**ELEVATE LABS**

AI & ML Internship

Project report

**AI-Powered Resume Ranker**

Submitted by:

Venkata Sai Nalluri

Supervised by:

ELEVATE LABS

*Date of submission*: 23/05/2025

**Contents**

1. Abstract . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

2. Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

3. Data collection and pre processing . . . . . . . . . . . . .

4. Methodology . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

5. Experimental Setup . . . . . . . . . . . . . . . . . . . . . . . . . . .

6. Result . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

7. Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

8. References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

**Abstract**

Now a days many people are applying for the jobs in different companies and because of these the companies are getting lots of resumes for the job application. The companies want the people who has skills and is good for a particular job in the field. It is becoming a difficult task to give jobs. In the contemporary hiring landscape, efficiently identifying the most suitable candidates from a large pool of applicants is a critical yet time-consuming challenge.

This project presents an AI-Powered Resume Ranker, a web-based application that leverages Natural Language Processing (NLP) and machine learning techniques to automate the resume screening process. The system allows HR professionals to upload multiple resumes in PDF format alongside a job description. Using SpaCy for text preprocessing and TF-IDF vectorization with Scikit-learn, each resume is scored based on its relevance to the job requirements.

Additionally, AI-enhanced keyword extraction is performed via the Groq API to identify critical skills and qualifications, which are used to boost the ranking accuracy. The application provides a user-friendly interface built with Flask and TailwindCSS, enabling users to view ranked candidates and download detailed reports. This solution significantly reduces manual effort and enhances the precision of the recruitment process, making it a valuable tool for modern talent acquisition.

**Introduction**

In today’s fast-paced and highly competitive job market, organizations receive an overwhelming number of job applications for every open position. Manually reviewing and evaluating these resumes is not only time-consuming but also prone to human bias and inconsistency. To address these challenges, the integration of Artificial Intelligence (AI) and Natural Language Processing (NLP) into recruitment processes has become increasingly vital.

This project, titled **AI-Powered Resume Ranker**, is designed to streamline and enhance the resume screening process using modern AI techniques. It enables hiring teams to upload multiple resumes along with a job description and automatically ranks candidates based on their relevance to the role. The system incorporates NLP through SpaCy for text preprocessing and uses TF-IDF vectorization to numerically represent resumes and job descriptions. Cosine similarity is then applied to assess how closely each resume matches the job requirements.

To further refine the ranking mechanism, the project integrates the Groq API for AI-driven keyword extraction. This helps identify the most significant skills and qualifications from the job description, which are used to improve scoring accuracy. The solution is deployed as a web application using Flask, with a sleek and intuitive user interface built using HTML and TailwindCSS.

By automating the initial screening phase, this application not only saves time but also ensures a more objective and data-driven selection of candidates. It is particularly beneficial for HR teams and recruitment agencies seeking to optimize their hiring workflows and improve decision-making based on resume relevance and role fit.

**Data collection and preprocessing**

The effectiveness of any AI-driven system heavily relies on the quality of data and the robustness of preprocessing techniques. In this project, data is collected in the form of PDF resumes submitted by users through a web interface. Along with the resumes, a textual job description is provided by the recruiter or HR personnel to define the criteria for candidate evaluation.

1. Data Collection:

* **Resume Uploads:** Users upload resumes in PDF format via the frontend form. Each file is processed individually using the PyPDF2 library to extract raw text content from each page.
* **Job Description Input:** The job description is manually entered through a text area in the web form. This description outlines the skills, experience, and qualifications required for the job.

2. Text Extraction:

* **PDF Parsing:** PyPDF2.PdfReader is used to parse each PDF and extract text from all pages.
* **Error Handling:** Basic exception handling is implemented to manage improperly formatted or unreadable PDF files, ensuring robustness during file processing.

3. Preprocessing with SpaCy:

To standardize and clean the extracted text for further analysis, the following NLP preprocessing steps are applied using the SpaCy library:

* **Lowercasing:** Converts all text to lowercase to ensure uniformity.
* **Tokenization:** Breaks down the text into individual words or tokens.
* **Lemmatization:** Converts each word to its base or dictionary form, enhancing the matching accuracy.
* **Stopword Removal:** Eliminates common words (e.g., "and", "the", "is") that do not contribute meaningful information.
* **Punctuation Removal:** Filters out punctuation marks to focus on textual content.

This preprocessing pipeline ensures that the resumes and job descriptions are clean, comparable, and ready for vectorization and scoring. It also reduces noise in the data, which could otherwise affect the accuracy of the similarity calculations and overall ranking quality.

Tools used:

• Python: Core programming language for backend logic.

• SpaCy: NLP library for text preprocessing and tokenization.

• Scikit-learn: For TF-IDF vectorization and cosine similarity calculations.

• PyPDF2: For extracting text from PDF resumes.

• Flask: Web framework for building the user interface and backend. 1

• Pandas: For generating CSV reports.

• Groq API: AI-powered keyword extraction from job descriptions. Used for the rating of the applicants resumes.

• HTML/CSS: used for creating the web interface.

**Methodology**

The core objective of the AI-Powered Resume Ranker is to efficiently evaluate resumes against a given job description using a Natural Language Processing (NLP), machine learning techniques, and AI-enhanced keyword extraction. The methodology consist ofpipeline from data input to final ranking and reporting:

1. User Input Collection:

* The user provides a **job description** via a text input field and uploads **multiple PDF resumes** through a web interface.
* These inputs are submitted to the Flask backend for processing.

2. Resume Text Extraction:

* Text from each page is extracted and concatenated into a single string representing the full content of each resume.

