BONE FRACTURE DETECTION USING MACHINE LEARNING

A main Project Report submitted in the partial fulfilment of the

requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Badri Venkata Durga Praveen	**(20471A05D7)**
Elchuri Rushi Naga Manikanta	**(20471A05E7)**
Thota Sumanth	**(20471A05J0)**

Under the esteemed guidance of

Dr. D. Venkata Reddy, M.Tech., (Ph.D.,)

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Cycle-I
NIRF rank in the band of 251-320 and an ISO 9001:2015 Certified
Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada
KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2023-2024

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name "Bone Fracture Detection using Machine Learning" is a bonafide work done by the team Badri Venkata Durga Praveen (20471A05D7), Elchuri Rushi naga Manikanta (20471A05E7), Thota Sumanth (20471A05J0) in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in the Department of **COMPUTER SCIENCE AND ENGINEERING** during the **2023-2024.**

PROJECT GUIDE	PROJECT-COORDINATOR
Dr. D. Venkata Reddy, M.Tech., (Ph.D.)	**Dr. M. Sireesha,** M.Tech., (Ph.D.)
Assistant Professor	**Associate Professor**

HEAD OF THE DEPARTMENT	EXTERNAL EXAMINER
Dr. S. N. Tirumala Rao, M.Tech., (Ph.D.)	
Professor & HoD	

DECLARATION

We declare that this project work titled "BONE FRACTURE DETECTION USING MACHINE LEARNING" is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has been submitted for any other degree or professional qualification except as specified.

B. Venkata Durga Praveen	(20471A05D7)
E. Rushi Naga Manikanta	(20471A05E7)
T. Sumanth	(20471A05J0)

ACKNOWLEDGEMENT

We wish to express my thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman Sri **M. V. Koteswara Rao,** **B.Sc.,** who took keen interest in us in every effort throughout this course. We owe out sincere gratitude to our beloved principal **Dr. M. Sreenivasa Kumar,** **M.Tech., (Ph.D.,) MISTE., FIE(I).,** for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao,** **M.Tech., (Ph.D.,)** HoD of CSE department and also to our guide **Dr. D. Venkata Reddy,** **M.Tech., (Ph.D.,)** of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dr. M. Sireesha,** **M.Tech., (Ph.D.,)** Associate professor & Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

By

B. Venkata Durga Praveen	(20471A05D7)
E. Rushi Naga Manikanta	(20471A05E7)
T. Sumanth	(20471A05J0)

NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project Course Outcomes (CO'S):

CO421.1: Analyse the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature

CO421.4: Design and Modularize the project

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		√											√		
C421.2	√		√		√								√		
C421.3				√		√	√	√					√		
C421.4			√			√	√	√					√	√	
C421.5					√	√	√	√	√	√	√	√	√	√	√
C421.6									√	√	√		√	√	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. **Low level**

2. **Medium level**

3. **High level**

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	Description of the device	Attained PO
C2204.2, C22L3.2	Gathering the requirements and defining the problem, plan to develop a system for bone fracture detection.	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement is critically analyzed, the process model is identified and divided into five modules.	PO2, PO3
CC421.2, C2204.2,C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3,C22L3.2	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC421.4, C2204.4,C22L3.2	Documentation is done by all our three members in the form of a group	PO10
CC421.5, C2204.2,C22L3.3	Each and every phase of the work in group is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementation is done and the project will be handled by the hospital management and in future updates in our project can be done based on deep learning techniques	PO4, PO7
C32SC4.3	The design includes software components like model file and python application	PO5, PO6

ABSTRACT

In today's interconnected world, computers play a pivotal role across diverse domains, revolutionizing aspects such as banking, online commerce, communication, education, research, and healthcare. To enhance medical practices and patient care, innovative technological solutions have emerged. Traditional X-ray scanners often produce indistinct images, posing a risk of misdiagnosis for bone fractures. A comprehensive approach that includes steps like pre-processing the x-ray image, bone edge finding, feature extraction, and machine learning classifiers has been designed to handle this difficulty. The algorithms accuracy evaluations range from 0.62 to 0.94. Remarkably, SVM stands out with the highest accuracy, surpassing most comparable studies. This statistical finding underscores the potential of SVM in fracture detection, reflecting advancements in medical imaging analysis.

INDEX

LIST OF FIGURES

1. INTRODUCTION

1.1 Introduction

The 206 bones that make up the human body vary in size, complexity, and form, with wrist fractures being common [1]. Machine learning in medical imaging has gained attention for aiding accurate diagnosis and treatment planning. This has helped doctors diagnose patients more precisely and create treatment programs that work best for them [2]. Because bone fractures can occur to anybody at any time and are becoming more common worldwide, especially in wealthy countries, prompt diagnosis and treatment are essential [3].

X-rays, supported by the DICOM standard, are widely used for diagnosing bone fractures. Because they are quick, inexpensive, and easy to use, X-rays are one of the most common diagnostic tools for bone fractures [4-5]. Medical imaging has developed quickly since Wilhelm Roentgen discovered X-rays in 1895, and it is now an essential part of contemporary diagnosis. Digital X-ray imaging equipment are widely used in many different medical settings due to their portability and advances in computerized image processing [4]. Medical data analytics relies heavily on machine learning, which calls for complex algorithms to analyze and spot anomalies in skeleton-related medical pictures [5].

Considering the variety of origins of bone fractures, timely and precise diagnosis is essential for successful treatment. X-rays are usually ordered by physicians or radiologists when a fracture is suspected in order to determine its kind and severity [6 Manual examination and conventional X-ray techniques are used to find fracture but they are laborious and error-prone. By using computer vision technologies, X-ray pictures can be screened for anomalies that could indicate a fracture and notify the treating physician [7]. With unacceptable faults coming from depending just on human specialists, the idea of automated diagnostic tools has gained momentum. Numerous techniques, such as preprocessing and fracture recognition, have been put forth for identifying and categorizing fractures to the leg bones seen in X-rays [8–9].

Identification of the fracture type is essential for choosing the best course of action and prognosis. More than twenty percent of hospital admissions are due to tibia fractures, which are the most prevalent type of bone fracture [10]. This puts a great deal of pressure on physicians, who have to review a large number of X-ray pictures every day. Over a century has passed since the invention of X-ray technology, but it's still the principal way to make a diagnostic [11]. Nonetheless, radiographic interpretation is frequently carried out without having access to knowledgeable colleagues for advice [12–13].

A fracture's best course of treatment and prognosis depend on its accurate classification into recognized categories. In this sense, computer-aided diagnostic (CAD) systems have potential to help physicians. Prior research on fracture categorization and identification has led to substantial use of traditional machine learning methodologies, such as preprocessing, feature extraction, and classification; nevertheless, machine learning algorithms have recently spurred breakthroughs in this field [14].

Using a variety of preprocessing techniques, the first stage of the process entails noise removal and early picture processing. The difficult work of extracting distinguishing features from the photos is what comes next. Finally, a number of machine learning classification methods are used, and they are assessed through the use of conventional testing protocols.

1.2 Existing System

Bone fracture detection has historically relied on manual inspection by radiologists and conventional X-ray imaging techniques. While these methods have been effective to some extent, they suffer from limitations such as subjectivity in interpretation and the potential for human error. With the rise of modern technological advancements, there has been a shift towards more automated and accurate approaches to bone fracture detection.

Traditional methods for bone fracture detection involve visual inspection of X-ray images by trained radiologists. This process is highly reliant on the expertise and experience of the radiologist, leading to variability in diagnosis and the possibility of missed fractures. Additionally, conventional X-ray imaging techniques may produce indistinct images, making it challenging to accurately identify fractures, especially in complex cases.

The main challenges associated with traditional methods include subjectivity in interpretation, limited scalability, and the potential for oversight or misdiagnosis. Manual inspection of X-ray images is time-consuming and labor-intensive, leading to delays in diagnosis and treatment. Moreover, human error can result in missed fractures or false positives, impacting patient care and outcomes.

Recent advancements in medical imaging and machine learning have revolutionized bone fracture detection. Techniques such as computer-aided diagnosis (CAD), deep learning, and image processing algorithms have shown promising results in automating fracture detection and improving diagnostic accuracy. These methods leverage large datasets and powerful computational tools to analyse X-ray images and identify subtle abnormalities indicative of fractures.

While existing methods have shown promise in automating bone fracture detection, there are still limitations to be addressed. Variability in performance metrics, such as accuracy, sensitivity, and specificity, across different studies underscores the need for standardized evaluation criteria and benchmark datasets. Additionally, challenges related to computational complexity, generalizability, and integration into clinical workflows remain areas of active research and development.

In summary, the existing system of bone fracture detection comprises a combination of traditional methods and modern technological advances. While traditional approaches have served as the foundation for diagnosis, recent advancements in medical imaging and machine learning offer opportunities for more accurate, efficient, and scalable fracture detection solutions. However, challenges such as variability in performance and integration into clinical practice persist, highlighting the need for continued innovation and research in this field.

1.3 Proposed System

The proposed system introduces a novel approach for bone fracture detection using machine learning techniques applied to X-ray images. By leveraging advanced preprocessing methods, feature extraction techniques, and machine learning algorithms, the system aims to enhance the accuracy and efficiency of fracture detection, ultimately improving patient care and outcomes.

The proposed system consists of four main stages: preprocessing, feature extraction, classification, and evaluation. X-ray images are first pre-processed to enhance image quality and reduce noise. Next, informative features are extracted from the pre-processed images using techniques such as Gray-Level Co-occurrence Matrix (GLCM) analysis. These features are then used as input to various machine learning algorithms for fracture detection and classification. Finally, the performance of the system is evaluated using standard metrics such as accuracy, precision, recall, and F1-score.

In the preprocessing stage, X-ray images undergo several transformations to improve their quality and suitability for analysis. This includes noise removal using a Gaussian filter to mitigate "salt and pepper" noise commonly found in X-ray images. Additionally, adaptive histogram equalization is applied to enhance image contrast, followed by Canny edge detection to extract structural information from the images.

Feature extraction plays a crucial role in capturing relevant information from X-ray images for fracture detection. In this system, features are extracted using the GLCM approach, which calculates textural properties such as contrast, correlation, homogeneity, energy, and dissimilarity. These features provide valuable insights into the spatial relationships between pixel intensities, aiding in the identification of bone fractures.

Several machine learning algorithms are employed for fracture detection and classification, including Decision Tree, Naïve Bayes, Nearest Neighbours, Random Forest, and Support Vector Machine (SVM). These algorithms are trained on a dataset comprising X-ray images of both fractured and intact lower leg bones, with the goal of accurately distinguishing between the two classes.

The proposed system is evaluated using a dataset obtained from the Kaggle, containing 120 fractured and 70 non-fractured X-ray images of lower leg bones. The dataset is randomly split into training and testing sets, with 80% used for training and 20% for testing. Performance is evaluated using standard evaluation metrics, including accuracy, precision and recall.

Experimental results demonstrate the effectiveness of the proposed system in accurately detecting and classifying bone fractures. Performance metrics for various machine learning algorithms are reported, with SVM achieving the highest accuracy of 94%. These results highlight the potential of the proposed system to streamline the diagnostic process for bone fractures, enabling timely and accurate treatment decisions.

The results obtained validate the efficacy of the proposed system in automating bone fracture detection using machine learning techniques. However, there are still areas for improvement, such as optimizing preprocessing techniques, exploring additional feature extraction methods, and enhancing the generalizability of the system across different datasets and clinical settings. Overall, the proposed system represents a significant advancement in medical imaging analysis, with implications for improving patient care and outcomes in fracture diagnosis.

In conclusion, the proposed system offers a novel approach to bone fracture detection using machine learning algorithms applied to X-ray images. By combining advanced preprocessing techniques, feature extraction methods, and machine learning algorithms, the system achieves high accuracy in fracture detection, paving the way for more efficient and accurate diagnosis in clinical practice.

1.5 System Requirements

1.5.1 Hardware Requirements:

System Type	: Intel Core i3 or above
Cache Memory	: 4MB (Megabyte)
RAM	: 8 gigabytes (GB)
Bus Speed	: 5 GT/s DBI2
Number of Cores	: 2
Number of Threads	: 4

1.5.2 Software Requirements:

- Operating System: Windows 10 Home, 64-bit Operating System
- Coding Language: Python
- Python Distribution: Google Colab

2. LITERACTURE SURVEY

A summary of the literature on Bone fracture identification and categorization is provided throughout this section, covering both traditional and novel approaches.

A meta classifier that combined neural network (NN) and decision tree (DT) methodologies was exhibited by E. Mysuru et al. [6] and showed enhanced accuracy, reaching 85%. Segmentation, detection of edges, initial processing, and extraction of features are some of the distinct steps in the process that result in the classification of bone into fractured and non-fractured categories.

Support Vector Machine (SVM) learning approach by , D. Prakash et al. [11] presented a method for long bone fracture classification that achieved a 78% detection rate for transverse and oblique fractures.

An approach by A. Rajput et al. [14] employs preprocessing, feature extraction, and SVM classification, achieving an accuracy of 84.7% in fracture detection.

N. Singh et al. [15] proved the efficacy of the Canny Edge Detection algorithm for edge detection in X-ray pictures, offering improved image analysis through intensity discontinuities.

A CNN technique using Spatial Fuzzy C-Means (SFCM) was presented by Y. Prathyusha et al. [16]. To reach a 78% accuracy and preprocessing stages.

Using local Shannon entropy computation, a hybrid technique reported by E. Sorantin et al. [17] successfully detects pediatric ulna and radius fractures with an astounding 91% accuracy rate.J.D. Obando et al. [18] proposed an X-ray image processing technique, with preprocessing enhancements such as CLAHE, achieving an 80% accuracy.

A deep neural network model was created by S. Rathor et al. [19] to differentiate between healthy and fractured bone. After correcting for overfitting using data augmentation, the model achieved a 92.44% classification accuracy.

Finally, R. Madupu et al. [20] compared their proposed method with Harris corner detection, demonstrating the superior accuracy (91%) of BPNN paired with cautious smoothing and perceptive edge detection for automated bone fracture identification.

2.1 Machine Learning

Machine learning, a subset of artificial intelligence (AI), has emerged as a powerful computational paradigm revolutionizing various fields, including healthcare. At its core, machine learning algorithms enable computers to learn from data, identify patterns, and make predictions or decisions without being explicitly programmed. In the context of medical imaging, machine learning holds tremendous potential for improving diagnostic accuracy, treatment planning, and patient outcomes.

In recent years, the application of machine learning in medical imaging has garnered considerable attention due to its ability to analyze vast amounts of image data rapidly and accurately. This has led to significant advancements in disease detection, diagnosis, and prognosis across numerous medical specialties.

