X-RAY ANALYSIS OF PALM AND FINGERS FOR FRACTURE DETECTION

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

BOJJA YASWANTH KUMAR REDDY (21UEID0500) **(VTU23983)**
VIBRAPATANAM VENKATA RAO (21UECS0750) **(VTU23998)**
MUNDRU SAI SANDEEP (21UEID0502) **(VTU24052)**

Under the guidance of
Mr. Anand Krishnan Iyer, Principal Consultant, ICC Global support centre
and
Dr. M.SARAVANA KARTHIKEYAN M.E.,M.B.A.,Ph.D.,
ASSISTANT PROFESSOR - SENIOR GRADE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

X-RAY ANALYSIS OF PALM AND FINGERS FOR FRACTURE DETECTION

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

BOJJA YASWANTH KUMAR REDDY	(21UEID0500)	**(VTU23983)**
VIBRAPATANAM VENKATA RAO	(21UECS0750)	**(VTU23998)**
MUNDRU SAI SANDEEP	(21UEID0502)	**(VTU24052)**

Under the guidance of
Mr. Anand Krishnan Iyer, Principal Consultant, ICC Global support centre
and
Dr. M.SARAVANA KARTHIKEYAN M.E.,M.B.A.,Ph.D.,
ASSISTANT PROFESSOR - SENIOR GRADE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "X-RAY ANALYSIS OF PALM AND FINGERS FOR FRACTURE DETECTION" by "BOJJA YASWANTH KUMAR REDDY (21UEID0500).VIBRAPATANAM VENKATA RAO(21UECS0750), MUNDRU SAI SANDEEP (21UEID0502)" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Industry Supervisor
Mr. Anand Krishnan Iyer
Principal Consultant
ICC Global support centre
May, 2024

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents my ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

BOJJA YASWANTH KUMAR REDDY

Date: / /

VIBRAPATANAM VENKATA RAO

Date: / /

MUNDRU SAI SANDEEP

Date: / /

APPROVAL SHEET

This project report entitled X-RAY ANALYSIS OF PALM AND FINGERS FOR FRACTURE DETECTION by BOJJA YASWANTH KUMAR REDDY (21UEID0500), VIBRAPATANAM VENKATA RAO (21UECS0750), MUNDRU SAI SANDEEP (21UEID0502) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Dr.M. Saravana Karthikeyan M.E.,M.B.A.,Ph.D.,
Assistant Professor

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We are highly indebted to our **Industry Supervisor Mr. ANAND KRISHNAN LYER, PRINCIPAL CONSUITANT, ICC GLOBAL SUPPORT CENTER** for his guidance and constant supervision as well as for providing necessary information and support in completing the project

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Dr.M. SARAVANA KARTHIKEYAN M.E.,M.B.A.,Ph.D.,** for his/her cordial support, valuable information and guidance, he/she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. U.HEMAVATHI, M.E., Ms.E.CHANDRALEKHA,M.Tech.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

BOJJA YASWANTH KUMAR REDDY (21UEID0500)

VIBRAPATANAM VENKATA RAO (21UECS0750)

MUNDRU SAI SANDEEP (21UEID0502)

ABSTRACT

X-ray imaging plays a pivotal role(very important role) in medical diagnosis, security screening, and industrial inspection due to its ability to penetrate opaque objects. Efficient and accurate analysis of X-ray images is crucial for effective decision-making in various domains.In recent years, machine learning techniques, particularly Convolutional Neural Networks (CNNs), have shown remarkable success in image classification tasks. By analyzing complex patterns in medical images, these algorithms empower clinicians to make more accurate.Experimental results demonstrate the effectiveness of the proposed approach in accurately classifying X-ray images into different categories such as normal, abnormal, and specific pathology types.The texture of a cancer bone is different compared to a healthy bone in the affected region. But in the dataset, several images of cancer and healthy bone are having similar morphological characteristics. This makes it difficult to categorize them. To tackle this problem, we first find the best suitable edge detection algorithm after that two feature sets one with hog and another without hog are prepared. To test the efficiency of these feature sets, two machine learning models, support vector machine (SVM) and the Random forest, are utilized. The features set with hog perform considerably better on these models.The accuracy of the project is 95 percentage.

Keywords: Convolutional Neural Networks, Deep learning, Early detection, Medical image X-ray, SVM.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolutional Neural Network
RFS	Random Forest Algorithm
MRI	Magnetic Resonance Imaging
ReLE	Rectified Linear Unit
SVM	Support Vector Machine
SGD	Stochastic Gradient Descent

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Convolutional Neural Networks (CNNs) are a class of deep neural networks, primarily used for analyzing visual imagery. They are inspired by the biological visual cortex and designed to automatically and adaptively learn spatial hierarchies of features from data. CNNs have been instrumental in various computer vision tasks such as image classification, object detection, segmentation, and more. These are the building blocks of CNNs. A convolutional layer consists of a set of learnable filters or kernels. Each filter is small spatially (along width and height) but extends through the full depth of the input volume. Convolution involves sliding these filters over the input image, performing element-wise multiplication with the local region, and summing up the results to produce a feature map.Pooling Layers: Pooling layers are used to progressively reduce the spatial size of the representation to decrease the computational complexity and number of parameters in the network, thus controlling overfitting.

Common pooling operations include max pooling and average pooling.Activation Functions: Activation functions introduce non-linearity to the network, enabling it to learn complex patterns. Popular choices include ReLU (Rectified Linear Unit), sigmoid, and tanh. Fully Connected Layers: After several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have connections to all activations in the previous layer, similar to traditional neural networks.Flattening: Before passing the output from convolutional and pooling layers to the fully connected layers, the data is usually flattened into a vector.

Backpropagation: CNNs are trained using backpropagation, a supervised learning algorithm. During training, the network adjusts its parameters (weights and biases) through optimization algorithms like gradient descent to minimize the difference between predicted and actual labels.
Loss Function: The loss function measures how well the network's predictions match the actual target labels during training. Common loss functions for classification tasks include cross-entropy loss.

Optimization: Optimization algorithms like Stochastic Gradient Descent (SGD), Adam, or RMSprop are used to update the network's parameters during training.

1.2 Aim of the project

To creating an X-ray detector for organ dysfunction using Convolutional Neural Networks (CNNs) in machine learning. CNNs are particularly well-suited for image recognition tasks where X-ray detector takes input as the image

.

1.3 Project Domain

Develop a deep learning model using Convolutional Neural Networks (CNNs) to assist radiologists in diagnosing pneumonia from chest X-ray images. Pneumonia is a common and potentially life-threatening condition, and early detection is crucial for timely treatment and improved patient outcomes. rain the CNN model using the preprocessed chest X-ray images. Utilize techniques such as transfer learning and early stopping to improve training efficiency and prevent overfitting. Monitor training progress by tracking metrics such as loss and accuracy on the validation set.Evaluate the trained model's performance on the test set using standard evaluation metrics such as accuracy, precision, recall, and F1-score.Visualize model predictions and examine misclassified images to gain insights into potential areas for improvement.Deploy the trained model as a web or mobile application that allows healthcare professionals to upload chest X-ray images for automatic pneumonia detection.Integrate the model into existing clinical workflows to assist radiologists in their diagnostic decision-making process.

1.4 Scope of the Project

The abstract covers various domains where ML is applied in healthcare, including medical imaging, personalized medicine, patient monitoring, drug discovery, and natural language processing. It explores how ML algorithms enhance diagnostic accuracy, treatment efficacy, and operational efficiency in healthcare settings. Additionally, the abstract addresses challenges and considerations associated with the integration of ML technologies into medical practice, such as data privacy, regulatory compliance, and algorithm bias. The scope also encompasses current research findings, methodologies, and emerging trends in the field of ML-driven healthcare innovations.

Chapter 2

LITERATURE REVIEW

[1]V. L. F. Lum, W. K. Leow, Y. Chen, T. Sen Howe, and M. A. Png, "Combining classifiers for bone fracture detection in x-ray images," Proc. - Int. Conf. Image Process. ICIP, vol. 1, pp. 1149–1152, 2005, doi: 10.1109/ICIP.2005.1529959.

[2]Sujay Kumar Mandal, Muralidhar Pullakandam, Rama Muni Reddy Yanamala, "Pneumonia Detection Using Transfer Learning And Hardware Implementation in Edge TPU", 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp.1-5, 2023.

[3]Sashikanta Prusty, Srikanta Patnaik, Sujit Kumar Dash, "ResNet50V2: A Transfer Learning Model to Predict Pneumonia with chest X-ray images", 2022 International Conference on Machine Learning, Computer Systems and Security (MLCSS), pp.208-213, 2022.

[4]Chamodi Fernando, Shammi Kolonne, Hashara Kumarasinghe, Dulani Meedeniya, "Chest Radiographs Classification Using Multi-model Deep Learning: A Comparative Study", 2022 2nd International Conference on Advanced Research in Computing (ICARC), pp.165-170, 2022.

[5]Kim HE, Cosa-Linan A, Santhanam N, Jannesari M, Maros ME, Ganslandt T (2022) Transfer learning for medical image classification: a literature review. BMC Med Imaging 22:69

[6]Sarvamangala DR, Kulkarni RV (2022) Convolutional neural networks in medical image understanding: a survey. Evol Intell 15:1–22

[7]M. Lotfy, R. M. Shubair, N. Navab, and S. Albarqouni, "Investigation of Focal Loss in Deep Learning Models for Femur Fractures Classification," 2019 Int. Conf. Electr. Comput. Technol. Appl. ICECTA 2019, pp. 2–5, 2019, doi: 10.1109/ICECTA48151.2019.8959770.

[8]Z. Wu, X. Mo, H. Zhou, L. Liu, and J. Li, “Classification of reservoir fracture development level by convolution neural network algorithm,” ICNC-FSKD 2018 - 14th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov., pp. 243–250, 2018, doi: 10.1109/FSKD.2018.8687232.

[9]Yahya Sugandi, Indah Soesanti, Hanung Adi Nugroho, “ A Systematic Literature Review of Convolutional Neural Network Architecture for Lung Disease Detection”, 2023 6th International Conference on Information and Communications Technology (ICOIACT), pp.230-235, 2023

[10]L. Wang, H. Cheng, H. Lan, Y. Zheng, and K. Li, “Automatic recognition of pertrochanteric bone fractures in femur using level sets,” Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, vol. 2016-October, pp. 3851–3854, 2016, doi: 10.1109/EMBC.2016.7591568.

[11]R. S. Prihatini, A. H. Setyaningrum, and I. M. Shofi, “Texture analysis and fracture identification of lower extremity bones X-ray images,” Int. Conf. Electr. Eng. Comput. Sci. Informatics, vol. 2017-December, no. September, pp. 19–21, 2017, doi: 10.1109/EECSI.2017.8239113.

[12]R. Bagaria, S. Wadhwani, and A. K. Wadhwani, “Different techniques for identification of a bone fracture in analysis of medical image,” Proc. - 2020 IEEE 9th Int. Conf. Commun. Syst. Netw. Technol. CSNT 2020, pp. 327–332, 2020, doi: 10.1109/CSNT48778.2020.9115760.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

Patients usually start by contacting the doctor's office or clinic to schedule an appointment. Some may be referred by another healthcare provider.Upon arrival, patients fill out registration forms and provide identification, insurance information, and medical history.

A nurse or medical assistant may take the patient's vital signs (e.g., blood pressure, pulse, temperature) and gather basic information about the reason for the visit.The patient's medical records, if available, may be reviewed for any relevant history.The doctor conducts an initial interview with the patient to gather a detailed medical history, including current symptoms, past medical conditions, medications, allergies, family history, and lifestyle factors.The doctor may ask about the patient's chief complaint, how long symptoms have been present, and any factors that exacerbate or alleviate the condition.The doctor performs a physical examination, which can vary based on the patient's complaint and medical history.

This may involve listening to the heart and lungs, checking reflexes, or examining specific body areas.A general physical examination assesses overall health and can help identify issues unrelated to the chief complaint.Depending on the patient's condition and symptoms, the doctor may order diagnostic tests such as blood tests, imaging (X-rays, MRI, CT scans), or other specialized tests to aid in diagnosis.Based on the patient's history, physical examination, and diagnostic test results, the doctor formulates a diagnosis.The doctor discusses the diagnosis with the patient and proposes a treatment plan, which may include medication, lifestyle changes, physical therapy, surgery, or referral to a specialist.

