Automated Document Classification and Sorting Using Template Matching

Savidhan Ambhore[1], Huzefa Shaikh[2], Khalid Siddiqui[3], Vishakha Nayak[4]
Department of Electronics and Computer Science Shah and Anchor Kutchhi Engineering Colleg, Chembur, Mumbai 88.

Abstract—**Handling large quantities of documents manually often results in mistakes and reduced efficiency, particularly when documents appear alike but fall into different categories. This challenge can be addressed through template matching. Template matching helps automatically detect the document type and categorize it into appropriate databases. Technologies like Optical Character Recognition (OCR), Artificial Intelligence (AI), and Natural Language Processing (NLP) are employed to enhance document handling by improving accuracy, saving time, and streamlining storage. The system is built using Python, Tesseract OCR, Firebase (Supabase), and Node.js, providing a reliable and scalable document management solution.**

Index Terms—**Template Matching, Tesseract OCR, NLP, Sorting, Node js, Supabase,**

1. INTRODUCTION

In modern organizational processes, the capability to efficiently detect and classify large volumes of documents is critical for sustaining productivity and ensuring data integrity. With the growing occurrence of document-related fraud involving subtle changes to official templates, manual verification methods are no longer adequate, often resulting in time inefficiencies and potential mistakes. To tackle this issue, the template matching algorithm enables the development of an automated system that recognizes document types and directs them to the correct database.
By incorporating template matching techniques along with OCR technology [1], documents are swiftly identified, categorized, and stored in their designated repositories. This automation enhances the validation process, reduces reliance on manual checks, and significantly improves the overall operational flow within organizations [2].

The method starts by extracting structural and textual characteristics through OCR, followed by preprocessing operations that refine and normalize the obtained data. Important identifiers such as specific logos, standard phrases, header formatting, and document structure are examined to highlight unique traits. Using template matching, the system compares these extracted components against a collection of pre-defined document templates to locate the best match—even when minor discrepancies exist due to formatting shifts, noise interference, or deliberate alterations.

Once recognized, the system automatically sorts the document into its respective database, eliminating manual classification. This not only cuts down processing time but also reduces the likelihood of errors like misplacement or incorrect sorting, thereby boosting overall efficiency. With backend solutions like Supabase, the system ensures secure, scalable storage and seamless access to categorized documents as required.

Overall, the template matching algorithm offers a powerful base for evolving traditional document handling methods. It enhances both internal processes and the organization's security framework by promptly flagging irregularities in document structures. Additionally, the modular design allows for easy adaptation—new document types can be incorporated by simply adding new templates, without the need for major structural adjustments [1][2][3].

2. METHODOLOGY

The workflow is divided into several stages:
1. Document Upload**:** The initial phase begins with submitting the document into the system. Users can upload scanned images, photographs, or

digital copies obtained from different devices [4]. The document must be in an accepted file format, such as JPEG or PNG.

This stage serves as the entry point for the system, making sure documents are prepared for automated processing. Validation procedures may be applied to ensure the file's size, format, and resolution meet the essential criteria for precise OCR and further analysis.

2. OCR Extraction: After the document is uploaded, it is processed using Tesseract OCR. This OCR technology transforms the visual content of the image into machine-readable text. Tesseract thoroughly scans the document to detect and extract all visible characters, digits, symbols, and words. By the end of this phase, the system generates a raw text output that captures the entire textual content present in the document [5].
3. Preprocessing: The raw text obtained from OCR is refined through cleaning techniques, and key elements—such as titles, relevant keywords, and logos—are detected and highlighted for further analysis.
4. Template Matching: Following preprocessing, the system carries out template matching to determine the document category. The extracted attributes—such as keywords, structural layout, and logos—are compared against a predefined collection of known document templates. Each template includes standard patterns and characteristics that represent specific document types like Aadhaar or PAN Card. Matching algorithms are then used to compute a similarity score between the uploaded document and the templates in the database.
5. Document Classification: Based on the similarity score from the template matching process, the system determines and labels the document type accordingly.
6. Database Sorting: Once the document is classified, it is routed to its appropriate storage location within the system. The identified documents are stored in designated collections or buckets within databases like Supabase [6], with each one aligned to a specific document type. Additional metadata—such as document category, upload timestamp, user ID (if available), and confidence score—is also saved alongside the file to facilitate organized management.

