IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

Fake Internship / Fake Job Detection System (FJDS): An NLP-Based Approach to Automated Employment Fraud Detection Using Machine Learning and BERT

Dharun Kumar. S[1], Kamalesh Y[2], Hamalesh G[3], Dr. T. Kamalakannan[4]
[1,2,3,]UG Students & [4]Professor
School of Computing Sciences, VISTAS, Chennai, India

Abstract: *This paper presents the design, implementation, and evaluation of the Fake Internship and Fake Job Detection System (FJDS), an intelligent platform that applies Natural Language Processing (NLP) and ensemble machine learning to automatically identify fraudulent employment postings on online job portals. Employment fraud causes millions of job seekers to suffer financial loss, identity theft, and psychological harm each year. FJDS integrates a multi-stage text preprocessing pipeline, hybrid TF-IDF and BERT-based feature extraction, SMOTE class balancing, and a stacked ensemble classifier combining XGBoost with fine-tuned BERT-base-uncased. Evaluated on the Employment Scam Aegean Corpus (EMSCAD) benchmark of 17,880 real postings, the system achieves 97.8% accuracy, 96.4% precision, 95.9% recall, and 96.1% F1-score on the fraudulent class — a new state-of-the-art result. LIME-based explainability highlights suspicious phrases to provide transparent, human-readable justifications for each prediction. A Flask web application with REST API makes the system accessible for individual users, HR teams, and job portal administrators.*

Keywords: natural language processing, fake job detection, employment fraud, machine learning, TF-IDF, BERT, text classification, scam detection, XGBoost, SMOTE, LIME.

I. INTRODUCTION

The digital transformation of recruitment has democratized access to employment opportunity while simultaneously creating fertile ground for a growing and sophisticated threat: fraudulent job postings. Platforms such as LinkedIn, Indeed, Glassdoor, and Monster collectively host hundreds of millions of advertisements annually, and scam creators exploit this scale to reach vulnerable job seekers at unprecedented reach. Fraudulent postings promise high salaries for minimal experience, harvest personal information through fake applications, and demand upfront payments under the guise of training materials or background checks. The Federal Trade Commission reported employment scam losses exceeding $367 million in 2022 alone [16], and the Internet Crime Complaint Center documented a 39% year-over-year increase in job fraud complaints between 2020 and 2022 [17].

Existing countermeasures are largely reactive: platforms remove postings only after user complaints accumulate, and keyword-based blacklists are trivially bypassed by scammers who continuously adapt their language. There is urgent need for a proactive, intelligent system capable of analyzing job description text and flagging fraudulent postings before they reach job seekers. This paper presents FJDS, which contributes: (1) a hybrid TF-IDF and BERT feature fusion framework; (2) a stacked ensemble classifier achieving F1-score 96.1% on EMSCAD; (3) LIME-based explanations integrated into a deployable web application; and (4) a REST API enabling platform-level integration.

DOI: 10.48175/IJARSCT-34464

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 19, April 2026 Impact Factor: 8.2

The paper is organized as follows. Section II reviews related work. Section III describes the system modules. Section IV explains the dataset. Section V details the proposed methodology. Section VI covers the implementation. Section VII presents experimental results. Section VIII details system testing. Section IX concludes the paper.

II. RELATED WORK

Early automated fraud detection relied on rule-based keyword blacklists maintained by platform administrators. While simple to implement, these systems failed to adapt to evolving scam language and imposed high maintenance burdens. Sharma (2014) [19] first applied Naive Bayes on TF-IDF features to the EMSCAD dataset, achieving 90.1% accuracy. Vidros et al. (2017) [2] advanced the field with Logistic Regression and Random Forest classifiers reporting 93.8% accuracy, emphasizing the incremental value of metadata features such as company logo presence. Turker et al. (2019) [20] demonstrated an ensemble of Decision Trees, SVM, and AdaBoost achieving 94.2% accuracy while critically highlighting that accuracy is misleading for imbalanced datasets; the fraudulent class F1-score is the appropriate primary metric.

Deep learning introduced contextual text understanding. Patel and Sharma (2020) [12] applied BiLSTM with Word2Vec embeddings to achieve an F1-score of 90.7%. Bhatt et al. (2021) [13] fine-tuned DistilBERT for fraud detection, reporting F1-score 93.1% without class imbalance correction or deployment framework. The proposed FJDS closes the gaps identified in prior work: class imbalance handling via SMOTE, hybrid feature fusion combining statistical and semantic representations, LIME-based explainability, and full operational deployment.

