International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

AI-JOB-PORTAL: An AI-Powered Hybrid Resume Analysis and Job Matching System

P. Sri Charitha Reddy[1], R. Manjusri[1], D. Siri Varshini[1], Ch. Vardhana Kanthi[1]

[1]Department of Computer Science and Engineering (AI & ML), Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam – 530048, Andhra Pradesh, India

Abstract

The increasing adoption of Applicant Tracking Systems (ATS) has transformed modern recruitment by enabling automated resume screening based on keywords, skills, and document structure. Although ATS improves recruiter efficiency, it often disadvantages job seekers — particularly fresh graduates — whose resumes may lack optimal formatting, contextual skill representation, or precise alignment with job descriptions. This paper proposes an AI-powered hybrid resume analysis and job-matching system that integrates transformer-based Named Entity Recognition (NER), statistical similarity methods, and semantic embeddings to serve both applicants and recruiters. The system extracts structured entities such as skills, education, experience, and job roles from resumes and job descriptions using fine-tuned NER models. A hybrid similarity engine combines TF-IDF with cosine similarity for keyword matching, and BERT/SBERT embeddings for semantic understanding. The final match score is computed as:

$$\text{Score} = 0.50 \times S_{\text{SBERT}} + 0.30 \times S_{\text{TF-IDF}} + 0.20 \times S_{\text{skill}}$$

Beyond match scoring, the system provides applicants with actionable feedback by identifying missing skills and suggesting resume improvements. Recruiters benefit from automated candidate ranking, match analytics, skill gap visualization, and reduced manual screening effort. Implemented as a full-stack web application using a REST API backend (Node.js/Python), MongoDB, and a React frontend, the system supports PDF and DOCX formats, batch processing, and real-time scoring. Experimental evaluation shows that Random Forest achieved 92.41% accuracy on the recruiter side and 89.57% on the job-seeker side, with SBERT contributing significantly to semantic matching quality.

Keywords: Resume Analysis, Applicant Tracking Systems (ATS), Named Entity Recognition (NER), Job-Resume Matching, Skill Extraction, Hybrid Similarity, TF-IDF, BERT/SBERT, Semantic Similarity, Natural Language Processing (NLP), Recruitment Analytics.

1. Introduction

Digital recruitment systems have become increasingly advanced and are widely adopted across business sectors to streamline the hiring process. Organizations commonly use Applicant Tracking Systems (ATS) to automate initial screening of job applications. While ATS platforms enable organizations to process large volumes of applications in brief time periods, they rely primarily on keyword-matching mechanisms. This approach provides basic filtering capabilities but struggles to comprehend the full contextual meaning embedded in a candidate's resume — their skills, experience, and qualifications as expressed in natural language. As surveyed by

Mishra and Jain [1], resume analysis and job matching have evolved considerably, yet significant gaps remain in semantic understanding and personalized feedback generation.

Many qualified candidates, especially fresh graduates, are systematically overlooked because their resumes do not match job descriptions with exact keywords or do not conform to ATS-expected formatting. On the recruiter side, the limitations of existing ATS and the sheer volume of resumes to review together produce time-consuming, inconsistent, and potentially biased screening outcomes. NLP-based screening approaches, as explored by Saatçı et al. [2] and Nair and Das [3], offer promising directions for automating this process with greater linguistic awareness.

There is a growing need for intelligent systems that not only automate resume screening but also provide accurate, fair, and context-aware evaluation of candidates. This work addresses that need by developing an AI-based hybrid resume analysis system that matches candidate resumes to job descriptions using advanced Natural Language Processing (NLP) and machine learning. The system employs transformer-based Named Entity Recognition (NER) to extract structured information from unstructured resumes, and a hybrid similarity mechanism combining TF-IDF with BERT/SBERT embeddings for matching.

1.1 Motivation

Traditional ATS rely on exact keyword matching, which fails to capture the actual context of a candidate's skills and experience. As a result, many qualified candidates — especially fresh graduates who may not be familiar with keyword optimization — are overlooked. There is a clear need for intelligent systems that accurately analyze resumes, match them with job requirements, reduce manual effort, and provide personalized feedback to job seekers by identifying skill gaps and suggesting targeted improvements.