3.2 Proposed System

The proposed detection of organ dysfunction using ml through cnn presents numerous advantages over traditional manual methods. Firstly, by leveraging image recognition technology, the system offers a highly accurate and efficient way to track attendance. Facial recognition algorithms can quickly and accurately identify individuals, eliminating the need for manual entry or physical sign-

in sheets. This not only reduces the potential for human error but also streamlines the attendance tracking process, saving time and resources for administrators.

Bone fractures are a prevalent issue faced by many individuals, often resulting from accidents. X-rays are commonly used by doctors to predict fractures. However, manually interpreting X-rays can be challenging, as small fractures may be overlooked, leading to potential future harm. This project aims to address this issue by utilizing artificial intelligence applications, such as ML and DL techniques, to analyze and classify images of hand, leg, chest, fingers, and wrist fractures in a precise manner. Specifically, Convolutional Neural Networks (CNNs) are employed to develop various models that offer a step-by-step image analyzing algorithm to accurately predict whether a bone is fractured or normal, providing a better solution for fracture detection.

3.3 Feasibility Study

3.3.1 Economic Feasibility

Estimate the costs associated with data acquisition, preprocessing, model development, and software implementation. Consider expenses for hardware infrastructure, software licenses, personnel salaries, and any additional resources required for the project.Evaluate the potential economic benefits of deploying the bone fracture detection system in clinical practice. Estimate the savings in diagnostic time, healthcare costs, and patient outcomes achieved through more accurate and timely fracture detection. Compare the projected ROI with the initial investment to assess the economic viability of the project.Conduct a cost-benefit analysis to weigh the economic advantages of using automated fracture detection against the expenses involved in system development and deployment. Consider both short-term and long-term financial implications to make informed decisions about resource allocation and investment strategies.

3.3.2 Technical Feasibility

Assess the availability of labeled X-ray datasets containing bone fracture images. Ensure the datasets are sufficiently large and diverse to train robust CNN models. Evaluate the quality and consistency of the data, considering factors like image resolution, noise, and annotation accuracy.: Determine the technical feasibility of implementing CNN architectures for bone fracture detection. Investigate the computational resources required for training and inference, including GPU availability, memory requirements, and processing speed.Conduct preliminary experiments to evaluate the performance of CNN models on the available datasets. Assess metrics such as accuracy, sensitivity,

specificity, and computational efficiency to gauge the feasibility of achieving reliable fracture detection results.

3.3.3 Social Feasibility

Assess the potential societal benefits of implementing a bone fracture detection system, such as improved patient outcomes, reduced diagnostic errors, and enhanced healthcare efficiency. Consider the impact on healthcare providers, patients, and healthcare systems in terms of workload, resource allocation, and overall quality of care.Evaluate the accessibility of the detection system to diverse populations, including underserved communities and regions with limited healthcare infrastructure. Ensure that the technology does not exacerbate existing disparities in healthcare access and treatment outcomes. Address ethical concerns related to patient privacy, informed consent, and the responsible use of medical data. Develop policies and procedures to safeguard patient confidentiality, mitigate algorithmic biases, and ensure transparent communication with stakeholders regarding the capabilities and limitations of the technology.

3.4 System Specification

3.4.1 Hardware Specification

- GPU: A dedicated graphics processing unit (GPU) with CUDA support is recommended for accelerated training of deep learning models. NVIDIA GPUs, such as GeForce RTX or Tesla series, are commonly used for machine learning tasks.
- CPU: A multicore processor with sufficient processing power to handle data preprocessing, model training, and inference tasks. Intel Core i7 or AMD Ryzen processors are suitable choices.
- RAM: A minimum of 16 GB of RAM is recommended to support memory-intensive operations during model training and data manipulation.
- Storage: Solid-state drives (SSDs) with ample storage capacity (e.g., 500 GB or more) for storing datasets, model checkpoints, and software libraries.

3.4.2 Software Specification

- Deep Learning Framework: Choose a deep learning framework for building and training CNN models. Popular frameworks include TensorFlow, PyTorch, and Keras, which offer comprehensive APIs and support for GPU acceleration.

- Python: Utilize the Python programming language for scripting and implementing machine learning algorithms. Ensure compatibility with the selected deep learning framework and associated libraries.
- Data Processing Libraries: Install libraries such as NumPy, pandas, and scikit-learn for data preprocessing, manipulation, and feature extraction from X-ray images.
- Development Environment: Set up an integrated development environment (IDE) such as Jupyter Notebook, PyCharm, or Visual Studio Code for writing code, debugging, and experimentation.
- Version Control: Use version control systems like Git to manage codebase revisions, collaborate with team members, and track changes to the project over time.

3.4.3 Standards and Policies

Standards and policies play a crucial role in ensuring the reliability, security, and ethical use of the attendance automation system with image recognition technology.

Data Protection Regulations: Compliance with data protection regulations such as GDPR (General Data Protection Regulation) or CCPA (California Consumer Privacy Act) ensures that personal data, including facial images, is collected, stored, and processed in a lawful and transparent manner, with appropriate measures to protect individuals' privacy rights.

Ethical Use Policy: Establishing clear guidelines for the ethical use of facial recognition technology is essential to prevent misuse or abuse. This policy should outline principles for fair and responsible use, including obtaining consent, ensuring accuracy and fairness, and protecting against bias and discrimination.

Security Standards: Adherence to security standards such as ISO 27001 helps safeguard sensitive data and prevent unauthorized access or breaches. Implementing encryption, access controls, and regular security audits ensures the confidentiality, integrity, and availability of attendance data.

Accessibility Guidelines: Following accessibility guidelines such as WCAG (Web Content Accessibility Guidelines) ensures that the attendance automation system is usable by individuals with disabilities, providing equal access to all users.

Training and Awareness Programs: Regular training and awareness programs educate users and administrators about the system's capabilities, limitations, and ethical considerations, promoting responsible usage and mitigating risks associated with misuse or misunderstanding.

Chapter 4

METHODOLOGY

4.1 Architecture Diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection

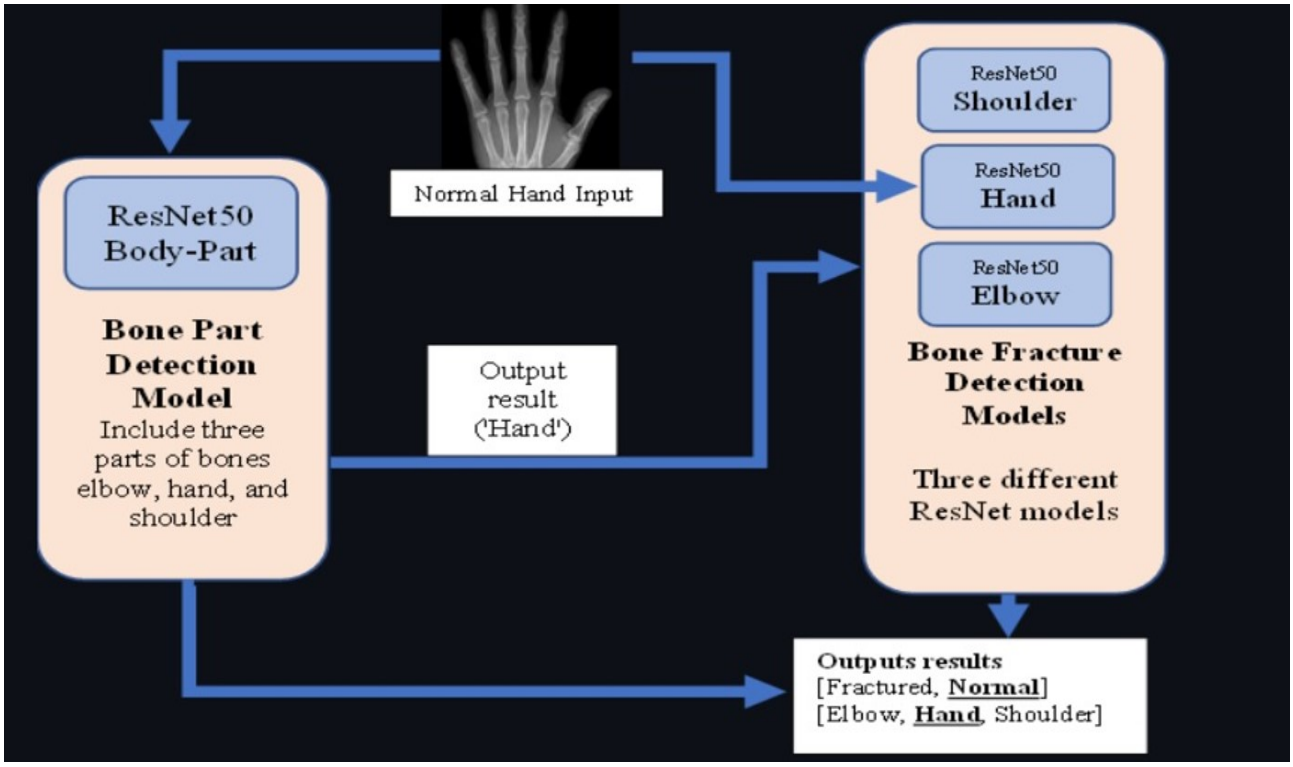

Figure 4.1: **Architecture for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

The detection of organ dysfunction using machine learning through a Convolutional Neural Network (CNN) involves several architectural components as shown in figure 4.1. The CNN model can be trained on electronic medical record data, including heart rate, blood pressure, respiratory rate, and unstructured text data The CNN model can also predict acute organ failure in critically ill patients with sickle cell disease (SCD) up to 6 hours before onset, with an average sensitivity and specificity of 96 and98,respectively.

4.2 Design Phase

4.2.1 Data Flow Diagram

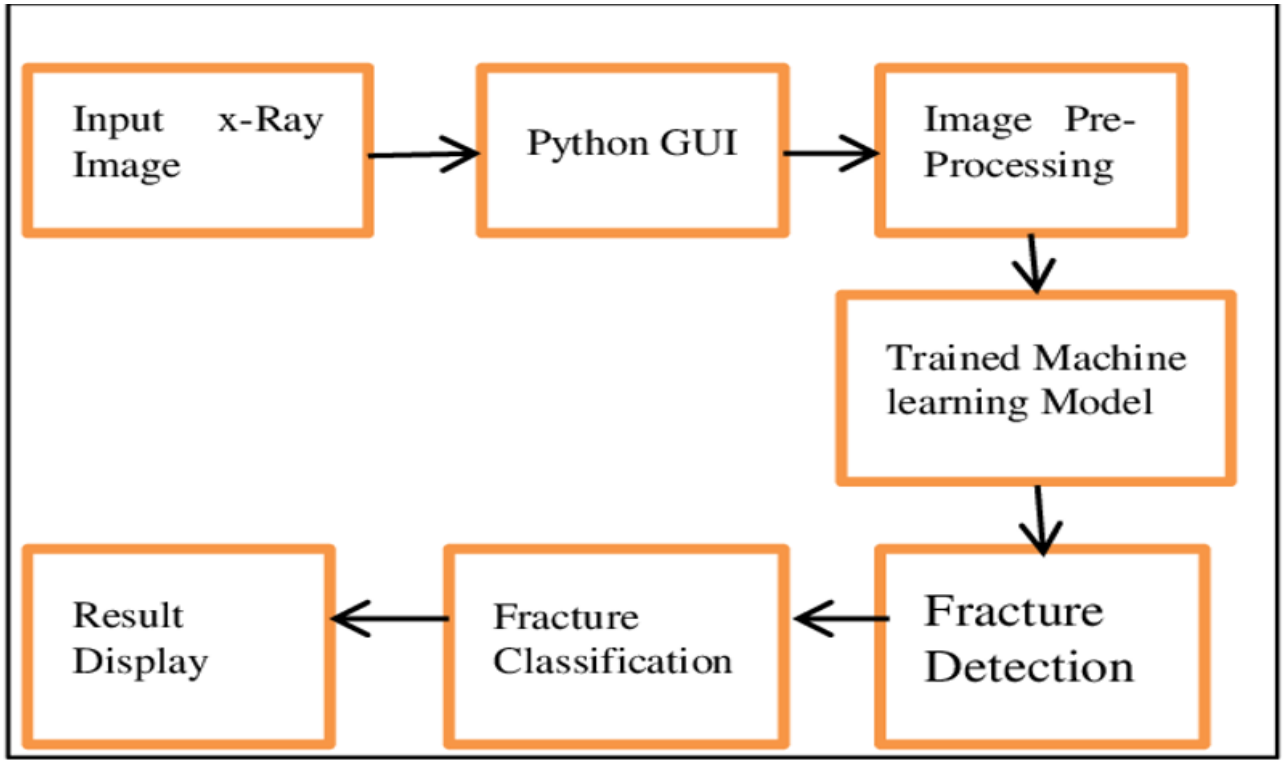

Figure 4.2: **Data Flow Diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

This data flow diagram outlines the sequential flow of steps involved in the bone fracture detection process, starting from the input X-ray image to the final prediction of the fracture status figure 4.2. It illustrates how data moves through different stages of preprocessing, feature extraction, classification, and prediction to detect and classify bone fractures accurately.

