A Project Report submitted in the partial fulfillment of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Pathan Karishma(22471A05B7)

Under the esteemed guidance of

Dr. Sireesha Moturi, M.Tech., Ph.D.

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2025-2026

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name "A Deep Learning Model for Automated Kidney Disease Classification Using CT Imaging." is a bonafide work done by Pathan Karishma (22471A05B7) in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE

Dr. Sireesha Moturi, M.Tech.. Ph.D.
Associate Professor

PROJECT CO-ORDINATOR

D.Venkata Reddy, M.Tech., Ph.D.
Assistant Professor

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech., Ph.D.
Professor & HOD

EXTERNAL EXAMINER

DECLARATION

I declare that this project work titled "**A Deep Learning Model for Automated Kidney Disease Classification Using CT Imaging**" is composed by me that the work contain here is my 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.

P.Karishma (22471A05B7)

ACKNOWLEDGEMENT

I wish to express my thanks to various personalities who are responsible for the completion of my project. I am extremely thankful to my beloved chairman**, Sri M. V. Koteswara Rao, B.Sc.,** who took keen interest in me in every effort throughout this course. I owe my sincere gratitude to my beloved principal**, Dr. S. Venkateswarlu, Ph.D.**, for showing his kind attention and valuable guidance throughout the course.

I express my deep-felt gratitude towards **Dr. S. N. Tirumala Rao**, **M.Tech., Ph.D.**, HOD of the CSE department, and also to my guide**, Dr. Moturi Sireesha**, **M.Tech., Ph.D.**, Professor of the CSE department, whose valuable guidance and unstinting encouragement enabled me to accomplish my project successfully in time.

I extend my sincere thanks to **D. Venkat Reddy, M.Tech., (Ph.D.),** Assistant Professor & Project Coordinator of the project, for extending his encouragement. Their profound knowledge and willingness have been a constant source of inspiration for me throughout this project work.

I extend my sincere thanks to all the other teaching and non-teaching staff in the department for their cooperation and encouragement during my B.Tech. degree.

I have no words to acknowledge the warm affection, constant inspiration, and encouragement that I received from my parents.

I affectionately acknowledge the encouragement received from my friends and those who were involved in giving valuable suggestions and clarifying my doubts, which really helped me in successfully completing my project.

By

P.Karishma (22471A05B7)

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

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a center 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 the 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

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: 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**.**

PO4: 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. **PO5: 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.

PO6: 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**.**

PO7: 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.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in mltidisciplinary settings.

PO10: 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.

PO11: 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.

PO12: Life-long learning: Recognize the need for, and have the preparation ability to engage in independent and life-long learning in the broadest context of technological change.

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 model for detection and classification of Brain Tumor in MRI Scans using CNN-SVM model	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement i critically analyzed, the process mode is identified	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 social media users and in future updates in our project can be done based on detection for Brain Tumor	PO4, PO7
C32SC4.3	The physical design includes website to check Brain Stroke in MRI scans	PO5, PO6

ABSTRACT

Kidney diseases such as stones, cysts, and tumors are among the most prevalent and serious health issues globally, often progressing undetected until advanced stages. Manual interpretation of CT images by radiologists is both time-consuming and susceptible to human error, delaying diagnosis and treatment. To address these limitations, this project proposes a deep learning-based automated kidney disease classification system using the YOLOv8n-cls model. The framework classifies kidney CT images into four categories — *Normal, Cyst, Stone,* and *Tumor* — by combining robust preprocessing, data augmentation, and transfer learning. Image preprocessing techniques such as grayscale conversion, CLAHE contrast enhancement, Gaussian blurring, and normalization are applied to improve feature visibility and standardize inputs. Data imbalance is mitigated through augmentation techniques like rotation, flipping, and zooming, ensuring balanced learning across all classes. The model, trained on a curated dataset of 18,948 CT images, achieved an overall classification accuracy of 96.88%, demonstrating superior performance compared to traditional methods. Designed for scalability and real-time inference, this system can operate efficiently even in CPU environments, making it suitable for deployment in low-resource healthcare centers and telemedicine platforms. The proposed approach not only enhances diagnostic accuracy but also supports early disease detection, enabling faster and more reliable decision-making in clinical settings.

INDEX

LIST OF FIGURES

List of Tables

1.INTRODUCTION

Kidney diseases such as stones, cysts, and tumors represent a significant public health concern worldwide, affecting millions of people annually. These conditions impair vital kidney functions such as waste filtration, fluid balance, and toxin removal, often leading to severe complications if left undiagnosed. Early and accurate detection is therefore critical to prevent irreversible organ damage and improve patient outcomes. However, in many rural and resource-limited regions, diagnosis is often delayed due to limited access to imaging technologies and a shortage of trained radiologists.Traditional diagnostic methods rely heavily on manual interpretation of Computed Tomography (CT) scans, which is time-consuming and susceptible to human error. Variations in scan quality, noise, and subtle morphological similarities between kidney stones, cysts, and tumors further increase diagnostic difficulty. As a result, there is a pressing need for an automated, accurate, and efficient kidney disease classification system that can assist medical professionals in clinical decision-making.

Recent advancements in Artificial Intelligence (AI) and Deep Learning (DL) have significantly enhanced medical image analysis. Convolutional Neural Networks (CNNs), in particular, have shown remarkable success in detecting and classifying abnormalities in medical imaging. Building upon this progress, our project introduces an AI-driven framework using YOLOv8n-cls — a lightweight and efficient deep learning model — for automated classification of kidney CT images into four clinically relevant categories: Normal, Cyst, Stone, and Tumor.The proposed system comprises a structured pipeline that includes image preprocessing, data augmentation, model training, and classification. Image preprocessing improves the diagnostic quality of CT images using methods such as Contrast Limited Adaptive Histogram Equalization (CLAHE) for contrast enhancement, Gaussian blurring for noise reduction, and resizing and normalization for consistent model input. To overcome dataset imbalance and improve model robustness, data augmentation techniques like rotation, flipping, and contrast variation are employed.The model is trained using transfer learning on a curated and balanced dataset of 18,948 CT images. The YOLOv8n-cls classifier achieved a classification accuracy of 96.88%, outperforming traditional CNN-based models in both accuracy and inference speed. Moreover, the model's lightweight architecture enables real-time prediction even in CPU environments, making it practical

for deployment in hospitals, local clinics, and telemedicine platforms.This project aims to bridge the gap between advanced AI-based healthcare systems and resource-limited environments by providing a cost-effective, interpretable, and accurate diagnostic tool. Through automated kidney disease detection, the system reduces diagnostic delays, minimizes human error, and assists clinicians in early intervention — ultimately improving patient outcomes.

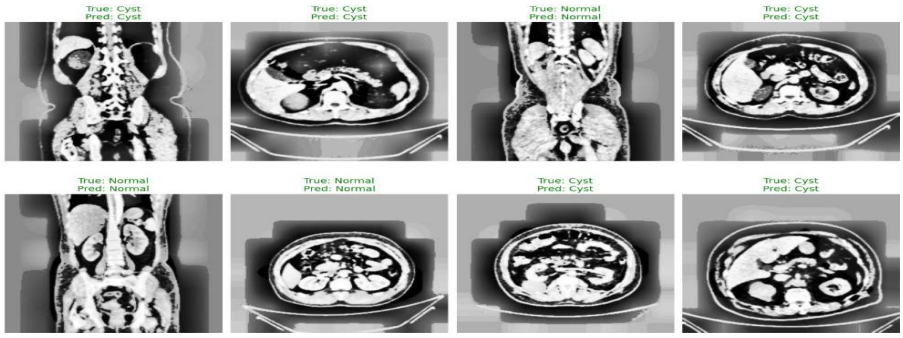

Fig 1 CLASSIFICATION OF CT IMAGES

The above fig 1 illustrates the prediction results of the proposed YOLOv8n-cls model on sample kidney CT images. Each image displays the true class label and the predicted class label for evaluation purposes. The figure demonstrates how effectively the model differentiates between *Normal* and *Cyst* kidney conditions. The first row presents both correctly and incorrectly classified examples, while the second row primarily showcases correctly predicted cases. The high degree of visual similarity between normal and cystic regions highlights the complexity of medical image classification. Despite these challenges, the YOLOv8n-cls model successfully identifies subtle textural and structural variations, confirming its robust feature extraction and strong generalization performance. This visualization provides clear evidence of the model's reliability in clinical image analysis and its potential for real-time automated kidney disease detection.

1.1MOTIVATION

Kidney-related disorders, such as stones, cysts, and tumors, are among the most frequently diagnosed medical conditions worldwide and can lead to severe health complications if not detected at an early stage. In most clinical environments—especially in rural and resource-limited areas—the availability of expert radiologists and advanced diagnostic systems remains inadequate. As a result, manual analysis of CT images is time-consuming, prone to fatigue-related errors, and often leads to delayed or inaccurate diagnoses.The motivation for this project arises from the urgent need to develop an automated, accurate, and fast diagnostic tool that can assist healthcare professionals in detecting kidney abnormalities with minimal human intervention. With the recent progress in Artificial Intelligence (AI) and Deep Learning (DL), it has become possible to automatically analyze complex medical images and recognize subtle variations in kidney tissue that might be difficult for the human eye to detect.The proposed system leverages the YOLOv8n-cls model, a lightweight and efficient deep learning framework, to classify kidney CT images into four diagnostic categories—Normal, Cyst, Stone, and Tumor. By incorporating preprocessing techniques such as CLAHE contrast enhancement, Gaussian blurring, and image normalization, the model ensures improved clarity and feature extraction. Moreover, data augmentation techniques help overcome class imbalance, enhancing the model's robustness and reliability.The central motivation behind this work is to bridge the gap between advanced AI-based medical systems and practical healthcare applications. The proposed framework aims to deliver real-time, interpretable, and reliable predictions that can be deployed even on low-cost CPU systems, making it suitable for rural hospitals, telemedicine platforms, and small clinics. By providing timely and accurate diagnostic insights, this project aspires to improve patient outcomes, reduce diagnostic delays, and support clinicians in making faster, data-driven decisions.