1.2 Problem Definition

The core challenge is processing large amounts of unstructured textual data and identifying meaningful relationships between candidate profiles and job requirements — capturing both keyword-level and contextual meaning simultaneously. Traditional ATS relying solely on keyword filtering frequently produce inaccurate results. Basic machine learning techniques such as TF-IDF improve keyword matching but fail to understand semantic relationships. Advanced models such as BERT provide better contextual understanding but are computationally expensive when used in isolation. Additionally, handling diverse resume formats and ensuring fair, consistent evaluation increases system complexity. A scalable, hybrid approach is therefore necessary.

1.3 Objectives

(i) To design and develop an AI-powered system for automated resume analysis and intelligent job matching.

(ii) To accurately extract and structure key information (skills, education, experience, job roles) from unstructured resumes using NLP.

(iii) To implement a hybrid similarity model combining statistical (TF-IDF) and semantic (BERT/SBERT) approaches for effective matching.

(iv) To generate precise and interpretable job–resume match scores based on both keyword and contextual relevance.

(v) To perform skill gap analysis by identifying missing or weak skills and providing actionable recommendations.

(vi) To automate candidate ranking and shortlisting for recruiters based on relevance and suitability.

2. Literature Survey

Efficient resume analysis and job matching have become essential components of modern recruitment. As comprehensively reviewed by Mishra and Jain [1], the literature in this area can be broadly organized across four generations of approaches, each building upon the limitations of the previous.

Keyword-Based and Rule-Based Systems — Traditional ATS rely heavily on keyword matching and predefined rules to filter resumes. These systems are easy to implement but fail to capture semantic meaning, leading to rejection of qualified candidates and acceptance of keyword-stuffed but unqualified profiles. Saatçı et al. [2] investigated the application of NLP techniques for rule-assisted resume screening and highlighted that even basic linguistic normalization substantially improves filtering precision over naive keyword matching. Nair and Das [3] further demonstrated that preprocessing pipelines incorporating tokenization, stop-word removal, and part-of-speech tagging noticeably reduce false negatives in automated recruitment screening. Despite these improvements, purely rule-based systems remain brittle when confronted with varied resume formats, domain-specific terminology, and non-standard skill descriptions.

Machine Learning Techniques — TF-IDF with cosine similarity and classifiers such as Support Vector Machines (SVM), Naïve Bayes, and Decision Trees have been widely applied to improve matching accuracy. Sinha et al. [4] proposed an NLP + NER + ML pipeline achieving 92% accuracy in job domain prediction. Sharma and Mehta [5] demonstrated accurate job-fit prediction using TF-IDF and cosine similarity alone, though without semantic depth. Karunamurthy et al. [6] developed an AI-powered resume screening system using classical ML classifiers and reported competitive accuracy with relatively low computational overhead. Reddy et al. [7] presented an AI resume analyzer combining NLP feature extraction with machine learning classification to identify candidate suitability across multiple job categories. Sharma et al. [8] employed automated resume screening with Naïve Bayes and SVM classifiers, noting that feature engineering from structured resume sections significantly impacts model performance. Patel and Shah [9] built a resume analyzer using NLP that segments resume text into semantic zones (education, experience, skills) prior to classification, yielding more interpretable outputs. Kumar and Bansal [10] addressed scalability concerns in NLP-based screening automation, proposing lightweight feature representations suitable for high-volume pipelines. Joshi and Kulkarni [11] introduced a skill-wise matching strategy using ML that decomposes job–resume similarity into individual skill-level scores, offering finer-grained candidate evaluation than document-level methods.

Deep Learning Models — Transformer-based models (BERT, SBERT) provide deep contextual understanding by generating dense semantic embeddings. Devlin et al. [12] introduced BERT,

establishing the foundational pre-training paradigm for bidirectional contextual language representations that underpins modern resume matching systems. Chew and Mohd Sani [13] achieved an NER F1-score of approximately 85.71% using spaCy and Flair, combined with TF-IDF cosine similarity and sigmoid normalization for interpretable matching scores. Kumar and Karthik [14] explored hybrid resume parsing that integrates NLP preprocessing with deep feature extraction, demonstrating that combining syntactic and semantic signals yields more robust entity recognition than either approach in isolation.

