Crux Internship Assignment

Md Aakib Alam Ansari

February 18, 2024

1. OVERVIEW:

The CV Shortlister Backend is a Django-based server application designed to automate the process of shortlisting resumes according to job descriptions using Google's generative AI. The system parses resumes provided in PDF format, extracts relevant information such as education, projects, and work experiences, and calculates a relevancy score based on similarity to the provided job description, showing only the top resumes based on this relevancy score.

2. IMPLEMENTATION DETAILS:

- **Framework:** Built using Django, a high-level Python web framework, for rapid development and clean, maintainable code.
- **Modules:** The application is structured into multiple modules:
  - `api`: Handles API endpoints and logic for parsing resumes and extracting information.
  - `projects.py, education.py, workex.py`: Modules responsible for extracting projects, education, and work experience sections from resumes, respectively, using Google's generative AI.
  - `embeddings.py`: This module calculates similarity scores between textual data using pre-trained models from Google's generative AI library. It generates embeddings for input text and computes similarity scores using cosine similarity.
  - `apikey.py`: Manages the API keys required for accessing Google's generative AI services. It dynamically rotates between multiple API keys to ensure continuous availability and efficient utilization of resources.
  - `parser.py`: Contains functions to extract basic information like name and email from resumes using regular expressions.
- **Dependencies:** Utilizes PyPDF2 for parsing PDF resumes, NLTK for text processing tasks, and Google's generative AI library for content generation and extraction.
- **API Endpoint:** Provides a single endpoint '/submit' for receiving job descriptions and resumes, processing them, and returning shortlisted candidates with relevant information.

3. HYPERPARAMETERS:

- **Generative AI Configuration:** Hyperparameters such as temperature, top_p, top_k, and max_output_tokens are configured for Google's generative AI model. These parameters control the generation of content and the length of the output text.

4. INTEGRATION WITH REACT FRONTEND:

- **Frontend Development:** A React-based frontend application has been developed to interact with the backend server. This frontend provides a user-friendly interface for users to input job descriptions and upload resumes for shortlisting.
- **Github repository:** https://github.com/AakibAlam/cv-shortlister-frontend
- **Deployment:** The project has been deployed on Azure App Services for testing and accessibility purposes. https://mango-flower-01fa6f610.4.azurestaticapps.net/

5. DEPLOYMENT ON AZURE:

- **Successful Deployment:** The project has been successfully deployed on Azure App Services for hosting the backend server.
- **Integration Challenges:** While both the frontend and backend components are deployed individually and functioning correctly, there are integration issues between them when deployed on Azure. These issues are currently being addressed, and efforts are underway to resolve them.
- **Continuous Improvement:** Work is ongoing to ensure seamless integration between the frontend and backend on the Azure platform. Once resolved, the deployed project will provide a fully functional solution for automating resume shortlisting.

6. EVALUATION RESULTS:

- **Relevancy Score Calculation:** The relevancy score for each resume is calculated based on the similarity of project descriptions and work experiences to the provided job description. The maximum relevancy score among projects and work experiences is averaged to obtain the final score.
- **Threshold:** Resumes with a relevancy score higher than a predefined threshold value (threshold_value_for_resume_selection) are considered for shortlisting.
- **Evaluation Metrics:** The effectiveness of the shortlisting algorithm can be assessed using metrics like precision, recall, and F1-score. Regular testing with various resumes and job descriptions is crucial for refining the algorithm and enhancing its accuracy over time.

7. INSIGHTS GAINED:

- **Deployment Challenges:** Deploying complex applications involving frontend and backend components requires thorough testing and debugging to ensure seamless integration.
- **Cloud Deployment:** Deploying on cloud platforms like Azure App Services offers scalability and accessibility benefits but requires careful configuration to ensure compatibility between frontend and backend components.
- **Effective Utilization of AI:**Leveraging Google's generative AI model enables automated extraction of detailed information from resumes, enhancing the efficiency of the shortlisting process.
- **Challenges:** Handling variations in resume formats and structures poses challenges in accurate parsing and extraction. Regular updates and refinements to the parsing algorithms are necessary to handle diverse resume styles.

- **Scalability:** The modular structure of the application allows for easy scalability and maintenance. Additional features and improvements can be integrated seamlessly to enhance functionality and performance.

8. CONCLUSION:

The CV Shortlister Backend, along with its React-based frontend, offers a comprehensive solution for automating resume shortlisting. While the backend functions smoothly in both local and cloud environments, efforts are underway to resolve integration issues with the frontend on Azure App Services. Once resolved, the deployed project will provide recruiters with an efficient tool for screening resumes based on job descriptions.

Link to deployment: https://mango-flower-01fa6f610.4.azurestaticapps.net/