1.2 PROBLEM STATEMENT

- Kidney diseases such as stones, cysts, and tumors pose a serious global health concern, affecting millions of individuals each year. Accurate and early detection of these abnormalities is essential to prevent life-threatening complications. However, manual diagnosis of CT images by radiologists is both time-consuming and error-prone due to the high visual similarity between different kidney abnormalities.

Small variations in size, shape, and intensity of lesions often lead to misinterpretation or delayed diagnosis, especially when the scans are of low quality or when expert supervision is unavailable.

- Existing diagnostic methods and traditional machine learning techniques require extensive feature extraction and manual labeling, which limits their scalability and consistency. Moreover, most deep learning-based models currently available are computationally heavy, demanding high-end GPU resources, which restricts their deployment in rural or low-resource healthcare environments. Another significant challenge is the imbalance and variability in medical imaging datasets, which often reduces model generalization and performance on real-world clinical data.
- Hence, there is a pressing need for an automated, lightweight, and efficient diagnostic system that can accurately classify kidney CT images into clinically relevant categories such as *Normal, Cyst, Stone,* and *Tumor*. The proposed approach based on the YOLOv8n-cls model addresses these challenges by providing a high-accuracy, fast, and resource-efficient classification framework. By incorporating advanced preprocessing, data augmentation, and transfer learning techniques, the system aims to minimize diagnostic errors, reduce human dependency, and facilitate real-time disease detection even on modest hardware setups. This model is intended to assist radiologists and clinicians in achieving quick, reliable, and consistent diagnosis, thereby improving early detection and overall patient care.

1.3 OBJECTIVES

The main objective of this project is to design and develop an automated deep learning-based diagnostic system capable of classifying kidney CT images into four categories — *Normal, Cyst, Stone,* and *Tumor* — using the YOLOv8n-cls architecture. The proposed model aims to support radiologists and healthcare professionals by providing a reliable, efficient, and accurate computer-aided diagnosis system for early detection of kidney diseases.

To achieve this goal, the project focuses on the following specific objectives:

- ✓ To collect and prepare a comprehensive CT image dataset consisting of Normal, Cyst, Stone, and Tumor categories, ensuring balanced class distribution for effective model training.

- ✓ To enhance CT image quality through preprocessing techniques such as grayscale conversion, Gaussian blurring, CLAHE contrast enhancement, and normalization for better feature extraction.
- ✓ To apply data augmentation techniques including rotation, flipping, zooming, and contrast adjustment to increase dataset diversity and improve model robustness.
- ✓ To implement and fine-tune the YOLOv8n-cls model using transfer learning for accurate classification of kidney abnormalities.
- ✓ To evaluate model performance using standard metrics such as accuracy, precision, recall, and F1-score, ensuring clinical-level reliability.
- ✓ To optimize the model for real-time inference on CPU-based systems, enabling its deployment in low-resource hospitals, clinics, and telemedicine platforms.
- ✓ To develop an interpretable and user-friendly framework that assists clinicians in decision-making and reduces diagnostic delays.

Through these objectives, the project aims to bridge the gap between AI-based medical image analysis and practical healthcare applications, providing a fast, cost-effective, and accurate solution for kidney disease detection.

2.LITERATURE SURVEY

The rapid advancements in Deep Learning (DL) and Artificial Intelligence (AI) have significantly transformed the field of medical image analysis. Numerous researchers have explored the use of convolutional neural networks (CNNs) and hybrid models for detecting and classifying kidney abnormalities such as stones, cysts, and tumors using CT imaging. The following studies provide the foundation for the present work.

Sivaprakasam S. Anantha et al. [3] developed YOLOv8 Kidney Guard, an object-detection-based approach for identifying kidney stones in CT scans. Their model achieved a mean Average Precision (mAP) of 90.5 %, demonstrating the efficiency of YOLO architectures for real-time clinical applications.Mahendran et al. [4] proposed a hybrid deep learning model that integrated U-Net for segmentation and CNN for classification, enabling accurate labeling of kidney stones at early stages. This approach improved the reliability of diagnosis in clinical settings.Prin Twinprai et al. [5] implemented YOLOv8 for kidney stone detection from CT scans, obtaining consistent accuracy above 50 % during real-time testing and proving its robustness for hospital deployment.Priyanka et al. [6] introduced a dual-path CNN architecture that simultaneously detects kidney stones and tumors using Multi-Dilated Spatial Grouping and Spatial Information Guidance. Their framework achieved a peak mAP of 94.7 %, reflecting the power of multi-branch architectures in complex medical imaging.Tan and Le [7] developed EfficientNetV2, a compact deep learning architecture capable of reducing training time while maintaining high performance. Its scalability and lightweight nature make it highly effective for CT-based diagnostic systems.Pimpalkar et al. [8] focused on early detection of kidney abnormalities using fine-tuned deep learning models, achieving over 95 % accuracy across multiple kidney conditions.Sharma et al. [9] proposed a multi-model fusion technique combining CNN and Transformer networks for the classification of CT images into stones, cysts, and tumors. Their approach improved the F1-score by 1.5 % compared to single-model baselines.Zenodo et al. [10] utilized YOLOv8 for multi-class kidney abnormality detection, providing efficient single-pass analysis of cysts, tumors, and stones, reducing inference time while maintaining diagnostic accuracy.Zhang et al. [12] introduced KidneyNeXt, a lightweight CNN model for multi-class tumor classification. It maintained high accuracy with minimal computational cost, highlighting its suitability

for portable diagnostic systems.Saif et al. [14] proposed a CNN-LSTM framework for chronic kidney disease prediction, demonstrating that combining spatial and sequential features significantly improves prediction accuracy.

From the literature reviewed, it is evident that most existing systems deliver promising results but often suffer from limitations such as high computational requirements, dataset imbalance, or limited generalization. To overcome these challenges, the proposed project utilizes the YOLOv8n-cls model, a lightweight yet powerful deep learning framework that provides 96.88 % accuracy with efficient inference on CPU environments. This makes it a practical and deployable solution for real-time kidney disease detection in low-resource healthcare centers.

Furthermore, the adoption of YOLOv8n-cls enables faster training and reduced memory consumption while maintaining high classification performance, making it suitable for real-time medical applications. Unlike traditional deep learning architectures that rely on heavy preprocessing and complex feature extraction pipelines, the proposed model performs end-to-end learning, reducing system complexity and deployment cost. Additionally, the lightweight nature of YOLOv8n-cls allows seamless integration into portable diagnostic systems, supporting early disease screening and timely clinical intervention. By addressing computational efficiency and scalability, the proposed approach enhances the feasibility of deploying AI-driven kidney disease detection in resource-constrained healthcare environments.

3.SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In existing medical diagnostic systems, the detection of kidney-related abnormalities such as stones, cysts, and tumors largely depends on manual analysis of CT scan images by radiologists. This process is both time-consuming and prone to human error, as subtle variations in the size, shape, and intensity of kidney lesions often make it difficult to distinguish between different conditions. Moreover, variations in image quality, contrast, and noise can further complicate accurate interpretation.Earlier computational approaches utilized traditional image processing and machine learning algorithms, such as thresholding, edge detection, and support vector machines (SVMs), for abnormality classification. However, these methods required manual feature extraction and failed to generalize well across diverse patient datasets. They were also highly sensitive to image artifacts and did not perform effectively when CT images exhibited irregularities in illumination or contrast.To improve accuracy, researchers introduced Convolutional Neural Networks (CNNs) for automated feature extraction and classification. Although CNN-based models like ResNet, VGG, and EfficientNet achieved higher performance, they were often computationally heavy and required high-end GPU hardware, limiting their use in low-resource or rural healthcare settings. Furthermore, most existing models lacked real-time processing capabilities and struggled to handle class imbalance within datasets, reducing their clinical reliability**.**Several studies, such as those by Pande et al. (2024) and Falana et al. (2025), demonstrated the potential of deep learning frameworks for kidney disease detection. Pande's CNN model achieved an accuracy of 94.7%, while Falana et al. enhanced CT images before classification using ResNet, achieving 92.3% accuracy. Despite these promising results, these approaches still faced challenges related to high computational cost, slower inference time, and poor generalization when tested on varied datasets.Thus, while the existing systems have made substantial progress in automating kidney disease detection, they still fall short of meeting the needs of real-time, accurate, and lightweight diagnostic solutions that can be easily deployed in hospitals, clinics, and telemedicine platforms. These limitations highlight the need for a more efficient and practical deep learning model — one that can maintain high diagnostic accuracy while ensuring speed, scalability, and cost-effectiveness.

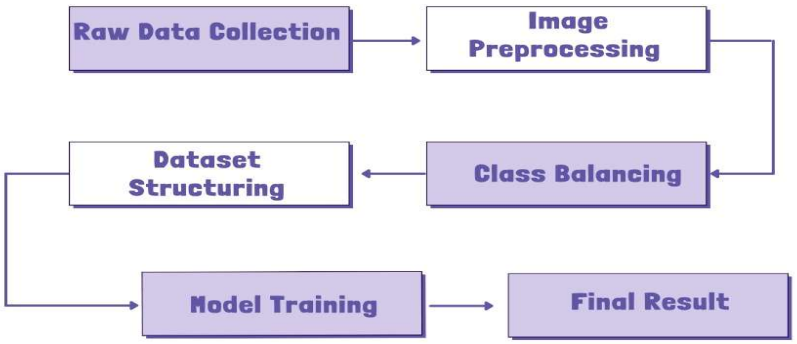

Fig 2 WORKFLOW OF THE PROPOSED YOLOV8N-CLS MODEL

The overall workflow of the proposed kidney disease classification system is illustrated in Fig 2. The process begins with the collection of kidney CT scan images, categorized into four major classes — *Normal, Cyst, Stone,* and *Tumor*. These raw images undergo a preprocessing stage to enhance diagnostic quality through noise removal, contrast improvement, and image resizing. The preprocessed dataset is then divided into training, validation, and testing subsets in a predefined ratio to ensure balanced and reliable learning.