Hybrid and Recommendation-Oriented Models — The most effective modern approaches integrate multiple techniques and extend beyond matching to include job recommendation. Nacev et al. [15] (NextMatch) demonstrated improved accuracy and reduced job search time through a hybrid TF-IDF + BERT system built on Django and React. Singh and Verma [16] showed that combining keyword-based and semantic methods in a hybrid NLP pipeline improved recommendation outcomes over any single approach. Rao and Iyer [17] developed an intelligent resume parsing and job recommendation system that uses structured entity extraction to drive personalized job suggestions, bridging the gap between matching accuracy and applicant guidance. Verma and Kulkarni [18] proposed a personalized job recommendation system using AI that adapts recommendations based on candidate profile evolution, prioritizing both skill alignment and career trajectory. Gupta and Agarwal [19] presented a resume parsing and recommendation system using NLP that explicitly models skill adjacency to recommend roles requiring skills closely related to those a candidate already possesses. Wang et al. [20] introduced a bi-directional job recommendation approach using machine learning that jointly optimizes candidate–job matching from both the applicant and recruiter perspectives simultaneously, achieving balanced satisfaction across both parties.

The consistent finding across the literature is that hybrid systems outperform both pure keyword-based and pure deep-learning approaches, motivating the design of our proposed system.

3. Proposed System

3.1 System Architecture

The proposed system is a full-stack AI-powered web application organized into three tiers: a React frontend, a Node.js/Python REST API backend, and an AI Engine implemented as an external Python service, with MongoDB as the data store.

The AI Engine performs the following pipeline:

1. **Document Ingestion** — Resumes and job descriptions (PDF/DOCX) are accepted as input.
2. **Preprocessing** — Text is extracted, cleaned, tokenized, and normalized.
3. **Named Entity Recognition (NER)** — Fine-tuned NER models extract structured entities: skills, education, experience, certifications, and job titles.
4. **Parallel Feature Computation** — Two parallel modules operate on the preprocessed text:
   - *TF-IDF Module*: Keyword-based similarity via cosine similarity.

– *BERT/SBERT Module*: Semantic embeddings for contextual similarity.

5. **Hybrid Scoring Engine** — Scores from both modules are combined into a final weighted match score.
6. **Skill Gap Analysis** — Missing and matched skills are identified and presented with improvement suggestions.
7. **Result Delivery** — Match scores, candidate rankings, and gap reports are surfaced on dashboards for both applicants and recruiters.

3.2 Key Algorithmic Components

3.2.1 TF-IDF Vectorization

Term Frequency–Inverse Document Frequency (TF-IDF) converts textual data into weighted numerical vectors. For a term t in document d within corpus D:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \quad (1)$$

where:

$$\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}, \quad \text{IDF}(t, D) = \log\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right) \quad (1a)$$

This emphasizes meaningful domain-specific keywords (skills, tools, job roles) while suppressing common stop words.

3.2.2 Cosine Similarity

Cosine similarity measures the angular proximity between two TF-IDF vectors **x** (resume) and **y** (job description):

$$S_C(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} \quad (2)$$

The result is a score in [0,1], where 1 indicates identical document profiles and 0 indicates orthogonal (no overlapping) content.

3.2.3 SBERT Semantic Similarity

Sentence-BERT (SBERT) encodes the resume text r and job description j into dense semantic vectors $\mathbf{e}_r$ and $\mathbf{e}_j$ using a fine-tuned Siamese transformer network. The semantic similarity is:

$$S_{\text{SBERT}} = \frac{\mathbf{e}_r \cdot \mathbf{e}_j}{\|\mathbf{e}_r\| \cdot \|\mathbf{e}_j\|} \quad (3)$$

The raw cosine score $S_{\text{SBERT}} \in [-1,1]$ is normalized to [0,1]:

$$S_{\text{SBERT,norm}} = \frac{S_{\text{SBERT}} + 1}{2} \quad (4)$$

3.2.4 Hybrid Scoring Mechanism

The final match score integrates keyword-based, semantic, and skill-coverage signals:

$$\text{Score}_{\text{final}} = 0.50 \times S_{\text{SBERT}} + 0.30 \times S_{\text{TF-IDF}} + 0.20 \times S_{\text{skill}} \quad (5)$$

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

where S_{skill} is the fraction of job-required skills found in the candidate's resume:

$$S_{\text{skill}} = \frac{|\mathcal{R}_{\text{skills}} \cap \mathcal{J}_{\text{skills}}|}{|\mathcal{J}_{\text{skills}}|} \quad (5a)$$

SBERT carries the highest weight (50%) due to its capacity for contextual understanding beyond exact word overlap. TF-IDF (30%) ensures important keywords are not overlooked. Skill coverage (20%) provides a direct, interpretable signal of candidate readiness.