4.2.2 Use Case Diagram

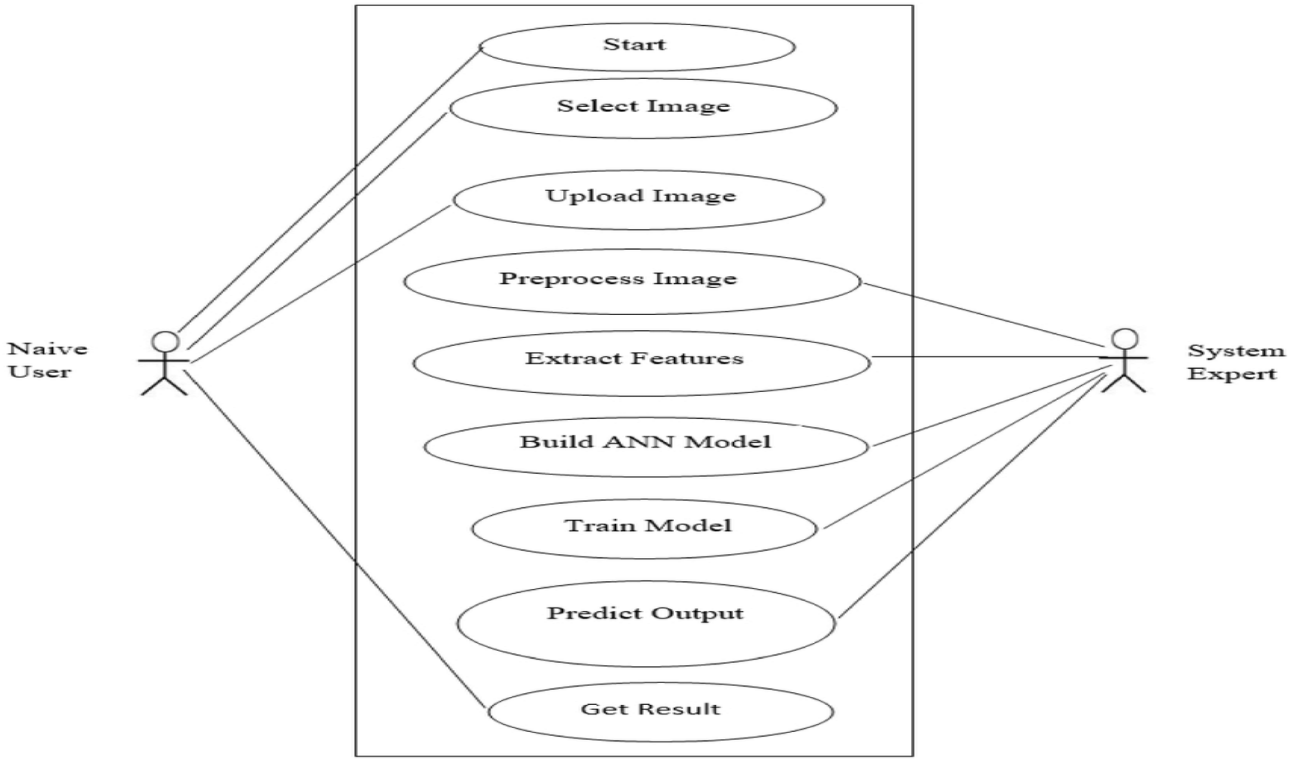

Figure 4.3: **Use Case Diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

This use case diagram illustrates the interactions between patients and healthcare professionals in the bone disease detection system. It includes the use cases of inputting patient information, uploading bone density scan images, receiving risk assessments, and receiving personalized treatment recommendations as shown in figure 4.3. Additionally, healthcare professionals can review patient information, analyze images, evaluate risk, and prescribe treatment plans.

4.2.3 Class Diagram

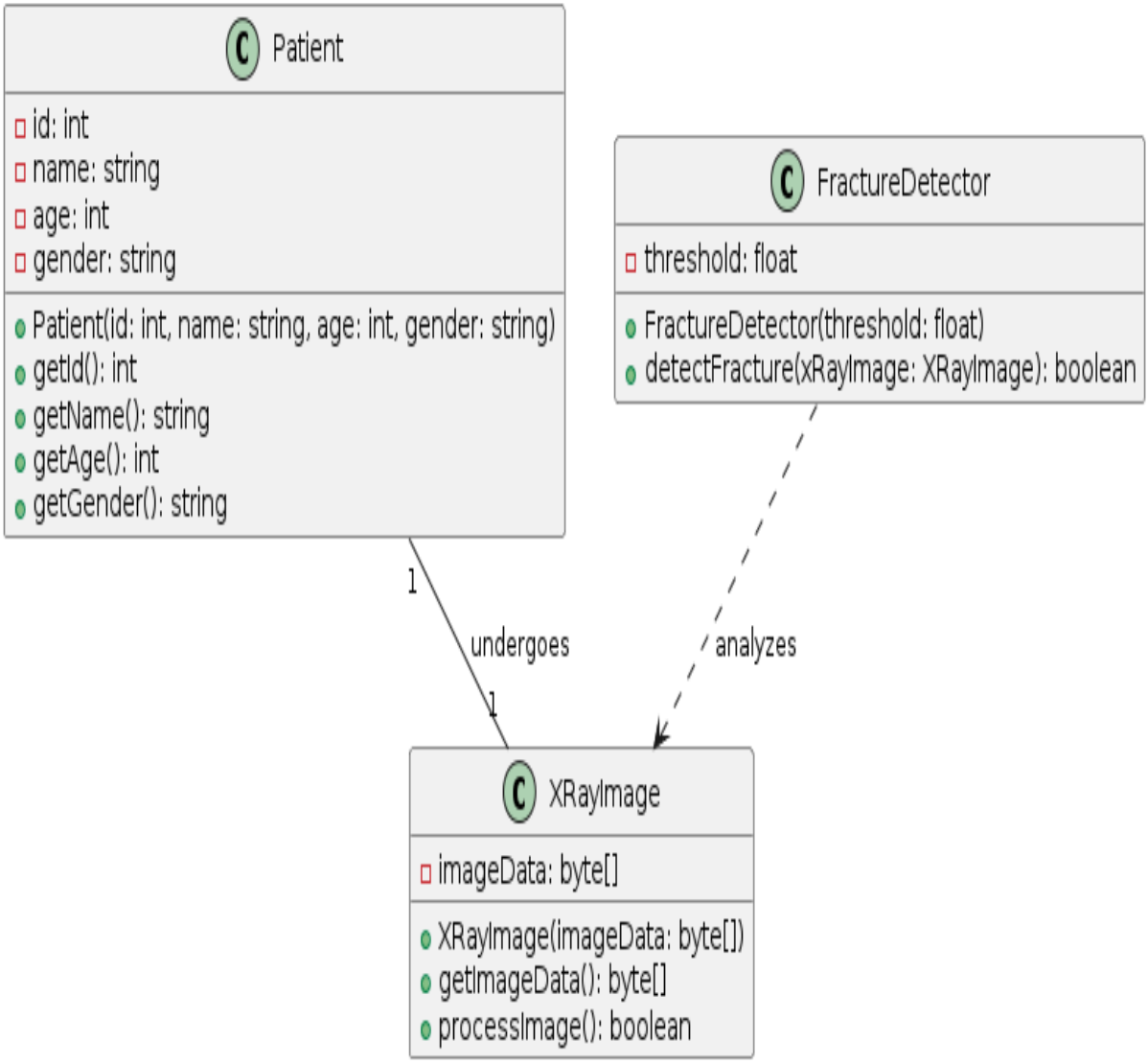

Figure 4.4: **class diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

This class represents the bone scan image data, including the image itself and any associated metadata. This class represents the presence or absence of a fracture in the bone scan image as shown in figure 4.4. This class represents the algorithm used to analyze the bone scan image and detect any abnormalities, including fractures.

4.2.4 Sequence Diagram

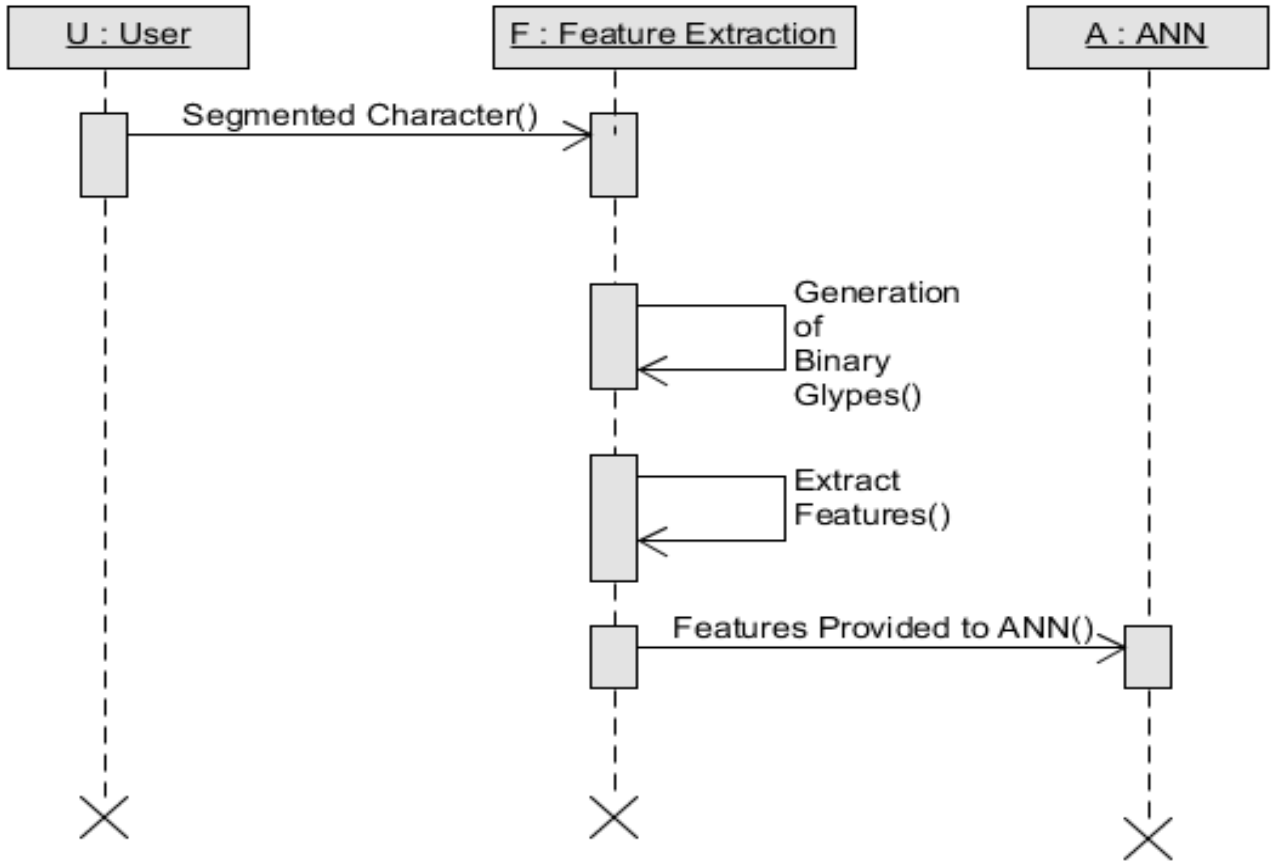

Figure 4.5: **Sequence Diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

The user uploads an X-ray image of a bone through a web interface. The image is preprocessed to remove any artifacts or noise. The preprocessed image is fed into the CNN model for feature extraction. The CNN model extracts features from the image using convolutional and pooling layers. The extracted features are flattened and fed into a fully connected layer for classification as shown in figure 4.5. The fully connected layer outputs a probability distribution over the classes (fractured or not fractured). The class with the highest probability is selected as the final prediction. The prediction is displayed to the user through the web interface.