3. Text Preprocessing:

* Both the job description and the resume texts undergo preprocessing.
  + Lowercasing
  + Tokenization
  + Lemmatization
  + Stopword and punctuation removal
* This ensures uniformity and reduces linguistic noise, improving the relevance of comparisons.

4. AI-Powered Keyword Extraction:

* The preprocessed job description is sent to the **Groq API**, which uses AI to extract key **skills**, **technologies**, and **qualifications**.
* The extracted keywords are added to the job description text to enrich the semantic information used in the next phase.

5. Vectorization using TF-IDF:

* The enriched job description and all resumes are converted into vector representations using **Term Frequency–Inverse Document Frequency (TF-IDF)** from Scikit-learn.
* This technique captures the importance of terms relative to each document and the entire dataset.

6. Similarity Computation and Scoring:

* **Cosine similarity** is calculated between the job description vector and each resume vector.
* Each score is further enhanced by checking for the presence of AI-extracted keywords in the original resume text.
* Final score formula:

Final Score=Cosine Similarity+0.5×(Total KeywordsNumber of Keyword Matches )

7. Resume Ranking:

* Resumes are sorted in descending order of their final score.
* Each resume is associated with a score and rank.

8. Result Presentation and Reporting:

* The ranked results are displayed on a dynamic HTML page using TailwindCSS.
* A **CSV report** of the ranked resumes is generated using Pandas and made available for download.

This methodology ensures that resumes are evaluated both statistically and semantically, resulting in a robust and reliable ranking system tailored for HR screening needs.

**Experimental setup**

The experimental setup for the AI-Powered Resume Ranker focuses on creating a controlled environment to test and validate the system's ability to accurately rank resumes based on a given job description.

1. Application Configuration:

* **Groq API:** API keys were securely integrated and used to fetch AI-generated keywords from job descriptions.
* **UI Setup:** TailwindCSS was used to style the index.html and results.html files, providing a modern and responsive design.

2. Sample Dataset:

* **Job Description:** A manually written job description outlining the required skills and qualifications for roles such as NLP Engineer, Web Developer, and Blockchain Developer.
* **Resumes:** Four sample resumes were collected in PDF format, each representing a candidate with a different skill set and background.

5. Execution Flow:

* The application is started locally using python resume\_ranker.py.
* The frontend is accessed via <http://localhost:5000/>.
* Resumes and a job description are uploaded through the UI.
* The backend processes the inputs, calculates scores, and renders results.
* A CSV file containing the ranked results is generated and made available for download.

**Results**

The AI-Powered Resume Ranker was tested using multiple sample resumes and job descriptions to evaluate its performance in ranking candidates based on their relevance to a given job profile. The results demonstrate the effectiveness of the implemented methodology in prioritizing candidates who closely match the job requirements.

1. Input Summary:

* **Job Description Example:** Required skills included Python, NLP, and machine learning.
* **Uploaded Resumes:** Four PDF resumes representing candidates with backgrounds in:
  + NLP and machine learning
  + Front-end development
  + Web3 and blockchain technologies
  + General software development

2. Ranking Output:

The system produced a ranked list of resumes with calculated scores based on:

* Cosine similarity between the job description and resume content
* Number of AI-extracted keywords present in each resume.

**Example Output Table:**

|  |  |  |
| --- | --- | --- |
| **Rank** | **Resume** | **Score (%)** |
| 1 | Emily\_Williams.pdf | 87.5% |
| 2 | Deepak\_Padhi.pdf | 74.3% |
| 3 | Jane\_Smith.pdf | 58.2% |
| 4 | Alex\_Johnson.pdf | 47.6% |

3. CSV Report:

A downloadable CSV file was generated containing:

* Resume filename
* Computed score (float)
* Rank (implied by ordering)

4. Performance Observations:

* **Accuracy:** Resumes with strong matches to the job description consistently received higher scores.
* **Robustness:** The system handled various PDF formats and lengths effectively.

5. User Feedback:

* The UI was intuitive and visually appealing.
* The ranking was perceived as relevant and aligned well with human judgment.
* The ability to download a report was appreciated for documentation and further processing.

**Conclusion**

The AI-Powered Resume Ranker successfully automates resume screening by combining NLP techniques, AI keyword extraction via the Groq API, and a user-friendly Flask interface. The system efficiently ranks resumes based on their relevance to a job description, providing HR teams with a scalable tool to streamline recruitment. The downloadable CSV report ensures easy integration into existing workflows.

The AI-Powered Resume Ranker successfully automates the initial screening of resumes by leveraging NLP and AI-driven keyword extraction. It provides accurate and efficient candidate ranking based on job relevance, reducing manual effort and bias in the hiring process.

**Future enhancements** may include support for DOCX files, deeper semantic analysis using advanced AI models, integration with professional platforms like LinkedIn, and analytics dashboards for HR insights.

**References**

1. Resume Ranker: AI-Based Skill Analysis and Skill Matching System

Publisher: IEEE Authors:Nikethani Gangoda; Kavindu Piumal Yasantha; Chamina Sewwandi; Navindu Induvara; Samantha Thelijjagoda; Nishantha Giguruwa.

2. Automated Resume Parsing and Ranking using Natural Language Processing

Publisher: IEEE Authors: K Thangaramya; G Logeswari; Sudhakaran Gajendran; J Deepika Roselind; Neha Ahirwar.

3. Natural Language Processing (NLP) in AI-Driven Recruitment Systems Sudheer Devaraju Walmart, USA

4. Artificial intelligence: the way ahead for employee engagement in corporate India Authors: Manisha sexena , Dharmesh K. Mishra