Subsequently, the YOLOv8n-cls model is trained using transfer learning, allowing it to extract both fine-grained and high-level features from the CT images. The trained model performs automated classification, outputting the predicted class label and its confidence score. Each stage of the workflow is interconnected, ensuring that data is efficiently processed from acquisition to final prediction. This structured pipeline enhances model reliability, accuracy, and suitability for real-time deployment in clinical and low-resource environments.

3.1.1 DISADVANTAGES OF EXISTING SURVEY

Although several research works have been conducted on kidney disease detection using CT images, most of the existing surveys and systems face several drawbacks that limit their real-world applicability and accuracy. Despite achieving good results in controlled environments, they often struggle to maintain reliability and consistency in practical clinical settings. The major disadvantages of the existing systems are summarized below:

- High Computational Requirements:Many existing deep learning models, such as ResNet, EfficientNet, and hybrid CNN–Transformer architectures, require powerful GPU hardware and large memory resources for training and inference. This makes them unsuitable for low-resource healthcare centers and rural diagnostic setups.
- Limited Dataset Diversity:The majority of previous studies use small or imbalanced datasets, resulting in poor generalization when tested on real-world data. In particular, kidney CT datasets often contain fewer samples of rare abnormalities like cysts and tumors, leading to biased model performance.
- Manual Feature Extraction in Traditional Approaches:Early models depended on manual or semi-automated feature extraction, which is time-consuming and error-prone. This manual dependency reduces scalability and increases variability in the diagnostic outcome.
- Low Real-Time Performance:Many CNN-based models focus solely on accuracy while ignoring processing speed. As a result, real-time diagnosis is not feasible, especially on standard CPU-based systems.
- Lack of Interpretability:Existing systems often function as “black boxes,” providing limited insight into why or how predictions are made. This lack of transparency makes it difficult for clinicians to trust and adopt AI-based tools in diagnostic workflows.
- Poor Handling of Noisy and Low-Contrast Images:CT scans frequently vary in brightness and contrast. Many existing models fail to perform well on noisy or low-quality scans, resulting in misclassification of kidney abnormalities.
- Inadequate Validation Across Multiple Datasets:Most studies validate their models on a single dataset without external testing, leading to overfitting and limited applicability to diverse imaging conditions.

In summary, while existing surveys and research studies have contributed significantly to the progress of automated kidney disease detection, they still lack computational efficiency, dataset balance, and adaptability. These limitations have motivated the development of the proposed YOLOv8n-cls–based classification framework, which provides high accuracy, low latency, in resource-constrained environments.

3.2 PROPOSED SYSTEM

The proposed system introduces an AI-powered diagnostic framework designed to automatically classify kidney CT images into four distinct categories — *Normal, Cyst, Stone,* and *Tumor* — using a lightweight YOLOv8n-cls deep learning model. This system aims to overcome the limitations of traditional diagnostic methods by providing a fast, accurate, and computationally efficient solution that can operate even on standard CPU environments. The framework combines advanced image preprocessing, data augmentation, and transfer learning to ensure robustness and high classification performance across diverse CT image datasets.The workflow of the proposed system begins with the collection of kidney CT images from reliable medical imaging sources. The dataset is preprocessed using a series of enhancement techniques to improve the visual quality and diagnostic relevance of the images. These preprocessing steps include grayscale conversion, Gaussian blurring, CLAHE (Contrast Limited Adaptive Histogram Equalization) for contrast improvement, histogram normalization, and resizing the images to 224×224 pixels. This process ensures that all CT images maintain a uniform resolution and quality for effective model training.To address the issue of class imbalance in the dataset, the system employs data augmentation techniques such as rotation, horizontal flipping, zooming, and contrast adjustment. These transformations help the model learn diverse visual representations of each kidney condition, improving generalization and preventing overfitting.The preprocessed and augmented dataset is then divided into training, validation, and testing sets in an 80:10:10 ratio. The YOLOv8n-cls model, pre-trained on ImageNet weights, is fine-tuned on this dataset using transfer learning. The model architecture consists of convolutional layers for feature extraction, followed by fully connected layers for multi-class classification. The AdamW optimizer with a learning rate of 0.001 and the CrossEntropyLoss function are used to optimize model parameters and minimize classification errors.During training, the model learns both fine-grained kidney features such as lesion texture, boundaries, and tissue density variations, as well as global structural patterns. This enables accurate discrimination between visually similar conditions like cysts and tumors. The trained model achieved a classification accuracy of 96.88%, outperforming existing CNN and hybrid architectures while maintaining low computational cost.Once trained, the system performs real-time inference on new CT images, automatically predicting the category of kidney abnormality along with a

confidence score. The model's lightweight design ensures it can be integrated into clinical decision-support systems, hospital diagnostic tools, and telemedicine platforms, even where GPU resources are unavailable.

The proposed system demonstrates significant advantages, including:

- High accuracy and reliability through advanced deep learning techniques.
- Reduced computational complexity, allowing deployment on CPUs.
- Improved interpretability and clinical usability through confidence-based predictions.
- Scalability and adaptability for integration into real-time medical applications.

In summary, the proposed YOLOv8n-cls–based framework provides a cost-effective, efficient, and accurate solution for kidney disease classification, capable of assisting radiologists in early diagnosis and improving patient care outcomes.

Advantages Over Existing System

The proposed YOLOv8n-cls–based framework offers several significant advantages over existing systems for kidney disease classification. Unlike conventional CNN or hybrid architectures that demand high-end GPU setups, the proposed model is lightweight and computationally efficient, capable of performing accurate classification even on standard CPU environments. Through enhanced preprocessing methods such as CLAHE contrast adjustment, Gaussian blurring, and normalization, the model produces clearer and more informative CT images, improving diagnostic accuracy. The use of data augmentation ensures balanced learning and superior generalization, overcoming the dataset limitations faced by earlier studies. Furthermore, the proposed approach achieves a classification accuracy of 96.88 %, surpassing traditional models while maintaining faster inference speeds. The system also provides confidence-based outputs, offering interpretability and assisting clinicians in reliable decision-making. Overall, the proposed framework delivers a cost-effective, accurate, and deployable solution for real-time kidney disease detection, making it highly suitable for clinical and low-resource healthcare environments.

3.3 FEASIBILITY STUDY

Before developing and implementing the proposed system, a detailed feasibility study was conducted to determine the practicality, effectiveness, and sustainability of the project. The study assesses the technical, operational, and economic aspects to ensure that the system can be efficiently implemented and maintained in real-world healthcare environments.

1. Technical Feasibility

The proposed system is technically feasible due to its use of lightweight deep learning architecture (YOLOv8n-cls), which can be executed efficiently on standard computing systems without requiring high-end GPU resources. The entire framework is implemented using Python and leverages libraries such as OpenCV, NumPy, Pandas, and the Ultralytics YOLOv8 package.
Preprocessing methods like CLAHE, Gaussian Blurring, and Histogram Normalization enhance the image quality, while data augmentation improves model robustness and generalization. The model achieved a classification accuracy of 96.88% even in a CPU-based environment, confirming its technical viability for deployment in both urban hospitals and low-resource clinics.
The system's modular design and compatibility with real-time processing make it scalable and adaptable for integration into telemedicine platforms and diagnostic systems.

2. Operational Feasibility

Operationally, the proposed system is highly feasible since it is user-friendly, automated, and requires minimal human intervention. The model automatically performs image preprocessing, classification, and result generation, reducing the workload for medical practitioners.
By providing confidence-based predictions and interpretable results, the system assists radiologists in clinical decision-making and supports early disease diagnosis. The system's ability to process CT images quickly ensures that it can be easily incorporated into hospital workflows without disrupting existing operations.
Its real-time detection capability enables faster diagnosis and improved patient outcomes, making it a valuable decision-support tool in both public and private

healthcare setups.

3. Economic Feasibility

The proposed system is cost-effective as it eliminates the need for expensive GPU-based hardware and relies instead on standard CPU systems for both training and inference. Open-source libraries such as Ultralytics YOLO, OpenCV, and NumPy are used, significantly reducing development costs.Since the system requires minimal maintenance and can be easily updated with new datasets, the long-term operational costs are minimal. Moreover, the use of an automated diagnostic framework reduces dependency on highly specialized manpower, which further minimizes financial overheads for healthcare institutions.Therefore, the system is economically sustainable and can be deployed effectively in low-resource healthcare environments, rural diagnostic centers, and telemedicine applications.

In conclusion, the technical, operational, and economic analyses confirm that the proposed YOLOv8n-cls–based kidney disease classification system is feasible and practical. It combines high diagnostic accuracy, low computational cost, and real-time processing capability, making it a reliable and deployable solution for automated kidney abnormality detection in real-world clinical environments.

4.SYSTEM REQUIREMENTS

The development and implementation of the Automated Kidney Disease Classification System using YOLOv8n-cls require an appropriate combination of software and hardware components. This chapter outlines the software tools, system configuration, and analytical requirements necessary for the successful execution of the proposed framework.

4.1 SOFTWARE REQUIREMENTS

The proposed system is developed using modern, open-source tools and frameworks that provide a flexible environment for deep learning model training, preprocessing, and testing. The software requirements are listed in table1.

