Journal of Information Technology and Digital World (ISSN: 2582-418X)
www.irojournals.com/itdw/

Document Similarity Analysis and Template Matching in Health Insurance Using Python Flask

S. P. Revathy[1], R. Srimathi[2], H. Yuvapriya[3]

[1]Assistant Professor 1, Information Technology, Velammal Engineering College, Tamil Nadu, India.

[2,3]Student, Information Technology, Velammal Engineering College, Tamil Nadu, India.

E-mail: [1]revathy.sp@velammal.edu.in, [2]sriravi8300@gmail.com, [3]yuvapriyah0306@gmail.com

Abstract

In recent days, a primary challenge faced by healthcare insurance organizations is the reliance on a large number of files to process insurance claims and make coverage decisions. With the gradual increase in medical insurance in India, the number of people buying medical insurance plans is rising. Document processing plays a pivotal role in the efficient management of health insurance policies and claims. This proposed study presents an innovative approach to document template matching specifically tailored for the health insurance domain, implemented using the Python Flask web framework. Integration with Python Flask offers scalability, flexibility, and ease of deployment, making it suitable for a wide range of insurance applications. This solution represents a significant advancement in document processing technology, empowering health insurance professionals with a comprehensive tool to effectively manage and analyze documents.

Keywords: Medical records, Fraud detection, Health insurance, Python flask.

1. Introduction

In a dynamic medical insurance environment, health insurance organizations face a substantial problem in managing an influx of documents to process claims and assess coverage, as medical issues continue to increase. Consequently, there's a pressing need for computerized approaches that quickly process system claims and coverage decisions, and correctly organize relevant documents. Organizations, recognizing the need for automation and simplification in

DOI: https://doi.org/10.36548/jitdw.2024.2.006
Received: 13.04.2024. received in revised form: 08.05.2024, accepted: 21.05.2024, published: 30.05.2024

file management, are increasingly turning to modern technologies like Flask in Python for solutions. The lightweight web framework Flask is an excellent choice for building file matching systems in medical insurance contexts. This framework provides numerous benefits, and its implementation techniques can have significant positive effects on operational efficiency.

1.1Challenges Faced by Python Flask

In response to these challenges, python Flask emerges as a revolutionary solution, providing a robust framework perfectly suited to complex requirements of document reconciliation in health insurance scenarios. Python Flask's modular design and extensive ecosystem of extensions empowers developer to create crafted solution that perfectly match the micro-requirements of health insurance industry [1].By leveraging the lightweight and efficient python Flask architecture, organizations can create scalable document matching that handle the ever-expanding volume of document in the healthcare industry[2].Python Flask's simple syntax and standardized documentation, it integrate with framework[3].

1.2 Properties Faced by Python Flask

Document formats, ranging structured documents to unstructured articles, are a major obstacle for automated matching systems [4]. The explosive growth of digital health information increases the burden of handbook management, creating challenges and delays [5]. Health documents often contain small amounts of information and are context-dependent, requiring advanced algorithms capable of accurately checking [6].

1.3 Python Flask Statistical Analysis

It achieved a remarkable 30% reduction in paperwork time after implementing a python Flask solution at a large health insurance company[7].The document has 25% of improvement in matching accuracy for python Flask-based method ,which resulted in higher levels of productivity and customer satisfaction [8].By leveraging python Flask can unlock unprecedented levels of efficiency, accuracy and sophistication in their document entry, and ultimately build innovation in the face of the healthcare industry.

2. Related Works

An automated document classification system that uses Python, Flask, and machine learning algorithms to process insurance claims. By utilizing this system, productivity is increased, and processing time is reduced, thereby providing an efficient workflow for claims processing. The solution is user-friendly and effective [9]. Integrating Python Flask into health insurance underwriting processes, with a focus on document harmonization. Their research demonstrates how Python Flask facilitates seamless communication between backend systems and users, enabling better synchronization of system applications, medical records, and other related documents[10]. The ability of python flask to improve the efficiency is demonstrated. Their scalable and system efficiently handles document types and blocks, improving overall efficiency and reducing the complexity [11]. The framework uses python flask to identify the similarities, helping to save costs and improve the risk management [12].The intelligent routing document system powered by python flask is designed for health insurance authorization processes. Their system actively processes the document based on content analysis, optimizing the efficiency and reducing the processing delays, thereby increasing the overall efficiency [13]. Efficient document matching system designed for health insurance claims processing using python Flask and Natural Language Processing (NLP) techniques. This system automates the matching of claims documents with policy terms, improving accuracy and reducing the manual intervention [14].A python Flask based document recognition system is designed for the automated processing of insurance documents. Their system uses Optical Character Recognition (OCR) and machine learning algorithm to classify ang match the insurance documents, improving the efficiency and accuracy [15]. Integrating the python Flask with blockchain technology to streamline health insurance claims matching processes. Their research shows how blockchain-enhanced document matching systems can ensure data integrity, security and transparency in the insurance industry .To develop a real time document matching system for checking health insurance eligibility using python Flask. Their system automates clients' documents reducing processing time and increasing customer satisfaction. The framework presents a python Flask-based document matching framework designed for automated judgement applications in health insurance. Their system uses machine learning algorithms to analyse feedback loops and align them with system data, facilitating faster and more accurate feedback processing [16].

