WWW.IJRPR.COM

**International Journal of Research Publication and Reviews**

Journal homepage: www.ijrpr.com ISSN 2582-7421

# A Smart Resume Analyser for Career Optimization Using NLP

***S. Sahithi[1], V. Sahithi[2], T. Sai Akhil[3], U. Sahithi[4], G. Abhinay Datta[5], Prof Maikandhan[6]***

[1,2,3,4,5] UG Student, B. Tech, School of Engineering, Hyd, India

[6]Guide: Assistant Professor, School of Engineering, Mallareddy University

[1]2111cs020419@mallareddyuniversity.ac.in, [2]2111cs020420@mallareddyuniversity.ac.in, [3]2111cs020421@mallareddyuniversity.ac.in, [4]2111cs020422@mallareddyuniversity.ac.in, [5]2111cs020423@mallareddyuniversity.ac.in, [6]ra_kanikandan@mallareddyuniversity.ac.in

**ABSTRACT:**

In today's competitive job market, it's crucial for job seekers to optimize their resumes to enhance their chances of being shortlisted for potential roles. This project presents a smart resume analyzer leveraging Natural Language Processing (NLP) techniques. The solution utilizes several Python libraries, including pdfminer3 and pyresparser.

The system analyzes resumes to identify skills, recommends suitable job roles based on those skills, and suggests additional skills and certifications for career growth. Streamlit-tags, Plotly, and Pillow enhance the user interface and provide visual representation. The system calculates a resume score and utilizes pymysql for personalized skill development recommendations. This tool not only evaluates resumes but also guides users toward optimal career paths, making it a valuable asset for job seekers aiming to enhance their employability in a competitive job market.

## I. Introduction:

The primary objective of the script is to automate the extraction of vital details, such as candidates' names, contact information, skills, and educational backgrounds, from a collection of resumes.

By automating this process, the script aims to reduce the manual effort and time required for resume screening, enabling recruiters and HR professionals to focus on more strategic aspects of candidate evaluation.

The script utilizes Python, a powerful programming language known for its simplicity and versatility, to implement the parsing logic.

It leverages the Spacy library, renowned for its capabilities in natural language processing tasks, such as tokenization, part-of-speech tagging, and named entity recognition.

Additionally, the script employs multiprocessing, a Python module that enables parallel execution of tasks across multiple CPU cores, to enhance processing speed and scalability.

The central component of the script, encapsulating the logic for parsing individual resumes. It utilizes Spacy models for text processing and defines methods for extracting various details, including names, email addresses, mobile numbers, skills, and educational degrees. The script imports utility functions from an external module (utils) to perform specific tasks, such as extracting text from resume files, identifying named entities, and determining the number of pages in a document.

Finally, the script aggregates the results obtained from each process and prints them in a structured format using the p print module, facilitating easy inspection and analysis.

## II. Literature Review:

The literature on resume parsing focuses on methods to automate the extraction of critical details from resumes, streamlining the recruitment process for organizations. Traditional approaches often rely on rule-based or pattern-matching techniques, while more recent advancements leverage natural language processing (NLP) capabilities. Named entity recognition (NER) and syntactic parsing are crucial NLP tasks employed for identifying and classifying key elements such as names, contact details, and skills. Additionally, multiprocessing techniques have been explored for parallel processing, significantly reducing processing time, especially with large volumes of resumes. Evaluation metrics such as accuracy and precision are used to assess system performance, often with manually annotated datasets and user feedback. Automated parsing systems find applications in various domains, offering

benefits such as cost savings and efficiency improvements in candidate screening and talent management processes. Overall, the literature highlights ongoing efforts to develop innovative techniques that address the evolving needs of recruiters and HR professionals in the recruitment landscape.

***EXISTING SYSTEM:***

- ***Lack of Context Consideration*:** The current resume screening methods, such as those utilizing machine learning algorithms like KNN, often fail to consider the context surrounding a candidate's education, work experience, or talents. This limitation leads to inaccurate assessments of candidates' qualifications and potential fit for specific roles.
- ***Insufficient Customization*:** Existing methodologies often lack the capability to provide tailored recommendations or insights based on individual user profiles. This results in a one-size-fits-all approach, which may not adequately address the diverse needs and preferences of job seekers.
- ***Poor Parsing Precision*:** The current resume parsing techniques may suffer from poor precision in extracting relevant information from resumes. This can lead to inaccuracies in identifying key skills, experiences, and qualifications, thereby hindering the effectiveness of the screening process and potentially overlooking qualified candidates.