In the domain of bone fracture detection, machine learning techniques offer innovative solutions to address the challenges associated with traditional diagnostic methods. Conventional approaches, such as manual examination and interpretation of X-ray images, are labor-intensive, time-consuming, and subject to human error. By contrast, machine learning algorithms can automate the process of fracture detection, providing faster and more consistent results while minimizing the risk of misdiagnosis.

Machine learning algorithms in fracture detection typically involve several key steps, including preprocessing of medical images to enhance clarity and remove noise, feature extraction to capture relevant information from the images, and classification or segmentation of fractures based on extracted features. These algorithms can be trained on large datasets of annotated medical images to learn complex patterns and relationships, enabling them to accurately identify fractures in new, unseen images.

The integration of machine learning into medical imaging analysis has the potential to revolutionize fracture diagnosis by improving accuracy, efficiency, and accessibility. By leveraging advanced computational techniques, healthcare professionals can benefit from more reliable diagnostic tools, leading to better patient care and outcomes.

In this study, we explore the application of machine learning algorithms in bone fracture detection, aiming to enhance diagnostic capabilities and contribute to the ongoing evolution of medical imaging analysis. By harnessing the power of machine learning, we strive to address the challenges inherent in traditional fracture detection methods and pave the way for more effective and efficient healthcare practices.

2.2 Types of Machine Learning

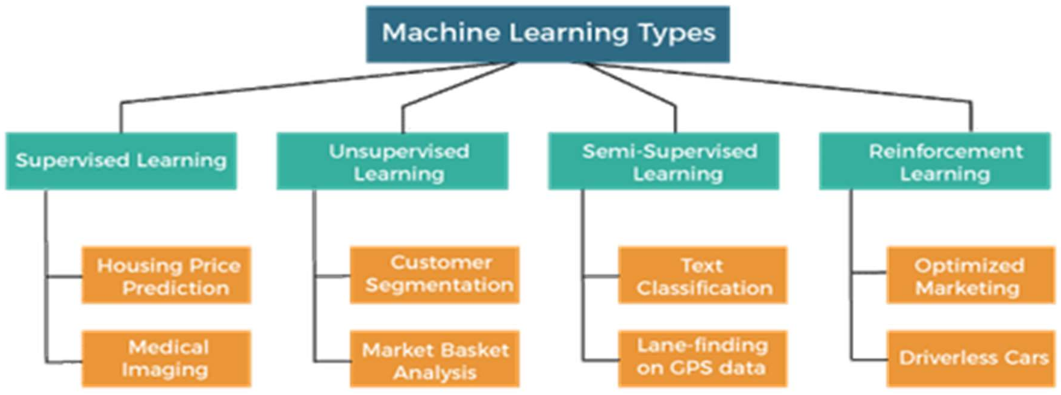


Fig. 2.2.1 Types of Machine Learning

A. Supervised Learning

- In supervised learning, the algorithm learns from labeled data, meaning the input data is paired with the corresponding correct output.
- The algorithm learns to map input features to the desired output by finding patterns in the data.
- Common tasks in supervised learning include classification (predicting categorical labels) and regression (predicting continuous values).
- Examples of supervised learning algorithms include linear regression, logistic regression, decision trees, random forests, support vector machines (SVM), and neural networks.

B. Unsupervised Learning

- In unsupervised learning, the algorithm learns from unlabeled data, where the input data lacks corresponding output labels.
- The goal of unsupervised learning is to find hidden patterns or structures in the data, such as clusters or associations.
- Common tasks in unsupervised learning include clustering (grouping similar data points together) and dimensionality reduction (reducing the number of features while preserving relevant information).
- Examples of unsupervised learning algorithms include k-means clustering, hierarchical clustering, principal component analysis (PCA), and autoencoders.

C. Semi-Supervised Learning

- Semi-supervised learning combines elements of both supervised and unsupervised learning, utilizing a small amount of labeled data along with a large pool of unlabeled data.
- The primary goal of semi-supervised learning is to leverage the additional unlabeled data to improve the performance of the model trained on limited labeled samples.
- Semi-supervised learning methods typically involve first using unsupervised learning techniques to identify underlying patterns or structures in the unlabeled data.
- Then, the model incorporates the labeled data to refine and adjust its predictions based on the discovered patterns, resulting in enhanced performance compared to using only labeled data.
- Examples of semi-supervised learning algorithms include self-training, co-training, and semi-supervised support vector machines (SVM). These algorithms iteratively update the model using both labeled and unlabeled data, gradually improving its performance over time.

D. Reinforcement Learning

- Reinforcement learning is a type of learning where an agent learns to make decisions by interacting with an environment.
- The agent receives feedback in the form of rewards or penalties based on its actions, and its goal is to learn the optimal policy that maximizes cumulative rewards over time.
- Reinforcement learning is used in scenarios where the algorithm must learn through trial and error, such as game playing, robotics, and autonomous vehicle control.
- Examples of reinforcement learning algorithms include Q-learning, Deep Q-Networks (DQN), and policy gradient methods.

2.3 Applications of Machine Learning

- Disease Diagnosis
- Drug Discovery

- Personalized Treatment
- Fraud Detection
- Risk Assessment
- Algorithmic Trading
- Personalized Recommendations
- Demand Forecasting
- Supply Chain Optimization
- Predictive Maintenance
- Quality Control
- Process Optimization
- Self-Driving Cars
- Traffic Management

2.4 Implementation of Machine Learning using Python

2.4.1 Introduction to Python

Python is a programming language created by Guido van Rossum and first released in 1991, has emerged as one of the most popular programming languages worldwide. Known for its simplicity, readability, and versatility, Python has gained widespread adoption across various domains, including web development, software engineering, scientific computing, artificial intelligence, and data analysis.

The most recent major version of Python is Python 3. However, Python 2, although not being updated with anything other than security updates, is still quite popular. It is possible to write Python in an Integrated Development Environment, such as Google Colab, Pycharm , Anaconda which are particularly useful when managing larger collections of Python files. Python was designed for its readability. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

In the older days, people used to perform Machine Learning tasks manually by coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules.

Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reason is its vast collection of libraries.

2.3.2 Python Libraries

Python libraries that used for project are:

A. Numpy

It is the fundamental package for scientific computing in Python. It provides support for powerful N-dimensional array objects and tools for working with these arrays. Numpy also includes functions for performing mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more.

B. Scikit-learn

A user-friendly library for machine learning in Python. It provides simple and efficient tools for data mining and data analysis, built on top of Numpy, Scipy, and Matplotlib. Scikit-learn includes various supervised and unsupervised learning algorithms, as well as tools for model evaluation, parameter tuning, and data preprocessing.

C. TensorFlow

An open-source deep learning framework developed by Google Brain. It enables developers to build and train machine learning models, particularly deep neural networks, efficiently. TensorFlow supports both CPU and GPU computation, distributed computing, and production deployment. It offers high-level APIs like Keras for ease of use and low-level APIs for flexibility and customization.

D. Keras

A high-level neural networks API written in Python and capable of running on top of TensorFlow, Theano, or CNTK. Keras facilitates fast prototyping and experimentation with deep learning models by providing a simple and consistent interface. It supports convolutional networks, recurrent networks, and their combinations, making it suitable for a wide range of applications.

E. Pandas

A powerful data analysis and manipulation library for Python. It provides data structures like DataFrame and Series for handling structured data and time series data effectively. Pandas offers functions for reading and writing data from various file formats, data cleaning, transformation, aggregation, and statistical analysis.

F. Matplotlib

A comprehensive library for creating static, interactive, and animated visualizations in Python. Matplotlib enables users to generate a wide variety of plots, charts, and graphs for data exploration, analysis, and presentation. It supports customization of plot appearance, labeling, annotation, and layout to create publication-quality figures.

G. OpenCV

OpenCV (Open Source Computer Vision Library) is a popular computer vision library that provides tools and functions for image and video processing, including image manipulation, object detection, and feature extraction.

H. Skimage

Also known as scikit-image, this library provides a collection of algorithms for image processing and computer vision tasks. It includes functions for image filtering, segmentation, feature extraction, and more.

I. Joblib

The joblib library in Python is a tool for efficiently saving and loading Python objects, particularly those containing large data arrays. It is commonly used for serializing and deserializing machine learning models, allowing users to save trained models to disk and reload them later for inference or further training without needing to retrain the model from scratch.

2.3.2 Machine Learning Methods

Machine Learning methods used in project are:

A. **Support Vector Machine (SVM)** is a powerful supervised learning algorithm used for classification and regression tasks. It is particularly effective in solving complex classification problems by finding the optimal hyperplane that best separates the data points into different classes. SVM works by mapping the input data into a high-dimensional feature space and finding the hyperplane that maximizes the margin between the classes.

B. **K-Nearest Neighbors (KNN)** is a simple yet powerful supervised learning algorithm used for classification and regression tasks. Unlike parametric models such as linear regression or logistic regression, KNN is a non-parametric algorithm, meaning it does not make any underlying assumptions about the distribution of the data.

C. **Decision Trees** are versatile supervised learning algorithms used for both classification and regression tasks. They are powerful tools for predictive modeling and are popular due to their simplicity and interpretability.

D. **Naive Bayes** is a simple yet effective probabilistic classifier based on Bayes' theorem with strong independence assumptions between the features. Despite its simplicity, Naive Bayes classifiers are widely used in various machine learning applications, particularly in text classification and spam filtering tasks.

E. **Random Forest** is a versatile and powerful ensemble learning method for classification and regression tasks. It operates by constructing a multitude of decision trees during training and outputting the mode (for classification) or the average prediction (for regression) of the individual trees.

F. **CLAHE** stands for Contrast Limited Adaptive Histogram Equalization. It is a variant of traditional Histogram Equalization (HE) that enhances the contrast of images by redistributing the intensity values of pixels. However, unlike traditional HE, CLAHE operates on small regions of the image, rather than the entire image at once. This adaptive approach prevents over-amplification of noise in regions with low contrast.

G. **GLCM** stands for Gray-Level Co-occurrence Matrix. It is a statistical method used in image processing and texture analysis to quantify the spatial relationships between pairs of pixels at various distances and angles within an image.

H. **Gaussian filter**, a core tool in image processing, smooths images by convolving them with a Gaussian kernel, reducing noise and preserving edges. Widely used for tasks like noise reduction and image smoothing, it offers adjustable parameters for customized blurring. Implemented efficiently in libraries like OpenCV and scipy, it remains indispensable for enhancing image quality.

3. SYSTEM ANALYSIS

3.1 Scope of Project

The scope of the project encompasses the development of a comprehensive bone fracture detection system using machine learning techniques applied to X-ray images. The system aims to automate the process of fracture detection and classification, thereby improving the efficiency and accuracy of diagnosis in clinical practice.

3.2 Analysis

The system consists of 5 main modules i.e Dataset collection, Pre-processing, Edge detection, Feature extraction, Classification.

The dataset was taken from Kaggle repository which contains 190 images out of 120 belongs to fractured and 70 are not fractured images. All the images are in jpeg format.

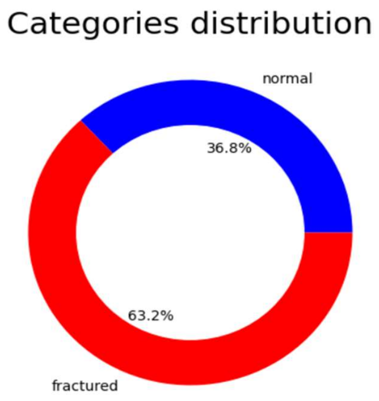


Fig. 3.2.1 Dataset distribution

In the analysis phase, various aspects of bone fracture detection are examined, including the challenges associated with manual diagnosis, the limitations of existing methods, and the potential benefits of automated detection systems. The analysis helps identify the key requirements and objectives of the proposed system.

3.3 Data Preprocessing

Data preprocessing is essential to enhance image quality, address noise, and ensure consistency in the dataset. The following paragraphs detail the specific procedures involved:

A. Conversion to Grayscale Format

The initial step in data preprocessing involves converting the input RGB images to grayscale format. Grayscale images simplify the analysis process by representing each pixel's intensity using a single value ranging from 0 (black) to 255 (white). This conversion reduces the complexity of the data while retaining essential information about the bone structure.

B. Noise Cancellation

X-ray images often suffer from various types of noise, such as "salt and pepper" noise caused by imaging or transmission artifacts. To mitigate the impact of noise on subsequent analysis, noise cancellation techniques are applied. One effective method is the use of a Gaussian filter to smooth out pixel intensities and remove extraneous noise. By replacing significantly different pixels with their median values, the Gaussian filter improves the overall quality of the images, enhancing the accuracy of fracture detection algorithms.

C. Enhancement of Contrast

Another critical aspect of data preprocessing is contrast enhancement, which aims to improve the visibility of features within the X-ray images. Adaptive histogram equalization, a digital image processing technique, is employed to adjust the image's contrast on a local level. Unlike traditional histogram equalization, which operates globally, adaptive histogram equalization adapts to the specific characteristics of each image region. By redistributing pixel intensities to achieve a more balanced histogram, this technique enhances the contrast of the X-ray images, making fractures and other structural details more discernible.

D. Edge detection

Edge detection plays a crucial role in the bone fracture detection system by identifying significant changes in pixel intensity, which correspond to structural disruptions in X-ray images. Utilizing the Canny edge detection technique, the system highlights boundaries and contours of fractures, facilitating precise localization and segmentation. These edges serve as essential indicators of fracture presence, allowing for enhanced feature extraction focused on fracture morphology. By accentuating structural details and suppressing background noise, edge detection contributes to noise reduction and artifact removal, improving overall image quality. Integrated seamlessly into the preprocessing pipeline, edge detection complements earlier steps such as noise cancellation and contrast enhancement, optimizing system performance and diagnostic accuracy.

These preprocessing steps are essential for optimizing the quality of the input data and ensuring that subsequent analysis algorithms can accurately detect and classify bone fractures. By addressing noise, enhancing contrast, and standardizing image formats, data preprocessing lays the foundation for effective fracture detection in medical X-ray scans.

3.4 Feature Extraction

Feature extraction plays a crucial role in identifying distinctive patterns and characteristics that distinguish fractured bones from healthy ones. One of the primary techniques employed for feature extraction is the Gray-Level Co-occurrence Matrix (GLCM). GLCM quantifies the spatial relationships between pixel intensities in an image, providing valuable information about textural patterns and structures. By analyzing the frequency of pixel pairs with specific intensity values and distances, GLCM extracts texture features that are indicative of bone fractures.

GLCM-derived textural features, such as contrast, correlation, homogeneity, energy, and dissimilarity, are computed to characterize the variations in pixel intensities within the X-ray images. These features capture important aspects of the image's texture and spatial distribution, which are essential for distinguishing fractured regions from background noise. Each textural feature provides unique insights into the underlying bone structure and helps differentiate between healthy and fractured areas.

