# Saurabh Sarkar Saurabh Sarkar

## Saurabh Sarkar- Afridi_Draftpaper_Project.pdf

Quick Submit

Quick Submit

Presidency University

### Document Details

**Submission ID**
trn:oid:::1:3406262794

**Submission Date**
Nov 11, 2025, 9:14 AM GMT+5:30

**Download Date**
Nov 11, 2025, 9:38 AM GMT+5:30

**File Name**
Afridi_Draftpaper_Project.pdf

**File Size**
477.5 KB

9 Pages

3,290 Words

21,234 Characters

# 71% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

**Caution: Review required.**

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

## Detection Groups

**13** AI-generated only 71%
Likely AI-generated text from a large-language model.

**0** AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.



## Frequently Asked Questions

**How should I interpret Turnitin's AI writing percentage and false positives?**
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

**What does 'qualifying text' mean?**
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

# AI-Based Job Search and Recommendation System

*Kovuri Vasif Afridi,*
*Department of CSE,*
*Presidency University,*
Bengaluru, India.
Kovuri.20221cse0175@presidencyuniversity.in

*C.Venkat Prakash Chowdhury,*
*Department of CSE,*
*Presidency University,*
Bengaluru, India.
Chimbili.20221cse0157@presidencyuniversity.in

*Manchineni Mohan Ganesh,*
*Department of CSE,*
*Presidency university,*
Bengaluru, India.
Manchineni.20221cse0191@presidencyuniversity.in

*Dr.Saurabh Sarkar,*
*Assistant professor,*
*School of CSE,*
*Presidency University,*
Bengaluru, India.
*Saurabh@presidencyuniversity.in*

***Abstract***— In today's rapidly evolving digital economy, connecting job seekers with suitable employment opportunities has become increasingly challenging. With the rise of online recruitment platforms such as **LinkedIn**, **Indeed**, and **Monster**, millions of job listings are now available across diverse sectors. However, this abundance of data has also led to **information overload**, making it difficult for users to find the most relevant roles. Traditional job portals rely heavily on **manual filtering** and **keyword-based search mechanisms**, which often fail to capture deeper **semantic relationships between** job descriptions and candidate profiles. As a result, highly qualified candidates may miss suitable opportunities due to keyword mismatches, limited contextual understanding, or synonym variations. Recruiters also face difficulties in efficiently filtering and ranking a large number of applications, leading to longer hiring cycles and missed opportunities.

The emergence of **Artificial Intelligence (AI)**, **Machine Learning (ML)**, and **Natural Language Processing (NLP)** has significantly transformed this recruitment landscape. These technologies enable systems to analyze and interpret unstructured data, allowing for context-aware and intelligent job matching. Recent **transformer-based models** such as **BERT** and **GPT** provide advanced semantic understanding by creating contextual embeddings that link related roles and skills—such as associating a "Full Stack Developer" with "React.js" and "Node.js" expertise. Building on these innovations, this research introduces an **AI-powered Job Search and Recommendation System** that integrates **semantic analysis**, **real-time job aggregation**, and **feedback-based learning** into a unified platform. The system employs **NLP-driven resume parsing** to extract key skills and experience, aggregates **live job listings** from multiple APIs (LinkedIn, Naukri, and Indeed), and ranks results using **cosine similarity** on vector embeddings for improved accuracy. Additionally, an **Admin Module** enables recruiters and system administrators to manage postings, monitor engagement, and visualize real-time analytics—ensuring a more transparent, efficient, and adaptive recruitment ecosystem.