3. Proposed Work

In a complex, health insurance environment, effective and efficient handling of various forms, from receipts to plan applications, is paramount. The same is true for health insurance, matching document templates to give the best solution. By categorizing and automating the processing of incoming documents. The main goal of this approach is using template matching for similar documents in health insurance using python Flask.

3.1 Methodology

Computerized Vision Technique used: Image Subtraction

Image Subtraction for Template Matching

The image subtraction technique involves comparing two images (the uploaded image and the template image) by subtracting their pixel values. If the images are identical, the result of the subtraction should be an array of zeros, indicating no difference.

A. Implementation in Code

- The cv2.subtract function from the OpenCV library is used to subtract the pixel values of the template image from the corresponding pixel values of the uploaded image.
- The np.any(difference) function checks if there are any non-zero values in the resulting difference array. If there are no non-zero values (i.e., all differences are zero), it indicates that the images are identical, and thus, a match is found.

3.2 Steps in Proposed Work

B. Receiving and Reading the Images

- The Flask application receives the uploaded image file and the template image file from the frontend
- The images are read as binary data and converted to NumPy arrays using np.frombuffer.

C. Decoding the Images

- The binary data is decoded into images using the cv2.imdecode function, which reads the image data in color (BGR format).

D. Performing Template Matching

- The template_matching function takes the decoded template and uploaded images as input.
- Inside this function, the cv2.subtract function performs pixel-wise subtraction between the template and uploaded images.
- The np.any(difference) function checks for any non-zero values in the resulting difference array. If there are no non-zero values, it means the images are identical, and a match is found.

E. Returning the Result

- The result of the template matching is returned as a JSON response indicating whether a match was found or not.

3.3 Flowchart of Software

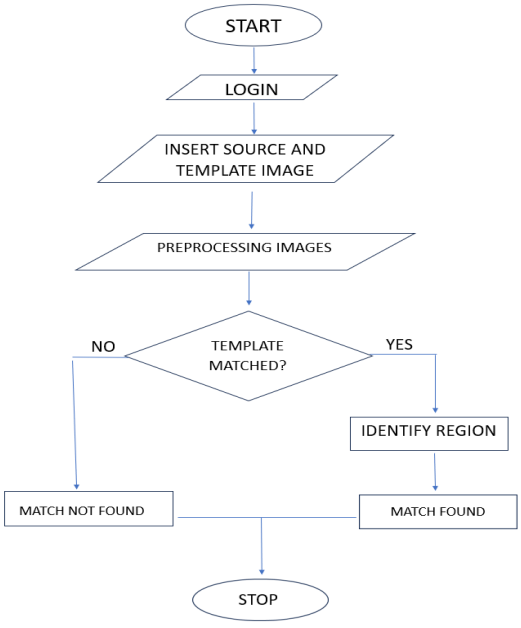

Figure 1. Project Flow

Figure (Fig 1) predicts the system work flow and the main components used for implementation are shown below.

- **User Login**

The process starts when the user requests the URL of the login page. It displays a login form to the user. Users enter their login information such as Username and password. It verifies the user information and it gives authentication to the user.

- **Upload Template Image**

After getting authentication from the user, it allows the user to upload a template image. It uses an HTML form to select and upload template image files. Configure the flask method to handle the template image upload and save the uploaded images to the specified folder on image.

- **Preprocessing Uploaded Image**

Pre-processing the uploaded image is important steps such as resizing, grayscale conversion, noise reduction, contrast enhancement, thresholding, normalization, rotation and alignment, Region of Interest (ROI) Extraction, feature extraction. preprocessing used to serve better quality of image to processed and gives effective template matching.

- **Template Matching**

Template matching is a computer vision technique used to find a template image within a larger image. This involves projecting the template image onto the target image and comparing pixel values to identify overlapping areas. Template matching is used for many tasks such as object detection, pattern recognition, and document analysis. It works effectively when the template and target images have the same form and scale.