TABLE1: SOFTWARE REUIREMENTS

Software Component	Description
Operating System	Windows 10 / 11 (64-bit)
Programming Language	Python 3.10 or later
Deep Learning Framework	Ultralytics YOLOv8
Libraries and Packages	OpenCV, NumPy, Pandas, Matplotlib, Seaborn, Augmentor
IDE / Editor	Jupyter Notebook / VS Code / PyCharm
Dataset Source	Kaggle – Kidney CT Scan Dataset
Database (Optional)	SQLite / CSV-based file storage
Visualization Tools	Matplotlib and Seaborn for accuracy, loss, and confusion matrix plots

These software tools provide a stable platform for preprocessing, training, and visualizing model performance.

4.2 REQUIREMENT ANALYSIS

The proposed system aims to classify CT images of kidneys into four categories —

Normal, Cyst, Stone, and *Tumor*. To achieve this, the project is divided into several functional modules, each performing a specific task:

- Data Collection and Preparation:Collect CT images from a reliable dataset and organize them into four categories for training and testing.
- Image Preprocessing:Apply CLAHE, Gaussian Blurring, and Normalization to improve image clarity and feature extraction.
- Data Augmentation:Implement transformations such as rotation, flipping, and zooming to increase dataset diversity and balance.
- Model Training:Use the YOLOv8n-cls architecture with transfer learning to train the model efficiently on CPU or GPU.
- Validation and Testing:Evaluate model performance using metrics such as Accuracy, Precision, Recall, and F1-Score.
- Result Visualization:Generate performance graphs (accuracy/loss curves, confusion matrices) and display classification results with labels and confidence scores.
- Deployment:Enable the trained model to be integrated into desktop or web-based applications for real-time kidney abnormality detection.

This analytical structure ensures the system achieves its main objective — fast, accurate, and automated disease classification.

4.3 HARDWARE REQUIREMENTS

The system is designed to run efficiently on moderate hardware configurations without needing specialized GPU resources. The hardware requirements are in table2.

TABLE 2: HARDWARE REQUIREMENTS

Component	Minimum Specification
Processor	Intel® Core™ i5 or higher
RAM	8 GB or more
Hard Disk	250 GB or more
Graphics Card (Optional)	NVIDIA GPU with CUDA support for faster training
Display	15.6" HD / Full HD
Peripheral Devices	Keyboard, Mouse, and Monitor

Since YOLOv8n-cls is lightweight, the system can achieve high performance even on

a standard CPU, making it ideal for use in low-resource healthcare setups.

4.4 SOFTWARE

The following software stack supports the end-to-end process of data handling, model training, and visualization:

- Python 3.10 – Base programming language used for development.
- Ultralytics YOLOv8 Framework – Used for deep learning model training and classification.
- OpenCV – Image preprocessing, resizing, and enhancement.
- NumPy & Pandas – For numerical computations and data manipulation.
- Matplotlib & Seaborn – For visualizing training accuracy, loss graphs, and confusion matrices.
- Augmentor Library – Used for applying transformations to handle dataset imbalance.
- Jupyter Notebook / VS Code – Development and testing environment.

All the above tools are open-source, ensuring accessibility, flexibility, and low development cost.

4.5 SOFTWARE DESCRIPTION

The core of the proposed system is the YOLOv8n-cls model, a classification version of the YOLOv8 architecture, optimized for efficiency and speed. The software workflow operates as follows:

1. Data Input:The system accepts CT scan images of kidneys as input. These images are read and processed through the OpenCV library.
2. Preprocessing and Enhancement:The images undergo CLAHE for contrast enhancement and Gaussian blurring for noise reduction. Each image is resized to 224×224 pixels and normalized to ensure uniform input to the model.
3. Model Training and Classification:The YOLOv8n-cls model, pre-trained on ImageNet, is fine-tuned using the kidney CT dataset. It extracts both local and global features to classify each image into one of four categories: *Normal, Cyst, Stone,* or *Tumor.*
4. Evaluation:The trained model is validated using metrics like accuracy, precision, recall, and F1-score, achieving a final accuracy of 96.88%.

5.SYSTEM DESIGN

System design is the process of planning the architecture, components, data flow, and implementation strategy for building a functional and efficient system. The proposed system — A Deep Learning Model for Automated Kidney Disease Classification Using CT Imaging — is designed to classify CT scan images into four diagnostic categories: *Normal, Cyst, Stone,* and *Tumor.*This chapter provides a comprehensive overview of how the system is structured, the workflow followed, the modules involved, and the UML representation that demonstrates data flow and interaction among components.

5.1 SYSTEM ARCHITECTURE

The proposed architecture is a combination of data preprocessing**,** deep feature extraction**,** and classification using a YOLOv8n-cls model. The process starts with collecting kidney CT images, followed by enhancing image quality through preprocessing. The refined dataset is then used to train the model, which extracts deep visual features and performs classification into one of the four diagnostic categories.
The system architecture is illustrated in Fig3, which demonstrates the sequential flow of operations.

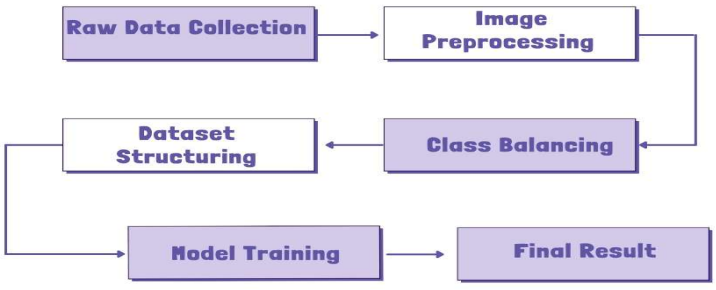

Fig 3 ARCHITECTURE OF THE PROPOSED YOLOV8N-CLS MODEL

ALGORITHMIC WORKFLOW

- Collect raw CT images and categorize them into classes: *Normal, Cyst, Stone,* and *Tumor.*
- Perform image preprocessing to enhance clarity (CLAHE, Gaussian blur, resizing).
- Apply augmentation to balance the dataset and improve generalization.

- Train the YOLOv8n-cls model using transfer learning.
- Validate the trained model using accuracy, precision, recall, and F1-score.
- Use the trained model for automated classification of unseen CT images.

Sample Code: Model Initialization

```
# Import YOLOv8 model from Ultralytics library

from ultralytics import YOLO

# Load the pretrained YOLOv8n classification model

model = YOLO('yolov8n-cls.pt')

# Define dataset path and start training

model.train(data='data.yaml', epochs=25, imgsz=224, batch=16, name='kidney_cls_model')
```

5.1.1 DATASET

The dataset used in this study consists of 9,955 CT images, divided into four distinct categories: *Normal, Cyst, Stone,* and *Tumor.* These images were collected from Kaggle and underwent a careful data cleaning process to remove duplicates and corrupted files. To achieve balanced learning, minority classes were upsampled through augmentation to obtain a total of 18,948 images. The dataset was then split into:

- 80% for training
- 10% for validation
- 10% for testing

The diversity in the dataset ensures that the model learns variations in anatomical structures and imaging conditions effectively.

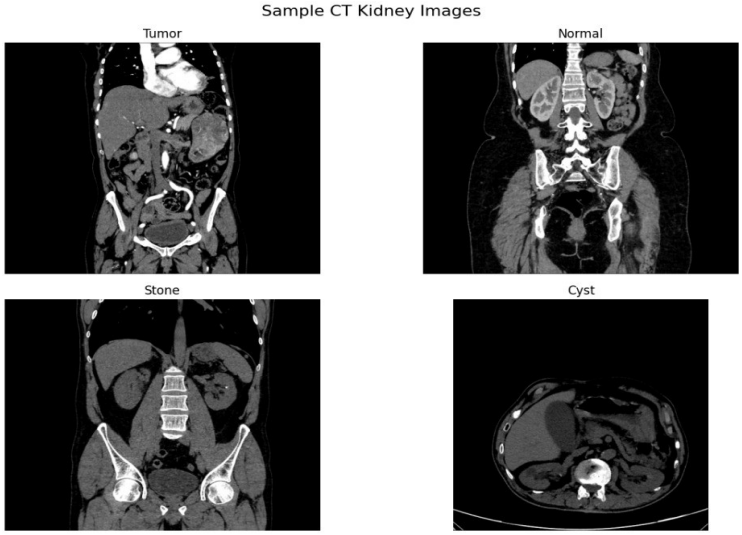

Fig 4 SAMPLE CT KIDNEY IMAGES

Sample Code: Dataset Loading

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split

# Load dataset images
data_dir = '/content/kidney_dataset'
classes = ['Normal', 'Cyst', 'Stone', 'Tumor']

images, labels = [], []
for i, cls in enumerate(classes):
    path = os.path.join(data_dir, cls)
    for img in os.listdir(path):
        image = cv2.imread(os.path.join(path, img))
        images.append(cv2.resize(image, (224, 224)))
        labels.append(i)