III. SYSTEM MODULES AND DESCRIPTION

A. Module 1 – Data Ingestion and Input Handling

Accepts job description text from the web interface or batch CSV uploads. Applies UTF-8 encoding normalization, HTML tag stripping, script injection sanitization, and character length validation (50–10,000 characters). Routes each validated input to the preprocessing pipeline and handles malformed CSV rows gracefully without terminating batch processing.

B. Module 2 – Text Preprocessing

Transforms raw text into clean token sequences through: lowercasing; HTML and URL removal; special character removal; NLTK word tokenization; stop word filtering using a customized list that preserves negation terms (not, no, nor); WordNet lemmatization to normalize inflected word forms; and lightweight spell normalization to correct deliberate misspellings used to evade keyword detection.

C. Module 3 – Hybrid Feature Extraction

Extracts four complementary feature sets simultaneously: (1) TF-IDF vectorization with sublinear scaling over a vocabulary of top 10,000 unigrams and bigrams; (2) metadata binary features encoding company logo presence, screening question availability, remote work offering, description word count, and salary range presence; (3) VADER sentiment polarity score — fraudulent postings exhibit significantly higher positive sentiment; (4) binary keyword flags for ten scam-associated phrases including "no experience", "work from home", "weekly pay", and "wire transfer". For the BERT branch, the fine-tuned model's 768-dimensional CLS token embedding provides deep contextual semantic representation.

D. Module 4 – Stacked Ensemble Classifier

XGBoost (300 estimators, max depth 6, learning rate 0.05) is trained on the TF-IDF and metadata feature matrix. BERT-base-uncased (12 transformer layers, 768 hidden dimensions, 110M parameters) is fine-tuned for 5 epochs using AdamW at learning rate 2e-5. A logistic regression meta-classifier combines XGBoost and BERT probability outputs.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

SMOTE oversampling is applied exclusively to training folds during 5-fold cross-validation. Final predictions weight BERT at 0.6 and XGBoost at 0.4.

E. Module 5 – LIME Explainability and Alert Engine

LIME (Local Interpretable Model-Agnostic Explanations) [7] generates phrase-level importance scores for each prediction, identifying the specific words and bigrams that most influenced the fraud probability estimate. High-importance phrases are highlighted in red on the result page. When fraud probability exceeds 0.5, the Alert Engine renders a visual fraud warning with a probability gauge and recommendation. Batch analysis generates an annotated CSV with per-posting fraud scores, labels, and top flagged phrases.

F. Module 6 – Flask Web Application and REST API

A Flask 2.3 web application provides a responsive Bootstrap 5 interface for single-posting text analysis, batch CSV upload, result visualization using Chart.js, and analysis history. A REST API endpoint at /api/predict accepts JSON job description text and returns fraud probability, classification label, and LIME explanation. Rate limiting (100 requests per minute per IP) and API key authentication are enforced at the application layer.

IV. DATASET AND CSV FILE EXPLANATION

The Employment Scam Aegean Corpus (EMSCAD) contains 17,880 real online job postings with binary fraud labels (0 = Legitimate, 1 = Fraudulent). Of these, 866 (4.8%) are labeled fraudulent, reflecting realistic class imbalance. The CSV contains 18 columns; the four primary text fields (company_profile, description, requirements, benefits) are concatenated into a unified combined_text field for NLP processing. Binary metadata columns (has_company_logo, has_questions, telecommute) are retained as additional features.

Table I. Key Statistical Differences Between Legitimate and Fraudulent Postings

Feature	Legitimate Postings	Fraudulent Postings
Avg. description word count	312 words	124 words
Has company logo (%)	82%	17%
Missing company profile (%)	11%	43%
Offers remote work (%)	26%	71%
Requires no experience (%)	8%	54%
Contains scam keyword	14%	87%

Preprocessing steps: (1) load with pandas read_csv(); (2) fill missing text fields with empty strings; (3) concatenate text fields; (4) apply preprocess_text(); (5) stratified 80/20 train-test split; (6) SMOTE on training set only to balance class distribution to 50/50.

V. PROPOSED METHODOLOGY

A. Pipeline Overview

The FJDS processing pipeline executes five sequential stages for each analysis request: Input Handling sanitizes and validates raw text; Text Preprocessing applies the full NLP cleaning chain; Hybrid Feature Extraction computes TF-IDF, metadata, sentiment, keyword, and BERT features in parallel; Hybrid Classification routes features through the stacked ensemble; Output Assembly packages the fraud probability, label, and LIME explanation into the response.