- **Contrast:** Measures the variation in intensity between neighboring pixels, with higher values indicating more pronounced differences in texture.
- **Correlation:** Evaluates the linear relationship between pixel intensities, reflecting the degree of similarity in texture across the image.
- **Homogeneity:** Indicates the uniformity of pixel intensities, with higher values corresponding to smoother textures and more consistent structures.

- **Energy:** Represents the uniformity or regularity of pixel intensities, with higher values indicating a more homogeneous texture.
- **Dissimilarity:** Quantifies the average difference in intensity between neighboring pixels, highlighting areas of varied texture and structure.

3.5 Parameters Setting

Parameter setting for machine learning models is a crucial step in optimizing their performance for bone fracture detection. Each algorithm, such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Trees (DT), Naive Bayes, and Random Forests, has specific parameters that govern its behavior and predictive capability. For instance, in SVM, parameters like the kernel type, regularization parameter (C), and kernel coefficient (gamma) significantly impact the model's decision boundary and generalization ability. Similarly, k-NN requires tuning of the number of neighbors (k) to balance between bias and variance. Decision Trees involve parameters such as the maximum depth of the tree and the minimum number of samples required to split a node, affecting the model's complexity and overfitting tendency. Naive Bayes models typically do not have many tunable parameters but may require adjustments in smoothing techniques for handling sparse data. Random Forests offer parameters for the number of trees in the forest and the maximum depth of individual trees, influencing ensemble diversity and model complexity. Parameter setting involves techniques like grid search or random search to explore the hyperparameter space efficiently and find the combination that yields the best performance

3.6 Evaluation Metrics

Evaluation metrics are essential tools for assessing the performance of machine learning models in various tasks, including bone fracture detection from X-ray images. These metrics provide valuable insights into the effectiveness and reliability of the models, allowing researchers and practitioners to make informed decisions about their use in clinical settings. Some commonly used evaluation metrics for classification tasks include:

A. Accuracy

Accuracy measures the proportion of correctly classified instances out of the total number of instances. It is a straightforward metric that provides an overall assessment of model performance. However, accuracy alone may not be sufficient, especially when dealing with imbalanced datasets where one class is much more prevalent than others.

B. Precision

Precision measures the proportion of true positive predictions (correctly predicted fractures) out of all instances predicted as positive (both true positives and false positives). Precision is particularly useful when the cost of false positives is high, as it indicates the model's ability to avoid false alarms.

C. Recall

Recall, also known as sensitivity, measures the proportion of true positive predictions out of all actual positive instances (fractures). It is a crucial metric when the cost of false negatives (missed fractures) is high, as it assesses the model's ability to correctly identify positive cases.

D. Receiver Operating Characteristic (ROC) Curve

The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at various threshold settings. The AUC represents the area under the ROC curve and provides a single metric to summarize the classifier's performance across all possible threshold values. A higher AUC indicates better discrimination between positive and negative instances.

E. Confusion Matrix

A confusion matrix is a tabular representation of a classifier's predictions compared to the actual class labels. It provides a detailed breakdown of true positives, true negatives, false positives, and false negatives, allowing for a more nuanced understanding of the model's performance.

4. DESIGN ANALYSIS

In the design analysis phase, we delve into the architectural design and methodology employed in the development of the bone fracture detection system. This section outlines the key components, workflows, and design considerations essential for understanding the system's functionality and implementation.

4.1 Architectural Design

The architectural design of the bone fracture detection system encompasses several interconnected modules, each serving a specific purpose in the detection and classification process. These modules include:

A. **Preprocessing Module:**
   Responsible for enhancing image quality through noise reduction, contrast enhancement, and edge detection techniques.
B. **Feature Extraction Module:**
   Extracts informative features from pre-processed images using methods such as Gray-Level Co-occurrence Matrix (GLCM) analysis.
C. **Classification Module:**
   Utilizes machine learning algorithms, such as Decision Tree, Naïve Bayes, Random Forest, and Support Vector Machine (SVM), to classify X-ray images as fractured or non-fractured.
D. **Evaluation Module:**
   Evaluates the performance of the system using metrics such as accuracy, precision, recall.

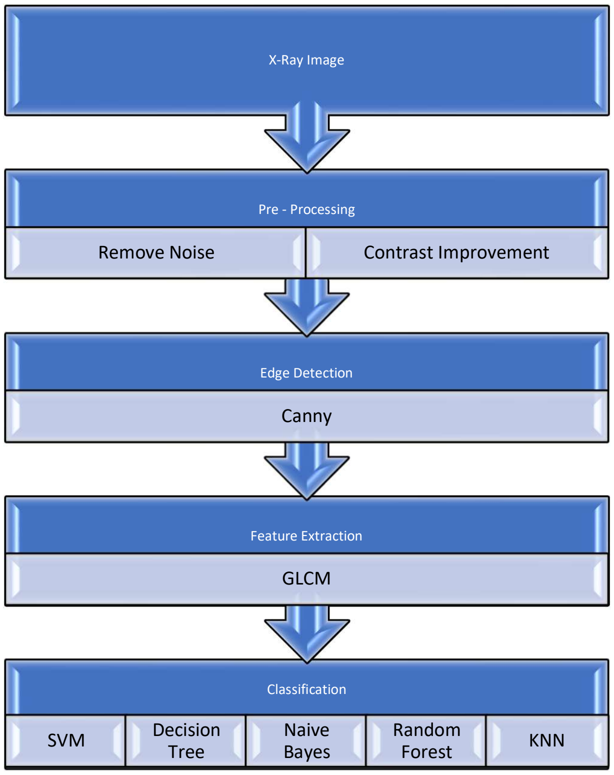


Fig. 4.1.1 Architecture

4.2 Workflow

The workflow of the bone fracture detection system can be summarized as follows:

A. Image Acquisition:

X-ray images of wrist bones are obtained from the dataset.

B. Preprocessing:

The acquired images undergo preprocessing steps, including noise reduction, contrast enhancement, and edge detection, to improve image quality and highlight structural features.

C. Feature Extraction:

Informative features are extracted from pre-processed images using GLCM analysis, capturing textural properties essential for fracture detection.

D. Classification:

Machine learning algorithms are applied to the extracted features to classify X-ray images as either fractured or non-fractured.

E. Evaluation:

The performance of the classification model is evaluated using standard evaluation metrics, providing insights into its accuracy and effectiveness.

4.3 Design Considerations

Several design considerations are crucial for the successful implementation of the bone fracture detection system:

A. **Scalability:**

The system should be scalable to accommodate large datasets and handle a high volume of X-ray images efficiently.

B. **Robustness:**

The system should be robust to variations in image quality, lighting conditions, and anatomical differences among patients.

C. **Interpretability:**

The classification results should be interpretable, enabling clinicians to understand the rationale behind the system's decisions.

D. **Modularity:**

The system should be modular, allowing for easy integration of new preprocessing techniques, feature extraction methods, and machine learning algorithms.

E. **Usability:**

The system should have a user-friendly interface, facilitating seamless interaction with clinicians and healthcare professionals.

5. IMPLEMENTATION

5.1 Modal development code

Importing necessary python modules for modal build:

```
import glob
import random
import matplotlib.pyplot as plt
import cv2
import numpy as np
from keras.preprocessing.image import  ImageDataGenerator
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import precision_score, recall_score, accuracy_score,confusion_matrix
from sklearn.model_selection import train_test_split
import seaborn as sns
from tabulate import tabulate
import pandas as pd
from skimage.feature import graycomatrix, graycoprops
```

Dataset initialization:

```
normal = glob.glob('/content/drive/MyDrive/CB3-Project/Dataset/not fractured/*.jpg')
fractured= glob.glob('/content/drive/MyDrive/CB3-Project/Dataset/fractured/*.jpg')
random.shuffle(normal)
random.shuffle(fractured)
```

Category distribution visualization of dataset:

```
def plt_pie (data,labels,colors,title):
    plt.figure(figsize=(5,5))
    my_circle=plt.Circle( (0,0), 0.7, color='white')
    plt.pie(data,labels = labels, colors = colors , autopct='%1.1f%%')
    p=plt.gcf()
    p.gca().add_artist(my_circle)
    plt.title (title,fontsize = 22)
    plt.show()
data = [len (normal) , len (fractured) ]
labels = ['normal' , 'fractured' ]
colors = ['blue','red']
plt_pie (data,labels,colors , 'Categories distribution')
```

Visualization of 5 random dataset images from each label:

```
def plot_images (data ,image_title,num_of_images ,type_image = 'bgr'):
    plt.figure(figsize=(15,10))
    for i in range (num_of_images):
        random_image = random.choice(data)
        if type (data) == list:
            random_image = cv2.imread(random_image)
            if type_image.lower() == 'bgr':
                random_image =  cv2.cvtColor(random_image, cv2.COLOR_BGR2GRAY)
                img=cv2.addWeighted (random_image,4, cv2.GaussianBlur(random_image, (5, 5),
0) ,-4 ,20)
        plt.subplot(1,num_of_images,i+1)
        plt.imshow( random_image,cmap='gray')
        plt.axis('off')
        plt.title (image_title)
    plt.show()
plot_images (normal ,'Normal',5)
plot_images (fractured ,'fractured',5)
```

Preprocessing of a x-ray image:

```
def show(image_path):
    original_image = cv2.imread(image_path)
    grayscale_image = cv2.cvtColor(original_image, cv2.COLOR_BGR2GRAY)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
    enhanced_image = clahe.apply(grayscale_image)
    img = cv2.GaussianBlur(enhanced_image, (5, 5), 0)
    edges = cv2.Canny(enhanced_image, 50, 80)
    plt.figure(figsize=(12, 4))
    plt.subplot(1, 4, 1)
    plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
    plt.title('Original Image')
    plt.subplot(1, 4, 2)
    plt.imshow(enhanced_image, cmap='gray')
    plt.title('Enhanced Image')
    plt.subplot(1, 4, 4)
    plt.imshow(edges,cmap='gray' )
    plt.title('Edge-detected Image')
    plt.subplot(1, 4, 3)
    plt.imshow(img, cmap='gray')
    plt.title('gausian filter')
    plt.show()
image_path = '/content/drive/MyDrive/CB3-Project/Dataset/not fractured/1.jpg'
show(image_path)
```

Preprocessing and augmentation of dataset:

```
def image_augmentation(low_data,target , p=1):
    images =[]
    while (len (images) < target):
        for image in low_data:
            img = cv2.resize(cv2.imread(image), (350, 350) )
            img =  cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            img = cv2.GaussianBlur(img, (5, 5), 0)
            clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
            img=clahe.apply(img)  #enhance image
            img = cv2.Canny(img, 50, 80)  # Apply Canny Edge Detection
            if img.shape[0] !=350:
                img = cv2.resize (img , (350, 350))
            images.append (img)
            if (len (images) >= target):
                break
    return np.array (images)
normal_images = image_augmentation(normal,130,p=1)
fractured_images = image_augmentation(fractured,120,p=1)
plot_images (normal_images ,'Normal',5)
plot_images (fractured_images ,'fractured',5)
```

Feature extraction on dataset:

```
def extract_glcm_features(image):
    distances = [1, 3, 5, 9]
    angles = [0, np.pi / 4, np.pi / 2, 3 * np.pi / 4, 5 * np.pi / 4, 3 * np.pi / 2, 7 * np.pi / 4]
    glcm_mat = graycomatrix(image, distances=distances, angles=angles)
    properties = ['energy', 'correlation', 'dissimilarity', 'homogeneity', 'contrast']
    glcm_features = [graycoprops(glcm_mat, prop).ravel() for prop in properties]
    return np.concatenate(glcm_features)
normal_glcm_features = [extract_glcm_features(img) for img in normal_images]
fractured_glcm_features = [extract_glcm_features(img) for img in fractured_images]
```

Showing GLCM features of 5 random images:

```
features = ['Energy', 'Correlation', 'Dissimilarity', 'Homogeneity', 'Contrast']
dfs = []
for i, feature in enumerate(normal_glcm_features):
    data = {
        'Image': f'Image_{i+1}',
        'Energy': feature[0],
        'Correlation': feature[1],
        'Dissimilarity': feature[2],
        'Homogeneity': feature[3],
        'Contrast': feature[4]
    }
    dfs.append(pd.DataFrame(data, index=[0]))
df = pd.concat(dfs, ignore_index=True)
print(df)
```

Dataset labelling:

```
Y = np.concatenate ([np.zeros (len(normal_images)) , np.ones (len (fractured_images)) ])
X = np.concatenate ([normal_images,fractured_images],axis = 0)
```

Splitting dataset:

```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
print (x_train.shape)
print (y_train.shape)
print (x_test.shape)
print (y_test.shape)
```

Reshaping and rescaling of dataset:

```
train_datagen =
ImageDataGenerator(rescale=1./255,rotation_range=45,width_shift_range=0.2,height_shift_r
ange=0.2,shear_range=0.2,zoom_range=0.2,horizontal_flip=True,fill_mode='nearest')
test_datagen = ImageDataGenerator(rescale=1./255,)
X_train_batches = x_train.reshape((-1, 350, 350, 1))  # Add a channel dimension for
grayscale
X_test_batches = x_test.reshape((-1, 350, 350, 1))
train_generator = train_datagen.flow(X_train_batches, y_train, batch_size=16, shuffle=True)
test_generator = test_datagen.flow(X_test_batches, y_test, batch_size=16, shuffle=False)


X_train_flat = X_train_batches.reshape((X_train_batches.shape[0], -1))
X_test_flat = X_test_batches.reshape((X_test_batches.shape[0], -1))
```

Support Vector Machine(SVM)

```
# Initialize and train the SVM model
svm_model = SVC(kernel='rbf', C=1000)  # You can adjust the kernel and C parameter
```

```
svm_model.fit(X_train_flat, y_train)
# Predict on the test set
y_pred_svm = svm_model.predict(X_test_flat)
# Evaluate the performance
precision_svm = precision_score(y_test, y_pred_svm)
recall_svm = recall_score(y_test, y_pred_svm)
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print(f"SVM Precision: {precision_svm:.2f}")
print(f"SVM Recall: {recall_svm:.2f}")
print(f"SVM Accuracy: {accuracy_svm * 100:.2f}%")
```

SVM confusion matrix:

```
cm = confusion_matrix(y_test, y_pred_svm)
sns.heatmap(cm, annot=True, fmt="d", cmap="Oranges", cbar=True)
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")
plt.show()
```

Random Forest Classifier

```
# Initialize and train the Random Forest model
rf_model = RandomForestClassifier()  # You can adjust the number of estimators
rf_model.fit(X_train_flat, y_train)
# Predict on the test set
y_pred_rf = rf_model.predict(X_test_flat)
# Evaluate the performance
precision_rf = precision_score(y_test, y_pred_rf)
recall_rf = recall_score(y_test, y_pred_rf)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f"Random Forest Precision: {precision_rf:.2f}")
print(f"Random Forest Recall: {recall_rf:.2f}")
print(f"Random Forest Accuracy: {accuracy_rf * 100:.2f}%")
```

