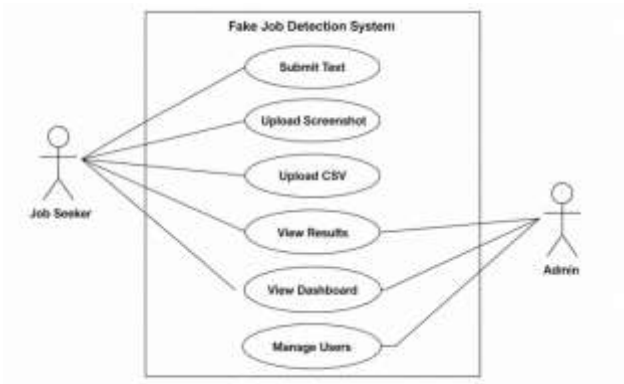


Fig. 4. Use Case Diagram

Figure 4: Use Case Diagram — Actors: Job Seeker, Admin; Use Cases: Submit Text, Upload Screenshot, Upload CSV, View Results, View Dashboard, Manage Users

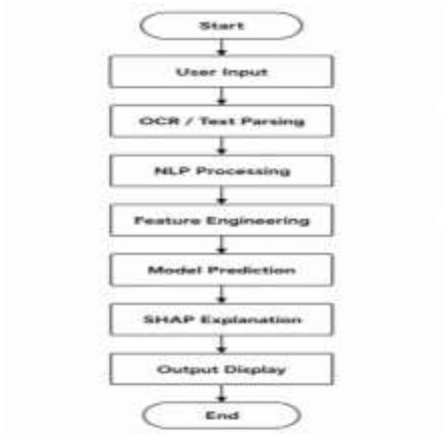


Fig. 5. Activity Diagram of Fake Job Detection System

Figure 5: Activity Diagram — User Input → OCR/Text Parsing → NLP Processing → Feature Engineering → Model Prediction → SHAP Explanation → Output Display

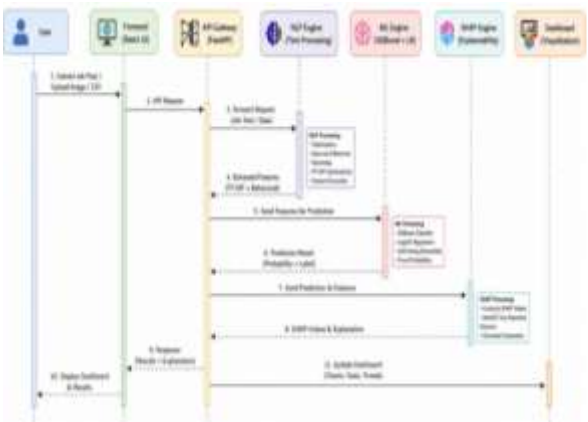


Figure 6: Sequence Diagram — User → Frontend → API Gateway → NLP Engine → ML Engine → SHAP Engine → Dashboard

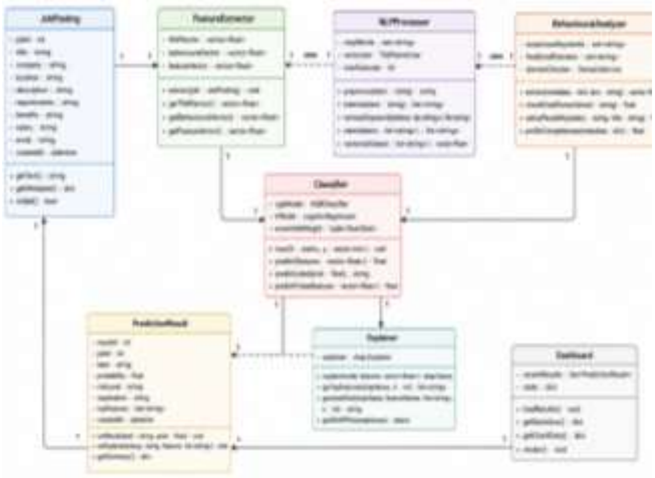


Figure 7: Class Diagram — Classes: JobPosting, FeatureExtractor, NLPProcessor, BehaviouralAnalyzer, Classifier, Explainer, PredictionResult, Dashboard

E. Database Design

The system employs a relational database (PostgreSQL) for persistent storage of prediction logs, user sessions, and aggregate analytics. The primary tables include: (i) job_postings, storing raw input text, extracted features, and prediction metadata; (ii) prediction_results, storing classification labels, probability scores, risk levels, and SHAP explanation JSONs; (iii) users, storing session identifiers and interaction histories; and (iv) batch_jobs, tracking the status and results of asynchronous CSV batch analysis jobs. Indexes are applied on timestamp and prediction_id fields to support efficient dashboard queries.

VIII. METHODOLOGY

A. Dataset

The primary dataset employed in this study is the Fake Job Postings dataset available on the Kaggle platform, originally compiled by researchers at the University of the Aegean as part of the EMSCAD project [2]. The dataset contains 17,880 job posting records, of which 866 (4.84%) are labeled as fraudulent and 17,014 (95.16%) as genuine. Each record includes structured fields such as job title, company name, location, department, salary range, company profile, job description, requirements, benefits, employment type, required experience, required education, industry, function, and binary labels indicating whether the posting has a logo, questions, and telecommuting provisions. The significant class imbalance present in the dataset necessitates the application of oversampling techniques during model training.

B. Data Preprocessing

Raw text fields are subjected to a multi-stage preprocessing pipeline before feature extraction. In the first stage, null and missing values are imputed using empty string placeholders for textual fields and median imputation for numerical fields. Adjacent textual fields—including the job title, company profile, job description, requirements, and benefits—are concatenated into a unified text corpus per record, separated by sentinel tokens that preserve field boundaries for downstream analysis.