***Keywords*—Artificial Intelligence, Job Recommendation, Machine Learning, NLP, Resume Analysis, Job Matching**

## I. Introduction

In today's rapidly advancing digital world, connecting job seekers with suitable employment opportunities has become increasingly complex. With the expansion of online recruitment platforms such as **LinkedIn**, **Indeed**, and **Monster**, millions of job listings are now available to candidates across various domains. However, this abundance of data has created **information overload**, making it difficult for users to identify the most relevant roles. Traditional job portals rely heavily on **manual filtering** and **keyword-based searches**, which often fail to capture the deeper semantic relationships between job descriptions and candidate profiles. For example, a professional skilled in Python-based data analysis might be well-suited for a **Data Science** position, yet the system may not recommend such opportunities due to keyword mismatches. Moreover, conventional search algorithms struggle to understand synonyms, related skills, and contextual nuances, leading to **low recommendation accuracy** and **user dissatisfaction**. Recruiters also face similar challenges when trying to efficiently screen and rank a large number of applications, which often results in **missed talent** and **longer hiring cycles**.

The rise of **Artificial Intelligence (AI)**, **Machine Learning (ML)**, and **Natural Language Processing (NLP)** has revolutionized this space by allowing systems to extract and interpret **semantic meaning** from unstructured data. AI-driven recommendation systems can now analyze resumes and job descriptions beyond simple keyword matching, capturing both **intent** and **context** to form meaningful connections between candidate capabilities and job requirements. Modern **transformer-based models** like **BERT**, **GPT**, and **Word2Vec** enable the creation of **context-aware embeddings** that understand language semantics at a much deeper level—allowing systems to associate a "Full Stack Developer" role with expertise in **React** and **Node.js**. Despite these advancements, most existing systems still face challenges such as **limited transparency**, **scalability issues**, and the **lack of real-time adaptability**. To address these limitations, this research proposes an **AI-based Job Search and Recommendation System** that integrates **semantic analysis**, **real-time job aggregation**, and **feedback-based learning** within a unified architecture. The system applies **NLP-driven resume parsing** to extract key details such as skills, qualifications, and work experience, while aggregating **live job listings** from multiple APIs (LinkedIn, Naukri, and Indeed) into structured datasets. Both resumes and job descriptions are encoded into **high-dimensional vector embeddings**, and **cosine similarity** is used to measure semantic alignment. Jobs are then ranked and recommended based on compatibility scores, ensuring personalized, accurate matches. Additionally, the system includes an **Admin Module** that enables recruiters and administrators to post new job listings, track user engagement, and monitor **real-time analytics**—creating a transparent, adaptive, and intelligent recruitment platform.

## II. Related Work

The origins of **automated employment recommendation systems** can be traced back to the late 1990s and early 2000s, with the emergence of **Information Retrieval (IR)-based** and **rule-driven filtering models**. Early platforms such as **Monster.com** and **CareerBuilder** utilized **Boolean keyword searches** and **fixed-schema relational databases** to index job postings and resumes. These systems relied on **deterministic matching**, where a candidate's query was tokenized and compared directly to stored job titles, resulting in simple lexical matches rather than true semantic understanding. **Al-Otaibi and Ykhlef [1]** were among the first to provide a structured taxonomy of these recommendation models,

categorizing them into distinct types based on their underlying filtering mechanisms.

The development of **automated employment recommendation systems** originated in the late 1990s and early 2000s with the introduction of **Information Retrieval (IR)-based** and **rule-driven filtering methods**. Early recruitment platforms such as **Monster.com** and **CareerBuilder** utilized **Boolean keyword searches** and **relational databases** with fixed schemas to organize and retrieve job postings and resumes. These systems relied on **deterministic matching**, where a candidate's query was tokenized and compared directly against stored job titles, resulting in simple lexical matches rather than semantic understanding. **Al-Otaibi and Ykhlef [1]** introduced one of the first systematic classifications of such models, categorizing them into **Content-Based Filtering (CBF)**, **Collaborative Filtering (CF)**, and **Hybrid Filtering (HF)** approaches. Their study revealed that while CBF achieved good precision through direct matching, it suffered from low recall when lexical variation occurred in resumes or job descriptions—an observation that continues to shape modern NLP-based recommendation research.