RF confusion matrix:

```
cm = confusion_matrix(y_test, y_pred_rf)

sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", cbar=True)

plt.xlabel("Predicted")

plt.ylabel("True")

plt.title("Confusion Matrix")

plt.show()
```

K-Nearest Neighbors Classifier(KNN)

```
knn_model = KNeighborsClassifier(n_neighbors=30)  # You can adjust the number of
neighbors (k)

knn_model.fit(X_train_flat, y_train)

y_pred_knn = knn_model.predict(X_test_flat)

precision_knn = precision_score(y_test, y_pred_knn)

recall_knn = recall_score(y_test, y_pred_knn)

accuracy_knn = accuracy_score(y_test, y_pred_knn)

print(f"k-Nearest Neighbors Precision: {precision_knn:.2f}")

print(f"k-Nearest Neighbors Recall: {recall_knn:.2f}")

print(f"k-Nearest Neighbors Accuracy: {accuracy_knn * 100:.2f}%")
```

KNN confusion matrix:

```
cm = confusion_matrix(y_test, y_pred_knn)

sns.heatmap(cm, annot=True, fmt="d", cmap="Reds", cbar=True)

plt.xlabel("Predicted")

plt.ylabel("True")

plt.title("Confusion Matrix")

plt.show()
```

DecisionTreeClassifier

```
# Initialize and train the Decision Tree model with min_samples_split=40

dt_model = DecisionTreeClassifier(random_state=47, min_samples_split=30)

dt_model.fit(X_train_flat, y_train)

# Predict on the test set

y_pred_dt = dt_model.predict(X_test_flat)
```

```
# Evaluate the performance
precision_dt = precision_score(y_test, y_pred_dt)
recall_dt = recall_score(y_test, y_pred_dt)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print(f"Decision Tree Precision: {precision_dt:.2f}")
print(f"Decision Tree Recall: {recall_dt:.2f}")
print(f"Decision Tree Accuracy: {accuracy_dt * 100:.2f}%")
```

DT confusion matrix:

```
cm = confusion_matrix(y_test, y_pred_dt)
sns.heatmap(cm, annot=True, fmt="d", cmap="Greens", cbar=True)
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")
plt.show()
```

Gaussian Naive Bayes model

```
# Initialize and train the Gaussian Naive Bayes model
gnb_model = GaussianNB()
gnb_model.fit(X_train_flat, y_train)
# Predict on the test set
y_pred_gnb = gnb_model.predict(X_test_flat)
# Evaluate the performance
precision_gnb = precision_score(y_test, y_pred_gnb)
recall_gnb = recall_score(y_test, y_pred_gnb)
accuracy_gnb = accuracy_score(y_test, y_pred_gnb)
print(f"Gaussian Naive Bayes Precision: {precision_gnb:.2f}")
print(f"Gaussian Naive Bayes Recall: {recall_gnb:.2f}")
print(f"Gaussian Naive Bayes Accuracy: {accuracy_gnb * 100:.2f}%")
```

GNB confusion matrix:

```
cm = confusion_matrix(y_test, y_pred_gnb)
sns.heatmap(cm, annot=True, fmt="d", cmap="Greys", cbar=True)
```

```
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")
plt.show()
```

Testing new image on different models:

```
path = '/content/drive/MyDrive/CB3-Project/new data/fractured/Copy of 1.jpg'
img = cv2.resize(cv2.imread(path), (350, 350) )
img =  cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img = cv2.GaussianBlur(img, (5, 5), 0)
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
img=clahe.apply(img)  #enhance image
img = cv2.Canny(img, 50, 80)  # Apply Canny Edge Detection
if img.shape[0] !=350:
   img = cv2.resize (img , (350, 350))
test=np.array([img])
test_batches = test.reshape((-1, 350, 350, 1))
test_flat = test_batches.reshape((test_batches.shape[0], -1))
pred_svm = svm_model.predict(test_flat)
pred_knn = knn_model.predict(test_flat)
pred_dt = dt_model.predict(test_flat)
pred_rf = rf_model.predict(test_flat)
pred_gnb = gnb_model.predict(test_flat)
print("SVM Prediction:",pred_svm)
print("KNN Prediction:",pred_knn)
print("DT Prediction:",pred_dt)
print("RF Prediction:",pred_rf)
print("GNB Prediction:",pred_gnb)
```

Performance Analysis of ML models

```
models = ['SVM', 'Random Forest', 'Decision Tree', 'Gaussian NB','KNN']
precision_values = [precision_svm, precision_rf, precision_dt, precision_gnb,precision_knn]
```

```
recall_values = [recall_svm, recall_rf, recall_dt, recall_gnb,recall_knn]

accuracy_values = [accuracy_svm, accuracy_rf, accuracy_dt, accuracy_gnb,accuracy_knn]

precision_values = [round(val, 2) for val in precision_values]

recall_values = [round(val, 2) for val in recall_values]

accuracy_values = [round(val, 2) for val in accuracy_values]

table_data = zip(models, precision_values, recall_values, accuracy_values)

headers = ["Model", "Precision", "Recall", "Accuracy"]

table = tabulate(table_data, headers=headers, tablefmt="pretty")

print(table)

# Assuming you have precision, recall, and accuracy values for each model

models = ['SVM', 'Random Forest', 'Decision Tree', 'Gaussian NB',"KNN"]

precision_values = [precision_svm, precision_rf, precision_dt, precision_gnb,precision_knn]

recall_values = [recall_svm, recall_rf, recall_dt, recall_gnb,recall_knn]

accuracy_values = [accuracy_svm, accuracy_rf, accuracy_dt, accuracy_gnb,accuracy_knn]

# Round the values to two decimals

precision_values = [round(val, 2) for val in precision_values]

recall_values = [round(val, 2) for val in recall_values]

accuracy_values = [round(val, 2) for val in accuracy_values]

# Bar chart

fig, ax = plt.subplots(figsize=(10, 6))

bar_width = 0.2

index = range(len(models))

bar1 = ax.bar(index, precision_values, bar_width, label='Precision')

bar2 = ax.bar([i + bar_width for i in index], recall_values, bar_width, label='Recall')

bar3 = ax.bar([i + 2 * bar_width for i in index], accuracy_values, bar_width,
label='Accuracy')

ax.set_xlabel('Models')

ax.set_ylabel('Scores')

ax.set_title('Model Comparison')

ax.set_xticks([i + bar_width for i in index])

ax.set_xticklabels(models)
ax.legend()
plt.show(
```

K-Fold Cross Validation

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
# Specify the number of splits for K-fold cross-validation
kfold = KFold(n_splits=5, shuffle=True, random_state=42)
# Perform K-fold cross-validation and store the results
results = cross_val_score(svm_model, X_train_flat, y_train, cv=kfold)
# Print the cross-validation results
print("Cross-Validation Accuracy: %.2f%%" % (results.mean() * 100))
```

Joblib file download

```
from sklearn import svm
from sklearn import datasets
import joblib
```

Save the SVM model using joblib

```
joblib.dump(svm_model, 'svm_model.pkl')
from google.colab import files
```

Download the saved model file

```
files.download('svm_model.pkl')
```

5.2 Deployment code

app.py

```
from flask import Flask, render_template, request, session, redirect, url_for
from werkzeug.utils import secure_filename
import urllib.request
import re
import os
import bone_fracture_detector
app = Flask(__name__, static_folder='static', template_folder='templates')
```

```
UPLOAD_FOLDER = 'static/uploads/'
RESULT_FOLDER = 'static/results/'
app.secret_key = "secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['RESULT_FOLDER'] = RESULT_FOLDER
@app.route('/')
def home():
    return render_template('Index.html')
@app.route('/image')
def image():
    return render_template('Result.html')
@app.route('/image', methods=['POST'])
def upload_image():
    file = request.files['value1']
    filename = secure_filename(file.filename)
    pattern = r'^\d'
    if re.match(pattern, filename):
        file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        with open('filenames.txt', 'a') as f:
            f.write(filename + '\n')
        return render_template('Result.html', filename=filename)
    else:
        error_message = "Please upload X-ray image"
        return f'''
            <script>
                alert('{error_message}');
                window.history.back();
            </script>
        '''
@app.route('/image/display/<filename>')
```

```
def display_image(filename):
    session['filename'] = filename
    print(filename)
    return redirect(url_for('static', filename='uploads/' + filename), code=301)
@app.route('/image/detect/')
def detect():
    filename = session.get('filename', None)
    with open('filenames.txt', 'r') as f:
        filename = f.readlines()[-1].strip()
    path = 'static/uploads/'+str(filename)
    print(path)
    prediction = bone_fracture_detector.bone_image(path)
    print(filename, prediction)
    return render_template('Result.html', filename=filename, prediction=prediction)
if __name__ == '__main__':
    app.run(host="0.0.0.0",port=8080,debug=True)
```

bone_fracture_detector.py

```
from keras.preprocessing import image
from keras.applications.resnet50 import preprocess_input
from tensorflow.keras.models import load_model
import numpy as np
import os
import cv2
import joblib
os.environ['KERAS_BACKEND'] = 'tensorflow'
def bone_image(path):
    print(path)
    img = cv2.resize(cv2.imread(path), (350, 350) )
    img =  cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    img = cv2.GaussianBlur(img, (5, 5), 0)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
    img=clahe.apply(img)  #enhance image
    img = cv2.Canny(img, 50, 80)  # Apply Canny Edge Detection
    if img.shape[0] !=350:
        img = cv2.resize (img , (350, 350))
    test=np.array([img])
    test_batches = test.reshape((-1, 350, 350, 1))
    test_flat = test_batches.reshape((test_batches.shape[0], -1))
    model = joblib.load('svm_model.pkl')
    pred_svm = model.predict(test_flat)
    print(pred_svm)
    return pred_svm
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Bone Fracture Detection</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
    <style>
        .column {
        float: left;
        width: 33.33%;
        padding: 5px;
```

```
}
.row::after {
content: "";
clear: both;
display: table;
}
.c1 {
float: left;
width: 33.33%;
padding: 10px;
margin:60px;
}
.r1::after {
content: "";
clear: both;
display: table;
padding: 10px;
}
#div2 img {
width: 200px;
height: 250px;
display: block;
}

img {
width: 500px;
height: 400px;
margin-left:300px;
padding:10px;
display: block;
```

```
}
.column {
float: left;
width: 33.33%;
padding: 5px;
}
.row::after {
content: "";
clear: both;
display: table;
}
.c1 {
float: left;
width: 33.33%;
padding: 10px;
margin:40px;
}
.image{
		width:280px;
		height:300px;
	}

.r1::after {
content: "";
clear: both;
display: table;
padding: 10px;
}
#div2 img {
width: 200px;
```

```
        height: 250px;
        display: block;
        }
        h1{
            text-align: center;
        }
    </style>
</head>
<body>
<div class="container-fluid" style="background-color: #fff">
    <div class="jumbotron" style="background-color: #ff8566">
        <h1>Bone Fracture Detection</h1>
    </div>
    <div class="container">
        <div style="background-color:#b3ffb3;padding: 20px; margin: 40px">
            <div>
                <H1><a href="/image" >New predicton</a></H1>
            </div>
        </div>
    </div>
</div>
</body>
</html>
```

Result.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bone Fracture Detection</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link
      rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
    />
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
    <style>
      .column {
        float: left;
        width: 33.33%;
        padding: 5px;
      }
      .row::after {
        content: "";
        clear: both;
        display: table;
      }
      .c1 {
        float: left;
        width: 40.33%;
        padding: 10px;
        margin: 10px;
      }
      .image {
        width: 280px;
        height: 300px;
        margin-top: 80px;
      }
      .imag {
```

```
 width: 500px;
 height: 300px;
 margin-left: 70px;
}
h1{
    text-align: center;
 }
.r1::after {
 content: "";
 clear: both;
 display: table;
}
#div2 img {
 width: 200px;
 height: 250px;
 display: block;
}
.danger {
 color: red;
}
button {
 margin-top: 50px;
 margin-left: 100px;
 background-color: #008cba;
 font-size: 20px;
 color: white;
}
input {
 margin-top: 20px;
}
```

```
  </style>
</head>
<body>
  <div class="container-fluid" style="background-color: #fff">
    <div class="jumbotron" style="background-color: #ff8566">
      <h1>Bone Fracture Detection</h1>
    </div>
    <div class="container">
      <div class="jumbotron">
        <form
          class="form-horizontal"
          method="post"
          action="/image"
          enctype="multipart/form-data"
        >
          <div class="form-group" class="row">
            <label class="control-label col-sm-2" for="value1"
              >Upload X-Ray:</label
            >
            <div class="column" class="col-sm-10" id="div1">
              <div class="c1">
                <input
                  class="r1"
                  type="file"
                  id="value1"
                  name="value1"
                  accept="image/*"
                />
                <div class="r1" class="form-group">
                  <div class="col-sm-offset-2 col-sm-10">
            <input type="submit" value="Upload" />
          </div>
        </div>
      </div>
    </div>
    <div class="column" id="div2">
      <script>
        const chooseFile = document.getElementById("value1");
        const imgPreview = document.getElementById("div2");
        chooseFile.addEventListener("change", function () {
          getImgData();
        });
        function getImgData() {
          const files = chooseFile.files[0];
          if (files) {
            const fileReader = new FileReader();
            fileReader.readAsDataURL(files);
            fileReader.addEventListener("load", function () {
              imgPreview.innerHTML =
                '<img src="' + this.result + '" />';
            });
          }
        }
      </script>
    </div>
  </div>
</form>
{% if filename %}
<div class="r1">
  <div class="c1">
```

```
        <div class="display-image">
          <img
            src="{{ url_for('display_image', filename=filename) }}"
            class="image"
          />
        </div>
        <button
          type="button"
          onClick="parent.open('/image/detect/')"
          value="Detect"
        >
          Detect
        </button>
      </div>
      <div class="c1">
        {% if prediction==1 %}
        <h2>
          Prediction:
          <span class="danger">Opps! Your Bone is Fractured</span>
        </h2>
        <img
          class="imag"
          src="https://totalorthocare.in/wp-content/uploads/bone-fracture-foot-leg-male-
patient-being-examined-by-woman-doctor-hospital-1024x683.jpg"
          alt="Bone Image"
        />
        {% elif prediction==0 %}
        <h2>
          Prediction:
          <span class="safe"
            >No Worries ! Your Bone is not Fractured</span
```

```
                >
              </h2>
              <img
                class="imag"
                src="https://images.prismic.io/baker-tilly-www/a8a8b341-96a2-461e-8ff7-
a86dfdfd968e_Healthcare-Life-Sciences-Commercial-
Enablement.jpg?auto=compress,format&rect=0,425,3000,1222&w=1670&h=680"
                alt="Not Bone Fracture Image"
              />
              {% endif %}
            </div>
          </div>
          {% endif %}
        </div>
      </div>
    </div>
  </body>
</html>
```

6. Results Analysis

The results of the bone fracture detection system's performance provide valuable insights into its effectiveness and potential for clinical application. Here's an analysis of the key findings:

6.1 Pre-processing

A. Noise Removal

The application of noise removal techniques, such as Gaussian filtering, has significantly enhanced the quality of X-ray images, reducing the impact of artifacts and improving clarity for subsequent analysis.

