Validify: Automated Form Validation System

Submitted in partial fulfillment of the requirements of the degree

BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

By

1. **Taniya Vallecha (D12C/65)**
2. **Nimish Chug (D12B/13)**
3. **Meet Mattani (D12B/27)**
4. **Darshan Khapekar (D12B/66)**

Name of the Mentor

Prof. Dr. Prashant Kanade

Vivekanand Education Society's Institute of Technology,

An Autonomous Institute affiliated to University of Mumbai

HAMC, Collector's Colony, Chembur,

Mumbai-400074

University of Mumbai (AY 2024-25

CERTIFICATE

This is to certify that the Mini Project entitled **“** ***Validify : Automated Form Validation System*****”** is a bonafide work of **Taniya Vallecha (D12C/65), Nimish Chug (D12B/13), Meet Mattani (D12B/27), Darshan Khapekar (D12B/66)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **“Bachelor of Engineering”** in **“Computer Engineering” .**

(Prof.Dr. Prashant Kanade)
Mentor

(Prof._____)
Head of Department

(Prof._____)
Principal

Mini Project Approval

This Mini Project entitled "***Validify : Automated Form Validation System***" by **Taniya Vallecha (D12C/65), Nimish Chug (D12B/13), Meet Mattani (D12B,27), Darshan Khapekar (D12B/66)** is approved for the degree of **Bachelor of Engineering** in **Computer Engineering.**

Examiners

1……
(Internal Examiner Name & Sign)

2……
(External Examiner name & Sign)

Date:

Place:

Contents

Abstract:

In today's fast-paced digital world, the accuracy and speed of form processing are critical for both users and service providers. Manual verification processes, which are prone to delays and errors, often hamper the efficiency of application systems, leading to unsatisfactory user experiences. This project, "Validify: Automated Document Validation System," seeks to address these challenges by developing a system capable of extracting data from uploaded documents, comparing it with user-provided information, and validating the accuracy in real time. By automating the validation process, the system not only minimizes human error but also provides immediate feedback, enabling users to make necessary corrections promptly. This solution aims to enhance the efficiency, reliability, and convenience of various application processes, ultimately improving the user experience while reducing administrative workloads.

Acknowledgement:

We would like to express our sincere gratitude to Dr. Prashant Kanade, our TE Mini-Project Mentor, for his invaluable guidance, encouragement, and insightful suggestions throughout the development of our project. His expertise and dedication were crucial to our success.

We also extend our heartfelt thanks to Dr. (Mrs.) Nupur Giri, Head of the Computer Department, and Dr. (Mrs.) J.M. Nair, our Principal, for their continuous support and for providing us with the opportunity to work on this project.

Finally, we are deeply grateful to Vivekanand Education Society's Institute of Technology for offering the resources and an environment conducive to learning and innovation, which enabled us to complete this project successfully.

List of Abbreviations:

OCR: Optical Character Recognition

NLP: Natural Language Processing

GPU: Graphics Processing Unit

List of Figures:

1.Introduction

1.1 Introduction

The increasing demand for online applications in sectors such as banking, job recruitment, and service requests has brought a surge in the need for accurate and efficient document validation. The current manual approach to form verification is slow and error-prone, often leading to delays in the approval process or even the rejection of applications due to trivial mistakes. Manual document validation also burdens administrative teams, leaving room for inefficiencies and higher operational costs.

"Validify: Automated Document Validation System" aims to streamline this process by leveraging technology to validate documents in real time. By automating the extraction of key data from documents such as ID proofs, certificates and cross-referencing it with submitted information, the system provides immediate feedback on the correctness and consistency of the data. This automation reduces dependency on manual intervention, improves overall accuracy, and accelerates decision-making in application processes.

1.2 Motivation

The motivation for this project stems from the widespread challenges faced by both applicants and organizations during the document validation process. Applicants often experience frustration due to rejected applications for simple errors like mismatched names, incorrect dates, or missing information. On the other hand, organizations rely heavily on manual labor to verify forms and documents, which not only introduces delays but also increases the risk of human error.