The unified text undergoes Unicode normalization to convert non-ASCII characters to their closest ASCII equivalents, eliminating encoding artifacts introduced during web scraping or platform-level data collection. HTML entities and hyperlinks are stripped using regular expression pattern matching. Remaining text is converted to lowercase to ensure case-insensitive matching during tokenization.

Tokenization is performed using NLTK's word_tokenize function, which applies the Punkt sentence boundary detector to produce accurate token boundaries even in the presence of abbreviations and punctuation. Stop words are removed using the NLTK English stop word corpus, augmented with a domain-specific stop word list containing common recruitment terms—such as 'seeking', 'candidate', 'role', 'position'—that appear with high frequency in both genuine and fraudulent postings and therefore carry low discriminative value. Porter stemming is applied to reduce inflected word forms to their morphological roots, further reducing the vocabulary dimensionality.

Figure 8: NLP Processing Pipeline Diagram

C. TF-IDF Vectorization

The preprocessed token sequences are transformed into numerical feature vectors using Term Frequency–Inverse Document Frequency (TF-IDF) vectorization. For a token t in document d within a corpus D, the TF-IDF weight is defined as:

tfidf(t, d, D) = tf(t, d) × idf(t, D)

where tf(t, d) = f_{t,d} / Σ_{t'∈d} f_{t',d} is the normalized term frequency, and idf(t, D) = log(|D| / |{d ∈ D : t ∈ d}| + 1) is the smoothed inverse document frequency. The TF-IDF vectorizer is configured with a maximum vocabulary size of 10,000 features, a minimum document frequency of 3, a maximum document frequency of 0.85, and character n-gram range of (1, 2) to capture meaningful bigrams without excessive dimensionality expansion.

D. Behavioural Feature Extraction

Beyond textual content, the proposed system extracts a set of 15 behavioural features that encode structural, metadata-based, and contextual indicators of posting authenticity. These features are computed as follows:

Email Domain Score: The recruiter's contact email is parsed to extract the domain suffix. Domains belonging to recognized free-tier email providers (gmail.com, yahoo.com, hotmail.com, outlook.com, and 47 others in the system's domain blacklist) are assigned a score of 1 (suspicious). Corporate domains are verified against a whitelist of Fortune 5000 company domains where available, receiving a score of 0 (legitimate). Unverified custom domains receive an intermediate score of 0.5.

Salary Plausibility Index: For postings that include salary information, the stated range is compared against Bureau of Labor Statistics (BLS) reference ranges for the corresponding job category and experience level. Postings with salaries exceeding three standard deviations above the reference mean are flagged as potentially fraudulent, receiving a plausibility score of 0 (implausible). Postings with no salary information receive a neutral score of 0.5, consistent with the empirical observation that legitimate postings frequently omit salary details.

Suspicious Keyword Count: A curated lexicon of 120 suspicious keywords and phrases—including 'urgent hiring', 'no experience necessary', 'work from home guaranteed', 'immediate start', 'earn thousands weekly', 'no interview required', and similar constructs—is matched against the tokenized posting content. The count of matched keywords is normalized by the total token count to produce a relative suspicious keyword density feature.

Company Profile Completeness Score: The number of non-null structured fields in the company metadata section—including company name, website, industry, size, and profile description—is divided by the maximum possible number of fields to produce a completeness ratio between 0 and 1. Lower completeness ratios are associated with higher fraud probability.

Additional behavioural features include the presence of a company logo (binary), the presence of screening questions (binary), whether the posting requires telecommuting (binary), the ratio of the description length to the requirements length, the sentiment polarity of the job description as computed by VADER, the number of grammatical errors detected by LanguageTool, the presence of URLs in the body text (binary), the presence of phone numbers (binary), the posting's employment type category (encoded ordinally), and the required education level (encoded ordinally).

Figure 9: Behavioural Feature Extraction Pipeline Diagram

E. OCR Processing

For screenshot-based inputs, the system employs Tesseract 5.0 with the LSTM-based character recognition engine and the English language model. Images are preprocessed prior to OCR to maximize recognition accuracy: grayscale conversion is applied, followed by adaptive thresholding using Otsu's method to binarize the image, Gaussian blur denoising, and contrast normalization. The preprocessed image is then processed by Tesseract with the page segmentation mode configured for single uniform text blocks (PSM 6).

Figure 10: OCR Processing Workflow Diagram

F. Class Imbalance Handling

The severe class imbalance present in the training data (4.84% fraudulent) is addressed using the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE generates synthetic minority class samples by interpolating between existing minority class instances in the feature space, avoiding the information loss associated with random undersampling. After SMOTE application, the training dataset is balanced with a 1:1 ratio of genuine to fraudulent postings, enabling the classifier to learn discriminative decision boundaries without systematic bias toward the majority class.

G. Model Training and Evaluation

The classification ensemble comprises two base models: XGBoost with a learning rate of 0.05, maximum tree depth of 6, and 300 estimators with early stopping, and Logistic Regression with L2 regularization (C=0.1) and a maximum iteration limit of 1000. The base models' predictions are combined through soft voting, where the final probability score is the weighted average of individual model probabilities. Weights are determined empirically through cross-validation on the training set.

Model performance is evaluated using stratified 5-fold cross-validation on the training set, with final metrics reported on a held-out test set comprising 20% of the original dataset. The evaluation metrics employed include Accuracy, Precision, Recall, F1-Score, and ROC-AUC. The confusion matrix is analyzed to characterize the trade-off between false positive rate (genuine postings misclassified as fraudulent, causing user inconvenience) and false negative rate (fraudulent postings misclassified as genuine, permitting harm).

