Bank Statement Aggregation Following Cognitive Architecture

Proposal and AI System Explanation

Submitted by:

Name: Shimul Chakraborty

Matriculation number: 6351889

Name: Vamsi Teja Andhavarapu

Matriculation number: 6352484

Confidential Draft – For Academic Purposes Only

1. Problem Description

Very often, companies get financial records from clients that are not digital, mostly as scanned PDFs. There are big differences in the quality, style, and structure of these bank accounts, which makes entering and integrating data by hand a slow, error-prone, and time-consuming process. It's harder to quickly study and collect financial data when it isn't digitalized, which is important for consultant jobs like accounting, financial planning, and compliance reports. The goal is to make an AI system that can:

- Read bank statements from PDFs of different quality, such as scanned pictures and text-based papers.
- Getting the date, title, name, and transaction details from these papers.
- Putting the gathered data into an organized database so that it can be analyzed and put together in a better way.

This system will make it easier for business consulting firms to keep financial records by managing the digitization of bank bills. It will also significantly reduce mistakes and human work.

2. Challenges

- **Document Quality Variance:** Bank statements may be of varying quality, including poor scans, low resolution, noise, and handwritten annotations.
- **Layout Differences:** Different banks use unique layouts and formats, complicating template-based extraction.
- **Text Recognition:** Extracting accurate text from varying fonts, sizes, and orientations.
- **Data Validation:** Ensuring the extracted data is accurate and susceptible to OCR errors.
- **Security and Privacy:** Handling sensitive financial data securely.

3. Related Work

Several AI-powered solutions have been developed to address the challenges of extracting data from bank statements including research-oriented work such as:

- **Klippa DocHorizon:** Provides an Intelligent Document Processing platform that automates bank statement processing using OCR and AI technologies, enabling seamless data extraction and integration. [1]
- **Quantiphi's Virtual Assistant:** Automates the classification and extraction of content from scanned financial documents, eliminating manual dependency in business process outsourcing. [2]
- In academic research, Luo et al. (2021) proposed a method for automatic table detection and data extraction from financial PDF documents using a Faster R-CNN model with a Feature Pyramid Network (FPN), achieving superior detection performance on a customized dataset. [3]
- Similarly, Patel and Bhatt (2020) introduced an approach for extracting key information from scanned invoices by integrating visual and textual features through

a BiLSTM model, effectively automating data extraction from unstructured financial documents. [4]

These systems primarily focus on automating the extraction of data from bank statements using OCR and AI technologies, converting unstructured data into structured formats for easier analysis and integration.

4. Chosen Technology

We propose a **Hybrid Cognitive Architecture** that integrates symbolic and sub-symbolic approaches, closely aligned with the provided code structure:

- **Perception:** Using **pdfplumber** for text extraction from PDFs and **pytesseract** for OCR on scanned images.
- **Attention:** Utilizing **regular expressions (regex)** to filter and extract relevant transaction data.
- **Memory:** Implementing an **SQLite database** for structured storage of extracted transactions.
- **Reasoning:** Applying **rule-based logic (regex)** for data validation and pattern matching.
- **Prospection:** Simulating potential extraction errors and refining regex patterns dynamically.
- **Python:** For implementing the entire pipeline.

5. Justification of Technology

- **Hybrid Cognitive Architecture:** Combines perception through OCR and text extraction with symbolic reasoning for data validation.
- **PDF Parsing:** pdfplumber for extracting text from PDFs
- **Pre-trained OCR Models:** Utilizing **pytesseract** reduces development time and effectively handles scanned documents.
- **SQLite Database:** Offers lightweight, reliable storage for extracted transactions.
- **Regex-based Extraction:** Provides flexible handling of varied bank statement layouts.
- **Python:** A versatile programming language with extensive libraries for text processing, OCR, and database management.

6. System Workflow & Methodology

The proposed AI system adopts a **cognitive architecture** for efficient bank statement processing, structured around key functional layers:

Data Ingestion

- Users upload bank statements in PDF format, which may include: digitally generated PDFs (text-based) and scanned PDFs (image-based).
- Preprocessing for scanned documents, and preprocessing techniques such as binarization and noise reduction are applied to enhance OCR accuracy.

Perception Layer

- For machine-readable PDFs, the system uses **pdfplumber** to extract text directly.

- For scanned PDFs, **pytesseract** (Tesseract OCR) is used to extract text from images.
- Basic validation checks are performed to flag unreadable pages or corrupted files.