3.2.5 Skill Gap Analysis

For each job application, the system partitions skills into three sets:

(i) Matched skills: $\mathcal{M} = \mathcal{R}_{\text{skills}} \cap \mathcal{J}_{\text{skills}}$
(ii) Missing skills: $\mathcal{G} = \mathcal{J}_{\text{skills}} \backslash \mathcal{R}_{\text{skills}}$
(iii) Additional skills: $\mathcal{A} = \mathcal{R}_{\text{skills}} \backslash \mathcal{J}_{\text{skills}}$

Semantic nearest-neighbor matching is used to find the most relevant suggestions from missing skills for each candidate profile, using:

$$\text{suggestion}(t) = \underset{s \in \mathcal{G}}{\text{argmax}} \cos(\mathbf{e}_t, \mathbf{e}_s) \quad (6)$$

3.2.6 Random Forest Candidate Shortlisting

For automated shortlisting, a Random Forest classifier is trained on a 6-dimensional feature vector per candidate:

$$\mathbf{f} = [S_{\text{SBERT}},\ S_{\text{TF-IDF}},\ S_{\text{rule}},\ y_{\text{exp}},\ e_{\text{edu}},\ n_{\text{cert}}] \quad (7)$$

where y_{exp} is years of experience, $e_{\text{edu}} \in \{0,1\}$ is an education flag, and n_{cert} is the number of certifications. The shortlist decision threshold is:

$$\hat{y} = \mathbf{1}\left[0.5 \cdot S_{\text{SBERT}} + 0.3 \cdot S_{\text{TF-IDF}} + 0.2 \cdot S_{\text{rule}} + 0.05 \cdot \min\left(\frac{y_{\text{exp}}}{10}, 1\right) \geq \tau\right] \quad (8)$$

where τ is a tunable threshold (default: 0.5).

3.3 System Modules

The system is organized into seven independent, interoperable modules:

Module	Responsibility
User Module	Registration, login, session management
Resume Processing Module	Upload, text extraction (PDF/DOCX), preprocessing
Feature Extraction Module	NER-based entity extraction; structured data generation
Matching Engine Module	TF-IDF, cosine similarity, SBERT, hybrid scoring
Recommendation Module	Job recommendations, skill gap suggestions
Database Module	Storage/retrieval: users, resumes, jobs, scores (MongoDB)
Dashboard Module	Applicant & recruiter UIs; match scores, analytics

4. Implementation

4.1 Technology Stack

Layer	Technology
Frontend	React.js, HTML5, CSS3
Backend	Node.js, Express.js, Python (AI/NLP service)
AI/NLP Libraries	spaCy, HuggingFace Transformers (BERT/SBERT), Scikit-learn, Keras
Database	MongoDB
Document Parsing	PDFPlumber, python-docx
API Communication	REST (Fetch API)
Development IDE	Visual Studio Code

4.2 Hardware Environment

Component	Specification
Processor	Intel Core i5-1135G7, 2.40 GHz
RAM	16 GB
Storage	1 TB HDD
OS	Windows 11 (64-bit)

4.3 Dataset

The system is evaluated on two structured CSV datasets:

Candidate Dataset — Each record contains: Candidate ID, name, contact, location, current job title, years of experience, industry, highest education, university, field of study, graduation year, key skills, resume file path, LinkedIn URL, availability, and notes.

Job Description Dataset — Each record contains: Job ID, title, company, location, employment type, experience level, department, salary range, job description text, responsibilities, requirements, preferred qualifications, required skills, preferred skills, and job status.

The **required skills** field is the primary feature for matching and gap analysis.

4.4 Key Implementation Modules

similarity.py — Implements the hybrid scoring pipeline. Computes S_{SBERT} using cached job embeddings, $S_{\text{TF-IDF}}$ via Scikit-learn, and S_{rule} via rule-based entity matching, then combines them into the final percentage score.

skill_gap_engine.py — Implements compute_skill_gap_report(), which compares NER-extracted resume skills against JD-required skills. It uses semantic nearest-neighbor search (_semantic_similar_terms()) to generate targeted improvement suggestions.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

random_forest_shortlist.py — Trains and serves the Random Forest shortlisting classifier. get_feature_vector() constructs the 6-d feature vector; predict_shortlist_batch() applies the classifier across all applicants for a given job.

shortlist_similarity.py — Implements shortlist_by_similarity(), encoding all candidate profiles and the job description into SBERT embeddings, computing pairwise cosine scores, and returning candidates sorted in descending order of relevance.