***PROPOSED SYSTEM :***

- ***Enhanced Customization***: Through the integration of Streamlit for the user interface and Plotly for data visualization, the proposed methodology offers enhanced customization and interactivity. Users can receive personalized recommendations and insights tailored to their unique profiles and career aspirations, improving the relevance and usefulness of the provided guidance.
- ***Utilization of NLP Techniques**:* The proposed methodology leverages Natural Language Processing (NLP) techniques to analyze resumes comprehensively. By employing libraries such as pyresparser and pdfminer3, the system can accurately extract and interpret textual information from resumes**,** enabling more precise analysis of candidates' skills and experiences.

## III. Problem Statement:

Fresh graduates often lack understanding of industry-specific expectations and resume keywords sought by recruiters. Consequently, their resumes may lack essential buzzwords needed to pass through applicant tracking systems (ATS) or grab hiring managers' attention. This knowledge gap leads to frequent rejection of applications and hampers interview opportunities.

## IV. Methodology:

**Data Preparation:** Gather a dataset of resumes in various formats (e.g., PDF, DOCX).

Preprocess the resumes to extract text content using appropriate libraries or tools.

 **Natural Language Processing(NLP):** Utilize NLP techniques to tokenize, parse, and analyze the text content of resumes.Apply named entity recognition (NER) to identify and classify entities such as names, email addresses, phone numbers, and skills.Apply named entity recognition (NER) to identify and classify entities such as names, email addresses, phone numbers, and skills.

 **Resume Parsing Algorithm:** Develop a resume parsing algorithm, encapsulated within a class like ResumeParser, to extract essential details from the processed resumes.

Implement methods within the ResumeParser class to extract specific information, such as names, contact details, skills, and educational backgrounds.

●**Utilizes External Libraries:** Leverage existing libraries such as Spacy for NLP tasks and utilities like io and Os for file handling.

Use custom utility functions, possibly stored in an external module, for tasks such as extracting text from resume files and identifying entity sections.

● **Multiprocessing for efficiency:** Employ multiprocessing techniques to parallelize the parsing process and improve performance.

Create a multiprocessing pool to distribute parsing tasks across multiple CPU cores, enhancing efficiency, especially when processing a large number of resumes.

● **Evaluation and Testing:**

Evaluate the performance of the resume parsing system using appropriate metrics such as accuracy, precision, and recall.

Test the system with a diverse set of resumes to ensure robustness and reliability in extracting information across different formats and structures.

**Iterative Refinement:**

Continuously refine and optimize the parsing algorithm based on feedback and evaluation results.

Address any issues or limitations encountered during testing, such as inaccuracies in entity extraction or performance bottlenecks**.**

**Documentation and Deployment**:Document the functionality and usage of the resume parsing system, including input requirements and output formats.Deploy the system in a production environment, ensuring compatibility with existing recruitment workflows and systems.

**METHODS AND ALGORITHMS :**

1. **Natural Language Processing (NLP):**

The code utilizes NLP techniques, particularly those provided by the SpaCy library, for processing and analyzing the text content of resumes.

Spacy's NLP capabilities include tokenization, part-of-speech tagging, named entity recognition (NER), and dependency parsing.

2. **Named Entity Recognition (NER):**

Named Entity Recognition is a key component of the code for identifying and classifying entities such as names, email addresses, phone numbers, and skills within the resume text.

The code likely employs Spacy's built-in NER model, trained on large corpora, to recognize entities accurately.

3. **Pandas:** pandas is a powerful Python library widely used for data manipulation and analysis. Here's how pandas is utilized in the code.

