© DEC 2025 | IRE Journals | Volume 9 Issue 6 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I6-1712708

AI- Resume Analyzer

ZOYA KAUSER V[1], AAMNA NOORAIN[2], PANIKA N[3]
[1, 2, 3] *5th Semester B.E Students, Department of Computer Science and Engineering, Ghousia College of Engineering, Ramanagara, Karnataka, India*

Abstract- ***The aim of this project is to design and develop a tool that results into an easy and helpful solution for applicants as well as recruiters. The rapid growth of job applications across industries has created a significant challenge for recruiters who must manually evaluate large volumes of resumes. This process is often time-consuming, inconsistent, and prone to human bias. The AI Resume Analyzer aims to address these limitations by leveraging Artificial Intelligence and Natural Language Processing (NLP) to automate the process of resume screening and shortlisting. The system extracts key information such as skills, education, and work experience from resumes, and compares them with job descriptions to generate relevance scores. Using machine learning algorithms, it ranks candidates based on their suitability for the role, thereby improving efficiency, accuracy, and fairness in the recruitment process. The proposed solution provides a user-friendly interface, ensures faster decision-making, and supports organizations in streamlining their hiring workflow. This project demonstrates how AI-powered automation can significantly enhance modern recruitment practices.***

I. INTRODUCTION

The advancement of Artificial Intelligence and Natural Language Processing has opened new opportunities to automate and enhance the hiring process. The AI Resume Analyzer is designed to address these challenges by transforming how resumes are interpreted and evaluated. By intelligently extracting key information such as skills, experience, education, and achievements, the system provides an objective and data driven method to assess candidate relevance for a given job role. This project focuses on building a reliable, scalable, and efficient solution that streamlines the recruitment workflow. Through automated parsing, skill matching, and scoring algorithms, the AI Resume Analyzer reduces manual workload, improves accuracy in shortlisting candidates, and speeds up decision-making. It serves as a valuable tool for HR departments, placement cells, and job portals seeking to modernize their hiring practices with intelligent automation.

II. LITERATURE SURVEY & RELATED WORK

- The evolution of music players has historically prioritized storage and fidelity over adaptability.
- Manual & Static Systems: Traditional players (Winamp, iTunes) and modern streaming services (Spotify, Apple Music) rely on active user input. While "Mood Mixes" exist, they are static entities requiring the user to explicitly state, "I am sad," before the music matches their state.
- Wearable Sensor Systems: Research by Kirke et al. utilized EEG and galvanic skin response sensors to detect mood. While accurate, these methods are highly intrusive, expensive, and impractical for casual daily use.
- Facial Expression Analysis: Early attempts utilized geometric feature-based methods (measuring distances between eyes, mouth curvature). However, recent advancements in Deep Learning, specifically Convolutional Neural Networks (CNNs), have surpassed these methods in accuracy and robustness against minor facial variations.
- Research Gap: There is a distinct lack of lightweight, local-processing systems that integrate AI directly into a desktop playback engine without requiring external hardware, internet-dependent APIs, or intrusive wearables. This paper addresses this gap by proposing a privacy-focused, local-processing solution.

III. SYSTEM ARCHITECTURE

FRONTEND:

Built using Streamlit (Python-based web framework).

Runs in the user's web browser as a simple web app.

Implemented in: app.py

© DEC 2025 | IRE Journals | Volume 9 Issue 6 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I6-1712708

APPLICATION LAYER

Resume Parsing Module

- File: resume_parser.py
- Uses pdf plumber and docx2txt to extract raw text from uploaded PDF/DOCX resumes.

Text Processing & NLP Module

- File: text_processing.py
- Cleans text (remove extra spaces, special characters, lowercase).
- Uses spacy for tokenization and lemmatization (preprocessing for NLP).

Similarity Computation Module

- File: similarity.py

Keyword Extraction & Recommendation Module

- File: keyword_extraction.py

Result Aggregation & Business Logic

DATA LAYER

Input Data:

Uploaded Resume Files (PDF, DOCX).

Job Description Text entered by the user.

Model / Library Resources:

- spacy language model
- Python libraries: scikit-learn, YAKE, pdf plumber, docx2txt, etc.

OVERFLOW (End-to-End)

1. User opens the Streamlit web app in a browser.
2. User uploads a resume and pastes a job description.
3. resume_parser.py extracts text from the resume.
4. text_processing.py cleans and preprocesses both resume and job description text.
5. similarity.py computes the match percentage using TF-IDF and cosine similarity.
6. keyword_extraction.py extracts skills, finds missing skills, and recommends resources.
7. app.py displays the match score, missing skills, and recommendations on the UI.

