iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.78423

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

JobFit-AI - AI Powered Smart Resume & Job Match Analyzer

Vasu[1], Sunidhi[2], Sudhanshu[3], Prince Gautam[4], Mrs. Ritu Singh[5]
[1, 2, 3, 4]*Department of Computer Science and Engineering (IoT), Meerut Institute of Engineering & Technology, Meerut, Uttar Pradesh, India*
[5]*Assistant Professor, Department of Computer Science and Engineering (CSE-IoT) Meerut Institute of Engineering & Technology, Meerut, Uttar Pradesh, India*

Abstract: ***The recruitment process today relies heavily on Applicant Tracking Systems (ATS), which often act as strict gatekeepers. Studies suggest that nearly 75% of qualified resumes are rejected by these systems, often due to simple formatting errors or missing keywords rather than a lack of skills. To address this issue, this paper proposes "JobFit-AI" a web application developed to help candidates navigate automated screening. The system is built using the MERN stack (MongoDB, Express.js, React, Node.js). Unlike traditional parsers that rely on basic keyword counting or Cosine Similarity, our approach integrates the Google Gemini API. This allows for semantic analysis, enabling the system to understand the actual context of a resume PDF when compared to a job description. Leveraging Large Language Models (LLMs), the application generates a "Fit Score," identifies missing skills, and offers specific suggestions for improvement. Testing indicates that this LLMbased method provides significantly higher accuracy than standard string-matching techniques, as it utilizes advanced prompting to interpret a candidate's true potential..***

I. INTRODUCTION

Recruitment methods have evolved significantly due to advancements in technology. Currently, large organizations receive thousands of applications for a single vacancy. To manage this high volume, companies rely on Applicant Tracking Systems (ATS) to automate the initial screening process. Although these systems improve efficiency for recruiters, they often act as a significant barrier for job seekers, particularly for students and fresh graduates who may not be familiar with how these algorithms work.

II. LITERATURE REVIEW

The evolution of resume analysis has shifted significantly from manual verification to automated software solutions. A review of existing platforms and academic research reveals that most current systems fall into two primary categories: commercial "General Scoring" tools or academic "Keyword Matching Systems."

A. Analysis of Existing Approaches

1) General Scoring Platforms: Commercial tools, such as those analyzed by Gupta et al. [1], primarily focus on the aesthetic and structural aspects of a resume. They check for consistent formatting, active verb usage, and section completeness. However, these tools provide a static score based on fixed rules (e.g., "Resume must be one page"). They fail to determine if a candidate is suitable for a specific role, such as a "Senior Backend Developer." A resume might achieve a high score for formatting yet still be rejected for lacking specific technical competencies.

2) Keyword-Based Matching Systems: Research by S. Sharma and V.P. Singh [2] utilized TF-IDF (Term Frequency-Inverse Document Frequency) to count keyword occurrences. While useful for filtering, this method is rigid. For instance, if a student lists "MERN stack" but the job description explicitly demands "MongoDB, Express, React, Node" as separate terms, the system may fail to match them. Furthermore, these systems are designed for recruiters and offer no constructive feedback to the applicant.

3) Ontology-Based Semantic Parsers: K. Smith et al. [3] proposed using ontologies to map words to concepts (e.g., recognizing "Coder" as "Developer"). While this approach improves accuracy, it is computationally expensive and limited by the dictionary's size. Crucially, it cannot advise users on how to rewrite their content to better align with a job profile.

©IJRASET: All Rights are Reserved

4) Machine Learning Classification: Systems proposed by M. Kumaran and A. Safvan [4] use NLP and algorithms like Naïve Bayes to learn from historical hiring data. The major drawback here is the "Cold Start" problem; the model requires thousands of labeled resumes to function. It also operates as a "Black Box," rejecting candidates without explanation. JobFit-AI avoids this by using a pre-trained Large Language Model (Gemini), eliminating the need for a massive initial dataset.
5) Latent Semantic Analysis (LSA): S. G. Rao et al. [5] attempted to use Singular Value Decomposition (SVD) to identify context beyond keywords. However, LSA ignores word order, meaning it cannot distinguish between "I managed the team" and "The team managed me." JobFit-AI leverages Gemini's natural language understanding to correctly interpret sentence structure and leadership roles.
6) OCR and Rule-Based Mining: K. Bhalla and A. J. Singh [6] focused on scanning imagebased resumes. This approach relies on OCR and fixed layout rules, making it fragile. If a user employs a creative design or non-standard headings (e.g., "Academic History" instead of "Education"), the extraction often fails. JobFit-AI analyzes the content itself rather than relying on header placement, ensuring compatibility with diverse formats.