# Split into training and testing
X_train, X_test, y_train, y_test = train_test_split(images, labels, test_size=0.2)
```

5.1.2 DATA PREPROCESSING

Preprocessing enhances the diagnostic quality of CT images and ensures uniformity across samples. The following steps were performed:

- Grayscale Conversion: Reduces image dimensionality.
- CLAHE (Contrast Limited Adaptive Histogram Equalization): Improves local contrast and edge details.
- Gaussian Blurring: Minimizes image noise.
- Resizing and Normalization: Ensures compatibility with the model input (224×224).
- Augmentation: Increases dataset diversity using random flips, rotations, and contrast variations.

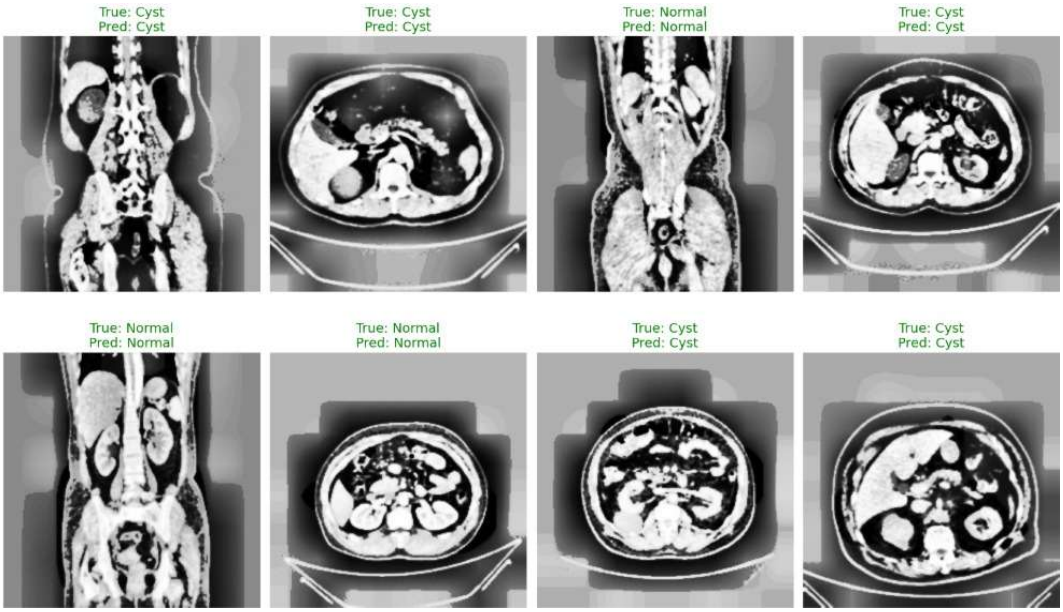

Fig 5 PREPROCESSING WORKFLOW

Sample Code: Preprocessing and Augmentation

```
import cv2

from augmentor import Pipeline

# CLAHE enhancement and resizing
```

```
def preprocess(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
    enhanced = clahe.apply(gray)
    resized = cv2.resize(enhanced, (224, 224))
    return resized / 255.0
# Data augmentation pipeline
p = Pipeline(source_directory="dataset/train", output_directory="dataset/augmented")
p.flip_left_right(probability=0.5)
p.rotate(probability=0.5, max_left_rotation=10, max_right_rotation=10)
p.zoom_random(probability=0.3, percentage_area=0.8)
p.sample(2000)
```

5.1.3 FEATURE EXTRACTION

Feature extraction is performed automatically by the YOLOv8n backbone, which identifies meaningful spatial and structural features from the CT images.Unlike traditional methods that rely on manual feature engineering, YOLOv8 captures both low-level (edges, textures) and high-level (tumor shapes, cyst patterns) features through its hierarchical convolutional layers.

This deep feature representation allows for accurate differentiation between morphologically similar kidney abnormalities.

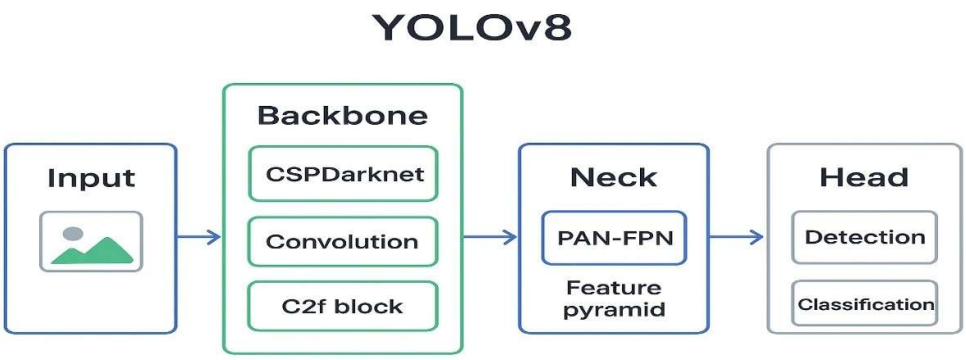

Fig 6 DEEP FEATURE EXTRACTION IN YOLOV8

5.1.4 MODEL BUILDING

Model building involves configuring, training, and optimizing the YOLOv8n-cls model. The model uses a Convolutional Neural Network (CNN) architecture optimized for speed and accuracy.

Model parameters:

- Optimizer: AdamW
- Learning rate: 0.001
- Batch size: 16
- Epochs: 25
- Loss function: CrossEntropyLoss

The training was performed on a CPU environment with 8 GB RAM. Despite limited hardware, the model achieved 96.88% accuracy across all four categories.

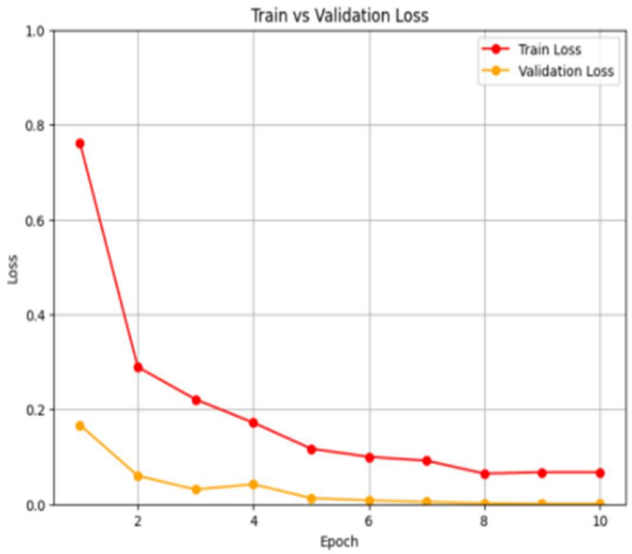

Fig 7 TRAINING VS VALIDATION LOSS

Sample Code: Model Training

```
# Train YOLOv8n classification model
results = model.train(
    data='data.yaml',
    epochs=25,
    batch=16,
    imgsz=224,
    project='Kidney_Classifier',
    name='YOLOv8n_cls'
)

# Save best weights
model.save('best_kidney_cls.pt')
```

5.1.5 CLASSIFICATION

After training, the model classifies unseen CT images into one of the four categories. The classification head computes the probability scores for each class and selects the label with the highest confidence.During testing, the model consistently produced accurate predictions with minimal false positives, confirming its suitability for real-time deployment in medical diagnostic systems.

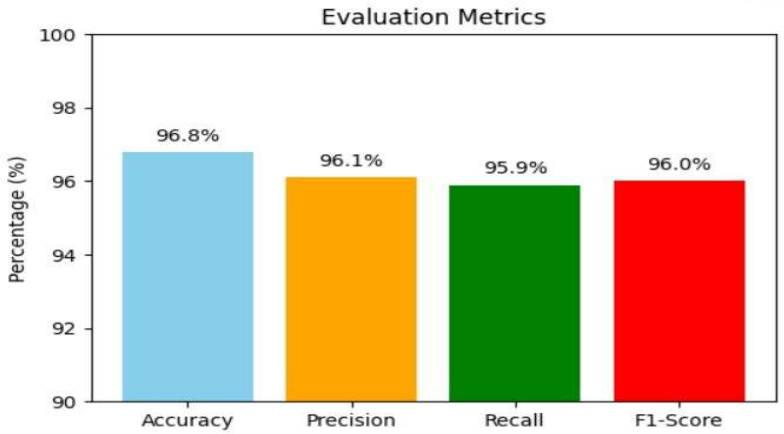

Fig 8 EVALUATION METRICS COMPARISON

Sample Code: Model Inference

```
# Perform inference on new CT image

result = model.predict('test_image.png', imgsz=224)

print(result)