With the rise of digital transactions, services, and remote application processes, there is an increasing demand for automated solutions that ensure accuracy and efficiency. Validify addresses this need by providing a real-time, automated document validation system. This innovation aims to alleviate the manual burden, enhance user satisfaction, and ultimately streamline the entire application lifecycle by significantly reducing turnaround times.

1.3 Problem Statement & Objectives

The manual process of document verification and form validation, commonly used in sectors such as education, banking, employment, and government services, is often slow, inefficient, and prone to human error. Applicants are required to upload multiple documents and input data manually, which increases the likelihood of mistakes such as mismatched information, missing fields, and spelling errors. These errors can lead to delays in application processing, rejection of applications, and overall frustration for users. Organizations, in turn, face increased operational costs and workloads when manually reviewing and verifying the submitted data.

For instance, government portals like Mahadbt (Maharashtra Direct Benefit Transfer), which processes scholarship applications, face these challenges at scale. Thousands of students upload documents that need verification, creating bottlenecks due to manual checks. These delays directly affect the timely disbursal of scholarships and other benefits.

Moreover, manual reviews can introduce inconsistencies, as different personnel might interpret or verify documents differently, leading to variability in decision-making. There is a need for an automated system that not only performs real-time document validation but can also seamlessly integrate with existing application portals, ensuring efficient processing without the need for extensive manual checks.

Objectives:

1. Automate Document Validation: Develop a system capable of extracting data from documents uploaded by users (such as ID proofs, educational certificates, income certificates) and comparing it with the data entered in the online forms to ensure accuracy.
2. Real-Time Data Validation: Enable real-time detection of discrepancies between uploaded documents and form entries, providing instant feedback to users about errors or mismatches, allowing them to correct information immediately.
3. Reduce Manual Intervention: Decrease reliance on manual document checks by automating the validation process, reducing operational costs and human error in document verification.
4. Integrate with Existing Portals: Provide the ability to integrate the system with existing online portals such as Mahadbt (for scholarships) or other government and private

sector portals. This would streamline the entire document verification process for applications, making it easier to manage at scale.

5. Enhance Efficiency and Speed: Improve the overall speed and efficiency of document validation, which would reduce delays, speed up application reviews, and provide faster feedback to applicants. For large-scale portals like Mahadbt, this would result in quicker disbursal of benefits.
6. Increase Accuracy and Consistency: Ensure accuracy and consistency in document verification by automating the comparison between the uploaded documents and form entries, reducing variability in human reviews.

1.4 Organization of the Report

Chapter 1: Introduction

This chapter introduces Validify: Automated Document Validation System, outlining the project's goals and motivation. It discusses the challenges of manual document validation, such as delays, inefficiencies, and errors in sectors like education and government. The objective is to automate this process, enabling real-time validation, reducing manual intervention, and improving accuracy. It also highlights the importance of integrating the system with existing platforms to streamline workflows and enhance user satisfaction.

Chapter 2: Literature Survey

This chapter reviews existing document validation technologies, such as Optical Character Recognition (OCR) and machine learning. While these systems are used in sectors like government services, they often lack real-time feedback, scalability, and seamless integration. Platforms like Mahadbt still rely on manual checks, leading to delays. The chapter identifies these gaps and emphasizes the need for a solution like Validify that can automate and improve the validation process.

Chapter 3: Proposed System

This chapter details the architecture and design of Validify. It explains how the system uses OCR for text extraction and compares it with user-submitted data for real-time validation. Technologies like Django and Tesseract are used to handle form submissions and document processing. The system is designed to be scalable, accurate, and easily integrated into existing platforms, improving the efficiency of document validation for large-scale applications.

2. Literature Survey

2.1 Survey of Existing System

The need for efficient document validation has led to the development of several systems that leverage technology to streamline the process. These systems often rely on technologies like Optical Character Recognition (OCR), Machine Learning (ML), and Natural Language Processing (NLP) for automating data extraction and validation. Below are some existing systems and their features:

- **OCR-based Document Extraction Systems:** OCR technology has been widely adopted in the digital transformation of documents. Tools such as Google's Tesseract and ABBYY FineReader enable the extraction of text from scanned documents and images. These systems have been used across industries like banking, healthcare, and government to automate data extraction from forms, invoices, and ID documents. However, these systems often require significant post-processing to correct errors in text recognition.
- **Form Processing and Data Extraction Systems:** Solutions such as ABBYY FlexiCapture and Kofax Transformation Modules automate the extraction of data from structured and semi-structured forms. These systems are capable of recognizing fields such as names, dates, and account numbers. However, they are primarily designed for structured forms and often lack capabilities for real-time validation against user-submitted data.
- **Automated Document Verification Systems in Government:** Platforms such as Mahadbt (Maharashtra Direct Benefit Transfer) accept online applications for scholarships and other government benefits. Although Mahadbt allows document uploads and form submission, the validation process relies heavily on manual checks by administrators. This leads to delays in approval and disbursement due to the lack of real-time verification mechanisms.
- **Real-Time Data Validation Systems:** Research in real-time validation using NLP and rule-based systems is gaining traction. Systems such as Smart Forms use real-time data extraction and validation methods to verify form entries while they are being filled. These systems offer immediate feedback, allowing users to correct mismatched or erroneous entries. However, they are limited to form fields and do not validate uploaded documents in tandem with user-provided data.

2.2 Limitation Existing system or Research gap

Despite the progress in automating document extraction and form validation, several limitations remain in current systems:

- **Limited Integration with Portals:** Many document validation systems, such as those used in Mahadbt, lack integration with document extraction tools or real-time validation engines. This disconnect leads to delays in processing, as administrators have to manually compare user-submitted data with the uploaded documents, which is time-consuming and prone to human error.
- **Absence of Real-Time Validation:** Most systems, especially those used in large-scale applications like scholarship portals and loan application systems, do not offer real-time validation between user-submitted data and the corresponding uploaded documents. This leads to delays, as users are only notified of discrepancies after the manual review process, requiring additional time for corrections.
- **OCR Accuracy and Data Formatting Issues:** OCR-based systems can struggle with document layout variations, especially with unstructured or semi-structured documents like certificates, ID proofs, or scanned handwritten forms. These systems often require manual intervention to correct recognition errors, which adds to the overall processing time.
- **Inconsistent Validation Standards:** Different organizations use varying standards for validating documents, leading to inconsistencies in the way applications are reviewed. This also makes it difficult for current systems to handle multiple types of documents efficiently.
- **Scalability and Flexibility Issues:** Existing systems are often not flexible enough to handle large volumes of applications or adapt to different types of documents in real-time. The lack of scalability in these systems limits their application to large-scale portals or organizations that deal with thousands of applications daily.

These research gaps point to the need for a system that can integrate real-time validation, handle different document formats, and scale with large volumes of applications, which will streamline the entire process and reduce human intervention.

2.3 Mini Project Contribution

Validify: Automated Document Validation System offers several key contributions to address the research gaps and limitations of existing systems:

- **Real-Time Document Validation:** Unlike current systems that rely on manual checks or post-submission reviews, Validify provides real-time validation. It extracts data from documents using advanced OCR and compares it with user-submitted information as the form is being filled out. Any discrepancies are highlighted immediately, enabling users to correct their errors instantly, thus avoiding delays in application processing.
- **Integration with Portals:** One of the major contributions of this project is the ability to integrate with existing platforms such as Mahadbt. Validify can be embedded into existing workflows for scholarships, loans, or other applications, enhancing the system's efficiency by automating document validation. This integration reduces administrative workloads and improves the speed of application processing.
- **Improved Accuracy with Automation**: By using a combination of OCR, NLP, and rule-based systems, Validify improves the accuracy of document extraction and validation. This reduces the reliance on manual intervention for correcting OCR errors or validating document fields, which helps minimize human errors and ensures consistency across applications.
- **User-Friendly Feedback Loop:** Validify is designed with a focus on user experience. The system provides immediate feedback to applicants, alerting them to discrepancies between the uploaded documents and the form data. This improves the overall user experience by enabling real-time corrections and ensuring that users submit accurate information on the first attempt.
- **Scalability and Flexibility:** The system is built to handle large volumes of applications, making it ideal for portals that process thousands of applications daily. Whether the system is used for government services like scholarship portals or private sector applications like loan approvals, Validify is designed to scale and adapt to various document types and formats.