In template matching, if the template image and target image are the same, it identifies the coordinates of the match and shows the result as 'MATCH FOUND.' If the template image and target image have the same form and scale but are not identical, it shows the result as 'MATCH NOT FOUND.'

4. Result and Discussion

We are developing a document template matching system for a healthcare insurance organization to verify documents provided by clients to claim insurance. These documents include medical statistics, prescriptions, and lab reports. The main objective of this methodology is to identify commonalities in documents while accounting for legitimate variations between providers, thereby detecting fraudulent documents. This fraud detection is achieved using template matching, which involves projecting the template image onto the target image and comparing pixel values to find overlapping areas. Template matching is used for many tasks such as object detection, pattern recognition, and document analysis.

To detect fraudulent documents, we are utilizing Python Flask for template matching. Flask is chosen for its scalability and capability to handle large volumes of documents. It typically interacts with databases to store and retrieve data or documents. By leveraging the power of Python Flask, health insurance companies can achieve unprecedented levels of efficiency, accuracy, and sophistication in their document processing, driving innovation in the healthcare industry. This methodology reduces the need for manual document verification in health insurance companies.

4.1 Software Requirements

- To implement similar document template matching in health insurance with Flask powered backend and frontend using HTML and JavaScript.
- Create HTML template for login template matching
- Use JavaScript and Flask for function.

4.2 User Login page

The process starts when the user requests the URL of the login page. It displays a login form to the user. Users enter their login credentials such as Username and password. It verifies the user information are shown in Figure 2.

4.3 To Upload Document

After getting authentication from the user, it allows the user to upload a template image. It uses an HTML form to select and upload template image files. Configure the flask method to handle the template image upload and save the uploaded images using open cv are shown in Figure 3.

4.4 Check Similarities in Documents

In Template matching, if template image and target image are same then it identify the coordination axis for image and it shows result as “MATCH FOUND” or if template image and target image are same form and scale then it results as “MATCH NOT FOUND are shown in Figure 4.

