Document: Architectural Design and Challenges for Implementing CV Analyzer with Open-Source NLP Tools

# 1. Architecture Design

The architecture for the CV Analyzer solution leverages a combination of open-source natural language processing (NLP) models and libraries to extract relevant information from PDF and Word documents. The solution is designed to process unstructured text, detect personal and professional details, and display the extracted information using a GUI built with Tkinter. Below is a detailed explanation of the system architecture.

## 1.1. System Components

1. **File Selection Interface (Tkinter)**:
   * **Purpose**: Provides a graphical user interface (GUI) for users to upload a CV in either PDF or Word format.
   * **Input**: File path selected by the user via the file chooser.
   * **Output**: The path to the selected file is passed to the document processing module.
2. **Document Processing (DocumentAnalyzer Class)**:
   * **Purpose**: Reads and extracts text from the selected file. It supports two types of files:
     + **PDF Files**: Processed using the pdfplumber library.
     + **Word Documents (.docx)**: Processed using the python-docx library.
   * **Input**: PDF or Word document containing the CV.
   * **Output**: Raw text extracted from the document.
3. **Information Extraction (spaCy NLP)**:
   * **Purpose**: Performs named entity recognition (NER) and keyword-based extraction using the spaCy NLP model. The model processes the extracted text to find entities such as:
     + **Name**
     + **Email**
     + **Phone Number**
     + **Education Information**
     + **Work Experience**
   * **Input**: Raw text from the document processing step.
   * **Output**: Structured data in the form of a Python dictionary, containing extracted information (Name, Email, Phone, Education, and Work Experience).
4. **Custom Logic for Name Detection**:
   * **Purpose**: Enhances name detection by identifying the two words before the occurrence of the keyword "Education" as a fallback mechanism. This ensures the system captures a proper name when spaCy fails or returns incorrect results.
   * **Input**: Raw text and extracted name from spaCy.
   * **Output**: Corrected name based on fallback logic if the extracted name is incomplete or incorrect.
5. **User Interface for Display (Tkinter)**:
   * **Purpose**: Displays the extracted information in a structured format (Name, Contact Information, Education, and Work Experience) using the Tkinter library. Each piece of information is shown in its own labeled box with text fields.
   * **Input**: Extracted information from the NLP module.
   * **Output**: GUI displaying the extracted data with sections for Name, Email, Phone, Education, and Work Experience.

## 1.2. Data Flow Diagram

The architecture follows a step-by-step flow as illustrated below:

1. **File Selection (Tkinter)**:
   * User selects a file.
   * Path is sent to the DocumentAnalyzer.
2. **Document Processing (PDF/Word Processing)**:
   * Extracts text from the document.
   * Sends raw text to the NLP model.
3. **Information Extraction (spaCy)**:
   * Named entities and keywords are extracted from the text.
   * Results are structured into fields (Name, Contact Info, Education, Work Experience).
4. **Name Verification and Correction**:
   * Custom logic compares the detected name to fallback logic using text before "Education."
   * Corrected or verified name is returned.
5. **Display in GUI (Tkinter)**:
   * Information is displayed in a user-friendly interface.
   * Users can view, scroll, and interact with the fields.

# **2**. Challenges and Areas for Improvement

Despite the functionality of the CV analyzer, certain challenges were encountered during implementation, particularly when dealing with the wide variety of CV templates and formats. Below is a list of the key challenges and areas where additional work or improvements are needed:

## 2.1. Parsing and Text Extraction

* **Challenge**:
  + Extracting text from PDF files, especially those with complex layouts such as multi-column designs, images, or non-standard fonts, is difficult.
  + PDFs with different encoding formats might not always allow for clean text extraction, leading to missing or misaligned data.
* **Improvement Needed**:
  + Implement specialized parsing mechanisms to handle complex layouts. Tools like PyMuPDF or Tesseract OCR can be used as fallback options for PDFs with non-text elements or unusual encodings.

## 2.2. Inconsistent CV Templates

* **Challenge**:
  + CVs come in a wide range of formats and templates, making it difficult for spaCy to consistently extract the correct information. For instance, the position of the name, contact information, or section headers (such as "Education" and "Work Experience") can vary widely.
* **Improvement Needed**:
  + Train a custom model fine-tuned specifically for CVs. This model would be trained on a diverse set of CV templates to improve the consistency and accuracy of information extraction.
  + Use template recognition algorithms to detect standard sections like "Education" and "Work Experience" more reliably.

## 2.3. Name Extraction and Fallback Logic

* **Challenge**:
  + SpaCy sometimes fails to recognize the correct name in the document, especially when names are not clearly distinguished from other text. The fallback logic (i.e., selecting two words before "Education") is useful but not foolproof.
* **Improvement Needed**:
  + Implement a more sophisticated name extraction mechanism, such as looking at patterns of capitalization, font size, or the first few lines of the document. Additionally, heuristics like matching common name structures could improve accuracy.

## 2.4. Handling Non-English CVs

* **Challenge**:
  + The system currently relies on the en\_core\_web\_sm English model from spaCy. Non-English CVs or CVs with mixed-language content might not be processed accurately.
* **Improvement Needed**:
  + Add support for multi-language models in spaCy or use multilingual alternatives like BERT to handle non-English CVs more effectively.

## 2.5. Contact Information Extraction

* **Challenge**:
  + Extracting phone numbers and emails can sometimes lead to false positives if the document contains more than one phone number or email (e.g., references or previous employers’ contact details).
* **Improvement Needed**:
  + Implement additional filtering logic to detect and prioritize the CV owner's contact information, perhaps by examining proximity to the name or using section headers like "Contact Information."

## 2.6. Display Formatting in GUI

* **Challenge**:
  + For very large CVs with extensive education or work experience sections, the GUI display can become cluttered or hard to navigate, especially in terms of scrolling through content.
* **Improvement Needed**:
  + Introduce pagination or section collapse/expand functionality in the GUI to allow for more user-friendly navigation when viewing long documents.

# Conclusion

This CV analyzer solution effectively processes and extracts key information from CVs using open-source NLP tools like spaCy. However, challenges related to document format variety, text extraction consistency, and generalizability to non-English CVs need further attention. By addressing these issues, we can improve the system’s robustness, accuracy, and scalability.