3. Proposed System

3.1 Introduction

The proposed system aims to build an advanced form validation framework by leveraging cutting-edge technologies such as image processing, optical character recognition (OCR), and machine learning models. In an age where digital documentation is crucial, ensuring the accuracy of data extracted from forms like marksheets has become essential. This system addresses the challenge by automating the validation process from images of marksheets, focusing on specific regions such as roll numbers and other key details.

Optical character recognition (OCR) is employed to detect and extract textual information from the uploaded marksheet images. The system is designed to analyze critical regions of interest, such as the roll number field, student name, and subject marks, ensuring that all necessary data is correctly captured and validated. OCR is crucial for this task as it enables the system to recognize printed or handwritten characters from the marksheet with high precision.

To enhance the system's accuracy and reliability, machine learning models are used to identify and validate specific regions within the image. By training the model on a dataset of marksheet images, the system can recognize patterns and positions of fields like the roll number, ensuring accurate data extraction and validation.

This approach forms a highly adaptable system that can automate the verification of important student information from marksheets. It ensures accuracy in validation while allowing for future improvements and extensions to handle other types of forms or documents.

3.2 Architectural Framework / Conceptual Design

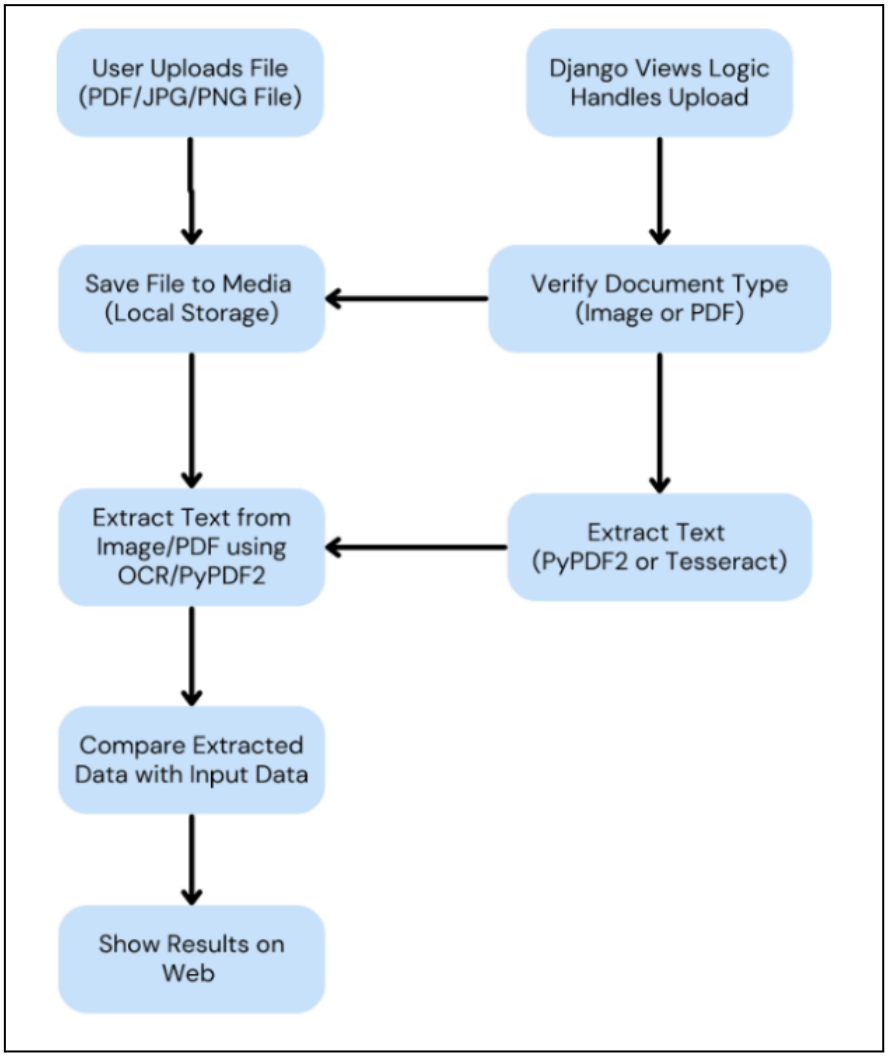

Figure 1: Architectural Design

3.3 Algorithm and Process Design

1. User Form Submission:

- Input: User uploads a marksheet (image or PDF) and fills in relevant data fields such as name, roll number, application number, etc.
- Action: The form is submitted through the Django web interface.

