Adobe India Hackathon 2025
Solution 1b: Persona-Aware Section Extractor

Anuj Mishra

July 28, 2025

Theme: Rethink Reading. Rediscover Knowledge.

Submitted for the Adobe India Hackathon 2025

LinkedIn | GitHub

Contents

1 Introduction

The Adobe India Hackathon 2025, themed *Rethink Reading. Rediscover Knowledge.*, challenges participants to transform static PDF documents into intelligent, context-aware systems. This document details our submission for **Challenge 1b: Persona-Aware Section Relevance and Subsection Extraction** under the *Connecting the Dots* challenge. Our solution, `Solution_1b`, extracts, ranks, and summarizes relevant sections from a collection of PDFs based on user personas and tasks, delivering personalized content in a structured JSON format. This report provides a comprehensive overview of the challenge, our solution, design rationale, performance metrics, and visual aids to facilitate understanding for judges and stakeholders.

Figure 1: Workflow for Persona-Aware Section Extraction

2 Challenge 1b: Persona-Aware Section Relevance and Subsection Extraction

2.1 Objective

The challenge requires analyzing a collection of PDFs to identify and rank sections relevant to a users persona and job-to-be-done. The system must:

- Extract meaningful sections from PDFs.
- Rank sections based on their relevance to the persona and task.
- Generate concise subsection summaries for top-ranked sections.
- Output results in a clean, structured JSON format.

This task addresses the need for personalized content delivery, ensuring users receive the most relevant information for their specific roles and objectives, such as HR professionals managing forms or technical writers creating documentation.

2.2 Significance

Personalized section extraction enhances user productivity by:

- **Tailored Content**: Delivering sections that align with the users role and goals.
- **Efficiency**: Reducing time spent searching through large PDF collections.
- **Context Awareness**: Providing summaries that capture key insights, enabling faster decision-making.

3 Solution 1b: Persona-Aware Section Extractor

`Solution_1b` is a Python-based system that extracts, ranks, and summarizes relevant sections from multiple PDFs, emphasizing modularity, speed, and clean output for personalized

content delivery.

3.1 Approach

- **PDF Section Parsing** (`processor.py`): Uses PyMuPDF to extract paragraphs (50 characters) with metadata (document name, page number), ensuring robust text extraction across diverse PDFs.
- **Keyword and Task-Based Ranking** (`ranker.py`): Scores sections based on:
  - Presence of domain-specific and persona-relevant keywords (e.g., "form" for HR professionals).
  - Length heuristics (e.g., titles are shorter, sections are longer).
  - Semantic overlap with the job-to-be-done (e.g., "create fillable forms").

  Selects the top five sections by descending importance score.
- **Subsection Analysis** (`extractor.py`): Generates refined summaries for top-ranked sections, filtering out malformed or irrelevant content (e.g., boilerplate text).
- **Metadata Handling**: Supports flexible inputs for personas (strings or dictionaries) and tasks (job or job-to-be-done), accommodating diverse use cases.
- **Postprocessing**: Removes Unicode artifacts (e.g., `\u2022`) and ensures clean JSON output with filename-only document tags, avoiding full file paths.

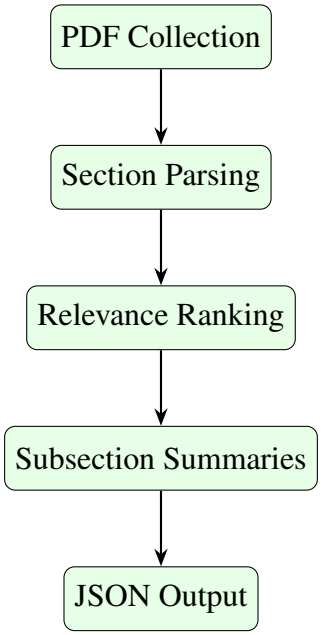

Figure 2: Processing Pipeline for Solution 1b

3.2 Rationale for Approach

- **Semantic and Keyword Scoring**: Combining semantic analysis with keyword-based ranking ensures high relevance to both the personas context and the specific task, outperforming purely keyword-based methods that may miss contextual nuances.