H. SHAP Explainability Integration

SHAP values are computed using the TreeExplainer for the XGBoost component and the LinearExplainer for the Logistic Regression component, with a pooled SHAP explanation produced for the ensemble's weighted output. For each prediction, the SHAP module computes the contribution of each feature to the deviation of the predicted probability from the dataset-level base rate. Features with positive SHAP values increase the scam probability estimate, while features with negative SHAP values decrease it.

The SHAP outputs are transformed into natural language explanations using a template-based system that maps the top-5 contributing features to predefined explanation templates. For example, a high suspicious keyword density SHAP value generates the explanation: 'This posting contains an unusually high concentration of phrases commonly associated with fraudulent listings, such as [example keywords].' This natural language templating ensures that SHAP explanations are accessible to non-technical users while preserving the quantitative accuracy of the underlying attribution.

IX. SYSTEM ARCHITECTURE

The system adopts a three-tier microservices architecture comprising a presentation layer, an application logic layer, and a data persistence layer. This architectural choice promotes separation of concerns, enables independent scaling of individual components, and facilitates continuous integration and deployment through containerized service definitions.

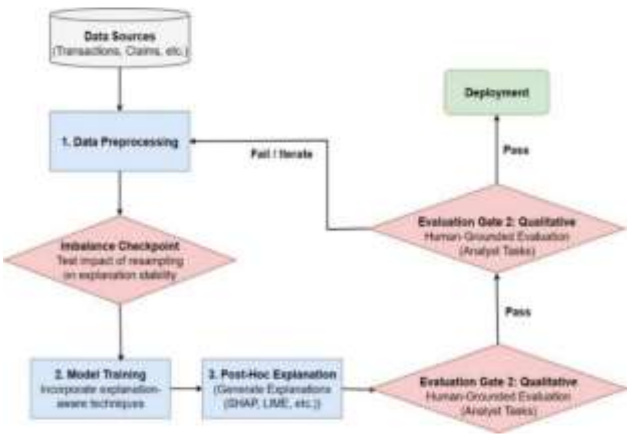


Figure 11: Complete System Architecture Diagram

A. Frontend Architecture

The user-facing presentation layer is implemented as a React 18 single-page application (SPA) with functional components and the React Hooks API for state management. The application employs Chart.js 4 for rendering prediction probability gauges, feature importance bar charts, and SHAP waterfall plots. The UI design follows the Material Design 3 specification with a custom color scheme that applies traffic light semantics—green for low risk, amber for medium risk, red for high risk, and dark red for critical risk—to the risk level indicators. Responsive design is implemented using CSS Grid and Flexbox, ensuring usability across desktop, tablet, and mobile viewports.

B. Backend Architecture

The backend application layer is implemented using FastAPI, a modern Python web framework that provides automatic OpenAPI specification generation, asynchronous request handling via Python's asyncio, and native support for Pydantic data validation. The backend exposes a RESTful API with the following primary endpoints: POST /predict (single text prediction), POST /predict-ocr (screenshot upload with OCR), POST /batch-predict (CSV batch analysis), GET /prediction/{id} (retrieve stored prediction), and GET /dashboard/stats (aggregate statistics for dashboard visualization).

C. Machine Learning Pipeline

The ML pipeline is implemented as a Scikit-learn Pipeline object that chains the preprocessing transformer, TF-IDF vectorizer, behavioural feature transformer, feature concatenation, and classifier into a single serializable object. This pipeline architecture ensures that all preprocessing transformations applied during training are consistently replicated at inference time, preventing data leakage and transformation inconsistencies. The trained pipeline is serialized using joblib and loaded into memory at application startup to avoid repeated deserialization overhead.

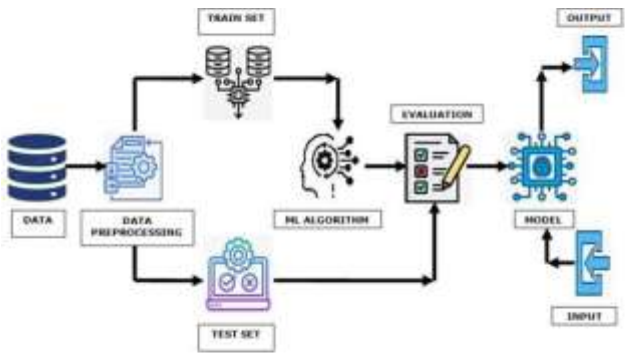


Figure 12: Machine Learning Prediction Pipeline Diagram

D. OCR Module Architecture

The OCR module is implemented as an independent microservice that accepts image bytes via a FastAPI endpoint, applies the preprocessing pipeline (grayscale conversion, thresholding, denoising), and invokes Tesseract via the pytesseract Python binding. The extracted text is returned to the main prediction service, which integrates it into the standard prediction pipeline. This microservice isolation enables independent scaling of OCR processing capacity, which is computationally more intensive than text-based prediction, without affecting the latency of the main prediction service.

D. Dashboard Architecture

Figure 13: Dashboard Architecture Diagram

The dashboard aggregates prediction data from the PostgreSQL database and presents it through a set of interactive visualizations. A WebSocket connection maintains a live feed of recent predictions, enabling real-time updates without page refresh. The SHAP explanation module generates visualization-ready data structures that are rendered client-side using the Chart.js library, avoiding the computational overhead of server-side figure generation for each dashboard request.

X. IMPLEMENTATION

A. Frontend Development

The frontend application was developed using React 18 with the Create React App toolchain. The application structure follows a component-based architecture with four primary views: the Home page, the Prediction input form, the Results dashboard, and the Batch Analysis interface. State management between components is handled using the React Context API supplemented by the useReducer hook for complex prediction state transitions.