B. Image Smoothing and Adaptive Histogram Equalization

These preprocessing steps have contributed to further improving image quality by reducing inconsistencies and enhancing contrast, essential for accurate feature extraction and classification.

C. **Canny Edge Detection:**

The Canny edge detection algorithm has effectively identified structural boundaries in X-ray images, enabling the extraction of informative features crucial for fracture detection.

Fig. 6.1.1 depicts the initial X - ray image prior to preprocessing. The X-ray image after a Gaussian filter is applied to remove noise is shown in Fig. 6.1.2. Fig. 6.1.3 showcases the X-ray image after enhancement and adaptive histogram equalization. Fig. 6.1.4 showcases the detection of bone structures and edges using the Canny edge detection algorithm.

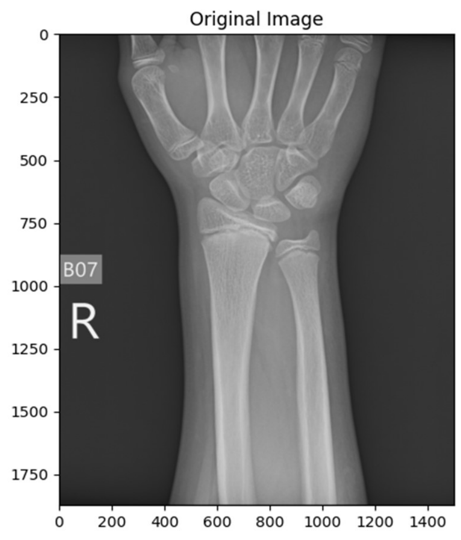


Fig. 6.1.1. X - ray image original

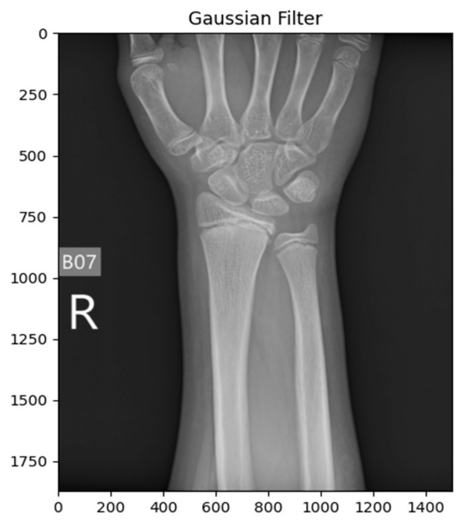


Fig. 6.1.2. X - ray after gaussian filter

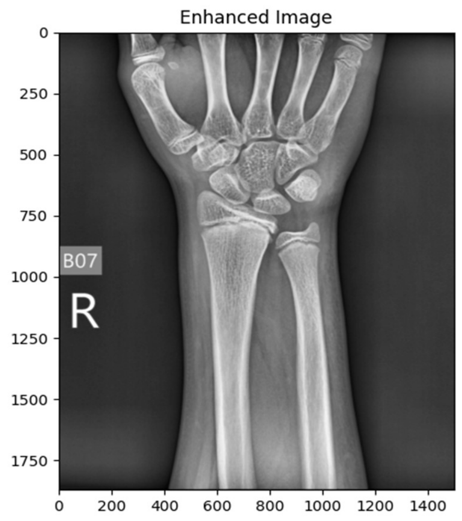


Fig. 6.1.3. X - ray after enhancing

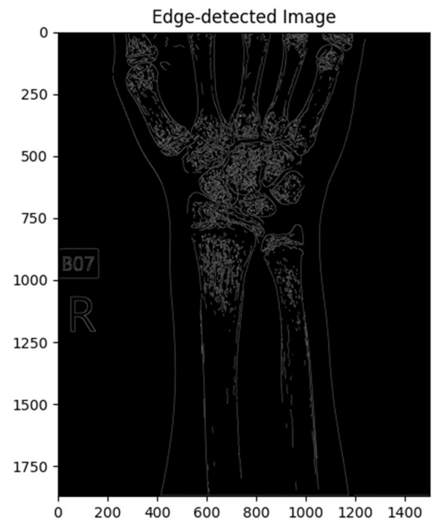
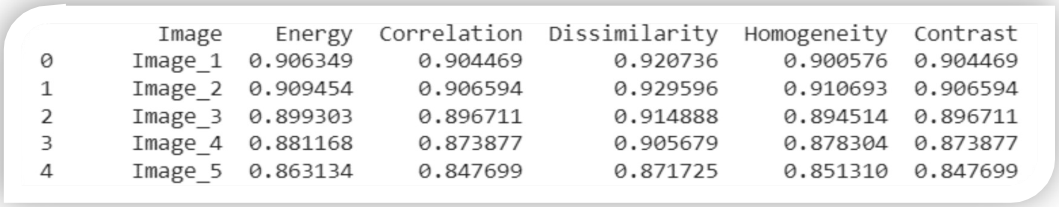


Fig. 6.1.4. X-ray after edge detection

6.2 Feature Extraction

The use of GLCM for feature extraction has yielded a rich set of textural properties, including energy, correlation, dissimilarity, homogeneity, and contrast, capturing essential characteristics of bone structures in X-ray images. Multiple Properties, Distances, and Angles: By considering multiple properties, distances (1, 3, 5, 9), and angles (0°, 45°, 90°, 135°, 180°, 225°, 270°), a comprehensive feature set comprising 140 features per image has been generated, enhancing the discriminative power of the classification model.



```
       Image    Energy  Correlation  Dissimilarity  Homogeneity  Contrast
0    Image_1  0.906349     0.904469       0.920736     0.900576  0.904469
1    Image_2  0.909454     0.906594       0.929596     0.910693  0.906594
2    Image_3  0.899303     0.896711       0.914888     0.894514  0.896711
3    Image_4  0.881168     0.873877       0.905679     0.878304  0.873877
4    Image_5  0.863134     0.847699       0.871725     0.851310  0.847699
```

Fig. *6.2.1*. GLCM features for 5 random images

6.3 Classification

Various machine learning algorithms, including Decision Tree, Naïve Bayes, Nearest Neighbors, Random Forest, and Support Vector Machine (SVM), were evaluated for fracture detection.

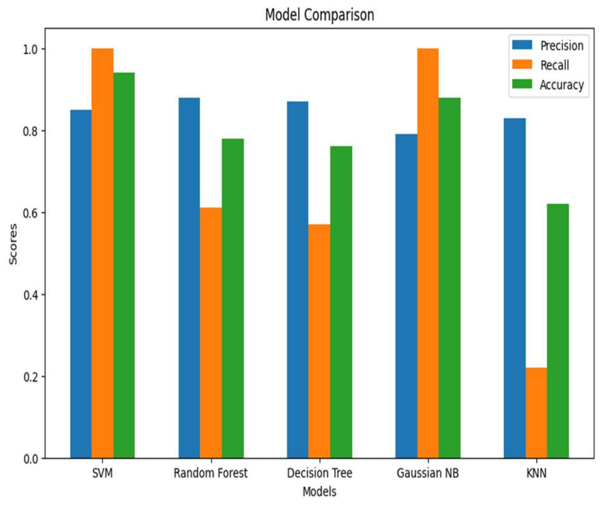


Fig. 6.3.1. Evaluation of different ML models performances

Table 1: Accuracy, Precision and Recall

Model	Precision	Recall	Accuracy
SVM	0.85	1.0	0.94
Random Forest	0.88	0.61	0.78
Decision Tree	0.87	0.57	0.76
Gaussian NB	0.79	1.0	0.88
KNN	0.83	0.22	0.62

Table 2: Comparing this paper to others

Citation	Year	Strategy	Accuracy
[6]	2015	Decision Tree	0.85
[11]	2017	SVM	0.78
[14]	2017	SVM	0.84
[16]	2019	CNN with SFCM	0.78
[18]	2019	SVM with CLACHE	0.80
[19]	2020	CNN	0.92
[20]	2020	Back propagation Neural Network	0.91
Current	**2024**	**SVM**	**0.94**

The performance of these algorithms was assessed in terms of precision, recall, and accuracy, with SVM emerging as the top-performing algorithm with an accuracy of 94%, outperforming other methods such as Decision Tree (76%) and Nearest Neighbors (62%).

The accuracy achieved by the proposed system is compared against benchmark models and existing studies, demonstrating its competitive performance and effectiveness in fracture detection. Figure 5 showcases the system's effectiveness compared to other studies, affirming its potential utility and contribution to the field of medical image analysis.

6.4 User Interface

In our project, the user interface (UI) is crafted with simplicity and functionality in mind, ensuring a smooth and intuitive experience for users interacting with the medical image classification system.

The UI features a straightforward design, guiding users through the process of uploading an X-ray image for fracture detection which shown in Fig. 20. Users begin by clicking on the "New Predictions" button as shown in Fig. 19, initiating the prediction process. They then upload the X-ray image like in Fig. 20 and click "Detect" to trigger the classification task. The UI presents the prediction results clearly, indicating whether the X-ray is fractured or not like in Fig. 21 & 22.

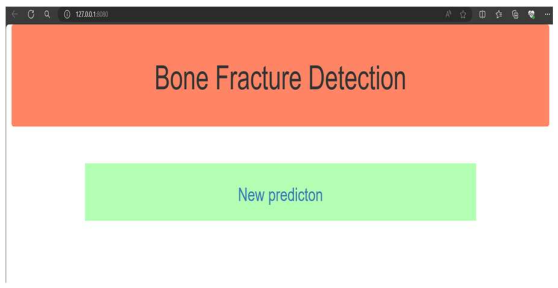
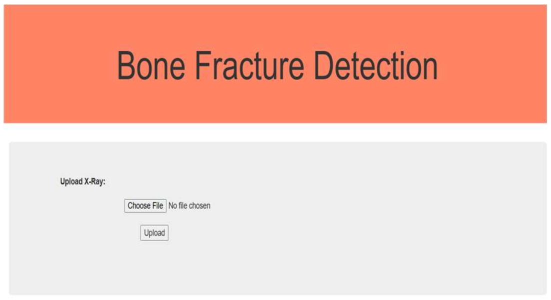
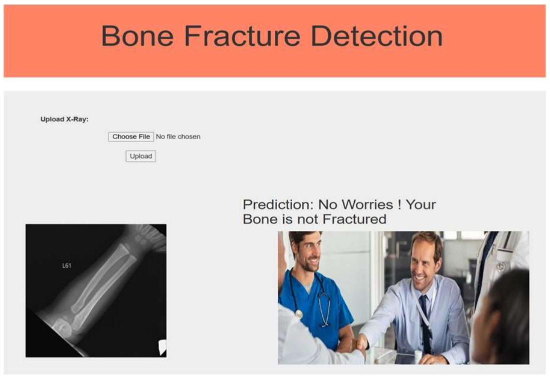


Fig. 6.4.1. Home Page



Bone Fracture Detection

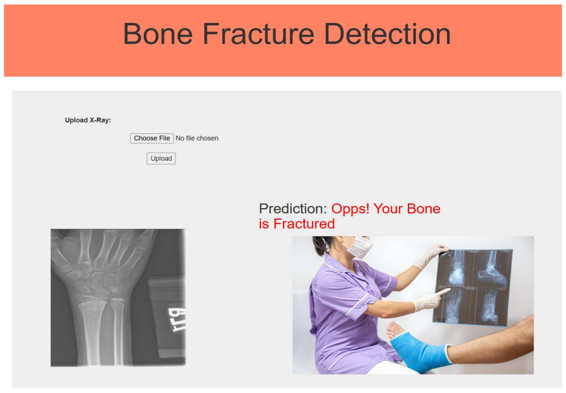
Fig. 6.4.2. X-ray upload



Bone Fracture Detection

Prediction: No Worries ! Your Bone is not Fractured

Fig. 6.4.3. Classification result if bone is not fractured



Bone Fracture Detection

Prediction: Opps! Your Bone is Fractured

Fig. 6.4.4. Classification result if bone is fractured

Overall, the results highlight the robustness and efficacy of the bone fracture detection system in accurately identifying fractures in lower leg bones from X-ray images. The combination of advanced preprocessing techniques, feature extraction methods, and machine learning algorithms has culminated in a system capable of achieving high accuracy and performance, with implications for enhancing clinical decision-making and patient care in healthcare settings.

7. CONCLUSION

The primary objective of this research was to develop a program aiding doctors in swiftly and accurately determining whether a patient's wrist bone is fractured or not. This study presents a machine learning-based approach for automating the detection and classification of bone fractures using X-ray images.

Both fractured and intact human bones, along with their corresponding X-ray images, were utilized in the experiment. With the increasing prevalence of bone fractures globally, the ability to detect even minor fractures holds significant value in medical practice. Thus, the proposed technique offers the capability to distinguish fractured bones from intact ones.

The Canny edge detector effectively identifies bone edges, while the Gray-Level Co-occurrence Matrix (GLCM) is utilized for extracting multiple features from X-ray images to be classified by machine learning algorithms. The system integrates a suite of image processing methods and machine learning algorithms tailored for detecting lower leg bone fractures.

Throughout the study, various machine learning methods including Naïve Bayes, Decision Tree, Nearest Neighbours, Random Forest, and Support Vector Machine (SVM) demonstrated accuracies ranging from 0.66 to 0.94. Notably, SVM exhibited statistically significant improvements over the baseline, indicating its efficacy in fracture detection and classification.

In conclusion, the developed system represents a promising approach to enhance the efficiency and accuracy of diagnosing wrist bone fractures, ultimately contributing to improved patient care and outcomes in medical practice.

8. FUTURE SCOPE

While the current research has laid a solid foundation for automated detection and classification of wrist bone fractures using machine learning techniques, there are several avenues for future exploration and enhancement. One potential area for further investigation is the expansion of the system's capabilities to detect fractures in other anatomical regions beyond the lower leg bones. Extending the algorithm's applicability to different bone structures would significantly broaden its utility in clinical settings, allowing for more comprehensive fracture diagnosis across various body parts.

Furthermore, incorporating deep learning architectures, such as convolutional neural networks (CNNs), could offer improvements in fracture detection accuracy and robustness. CNNs have demonstrated remarkable performance in image classification tasks, and their integration into the proposed system could potentially enhance its ability to accurately identify fractures from X-ray images with greater precision and efficiency.