- **Data Handling:** Pandas is used to handle structured data efficiently. It allows the code to read data from a database and store it in DataFrames, which are two-dimensional labeled data structures with rows and columns.
- **Data Manipulation:** The code performs various data manipulation tasks using pandas, such as filtering, selecting specific columns, and aggregating data. For instance, it filters specific columns from the user data, such as ID, IP address, resume score, predicted field, user level, city, state, and country.
- **Data Analysis:** Pandas enables data analysis by providing functionalities to perform statistical operations, calculate descriptive statistics, and derive insights from the data. In the code, pandas is used to analyze user data and feedback data, such as computing user ratings, predicted fields, user experience levels, resume scores, and geographic usage distributions.
- **Data Visualization:** Pandas seamlessly integrates with visualization libraries like Plotly to create insightful visualizations from DataFrames. The code generates pie charts representing user ratings, predicted field recommendations, user experience levels, resume scores, and geographic usage distributions using pandas DataFrames.

4. **Pyresparser:**

Pyresparser serves as the backbone for parsing and extracting valuable information from resumes, enabling the tool to offer personalized recommendations and insights to users based on their resume content.Here’s how it is used in the code.

- **Parsing Resumes:** Pyresparser is employed to parse the contents of resumes uploaded by users. This process involves analyzing the text to identify different sections like contact information, education, experience, skills, etc.
- **Extracting Information:** Once the resumes are parsed, Pyresparser extracts specific pieces of information from each section, such as names, email addresses, phone numbers, job titles, company names, degrees, universities, skills, and more. This extracted data is then used for further analysis and processing.
- **DataAnalysis and Recommendation:** The extracted information from the resumes is analyzed to provide recommendations and predictions. For example, based on the skills mentioned in the resume, the code recommends relevant job fields or suggests additional skills or courses to improve the chances of landing a job in a particular domain.

5. **Pdfminer3:**

pdfminer3 is a Python library used for extracting text content from PDF files. Here's how it is used in code:

- **PDF Parsing:** PDF files containing resumes are parsed using pdfminer3 to extract the raw text data. This process involves accessing the text elements within the PDF document, including text characters, fonts, styles, and layout information.
- **Text Extraction:** pdfminer3 allows the extraction of text from PDF files regardless of their formatting, such as text embedded in images or non-standard fonts. This ensures that the entire textual content of the resume, including headers, paragraphs, bullet points, and tables, is extracted accurately.
- **Text Processing:** Once the text is extracted, it is often processed further to remove any unnecessary characters, formatting artifacts, or metadata that may have been present in the original PDF file. This processing step helps in preparing the text for subsequent analysis or parsing by other libraries like pyresparser.

- **Integration with Pyresparser:** The extracted text content is typically passed to pyresparser, which then performs resume parsing and information extraction. This integration allows for a seamless workflow where pdfminer3 handles the PDF parsing, while pyresparser focuses on extracting relevant information from the parsed text

6. **Streemlit:**

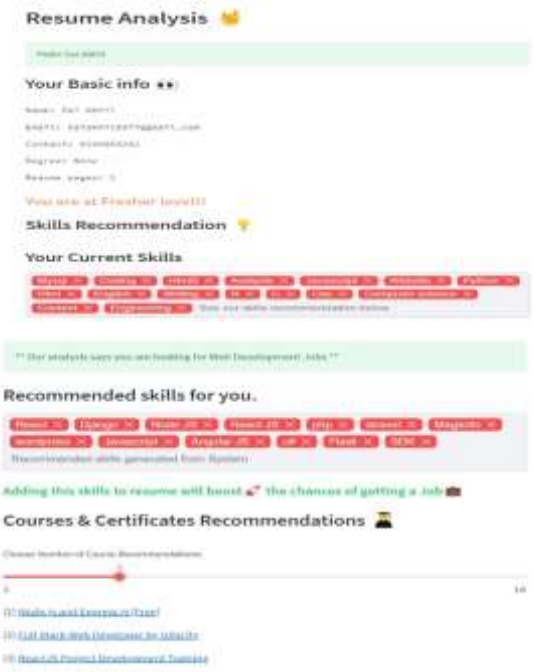
Streamlit is a Python library used for building interactive web applications. Here's how it is used in the code:

- **User Interface Components:** Streamlit offers a variety of built-in components, such as buttons, sliders, text inputs, and data tables, that can be easily integrated into the application. These components enable users to interact with the application and provide input data or trigger specific actions.