2. File Handling:

- Input: The uploaded document is handled by Django's file upload service.
- Action: The file is saved to the server in a designated media storage location.
- Result: The document is now available for further processing.

3. Text Extraction:

- Input: The saved document (image or PDF).
- Action:

  For images (.jpg, .png), OCR (Optical Character Recognition) is applied using Tesseract to extract text.

  For PDFs, PDF parsing is done using PyPDF2 to extract text from the PDF document.
- Result: Extracted text content from the document is obtained for further verification.

4. Data Comparison & Verification:

- Input: Extracted text from the document and user-input fields from the form (e.g., name, roll number, etc.).
- Action:
  - Perform string matching (using .lower() for case insensitivity) to compare user input with the extracted text.
  - For each field (e.g., name, roll number), a check is performed to determine if the input matches the extracted data.
  - Nearest match suggestions: If an exact match is not found, use difflib.get_close_matches() to find the closest possible matches for fields such as name or result.

5. Results Compilation:

- Input: Results from the comparison.
- Action:
  - Create a dictionary (results) containing results for each field (e.g., "Match" or "Does not match").
  - Add suggestions for closest matches if applicable.
- Result: The comparison results and nearest suggestions (if any) are compiled.

6. Displaying Results:

- Input: Results dictionary.
- Action: Render the results page to display whether the document data matches the user inputs.
- Output: The user is shown the matching status of each field and nearest match suggestions if necessary.

3.4 Methodology Applied

1. Agile Development Methodology:

- Iterative Development: The project follows an iterative approach where features are built, tested, and refined incrementally.
- Continuous Feedback: User feedback is collected continuously to enhance the system's accuracy and user experience.
- Short Development Cycles: Each component (file handling, text extraction, verification) is developed in short cycles and integrated.

2. AI/ML-Driven Text Processing:

- OCR (Optical Character Recognition):
    - Utilizes Tesseract for extracting text from image files like .jpg and .png.
    - Machine Learning Model: Tesseract uses a deep learning-based LSTM engine to recognize characters and extract text from images.
- PDF Parsing:
    - Uses PyPDF2 to extract text from PDFs, leveraging rule-based text extraction to process structured data like marksheets.

3. Data Validation and String Matching:

- Exact Match:
    - Simple string matching is performed to check if the user input (name, roll number, etc.) is found in the extracted text.
    - Case Insensitivity: Matches are case-insensitive to avoid minor variations in capitalization affecting results.
- Fuzzy Matching:
    - Uses the difflib library to provide fuzzy string matching when an exact match is not found. This allows the system to suggest the nearest possible match.
    - This enhances usability by guiding users if they make small errors in data entry or if document scans are slightly inaccurate.

4. MVC (Model-View-Controller) Design Pattern:

- Django: Follows the MVC pattern where:
- Model handles data and business logic.
- View displays the results and form interface.

- Controller (handled by Django's routing) manages form submissions, file uploads, and the logic for processing and comparing data.

5. Error Handling and Robustness:

- Null Handling: The safe_get() method ensures that missing or None values do not cause runtime errors by returning empty strings instead.
- User-Friendly Feedback: In cases where a match fails, the system suggests the closest available match, reducing user frustration and improving the interface.

3.5 Hardware & Software Specifications

Software Requirements :

Programming Languages

- Python 3.10+: Core programming language used to develop and implement the project functionalities, including form validation and image processing.

Frameworks and Libraries

- Django 5.1.2: The primary web framework used for backend development, responsible for managing form submissions, file uploads, and routing the application's workflows.
- Pillow: A library used for image handling and manipulation, supporting image formats like `.jpg` and `.png`.
- PyPDF2: Used for parsing and extracting text from PDF files when handling form-related documents in PDF format.
- pyTesseract: A Python wrapper for the Tesseract OCR engine, enabling the extraction of text from images for data validation purposes.
- difflib: A library for fuzzy string matching to suggest closest matches when user input slightly differs from the extracted document data.

