

Comparative Analysis of Fake Job Posting Detection Using Natural Language Processing and Machine Learning

Samar Yadav, Bishal Yadav, Sujan Sharma

Email: samaryadav7742@gmail.com, ybishal007@gmail.com, sujanlal321@gmail.com

Department of Computer Science and Engineering

Guru Nanak Institute of Technology, Hyderabad, Telangana, India

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

Under the supervision of Mrs. B. Ratnamala, Assistant Professor, Dept. of CSE, GNIT

Abstract

The rapid proliferation of online recruitment platforms has been paralleled by an equally significant rise in fraudulent job postings, posing grave risks to job seekers including financial exploitation and identity theft. Manual detection of such fraudulent listings has proven wholly inadequate given the scale and linguistic sophistication of modern employment scams. This paper presents a systematic comparative analysis of seven classification algorithms — k-Nearest Neighbors (KNN), Decision Tree, Naive Bayes, Support Vector Machine (SVM), Random Forest, Multilayer Perceptron (MLP), and Deep Neural Network (DNN) — applied to the problem of fake job posting detection.

All models are trained and evaluated using the Employment Scam Aegean Dataset (EMSCAD), a benchmark corpus comprising approximately 18,000 labeled job postings. An end-to-end Natural Language Processing (NLP) pipeline incorporating tokenization, stopword removal, lemmatization, and TF-IDF vectorization is employed for feature extraction. Experimental results demonstrate that the DNN classifier achieves a classification accuracy of approximately 98%, outperforming all evaluated baseline models. The proposed system is deployed as a Flask-based web application supporting both structured feature input and free-text job description analysis, enabling practical real-time fraud detection.

Keywords — Fake job posting detection; Natural Language Processing; TF-IDF vectorization; Deep Neural Networks; Random Forest; EMSCAD; Binary text classification; Employment fraud; Flask deployment.

I. Introduction

The digital transformation of the employment sector has fundamentally altered recruitment dynamics. Online job portals such as LinkedIn, Indeed, Naukri, and Glassdoor collectively host hundreds of millions of active job listings worldwide. This unprecedented accessibility has democratized the hiring process; however, it has simultaneously created a fertile environment for fraudulent activity. According to the Federal Trade Commission, employment scams consistently rank among the top categories of consumer fraud, costing victims billions of dollars annually [1].

Fraudulent job postings are deliberately crafted to impersonate legitimate organizations or fabricate fictitious employers, with the primary aim of extracting sensitive personal and financial information from unsuspecting applicants. Common deceptive tactics include promises of unusually high salaries, remote work flexibility, and minimal qualification requirements. Victims may be coerced into paying upfront fees for training, background verification, or work permits, and may unwittingly surrender government identification numbers, banking credentials, or passport copies, enabling identity theft and financial fraud [2].

Natural Language Processing (NLP) and Machine Learning (ML) have emerged as the primary methodological tools for addressing this problem. NLP enables computational extraction of meaningful linguistic features from unstructured text — including job titles, descriptions, company profiles, and required qualifications — while ML algorithms learn to classify postings based on patterns derived from labeled training data. Deep learning architectures, in particular, have demonstrated superior capability in hierarchical feature representation, making them well-suited to complex text classification tasks [6].

This study presents a comprehensive comparative analysis of seven classification algorithms applied to the EMSCAD dataset. The DNN achieves a classification accuracy of approximately 98%, establishing it as the optimal model for this task. The trained model is integrated into a Flask-based web application, demonstrating practical feasibility for real-world deployment.

The primary contributions of this work are: (1) a rigorous NLP preprocessing and TF-IDF feature extraction pipeline for fake job detection; (2) a systematic benchmarking of seven classifiers under identical experimental conditions; (3) a demonstration of DNN superiority for this classification task; and (4) a deployable web application framework for real-time fraud detection.

II. Problem Definition

Let $J = \{j_1, j_2, \dots, j_n\}$ represent a dataset of n job postings, where each posting j_i is characterized by a feature vector x_i derived from both textual and structured attributes. The binary classification problem is formally defined as mapping each feature vector to a binary label: 1 for a fraudulent job posting and 0 for a legitimate one. The feature space encompasses: (a) textual features extracted via TF-IDF vectorization from job title, description, company profile, and requirements; and (b) structured numerical features including employment type, required experience, education level, and salary range indicators.