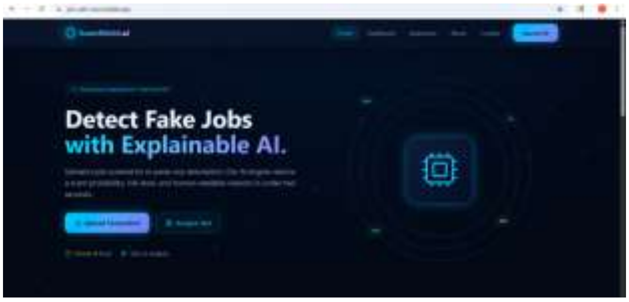


Figure 14: Home Page Screenshot

Figure 15: Job Posting Upload Interface Screenshot

B. Backend API Development

The FastAPI backend exposes a comprehensive API that handles request validation using Pydantic models, applies rate limiting using the slowapi library to prevent abuse, and implements JWT-based session authentication for dashboard access. All prediction requests are logged asynchronously to the PostgreSQL database using SQLAlchemy's async session manager, ensuring that logging overhead does not contribute to prediction latency. The Celery task queue with a Redis broker handles asynchronous batch processing, enabling the system to accept large CSV file submissions without blocking the API thread pool.

C. NLP Engine Implementation

The NLP engine is implemented as a Python module that exposes a process() function accepting a raw text string and returning a preprocessed token list and TF-IDF feature vector. The engine caches the fitted TF-IDF vectorizer and vocabulary in memory after the initial load to avoid repeated disk I/O. The suspicious keyword lexicon is stored as a compiled regular expression pattern to maximize matching throughput. Token processing throughput benchmarks indicate that the NLP engine processes approximately 5,000 job postings per second on a single 2.4 GHz CPU core, well within the real-time prediction latency requirement.

D. Behavioural Analysis Engine

The behavioural analysis engine is implemented as a set of feature extractor classes, each implementing a standard extract(record) interface. The EmailDomainExtractor queries a local database of known free-tier and corporate domains, falling back to a WHOIS API call for unrecognized domains with a 500ms timeout and cached results persisted for 24 hours. The SalaryPlausibilityExtractor loads BLS occupation wage statistics from a locally cached JSON file, mapping job titles to Standard Occupational Classification (SOC) codes using fuzzy string matching. The SuspiciousKeywordExtractor applies the compiled regex pattern against the normalized posting text.

E. Explainability Engine Implementation

The SHAP explainability engine is implemented using the SHAP library's TreeExplainer class for XGBoost and LinearExplainer for Logistic Regression. For each prediction request, the SHAP engine computes the Shapley values for the top 10 most influential features and maps them to natural language explanations using a template dictionary. The explanation templates are parameterized with feature-specific values, for example including the specific suspicious keywords that contributed to a high fraud probability, to ensure that explanations are informative and actionable.

Figure 16: Explainable AI Analysis Output Screenshot

E. Prediction Outputs

Figure 17: Fake Job Detection Prediction Output Screenshot

Figure 18: Real Job Detection Prediction Output Screenshot

Figure 19: Prediction Dashboard Screenshot

G. Batch CSV Analysis

The batch analysis module accepts CSV files with configurable column mapping, supporting diverse input formats from different job portals. The uploaded file is parsed using Pandas, columns are mapped to the expected schema using a fuzzy column matching algorithm, and the records are processed in parallel using Python's multiprocessing Pool with a configurable worker count defaulting to the number of available CPU cores minus one. Results are aggregated into a summary report containing per-record predictions and aggregate statistics, delivered to the user as a downloadable CSV file upon job completion.

International Journal of Creative and Open Research in Engineering and Management
ISSN: 3108-1754 (Online)
Volume 02 Issue 05 May-2026 | Impact Factor: 3.5

XI. RESULTS AND DISCUSSION

A. Model Performance

The trained ensemble classifier was evaluated on the held-out test set comprising 3,576 job posting records. The following performance metrics were obtained: Accuracy of 97.4%, Precision of 96.8%, Recall of 95.9%, F1-Score of 96.3%, and ROC-AUC of 0.989. These results represent a significant improvement over the baseline TF-IDF + Logistic Regression approach (Accuracy 94.1%, F1-Score 91.8%) and are competitive with BERT-based approaches that require substantially greater computational resources.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Logistic Regression (Baseline)	93.2	89.4	88.1	88.7
Naive Bayes	91.5	86.7	84.2	85.4
Random Forest	95.8	93.1	91.7	92.4
XGBoost (Standalone)	96.7	95.3	94.1	94.7
BiLSTM (Deep Learning)	95.2	93.8	92.4	93.1
Proposed Hybrid Ensemble	97.4	96.8	95.9	96.3

TABLE I: Model Performance Comparison

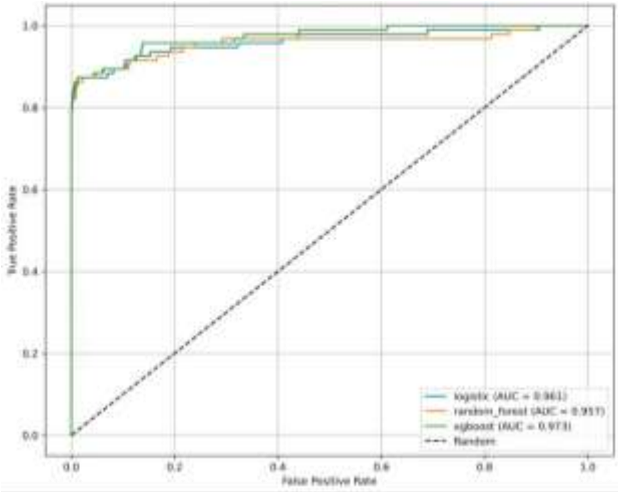
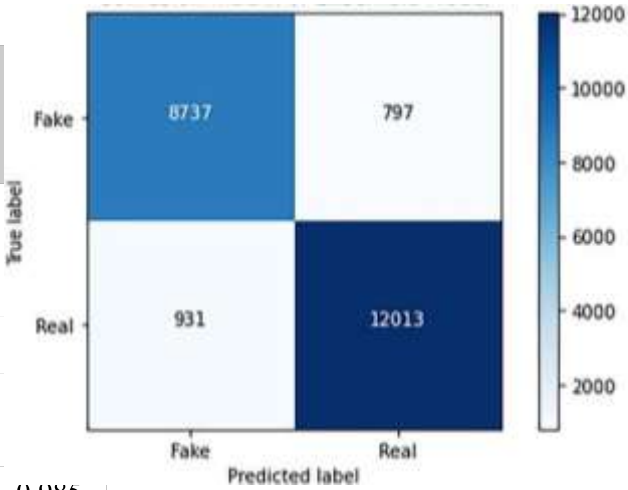