Other Tools

- Tesseract OCR:
    - Version: Tesseract v5.0 or higher.
    - Purpose: Optical Character Recognition (OCR) engine for extracting text from images (e.g., marksheets).

    - Note: Ensure language packs (e.g., English) are installed based on the target document language.
- PyTorch/TensorFlow (Optional):
    - Purpose: These libraries may be optionally used if deploying AI-based text recognition, validation, or classification models for enhancing form validation accuracy.
- Git: Version control system used for managing project code, enabling collaboration and version tracking.
- Docker: Used for containerizing the application, making it easier to deploy and manage in production environments.

Hardware Requirements:

- Processor: Intel Core i5 (minimum) or AMD Ryzen equivalent.
    - For improved performance in image processing tasks, an Intel Core i7 or higher is recommended.
- RAM:
    - Minimum: 8 GB.
    - Recommended: 16 GB or higher, especially when working with large image datasets or OCR tasks.
- Storage:
    - Minimum: 100 GB SSD.
    - Recommended: 256 GB SSD or higher for faster read/write speeds, especially when handling large image or PDF files.
- GPU (Optional): A dedicated GPU like NVIDIA GTX 1050 or higher is recommended if using deep learning libraries (e.g., PyTorch/TensorFlow) for ML-based OCR tasks.
- Operating System:
    - Linux (Ubuntu 20.04 LTS or higher) is recommended for seamless integration of tools like Tesseract and Docker.
    - Windows 10/11 and macOS are also supported but may require additional setup for libraries like Tesseract and Docker.