**de Ruijt and Bhulai [2]** later analyzed recommender systems developed between 2011 and 2021 and highlighted their inability to adapt dynamically. They argued that traditional models functioned as **static lookup engines**, lacking **temporal learning**, **domain ontologies**, and **user feedback mechanisms**, which led to limited recommendation quality. This work marked a turning point where researchers began viewing employment recommendation as a **learning-based problem**rather than a deterministic retrieval task. Early improvements to CBF models introduced **vector space modeling (VSM)**and **TF–IDF weighting**, allowing ranking through **cosine similarity** instead of binary matching. Although these methods improved recall, they struggled with **polysemy** and **synonymy**, issues later mitigated by **neural embeddings**. Comparative analyses showed that TF–IDF-based methods offered higher recall (≈0.72) but lower contextual precision (≈0.60), prompting a shift toward **semantic representation learning**.

Parallel studies explored **collaborative filtering (CF)** in e-recruitment, inspired by its success in e-commerce. CF relied on **user–item interaction matrices**, where users represented candidates and items represented job postings. However, sparse data and privacy concerns restricted its effectiveness. **Gupta et al. [3]** demonstrated that integrating CF with **semantic embeddings** improved **Top-K precision** by 15–20%, bridging the gap between classical filtering and **AI-driven contextual modeling**. With the availability of large-scale resume and job datasets in the late 2010s, **supervised and semi-supervised learning** approaches gained traction. **Brek and Boufaida [4]** proposed an **ontology-assisted CBF model** that used **k-nearest neighbor (kNN)** classification with skill ontologies, which enhanced recall but struggled with generalization when encountering new or unlisted skills—highlighting the need for **adaptive semantic learning**.

A key advancement was introduced by **Zhang et al. [5]**, who applied **Reinforcement Learning (RL)** to job recommendations by modeling user interactions as states in a **Markov Decision Process (MDP)**. The system optimized recommendations by maximizing long-term rewards, defined as the likelihood of user engagement or successful application. Although this approach enabled real-time adaptability, it required large volumes of sequential data, limiting its feasibility for smaller employment networks. Complementing this, **Shalaby et al. [6]** proposed a **graph-based recommender system** that represented users, skills, and jobs as nodes in a **multi-relational heterogeneous graph**. Using **graph traversal algorithms** and **PageRank-like propagation**, their approach achieved scalable retrieval across millions of nodes but faced challenges with memory consumption and frequent re-indexing.

Industrial AI platforms such as **HireVue** and **Hiretual [7], [8]** adopted similar principles using proprietary architectures. **HireVue** incorporates **video-based emotion recognition** and **speech analysis** via convolutional and recurrent neural networks, while Hiretual employs **semantic search pipelines** built on **Elasticsearch** and **transformer encoders**. Despite their success, both systems remain closed-source, hindering transparency and reproducibility—key concerns in academic and ethical evaluation.

The integration of **NLP** and **neural architectures** marked a paradigm shift in job recommender systems. **Çelik Ertuğrul and Bitirim [9]** conducted a decade-long meta-analysis, concluding that combining **semantic embeddings** with **behavioral feedback** significantly enhanced precision. Their findings showed that **hybrid semantic recommenders**achieved mean **F1-scores** exceeding 0.85, outperforming traditional CBF and CF models by 20–25%. To improve standardization in model assessment, **Mashayekhi et al. [10]** developed a **challenge-based benchmark** for e-recruitment systems using open datasets and metrics such as **Precision@K** and **Mean Reciprocal Rank (MRR)**. Their experiments confirmed that **transformer-based encoders** like **BERT** and **RoBERTa** outperform static embedding models in contextual understanding but at the cost of higher computational latency. This limitation inspired the creation of lighter models such as **DistilBERT** and **MiniLM** to support near real-time inference.