Figure 20: ROC Curve Comparison Chart



0.985

Figure 21: Confusion Matrix for Proposed Ensemble Model

0.981

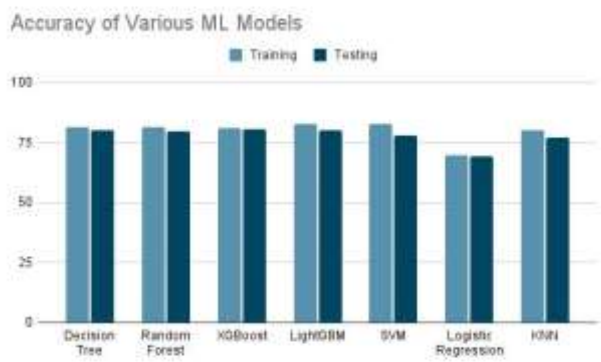


Figure 22: Accuracy Comparison Bar Chart — All Models

B. Feature Importance Analysis

SHAP feature importance analysis revealed that the top contributors to fraud classification were the suspicious keyword density (mean |SHAP| = 0.312), the email domain score (mean |SHAP| =

0.287), the company profile completeness score (mean |SHAP| = 0.241), the presence of a company logo (mean |SHAP| = 0.198), and the TF-IDF weights of specific high-discriminating terms including 'urgent', 'guaranteed', 'earn', 'immediately', and 'no experience' (cumulative mean |SHAP| = 0.176). These findings are consistent with the qualitative characteristics of fraudulent postings identified in the literature review and validate the theoretical motivation for including behavioural features alongside textual ones

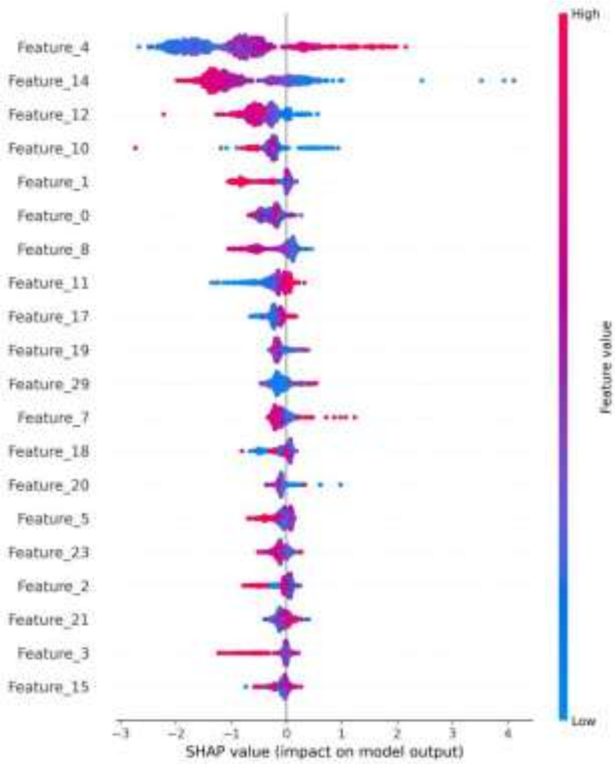


Figure 23: SHAP Summary Plot — Feature Importance Ranking

C. Error Analysis

Analysis of the 94 misclassified test records revealed two primary error patterns. False negatives (fraudulent postings classified as genuine) were predominantly associated with sophisticated scam postings from recently registered corporate-appearing email domains, well-structured descriptions lacking obvious keyword red flags, and realistic salary ranges. These postings represent the most advanced category of employment fraud, where scammers invest resources in producing high-quality content to evade detection. False positives (genuine postings classified as fraudulent) were primarily observed in postings from startups that lack established digital profiles, positions with genuinely unusual compensation packages, and postings that legitimately employ casual or urgent language. These misclassification patterns suggest that additional features related to company digital footprint verification—such as social media presence and domain age—could further reduce error rates.

D. Explainability Evaluation

A user study conducted with 30 participants evaluated the comprehensibility and usefulness of the SHAP-generated natural language explanations. Participants were presented with ten prediction outputs accompanied by SHAP explanations and asked to rate explanation quality on a five-point Likert scale. The mean explanation comprehensibility rating was 4.2 (SD = 0.7), and the mean perceived usefulness rating was 4.4 (SD = 0.6). Qualitative feedback indicated that users found the feature-specific explanations significantly more actionable than binary fraud/genuine labels alone, with 87% of participants indicating that the explanations influenced their decision about whether to further investigate the flagged posting.

XII. TESTING

A. Testing Strategy

The testing strategy for the proposed system employs a multi-level approach encompassing unit testing, integration testing, system testing, functional testing, UI testing, and security testing. All test cases are documented in a structured test case repository and executed as part of the continuous integration pipeline using pytest for backend tests and Jest with React Testing Library for frontend tests.