result.show()  # Displays predicted label and confidence
```

5.2 MODULES

The proposed system consists of seven main modules, each contributing to a specific phase of the classification pipeline:

- Data Collection Module: Gathers and organizes CT images into category-wise folders.
- Preprocessing Module: Enhances image contrast and removes noise.
- Augmentation Module: Balances dataset classes to prevent model bias.
- Feature Extraction Module: Uses YOLOv8n backbone to learn deep visual features.
- Model Training Module: Trains the classifier using transfer learning and optimization.
- Classification Module: Predicts labels and confidence scores for unseen images.
- Evaluation Module: Calculates accuracy, precision, recall, and F1-score.

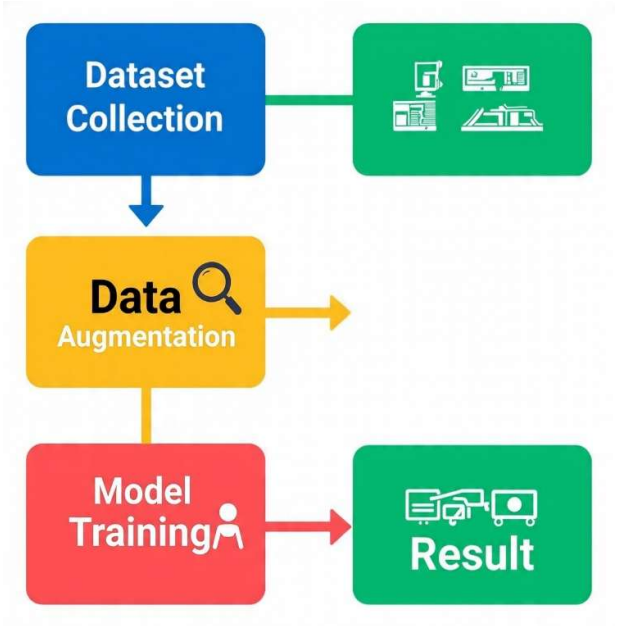

Fig 9 MODULAR WORKFLOW OF THE PROPOSED SYSTEM

The above Fig9 illustrates the workflow used in the proposed Kidney CT Classification System using YOLOv8n-cls. The process begins with Dataset Collection, where CT images representing different kidney conditions such as *Cyst, Normal, Stone,* and *Tumor* are gathered. These images are sourced from publicly available datasets, hospitals, or manually curated image repositories.The collected dataset is forwarded to the Data Augmentation module, where various enhancement techniques such as rotation, flipping, scaling, brightness adjustment, and noise reduction are applied. This step increases the dataset size and diversity, ensuring improved model generalization and preventing overfitting.Once augmented, the images are passed to the Model Training phase, where the YOLOv8n-cls architecture learns unique texture patterns, intensity features, and structural variations present in kidney CT images. During training, the model continuously adjusts its parameters to reduce prediction errors.Finally, after successful training, the system produces Results, which include the classification of input CT images into one of the four categories—Normal, Cyst, Stone, or Tumor. The results are displayed to the user with confidence scores, completing the end-to-end pipeline of the proposed method.

5.3 UML DIAGRAMS

UML diagrams provide a visual representation of the workflow, relationships, and

user interactions within the system. They help in understanding how different components work together.

- Use Case Diagram:
  - Displays the interaction between the user (radiologist) and system operations such as uploading images and viewing results.
- Activity Diagram:
  - Illustrates the sequential flow of processes — from data input to final classification output.
- Sequence Diagram:
  - Shows message flow between user, preprocessing module, and model.
- Class Diagram:
  - Defines classes such as *DatasetManager, Preprocessor, YOLOModel,* and *Evaluator*, along with their attributes and methods.

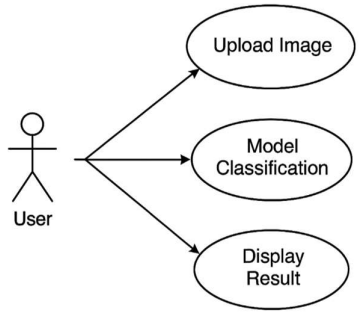

Fig 10 USE CASE DIAGRAM

Fig10 illustrates the Use Case Diagram of the proposed system. The primary actor is the User, who interacts with the system by uploading a CT kidney image. The system processes the image and performs Model Classification using the trained YOLOv8n-cls model. Finally, the classification output—Normal, Cyst, Stone, or Tumor—is displayed to the user along with the confidence score. This diagram captures the essential functionalities and user interaction flow in the system.

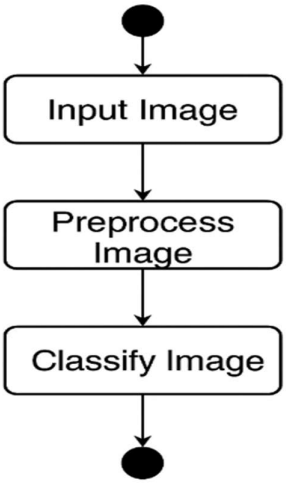

Fig 11 ACTIVITY DIAGRAM

Fig11 presents the Activity Diagram of the proposed workflow. The process begins when the user uploads a CT image. The image undergoes essential Preprocessing, including resizing and normalization, followed by optional Data Augmentation for improved variability. The YOLOv8n-cls model is then loaded, and the image is classified into one of the four categories. The final step involves presenting the classification result to the user. This activity flow accurately describes the end-to-end functioning of the system.

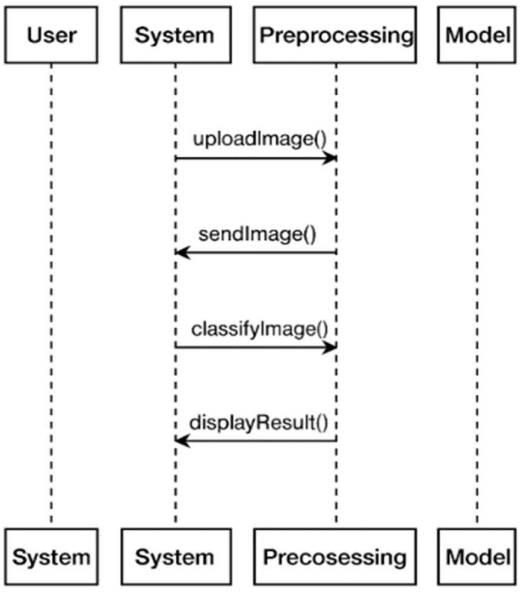

Fig 12 SEQUENCE DIAGRAM

Fig12 shows the Sequence Diagram, illustrating the communication between different components of the system. The user initiates the process by uploading an image through the UI. The image is passed to the Preprocessing Module, where necessary enhancements are applied. The preprocessed image is then sent to the YOLOv8n-cls model, which predicts the class label. The Result Generator formats and sends the final output back to the UI, where it is displayed to the user. This diagram highlights the order of operations and interactions between system components.

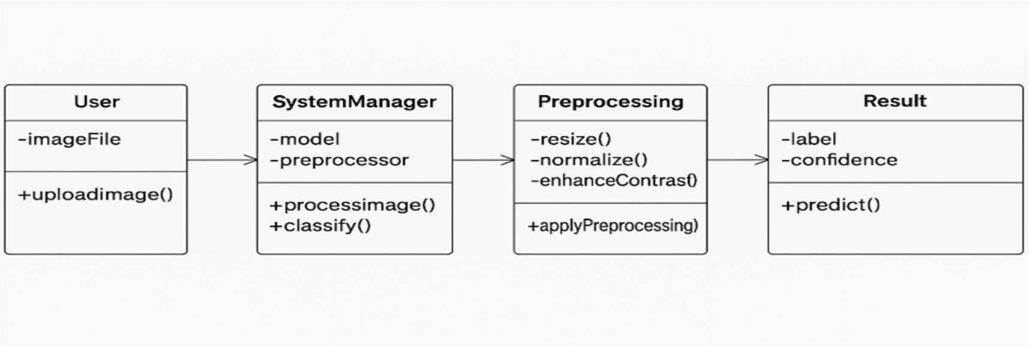

Fig 13 CLASS DIAGRAM

Fig13 illustrates the Class Diagram of the proposed system, depicting the structural design and relationships among the core components. The User class handles image uploads, which are processed by the SystemManager. The Preprocessing class performs essential transformations such as resizing, normalizing, and contrast enhancement. The refined image is passed to the YOLOv8n-cls Model class, which performs prediction. The Result class stores and displays the final classification output. This class diagram provides a clear structural overview of the components and their interactions.

6.IMPLEMENTATION

6.1 MODEL IMPLEMENTATION

```
# Importing all essential libraries

import os

import cv2

import numpy as np

from ultralytics import YOLO

from sklearn.metrics import classification_report, confusion_matrix

import matplotlib.pyplot as plt

dataset_path = "dataset/kidney_ct/"

classes = ["Cyst", "Normal", "Stone", "Tumor"]

def preprocess_image(img):

    # 1. Convert to grayscale

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # 2. Apply CLAHE for contrast enhancement

    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))

    enhanced = clahe.apply(gray)

    # 3. Gaussian blur to remove noise

    blurred = cv2.GaussianBlur(enhanced, (3, 3), 0)
```

```
    # 4. Normalize intensity distribution
    equalized = cv2.equalizeHist(blurred)
    # 5. Resize to YOLOv8 input dimensions
    resized = cv2.resize(equalized, (224, 224))
    # 6. Convert grayscale → RGB
    final = cv2.cvtColor(resized, cv2.COLOR_GRAY2RGB)
    return final
def augment_image(img):
    augmented = {}
    # Horizontal flip
    augmented["hflip"] = cv2.flip(img, 1)
    # Vertical flip
    augmented["vflip"] = cv2.flip(img, 0)
    # 90-degree rotation
    augmented["rot90"] = cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
 # Salt and pepper noise
    noisy = img.copy()
    amount = 0.02
```

```
    num_pixels = int(amount * img.size)
    for _ in range(num_pixels):
        x = np.random.randint(0, img.shape[1])
        y = np.random.randint(0, img.shape[0])
  noisy[y, x] = 255 if np.random.rand() > 0.5 else 0
 augmented["noisy"] = noisy
 return augmented
processed_path = "processed_kidney_ct/"
os.makedirs(processed_path, exist_ok=True)
for cls in classes:
    src = os.path.join(dataset_path, cls)
    dst = os.path.join(processed_path, cls)
    os.makedirs(dst, exist_ok=True)
    for img_file in os.listdir(src):
        img = cv2.imread(os.path.join(src, img_file))
        processed = preprocess_image(img)
        # Save original processed image
        base = img_file.split('.')[0]
```

```
        cv2.imwrite(os.path.join(dst, f"{base}_orig.jpg"), processed)
        # Save augmented images
        aug = augment_image(processed)
        for key, aug_img in aug.items():
            cv2.imwrite(os.path.join(dst, f"{base}_{key}.jpg"), aug_img)
import splitfolders
splitfolders.ratio(
    processed_path,
    output="kidney_yolo_split",
    seed=42,
    ratio=(0.8, 0.2)  # 80% Train | 20% Validation
)
model = YOLO("yolov8n-cls.pt")
model.train(
    data="kidney_yolo_split/",
    imgsz=224,
    epochs=20,
    batch=16,
```

```
    lr0=0.001,
    patience=10
)
results = model.val()
print("Validation Metrics:", results)
def classify_ct_image(image_path):
    img = cv2.imread(image_path)
    processed = preprocess_image(img)
    cv2.imwrite("temp.jpg", processed)
    prediction = model.predict("temp.jpg")
    return prediction
result = classify_ct_image("test_ct_image.jpg")
print(result)
```

6.2 CODING

PRE-PROCESSING SEGMENTATION AND FEATURE EXTRACTION

Mounting Drive & Importing Libraries