Moreover, the development of a user-friendly interface for clinicians to interact with the system would streamline its integration into routine medical practice. Designing intuitive interfaces that provide real-time feedback and interpretation of X-ray images could facilitate quicker decision-making by healthcare professionals, ultimately leading to expedited diagnosis and treatment planning for patients with suspected fractures.

Additionally, ongoing research efforts could focus on leveraging advanced imaging modalities, such as computed tomography (CT) and magnetic resonance imaging (MRI), in conjunction with X-ray imaging for more comprehensive fracture assessment. By integrating multi-modal imaging techniques, the system could potentially enhance its diagnostic accuracy by leveraging complementary information from different imaging modalities.

Furthermore, collaboration with orthopaedic specialists and radiologists could provide valuable insights for refining the system's algorithms and validation procedures. Incorporating expert knowledge and clinical feedback would ensure that the developed system aligns closely with the needs and standards of medical practice, ultimately enhancing its clinical utility and adoption.

In conclusion, the future scope of this research encompasses a wide range of opportunities for advancing the automated detection and classification of bone fractures using machine learning techniques. By addressing these avenues for further exploration and enhancement, the developed system has the potential to significantly impact clinical practice by improving the efficiency, accuracy, and accessibility of fracture diagnosis, ultimately leading to better patient care and outcomes.

9.REFERENCES

[1]F. Majeed, S. Adnan W. Abbas & M. Javid," Lower Leg Bone Fracture Detection and Classification Using Faster RCNN for X-Rays Images", IEEE (2020).

[2]A. Al-Ghaithi & S. Al Maskari," Artificial intelligence application in bone fracture detection", Journal of Musculoskeletal Surgery and Research (2021).

[3]C. McCollough, J. Bushberg, G. Fletcher & L. Eckel, "Answers to common questions about the use and safety of CT scans", Elsevier(2015).

[4]P. Cephas & H. Hepzibah, "A robust approach for detection of the type of fracture from X-ray images", International Journal of Advanced Research in Computer and Communication Engineering (2015).

[5]Sami H. Ismael, Shahab W. Kareem & Firas H. Almukhta, "Medical Image Classification Using Different Machine Learning Algorithms", AL-Rafidain Journal of Computer Sciences and Mathematics (2020).

[6]S. College, E. Mysuru & R. Raman, "Detection of bone fracture using image processing methods", International Journal of Computer Applications (2015).

[7]S.W. Kareem, "An evaluation of algorithms for classifying Leukocytes images", IEEE(2021).

[8]B. Czerniak, "Dorfman and Czerniak's Bone Tumors E-Book", Elsevier(2015).

[9]Shahab Wahhab, Kareemab Roojwan ScHawezia & Farah Sami Khoshabaa, "A Comparison of Automated Classification Techniques for Image Processing in Video Internet of Things", Elsevier (2022).

[10]N. Umadevi & D. GeethaJakshmi, "Multiple classification system for fracture detection in human bone x-ray images", IEEE (2012).

[11]A. Vishnu, D. Prakash & S. Sharmila, "Detection and classification of long bone fractures", International Journal of Applied Engineering Research (2017).

[12]L. Tanzi, E. Vezzetti, R. Moreno & S. Moos, "X-Ray Bone Fracture Classification Using Deep Learning: A Baseline for Designing a Reliable Approach", MDPI (2020).

[13]Hersh A. Muhamad, Shahab Wahhab Kareem & Amin Salih Mohammed, "A deep learning method for detecting Leukemia in real images", MDPI (2022).

[14]A. Tripathi, A. Upadhyay & A. Rajput, "Automatic detection of fracture in femur bones using image processing", Elsevier (2017).

[15]N. Johari & N. Singh, "Bone fracture detection using edge detection technique", Springer Link(2017).

[16]S.S. Sinthura, Y. Prathyusha, K. Harini, Y. Pranusha & B. Poojitha, "Bone fracture detection system using CNN algorithm", IEEE (2019).

[17]F. Hrzic, I. Stajduhar, S. Tschauner, E. Sorantin & J. Lerga, "Local-entropy based approach for X-ray image segmentation and fracture detection", MDPI (2019).

[18]E. Castro-Gutierrez, L. Estacio-Cerquin, J. Gallegos-Guillen & J.D. Obando, "Detection of Acetabulum Fractures Using X-Ray Imaging and Processing Methods Focused on Noisy Images", IEEE (2019).

[19]D. Yadav & S. Rathor, "Bone fracture detection and classification using deep learning approach", IEEE(2020).

[20]S. Karimunnisa, R. Madupu & P. Savarapu, "Detection of Bone Fractures Automatically with Enhanced Performance with Better Combination of Filtering and Neural Networks", IEEE(2020).

Machine Learning: Bone Fracture Detection

Venkata Reddy Dodda
Assistant Professor
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
doddavenkatareddy@gmail.com

Venkata Durga Praveen Badri
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
badrivdpraveen5@gmail.com

Rushi Naga Manikanta Elchuri
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
elchurirushi@gmail.com

Sumanth Thota
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
sumanththota2003@gmail.com

***Abstract*— In today's interconnected world, computers play a pivotal role across diverse domains, revolutionizing aspects such as banking, online commerce, communication, education, research, and healthcare. To enhance medical practices and patient care, innovative technological solutions have emerged. Traditional X-ray scanners often produce indistinct images, posing a risk of misdiagnosis for bone fractures. A comprehensive approach that includes steps like pre-processing the x-ray image, bone edge finding, feature extraction, and machine learning classifiers has been designed to handle this difficulty. The algorithms accuracy evaluations range from 0.62 to 0.94. Remarkably, SVM stands out with the highest accuracy, surpassing most comparable studies. This statistical finding underscores the potential of SVM in fracture detection, reflecting advancements in medical imaging analysis.**

***Keywords*— Computers, Interconnected world, medical practices, Technological solutions, X-ray scanners, Bone fractures, Diagnostic process, Machine learning classifiers, Accuracy assessment, medical imaging analysis.**

I. Introduction

The 206 bones that make up the human body vary in size, complexity, and form, with wrist fractures being common [1]. Machine learning in medical imaging has gained attention for aiding accurate diagnosis and treatment planning [2], especially given the rising global incidence of bone fractures [3].

X-rays, supported by the DICOM standard, are widely used for diagnosing bone fractures due to their speed and simplicity [4-5], with advancements in digital imaging enhancing their accessibility [4]. Machine learning algorithms play a crucial role in analysing medical images to identify abnormalities, particularly in skeletal imaging [5].

Computer vision systems offer efficient screening of X-ray images for potential fractures, mitigating errors associated with manual inspection [7]. Various methods, including preprocessing and fracture identification, have been proposed to enhance fracture detection accuracy [8-9].

Accurate classification of fractures, especially common ones like tibia fractures, is vital for determining optimal treatment strategies [10]. The reliance on X-ray technology for diagnosis, despite its longevity, underscores the need for advanced diagnostic tools [11], such as computer-aided diagnosis (CAD) systems driven by machine learning algorithms [14].

To achieve accurate fracture classification, noise reduction, feature extraction, and machine learning classification algorithms are employed, highlighting the iterative process of image analysis in medical diagnostics..

II. Related Works

This section presents a synthesis of existing literature, encompassing both conventional and cutting-edge methodologies for bone fracture detection and classification.

A meta filter that combined neural network and decision tree methodologies was exhibited by E. Mysuru et al. [6] and showed enhanced accuracy, reaching 85%. Pre-processing, segmentation, edge detection, and feature extraction are some of the distinct steps in the process that result in the classification of bone into fractured and non-fractured categories.

Support Vector Machine (SVM) learning approach by , D. Prakash et al. [11] presented a method for long bone fracture classification that achieved a 78% detection rate for transverse and oblique fractures.

An approach by A. Rajput et al. [14] employs preprocessing, feature extraction, and SVM classification, achieving an accuracy of 84.7% in fracture detection.

The effectiveness of the Canny Edge Detection algorithm for edge detection in X-ray images was demonstrated by N. Singh et al. [15], providing enhanced image analysis through intensity discontinuities.

A CNN technique using Spatial Fuzzy C-Means (SFCM) was presented by Y. Prathyusha et al. [16]. To reach a 78% accuracy and preprocessing stages.

A hybrid approach presented by E. Sorantin et al. [17] effectively identifies paediatric ulna and radius fractures using local Shannon entropy computation, achieving a remarkable 91% accuracy.

J.D. Obando et al. [18] proposed an X-ray image processing technique, with preprocessing enhancements such as CLAHE, achieving an 80% accuracy.

S. Rathor et al. [19] developed a deep learning model for distinguishing between fractured and healthy bone, achieving a classification accuracy of 92.44% after addressing overfitting through data augmentation.

Finally, R. Madupu et al. [20] compared their proposed method with Harris corner detection, demonstrating the superior accuracy (91%) of BPNN paired with cautious smoothing and perceptive edge detection for automated bone fracture identification.

III. Methods

Dataset collection, preprocessing, edge detection, key feature finding, and lastly classification are the five primary modules that make up the bone fracture detection system.

Figure 1 outlines the stages of the suggested procedure for locating fractured bones in X - ray scans.

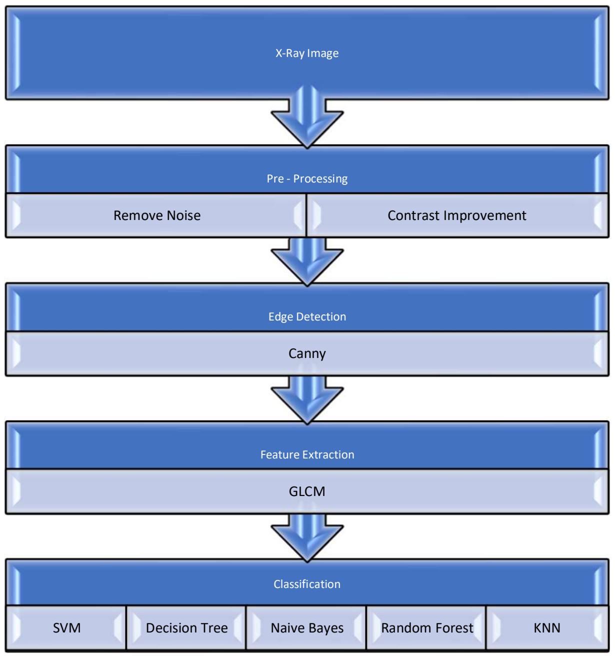


Fig. 1. Architecture

A. *Pre-processing*

By addressing noise, inconsistencies, and incompleteness, preprocessing is essential to improving the image's accuracy. The initial step involves converting the image from RGB to shades of grey format and then Gaussian filter used for cancellation of noise.

1. Noise Cancellation:

Noise is the word for extraneous pixels that degrade the general quality of the picture. Noise can manifest in various forms, with "salt and pepper" noise being common in X-ray images. This kind of noise shows up in the image as sporadic bright and dark patches, typically resulting from capture or transmission malfunctions.

Pixels that are significantly different from their neighbors are replaced with the median value, enhancing image quality.

This noise cancellation step is crucial for improving the accuracy of subsequent analysis in the bone fracture detection system.

2. Enhancement of Contrast:

A digital image processing method called adaptive histogram equalization is used to improve image contrast. This method has demonstrated success in improving the contrast of medical X-ray images.

B. *Canny Edge Detection*

A popular method for obtaining useful structural information from a variety of objects in vision systems is called "canny edge detection," which significantly lowers the volume of images that needs to be examine. This technique examines the image's fluctuating intensity over time. A few examples of the variables that affect edge detection quality are noise levels, edge density, and the presence of objects with similar intensities.

It has been found in this study that Canny edge detection technique can be modified to get the best results by adding an enhancement to improve contrast of image.

C. *Feature Extraction*

A crucial step in image processing, especially when it comes to bone fracture identification, is feature extraction. For the purpose of extracting textural qualities like contrast, correlation, homogeneity, energy, and dissimilarity, the GLCM technique is a useful tool.

These characteristics play a significant role in discerning subtle differences in bone structures, aiding in fracture identification. Below are the formulas for calculating each of these textural properties using the GLCM approach:

1. Contrast:

Contrast is a metric used to quantify the variation in strength between pixels next to each other in an image.

$$Contrast = \sum_{i,j} P(i,j) \cdot |i-j|^2 \quad (1)$$

Higher contrast values indicate larger differences in intensity levels between adjacent pixels, suggesting a more pronounced variation in the image.

2. Correlation:

Correlation assesses the degree of linear dependence between pixel intensities in the image.

$$Correlation = \sum_{i,j} \frac{(i-\mu_i)\cdot(j-\mu_j)\cdot p(i,j)}{\sigma_i \cdot \sigma_j} \quad (2)$$

A strong linear relationship between pixel intensities is shown by a correlation value near 1, which points to a more equal distribution of intensities in the image.

3. Homogeneity:

Homogeneity measures the closeness of pixel intensities to the diagonal of the GLCM.

$$Homogeneity = \sum_{i,j} \frac{p(i,j)}{1+|i-j|} \quad (3)$$

Higher homogeneity values indicate that pixel pairs with similar intensities are more likely to occur, suggesting a smoother texture in the image.

4. Energy:

Energy (also known as uniformity or angular second moment) quantifies the uniformity of pixel intensities in the image.

$$Energy = \sum_{i,j} p(i,j)^2 \quad (4)$$

Higher energy values indicate a more uniform distribution of pixel intensities in the image, suggesting a homogeneous texture.

5. Dissimilarity:

Dissimilarity measures the average intensity difference between pixel pairs in the image.

$$Dissimilarity = \sum_{i,j} p(i,j) \cdot |i-j| \quad (5)$$

Higher dissimilarity values indicate greater intensity differences between neighboring pixels, suggesting a more heterogeneous texture in the image.

These textural properties provide valuable insights into the spatial relationships between pixel intensities, allowing for effective characterization of bone structures and helping to identify fractures in X-ray pictures.

IV. Results

The bone fractured dataset consisted of X-ray scans obtained from the Kaggle containing 120 broken and 70 normal images of wrist bones. The methodology outlined in the previous sections was applied to the dataset, yielding the following results:

A. Pre-processing

Fig. 2 depicts the initial X - ray image prior to preprocessing. The X-ray image after a Gaussian filter is applied to remove noise is shown in Fig. 3.
Fig. 4 showcases the X-ray image after enhancement and adaptive histogram equalization.
Fig. 5 showcases the detection of bone structures and edges using the Canny edge detection algorithm.