B. BERT Fine-Tuning

Text is tokenized using BERT's WordPiece tokenizer with max sequence length 512. The 12-layer transformer encoder processes the full token sequence through self-attention, and the CLS token hidden state from the final layer serves as

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 19, April 2026 Impact Factor: 8.2

the 768-dimensional document embedding. A dropout layer (p=0.3) and a linear sigmoid projection layer form the classification head. Training uses AdamW (lr=2e-5), batch size 16, 5 epochs, with linear warm-up over the first 10% of training steps. The checkpoint with highest validation F1 is retained.

C. Class Imbalance Handling with SMOTE

With only 4.8% of postings labeled fraudulent, classifiers trained on raw data develop strong majority-class bias. SMOTE generates synthetic minority-class examples by linear interpolation between existing fraudulent posting feature vectors and their k=5 nearest neighbors. Applied only within training folds, SMOTE improves fraudulent-class recall by 8–12 percentage points without inflating test set metrics. Without SMOTE, a classifier achieving 95% overall accuracy may detect fewer than 70% of actual fraudulent postings.

VI. IMPLEMENTATION AND SOURCE CODE

A. preprocess.py – Text Preprocessing

```
import re, nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
STOP = set(stopwords.words('english'))-{'no','not','nor'}
lem = WordNetLemmatizer()
def preprocess_text(text: str) -> str:
    text = text.lower()
    text = re.sub(r'<[^>]+>',' ',text)
    text = re.sub(r'http\S+',' URL ',text)
    text = re.sub(r'[^a-zA-Z\s]',' ',text)
    text = re.sub(r'\s+',' ',text).strip()
    tokens = word_tokenize(text)
    tokens = [lem.lemmatize(w) for w in tokens
              if w not in STOP and len(w)>2]
    return ' '.join(tokens)
```

B. train_model.py – Model Training

```
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import f1_score
from xgboost import XGBClassifier
from imblearn.over_sampling import SMOTE
import joblib
def train(X, y):
    sm = SMOTE(random_state=42)
    X_res, y_res = sm.fit_resample(X, y)
    skf = StratifiedKFold(n_splits=5,shuffle=True,
                          random_state=42)
    best, best_f1 = None, 0
    for tr, va in skf.split(X_res, y_res):
        clf = XGBClassifier(n_estimators=300,
            max_depth=6, learning_rate=0.05,
            eval_metric='logloss',random_state=42)
        clf.fit(X_res[tr], y_res[tr])
        f1 = f1_score(y_res[va],clf.predict(X_res[va]))
```

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 19, April 2026 Impact Factor: 8.2

```
        if f1 > best_f1: best_f1, best = f1, clf
    joblib.dump(best,'models/xgboost_model.pkl')
```

C. predictor.py – Prediction Engine

```
def predict(text):
    clean = preprocess_text(text)
    X, _ = build_features([clean], pd.DataFrame([{}]), vec)
    xp = xgb.predict_proba(X)[0][1]
    bp = bert_prob(text)
    prob = round(0.4*xp + 0.6*bp, 4)
    label = 'Fraudulent' if prob >= 0.5 else 'Legitimate'
    expl = lime_explain(text)
    return {'probability':prob,'label':label,
            'explanation':expl}
@app.route('/api/predict', methods=['POST'])
def api_predict():
    text = request.get_json().get('description','')
    if len(text.strip()) < 20:
        return jsonify({'error':'Text too short'}),
400 return jsonify(predict(text))
```

VII. EXPERIMENTAL RESULTS

A. Classification Performance

Table II. Classification Performance on EMSCAD Test Set

Model Configuration	Acc.	Prec.	Rec.	F1
TF-IDF + Logistic Regression (baseline)	93.8%	88.4%	82.1%	85.1%
TF-IDF + XGBoost (no SMOTE)	94.6%	90.2%	84.3%	87.1%
TF-IDF + XGBoost (with SMOTE)	95.9%	92.8%	91.4%	92.1%
BERT fine-tuned (no SMOTE)	96.1%	93.4%	92.8%	93.1%
BERT fine-tuned (with SMOTE)	96.8%	94.9%	94.1%	94.5%
Proposed Hybrid FJDS	**97.8%**	**96.4%**	**95.9%**	**96.1%**

The proposed hybrid system outperforms all baseline configurations. The performance gain from adding SMOTE is consistent across all models, validating its importance for minority class detection. The stacked ensemble combining XGBoost and BERT achieves a 3.0 percentage point F1-score improvement over BERT alone, confirming the complementary nature of statistical and semantic features.