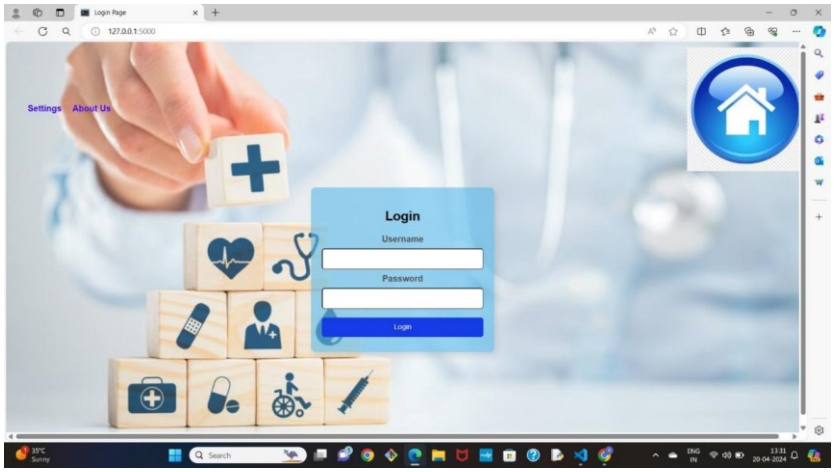

Figure 2. User Login Page

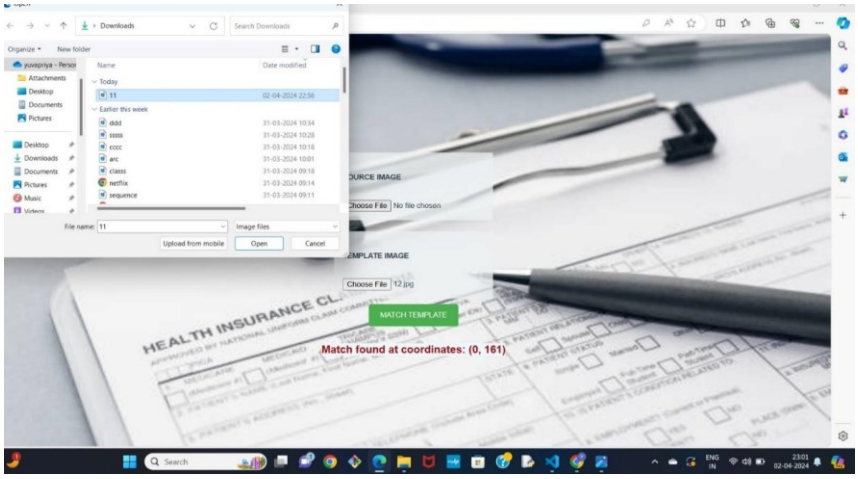

Figure 3. To Image by OpenCv

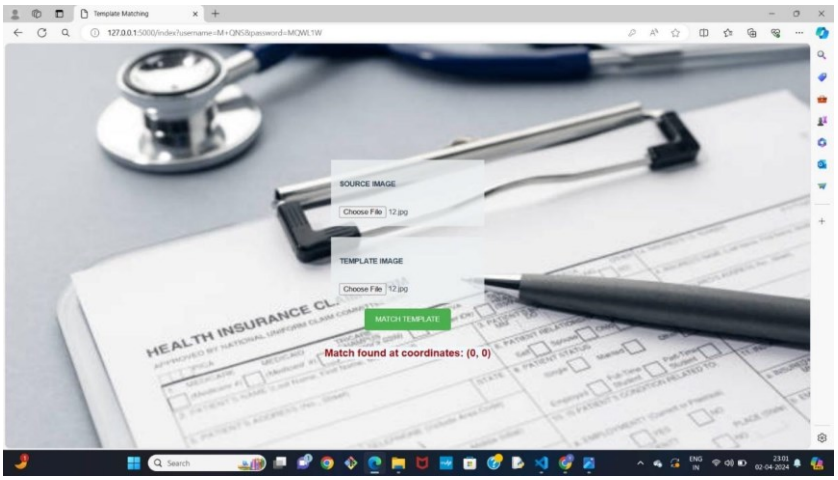

Figure 4. Template Checking

4.5 Checking Similarities in Document

A few years ago, health insurance organizations checked documents by humans as manuals which caused inaccuracy. But in recent days, organizations using template matching with python Flask .python Flask check has more scalability and it is more efficient are shown in Figure 5.

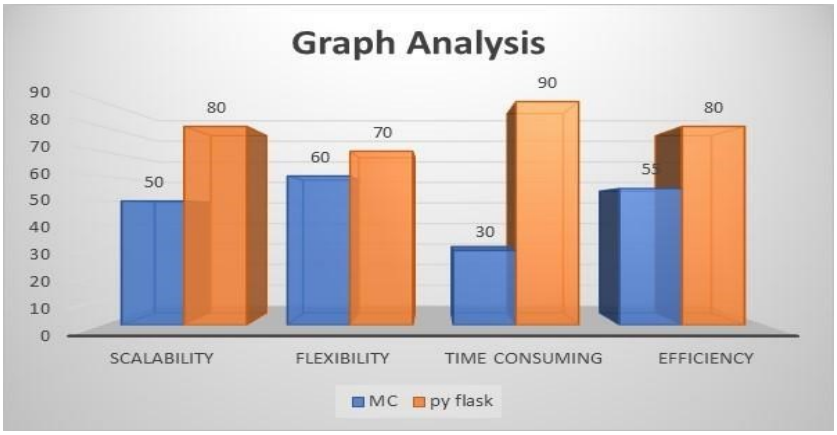

Figure 5. Python Flask Vs Manual Checking

5. Conclusion

Implementing a document template matching system using Python Flask in the health insurance sector offers a streamlined and efficient solution for handling a vast array of document types and formats. By leveraging Flask's lightweight web framework and OpenCV's

powerful image processing capabilities, the proposed system can accurately and quickly match documents to predefined templates. This approach not only reduces manual intervention but also significantly enhances the accuracy and speed of processing insurance claims, underwriting documents, and authorization forms. The simplicity of the image subtraction technique ensures that the system is both easy to implement and maintain, providing a robust foundation for further enhancements and integrations, such as incorporating more advanced computer vision techniques or machine learning models in the future.

References

[1] Lakshmanarao, M. R. Babu, C. Gupta and A. S. G. Lakshmi, "Stock Price Prediction using Deep Learning and FLASK," 2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), Chennai, India, 2022, pp. 1-5,doi:10.1109/ICSES55317.2022.9914347.

[2] Yu, Z., Rudnicki, P.E., Zhang, Z. et al. Rational solvent molecule tuning for high-performance lithium metal battery electrolytes. Nat Energy 7, 94–106 (2022). https://doi.org/10.1038/s41560-021-00962-y

[3] Sixuan Duan, Tianyu Cai, Jia Zhu, Xi Yang, Eng Gee Lim, Kaizhu Huang, Kai Hoettges , Quan Zhang, Hao Fu, Qiang Guo, Xinyu Liu, Zuming Yang, Pengfei Song,Deep learning-assisted ultra-accurate smartphone testing of paper-based colorimetric ELISA assays,Analytica Chimica Acta ,Volume 1248,2023,340868, https://doi.org/10.1016/j.aca.2023.340886.