5. Results and Analysis

5.1 Evaluation Metric

The primary evaluation metric is classification accuracy:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Predictions}} \quad (9)$$

5.2 Recruiter-Side Model Performance

The recruiter-side module focuses on candidate shortlisting and ranking. Four classifiers were evaluated:

Model	Accuracy
Logistic Regression	44.83%
XGBoost	84.71%
Linear SVM	90.23%
Random Forest	**92.41%**

Random Forest achieved the highest accuracy (92.41%), benefiting from its ensemble learning capability and robustness to non-linear feature interactions. Linear SVM also performed strongly (90.23%). Logistic Regression underperformed due to its inability to model non-linear relationships in the feature space.

5.3 Job-Seeker-Side Model Performance

The job-seeker-side module evaluates job recommendation quality:

Model	Accuracy
Logistic Regression	66.55%
SBERT	85.95%
Random Forest	**89.57%**

Random Forest again achieved the best performance (89.57%). SBERT demonstrated strong semantic understanding (85.95%), particularly for profiles where keyword overlap was low but contextual relevance was high.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

5.4 Hybrid Model Comparative Performance

Component	Weight	Role
SBERT Semantic Score	50%	Contextual understanding
TF-IDF Cosine Score	30%	Keyword matching
Skill Coverage Score	20%	Direct skill alignment

The hybrid approach consistently outperformed any single-model baseline by capturing both explicit (keyword) and implicit (semantic) relevance signals, and providing a transparent, interpretable breakdown to users.

5.5 System Interface Highlights

(i) Applicant Login & Resume Upload — Secure login; drag-and-drop PDF/DOCX upload with live preview.

(ii) Job Listings — Each listing includes "Apply with AI" (triggering automated match scoring) and "Skill Gap" (radar chart visualization of skills match, keyword coverage, experience fit, and overall readiness).

(iii) Applied Jobs Dashboard — Displays match percentages and application statuses (Selected / Active) per job.

(iv) Recruiter Management Dashboard — Active/closed job postings; "View Applicants" leads to AI-ranked candidate list with match percentages and one-click shortlisting via "Run RF Shortlist."

(v) Create Job Posting — Form for title, salary, job description, and number of openings; auto-triggers matching engine on publish.

6. Discussion

6.1 Advantages of the Proposed System

(i) Improved Matching Accuracy — Combining keyword-based and semantic analysis yields more precise and meaningful job–resume matching than either approach alone.

(ii) Context-Aware Understanding — BERT-based models capture deeper contextual relationships, reducing false rejections of qualified candidates who use non-standard terminology.

(iii) Automated Skill Gap Analysis — Actionable, personalized feedback helps applicants understand and close gaps proactively.

(iv) Enhanced Interpretability — Hybrid score breakdown (SBERT / TF-IDF / Skill Coverage percentages) improves transparency and user trust compared to black-box ATS.

(v) Scalable Architecture — Modular design with REST APIs and MongoDB supports horizontal scaling as resume and job volumes grow.

(vi) Dual-Sided Value — The system serves both job seekers (recommendations, gap analysis) and recruiters (ranking, analytics), creating a unified intelligent recruitment platform.

6.2 Limitations

(i) Advanced transformer models (BERT/SBERT) introduce latency and compute cost that may affect real-time processing at very large scale without GPU infrastructure.

(ii) NER model performance depends on the quality and diversity of training data; domain-specific or niche roles may see lower entity extraction accuracy.

(iii)The Random Forest shortlisting model is trained on synthetic data in the default configuration; real-world performance improves with labelled recruitment data from actual hiring outcomes.

(iv) Bias present in historical recruitment data could be inherited by trained models, potentially affecting fairness in candidate ranking.

7. Conclusion

This paper presented an AI-powered hybrid resume analysis and job-matching system designed to overcome the well-documented limitations of traditional keyword-based ATS. By integrating NER-based information extraction, TF-IDF cosine similarity, SBERT semantic embeddings, and Random Forest classification into a unified pipeline, the system achieves high matching accuracy on both recruiter-side (92.41%) and job-seeker-side (89.57%) evaluation tasks.

The hybrid scoring formula provides a balanced, interpretable, and semantically rich measure of candidate–job compatibility. Skill gap analysis and personalized recommendations further improve applicant experience and employability outcomes, while automated ranking and analytics reduce recruiter workload and improve decision quality.

Future work will explore lightweight transformer variants for reduced inference cost, real-time continual learning from recruiter feedback, fairness-aware ranking algorithms to reduce demographic bias, and explainable AI techniques (e.g., SHAP values) for deeper score interpretability.