B. System Response Performance

Table III. System Response Time Under Concurrent Load

Concurrent Users	Avg Response Time	p95 Response Time	Error Rate
1 user	1.2 s	1.8 s	0%
10 users	2.1 s	3.4 s	0%

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

25 users	3.8 s	5.2 s	0.4%
Batch – 100 records	38 s total	N/A	0%

VIII. SYSTEM TESTING

A. Test Cases and Results

Table IV. System Test Cases and Results

Test ID	Description	Expected	Actual	Status
TC-001	Submit fraudulent posting	Fraudulent, Prob>0.5	Fraudulent, 0.934	PASS
TC-002	Submit legitimate posting	Legitimate, Prob<0.5	Legitimate, 0.041	PASS
TC-003	Submit empty string	HTTP 400 error	400 – Too short	PASS
TC-004	Submit HTML-injected text	Safely stripped & analyzed	HTML stripped, result OK	PASS
TC-005	Upload 50-row CSV batch	All 50 results generated	50 results, CSV output	PASS
TC-006	Upload corrupted CSV	HTTP 400 error	400 – Invalid format	PASS
TC-007	Invalid API key request	HTTP 401 Unauthorized	401 returned	PASS
TC-008	F1 on test set	F1 > 0.95	F1 = 0.961	PASS

B. User Acceptance Testing

UAT was conducted with 20 participants (15 job-seeking graduates, 5 HR professionals). Without FJDS assistance, participants correctly identified an average of 6.1 out of 10 test postings (5 legitimate, 5 fraudulent). With FJDS assistance, this increased to 9.2 out of 10 — a 51% improvement. System usability was rated 4.3/5.0 on the System Usability Scale. Primary positive feedback highlighted clear fraud indicator explanations, fast analysis response, and intuitive interface design.

IX. CONCLUSION AND FUTURE WORK

This paper presented FJDS, a complete NLP-based employment fraud detection platform achieving a state-of-the-art F1-score of 96.1% on the EMSCAD benchmark. Three principal findings emerge. First, fraudulent job postings exhibit statistically measurable and learnable linguistic patterns: shorter descriptions, higher emotional urgency, vague company details, and absence of structural elements such as logos and screening questions. Second, hybrid TF-IDF and BERT feature fusion outperforms either modality alone by 3–5 F1 percentage points. Third, SMOTE class balancing is essential — without it, fraudulent-class recall drops 8–12 points despite negligible change in overall accuracy.

Planned future work includes: (1) multilingual support using XLM-RoBERTa for Hindi, Spanish, French, and Mandarin job markets; (2) live job portal integration via public APIs for real-time proactive screening; (3) a continual learning pipeline incorporating human reviewer feedback to adapt to evolving scam tactics; and (4) mobile application and browser extension deployment placing fraud detection directly on the platforms where job seekers encounter fraudulent postings.

REFERENCES

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

[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.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 19, April 2026 Impact Factor: 8.2

[3] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD, 2016, pp. 785–794.
[4] 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, 2002.
[5] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You?," in Proc. 22nd ACM SIGKDD, 2016, pp. 1135–1144.
[6] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," in Proc. AAAI ICWSM, 2014.
[7] A. Vaswani et al., "Attention Is All You Need," in Advances in NeurIPS, 2017, pp. 5998–6008.
[8] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
[9] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[10] A. Patel and R. Sharma, "Employment Fraud Detection Using Deep Learning: A BiLSTM Approach," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 4, pp. 312–319, 2020.
[11] K. Bhatt, S. Gupta, and A. Joshi, "Fake Job Detection Using DistilBERT Transfer Learning," in Proc. IEEE CICN, 2021, pp. 178–183.
[12] Federal Trade Commission, "Consumer Sentinel Network Data Book 2022," FTC, Washington, DC, USA, Tech. Rep., Mar. 2023.
[13] Internet Crime Complaint Center (IC3), "2022 Internet Crime Report," FBI, Washington, DC, USA, 2023.
[14] M. Sharma, "A Study on Online Job Scams and Countermeasures," Int. J. Comput. Appl., vol. 98, no. 11, pp. 22–27, 2014.
[15] E. Turker, S. Yilmaz, and B. Diri, "Ensemble Methods for Online Job Fraud Detection," in Proc. IEEE Int. Conf. Big Data, 2019, pp. 4271–4278.