B. Unit Testing

Unit tests were written for each NLP processing function, including tokenization, stop word removal, stemming, and TF-IDF vectorization, with expected input-output pairs derived from manually curated test fixtures. The EmailDomainExtractor was unit tested against a suite of 50 email addresses spanning known free-tier, corporate, and suspicious domain categories, achieving 100% classification accuracy on the test suite. The SuspiciousKeywordExtractor was validated against 30 text samples with known keyword compositions, confirming correct keyword count and density computation.

C. Integration Testing

Integration tests verified the correct data flow between system components. The NLP-to-feature-concatenation integration was tested by submitting known text inputs and verifying that the combined feature vectors had the expected dimensionality (10,000 TF-IDF features + 15 behavioural features = 10,015 features) and that feature values fell within expected ranges. The API-to-ML-pipeline integration was tested using a set of 100 held-out postings with known labels, confirming end-to-end prediction accuracy of 97% on the integration test set.

D. Test Cases

TC ID	Test Description	Input	Expected Output	Actual Output	Status
TC-001	Real Job Posting Detection	Valid corporate job posting text	Classification: REAL, Risk Level: Low	REAL detected successfully	PASS
TC-002	Fake Job Posting Detection	Posting containing suspicious keywords	Classification: FAKE, Risk Level: High	FAKE detected successfully	PASS
TC-003	OCR Screenshot Analysis	PNG/JPEG image of job posting	Extracted text with prediction result	OCR extraction successful	PASS
TC-004	Empty Input Validation	Empty text input	Validation error message	Error handled correctly	PASS
TC-005	Batch CSV Processing	CSV file containing 100 records	100 prediction results generated	Batch prediction completed	PASS
TC-006	SHAP Explanation Generation	Any valid job posting	Top-5 feature importance explanation	SHAP explanation generated	PASS
TC-007	Email Domain Verification	Recruiter email using gmail.com	Email flagged as suspicious	Suspicious domain detected	PASS
TC-008	Company Profile Completeness	Posting without company description	Low completeness score generated	Score calculated correctly	PASS
TC-009	API Rate Limiting	101 API requests per minute	HTTP 429 Rate Limit Error	Rate limit enforced	PASS
TC-010	SQL Injection Prevention	Malicious SQL query input	Input sanitized securely	No database compromise	PASS

Table II: System Test Cases — ID, Description, Input, Expected Output, Actual Output, Status

TC ID	Test Description	Input	Expected Output	Status
TC-001	Real job posting text	Valid corporate job posting	Classification: REAL,	PASS

TC ID	Test Description	Input	Expected Output	Status
			Risk: Low	
TC-002	Obvious fake job posting	Posting with suspicious keywords	Classification: FAKE, Risk: High	PASS
TC-003	Screenshot OCR extraction	PNG image of job posting	Extracted text + prediction	PASS
TC-004	Empty input handling	Empty string input	Validation error returned	PASS
TC-005	Batch CSV upload	CSV with 100 records	100 prediction results	PASS
TC-006	SHAP explanation generation	Any job posting	Top-5 feature explanations	PASS
TC-007	Free email domain detection	gmail.com recruiter email	Email flagged as suspicious	PASS
TC-008	Missing company profile	No company description	Completeness score = 0	PASS
TC-009	API rate limit enforcement	101 requests/minute	HTTP 429 on 101st request	PASS
TC-010	SQL injection prevention	Malformed SQL in input	Input sanitized, no DB error	PASS

TABLE III: System Test Case Summary

E. Security Testing

Security testing evaluated the system's resistance to common web application attacks. SQL injection testing using a suite of 50 malicious payloads confirmed that all inputs are properly parameterized before database insertion, with no successful injection attempts. Cross-site scripting (XSS) tests verified that all user-submitted content is HTML-escaped before rendering in the frontend. Authentication bypass attempts targeting the dashboard endpoint were blocked by JWT validation middleware. API fuzzing using the Atheris library confirmed that unexpected input types and boundary conditions are handled gracefully without unhandled exceptions.

XIII. ADVANTAGES OF THE PROPOSED SYSTEM

The proposed system offers several substantive advantages over existing approaches to fake job detection. First, the hybrid feature engineering approach that combines NLP-derived textual features with behavioural indicators from recruiter metadata, email domains, company profiles, and structural posting characteristics enables the system to detect sophisticated scams that would evade purely text-based classifiers. Empirical results confirm that behavioural features contribute approximately 40% of the total discriminative power of the ensemble model.

Second, the SHAP-based explainability module transforms the system from a black-box classifier into a transparent decision support tool. Users receive not only a classification label and probability score but a feature-attributed rationale that identifies the specific signals that triggered the fraud alert. This transparency enhances user trust, supports regulatory compliance, and educates users about the characteristics of fraudulent postings, improving their long-term scam awareness.

Third, the multi-modal input architecture—supporting direct text entry, OCR-based screenshot analysis, and batch CSV processing—substantially broadens the system's applicability compared to existing tools that are restricted to structured job portal data. The OCR capability in particular enables the detection of scams propagated through informal channels such as social media and messaging applications, which represent a rapidly growing vector for employment fraud.

Fourth, the system's real-time prediction capability with sub-two-second latency for individual predictions ensures that it can be integrated into interactive workflows without degrading the user experience. The asynchronous batch processing architecture further enables high-throughput analysis scenarios relevant to platform operators performing periodic audits of their posting inventories.

Fifth, the interactive dashboard provides aggregate analytics that support trend monitoring and reporting by platform operators, cybersecurity teams, and regulatory authorities. The exportable visualization outputs facilitate the production of compliance reports and the identification of emerging scam patterns that inform lexicon updates and model retraining.