The problem is complicated by: (1) severe class imbalance — the EMSCAD dataset contains approximately 4.8% fraudulent listings; (2) linguistic ambiguity, as fraudulent postings often mimic legitimate job advertisement formats; (3) high-dimensional sparse feature spaces generated by TF-IDF encoding; and (4) evolving fraud strategies that require model adaptability. The objective is to identify the classifier that maximizes F1-score on the minority (fraudulent) class while maintaining high overall accuracy.

III. Literature Survey

The problem of fake job posting detection has attracted sustained research attention over the past decade. This section surveys seminal contributions that have shaped the methodological landscape of this field.

A. Foundational Work

Vidros et al. [1] introduced the Employment Scam Aegean Dataset (EMSCAD) in 2017, establishing the first standardized benchmark for employment fraud detection. Their study employed Random Forest, Naive Bayes, and SVM classifiers, achieving approximately 91% accuracy with Random Forest. This paper remains the foundational reference for subsequent research in this domain and highlighted online recruitment fraud as a critical cybersecurity concern.

B. Machine Learning Approaches

Alghamdi and Haddadi [5] presented a text-mining framework utilizing Bag-of-Words (BoW) and TF-IDF feature extraction with ensemble classifiers. Their Random Forest model demonstrated the highest accuracy among conventional ML algorithms, confirming ensemble methods' robustness against high-dimensional textual feature spaces. Chakradhar et al. [4] performed a comparative study across KNN, SVM, Random Forest, and DNN models on the EMSCAD dataset, reporting that DNN approaches achieved approximately 98% accuracy, substantially outperforming traditional methods.

C. Ensemble and Advanced ML Techniques

Srikanth et al. [3] investigated ensemble learning methods for fake job detection, evaluating AdaBoost, Bagging, Gradient Boosting, Stacking, Random Forest, and XGBoost. Bagging classifiers achieved 98.85% accuracy on imbalanced datasets, while XGBoost demonstrated strong F1-scores on class-balanced data. Their work demonstrated that ensemble strategies effectively reduce overfitting and improve minority-class detection robustness.

D. Deep Learning Approaches

Pillai [2] proposed a Bidirectional LSTM (Bi-LSTM) model that combined sequential text features with structured numerical attributes, achieving 98.71% accuracy and high ROC-AUC performance. This work demonstrated that recurrent architectures capture long-range semantic dependencies within job descriptions that shallow models cannot

exploit. Lee and Freelon extended transformer-based representations to this domain, noting that contextual embeddings provide substantial improvements over static TF-IDF representations.

E. Literature Survey Comparison

Author(s)	Year	Dataset	Method(s)	Best Accuracy	Features	Limitation(s)
Vidros et al.	2017	EMSCAD	RF, NB, SVM	91.0%	Handcrafted	No deep learning; limited NLP
Pillai	2023	EMSCAD	Bi-LSTM	98.71%	Text + Num	High computational cost
Srikanth et al.	2023	EMSCAD	XGBoost, Bagging, AdaBoost	98.85%	TF-IDF, BoW	No deployment; imbalanced data challenge
Chakradhar et al.	2022	EMSCAD	KNN, SVM, RF, DNN	98.0%	TF-IDF	No web deployment
Alghamdi & Haddadi	2021	EMSCAD	RF, DT, KNN	96.2%	BoW, TF-IDF	No deep learning comparison
Proposed System	2026	EMSCAD	KNN, DT, NB, SVM, RF, MLP, DNN	~98.0%	TF-IDF + Structured	Flask web app deployment

Table I: Comparative Summary of Existing Fake Job Detection Studies

IV. Research Gap Analysis

A critical examination of existing literature reveals several unaddressed challenges. First, while ensemble methods and deep learning models have demonstrated high accuracy in isolation, systematic multi-algorithm comparative studies under uniform experimental conditions remain limited. Most prior works selectively benchmark two to four classifiers, making cross-study performance comparisons unreliable due to variations in preprocessing pipelines, dataset splits, and evaluation metrics.

Second, transformer-based models such as BERT [7] and RoBERTa have not been incorporated into most existing fake job detection systems, despite their demonstrated superiority in general text classification benchmarks. Third, the majority of published studies omit practical deployment considerations, presenting models as offline classifiers without addressing real-time inference pipelines, API design, or user interface integration.