Attention Layer

- **Regex** is used to filter and extract transaction details (date, description, amount) from the raw text.
- Extracted data is mapped to a structured format (e.g., a table with columns for date, description, and amount).
- Anomalies (e.g., missing fields or inconsistent formats) are flagged for manual review.

Reasoning Layer

- Rule-based logic ensures data integrity and consistency (e.g., checking for duplicate transactions or invalid dates).
- Common OCR mistakes (e.g., misread characters) are corrected using predefined rules.
- Inconsistencies or errors that cannot be resolved automatically are flagged for manual review.
- Validated and cleaned transactions are prepared for storage in the database.

Memory Layer

- After validation and cleaning, the transaction data is stored in an **SQLite database**.
- The database supports efficient querying and retrieval of financial data for further analysis.

Output

- The system exports the structured data in user-friendly formats such as **CSV** or **Excel**.
- Integration with existing financial systems is also supported.

7. Flow Diagram

The following flow diagram illustrates the cognitive architecture of the proposed AI system:

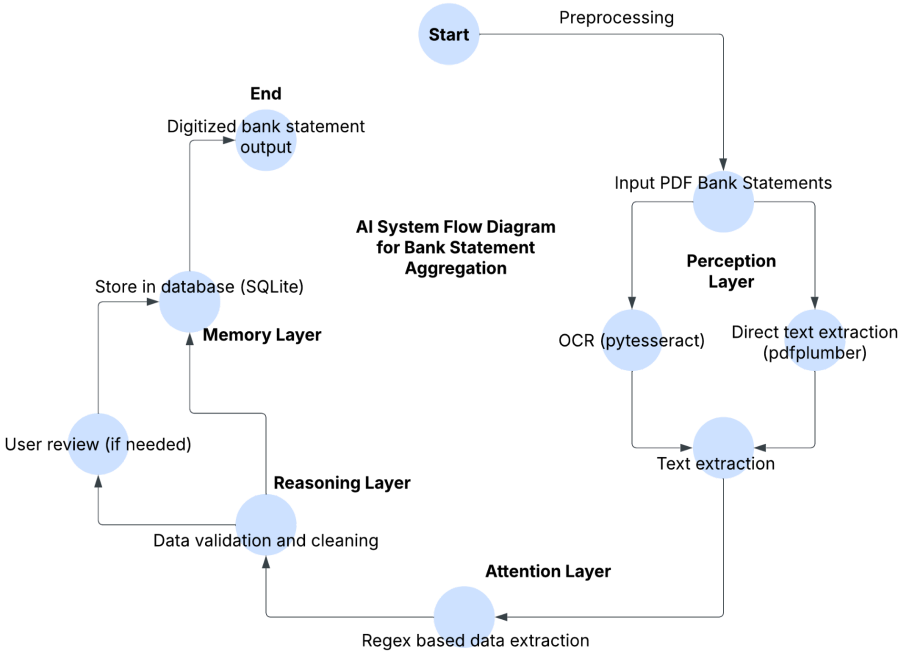

Fig: AI System Flow Diagram For Bank Statement Aggregation

8. Limitations

- **OCR Errors:** Low-quality scans may still result in incorrect extractions.
- **Regex Limitations:** Complex or unusual statement layouts may require manual regex updates.
- **Processing Time:** Large PDF files may slow down the extraction process.
- **Data Security:** Handling sensitive financial data requires strict privacy measures.
- **Scalability:** SQLite is suitable for small to medium datasets but may not scale for enterprise-level data.
- **Human Oversight:** Complex cases or anomalies still require manual review

9. Conclusion

In conclusion, the proposed AI system offers an efficient solution for automating bank statement aggregation using a hybrid cognitive architecture. By combining OCR technology, regular expressions, and rule-based reasoning, the system effectively extracts structured data from unstructured sources, streamlining financial data processing for corporate consulting companies. While challenges such as OCR accuracy and data privacy remain, the system provides a strong foundation for future enhancements.

10. References

[1] Klippa DocHorizon. (n.d.). *Intelligent Document Processing*. Retrieved from https://www.klippa.com

[2] Quantiphi. (n.d.). *Cognitive Document Processing*. Retrieved from https://quantiphi.com/case-studies/cognitive-document-processing

[3] Luo, J., Zhang, L., & Wang, H. (2021). *Automatic Table Detection and Data Extraction from Financial PDF Documents Using Faster R-CNN*. Retrieved from https://arxiv.org/abs/2102.10287

[4] Patel, R., & Bhatt, P. (2020). *Information Extraction from Scanned Invoices Using BiLSTM*. Retrieved from https://arxiv.org/abs/2009.05728