4.2.5 Collaboration diagram

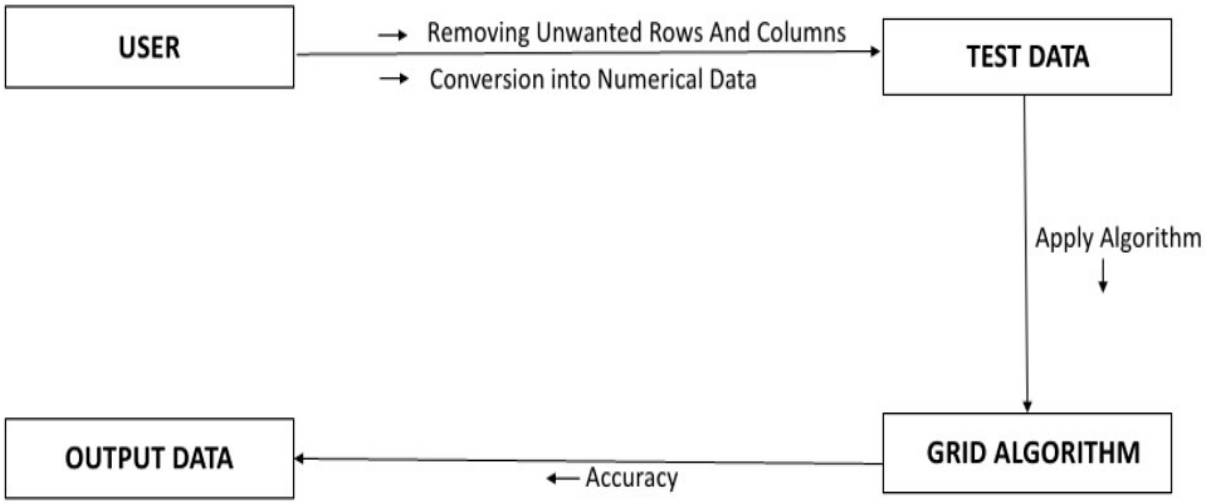

Figure 4.6: **Collaboration diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

This collaboration diagram shows the flow of data and processes involved in the bone disease detection system. It highlights the key steps involved in preprocessing, segmentation, feature extraction, feature reduction, BMD estimation, and classification to diagnose bone diseases accurately as shown in figure 4.6.

4.2.6 Activity Diagram

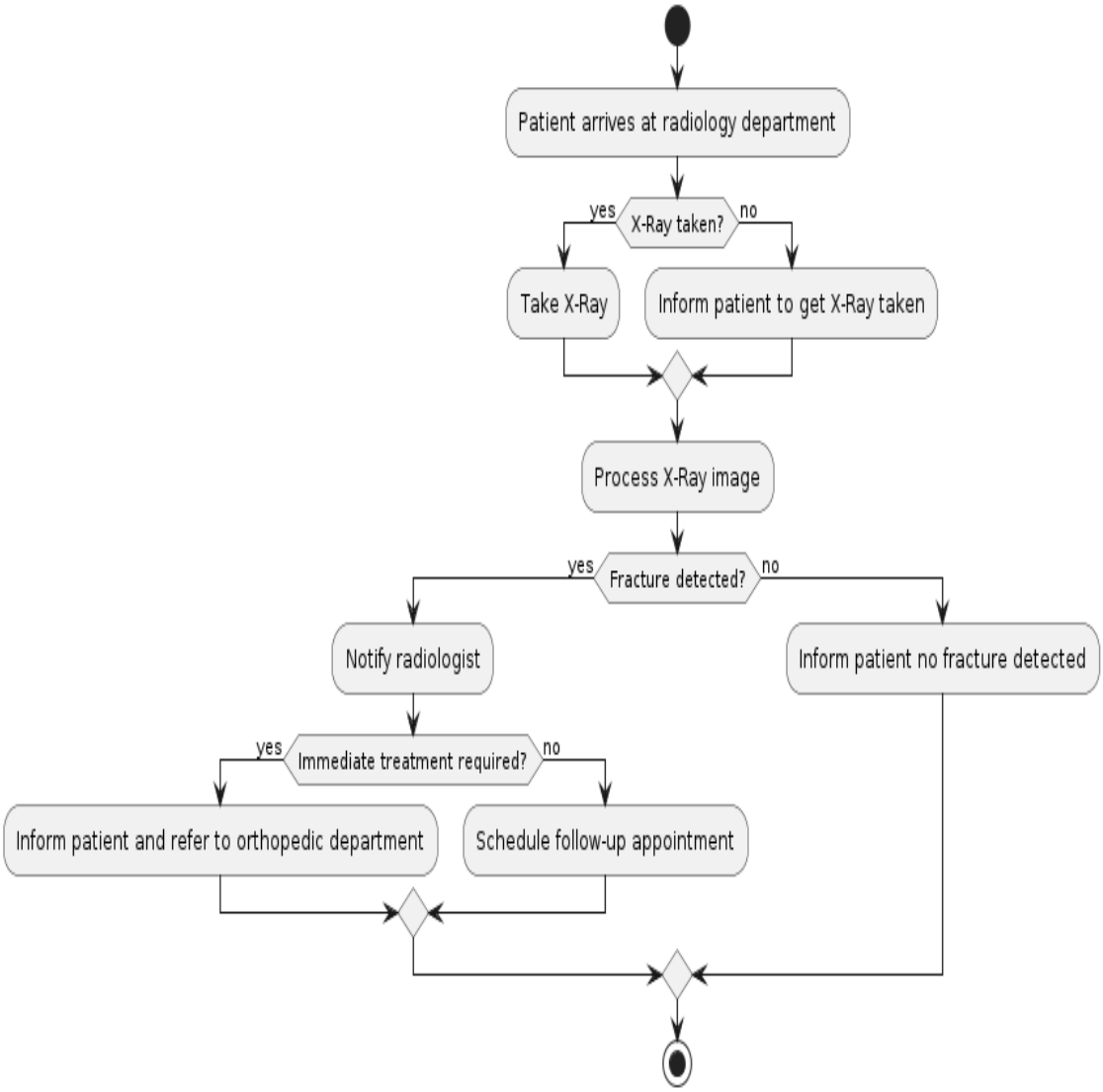

Figure 4.7: **Activity Diagram for X-RAY Analysis Of Palm And Fingers For Fracture Detection**

4.3 Algorithm & Pseudo Code

4.3.1 Convolutional Neural Network Algorithm

Data Preparation: Load the X-ray image dataset containing labeled examples of normal and fractured bones. Preprocess the images by resizing, normalizing pixel values, and augmenting the dataset (optional).

Model Training: Initialize a CNN architecture suitable for bone fracture detection, such as a series of convolutional layers followed by pooling and fully connected layers. Train the CNN model

using the preprocessed X-ray images: Forward pass: Input an image batch to the network and compute the predicted probabilities for each class (normal or fractured). Compute the loss between the predicted probabilities and the ground truth labels.

Backpropagation: Update the network's parameters (weights and biases) using gradient descent or another optimization algorithm to minimize the loss. Repeat the training process for multiple epochs until convergence or a predefined stopping criterion is met.

Model Evaluation: Assess the performance of the trained model on a separate validation or test set: Forward pass: Input the test images to the trained model and compute the predicted probabilities. Calculate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve (AUC) to quantify the model's performance. Visualize the model's predictions and examine misclassified images for further analysis.

Deployment: Deploy the trained CNN model for real-world fracture detection applications: Integrate the model into a user-friendly interface or application that allows healthcare professionals to upload X-ray images and receive automated fracture detection results. Ensure compatibility with existing healthcare IT systems and standards for seamless integration into clinical workflows.

4.3.2 Pseudo Code

```
# Import necessary libraries
import numpy as np
import tensorflow as tf

# Define CNN architecture
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(image_height, image_width,
        num_channels)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
20
21 # Train the model
22 model.fit(train_images, train_labels, epochs=num_epochs, validation_data=(val_images, val_labels))
23
24 # Evaluate the model
25 test_loss, test_acc = model.evaluate(test_images, test_labels)
26 print('Test accuracy:', test_acc)
27
28 # Make predictions
29 predictions = model.predict(test_images)
```

4.4 Module Description

4.4.1 Data collection and preprocessing

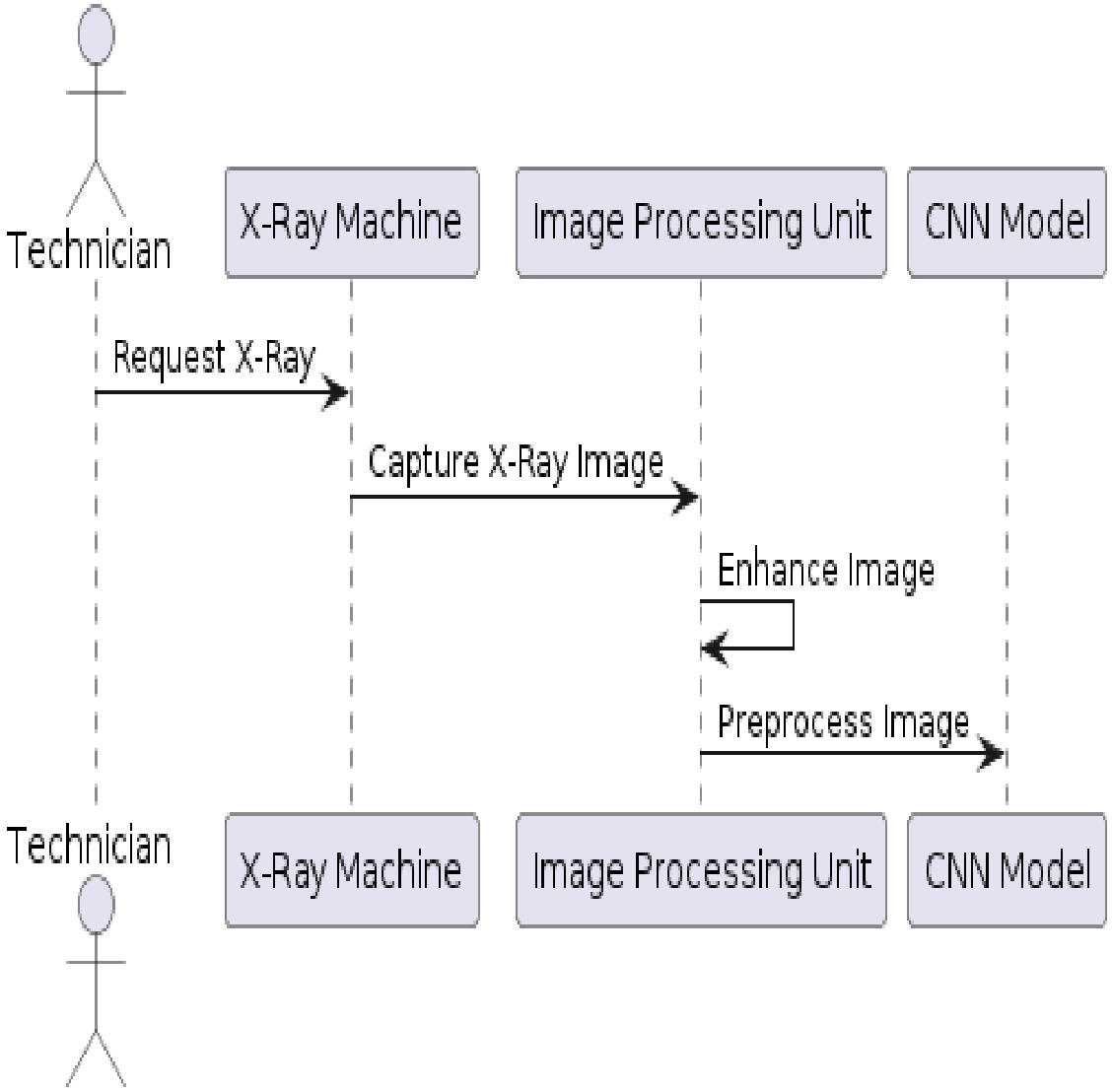

Figure 4.8: **Data Collection and Preprocessing**

Detecting bone fractures via medical imaging necessitates meticulous data collection and preprocessing. Initially, various imaging modalities such as X-rays, CT scans, or MRI scans are employed to gather the necessary data, with X-rays being a common choice due to their accessibility and speed. These images are then meticulously annotated to pinpoint fractures, a task performed manually by radiologists or through automated algorithms. Subsequently, data preprocessing steps are undertaken to refine the dataset. This involves rescaling and normalizing images to a standard resolution and intensity range, reducing noise, enhancing contrast, removing artifacts, and extracting the region of interest (ROI) containing bone structures.