B. The Research Gap

A significant gap exists in current tools: the lack of Target-Specific Tailoring. Most existing solutions operate on a "One-Size-Fits-All" model, suggesting a single "perfect" resume. In reality, a resume must be customized for every application. Current tools identify missing keywords (e.g., "Missing: Docker") but fail to assist the user in generating content that bridges this gap.

C. Uniqueness of JobFit-AI

JobFit-AI addresses these limitations by focusing on Target-Specific Optimization. Unlike generic models [1], our system analyzes the specific relationship between the Resume and the Job Description.

1) Contextual Logic: By integrating the Gemini API, the system understands context. For example, if a job description prioritizes "System Design," the system recognizes that the Projects section holds more weight than Extra-curricular activities.
2) Generative Feedback: Unlike standard scoring systems [2], JobFit-AI provides actionable advice. It acts as a personal career coach, suggesting specific sentence rewrites to align the candidate's actual experience with the job requirements.

III. METHODOLOGY / PROPOSED SYSTEM

The development of JobFit-AI moves beyond traditional keyword counting by embedding Generative AI within a standard MERN stack architecture. This approach allows for a semantic understanding of candidate profiles relative to job requirements.

A. Three-Tier Architecture

To ensure modularity and scalability the system is implemented using a classic three-tier architecture:

1) Presentation Layer: The user interface is constructed using React.js and styled with Tailwind CSS. This layer handles client-side operations, allowing users to upload resume PDFs and input job descriptions. React Hooks are utilized to manage state transitions, ensuring real-time updates for scorecards and improvement lists.
2) Application Layer: The core logic resides in a Node.js environment using the Express.js framework. This layer functions as the intermediary, managing API endpoints, parsing incoming PDF files to extract raw text and orchestrating communication with external AI services.
3) Data Layer: MongoDB serves as the non-relational database for the application. It is responsible for persisting user credentials and maintaining a historical log of past analyses, enabling users to track the progression of their resume quality over time.

B. AI Integration and Prompt Engineering

The central innovation of this project is the integration of the **Google Gemini API**. Unlike legacy NLP models that require extensive dataset training, this system leverages advanced Prompt Engineering. We construct a dynamic prompt by concatenating the extracted resume text with the specific job description.

To ensure the output is computationally usable, we enforce strict formatting constraints on the Large Language Model (LLM). The prompt explicitly instructs the model to: *"Act as a Technical Recruiter... Output a JSON object containing a 'match_score', 'missing_keywords', and 'improvement_tips'."* This constraint forces the LLM to structure its semantic understanding into a machine-readable JSON format, rather than returning unstructured conversational text.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

C. Algorithmic Workflow

The data flow within the system follows a linear execution path:

1) Input Processing: The workflow initiates when the user submits a Resume (PDF) and a Job Description (String).
2) Text Extraction & Cleaning: The backend parses the PDF binary to extract raw text, applying a cleaning function to remove formatting artifacts or non-printable characters.
3) Prompt Synthesis: The cleaned text is merged with the job description into the predefined prompt structure.
4) API Interaction: This synthesized prompt is transmitted to the Gemini API.
5) Response Handling: The system receives the raw response, parses the JSON object, and commits the results to the database before rendering the final analysis to the user.

IV. IMPLEMENTATION

The JobFit-AI prototype was deployed as a fully functional web application leveraging the MERN stack for robust data handling and the Gemini API for semantic processing.

1) Operational Workflow The user interaction begins at the client interface, developed in React.js, where the candidate uploads their resume (PDF) alongside the target job description. The Node.js and Express.js backend intercepts this request, performing immediate text extraction and sanitization to remove non-alphanumeric artifacts. This structured data is then securely transmitted to the Gemini API.
2) Output and Persistence The system returns a comprehensive analysis, including a JobFit Score (quantified as a percentage) and a categorized list of missing technical skills. To ensure data persistence, all analysis results are stored in MongoDB. This allows the dashboard to retrieve and display historical data, enabling users to track their improvements over multiple iterations.
3) Performance Metrics Initial testing of the deployed prototype indicates high efficiency. The average latency for a complete cycle—from file upload to the rendering of results—is recorded at approximately 2–3 seconds varying slightly based on the length of the PDF. Furthermore, semantic accuracy tests where the AI's output was manually verified against human recruiter judgments, showed a correlation rate of roughly **90%** significantly outperforming traditional keyword-matching algorithms.