3. FLOWCHART

The workflow of the algorithm is as shown on figure no.1 and follows following step

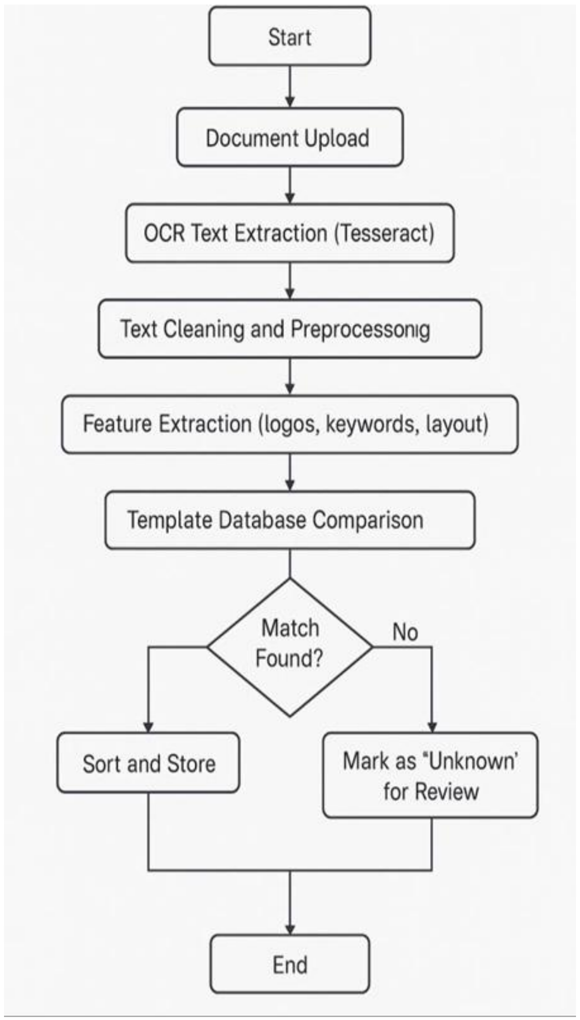

Start
Document Upload
OCR Text Extraction (Tesseract)
Text Cleaning and Preprocessonig
Feature Extraction (logos, keywords, layout)
Template Database Comparison
Match Found?
No
Sort and Store
Mark as 'Unknown' for Review
End

Fig. 1 Flowchart of the algorithm

STEP 1: Start

The process is initiated when a user submits a document to the system. This document may be a scanned image or photo and can include different types of ID proofs, certificates, or official government forms.

STEP 2: Document Upload

At this stage, the system accepts the uploaded document and carries out basic validation to confirm that the file format—such as JPG or PNG—is supported. Once verified, the document is readied for the next phase of processing.

STEP 3: OCR Text Extraction (Tesseract)
The document is processed using **Tesseract OCR**, which extracts the textual content from the image. This step transforms the visual data into machine-readable text, enabling subsequent analysis.
STEP 4: Text Cleaning and Preprocessing
The raw text obtained through OCR may include noise, irrelevant characters, or inconsistent formatting. To ensure clarity and accuracy, the text undergoes a cleaning process that involves removing special symbols.

The cleaned text is then tokenized into individual words or key phrases to simplify the extraction of important features.
STEP 5 Feature Extraction (Keywords, Layout)
From the refined text and overall document layout, key features are extracted for analysis: Significant keywords such as *"Government of India," "Aadhaar,"* and *"PAN Card"* are recognized. If necessary, image processing techniques can be used to detect logos and header formatting. Additionally, the spatial positioning of critical fields—such as ID numbers, names, and dates—is recorded for structural understanding [8].

STEP 6: Template Database Comparison
The identified features are matched against a predefined set of document templates stored in the system. Each template corresponds to a specific, recognized document type. The system computes a similarity score by comparing the extracted features with each template. If the score exceeds a defined confidence threshold (e.g., 80% or higher), the document is classified as that particular type.

STEP 7: Match Found (Decision Node) If a match is found:
The document is labelled with the identified type (e.g., "PAN card").
It moves to the "Sort and Store" step. If no satisfactory match is found:
The document is marked as "Unknown".
It is sent for manual review or stored separately for further investigation.
STEP 8: Sort and Store
For successfully identified documents:
The system categorizes and stores the document in the appropriate database using Supabase. For instance, Aadhaar cards are placed into the Aadhaar-specific database, while passports are directed to the Passport database.

STEP 9: Mark as "Unknown" for Review
For documents that do not match any existing template: Documents that cannot be confidently classified are marked as "Unknown". They are stored in a designated database or folder specifically reserved for manual review by an administrator.
This approach guarantees that every document is retained, even if it requires further inspection before classification.
STEP 10: End
The workflow is completed once the document is either stored in its designated database or marked for manual review. The system then resets, ready to handle the next document upload.