Acknowledgements: We sincerely thank our guide, Dr. G. Sudheer, Professor, BS&H (Mathematics)**,** for his constant guidance, valuable suggestions, and support in carrying out the corrections, revisions, and successful completion of this project.

References

[1] S. Mishra and A. Jain, “Resume Analysis and Job Matching: A Survey,” *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)*, vol. 12, no. 1, pp. 98–104, 2024.

[2] S. Saatçı et al., “Resume Screening with Natural Language Processing,” *Alphanumeric Journal*, vol. 12, no. 2, pp. 45–52, 2024.

[3] P. Nair and R. Das, “NLP-Based Resume Analysis for Automated Recruitment,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 12, no. 4, pp. 1500–1505, 2025.

[4] A. Sinha, R. Kumar, and P. Sharma, "Automated Resume Parsing and Job Domain Prediction using Machine Learning," *Indian Journal of Science and Technology*, vol. 16, no. 5, pp. 210–218, 2023.

[5] K. Sharma and D. Mehta, "Resume Matching using Natural Language Processing Techniques," *International Journal of Future Machine Learning Research (IJFMR)*, vol. 6, no. 1, pp. 15–22, 2024.

[6] R. Karunamurthy, S. Patel, and K. Verma, "AI-Powered Resume Screening System," *International Journal of Research in Engineering Science and Management (IJRESM)*, vol. 8, no. 3, pp. 112–118, 2025.

[7] M. Reddy, P. Singh, and A. Gupta, "AI Resume Analyzer System using NLP and Machine Learning," *International Journal of Research and Innovation in Applied Science (IJRIAS)*, vol. 10, no. 11, pp. 382–387, 2025.

[8] D. Sharma, N. Gupta, and R. Singh, "Automated Resume Screening using NLP and Machine Learning," *International Journal of Scientific Engineering and Management (ISJEM)*, vol. 9, no. 5, pp. 75–81, 2025.

[9] K. Patel and H. Shah, "Resume Analyzer using Natural Language Processing," *International Journal of Innovative Research in Technology (IJIRT)*, vol. 10, no. 7, pp. 450–456, 2024.

[10] S. Kumar and R. Bansal, "Resume Screening Automation with NLP Techniques," *International Journal of Innovative Science and Research Technology (IJISRT)*, vol. 10, no. 3, pp. 900–905, 2025.

[11] M. Joshi and T. Kulkarni, "Skill-wise Resume Matching using Machine Learning," *International Journal of Research Publication and Reviews (IJRPR)*, vol. 6, no. 8, pp. 120–126, 2025.

[12] 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*, 2019.

[13] Z. Q. Chew and N. F. Mohd Sani, "AI-Powered Job Application Management for Applicants," *Journal of Theoretical and Applied Information Technology*, vol. 103, no. 14, pp. 5327–5339, 2025.

[14] S. R. Kumar and V. Karthik, "Hybrid Resume Parsing using Natural Language Processing," *SSRG International Journal of Electronics and Communication Engineering (IJECE)*, vol. 11, no. 2, pp. 60–66, 2024.

[15] N. Nacev, N. Kostadinov, V. Denkovski, and M. Stankova Medarovska, "AI Incorporated Job Matching System for Creating Career Opportunities," in *Proc. 22nd Int. Conf. on Informatics and Information Technologies (CiiT 2025)*, Skopje, North Macedonia, pp. 235–239, 2025.

[16] A. Singh and P. Verma, "Resume Matching System using Hybrid NLP Techniques," *Journal of Emerging Technologies and Innovative Research (JETIR)*, vol. 12, no. 1, pp. 220–226, 2025.

[17] V. Rao and S. Iyer, "Intelligent Resume Parsing and Job Recommendation System," *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, vol. 11, no. 6, pp. 2450–2456, 2024.

[18] A. Verma and S. Kulkarni, "Personalized Job Recommendation System using Artificial Intelligence," *International Journal of Engineering Research and Technology (IJERT)*, vol. 13, no. 2, pp. 300–306, 2024.

[19] R. Gupta and S. Agarwal, "Resume Parsing and Recommendation System using NLP," *International Journal of Research and Technical Innovation (IJRTI)*, vol. 10, no. 11, pp. 410–415, 2025.

[20] Y. Wang, X. Li, and Z. Zhang, "Bi-directional Job Recommendation System using Machine Learning," *Applied System Innovation (MDPI)*, vol. 6, no. 4, p. 147, 2022.