IV. METHODOLOGY

The development of AI-Resume Analyzer follows a structured and modular approach, ensuring smooth integration between the frontend, backend, and deployment systems. a.Frontend: - • HTML5: HTML is the standard markup language for Web pages. With HTML you can create your own website. • CSS3: - CSS is the language we use to style an HTML document.CSS describes how HTML elements should be displayed. • JavaScript: - JavaScript is the world's most popular programming language. JavaScript is the programming language of the Web. • Streamlit: Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. In just a few minutes you can build and deploy powerful data apps. b. Backend: - • Python: - Python is a popular programming language. Python can be used on a server to create web applications. • JSON: - JSON is a text format for storing and transporting data. JSON is "self describing" and easy to understand. c. Requirement Analysis & Setup The setup includes: Python environment creation • • • Installing dependencies using requirements.txt Downloading spacy model • Running the Streamlit app • Python • Virtual environment (venv) • Dependency management (pip, requirements.txt) d. Testing While explicit test files are not included, testing can be inferred from: • Running the app locally (streamlit run app.py) • Checking output of NLP and similarity functions Testing is done manually through the Streamlit interface. Testing Approach: • Manual testing through Streamlit UI • Python debugging • Checking accuracy of similarity scores and keyword extraction.

V. EXPERIMENTAL RESULTS

The AI Resume Analyzer was tested using multiple sample resumes and job descriptions across different

domains such as software engineering, data science, and business analysis. The system successfully extracted text from both PDF and DOCX formats with high accuracy using the integrated parsing modules. The TF-IDF and cosine similarity–based matching algorithm produced consistent match percentages, ranging between 45% and 92% depending on the relevance between the resume and the job description.

Keyword extraction using YAKE effectively identified the core skills required for each role, and the system accurately highlighted missing skills in the submitted resumes. The skill-gap analysis also generated appropriate learning recommendations, demonstrating the usefulness of the recommendation module. The Streamlit interface displayed results in a user-friendly manner, ensuring smooth interaction and quick interpretation. Overall, experimental evaluation shows that the system performs efficiently, provides reliable match insights, and significantly reduces manual efforts in resume screening.

VI. ETHICAL & PRIVACY CONSIDERATIONS

1. Data Privacy and Confidentiality.

2. Responsible Use of User Data

3. Bias and Fairness in AI Decisions

4. Transparency in Scoring and Recommendations

5. Data Security and Safe Processing.

VII. CONCLUSION

The AI Resume Analyzer offers a powerful and efficient solution to the limitations of traditional resume screening. By leveraging Artificial Intelligence, Natural Language Processing, and machine learning techniques, the system automates the extraction, analysis, and comparison of resume data with job descriptions. This significantly reduces manual workload, minimizes human error, and ensures unbiased candidate evaluation. The tool provides clear insights such as match percentage, missing skills, and personalized recommendations, helping recruiters make faster and more accurate hiring decisions. Overall, the project demonstrates how AI-driven automation can transform recruitment into a smarter, more reliable, and highly efficient process that benefits organizations, HR teams, and job seekers alike.

VIII. FUTURE SCOPE

1. Integration with job portals and HR systems

2. Use of advanced deep learning models

3. Support for multimedia and OCR-based resumes

4. Multi-language resume analysis

5. Automated resume improvement suggestions

6. Predictive hiring and analytics

7. AI-powered candidate assistant chatbot

8. Cloud deployment and large-scale processing

REFERENCES

[1] Ghousia College of Engineering, Dept. of CS & Engineering, "AI- Resume Analyzer" Mini Project Report, 2025.

[2] Streamlit Documentation, "Streamlit — The fastest way to build data apps in Python." Available: https://docs.streamlit.io/

[3] IJITEE, "A Study on Resume Parsing Using Natural Language Processing," International Journal of Innovative Technology and Exploring Engineering, vol. 9, no. 7, 2020. Available: https://www.ijitee.org/wp-content/uploads/papers/v9i7/F4078049620.pdf

[4] Academia.edu, "Resume Parser with Natural Language Processing." Available: https://www.academia.edu/32543544/Resume_Parser_with_Natural_Language_Processing

[5] RChilli, "Resume Parsing 101 — Everything You Need to Know About Resume Parsing." Available: https://www.rchilli.com/blog/resume-parsing-101/

[6] Wikipedia, "Résumé Parsing." Available: https://en.wikipedia.org/wiki/R%C3%A9sum%C3%A9_parsing

[7] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine

Learning Research, vol. 12, pp. 2825–2830, 2011.

[8] Matthew Honnibal and Ines Montani, “spaCy 2: Natural Language Understanding with Bloom Embeddings,” Explosion AI, 2017.

[9] Ricardo Campos et al., “YAKE! Keyword extraction from single documents using multiple local features,” Information Sciences, vol. 509, 2020.

[10] pdfplumber Documentation, “PDF text extraction for Python.” Available: https://github.com/jsvine/pdfplumber