Documentation for Generative AI-Powered Resume Processing

Approach Overview

This project utilizes Generative AI and OCR (Optical Character Recognition) technologies to automate the process of extracting structured insights from resumes provided in PDF format. The solution focuses on accurately extracting critical fields such as personal details, educational background, skills, certifications, and AI/ML-related experience scores. The solution is implemented in Python, leveraging libraries and tools such as Google Generative AI, pdf2image, pytesseract, pandas, and concurrent processing for high efficiency.

Key Steps in the Approach:

1. **File Ingestion**:
    - The system takes a folder of resumes in PDF format from a specified Google Drive folder.
2. **Text Extraction**:
    - For text-based PDFs, the text is extracted directly using the PyPDF2 library.
    - For image-based PDFs, the text is extracted using Tesseract OCR after converting each PDF page to images.
3. **AI-Powered Insight Generation**:
    - Resume text is analyzed using the Google Generative AI API (Gemini-1.5 Flash model).
    - The AI model generates structured insights based on a predefined prompt and scoring mechanism for specific fields.
4. **Parallel Processing**:
    - The system processes multiple resumes simultaneously using Python's `concurrent.futures.ThreadPoolExecutor` for efficient batch handling.
5. **Output**:
    - The extracted and processed data is saved in an Excel file in a structured format for easy analysis.

Innovative Features and Use of Generative AI

Use of Generative AI:

The project integrates Google Generative AI to:

- Extract structured data from unstructured resume text.
- Assign scores to candidates for **Generative AI Experience** and **AI/ML Experience** based on predefined criteria.

- Infer key details (such as missing certifications or project details) when explicitly mentioned in the resume.

Innovative Features:

1. **Dynamic Scoring Mechanism**:
    - The AI model categorizes candidates' experience levels for Generative AI and AI/ML into three tiers: Exposed, Hands-on, and Advanced.
2. **OCR Optimization**:
    - Tesseract is configured with custom parameters (`--oem 3 --psm 6`) to improve OCR accuracy, enabling better text extraction from complex resume formats.
3. **Parallel Processing**:
    - Resumes are processed in parallel, significantly reducing processing time for large batches of resumes.
4. **Flexible Text Extraction**:
    - A hybrid approach is used to handle both text-based and image-based PDFs seamlessly.
5. **Customizable Prompt Design**:
    - The Generative AI prompt can be adjusted to include additional fields or scoring mechanisms, making the system adaptable to changing requirements.
6. **Scalable Solution**:
    - Designed to handle a high volume of resumes efficiently, making it suitable for enterprise-level HR operations.

Libraries and Tools Used

- **Google Generative AI (Gemini-1.5 Flash model)**: For contextual data extraction and scoring.
- **PyPDF2**: To check if a PDF is text-based and extract text directly.
- **pdf2image**: To convert PDF pages to images for OCR processing.
- **pytesseract**: To extract text from images using OCR.
- **pandas**: For data handling and exporting processed data to Excel.
- **concurrent.futures**: For parallel processing of resumes.
- **Google Drive API**: To access resumes stored in Google Drive.

Example Output Fields

For each resume, the following fields are extracted and structured:

1. **Name**: Candidate's full name.
2. **Contact Details**: Email and phone number.
3. **University**: Educational institution.

4. **Year of Study**: Timeframe of education.
5. **Course**: Program of study.
6. **Discipline**: Field of study.
7. **CGPA/Percentage**: Academic performance.
8. **Key Skills**: List of skills.
9. **Certifications**: List of certifications.
10. **Internships**: Internship details.
11. **Projects**: Names and brief descriptions of projects.
12. **Gen AI Experience Score**: Score (1-3) based on experience.
13. **AI/ML Experience Score**: Score (1-3) based on experience.

Usage Instructions

1. Install Dependencies

Install the required libraries and dependencies by running the following commands in your Colab notebook:

```
!apt-get update
!apt-get install -y poppler-utils tesseract-ocr
!pip install pdf2image pytesseract pandas google-generativeai openpyxl
!pip install pypdf2
```

2. Google Drive Setup

- Mount your Google Drive by running:
- `from google.colab import drive`
  `drive.mount('/content/drive')`

- Place all the resumes in a folder inside your Google Drive, e.g.,
  `/content/drive/MyDrive/Resume/Resume`

3. Set Up the API Key

Store your API key in the Google Colab Secrets:

1. Click on the **"Secrets"** tab in the left sidebar.
2. Add a new secret:
   - **Key Name**: API_KEY
   - **Value**: Your Google Generative AI API key.

Access it in code:

```
API_KEY = userdata.get('API_KEY')  # Retrieve the API key
genai.configure(api_key=API_KEY)  # Configure the API
```

4. Execute the Script in Colab

Run the provided Python script in Colab to process the resumes. The script will:

- Extract text from resumes (using OCR if necessary).
- Generate structured insights using Generative AI.
- Save the extracted data to an Excel file.

5. Scaling and Batch Processing

- **Parallel Processing**: The solution leverages Python's `concurrent.futures.ThreadPoolExecutor` to process multiple resumes simultaneously. This reduces processing time significantly when dealing with large datasets.
- **Scalability**: The system is designed to handle large batches of resumes by optimizing memory usage and processing time. Resumes are processed in chunks, ensuring that even high-volume datasets can be managed efficiently.
- **Batch Execution**: By specifying a folder path, the script automatically identifies all PDF files within the folder and processes them as a batch.

6. Output File

- The processed data will be saved as an Excel file at the specified location in your Google Drive. For example:

  ```
  output_file = "/content/drive/MyDrive/Resume/processed_resumes.xlsx"
  ```

- You can download or share this file for further analysis.

7. Example Execution

Assuming your resumes are located in `/content/drive/MyDrive/Resume/Resume`, run:

```
folder_path = "/content/drive/MyDrive/Resume/Resume"
processed_data = process_resumes_from_drive_parallel(folder_path)

# Save processed data to Excel
output_file = "/content/drive/MyDrive/Resume/processed_resumes.xlsx"
processed_data.to_excel(output_file, index=False, engine='openpyxl')
print(f"Processed data saved to {output_file}")
```

Future Enhancements

- **Real-Time Processing**: Implementing a web interface for real-time resume uploads and processing.
- **Expanded Fields**: Adding support for extracting additional fields such as hobbies or language proficiency.

- **Advanced Analytics**: Integrating with dashboards for visual analysis of processed data.
- **Multi-Language Support**: Enhancing OCR and AI capabilities to process resumes in multiple languages.

Conclusion

- This project demonstrates the power of combining Generative AI and OCR to streamline resume processing. By automating the extraction and analysis of structured insights, the solution saves significant time and effort for HR teams while ensuring scalability and accuracy. The use of advanced AI models and parallel processing techniques further enhances the efficiency and adaptability of the system. This approach can serve as a foundation for more comprehensive, AI-driven talent acquisition solutions in the future.