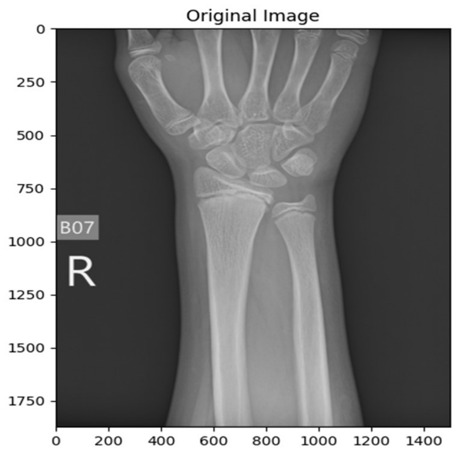


Fig. 2. X - ray image original

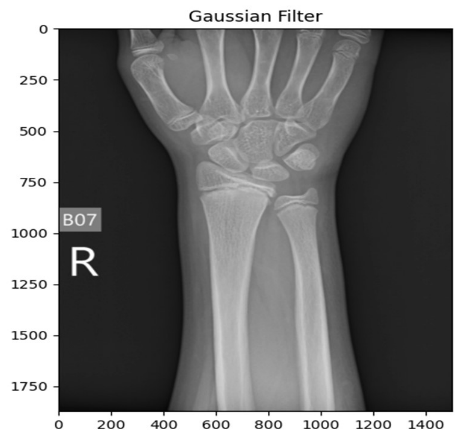


Fig. 3. X - ray after gaussian filter

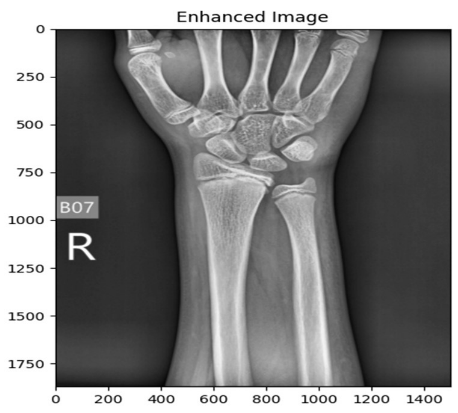


Fig. 4. X - ray after enhancing

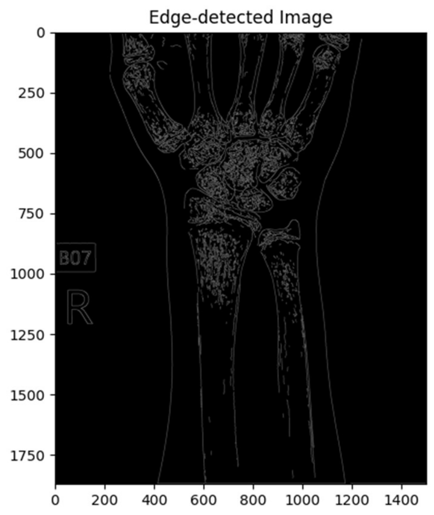
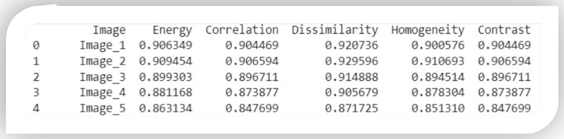


Fig. 5. X - ray after edge detection.

B. Feature Extraction

140 characteristics were extracted per image using GLCM, which produced five attributes for four distances and seven angles. To maximize accuracy, several combinations of characteristics, distances, and angles were tried while using Python programs to extract GLCM features.

Fig. 6 displays an example of extracted GLCM features for five photos with angle = 90° and distance = 1.



```
       Image    Energy  Correlation  Dissimilarity  Homogeneity  Contrast
0    Image_1  0.906349     0.904469       0.920736     0.900576  0.904469
1    Image_2  0.909454     0.906594       0.929596     0.910693  0.906594
2    Image_3  0.899303     0.896711       0.914888     0.894514  0.896711
3    Image_4  0.881168     0.873877       0.905679     0.878304  0.873877
4    Image_5  0.863134     0.847699       0.871725     0.851310  0.847699
```

Fig. ***6***. GLCM features for 5 random images

C. Classification

Using random selection, 80% of the dataset was implemented to learn and 20% for validation. The model was trained and tested using a variety of machine learning methods, such as Support Vector Machine (SVM), Naive Bayes, KNN, Random Forest, and Decision Tree. Preciseness, recall, and accuracy were assessed together with False Positive (FP), and False Negative (FN), True Positive (TP), True Negative (TN) results.

$$Precision = \frac{TP}{TP+FP} \qquad (6)$$

$$Recall = \frac{TP}{TP+FN} \qquad (7)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \qquad (8)$$

Table 1 displays a comparison of the efficacy of various machine learning methods. Accuracy of Naïve bayes is 0.88, Decision tree is 0.76, Random Forest is 0.62, KNN is 0.78, and SVM is 0.94 were attained.

Table 2 provides a comparison with other studies reviewed in this research, showcasing the effectiveness of the proposed system.

Table 1: Accuracy, Precision and Recall

Model	**Precision**	**Recall**	**Accuracy**
SVM	0.85	1.0	0.94
Random Forest	0.88	0.61	0.78
Decision Tree	0.87	0.57	0.76
Gaussian NB	0.79	1.0	0.88
KNN	0.83	0.22	0.62

Table 2: Comparing this paper to others

Citation	**Year**	**Strategy**	**Accuracy**
[6]	2015	Decision Tree	0.85
[11]	2017	SVM	0.78
[14]	2017	SVM	0.84
[16]	2019	CNN with SFCM	0.78
[18]	2019	SVM with CLACHE	0.80
[19]	2020	CNN	0.92
[20]	2020	Back propagation Neural Network	0.91

Existing	2024	SVM	0.94

V. Conclusion

The main target of this study was to produce a tool that would assist medical professionals quickly and effectively diagnose whether or not a patient had a fractured wrist bone. This study describes an automated method for classifying and identifying bone fractures from X-ray images using machine learning.

Both fractured and intact bones, along with their corresponding X-ray images, were utilized in the experiment. With the increasing prevalence of bone fractures globally, the ability to detect even minor fractures holds significant value in medical practice. Thus, the proposed technique offers the capability to distinguish broken bones versus whole ones.

The GLCM is utilized for obtaining various characteristics from X-ray scans in order to machine learning algorithms can use them for classification, however the Canny edge detection is an asset for edge detection. A multitude of machine learning techniques and methods for analyzing images are included into the system with the express purpose of identifying lower wrist bone fractures.

Throughout the research, various machine learning strategies including Decision Tree, Naive Bayes, KNN, SVM, and Random Forest demonstrated accuracies ranging from 0.66 to 0.94. Notably, SVM exhibited its effectiveness in fracture detection and classification more accurate.

In conclusion, the developed system represents a promising approach to enhance the efficiency and accuracy of diagnosing wrist bone fractures, ultimately contributing to improved patient care and outcomes in medical practice.

References

[1] F. Majeed, S. Adnan W. Abbas & M. Javid," Lower Leg Bone Fracture Detection and Classification Using Faster RCNN for X-Rays Images", IEEE (2020).

[2] A. Al-Ghaithi & S. Al Maskari," Artificial intelligence application in bone fracture detection", Journal of Musculoskeletal Surgery and Research (2021).

[3] C. McCollough, J. Bushberg, G. Fletcher & L. Eckel, "Answers to common questions about the use and safety of CT scans", Elsevier(2015).

[4] P. Cephas & H. Hepzibah, "A robust approach for detection of the type of fracture from X-ray images", International Journal of Advanced Research in Computer and Communication Engineering (2015).

[5] Sami H. Ismael, Shahab W. Kareem & Firas H. Almukhta, "Medical Image Classification Using Different Machine Learning Algorithms", AL-Rafidain Journal of Computer Sciences and Mathematics (2020).

[6] S. College, E. Mysuru & R. Raman, "Detection of bone fracture using image processing methods", International Journal of Computer Applications (2015).

[7] S.W. Kareem, "An evaluation of algorithms for classifying Leukocytes images", IEEE(2021).

[8] B. Czerniak, "Dorfman and Czerniak's Bone Tumors E-Book", Elsevier(2015).

[9] Shahab Wahhab, Kareemab Roojwan ScHawezia & Farah Sami Khoshabaa, "A Comparison of Automated Classification Techniques for Image Processing in Video Internet of Things", Elsevier (2022).

[10] N. Umadevi & D. GeethaJakshmi, "Multiple classification system for fracture detection in human bone x-ray images", IEEE (2012).

[11] A. Vishnu, D. Prakash & S. Sharmila, "Detection and classification of long bone fractures", International Journal of Applied Engineering Research (2017).

[12] L. Tanzi, E. Vezzetti, R. Moreno & S. Moos, "X-Ray Bone Fracture Classification Using Deep Learning: A Baseline for Designing a Reliable Approach", MDPI (2020).

[13] Hersh A. Muhamad, Shahab Wahhab Kareem & Amin Salih Mohammed, "A deep learning method for detecting Leukemia in real images", MDPI (2022).

[14] A. Tripathi, A. Upadhyay & A. Rajput, "Automatic detection of fracture in femur bones using image processing", Elsevier (2017).

[15] N. Johari & N. Singh, "Bone fracture detection using edge detection technique", Springer Link(2017).

[16] S.S. Sinthura, Y. Prathyusha, K. Harini, Y. Pranusha & B. Poojitha, "Bone fracture detection system using CNN algorithm", IEEE (2019).

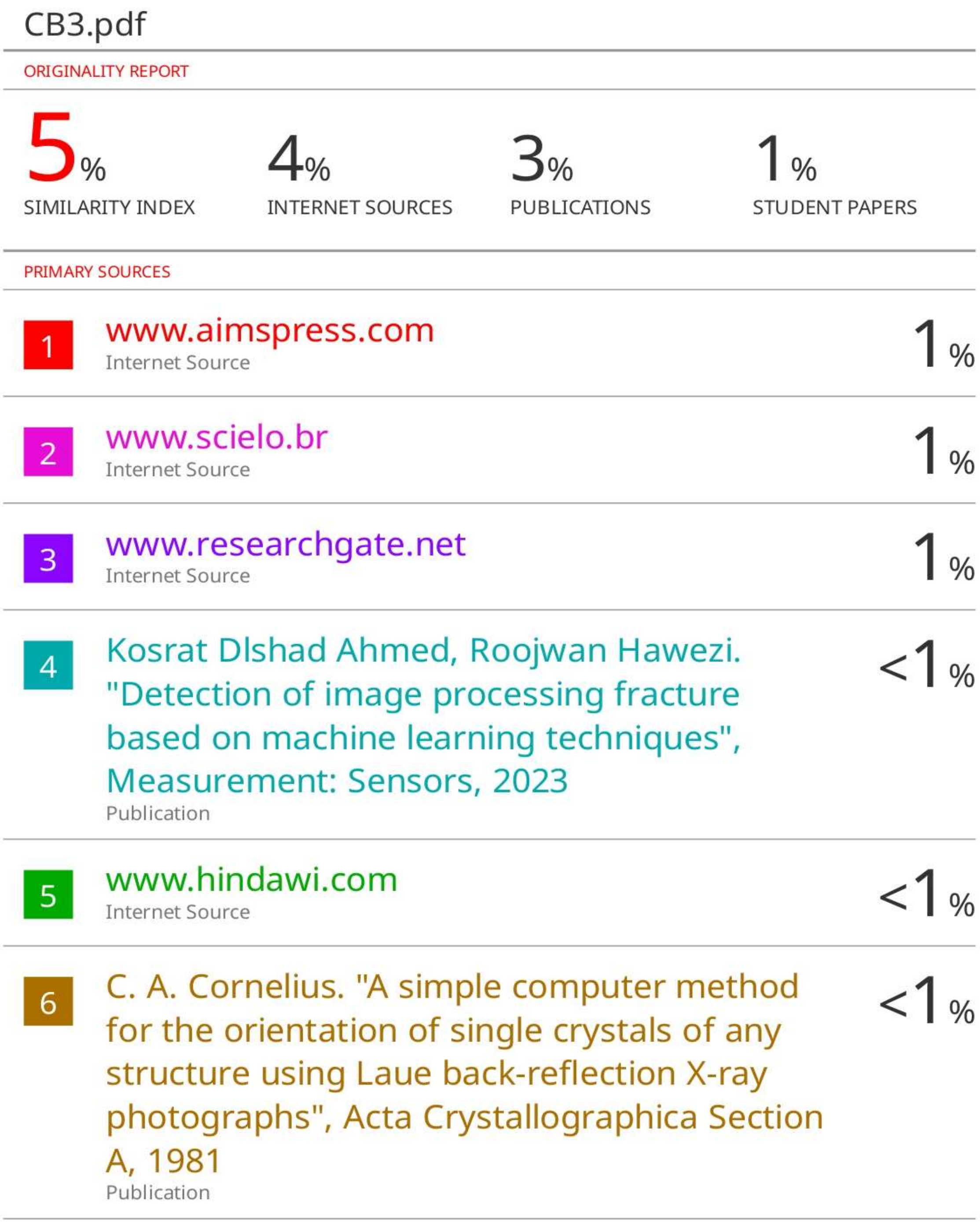
[17] F. Hrzic, I. Stajduhar, S. Tschauner, E. Sorantin & J. Lerga, "Local-entropy based approach for X-ray image segmentation and fracture detection", MDPI (2019).

[18] E. Castro-Gutierrez, L. Estacio-Cerquin, J. Gallegos-Guillen & J.D. Obando, "Detection of Acetabulum Fractures Using X-Ray Imaging and Processing Methods Focused on Noisy Images", IEEE (2019).

[19] D. Yadav & S. Rathor, "Bone fracture detection and classification using deep learning approach", IEEE(2020).

[20] S. Karimunnisa, R. Madupu & P. Savarapu, "Detection of Bone Fractures Automatically with Enhanced Performance with Better Combination of Filtering and Neural Networks", IEEE(2020).