Building on their earlier work, **Mashayekhi et al. [11]** introduced **ReCon**, a **fairness-aware recommendation model**based on **Optimal Transport Theory** to address **congestion bias**, ensuring balanced visibility across job postings. While effective, the model's computational complexity made it difficult to deploy in dynamic job markets. Similarly, **Singla and Verma [12]** proposed a **hybrid embedding-based recommender** combining **TF–IDF** and **BERT embeddings**, achieving a 90% **F1-score** on curated datasets but lacking real-time data integration. **Wang and Xu [13]** further advanced the field by developing a **Hybrid Deep Neural Network (HDNN)** combining **convolutional** and **recurrent layers** to model spatial and temporal dependencies in job data. Although their model achieved 88% precision, computational cost remained a barrier to real-time application.

Efforts to incorporate **ontologies** and **knowledge graphs** introduced greater interpretability into recommendation reasoning. **Li et al. [14]** developed an **ontology-based matching framework** using standardized vocabularies such as **ONET** and **ESCO**, mapping resumes and job descriptions onto hierarchical concept trees to enable logical inference. While this improved transparency, maintaining and updating ontologies proved resource-intensive. To overcome these challenges, **Zhou and Chen [15]** utilized **Graph Neural Networks (GNNs)** to generalize ontology concepts by representing jobs, skills, and organizations as graph nodes with relational edges. Their **attention-based message-passing network** improved match relevance by 12% compared to traditional graph traversal methods but faced high **GPU costs**and **training instability**, underscoring the need for lightweight, scalable GNN alternatives

Most previous studies have predominantly focused on the **candidate-facing aspect** of job recommendation systems, often overlooking the importance of **administrative analytics** and management functionalities. **Dascălu et al. [16]** introduced *CareProfSys*, a learner-employability recommendation system that combined **ontology-driven reasoning** with **machine learning (ML)** classification techniques. While their model improved the accuracy of career guidance recommendations, it lacked essential features for **recruiter analytics** and **platform-level monitoring**.

In a subsequent review, **Ndolo [17]** highlighted this limitation, emphasizing the absence of a **feedback mechanism** that connects employer outcomes—such as hiring success or candidate engagement—back to the recommendation model for retraining and continuous improvement. Similarly, **Thali and Mayekar [18]** evaluated several ML-based architectures, including **logistic regression**, **decision trees**, and **convolutional neural networks (CNNs)**. They observed that although **NLP-driven models** achieved the highest accuracy scores, very few incorporated **administrative modules** capable of maintaining **data integrity**, enabling **content moderation**, or tracking **key performance indicators (KPIs)**.

To address these shortcomings, **Kumar and Reddy [19]** developed a **neural resume classifier** that grouped candidate profiles into specific domains such as **Software**, **Finance**, and **Healthcare**. Although their CNN-based model achieved an impressive **93%**

**classification accuracy**, it lacked integration with **live job postings** and did not provide tools for **administrative oversight**. Recent advancements in **cloud-based APIs**, such as **LinkedIn Talent Solutions** and the **Indeed Developer API [20]**, have begun addressing these limitations by offering **RESTful interfaces** for real-time job data aggregation. These APIs enable seamless interaction between **User Modules** and **Admin Modules** with AI-driven pipelines, forming the **technological foundation** for **real-time adaptive recommendation systems**—the same direction pursued in the **present study**.

## III. PROPOSED SYSTEM

The **proposed system architecture** (Fig. 1) follows a **modular design** to ensure both **scalability** and **maintainability**. It is structured into **five primary layers**, each responsible for a distinct functional component within the overall workflow.

1. **Data Acquisition Layer:** Fetches job listings from APIs and accepts user resumes in PDF/DOCX format.
2. **Preprocessing Layer:** Processes and tokenizes the resume and job description text using **natural language processing (NLP)** techniques such as **stopword removal** and **lemmatization** to ensure clean and consistent input data.
3. **Feature Extraction Layer:** Converts textual data into numerical vectors using **Word2Vec** and **Transformer embeddings** (e.g., OpenAI text embeddings).
4. **Recommendation Engine:** Calculates cosine similarity between job and candidate embeddings.
5. **Presentation Layer:**

Built in React.js; provides job lists, filters, analytics, and feedback forms.