4.4.2 CNN Model Architecture Design and Training

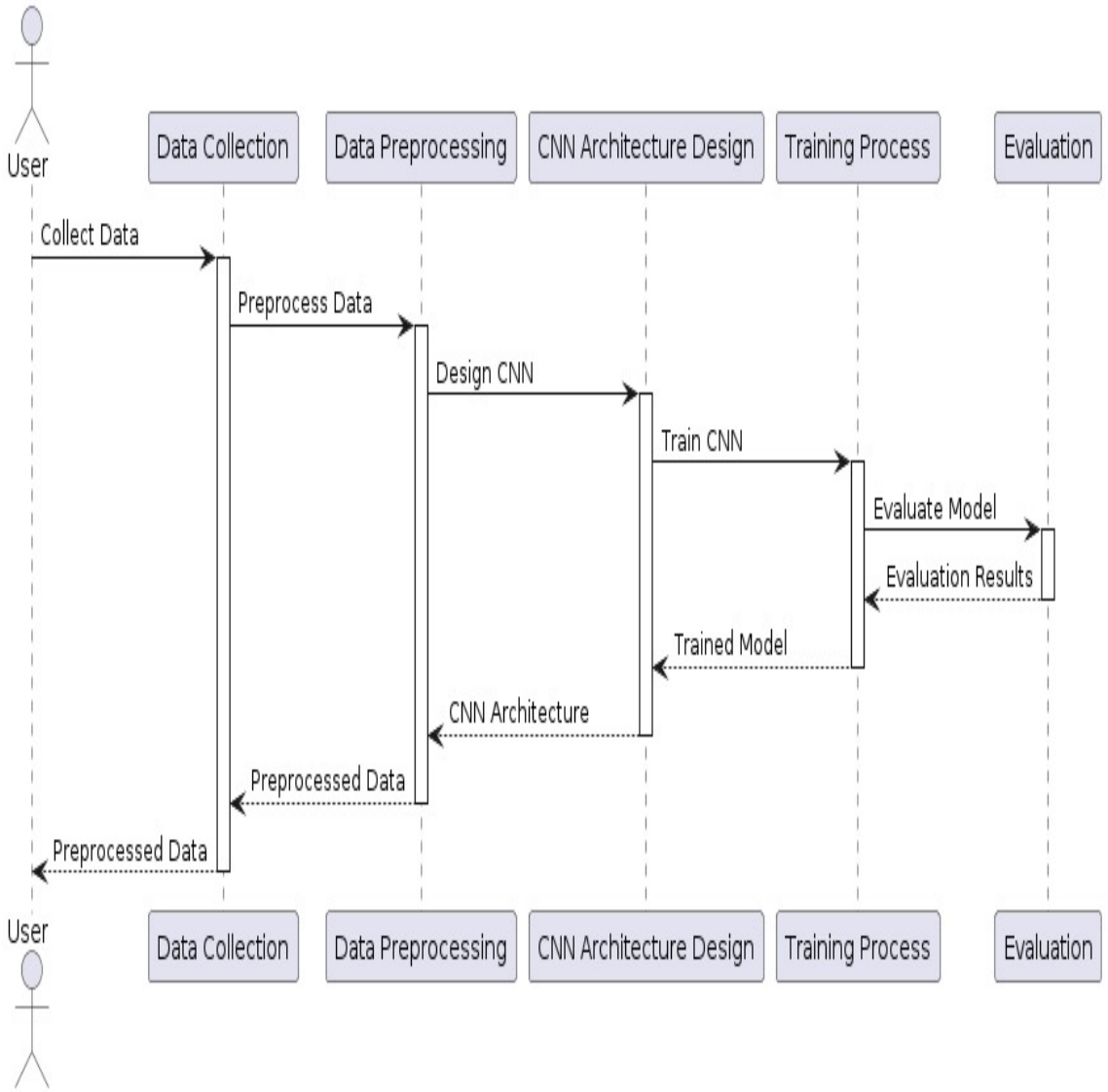

Figure 4.9: **CNN Model Architecture design and Training**

Designing and training a Convolutional Neural Network (CNN) for bone fracture detection involves several key steps. Initially, the dataset of bone images is prepared, organized into training, validation, and testing sets, with potential augmentation to enhance diversity. The choice of CNN architecture is critical, considering factors like image resolution and fracture complexity. Customizing the architecture may involve adding or modifying layers to optimize performance. Training the model entails adjusting hyperparameters and monitoring performance on the validation set to prevent overfitting. Evaluation metrics such as accuracy and F1 score assess the model's effectiveness, guiding further optimization efforts. Once refined, the model can be deployed, integrated into clinical workflows, and continuously monitored for real-world performance. Collaboration among data scientists, clinicians, and experts ensures the model meets clinical standards and contributes mean-

ingfully to patient care. Ongoing research may refine the model further to improve its accuracy and applicability. Collaboration among interdisciplinary teams comprising data scientists, clinicians, and domain experts ensures that the developed CNN model aligns with clinical standards and contributes meaningfully to patient care, potentially leading to advancements in diagnostic accuracy and patient outcomes.

4.4.3 Evaluation and Deployment

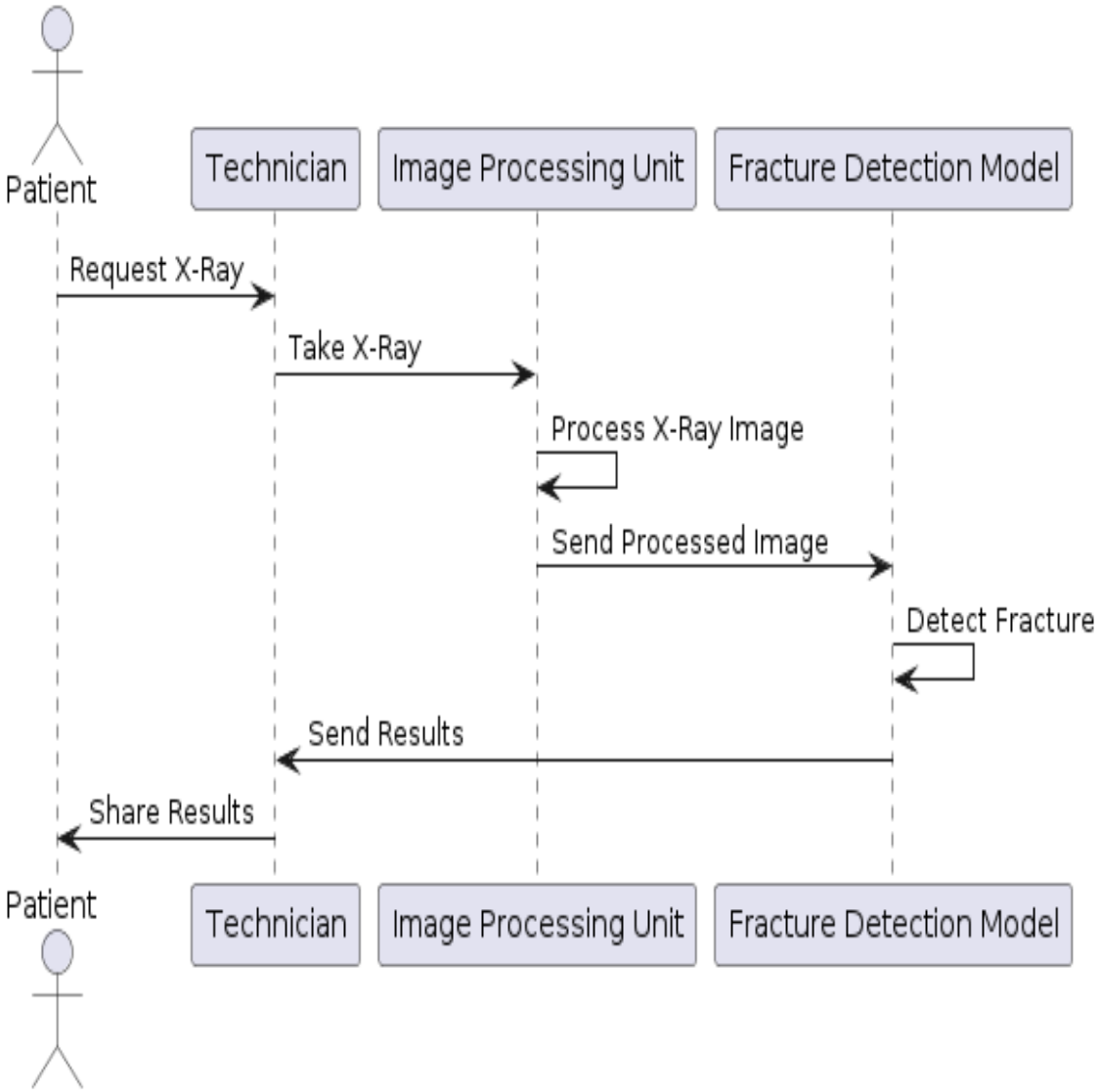

Figure 4.10: **Evalution and Deployment**

Evaluate the trained model's performance on a separate validation or test set to assess its accuracy and generalization ability. Compute evaluation metrics such as accuracy, precision, recall, and F1-score to quantify the model's performance. Once the model achieves satisfactory performance, deploy it for real-world use in healthcare settings, allowing healthcare professionals to upload X-ray images

and receive automated fracture detection results.

4.5 Steps to execute/run/implement the project

4.5.1 Planning and preparation:

Define Objectives and Requirements: Clearly define the objectives of the fracture detection system, such as improving diagnostic accuracy or speeding up the interpretation process. Identify the specific requirements, including target accuracy, processing speed, and integration with existing healthcare systems.

Gather Data: Collect a diverse dataset of bone images containing both fractured and non-fractured samples. Ensure the dataset is annotated with accurate labels indicating the presence and location of fractures, either through manual annotation by experts or using automated algorithms.

Select Imaging Modalities: Choose the appropriate imaging modalities for fracture detection, considering factors such as availability, cost, and diagnostic accuracy. Common modalities include X-rays, CT scans, and MRI scans, with X-rays being the most commonly used due to their wide availability and relatively low cost. .

4.5.2 Development and implementation:

Define Objectives and Scope: Clearly define the objectives of the fracture detection system, such as improving diagnostic accuracy, reducing interpretation time, or enhancing patient outcomes. Determine the scope of the project, including the types of fractures to be detected and the target patient population.

Gather Resources: Allocate human, financial, and technological resources necessary for the project. Ensure access to relevant expertise, including radiologists, data scientists, software engineers, and domain specialists.

Data Preprocessing: Preprocess the acquired data to enhance quality and standardize format. Perform tasks such as resizing images, normalizing pixel intensity, and removing noise to improve the consistency of the dataset.