.

V. RESULTS AND DISCUSSION

To validate the efficacy of the proposed system, we conducted a two-phase evaluation focusing on both algorithmic accuracy and user satisfaction.

A. Quantitative Evaluation

The system was tested using a diverse dataset comprising 100 unique resumes and 50 distinct job descriptions across various technical domains (e.g Web Development, Data Science, DevOps).

The primary metric for success was the relevance of the "JobFit Score." In manual verification tests, where human recruiters cross-checked the AI's output, JobFit-AI demonstrated an average accuracy rate of 90%. This indicates that the Gemini API successfully interpreted context (e.g recognizing "MERN" as a skill set matching "Node.js" requirements) where traditional keyword counters often failed.

B. User Feedback and Comparative Analysis

A pilot study was conducted with 12 active users to gather qualitative feedback. Participants reported that the specific "improvement tips" significantly altered their revision strategy, allowing them to align their profiles more closely with target roles.

When compared to established market tools like *Resume Worded*, a clear distinction emerged. Existing platforms typically analyze a resume in isolation, providing a static score based on general formatting rules. In contrast, JobFit-AI performs a dynamic comparison, factoring in the specific requirements of the provided Job Description. This allows for a deeper more relevant analysis that static tools cannot replicate.

VI. CONCLUSION AND FUTURE SCOPE

The development of JobFit-AI demonstrates the practical utility of integrating Generative AI with standard full-stack web architectures. By moving beyond the limitations of static keyword matching the system effectively addresses the "black box" problem of modern hiring. The successful implementation of the MERN stack alongside the Gemini API proves that semantic analysis can provide candidates with actionable insights, helping them identify and bridge the gap between their current skills and industry requirements.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

A. Future Scope

While the current prototype functions effectively as a student-centric tool, several enhancements are planned for future iterations:

1) Integration with Professional Networks: We aim to connect the system with external APIs (such as LinkedIn) to provide real-time job recommendations based on the user's improving "Fit Score."

2) Generative Rewriting Module: Currently the system suggests improvements. The next phase will involve an active "Auto-Correct" feature where the AI directly rewrites specific bullet points to match the target tone and keywords.