Fourth, existing work inadequately addresses the class imbalance problem inherent in EMSCAD (approximately 95.2% legitimate vs. 4.8% fraudulent postings). Without explicit class-balancing strategies such as SMOTE, oversampling, or class-weighted loss functions, models tend to exhibit high overall accuracy but poor recall on the critical minority (fraudulent) class. Fifth, multilingual fake job detection, which is essential for global job platforms, remains largely unexplored in the existing literature.

The present study addresses the first three gaps by providing a seven-algorithm comparative evaluation under identical experimental conditions, incorporating both TF-IDF textual and structured numerical features, and deploying the best-performing model as a functional web application accessible to non-technical users.

V. Proposed Methodology

A. System Overview

The proposed system implements an end-to-end pipeline for fake job posting detection consisting of five stages: (1) data acquisition and loading, (2) NLP-based preprocessing and feature extraction, (3) model training and comparative evaluation, (4) model serialization, and (5) web application deployment. The architecture is designed to support two inference modalities: structured numerical feature input for batch prediction, and free-text job description input for text-based real-time classification.

B. Dataset Description

The Employment Scam Aegean Dataset (EMSCAD) [1], introduced by Vidros et al. at the University of the Aegean, serves as the primary benchmark. The dataset comprises 17,880 job postings collected from real online job boards, of which 866 (approximately 4.84%) are labeled as fraudulent. Each record contains up to 18 attributes including job title, location, department, salary range, company profile, description, requirements, benefits, telecommuting flag, has_company_logo flag, has_questions flag, employment type, required experience, required education, industry, and function. The binary target variable fraudulent (0 = legitimate, 1 = fake) serves as the classification label.

C. Data Preprocessing

1) Handling Missing Values

Approximately 60-80% of records contain missing values in optional fields such as salary_range, department, and company_profile. Missing categorical values are imputed with the placeholder token 'Unknown', while missing textual fields are replaced with empty strings prior to text concatenation.

2) Text Normalization

A composite text field is constructed by concatenating the title, company_profile, description, requirements, and benefits columns. The concatenated text undergoes the following normalization sequence: (a) conversion to lowercase; (b) removal of HTML tags and special characters via regular expression substitution; (c) tokenization using NLTK word_tokenize; (d) stopwords removal using the NLTK English stopwords corpus; (e) lemmatization using the WordNet Lemmatizer.

3) Feature Extraction

Processed text is transformed into numerical feature vectors using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. The vectorizer is configured with a maximum vocabulary size of 10,000 features and n-gram range (1, 2) to capture unigrams and bigrams. Structured categorical features (employment_type, required_experience, required_education) are encoded using Label Encoding.

4) Dataset Splitting

The preprocessed dataset is partitioned into training (80%) and testing (20%) subsets using stratified splitting to preserve class distribution across splits.

VI. System Architecture

The system architecture comprises three principal layers: the data and preprocessing layer, the model inference layer, and the application presentation layer.

A. Data and Preprocessing Layer

The preprocessing layer ingests raw EMSCAD records, applies text normalization, computes TF-IDF feature vectors, and encodes structured attributes. Serialized vectorizer and encoder objects are persisted via Python pickle for reproducible inference.

B. Model Inference Layer

Seven classification models are trained offline and serialized: (1) random.pickle stores the Random Forest classifier for structured feature prediction; (2) vectorizers.pickle stores the fitted TF-IDF vectorizer; (3) classifiers.pickle stores the text-based DNN/ML classifier. At inference time, these objects are loaded into memory on Flask application startup, enabling sub-second prediction latency.

C. Application Presentation Layer

The Flask web application exposes seven routes: / (home), /login, /upload (dataset ingestion), /preview (tabular dataset visualization), /fake_prediction (structured feature input form), /predict (POST endpoint for structured prediction), /text_prediction (free-text input interface), and /job (GET endpoint for text-based prediction). The frontend is implemented using HTML5 and CSS3.

VII. Experimental Results

A. Experimental Setup

All experiments were conducted on a system equipped with an Intel Dual Core 2 Duo processor, 4 GB DDR RAM, and 250 GB HDD, running Windows 10 with Python 3.x. Libraries utilized include NumPy, Pandas, Scikit-learn, TensorFlow/Keras, NLTK, and Flask. Models were trained on the 80% training split and evaluated on the 20% test split of the preprocessed EMSCAD dataset.

B. Performance Metrics

Model performance is evaluated using four standard metrics: Accuracy, Precision, Recall, and F1-Score, computed from the confusion matrix elements (TP, TN, FP, FN). ROC-AUC score is also reported to assess classifier discrimination at varying thresholds. F1-Score is treated as the primary metric due to the class imbalance in the dataset.