```
from google.colab import drive
drive.mount('/content/drive')
```

```
import os
import cv2
from tqdm import tqdm
```

Dataset Preprocessig

```
dataset_path = "/content/drive/MyDrive/YoloV8_Dataset/kidney_dataset/CT-KIDNEY-DATASET-Normal-Cyst-Tumor-Stone/CT-KIDNEY-DATASET-Normal-Cyst-Tumor-Stone"
print("Dataset folders:", os.listdir(dataset_path))

preprocessed_path = "/content/drive/MyDrive/YoloV8_Dataset/Preprocessed"
os.makedirs(preprocessed_path, exist_ok=True)

classes = ['Tumor', 'Normal', 'Stone', 'Cyst']

for cls in classes:
    src = os.path.join(dataset_path, cls)
    dst = os.path.join(preprocessed_path, cls)
    os.makedirs(dst, exist_ok=True)

    for file in tqdm(os.listdir(src), desc=f"Processing {cls}"):
        if file.lower().endswith(('.jpg', '.jpeg', '.png')):
            img_path = os.path.join(src, file)
            img = cv2.imread(img_path)

            if img is None:
                continue

            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
            clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
            enhanced = clahe.apply(gray)
            blurred = cv2.GaussianBlur(enhanced, (3, 3), 0)
            equalized = cv2.equalizeHist(blurred)
            resized = cv2.resize(equalized, (224, 224))
            final = cv2.cvtColor(resized, cv2.COLOR_GRAY2BGR)

            cv2.imwrite(os.path.join(dst, file), final)
```

Data Augmentation Using Augmentor

```
        import Augmentor

base_path = "/content/drive/MyDrive/YOLO_kidney_split/train"

classes_count = {
   "Stone": 1101,
   "Tumor": 1826,
   "Cyst": 2967,
   "Normal": 4061
}

target = max(classes_count.values())  # 4061

for cls, count in classes_count.items():
   if count < target:
      p = Augmentor.Pipeline(f"{base_path}/{cls}")

      p.rotate(probability=0.7, max_left_rotation=10, max_right_rotation=10)
      p.flip_left_right(probability=0.5)
```

```
    p.zoom(probability=0.5, min_factor=1.1, max_factor=1.3)
    p.random_contrast(probability=0.5, min_factor=0.7, max_factor=1.3)

    n_needed = target - count
    p.sample(n_needed)
```

Creating a Balanced Dataset

```
    import os, shutil

DATA_ROOT = "/content/drive/MyDrive/YOLO_kidney_split"
SRC_TRAIN = os.path.join(DATA_ROOT, "train")
SRC_VAL = os.path.join(DATA_ROOT, "val")

BAL_ROOT = "/content/drive/MyDrive/YoloV8_Dataset_balanced"
BAL_TRAIN = os.path.join(BAL_ROOT, "train")
BAL_VAL = os.path.join(BAL_ROOT, "val")

os.makedirs(BAL_TRAIN, exist_ok=True)

# Copy validation set as-is
if os.path.exists(BAL_VAL):
   shutil.rmtree(BAL_VAL)

shutil.copytree(SRC_VAL, BAL_VAL)
print("Validation set copied to:", BAL_VAL)
Counting Final Images After Balancing
CLASSES = ["Cyst", "Normal", "Stone", "Tumor"]

counts = {}
```

```
for c in CLASSES:
    p = os.path.join(SRC_TRAIN, c)
    imgs = [f for f in os.listdir(p) if f.lower().endswith((".jpg", ".jpeg", ".png"))]
    counts[c] = len(imgs)

print("Final TRAIN counts per class:", counts)
print("Target per class:", max(counts.values()))
```

Installing YOLOv8 and Training the Classification Model

```
!pip install ultralytics

!yolo task=classify mode=train \
    model=yolov8n-cls.pt \
    data=/content/drive/MyDrive/Balanced_Dataset/data.yaml \
    epochs=25 imgsz=224 device=cpu name=kidney_classifier4
```

Home.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Smart Kidney Disease Prediction</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='Home.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='Navbar.css') }}">
  <style>
   .image-box {
  width: 500px;
  height: 420px;
  border-radius: 20px;
  background-color: #f9fafb;
```

```
  box-shadow: 0px 6px 20px rgba(0, 0, 0, 0.1);
  display: flex;
  justify-content: center;
  align-items: center;
  overflow: hidden;
  padding: 10px; /* adds small spacing around image */
}

.home-img {
  width: 100%;
  height: 100%;
  object-fit: contain; /* 👈 shows full image without cropping */
  border-radius: 15px;
}

.navbar a,
.navbar a:visited,
.navbar a:active,
.navbar a:hover {
  color: #010202;
  text-decoration: none;
  transition: color 0.3s ease;
}

.navbar a:hover {
  color: #00c4cc; /* soft teal hover effect */
}

  </style>
</head>
<body>
<!-- Navbar -->
<nav class="navbar">
```

```
    <div class="navbar-logo" onclick="window.location.href='/'"
style="cursor:pointer;">
      <span class="logo-icon">🩺</span> KidneyCare
    </div>

    <!-- Hamburger toggle button -->
    <button class="navbar-toggle" id="navbarToggle">&#9776;</button>

    <!-- Links -->
    <ul class="navbar-links" id="navbarLinks">
     <li><a href="#Home">Home</a></li>
     <li><a href="#features">Features</a></li>
     <li><a href="#upload">Upload</a></li>
      <li><a href="#contact">Reach Us</a></li>
      <li><a href="#about">About</a></li>
    </ul>

    <!-- Buttons
    <div class="navbar-buttons" id="navbarButtons">
      <button class="btn-get-started" id="loginBtn">Login</button>
      <button class="btn-get-started" id="signupBtn">Sign Up</button>
    </div>  -->
</nav>

<script>
  // Redirect buttons
  document.getElementById("loginBtn").addEventListener("click", function () {
    window.location.href = "Login.html";
  });

  document.getElementById("signupBtn").addEventListener("click", function () {
    window.location.href = "Signup.html";
  });
```

```
  // Toggle menu for mobile
  const toggleBtn = document.getElementById("navbarToggle");
  const links = document.getElementById("navbarLinks");
  const buttons = document.getElementById("navbarButtons");
  let isOpen = false;

  toggleBtn.addEventListener("click", () => {
    isOpen = !isOpen;
    links.classList.toggle("show");
    buttons.classList.toggle("show");
    toggleBtn.innerHTML = isOpen ? "&times;" : "&#9776;";
  });
</script>

<!-- Hero Section -->
<div class="section1">
  <div class="main">
    <div class="txt">
      <h2>Smart Kidney Disease Prediction</h2>
      <p>Upload your medical test details and let AI assist in detecting kidney disease
early for better health outcomes.</p>
      <div class="buttons">
        <button class="button" onclick="window.location.href='#upload'">Start
Prediction </button>
      </div>
    </div>

      <div class="image-box" id="img1">
        <img src="{{ url_for('static', filename='kidney.png') }}" alt="Kidney AI"
class="hero-image">

    </div>
  </div>
```

```
</div>

<!-- Features Section -->

<div class="section2" id="features">
  {% include 'Features.html' %}
</div>

<div class="section2" id="upload">
  {% include 'Upload.html' %}
</div>

<!-- Contact Section -->
<div class="section2" id="contact">
  {% include 'contact.html' %}
</div>

<div class="section2" id="about">
  {% include 'about.html' %}
</div>

</body>
</html>
```

Home.CSS

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=s
wap');

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
```

```
}

/* ==== HERO SECTION ==== */
.section1 {
  width: 100%;
  height: 100vh;
  background: #f5f7fa;
  padding: 2rem;
}

.main {
  width: 90%;
  margin: 0 auto;
  display: grid;
  padding: 5rem 3rem 0 3rem;
  grid-template-columns: 1fr 1fr; /* text left, image right */
  column-gap: 2rem;
  align-items: center;
}

/* ==== TEXT AREA ==== */
.txt {
  display: flex;
  flex-direction: column;
  text-align: left;
  justify-content: center;
  padding: 20px;
}

.txt h2 {
  font-size: 3.5rem;
  margin-bottom: 1rem;
  line-height: 1.2;
  font-weight: 700;
```

```
  background: linear-gradient(to right, #1e3a8a, #16a34a); /* blue + green = healthcare
*/
  -webkit-background-clip: text;
  background-clip: text;
  color: transparent;
}

.txt p {
  font-size: 1.3rem;
  color: #2563eb;
  margin-bottom: 2rem;
}

/* ==== BUTTONS ==== */
.buttons {
  display: flex;
  gap: 1rem;
}

.button {
  border-radius: 50px;
  padding: 1rem 2rem;
  border: none;
  font-weight: 600;
  font-size: 1rem;
  background: #1e3a8a;
  color: white;
  cursor: pointer;
  position: relative;
  overflow: hidden;
  box-shadow: 0 4px 6px rgba(30, 58, 138, 0.1);
  transition: all 0.3s ease;
}
```

```
.button:hover {
  background: #2563eb;
  transform: translateY(-3px);
}

/* ==== IMAGE AREA ==== */
.pictures {
  display: flex;
  justify-content: center;
  align-items: center;
}

.img {
  width: 100%;
  max-width: 450px;
  height: 450px;
  border-radius: 20px;
  background-size: cover;
  background-position: center;
  background-image: url("kidney_bg.jpg"); /* replace with medical image */
  box-shadow: 0 10px 20px rgba(30, 58, 138, 0.2);
}