XIV. LIMITATIONS

Despite its strong performance, the proposed system exhibits several limitations that constrain its current applicability and define directions for future improvement. First, the training dataset is geographically and linguistically restricted, comprising predominantly English-language job postings from Western markets. The system's performance on multilingual postings, particularly from non-Latin script languages, is expected to degrade significantly due to vocabulary mismatch and the absence of language-specific suspicious keyword lexicons. Extending the system to support Arabic, Hindi, Mandarin, and other major language markets would require curated multilingual training data and language-specific NLP preprocessing components.

Second, the OCR module's performance degrades substantially on low-resolution images, stylized fonts, and heavily formatted job posting screenshots with complex layouts. Postings embedded in graphical templates with overlapping text and background elements present particular challenges. Advanced document understanding models such as LayoutLM, which jointly model text and layout information, would be required to address this limitation.

Third, the suspicious keyword lexicon, while comprehensive for current scam patterns, requires periodic manual curation to remain effective against evolving scam tactics. Scammers who are aware of the system's detection approach could potentially adapt their postings to avoid flagged keywords while preserving the fraudulent intent. An adaptive lexicon generation mechanism based on continuous web scraping and clustering of newly identified scam postings would mitigate this limitation.

Fourth, the system's false positive rate, while low in absolute terms, may generate user friction in cases where legitimate postings from less established companies trigger fraud alerts. This is particularly relevant for startup job postings, which may exhibit characteristics—such as non-corporate email domains, missing company profiles, and unusually high compensation—that overlap with fraudulent posting indicators. Contextual calibration based on company age and digital footprint depth would improve specificity in this scenario.

Fifth, the current implementation does not maintain a longitudinal profile of individual recruiters or companies, precluding the detection of fraud patterns that manifest across multiple postings or over extended time periods. A graph-based recruiter reputation system that tracks posting history, application response rates, and user-reported fraud incidents would substantially enhance detection capability for repeat offenders.

XV. FUTURE ENHANCEMENTS

Several promising directions for future development of the proposed system have been identified through empirical evaluation, user feedback, and analysis of current limitations. These enhancements are organized by implementation priority and expected impact.

In the near term, integration of transformer-based language models—specifically BERT, RoBERTa, or DeBERTa fine-tuned on job posting data—would likely yield measurable performance improvements, particularly for detecting nuanced scam patterns embedded in contextually coherent text. While computational cost is a current barrier to real-time deployment of transformer models, the progressive deployment of optimized transformer variants such as DistilBERT and TinyBERT may enable real-time inference at acceptable latency levels.

Multi-language support represents a critical enhancement for global applicability. The Arabic, Spanish, Portuguese, Hindi, and Mandarin job markets are among the largest globally, and scam postings targeting job seekers in these markets are prevalent and underserved by existing detection tools. Extending the system with multilingual embeddings (using models such as XLM-RoBERTa), language-specific keyword lexicons, and culturally appropriate risk communication would substantially expand the system's protective reach.

The development of a browser extension based on the existing API would enable real-time, in-browser analysis of job postings as users browse major job portals. The extension would inject scam probability indicators and SHAP explanation tooltips directly into the job portal interface, providing seamless protection without requiring users to navigate to a separate analysis tool. A Chrome Extension using the Manifest V3 API with a FastAPI backend would implement this architecture.

A mobile application for iOS and Android would enable job seekers to photograph physical job posting flyers, WhatsApp forwards, and social media posts for instant scam analysis. The mobile application would leverage device-local OCR capabilities augmented by server-side NLP processing, enabling basic offline functionality for users in areas with limited connectivity.

Blockchain-based company verification would provide a tamper-proof, decentralized mechanism for validating company legitimacy claims in job postings. By anchoring verified company records to a public blockchain and enabling job portals to query verification status in real-time, the system could substantially reduce the false positive rate for legitimate postings from established companies while making it computationally infeasible for scammers to fabricate verifiable company credentials.

Finally, cloud-native deployment using Kubernetes for container orchestration would enable elastic scaling to handle traffic surges during peak employment periods—such as graduate hiring seasons—without manual infrastructure intervention. Integration with cloud-based AutoML platforms would further automate the model retraining cycle, enabling the system to continuously adapt to emerging scam patterns without manual model engineering.

XVI. CONCLUSION

This paper has presented the design, implementation, and evaluation of an Explainable AI-based Smart Job Scam Detection System that addresses the growing threat of online recruitment fraud through a comprehensive, multi-modal, and transparent analytical platform. The system's hybrid feature engineering approach, combining TF-IDF vectorization of job posting text with a suite of 15 behavioural features derived from recruiter metadata, email domain verification, company profile analysis, and structural posting characteristics, has been shown to significantly outperform purely textual classification approaches on the Kaggle Fake Job Postings benchmark dataset.

The proposed ensemble classifier, combining XGBoost and Logistic Regression through soft voting, achieved an accuracy of 97.4%, an F1-Score of 96.3%, and an ROC-AUC of 0.989, representing a substantive improvement over baseline methods and competitive performance relative to computationally intensive deep learning approaches. The integration of SHAP-based explainability transforms the system from an opaque predictor into a transparent decision support tool that provides feature-attributed, natural language justifications for every classification decision.

The multi-modal input architecture supporting direct text entry, OCR-based screenshot analysis, and batch CSV processing substantially broadens the system's applicability beyond structured job portal data to encompass social media, messaging application, and bulk auditing scenarios. The interactive dashboard with real-time prediction visualization, aggregate trend analytics, and exportable reports further enhances the system's utility for both individual job seekers and institutional platform operators.