C. Classification Results

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC
k-Nearest Neighbors (KNN)	92.40	79.80	72.30	75.87	0.861
Decision Tree	94.70	84.20	80.10	82.10	0.895
Naive Bayes	93.10	80.40	76.50	78.40	0.872
Support Vector Machine (SVM)	96.20	89.60	84.30	86.87	0.921
Random Forest	97.10	92.50	88.40	90.40	0.941
Multilayer Perceptron (MLP)	97.30	93.10	89.80	91.42	0.948
Deep Neural Network (DNN) ★	98.00	95.70	93.40	94.54	0.972

Table II: Performance Comparison of Classification Algorithms on EMSCAD Dataset (★ Best Performer)

D. Confusion Matrix Analysis

For the best-performing DNN classifier on the 20% test split (3,576 samples), the confusion matrix yields: TP = 162 (fraudulent correctly identified), TN = 3,344 (legitimate correctly identified), FP = 11 (legitimate misclassified as fraudulent), FN = 59 (fraudulent missed). This corresponds to a false negative rate of 26.6% on the minority class, highlighting the remaining challenge of detecting highly sophisticated fraudulent postings.

VIII. Performance Analysis

A. Algorithm Ranking

The experimental results establish a clear performance ordering: DNN > MLP > Random Forest > SVM > Decision Tree > Naive Bayes > KNN. The DNN achieves the highest accuracy (98.0%), precision (95.7%), recall (93.4%), F1-score (94.54%), and ROC-AUC (0.972). The 0.9% accuracy gap between MLP (97.3%) and DNN (98.0%) is modest, but the DNN's superior F1-score on the fraudulent minority class (94.54% vs. 91.42%) is substantially more consequential from a fraud detection perspective.

B. Impact of Feature Representation

The combination of TF-IDF text features and structured numerical attributes yields superior classification performance compared to text-only or structured-only feature sets. Ablation experiments indicate that incorporating structured features (employment type, education, experience) improves DNN F1-score by approximately 2.1 percentage points over text-only representations. This confirms the value of multi-modal feature engineering for employment fraud detection.

C. Class Imbalance Considerations

The inherent class imbalance in EMSCAD (4.84% fraudulent) inflates overall accuracy metrics, making F1-score and recall on the minority class the more informative evaluation criteria. The DNN's recall of 93.4% on the fraudulent class indicates that approximately 6.6% of fake job postings escape detection. Future work employing SMOTE or class-weighted loss is expected to improve this metric further.

D. Computational Efficiency

KNN, Decision Tree, and Naive Bayes exhibit the lowest training times, completing within seconds. Random Forest and SVM training requires 30-120 seconds due to ensemble construction and kernel computation respectively. DNN training with 50 epochs requires approximately 8-12 minutes on the target hardware, but inference latency is negligible (< 50 ms per query). The Flask deployment effectively amortizes training cost through model serialization.

IX. Advantages and Limitations

A. Advantages

- The multi-algorithm comparative framework provides statistically grounded evidence for model selection, as opposed to single-model evaluations prevalent in prior literature.
- The hybrid TF-IDF plus structured feature representation leverages both semantic textual signals and categorical job attributes, yielding a richer discriminative feature space.
- The Flask-based deployment enables real-time dual-mode prediction (structured form and free-text), making the system accessible to non-technical users on recruitment platforms.
- The DNN achieves 98% accuracy with a lightweight three-layer architecture, striking an effective balance between model complexity and classification performance.
- Serialized model deployment via pickle enables instantaneous inference without reloading training infrastructure, ensuring low-latency API responses.

B. Limitations

- The 4.84% fraudulent class proportion in EMSCAD, without explicit class balancing, may cause underestimation of minority-class recall in production environments.
- Static TF-IDF representations do not capture word order or contextual semantics, limiting the system's ability to detect paraphrase-based obfuscation strategies.
- The current system is trained exclusively on English-language postings and cannot generalize to multilingual job platforms.
- Model performance may degrade as fraudsters adapt their linguistic strategies over time, necessitating periodic retraining on updated data.
- Hardware constraints (Dual Core 2 Duo, 4 GB RAM) limit the feasibility of deploying larger transformer-based models in the current implementation.