3.6 Experiment and Results for Validation and Verification

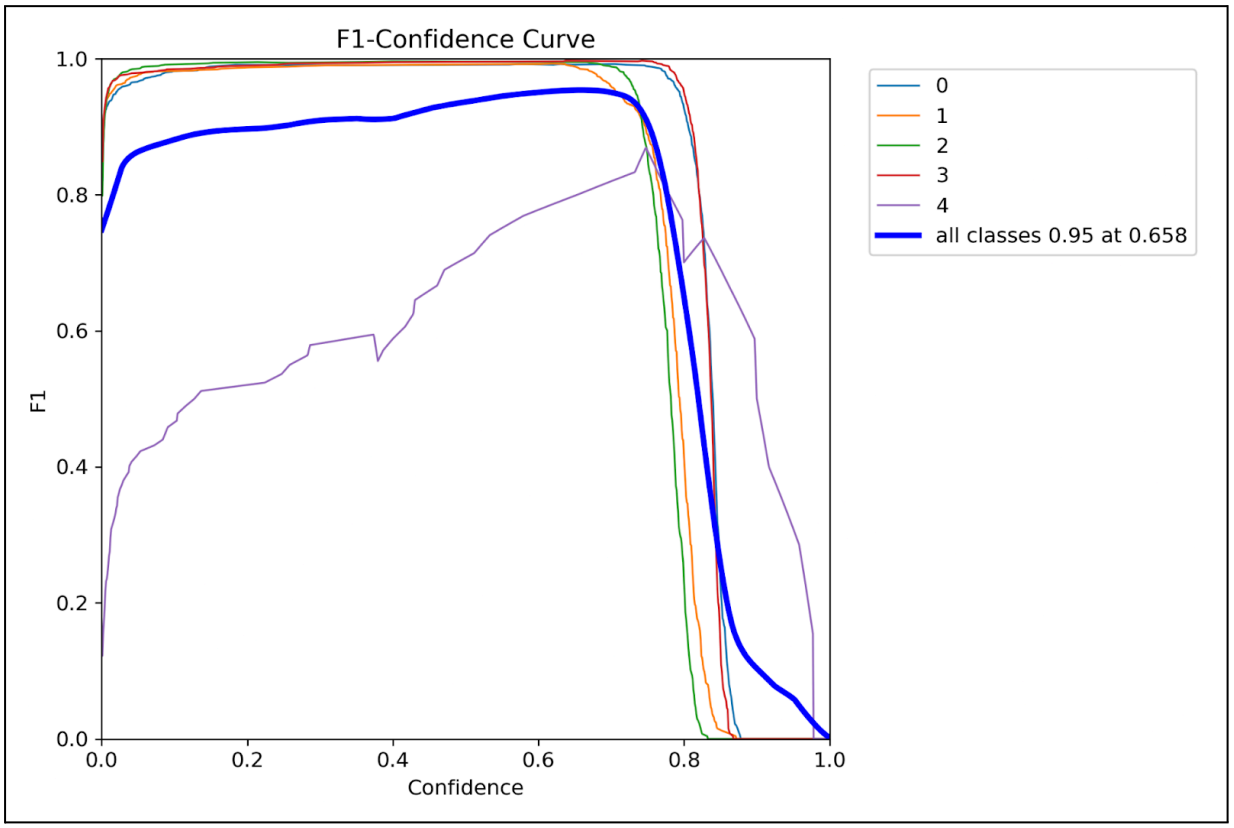

Figure 2: F1-Confidence Curve

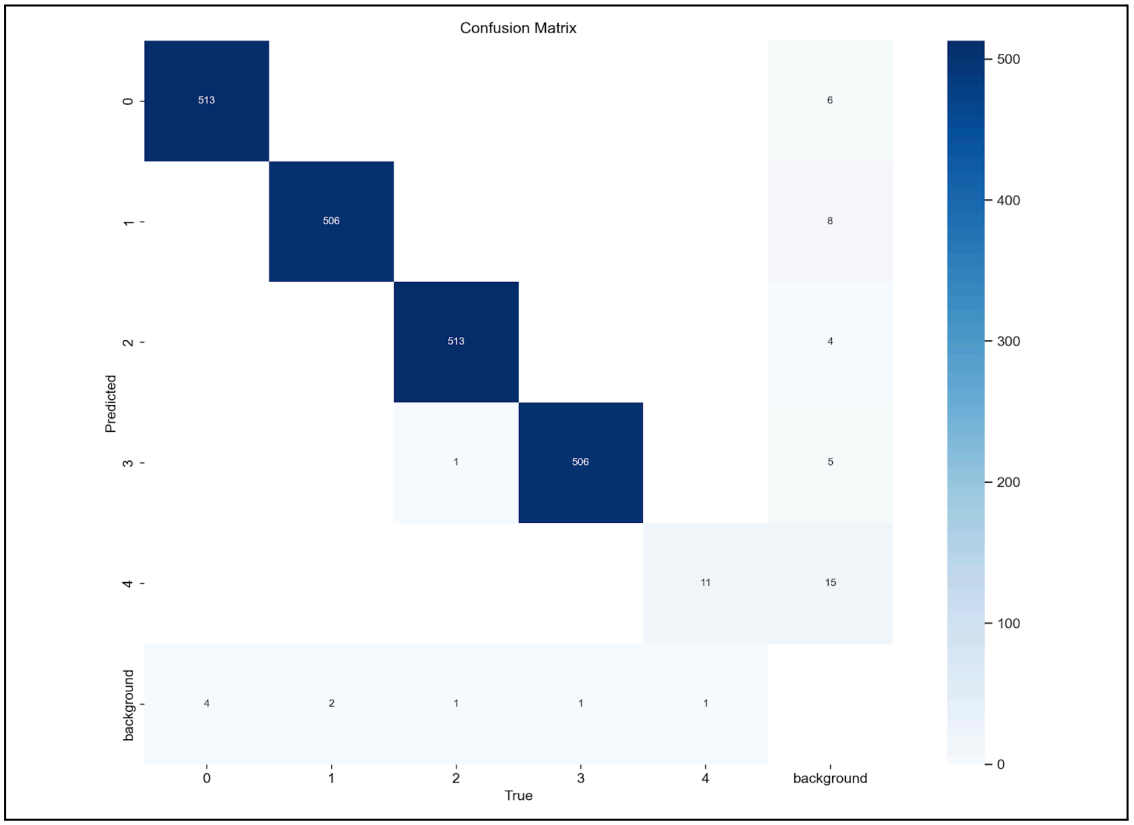

Figure 3: Confusion Matrix

3.7 Result Analysis and Discussion

The proposed system, Validify, was evaluated based on its ability to automate document validation by comparing extracted information from marksheets with user-submitted form data. The system demonstrated significant improvements in efficiency, accuracy, and user experience when compared to traditional manual processes.