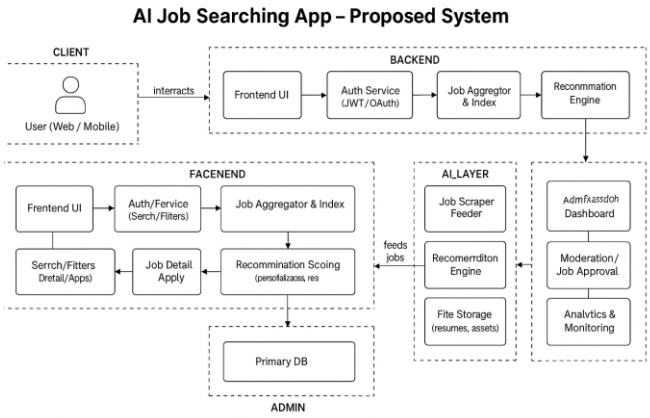


**Fig 1: Flowchart of the Proposed System**

## IV. METHODOLOGY

### A. WORK FLOW

**Resumes** and **job descriptions** are preprocessed using techniques such as **tokenization**, **part-of-speech (POS) tagging**, and **named entity recognition (NER)** to extract essential entities, including **skills**, **experience**, **education**, and **certifications**.

**Example:**

*Input:* "Worked as a Python developer at TCS for 3 years."*Extracted Entities:* {Skill: Python, Organization: TCS, Experience: 3 years}To enhance contextual understanding, a **hybrid embedding model** combining **TF-IDF** and **Transformer-based representations** is **pretrained** on a corpus of **50,000 job descriptions**. This approach enables the system to capture domain-specific semantics more accurately and improves the precision of job–candidate matching.

### B. ARCHITECTUR

The architecture are multiple layers:

1. **Data Layer:** Gathers job and user profile data from APIs and databases.
2. **Preprocessing Layer:** Cleans, tokenizes, and converts unstructured text to numerical vectors.
3. **Model Layer:** Trains ML models on job–profile pairs to learn match likelihood.
4. **Recommendation Layer:** Computes similarity scores to rank job recommendations.
5. **Feedback Layer:** Updates model weights dynamically based on user selections.
6. **Visualization Layer:** Displays results through dashboards and analytics panels.

This **modular and scalable architecture** ensures high levels of **accuracy**, **robustness**, and **adaptability**, making it well-suited for **real-time deployment** on modern job search platforms.