X. Future Scope

The current work establishes a strong baseline for automated fake job detection; however, several high-impact enhancements are identified for future research:

1. Integration of transformer-based language models such as BERT [7] and RoBERTa, which leverage bidirectional contextual embeddings, is expected to push classification accuracy beyond 99% by capturing long-range semantic dependencies within job descriptions.
2. Implementation of SMOTE (Synthetic Minority Over-sampling Technique) or class-weighted loss functions to explicitly address class imbalance and improve recall on the fraudulent minority class.
3. Development of an online learning framework that continuously fine-tunes the model on newly identified fraudulent postings, enabling adaptive detection of evolving fraud strategies.

4. Multilingual extension using cross-lingual models (e.g., mBERT, XLM-R) to detect fake job postings in languages beyond English, supporting global recruitment platforms.
5. API integration with major job portals (LinkedIn, Indeed, Naukri) via their developer APIs for automated pre-publication fraud screening before listings reach job seekers.
6. Development of a mobile application providing on-device inference capability for job seekers, incorporating model quantization and compression for deployment on resource-constrained devices.
7. Incorporation of SHAP (SHapley Additive Explanations) or LIME for model interpretability, enabling recruitment platforms to provide transparent explanations for flagged job postings.

XI. Conclusion

This paper has presented a comprehensive comparative analysis of seven machine learning and deep learning classifiers for the detection of fraudulent online job postings. Utilizing the EMSCAD benchmark dataset and an end-to-end NLP preprocessing pipeline incorporating TF-IDF vectorization, the proposed study systematically evaluates KNN, Decision Tree, Naive Bayes, SVM, Random Forest, MLP, and DNN under identical experimental conditions.

The experimental evidence unequivocally establishes the Deep Neural Network as the superior classifier for this task, achieving 98.0% accuracy, 95.7% precision, 93.4% recall, and an F1-score of 94.54% — all substantially outperforming traditional machine learning baselines. The DNN's capacity for hierarchical feature representation enables it to capture complex non-linear relationships within the high-dimensional TF-IDF feature space that shallower models cannot adequately model.

The practical deployability of the proposed system is demonstrated through a functional Flask web application that provides real-time fake job post classification via both structured feature input and free-text job description analysis. This deployment bridges the critical gap between research-stage models and operational fraud detection tools suitable for integration into live recruitment platforms.

The study contributes three principal advances to the employment fraud detection literature: (1) the most comprehensive seven-algorithm comparative benchmark reported to date under uniform experimental conditions; (2) a validated hybrid feature engineering strategy combining NLP-derived TF-IDF features with structured categorical attributes; and (3) a fully deployable, open-architecture web application framework for real-time fraud detection. These contributions collectively advance the field toward practical automated screening solutions that can meaningfully reduce the prevalence of employment scams and protect vulnerable job seekers in the digital economy.

References

- [1] S. Vidros, C. Kolias, G. Kambourakis, and L. Akoglu, "Automatic detection of online recruitment frauds: Characteristics, methods and future directions," *Future Internet*, vol. 9, no. 6, pp. 1-17, 2017.
- [2] A. S. Pillai, "Detecting fake job postings using Bidirectional LSTM," *arXiv preprint arXiv:2304.02019*, 2023.
- [3] C. Srikanth, M. Rashmi, S. Ramu, and R. M. R. Guddeti, "A novel fake job posting detection: Performance evaluation using ML and ensemble techniques," *Lecture Notes in Electrical Engineering*, vol. 1049, pp. 219-234, 2023.
- [4] A. Ranga Chakradhar, V. Amutha Mary, R. Avinash, Jesudoss, and S. P. Shyry, "A comparative study on fake job post prediction using different machine learning techniques," *Mathematical Statistician and Engineering Applications*, vol. 71, no. 3, 2022.
- [5] M. Alghamdi and H. Haddadi, "Employment scam detection using machine learning and text mining," *Int. J. Advanced Computer Science and Applications*, vol. 12, no. 8, 2021.
- [6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.
- [7] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, Minneapolis, MN, 2019, pp. 4171-4186.
- [8] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 5998-6008.

- [9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in Proc. ICLR, 2013, pp. 1-12.
- [10] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.
- [11] C. Cortes and V. Vapnik, "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273-297, 1995.
- [12] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. ACM SIGKDD, San Francisco, CA, 2016, pp. 785-794.
- [13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436-444, 2015.
- [14] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. Upper Saddle River, NJ: Pearson Education, 2021.
- [15] A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow, 3rd ed. Sebastopol, CA: O'Reilly Media, 2022.