4. ALGORITHM

To implement the algorithm, the input used is an image of an Aadhaar card approved by the Government of India. Initially, the image is uploaded into the system, which accepts the scanned copy or photograph, as illustrated in Figure 2.

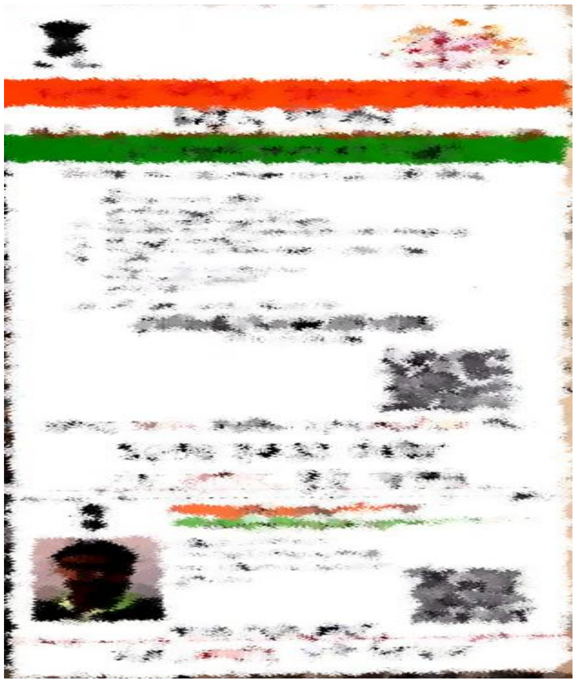


Fig. 2 Adhaar card sample

Tesseract OCR is employed to extract textual data from the uploaded document image. To ensure accuracy and consistency, the text undergoes a cleaning process that involves:
Removing special characters and noise. Converting all letters to lowercase.
The text into meaningful words or phrases.
Feature Extraction: Important attributes such as document titles, organization names, and specific keywords (e.g., *"Aadhaar"*, *"PAN Card"*) are identified. If required, structural elements like text alignment, heading placement, and logo presence are also detected and captured for further analysis.
Template Matching: The system compares the extracted features with a predefined set of document templates stored in a database. A similarity score is calculated to identify the closest matching template. If the confidence level exceeds a specified threshold (e.g., 80%), the document type is confirmed. This process is illustrated in Figure 3.

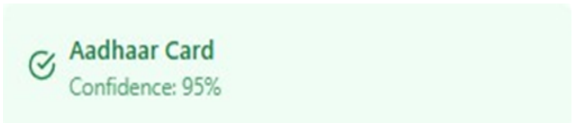
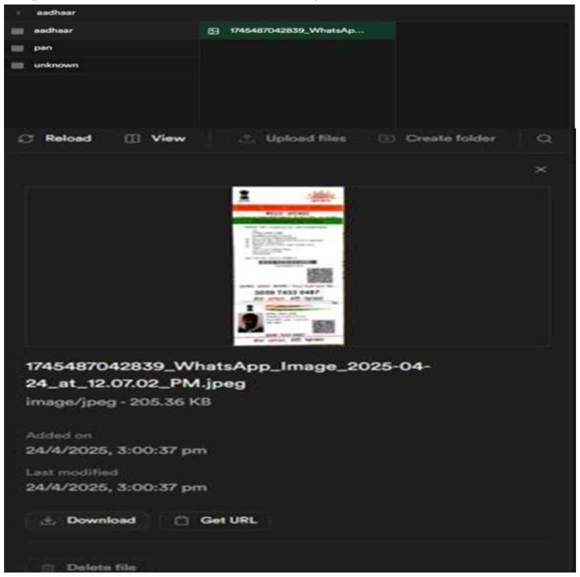