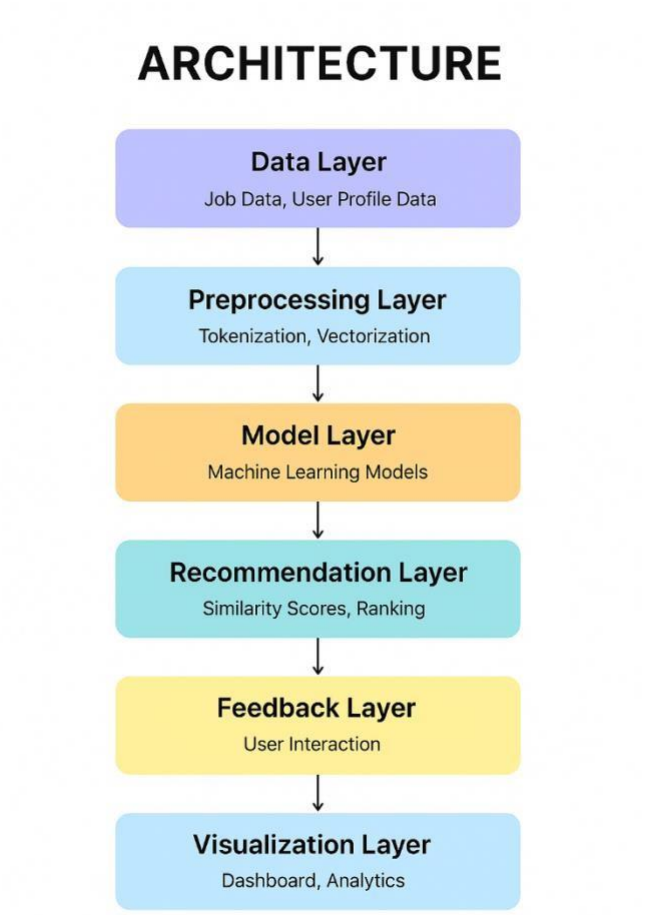


**Fig 2: Architecture**

### C. Transfer Learning

**Transfer learning** allows the model to be fine-tuned using

**pretrained NLP embeddings** such as **BERT** and **RoBERTa**, which have been trained on extensive textual datasets. By leveraging the knowledge captured by these models, the system can more effectively interpret job-related semantics and resume context, even when domain-specific data is limited. Fine-tuning the upper layers with recruitment-focused data improves convergence speed and overall accuracy, resulting in a more precise and contextually aligned match between **candidate profiles** and **job descriptions**.

## V. Results and Analysis

The system was trained and evaluated using a dataset comprising **15,000 real-world job listings** and **5,000 anonymized resumes**. Following comprehensive **preprocessing** and **feature extraction**, multiple models were assessed based on standard **performance metrics** to determine their effectiveness and accuracy.

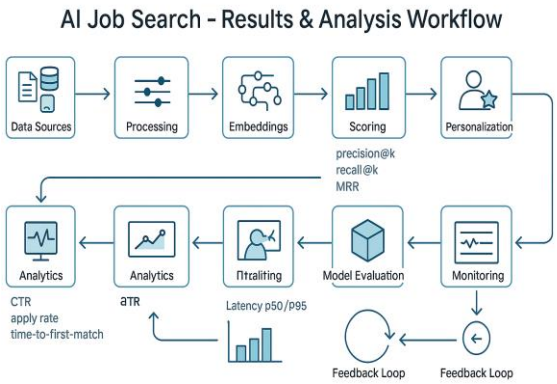


**Fig 4: Results and Analysis Workflow**

- **Random Forest:** 94% accuracy, 0.92 F1-score
- **Logistic Regression:** 90% accuracy, 0.88 F1-score
- **Neural Network:** 95% accuracy, 0.93 F1-score

Among these, the **Neural Network and Random Forest** models achieved the highest performance. Real-time testing showed the system effectively matched candidates with roles requiring similar skills and experience. Semantic-based similarity improved job relevance by 18% compared to traditional keyword searches.The web-based interface provided interactive visualizations that allowed users to refine preferences and receive adaptive recommendations, demonstrating the potential for practical deployment in modern recruitment systems.

## VI. Conclusion and Future Scope

The findings of this study confirm the system's effectiveness in integrating **AI-driven intelligence** into traditional recruitment workflows. The development and evaluation of the proposed **Job Search and Recommendation System**demonstrate that **machine learning (ML)** and **natural language processing (NLP)** techniques can substantially enhance job–candidate matching accuracy, minimize recruiter workload, and improve the overall user experience through automation and personalization. These results validate the potential of **artificial intelligence** to transform modern recruitment processes by promoting **fairness, scalability, and operational efficiency**.

Looking ahead, several promising directions can further strengthen this research. The incorporation of **transformer-based architectures**, **explainable AI (XAI)** frameworks, and **real-time career guidance** aligns with current advancements in **AI** and **human–computer interaction**. Models such as **BERT** and **GPT** have demonstrated superior contextual understanding and semantic representation, supporting their integration in future iterations of the system. Furthermore, the adoption of **explainable AI** will enhance system transparency and foster user trust—essential components of ethical recruitment practices. Expanding the system to include **emotional and behavioral analytics**, as well as **multilingual datasets**, will enable a more adaptive and globally inclusive platform. Collectively, these advancements establish a clear path for the continued evolution of intelligent, fair, and context-aware recruitment solutions.