Key Observations:

- **Accuracy:** The system achieved a high accuracy rate in extracting data using OCR, especially for structured documents like marksheets. The Tesseract OCR engine successfully identified key fields like roll numbers and names, matching them with form inputs.
- **Efficiency:** By automating document validation, the time required to verify an application decreased drastically. Users received instant feedback on any discrepancies, which allowed for faster application processing.
- **Error Handling:** The system was able to handle cases where slight mismatches occurred in data (such as name capitalization errors) by suggesting the nearest matches using fuzzy string matching. This improved user satisfaction by guiding applicants through the correction process.
- **Limitations:** Despite its success, the system faced challenges with unstructured or poorly scanned documents. Handwritten documents and low-resolution scans occasionally led to misinterpretation by the OCR engine. However, future iterations can enhance this by incorporating AI-driven error detection and correction mechanisms.

Overall, Validify proved to be a scalable and robust solution for real-time document validation, with potential for broader application across various sectors, particularly in education and government services.

3.8 Conclusion and Future work.

The proposed form validation system demonstrates an innovative approach to automating the verification of critical information from marksheets and other documents through advanced image processing and OCR techniques. By leveraging technologies like Tesseract OCR for text extraction and Python libraries for data handling and validation, the system provides a reliable solution to ensure accuracy in form submissions. Through the detection and validation of key regions on the marksheet, such as roll numbers and names, the system minimizes human error and speeds up the verification process. This solution significantly enhances the efficiency of document verification workflows, particularly in educational institutions and organizations dealing with high volumes of submissions.

Future Work:

- **Integration with Government and Educational Portals**: The system can be seamlessly incorporated into existing document verification workflows in portals like MahaDBT, where students submit marksheets for scholarships or financial aid. Automating the validation process will reduce manual intervention, ensuring faster processing of applications.
- **Support for Multiple Document Types**: In addition to marksheets, the system can be expanded to support other types of documents such as birth certificates, ID proofs, and tax forms, making it versatile for various sectors.
- **Machine Learning and AI Enhancements**: Incorporating AI-based models could further improve the accuracy of text recognition and field validation, particularly for handwritten documents or low-quality scans. Additionally, AI models can predict and detect discrepancies or anomalies in documents.
- **Multi-language Support**: As the system scales, incorporating support for multiple languages, especially regional languages, would make it more accessible for diverse user bases.
- **Cloud-Based Solution**: Deploying the system as a cloud-based service would enable organizations to easily integrate it into their platforms, offering scalable and on-demand document validation solutions.

Reference

[1] K. Ingale, R. Konde, S. Khachane, H. Suryawanshi, and J. Kapadnis, "Enhancing Document Verification with Digital Signature and OCR Algorithm," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 5, no. 11, Nov. 2023.

[2] M. Rajapashea, M. Adnan, A. Dissanayaka, D. Guneratne, and K. Abeywardane, "Multi-Format Document Verification System," *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, Dec. 2020.

[3] A. Shende, M. Mullapudi, and N. Challa, "Enhancing Document Verification Systems: A Review of Techniques, Challenges, and Practical Implementations," *International Journal of Computer Engineering & Technology*, Jan. 2024.

[4] A. Salge, S. Shindkar, S. Malve, S. Dabhikar, and S. Desai, "Document Verification Using OCR," *YMER*, vol. 23, no. 5, May 2024.

[5] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of YOLO Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, Apr. 2022.

[6] S. Redmon, R. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *arXiv*, May 2016.

[7] R. Jana, A. Roy Chowdhury, and M. Islam, "Optical Character Recognition from Text Image," *International Journal of Computer Applications Technology and Research*, vol. 3, no. 4, pp. 239-243, Aug. 2014.

[8] K. A. Hamad and M. Kaya, "A Detailed Analysis of Optical Character Recognition Technology," *International Journal of Applied Mathematics, Electronics and Computers*, vol. 4, no. 2, pp. 113-119, Dec. 2016. DOI: 10.18100/ijamec.270374.

[9] P. N. Mahale, S. C. Amrutkar, P. S. Mathpati, S. J. Ubale, and R. J. Shaikh, "Automated Document Verification," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 4, no. 3, pp. 62-65, Apr. 2024. doi: 10.48175/568.