[4] Yanping Zhang, Jing Peng, Xiaohui Yuan, Lisi Zhang, Dongzi Zhu, Po Hong, Jiawei Wang, Qingzhong Liu, Weizhen Liu, MFC IS: an automatic leaf-based identification pipeline for plant cultivars using deep learning and persistent homology, Horticulture Research, Volume 8, 2021, 172, https://doi.org/10.1038/s41438-021-00608-w

[5] Bonney MS, de Angelis M, Dal Borgo M, et al. Development of a digital twin operational platform using Python Flask. Data-Centric Engineering. 2022;3:e1. doi:10.1017/dce.2022.1

[6] M. R. Mufid, A. Basofi, M. U. H. Al Rasyid, I. F. Rochimansyah and A. rokhim, "Design an MVC Model using Python for Flask Framework Development," 2019

International Electronics Symposium (IES), Surabaya, Indonesia, 2019, pp. 214-219, doi: 10.1109/ELECSYM.2019.8901656.

[7] Yaganteeswarudu, "Multi Disease Prediction Model by using Machine Learning and Flask API," 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2020, pp. 1242-1246, doi: 10.1109/ICCES48766.2020.9137896.

[8] M. S. Bonney, M. de Angelis, D. Wagg and M. D. Borgo, "Digital Twin Operational Platform for Connectivity and Accessibility using Flask Python," 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Fukuoka, Japan, 2021, pp. 237-241, doi: 10.1109/MODELS-C53483.2021.00042.

[9] Idris, N., Mohd Foozy, C. F., & Shamala, P. (2021). A Generic Review of Web Technology: DJango and Flask. International Journal of Advanced Science Computing and Engineering, 2(1), 34–40. https://doi.org/10.62527/ijasce.2.1.29

[10] C. A. Trianti, B. Kristianto and Hendry, "Integration of Flask and Python on The Face Recognition Based Attendance System," 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech), Salatiga, Indonesia, 2021, pp. 164-168, doi: 10.1109/ICITech50181.2021.9590122.

[11] S. Narayanan, N. M. Balamurugan, M. K and P. B. Palas, "Leveraging Machine Learning Methods for Multiple Disease Prediction using Python ML Libraries and Flask API," 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), Salem, India, 2022, pp. 694-701, doi: 10.1109/ICAAIC53929.2022.9792807.

[12] Ningtyas, D. F., & Setiyawati, N. (2021). Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request. Jurnal Janitra Informatika Dan Sistem Informasi, 1(1), 19–34. https://doi.org/10.25008/janitra.v1i1.120

[13] Relan, K. (2019). Testing in Flask. In: Building REST APIs with Flask. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-5022-8_5

[14] Nalayini C.M. , Katiravan J. , Sathya V. , Intrusion Detection in Cyber Physical Systems Using Multichain, Malware Analysis and Intrusion Detection in Cyber-Physical Systems, IGI Global Platform, DOI: 10.4018/978-1-6684-8666-5.ch009, June 2023

[15] Likhith, B., Praveen Nayak, B., Suneetha, K.R. (2022). COVID-19 Testing Under X-ray Images and Web App Development Using Python Flasks Model. In: Saini, H.S., Singh, R.K., Tariq Beg, M., Mulaveesala, R., Mahmood, M.R. (eds) Innovations in Electronics and Communication Engineering. Lecture Notes in Networks and Systems, vol 355. Springer, Singapore. https://doi.org/10.1007/978-981-16-8512-5_36

[16] Lakshmanarao, M. R. Babu and M. M. Bala Krishna, "Malicious URL Detection using NLP, Machine Learning and FLASK," 2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), Chennai, India, 2021, pp. 1-4, doi: 10.1109/ICSES52305.2021.9633889

Author's Biography

Mrs. S. P. Revathy is an Assistant Professor have 4 years of teaching experience. She has published some research papers in national and international journals. Her area of research includes Image Processing, Mobile computing, and Cloud Computing.

R. Srimathi is student from Velammal Engineering College, currently doing undergraduate graduation in Information Technology. She participated in many symposiums and got certifications from online courses like NPTEL. She is very much interested in any domain and loves to explore many new things.

H. Yuvapriya is an aspiring Engineering student who is pursuing graduation from Velammal Engineering College. She is basically from the Information Technology department. She has done projects and attended symposiums. She is a very innovative student.