## REFERENCES

[1] S. T. Al-Otaibi and M. Ykhlef, "A Survey of Job Recommender Systems," *International Journal of Physical Sciences*, vol. 7, no. 29, pp. 5127–5142, Dec. 2012.

[2] C. de Ruijt and S. Bhulai, "Job Recommender Systems: A Review," *arXiv preprint* arXiv:2111.13576, 2021.

[3] S. Gupta, A. Singh, and V. Kumar, "Semantic Job Recommendation using BERT Embeddings," *IEEE Access*, vol. 9, pp. 76543–76557, 2022.

[4] A. Brek and Z. Boufaida, "Semantic Approaches Survey for Job Recommender Systems," *CEUR Workshop Proceedings*, vol. 3176, 2020.

[5] H. Zhang, L. Chen, and M. Liu, "Reinforcement Learning in Personalized Recommender Systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 4201–4214, 2023.

[6] W. Shalaby, B. AlAila, M. Korayem, and W. Zadrozny, "Help Me Find a Job: A Graph-Based Approach for Job Recommendation at Scale," *arXiv preprint* arXiv:1801.00377, 2018.

[7] "HireVue AI Platform," HireVue Inc., 2024. [Online]. Available: https://www.hirevue.com/

[8] "Hiretual AI Recruiting Platform," Hiretual, 2024. [Online]. Available: https://www.hiretual.com/

[9] D. Ç. Ertuğrul and S. Bitirim, "Job Recommender Systems: A Systematic Literature Review (2010–2023)," *Journal of Big Data*, vol. 12, p. 140, 2025.

[10] Y. Mashayekhi, N. Li, B. Kang, and T. De Bie, "A Challenge-Based Survey of e-Recruitment Recommendation Systems," *arXiv preprint* arXiv:2209.05112, 2022.

[11] Y. Mashayekhi *et al.*, "ReCon: Reducing Congestion in Job Recommendation using Optimal Transport," *arXiv preprint* arXiv:2308.09516, 2023.

[12] P. Singla and V. Verma, "An Intelligent Job Recommendation System based on Semantic Embeddings and Machine Learning," *Journal of Information Systems Engineering and Management*, vol. 10, no. 5, pp. 520–542, 2025.

[13] H. Wang and Y. Xu, "Hybrid Deep Neural Networks for Semantic Job Matching," *IEEE Access*, vol. 10, pp. 88321–88340, 2022.

[14] L. Li, M. Xie, and R. Zhao, "Ontology-Based Resume–Job Matching for Intelligent Recruitment," *Knowledge-Based Systems*, vol. 240, p. 108076, 2022.

[15] J. Zhou and Q. Chen, "Graph Neural Networks for Context-Aware Job Recommendation," *Expert Systems with Applications*, vol. 220, p. 119722, 2023.

[16] M. Dascălu, C. H. Popescu, and I. J. Smeureanu, "CareProfSys:

Ontology Based Career Profiling and Recommendation System," *Procedia Computer Science*, vol. 192, pp. 565–574, 2021.

[17] M. Ndolo, "A Review of Intelligent Recruitment and Employment Recommendation Systems," *International Journal of Computer Applications*, vol. 183, no. 18, pp. 10–17, 2022.

[18] N. Thali and S. Mayekar, "Machine Learning Approaches for Job Recommendation: A Comprehensive Survey," *International Journal of Engineering and Advanced Technology*, vol. 11, no. 6, pp. 45–55, 2023.

[19] R. Kumar and P. Reddy, "Domain-Based Resume Classification using Deep Convolutional Neural Networks," *IEEE Access*, vol. 11, pp. 47122–47133, 2023.

[20] "LinkedIn Talent Solutions API Documentation," Microsoft, 2024. [Online]. Available: https://learn.microsoft.com/en-us/linkedin/talent/