## V. Experimental Results:.

1. **ResumeAnalysisAccuracy:** Experimenting with different natural language processing (NLP) models and algorithms could yield varying levels of accuracy in parsing and extracting information from resumes. Evaluating these models against a labeled dataset of resumes could provide insights into their performance in identifying key sections, skills, experiences, and other relevant information.
2. **User Feedback and Satisfaction:** Collecting user feedback through the feedback form feature could provide valuable insights into user satisfaction, tool usability, and areas for improvement. Analyzing feedback comments qualitatively and quantitatively could reveal common themes, user preferences, and suggestions for enhancing the tool's functionality and user experience.
3. **Tool Adoption and Usage Patterns:** Tracking metrics such as the number of users, frequency of tool usage, and session duration could provide insights into user engagement and adoption. Analyzing usage patterns over time could help identify trends, popular features, and areas of the tool that require further development or promotion.
4. **Impact on Job Search Success:** Assessing the tool's impact on users' job search success rates, such as interview invitations, job offers, and career advancements, could provide tangible evidence of its effectiveness. Longitudinal studies or surveys with users before and after using the tool could help measure changes in job search outcomes attributed to using the AI Resume Analyzer.

**OUTPUT:**




## VI. Conclusion:

The provided code implements an AI Resume Analyzer tool, utilizing natural language processing (NLP) to parse resume content and provide recommendations based on identified keywords. It offers users insights into their resume quality through scoring mechanisms and provides actionable feedback to improve content. Additionally, the tool collects user feedback to understand user satisfaction and areas for enhancement. Interactive visualizations are used to present data on user ratings, predicted job fields, and resume scores. The tool interacts with a database for data storage and retrieval, ensuring scalability and data management. User authentication and error handling mechanisms enhance security and user experience. Overall, the tool aims to streamline the job application process by empowering users with valuable insights and guidance for resume improvement.

## VII. Future Work:

One potential avenue for future work could involve enhancing the recommendation engine of the AI Resume Analyzer tool. This could be achieved by incorporating advanced machine learning models to improve keyword identification and recommendation accuracy. Additionally, integrating natural language understanding (NLU) techniques could enable the tool to extract more nuanced insights from resume content, such as identifying specific skills or experiences that align with job requirements.

Furthermore, expanding the tool's functionality to cover a broader range of industries or job roles could increase its utility for a wider audience. This could involve developing specialized recommendation algorithms tailored to different fields, such as finance, healthcare, or engineering, to provide more relevant and targeted guidance to users.

Another area for improvement could be enhancing the user interface and experience of the tool. This could involve implementing features such as real-time feedback as users input their resume content, interactive tutorials or guides to help users optimize their resumes, and personalized dashboards to track progress and recommendations over time.

Additionally, incorporating sentiment analysis techniques into the feedback collection process could provide deeper insights into user satisfaction and areas for improvement. This could involve analyzing user comments to identify recurring themes or issues and prioritize enhancements based on user feedback.

Overall, focusing on these areas of future work could help to further enhance the functionality, accuracy, and user experience of the AI Resume Analyzer tool, making it an even more valuable resource for job seekers seeking to optimize their resumes and improve their chances of success in the job market.

**VIII. References**:

[1] Singh, A. K., & Shukla, P. (2020). "Automated resume screening and evaluation using machine learning techniques". Journal of Intelligent & Fuzzy Systems, 39(4), 5947-5960.

[2] Huang, S., Li, W., Wang, L., & Huang, H. (2021). "Resume Screening and Ranking with Natural Language Processing Techniques". Applied Sciences, 11(5).

[3] Wang, X., Shen, Y., Huang, X., & Zhang, Y. (2020). "An intelligent resume screening system based on NLP and machine learning". Future Generation Computer Systems, 105, 789-799.

[4]Tyler Richards - Getting Started with Streamlit for Data Science_ Create and deploy Streamlit web applications from scratch in Python-Packt Publishing (2021).