Aadhaar Card
Confidence: 95%

Fig. 3 Document identification

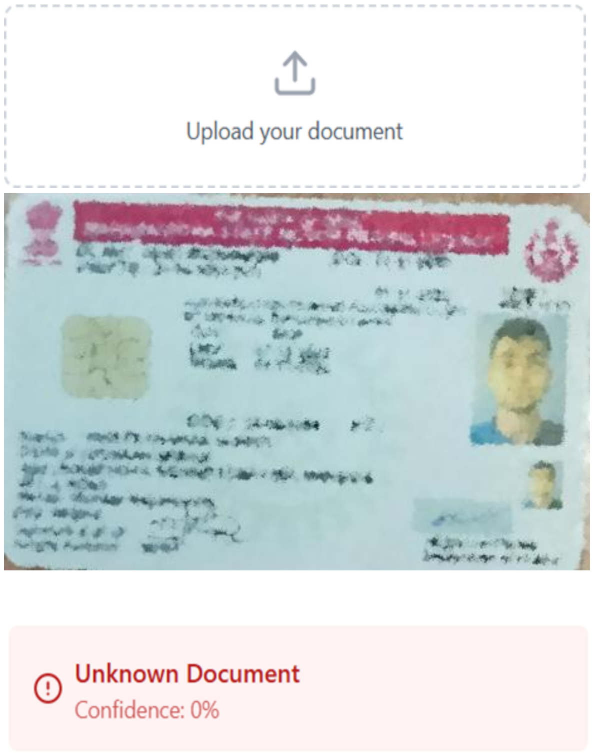
Once the document type is successfully identified, it is labeled accordingly—for example, as an Aadhaar Card. The system then automatically transfers the classified document into its designated database. In this case, Aadhaar documents are stored in the Aadhaar Database using database APIs such as Supabase, as illustrated in Figure 4.



1745487042839_WhatsApp_Image_2025-04-24_at_12.07.02_PM.jpeg
image/jpeg - 205.36 KB
Added on
24/4/2025, 3:00:37 pm
Last modified
24/4/2025, 3:00:37 pm

Fig. 4 sorting in Database

In cases where no satisfactory match is found, the document is marked as an "Unknown Document" and stored in a separate location for manual verification, as shown in Figure



Unknown Document
Confidence: 0%

Fig. 5 sorting in Database

5. CONCLUSION

We successfully developed a system capable of automatically identifying various types of documents and sorting them into their respective databases. By integrating OCR technology with a template matching algorithm, the system streamlines document processing, enhances classification accuracy, and significantly reduces manual effort and time.
The results demonstrate that an uploaded Aadhaar card is correctly identified, matched with its corresponding template, and sorted into the appropriate database. The same process has been validated for other documents approved by the Government of India.Looking ahead, we aim to strengthen the system by incorporating **deep** learning models to enable more advanced feature extraction and robust matching. This will further improve the system's ability to detect heavily altered or newly introduced document formats.

REFERENCES

[1] Neumann, L. and Matas, J. (2012). Real-Time Scene Text Localization and Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). DOI: 10.1109/CVPR.2012.6248013 – Focuses on OCR in real-world environments using advanced localization and recognition.

[2] Smith, R. (2013). History of the Tesseract OCR Engine: What Worked and What Didn'tmIn Proc. SPIE 8658, Document Recognition and Retrieval XX. DOI: 10.1117/12.2007413 – A follow-up to your [1], offering additional insights into Tesseract development.

[3] Brunelli, R. (2009). Template Matching Techniques in Computer Vision: Theory and Practice. Wiley Publishing. – A foundational book covering theory, implementations, and optimizations in template matching.

[4] Kim, J., Park, C., and Lee, J. (2014). Template Matching for Object Detection Using Edge Orientation Histograms. Sensors, vol. 14, no. 2, pp. 2818–2833. DOI: 10.3390/s140202818 – Uses edge features for robust matching under noise and transformations.

[5] Chen, Y., Zhu, Z., and Zhang, Z. (2015). Document Image Classification with Intra-Class Distance Constraint. In Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR). DOI: 10.1109/ICDAR.2015.7333903 – Focuses on learning-based classification of document types.

[6] Harley, A. W., Ufkes, A., and Derpanis, K. G. (2015). Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval. In ICDAR 2015. DOI: 10.1109/ICDAR.2015.7333885 – Explores how CNNs perform on document classification tasks.

[7] Simard, P. Y., LeCun, Y., and Denker, J. S. (2003). Efficient Pattern Recognition Using a New Transformation Distance. In Advances in Neural Information Processing Systems (NIPS). – Early paper on deep learning for pattern recognition, relevant for future enhancements of your project.

[8] Han, X., Leung, T., Jia, Y., Sukthankar, R., and Berg, A. C. (2015). MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching. In CVPR 2015. DOI: 10.1109/CVPR.2015.7298947 – Deep learning method for template/pattern matching tasks.

[9] Han, X., Leung, T., Jia, Y., Sukthankar, R., & Berg, A. C. (2015). "MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching." IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [DOI 10.1109/CVPR.2015.7298947] – Introduces a deep network for template matching tasks, learning both features and similarity metrics.

[10] Rocco, I., Arandjelović, R., & Sivic, J. (2017). "Convolutional Neural Network Architecture for Geometric Matching." CVPR 2017. [DOI: 10.1109/CVPR.2017.542] –Deep learning-based geometric alignment, helpful for template matching with deformations.