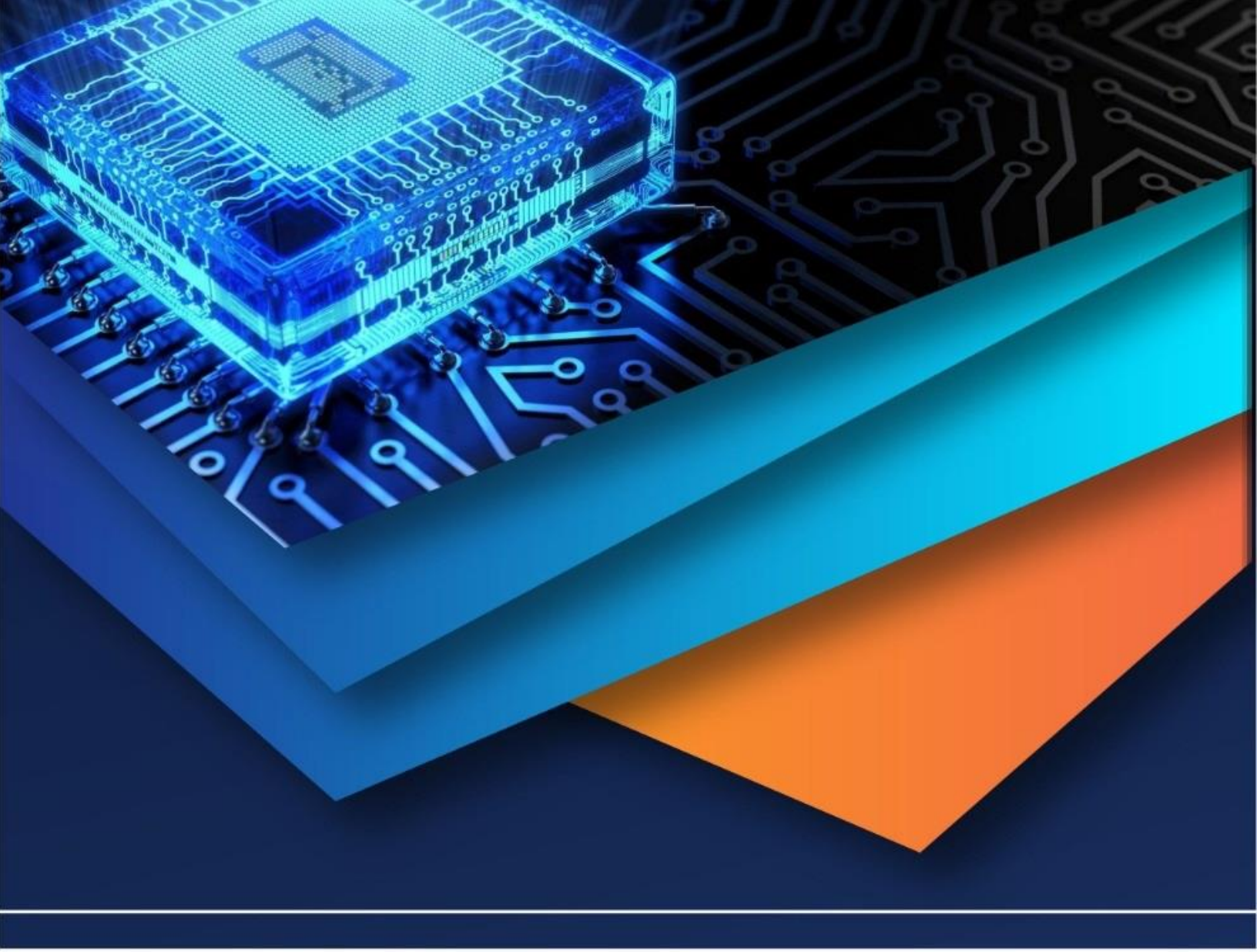
3) Mobile Portability: To increase accessibility the frontend will be adapted into a crossplatform mobile application using React Native.

4) Recruiter-Side Dashboard: The system's logic can be reversed to aid employers. We plan to build a "Batch Processing" module that allows recruiters to upload hundreds of resumes simultaneously and rank them based on semantic relevance rather than keyword density.

REFERENCES

S. No.	Author(s)	Title / Report Name	Conference / Journal / Publisher	Volume / Issue / Details	Year	Pages
1	A. Gupta and R. Kumar	Comparative Analysis of Automated Resume Evaluation Tools	International Journal of Computer Applications	Vol. 180, Issue 5	2021	12–18
2	S. Sharma and V. P. Singh	Implementation of Automated Resume Screening using TF-IDF	IEEE International Conference on Computing (ICC)	—	2020	45–50
3	K. Smith and J. Doe	Ontologybased Information Extraction for Recruitment	Journal of Information Science	Vol. 45, Issue 3	2019	301–315
4	M. Kumaran and A. Safvan	Resume Screening System using Natural Language Processing and Machine Learning	IEEE International Conference on Intelligent Systems (ICIS)	—	2022	1–6
5	S. Sinha, A. Das, and P. Dutta	Information Extraction from Resumes using BiLSTM and BERT	International Journal of Advanced Computer Science and Applications (IJACSA)	Vol. 12, Issue 4	2021	112–118
6	A. H. Al-Ghamdi and M. Al-Harbi	A Job Recommender System Based on Candidate Skills	International Journal of Advanced Computer Science and Applications	Vol. 11, Issue 2	2020	254–260
S. No.	Author(s)	Title / Report Name	Conference / Journal / Publisher	Volume / Issue / Details	Year	Pages
7	A. Vaswani, N. Shazeer, et al.	Attention Is All You Need (Transformer Architecture)	Advances in Neural Information Processing Systems (NeurIPS)	Vol. 30	2017	5998–6008

8	Z. Liu, Y. Lin, and M. Sun	A Survey on Transformerbased Pretrained Language Models	arXiv Preprint	arXiv:2106.071 39	202 2	1–12
9	C. O'Neil	Weapons of Math Destruction: How Big Data Increases Inequality	Crown Publishing Group	Book (ISBN: 978-0553418811)	201 6	100–125
10	M. Al-Bahadili	Modern Web Application Architecture: MERN Stack	Journal of Web Engineerin g	Vol. 20, Issue 1	202 1	40–55



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)