Conference Plagiarism Report



CB3.pdf

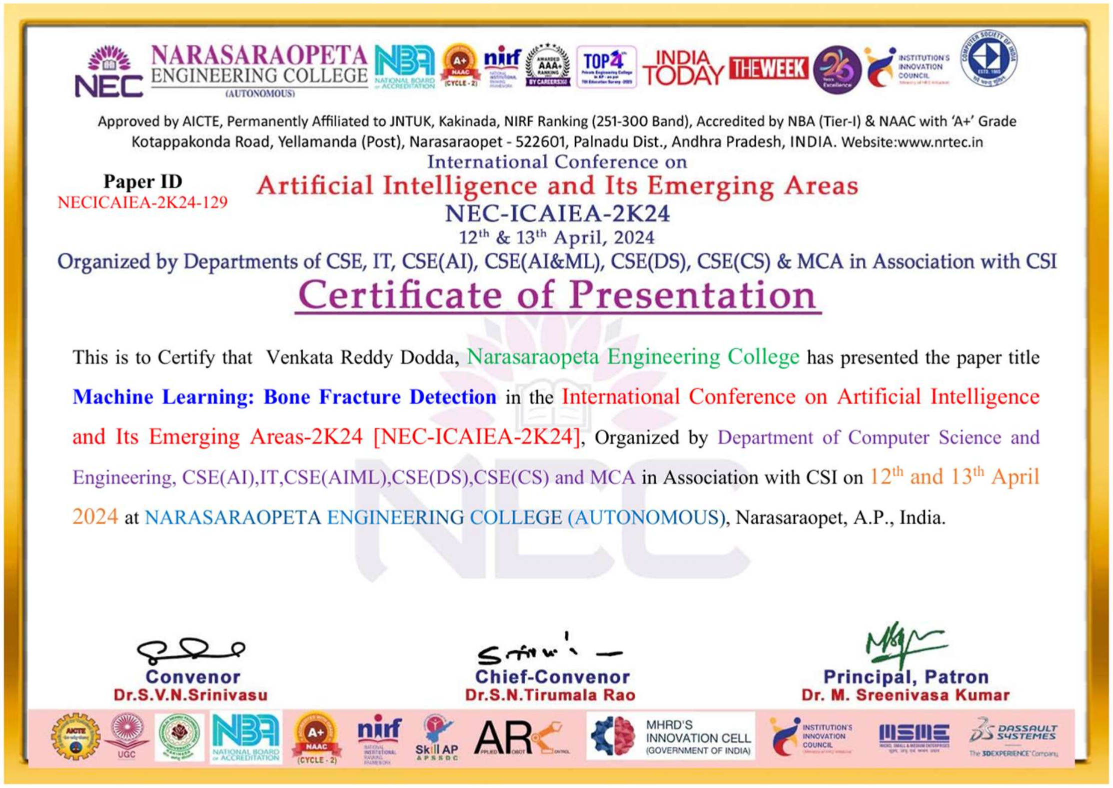
ORIGINALITY REPORT

5%	4%	3%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.aimspress.com Internet Source	1%
2	www.scielo.br Internet Source	1%
3	www.researchgate.net Internet Source	1%
4	Kosrat Dlshad Ahmed, Roojwan Hawezi. "Detection of image processing fracture based on machine learning techniques", Measurement: Sensors, 2023 Publication	<1%
5	www.hindawi.com Internet Source	<1%
6	C. A. Cornelius. "A simple computer method for the orientation of single crystals of any structure using Laue back-reflection X-ray photographs", Acta Crystallographica Section A, 1981 Publication	<1%

Conference Certificates



NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on

Paper ID
NECICAIEA-2K24-129

Artificial Intelligence and Its Emerging Areas

NEC-ICAIEA-2K24

12th & 13th April, 2024

Organized by Departments of CSE, IT, CSE(AI), CSE(AI&ML), CSE(DS), CSE(CS) & MCA in Association with CSI

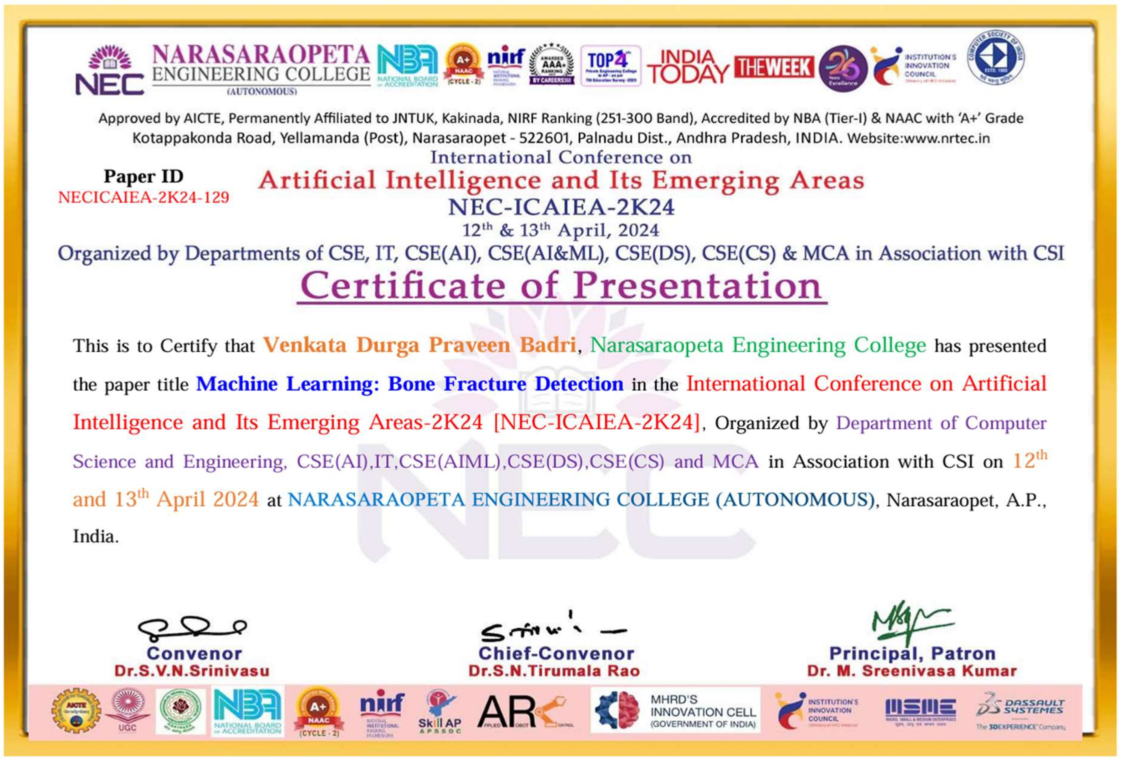
Certificate of Presentation

This is to Certify that Venkata Reddy Dodda, Narasaraopeta Engineering College has presented the paper title **Machine Learning: Bone Fracture Detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K24 [NEC-ICAIEA-2K24], Organized by Department of Computer Science and Engineering, CSE(AI),IT,CSE(AIML),CSE(DS),CSE(CS) and MCA in Association with CSI on 12th and 13th April 2024 at NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS), Narasaraopet, A.P., India.

[illegible]
Convenor
Dr.S.V.N.Srinivasu

[illegible]
Chief-Convenor
Dr.S.N.Tirumala Rao

[illegible]
Principal, Patron
Dr. M. Sreenivasa Kumar



NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on
Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K24
12th & 13th April, 2024
Organized by Departments of CSE, IT, CSE(AI), CSE(AI&ML), CSE(DS), CSE(CS) & MCA in Association with CSI

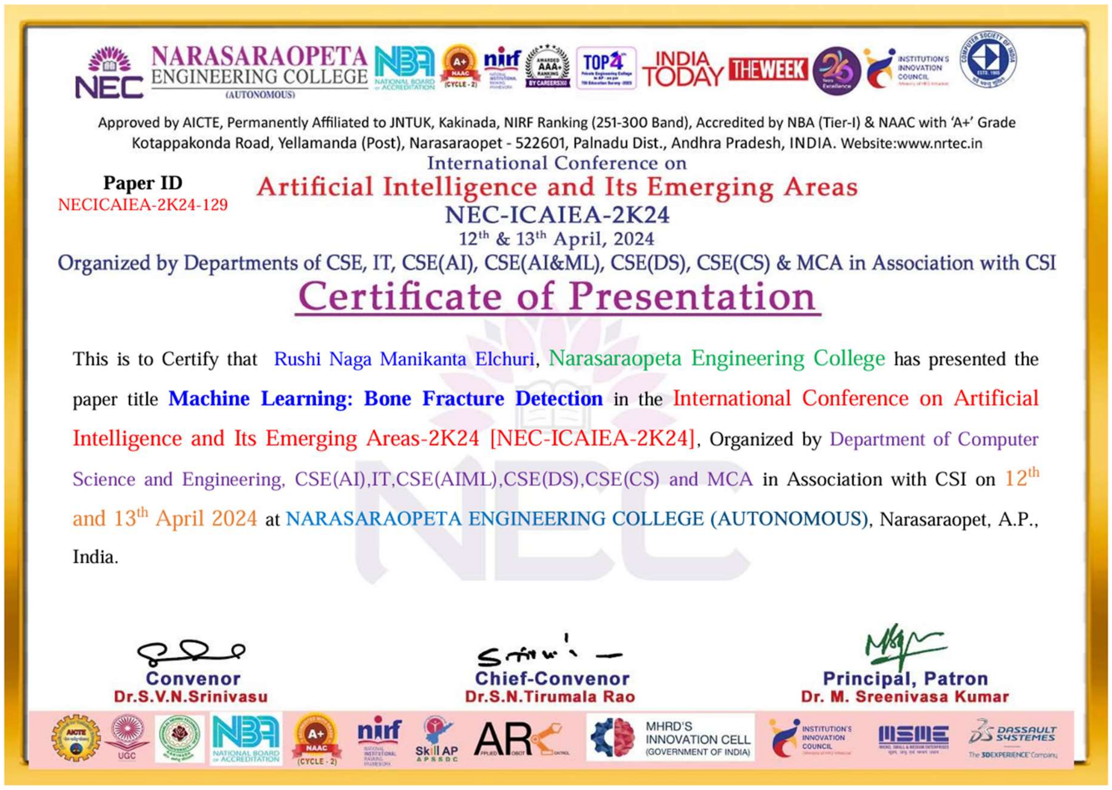
Paper ID
NECICAIEA-2K24-129

Certificate of Presentation

This is to Certify that **Venkata Durga Praveen Badri**, Narasaraopeta Engineering College has presented the paper title **Machine Learning: Bone Fracture Detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K24 [NEC-ICAIEA-2K24], Organized by Department of Computer Science and Engineering, CSE(AI),IT,CSE(AIML),CSE(DS),CSE(CS) and MCA in Association with CSI on 12th and 13th April 2024 at NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS), Narasaraopet, A.P., India.

[illegible]	[illegible]	[illegible]
Convenor	Chief-Convenor	Principal, Patron
Dr.S.V.N.Srinivasu	Dr.S.N.Tirumala Rao	Dr. M. Sreenivasa Kumar

MHRD'S INNOVATION CELL (GOVERNMENT OF INDIA)



NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on
Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K24
12th & 13th April, 2024

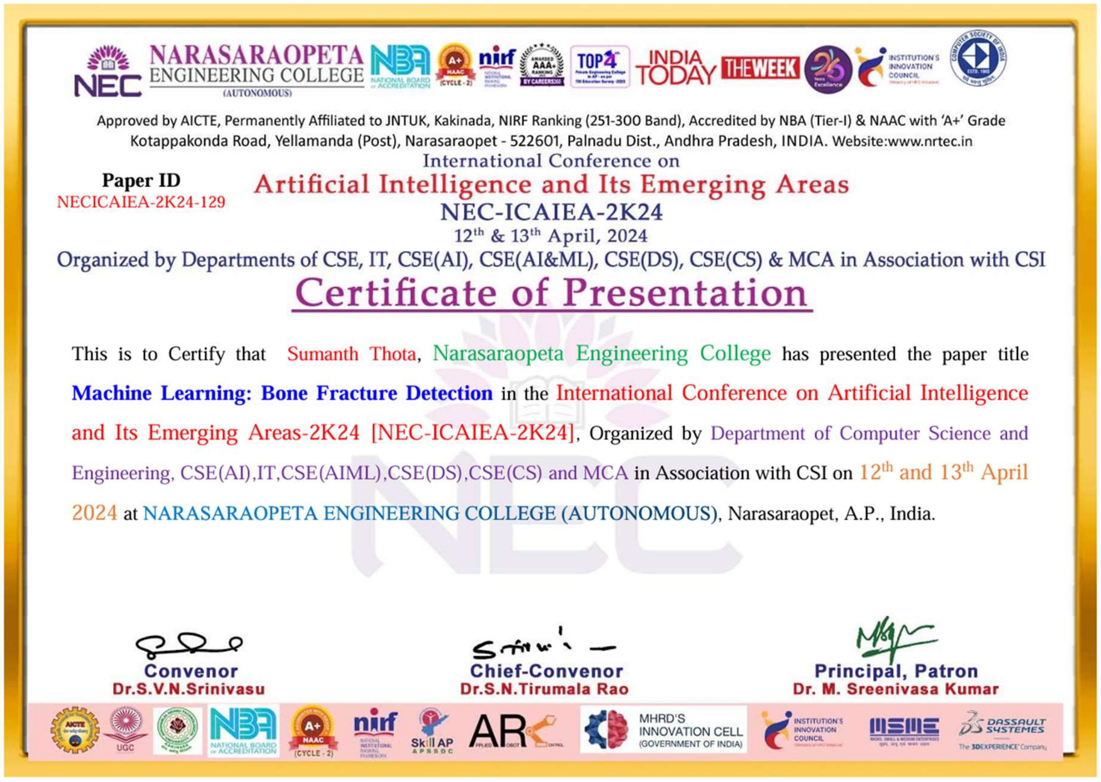
Paper ID
NECICAIEA-2K24-129

Organized by Departments of CSE, IT, CSE(AI), CSE(AI&ML), CSE(DS), CSE(CS) & MCA in Association with CSI

Certificate of Presentation

This is to Certify that Rushi Naga Manikanta Elchuri, Narasaraopeta Engineering College has presented the paper title **Machine Learning: Bone Fracture Detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K24 [NEC-ICAIEA-2K24], Organized by Department of Computer Science and Engineering, CSE(AI),IT,CSE(AIML),CSE(DS),CSE(CS) and MCA in Association with CSI on 12th and 13th April 2024 at NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS), Narasaraopet, A.P., India.

Convenor	Chief-Convenor	Principal, Patron
Dr.S.V.N.Srinivasu	Dr.S.N.Tirumala Rao	Dr. M. Sreenivasa Kumar



NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on

Artificial Intelligence and Its Emerging Areas

NEC-ICAIEA-2K24

12th & 13th April, 2024

Organized by Departments of CSE, IT, CSE(AI), CSE(AI&ML), CSE(DS), CSE(CS) & MCA in Association with CSI

Paper ID
NECICAIEA-2K24-129

Certificate of Presentation

This is to Certify that Sumanth Thota, Narasaraopeta Engineering College has presented the paper title **Machine Learning: Bone Fracture Detection** in the International Conference on Artificial Intelligence and Its Emerging Areas-2K24 [NEC-ICAIEA-2K24], Organized by Department of Computer Science and Engineering, CSE(AI),IT,CSE(AIML),CSE(DS),CSE(CS) and MCA in Association with CSI on 12th and 13th April 2024 at NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS), Narasaraopet, A.P., India.

[illegible]
Convenor
Dr.S.V.N.Srinivasu

[illegible]
Chief-Convenor
Dr.S.N.Tirumala Rao

[illegible]
Principal, Patron
Dr. M. Sreenivasa Kumar