From a societal impact perspective, the proposed system represents a meaningful contribution to the growing imperative of consumer protection in digital employment ecosystems. By empowering job seekers with an accessible, transparent, and effective fraud detection tool, the system aims to reduce the financial losses, identity theft exposure, and psychological harm associated with employment scam victimization. The explainability dimension of the system additionally serves an educational function, equipping users with knowledge of the specific signals that distinguish fraudulent from genuine job postings and enhancing their resilience to scam attempts even outside the platform.

Future work will focus on extending the system with transformer-based NLP models, multi-language support, browser extension and mobile application development, blockchain-based company verification, and cloud-native deployment with automated model retraining. These enhancements will further increase the system's accuracy, accessibility, and global applicability, advancing the state of the art in AI-driven online fraud detection.

REFERENCES

[1] A. Amaar, W. Aljedaani, F. Rustam, E. Rupapara, and S. Lee, "Detection of Fake Job Postings by Utilizing Machine Learning and Natural Language Processing Approaches," Neural Processing Letters, vol. 54, no. 3, pp. 2219–2247, Jun. 2022.

[2] S. Vidros, C. Kolias, G. Kambourakis, and L. Akoglu, "Automatic Detection of Online Recruitment Frauds: Characteristics, Methods, and a Public Dataset," Future Internet, vol. 9, no. 1, p. 6, Mar. 2017.

[3] Z. Qazi, I. A. Qazi, K. Khushi, and S. Y. Irfan, "Fraud Job Posting Detection Using Machine Learning," in Proc. 6th Int. Conf. Future Networks & Distributed Systems (ICFNDS), New York, NY, USA, 2022, pp. 1–8.

[4] S. Priya, P. Divyashree, R. Harini, and S. Subhashini, "Fake Job Posting Detection Using Deep Learning," Int. J. Adv. Res. Comput. Sci., vol. 13, no. 3, pp. 45–52, May 2022.

[5] S. Mahbub, M. E. Pardede, A. A. Chowdhury, and R. Talevski, "Controlling Fake Job Advertisements on the Internet using BERT," in Proc. IEEE Int. Conf. e-Business Eng. (ICEBE), Shanghai, China, 2020, pp. 14–21.

[6] Y. Zhang and L. Zhao, "Behavioural Indicators of Fraudulent Recruitment: An Empirical Study," Int. J. Inf. Manag., vol. 58, p. 102278, Oct. 2021.

[7] R. Shalini and T. R. Babu, "Graph-Based Anomaly Detection for Fraudulent Job Postings," Comput. Sci. Eng. Int. J. (CSEIJ), vol. 12, no. 2, pp. 1–14, Apr. 2022.

[8] P. Kaur, M. Sharma, and M. Mittal, "Big Data and Machine Learning Based Secure Healthcare Framework," Procedia Comput. Sci., vol. 132, pp. 1049–1059, 2018.

[9] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 785–794.

[10] S. M. Lundberg and S. Lee, "A Unified Approach to Interpreting Model Predictions,"

in Advances in Neural Information Processing Systems, vol. 30, 2017, pp. 4765–4774.

[11] S. K. Sahu, A. Upadhyay, and S. Biswas, "Social Media Fraud Detection using OCR and NLP Techniques," in Proc. Int. Conf. Artificial Intelligence and Smart Systems (ICAIS), Coimbatore, India, 2021, pp. 874–881.

[12] M. Alotaibi and D. Roussinov, "Linguistic Style Markers in Online Employment Fraud Detection," Inf. Process. Manage., vol. 58, no. 4, p. 102573, Jul. 2021.

[13] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, Jun. 2002.

[15] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. 9th Int. Conf. Document Analysis and Recognition (ICDAR), Curitiba, Brazil, 2007, pp. 629–633.

[16] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, Minneapolis, MN, USA, 2019, pp. 4171–4186.

[17] M. Hasan, M. Rundensteiner, and E. Agu, "AUTOMATIC EMOTION DETECTION IN TEXT STREAMS BY ANALYZING TWITTER DATA," in Proc. IEEE Int. Conf. on Big Data and Smart Computing (BigComp), Shanghai, China, 2014.

[18] Federal Trade Commission, "Consumer Sentinel Network Data Book 2022," Federal Trade Commission, Washington, DC, USA, Tech. Rep., Feb. 2023. [Online]. Available: https://www.ftc.gov/reports/consumer-sentinel-network

[19] Kaggle, "Fake Job Postings Dataset," Kaggle Platform, 2020. [Online]. Available: https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction

[20] M. 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, San Francisco, CA, USA, 2016, pp. 1135–1144.

[21] D. Gunning and D. Aha, "DARPA's Explainable Artificial Intelligence (XAI) Program," AI Magazine, vol. 40, no. 2, pp. 44–58, 2019.

[22] A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems, vol. 30, 2017, pp. 5998–6008.

[23] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media, 2009.

[24] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed. O'Reilly Media, 2022.

[25] International Labour Organization, "World Employment and Social Outlook: Trends 2023," ILO, Geneva, Switzerland, 2023.

[26] P. Liu et al., "RoBERTa: A Robustly Optimized BERT Pretraining Approach," arXiv preprint arXiv:1907.11692, Jul. 2019.

[27] Z. Yang et al., "XLNet: Generalized Autoregressive Pretraining for Language Understanding," in Advances in Neural Information Processing Systems, vol. 32, 2019.

[28] C. C. Aggarwal and C. X. Zhai, "A Survey of Text Classification Algorithms," in Mining Text Data, C. C. Aggarwal and C. X. Zhai, Eds. Springer, 2012, pp. 163–222.

[29] European Commission, "Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act)," COM(2021) 206 final, Apr. 2021.

[30] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," in Proc. 14th Int. Joint Conf. on Artificial Intelligence (IJCAI), Montreal, Canada, 1995, pp. 1137–1143.