4.5.3 Deployment and iteration:

Deployment Planning: Develop a deployment plan outlining the steps, timeline, and responsibilities for deploying the fracture detection system in clinical settings. Identify key stakeholders, including

healthcare providers, IT personnel, and administrators, and communicate deployment objectives and requirements. It analysis the result of image you uplode.

Integration with Clinical Workflow: Integrate the fracture detection system seamlessly into existing clinical workflows and medical imaging platforms to minimize disruption. Ensure compatibility with electronic health record (EHR) systems, picture archiving and communication systems (PACS), and other healthcare IT infrastructure.

User Training and Support: Provide comprehensive training sessions for healthcare professionals on using the fracture detection system effectively, including how to interpret results and integrate them into patient care workflows. Offer ongoing technical support and troubleshooting resources to address any issues that arise during deployment and use.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

Input design for bone fracture detection involves capturing data from medical images, such as X-rays or CT scans, to identify and diagnose bone fractures. This process typically involves pre-processing the images to remove noise, normalize the pixel values, and segment the bone from the background. The segmented bone image is then analyzed using various algorithms, such as edge detection or machine learning models, to identify any fractures or abnormalities.

5.1.2 Output Design

Output design for bone fracture detection involves presenting the results of the analysis in a clear and concise manner to the user. This may include displaying the original image with the identified fractures overlaid, generating a report with the diagnosis and recommended treatment, or providing a visual representation of the fracture pattern. The output design should be easy to understand and interpret, allowing healthcare professionals to make informed decisions about patient care.

5.2 Testing

Testing is the process of executing a program or part of a program with the in- tention of finding errors. The different phases of a test life cycle are Test Planning and Control, Test Analysis and Design, Test Implementation and Execution, Evalu- ating Exit Criteria and Reporting, and Test Closure.

5.3 Types of Testing

5.3.1 Unit testing

Testing the actual model: Instead of mocking dependencies like APIs, it's essential to test the actual model in machine learning. This allows for checks on the model's behavior, such as ensuring that loss

decreases with each batch and the model can overfit before wasting compute on an unpromising run.

Data validation and configuration management: Tools like Flask, Pydantic, or Fast API can help minimize issues due to data and settings inconsistencies.

Input

```
def analyze_dataset(file_path):
    # Load and preprocess the image
    img_array = preprocess_image(file_path)

    # Apply data augmentation
    augmented_images = []
    for batch in datagen.flow(img_array, batch_size=1):
        augmented_images.append(batch)
        if len(augmented_images) >= 5:  # Generate 5 augmented images
            break

    predictions = []
    for img in augmented_images:
        features = base_model.predict(img)
        features = np.reshape(features, (1, 7 * 7 * 2048))
        prediction = model.predict(features)
        predictions.append(prediction[0])

    avg_prediction = np.mean(predictions)

    if avg_prediction >= 0.5:
        result = ['The image is affected by pneumonia.']
    else:
        result = ['The image is not affected by pneumonia.']

    return result
```

Test result

5.3.2 Integration testing

Integrated testing in the context of machine learning (ML) involves testing the combined functionality of individual components, ensuring that modules interact correctly and that system and model standards are met.

In integrated testing, you can use the Arrange Act Assert methodology to set up different inputs to test on, execute the test, and assert on the expected outputs. For ML models, you can test the training, evaluation, inference, and deployment phases.

Input

```
import os
import uuid
import numpy as np
import tensorflow as tf
from keras.preprocessing import image
from flask import Flask, render_template, request

# load the models when import "predictions.py"
model_elbow_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_BodyParts.h5")
model_hand_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture\weights/ResNet50_Hand_frac.h5")
model_shoulder_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_Shoulder_frac.h5")
model_parts = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_BodyParts.h5")

# categories for each result by index

#   0-Elbow      1-Hand       2-Shoulder
categories_parts = ["Elbow", "PALM", "Shoulder"]

#   0-fractured      1-normal
categories_fracture = ['fractured', 'normal']


def predict(img, model="Parts"):
    # handle invalid model names
    if model not in ["Parts", "Elbow", "Hand", "Shoulder"]:
        raise ValueError(f"Invalid model name: {model}")

    size = 224

    # load image with 224px224p (the training model image size, rgb)
    temp_img = image.load_img(img, target_size=(size, size))
    x = image.img_to_array(temp_img)
    x = np.expand_dims(x, axis=0)
    images = np.vstack([x])

    # load the appropriate model based on the input
    if model == 'Parts':
        chosen_model = model_parts
    elif model == 'Elbow':
        chosen_model = model_elbow_frac
    elif model == 'Hand':
        chosen_model = model_hand_frac
    elif model == 'Shoulder':
        chosen_model = model_shoulder_frac
```

```
45
46      prediction = np.argmax(chosen_model.predict(images), axis=1)
47
48      # choose the category and get the string prediction
49      if model != "Parts":
50          prediction_str = categories_fracture[prediction.item()]
51      else:
52          prediction_str = categories_parts[prediction.item()]
53
54      return prediction_str
55
56
57  app = Flask(__name__)
58
59
60  @app.route('/')
61  def index():
62      return render_template('index.html')
63
64
65  @app.route('/predict', methods=['POST'])
66  def make_prediction():
67      if request.method == 'POST':
68          file = request.files['file']
69          # Generate a unique filename using uuid
70          unique_filename = str(uuid.uuid4()) + os.path.splitext(file.filename)[-1]
71          filename = "static/" + unique_filename
72          print(filename, file)
73
74          try:
75              # Save the file
76              file.save(filename)
77              model = request.form['model']
78              print(model)
79              prediction = predict(filename, model)
80              return render_template('index.html', prediction=prediction, image_file=filename)
81          except Exception as e:
82              return render_template('index.html', error="Error processing image. Please upload a
                    valid image file.",
83                                     image_file=None)
84
85
86  if __name__ == '__main__':
87      app.run(debug=True)
```

Test result

5.3.3 System testing

System tests in ML focus on testing the design of a system for expected outputs given inputs, encompassing training, inference, and deployment stages

Testing machine learning systems presents unique challenges compared to traditional software testing due to the learned logic in ML models. Solutions involve a combination of pre-train, post-train, and evaluation checks to ensure system reliability and performance.

Automated testing is crucial in continuous integration/continuous deployment (CI/CD) to ensure the correctness of pipelines with minimal manual intervention.

Input

```
def analyze_dataset(file_path):
    # Load and preprocess the image
    img_array = preprocess_image(file_path)

    # Apply data augmentation
    augmented_images = []
    for batch in datagen.flow(img_array, batch_size=1):
        augmented_images.append(batch)
        if len(augmented_images) >= 5:  # Generate 5 augmented images
            break

    predictions = []
    for img in augmented_images:
        features = base_model.predict(img)
        features = np.reshape(features, (1, 7 * 7 * 2048))
        prediction = model.predict(features)
        predictions.append(prediction[0])

    avg_prediction = np.mean(predictions)

    if avg_prediction >= 0.5:
        result = ['The image is affected by pneumonia.']
    else:
        result = ['The image is not affected by pneumonia.']

    return result
```

5.3.4 Test Result

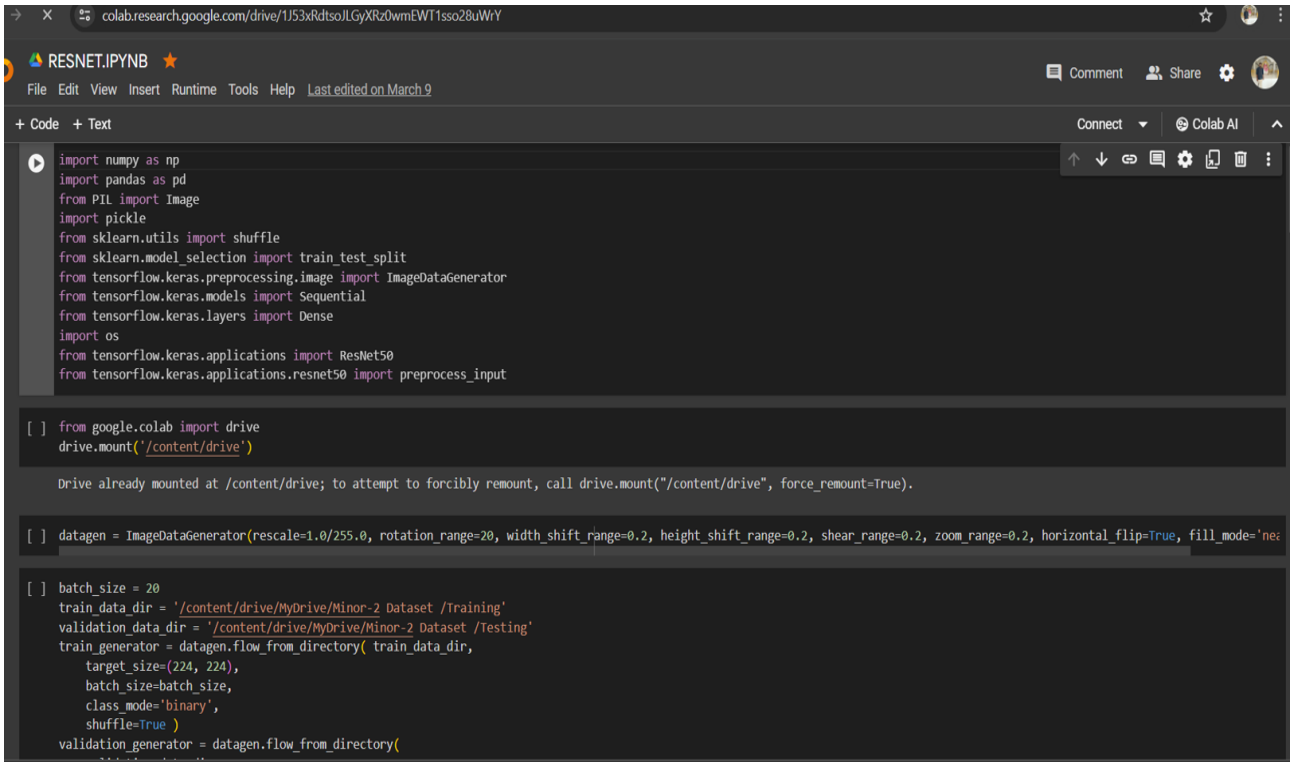

Figure 5.1: **Resnet Test**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The efficiency of the proposed attendance automation system with image recognition technology is multifaceted, offering benefits in accuracy, time-saving, resource optimization, and scalability.

Accuracy: By leveraging advanced image recognition algorithms, the system ensures high accuracy in identifying individuals and recording their attendance. This minimizes errors associated with manual data entry, leading to more reliable attendance records and reducing the risk of payroll discrepancies or compliance issues.

Time-saving: Automation streamlines the attendance tracking process, eliminating the need for manual entry or physical sign-in sheets. Employees or students can quickly clock in or out using facial recognition, saving time for both individuals and administrators. Moreover, real-time monitoring capabilities provide instant access to attendance data, enabling prompt decision-making and responsiveness to attendance-related issues.

Resource Optimization: The system reduces the administrative burden associated with manual attendance tracking, freeing up personnel to focus on more strategic tasks. Additionally, by automating repetitive tasks, the system increases operational efficiency and productivity, optimizing resource utilization within the organization.

Scalability: The proposed system is scalable, capable of accommodating growing attendance needs without significant additional investments. Whether deployed in a small classroom or a large corporate environment, the system can scale to handle varying attendance volumes efficiently.

6.2 Comparison of Existing and Proposed System

Existing System (Decision Tree): Decision trees are simple, interpretable models that make decisions based on a series of sequential splits in the data. While decision trees are easy to understand and implement, they may suffer from overfitting, especially when dealing with complex datasets or noisy data. Additionally, decision trees may not generalize well to unseen data, leading to lower predictive accuracy compared to more advanced algorithms.

Proposed System (Random Forest Algorithm): The random forest algorithm is an ensemble learning method that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. By aggregating the predictions of multiple trees, random forests can handle complex datasets more effectively and provide better generalization performance. Moreover, random forests are robust to outliers and noise in the data, making them suitable for a wide range of applications.