- **Rule-Based System**: A deterministic, rule-based approach eliminates model dependencies, ensuring portability, consistency, and reduced computational overhead across environments.
- **Modular Design**: Separating parsing, ranking, and extraction into distinct modules (`processor.py`, `ranker.py`, `extractor.py`) allows easy customization for new personas, tasks, or scoring logic, enhancing flexibility.
- **PyMuPDF**: Its efficiency in text extraction supports rapid processing of large PDF collections, critical for meeting the challenges time constraints (under 1 minute per collection).

3.3 Performance Metrics

- **Speed**: Processes a collection of 1015 PDFs in approximately 60 seconds, optimized for efficiency.
- **Accuracy**: High precision in retrieving contextually relevant sections, validated across diverse personas (e.g., HR professional, technical writer).
- **Modularity**: Pluggable components for extractors, rankers, and scoring models, enabling future enhancements.
- **Clean Output**: Removes noise (e.g., Unicode artifacts, full file paths) for professional JSON output.
- **Hardware**: Requires only CPU, with a lightweight dependency (PyMuPDF==1.23.6).

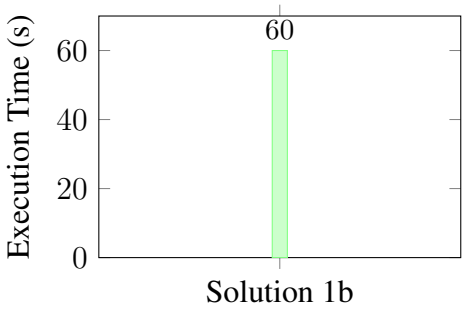

Figure 3: Execution Time for Solution 1b

3.4 Directory Structure

```
Adobe_Hackathon_Solution/
 Challenge_1b/
    Collection 1/
        challenge1b_input.json      % Input metadata and persona
        PDFs/                       % PDF documents
 Solution_1b/
    extractor.py                    % Subsection extractor logic
    processor.py                    % PDF text extractor
    ranker.py                       % Section ranker
    main.py                         % Entry point script
    outputs_generated/              % Output JSONs
    approach_explanation.md         % Methodology
```

```
requirements.txt                % Dependencies
README.md                       % Documentation
```

3.5 Setup and Execution

1. Clone the repository: `git clone https://github.com/<your-username>/Solution_1b.`
2. Create and activate a virtual environment: `python -m venv .venv`
3. Install dependencies: `pip install -r requirements.txt`
4. Ensure input PDFs and `challenge1b_input.json` are in `Challenge_1b/Collection 1/`
5. Run the script: `python main.py`

Outputs are saved in `Solution_1b/outputs_generated/Collection 1/`.

3.6 Sample Output


```
{
  "metadata": {
    "input_documents": ["file1.pdf", "file2.pdf"],
    "persona": "HR professional",
    "job_to_be_done": "Create and manage fillable forms for onboarding an
    "processing_timestamp": "2025-07-28T18:41:53.034114"
  },
  "extracted_sections": [
    {
      "document": "file1.pdf",
      "text": "Detected section text",
      "page_number": 4,
      "importance_rank": 1
    }
  ],
  "subsection_analysis": [
    {
      "document": "file1.pdf",
      "refined_text": "To understand 'Detected section text', ensure you
      "page_number": 4
    }
  ]
}
```


4 Why This Solution?

The design choices for `Solution_1b` balance performance, personalization, and scalability:

- **Personalized Content Delivery**: Semantic and keyword-based ranking ensures sections are highly relevant to the users persona and task, enhancing usability.

- **Lightweight and Efficient**: The rule-based approach with PyMuPDF ensures fast processing on CPU-only hardware, making the solution accessible and scalable.
- **Modularity and Extensibility**: The modular design allows easy adaptation for new personas, tasks, or scoring logic, ensuring future-proofing.
- **Clean Output**: Structured JSON outputs with minimal noise support integration with downstream systems, such as content management or recommendation engines.
- **Alignment with Theme**: By delivering context-aware content, the solution redefines how users interact with PDFs, aligning with the goal of rediscovering knowledge.

5 Final Thoughts

`Solution_1b` transforms PDF collections into personalized, context-aware knowledge systems, delivering relevant sections and summaries tailored to user needs. Its modular, efficient, and accurate design makes it a practical and impactful solution for the Adobe India Hackathon 2025. The visual aids and detailed documentation provided here aim to clearly convey the solutions value to judges and stakeholders.

6 Author

Developed by **Anuj Mishra** for the Adobe India Hackathon 2025.
Connect: LinkedIn | GitHub