Faster CNN Algorithm: Convolutional Neural Networks (CNNs) are a class of deep learning models commonly used for image recognition tasks. While CNNs are powerful and highly accurate, they can be computationally intensive, especially when dealing with large datasets or complex architectures. To address this challenge, researchers have developed faster CNN algorithms that optimize model architecture, training procedures, and computational efficiency without sacrificing accuracy. These algorithms may employ techniques such as model pruning, quantization, or specialized hardware acceleration to speed up inference and reduce computational overhead.

6.3 Sample Code

```
def analyze_dataset(file_path):
    # Load and preprocess the image
    img_array = preprocess_image(file_path)

    # Apply data augmentation
    augmented_images = []
    for batch in datagen.flow(img_array, batch_size=1):
        augmented_images.append(batch)
        if len(augmented_images) >= 5:  # Generate 5 augmented images
            break

    predictions = []
    for img in augmented_images:
        features = base_model.predict(img)
        features = np.reshape(features, (1, 7 * 7 * 2048))
        prediction = model.predict(features)
        predictions.append(prediction[0])

    avg_prediction = np.mean(predictions)

    if avg_prediction >= 0.5:
        result = ['The image is affected by pneumonia.']
    else:
        result = ['The image is not affected by pneumonia.']

    return result
```

Output

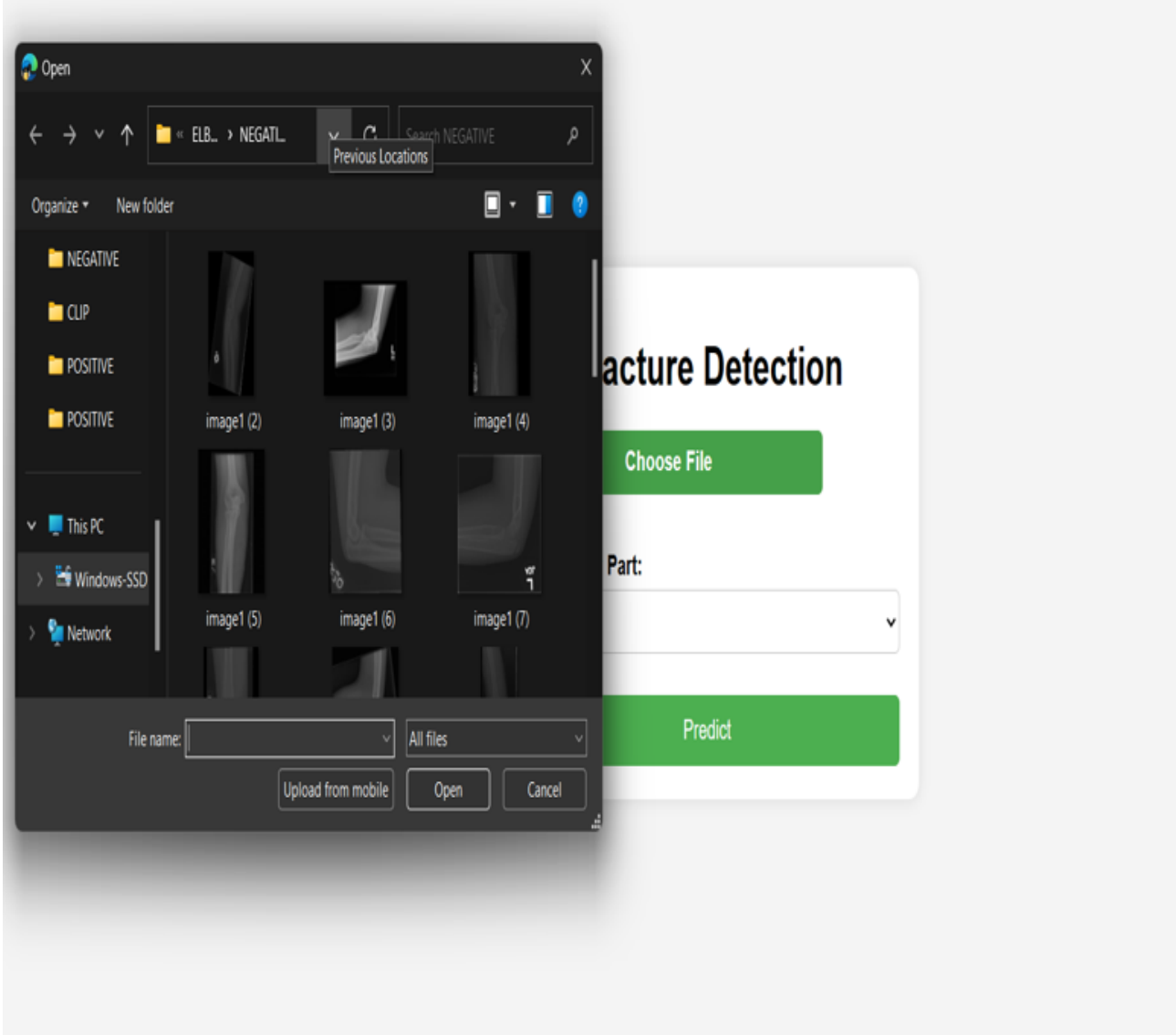

Figure 6.1: **Choosing a File**

Figure 6.2: **Prediction**

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

The main agenda of the project to save the during the time of the critical situation. The detection of organ dysfunction using machine learning through Convolutional Neural Networks (CNNs) has the capacity to study the x-ray and gives the most accurate results by this algorithm and the accuracy and timeliness of organ dysfunction detection and AKI prediction, leading to improved patient outcomes and reduced healthcare costs. CNNs are known for their robustness in handling complex data structures, making them suitable for detecting organ dysfunction in various medical scenarios. By this we can conclude Detection of Orgon Dysfunction using ML through CNNs have demonstrated superior performance in detecting organ dysfunction compared to traditional methods.

7.2 Future Enhancements

Diagnosing a child X-ray, complex osteoporosis and bone tumour case remains as an intellectual challenge for the radiologist. It is difficult to analyse when the situation is complex and machines can assist a radiologist in proper diagnosis and can draw their attention to an ignored or complex case. For example, in the case of osteoporosis, machines can predict that this condition can lead to fracture in the future and precautions are to be taken. Another case is the paediatric x-rays, the bones of a child in the growing stage till the age of 15–18 years. This is very well understood by radiologists, but a machine can predict it as an abnormality. In this situation, radiologist expertise is required to take the decision whether the case is normal or not. Therefore, we can say that there is no point that DL will replace the radiologist, but they both can go hand in hand and support the overall decision system.

Chapter 8

PLAGIARISM REPORT

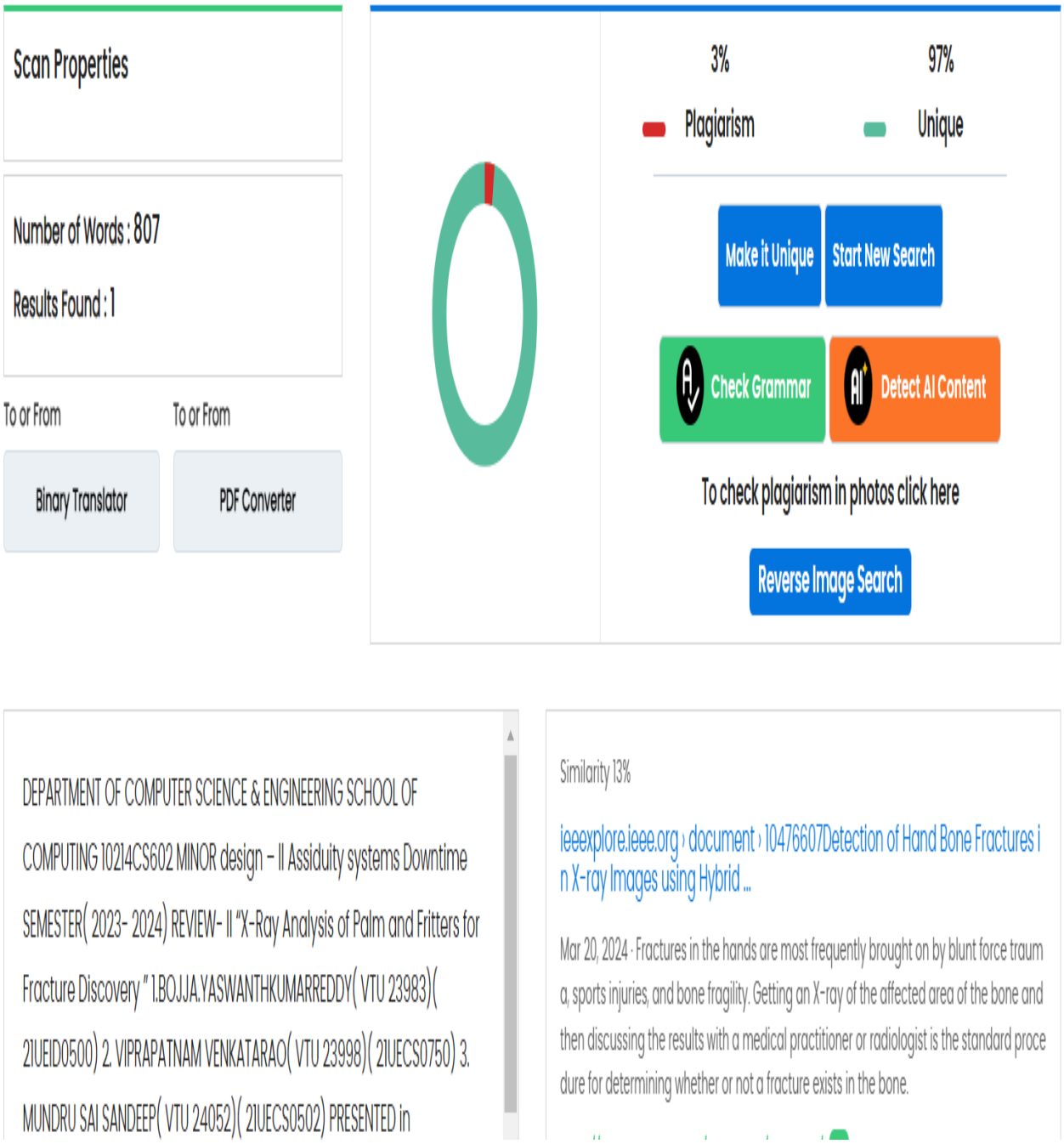

Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
import os
import uuid
import numpy as np
import tensorflow as tf
from keras.preprocessing import image
from flask import Flask, render_template, request

# load the models when import "predictions.py"
model_elbow_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_BodyParts.h5")
model_hand_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture\weights/ResNet50_Hand_frac.h5")
model_shoulder_frac = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_Shoulder_frac.h5")
model_parts = tf.keras.models.load_model("D:/Users/venky/OneDrive/Desktop/Bone Fracture/Bone Fracture/weights/ResNet50_BodyParts.h5")

# categories for each result by index

#   0-Elbow      1-Hand       2-Shoulder
categories_parts = ["Elbow", "PALM", "Shoulder"]

#   0-fractured      1-normal
categories_fracture = ['fractured', 'normal']


def predict(img, model="Parts"):
    # handle invalid model names
    if model not in ["Parts", "Elbow", "Hand", "Shoulder"]:
        raise ValueError(f"Invalid model name: {model}")

    size = 224

    # load image with 224px224p (the training model image size, rgb)
    temp_img = image.load_img(img, target_size=(size, size))
```

```
    x = image.img_to_array(temp_img)
    x = np.expand_dims(x, axis=0)
    images = np.vstack([x])

    # load the appropriate model based on the input
    if model == 'Parts':
        chosen_model = model_parts
    elif model == 'Elbow':
        chosen_model = model_elbow_frac
    elif model == 'Hand':
        chosen_model = model_hand_frac
    elif model == 'Shoulder':
        chosen_model = model_shoulder_frac

    prediction = np.argmax(chosen_model.predict(images), axis=1)

    # choose the category and get the string prediction
    if model != "Parts":
        prediction_str = categories_fracture[prediction.item()]
    else:
        prediction_str = categories_parts[prediction.item()]

    return prediction_str


app = Flask(__name__)


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/predict', methods=['POST'])
def make_prediction():
    if request.method == 'POST':
        file = request.files['file']
        # Generate a unique filename using uuid
        unique_filename = str(uuid.uuid4()) + os.path.splitext(file.filename)[-1]
        filename = "static/" + unique_filename
        print(filename, file)

        try:
            # Save the file
            file.save(filename)
            model = request.form['model']
            print(model)
            prediction = predict(filename, model)
            return render_template('index.html', prediction=prediction, image_file=filename)
        except Exception as e:
```

```
            return render_template('index.html', error="Error processing image. Please upload a
                valid image file.",
                                   image_file=None)


if __name__ == '__main__':
    app.run(debug=True)


----prediction.py


import numpy as np
import tensorflow as tf
from keras.preprocessing import image

# load the models when import "predictions.py"
model_elbow_frac = tf.keras.models.load_model("weights/ResNet50_Elbow_frac.h5")
model_hand_frac = tf.keras.models.load_model("weights/ResNet50_Hand_frac.h5")
model_shoulder_frac = tf.keras.models.load_model("weights/ResNet50_Shoulder_frac.h5")
model_parts = tf.keras.models.load_model("weights/ResNet50_BodyParts.h5")

# categories for each result by index

#   0-Elbow     1-Hand      2-Shoulder
categories_parts = ["Elbow", "PALM", "Shoulder"]

#   0-fractured     1-normal
categories_fracture = ['fractured', 'normal']


# get image and model name, the default model is "Parts"
# Parts - bone type predict model of 3 classes
# otherwise - fracture predict for each part
def predict(img, model="Parts"):
    size = 224
    if model == 'Parts':
        chosen_model = model_parts
    elif model == 'Elbow':
        chosen_model = model_elbow_frac
    elif model == 'Hand':
        chosen_model = model_hand_frac
    elif model == 'Shoulder':
        chosen_model = model_shoulder_frac

    # load image with 224px224p (the training model image size, rgb)
    temp_img = image.load_img(img, target_size=(size, size))
    x = image.img_to_array(temp_img)
    x = np.expand_dims(x, axis=0)
    images = np.vstack([x])
```

```
    prediction = np.argmax(chosen_model.predict(images), axis=1)

    # chose the category and get the string prediction
    if model == 'Parts':
        prediction_str = categories_parts[prediction.item()]
    else:
        print("hello")
        prediction_str = categories_fracture[prediction.item()]

    return prediction_str


--html file


<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fracture Detection</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }
        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            max-width: 400px;
            width: 100;
        }
        h1 {
            text-align: center;
            margin-bottom: 20px;
        }
        label {
            display: block;
            font-weight: bold;
            margin-bottom: 5px;
        }
        input[type="file"] {
```

```
            width: 0;
            height: 0;
            opacity: 0;
            overflow: hidden;
            position: absolute;
            z-index: -1;
        }
        .file-upload-wrapper {
            position: relative;
            margin-bottom: 15px;
        }
        .file-upload-button {
            background-color: #4CAF50;
            color: white;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            width: 70;
            display: block;
            text-align: center;
        }
        .file-upload-button:hover {
            background-color: #45a049;
        }
        .file-name {
            margin-top: 5px;
            overflow: hidden;
            white-space: nowrap;
            text-overflow: ellipsis;
        }
        input[type="submit"] {
            background-color: #4CAF50;
            color: white;
            padding: 12px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            width: 100;
            font-size: 16px;
            margin-top: 10px;
        }
        input[type="submit"]:hover {
            background-color: #45a049;
        }
        select {
            width: 100;
            padding: 10px;
            margin-bottom: 15px;
            border-radius: 5px;
```

```
            border: 1px solid #ccc;
            box-sizing: border-box;
        }
        img {
            display: block;
            margin: 0 auto;
            max-width: 100;
            height: auto;
            margin-top: 20px;
        }
    </style>
</head>
<body>
    <form action="/predict" method="post" enctype="multipart/form-data">
        <h1>Fracture Detection</h1>
        <div class="file-upload-wrapper">
            <label class="file-upload-button" for="file">Choose File</label>
            <input type="file" id="file" name="file">
        </div>
        <span class="file-name"></span>
        <br>
        <label for="model">Select Body Part:</label>
        <select id="model" name="model">
            <option value="Parts">Parts</option>
            <option value="Elbow">Elbow</option>
            <option value="Hand">Hand</option>
            <option value="Shoulder">Shoulder</option>
        </select>
        <input type="submit" value="Predict">
    </form>
    { if prediction }
    <h2>Prediction: {{ prediction }}</h2>
    <img src="{{ image_file }}" alt="Uploaded Image">
    { endif }
    <script>
        document.getElementById('file').addEventListener('change', function() {
            var fileName = this.value.split('\\').pop();
            var fileSpan = document.querySelector('.file-name');
            fileSpan.textContent = fileName;
        });
    </script>
</body>
</html>
```

9.2 Poster Presentation

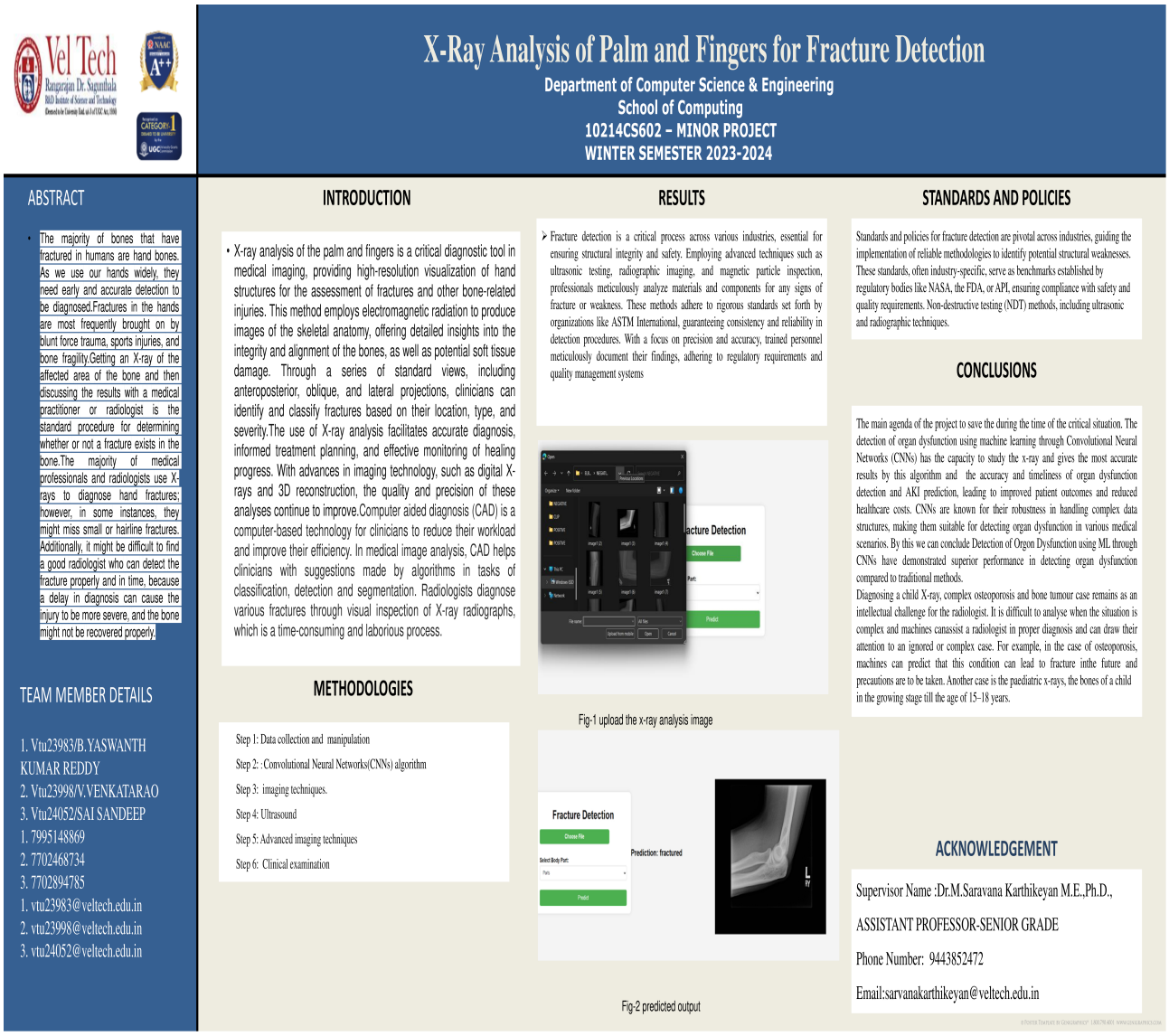

Figure 9.1: **Poster Presentation**

References

[1] Y Yahya Sugandi, Indah Soesanti, Hanung Adi Nugroho, " A Systematic Literature Review of Convolutional Neural Network Architecture for Lung Disease Detection", 2023 6th International Conference on Information and Communications Technology (ICOIACT), pp.230-235, 2023.

[2] Sujay Kumar Mandal, Muralidhar Pullakandam, Rama Muni Reddy Yanamala, "Pneumonia Detection Using Transfer Learning And Hardware Implementation in Edge TPU", 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp.1-5, 2023.

[3] Sashikanta Prusty, Srikanta Patnaik, Sujit Kumar Dash, "ResNet50V2: A Transfer Learning Model to Predict Pneumonia with chest X-ray images", 2022 International Conference on Machine Learning, Computer Systems and Security (MLCSS), pp.208-213, 2022.

[4] Chamodi Fernando, Shammi Kolonne, Hashara Kumarasinghe, Dulani Meedeniya, "Chest Radiographs Classification Using Multi-model Deep Learning: A Comparative Study", 2022 2nd International Conference on Advanced Research in Computing (ICARC), pp.165-170, 2022.

[5] Kim HE, Cosa-Linan A, Santhanam N, Jannesari M, Maros ME, Ganslandt T (2022) Transfer learning for medical image classification: a literature review. BMC Med Imaging 22:69

[6] Sarvamangala DR, Kulkarni RV (2022) Convolutional neural networks in medical image understanding: a survey. Evol Intell 15:1–22

[7] M. Lotfy, R. M. Shubair, N. Navab, and S. Albarqouni, "Investigation of Focal Loss in Deep Learning Models for Femur Fractures Classification," 2019 Int. Conf. Electr. Comput. Technol. Appl. ICECTA 2019, pp. 2–5, 2019, doi: 10.1109/ICECTA48151.2019.8959770.

[8] Z. Wu, X. Mo, H. Zhou, L. Liu, and J. Li, “Classification of reservoir fracture development level by convolution neural network algorithm,” ICNC-FSKD 2018 - 14th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov., pp. 243–250, 2018, doi: 10.1109/FSKD.2018.8687232.

[9] V. L. F. Lum, W. K. Leow, Y. Chen, T. Sen Howe, and M. A. Png, “Combining classifiers for bone fracture detection in x-ray images,” Proc. - Int. Conf. Image Process. ICIP, vol. 1, pp. 1149–1152, 2005, doi: 10.1109/ICIP.2005.1529959.

[10] L. Wang, H. Cheng, H. Lan, Y. Zheng, and K. Li, “Automatic recognition of pertrochanteric bone fractures in femur using level sets,” Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, vol. 2016-October, pp. 3851–3854, 2016, doi: 10.1109/EMBC.2016.7591568.

[11] R. S. Prihatini, A. H. Setyaningrum, and I. M. Shofi, “Texture analysis and fracture identification of lower extremity bones X-ray images,” Int. Conf. Electr. Eng. Comput. Sci. Informatics, vol. 2017-December, no. September, pp. 19–21, 2017, doi: 10.1109/EECSI.2017.8239113.

[12] R. Bagaria, S. Wadhwani, and A. K. Wadhwani, “Different techniques for identification of a bone fracture in analysis of medical image,” Proc. - 2020 IEEE 9th Int. Conf. Commun. Syst. Netw. Technol. CSNT 2020, pp. 327–332, 2020, doi: 10.1109/CSNT48778.2020.9115760.