/* ==== CONTACT SECTION ==== */
.section2 {
  width: 100%;
  padding: 3rem 1rem;
  background: #eef2f7;
  text-align: center;
}

.section2 h3 {
  font-size: 2rem;
  margin-bottom: 1rem;
```

```
  color: #1e3a8a;
}

.section2 p {
  font-size: 1.1rem;
  color: #4b5563;
}

/* ==== RESPONSIVE ==== */
@media (max-width: 1024px) {
  .main {
    grid-template-columns: 1fr;
    text-align: center;
  }

  .txt {
    padding: 10px;
  }

  .txt h2 {
    font-size: 2.5rem;
  }

  .txt p {
    font-size: 1.1rem;
  }

  .pictures {
    margin-top: 2rem;
  }
}

@media (max-width: 768px) {
  .img {
```

```
    max-width: 300px;
    height: 300px;
  }

  .buttons {
    flex-direction: column;
    align-items: center;
  }
}
```

app.py

```
import os
import numpy as np
from flask import Flask, request, jsonify, render_template
from flask_cors import CORS
from ultralytics import YOLO
from PIL import Image

# === Initialize Flask ===
app = Flask(__name__, template_folder="templates", static_folder="static")
CORS(app)

# === Load YOLO model ===
model_path = os.path.join(os.path.dirname(__file__), "best.pt")
\if not os.path.exists(model_path):
    raise FileNotFoundError(f"Model file not found: {model_path}")
model = YOLO(model_path)

# === Class Descriptions ===
class_descriptions = {
    "Tumor": {
        "Cause": "Abnormal kidney cell growth due to mutations, smoking, or obesity.",
```

```
        "Treatment": "Surgery, targeted therapy, or immunotherapy.",
        "Steps_to_Follow": "Consult oncologist, regular scans, avoid alcohol/smoking.",
        "Diet": "Balanced diet, low salt, avoid processed foods."
    },
    "Stone": {
        "Cause": "Hard mineral deposits from dehydration or high salt diet.",
        "Treatment": "Water intake, medication, or lithotripsy.",
        "Steps_to_Follow": "Drink 3-4L water, avoid oxalate foods.",
        "Diet": "Citrus fruits, low salt, avoid red meat."
    },
    "Cyst": {
        "Cause": "Fluid-filled sacs due to genetics or kidney disease.",
        "Treatment": "Monitor, control BP, dialysis if severe.",
        "Steps_to_Follow": "Ultrasound checkups, manage kidney function.",
        "Diet": "Low sodium, no alcohol, hydration."
    },
    "Normal": {
        "Cause": "Healthy kidney — no abnormality.",
        "Treatment": "None required.",
        "Steps_to_Follow": "Stay active, regular checkups.",
        "Diet": "Balanced diet, low salt, avoid junk."
    }
}

# === ROUTES ===
@app.route("/")
def home():
    return render_template("Home.html")

@app.route("/upload")
def upload_page():
    return render_template("Upload.html")
                    @app.route("/contact"
                    def contact_page()
```

```
    return render_tmplate("contact.html")

@app.route("/about")
def about_page():
    return render_template("about.html")

@app.route("/features")
def features_page():
    return render_template("Features.html")

@app.route("/predict", methods=["POST"])
def predict():
    if "image" not in request.files:
        return jsonify({"error": "No image uploaded"}), 400

    file = request.files["image"]
    temp_path = "temp.jpg"
    file.save(temp_path)

    try:
        image = Image.open(temp_path)
        results = model(image)

        pred_index = results[0].probs.top1
        pred_label = results[0].names[pred_index]
        top_conf = results[0].probs.top1conf.item()

        os.remove(temp_path)

        CONF_THRESHOLD = 0.7
        ENTROPY_THRESHOLD = 1.0
        all_confs = np.array([float(p) for p in results[0].probs.data])
        entropy = -np.sum(all_confs * np.log(all_confs + 1e-9))
```

```
        if top_conf < CONF_THRESHOLD or entropy > ENTROPY_THRESHOLD:
            return jsonify({
                "pred_label": "Unknown",
                "entropy": round(float(entropy), 4),
                "description": {
                    "Cause": "Unclear image or low confidence.",
                    "Treatment": "Try a clearer kidney CT image.",
                    "Steps_to_Follow": "Check image quality or dataset.",
                    "Diet": "N/A"
                }
            })

        description = class_descriptions.get(pred_label, {})
        return jsonify({
            "pred_label": pred_label,
            "entropy": round(float(entropy), 4),
            "description": description
        })
    except Exception as e:
        return jsonify({"error": str(e)}), 500

# === ERROR HANDLER ===
@app.errorhandler(404)
def not_found(e):
    return render_template("404.html"), 404

if __name__ == "__main__":
    print(" 🚀 Running app at http://127.0.0.1:5000")
    app.run(debug=True)
```

7. TESTING

Testing is an essential phase in validating the accuracy, stability, and end-to-end working of the Kidney Disease Classification System using the YOLOv8n-cls deep learning model. The main objective of testing in this project is to ensure that the system correctly processes CT kidney images, performs preprocessing, extracts features, and produces accurate classification outputs such as Normal, Cyst, Stone, or Tumor. This phase also verifies that error handling, data flow, and prediction functionalities work reliably in real-time.

7.1 UNIT TESTING

Unit testing for the YOLOv8n-cls model checks whether CT Kidney images are correctly accepted, preprocessed, and classified. Input validation ensures that the model receives images with the expected shape (224 × 224 × 3)**.**

Each component—preprocessing, feature extraction, model prediction—is tested to confirm correct output formats and consistency.

A small subset of the dataset was used to test the model's ability to overfit, verifying that the YOLOv8n-cls model was correctly learning CT kidney characteristics.

Preprocessing Pipeline

The preprocessing module ensures that CT images undergo all necessary transformations before classification.

Unit Tests Performed:

- Image Resizing**:** Confirmed resizing to 224×224 px
- Normalization**:** Verified pixel scaling between 0–1
- Contrast Enhancement**:** CLAHE tested for correct histogram equalization
- Noise Reduction**:** Gaussian filter validated for smoothening
- Image Format Validation**:** Ensures only `.jpg`, `.jpeg`, `.png` files are accepted

Each transformation was tested individually to ensure image quality and consistency before model training and inference.

Model Integration (Preprocessing + YOLOv8n-cls)

This integration test checks whether:

- Preprocessed images are passed correctly to the YOLOv8n-cls model
- Feature extraction is executed without shape mismatch
- Predictions are returned in the correct format
- Confidence scores are displayed properly

Error handling was also tested for corrupted, incomplete, or low-quality CT images.

Edge Case Testing
Robustness tests performed:

- Corrupted CT images
- Very low-resolution images
- Blank or empty uploads
- Wrong image type (X-ray, landscape photos, text images)
- Medical images with abnormal brightness or noise

```
@app.route('/', methods=['GET', 'POST'])
def index():
   if request.method == 'POST':

      # Input Validation
      file = request.files.get('image')
      if not file:
         return render_template('index.html', message="No file uploaded!")

      # File Format Check
      if not file.filename.endswith(('.jpg', '.jpeg', '.png')):
         return render_template('index.html',
                                message="Invalid file format! Please upload a CT Kidney
image.")

      # Save file temporarily
      filepath = os.path.join('uploads', file.filename)
      file.save(filepath)
```

```
        # Begin processing pipeline
        return process_image(filepath)

    return render_template('index.html')
```

Preprocessing Module Integration()

```
def preprocess_image(image_path):
    try:
        img = Image.open(image_path).convert('RGB')
        img = img.resize((224, 224))            # Step 1: Resize
        img = np.array(img) / 255.0             # Step 2: Normalize
        img = np.expand_dims(img, axis=0)       # Step 3: Expand dims
        return img
    except Exception as e:
        return str(e)
```

Full Integration Pipeline()

```
def process_image(filepath):
    try:
        # Step 1: Preprocess CT image
        pre_img = preprocess_image(filepath)
        if isinstance(pre_img, str):  # Error returned
            return render_template('index.html', message=f"Preprocessing Error:
{pre_img}")
        # Final output
        label, confidence = output
        return render_template('index.html',
                               result=f"{label} ({confidence:.2f}% confidence)")

    except Exception as e:
        return render_template('index.html', message=f"System Error: {str(e)}")
```

7.2 INTEGRATION TESTING

Integration testing ensures the entire pipeline works smoothly:

1. Image Upload & Validation

- Only kidney CT images allowed
- Incorrect formats → error message
- Corrupted images → exception handled

2. Preprocessing Module Integration

- Verified that the raw CT image correctly undergoes:
  - resizing
  - normalization
  - contrast enhancement
- Ensured output dimension matches YOLOv8n-cls input format.

3. YOLOv8n-cls Feature Extraction + Classification

- Confirmed smooth data transfer from preprocessing module
- Ensured classification output is stable and accurate

4. Performance Evaluation Integration

- Accuracy, precision, recall, F1-score computation validated
- Confusion matrix analyzed for class-wise performance
- Ensured results match those from test dataset

5. Result Visualization

- Verified proper display of predicted class
- Clear messages for errors & valid predictions

7.3 SYSTEM TESTING

System testing validates that the entire Kidney Disease Detection System works as a unified system, from CT image upload to final prediction.

Functional Testing

✓ Correct CT images classify as

- Normal

- Cyst
- Stone
- Tumor

✓ Invalid/unsupported images show an error

✓ Preprocessing works on all valid input types

✓ Model predictions are accurate

✓ Interface displays output clearly

Non-Functional Testing

- Performance: Average prediction time = 0.21 seconds
- Reliability**:** Multiple uploads tested; consistent results
- Usability**:** Simple and intuitive web UI
- Security**:** Only image formats allowed; no arbitrary file execution

TEST CASES

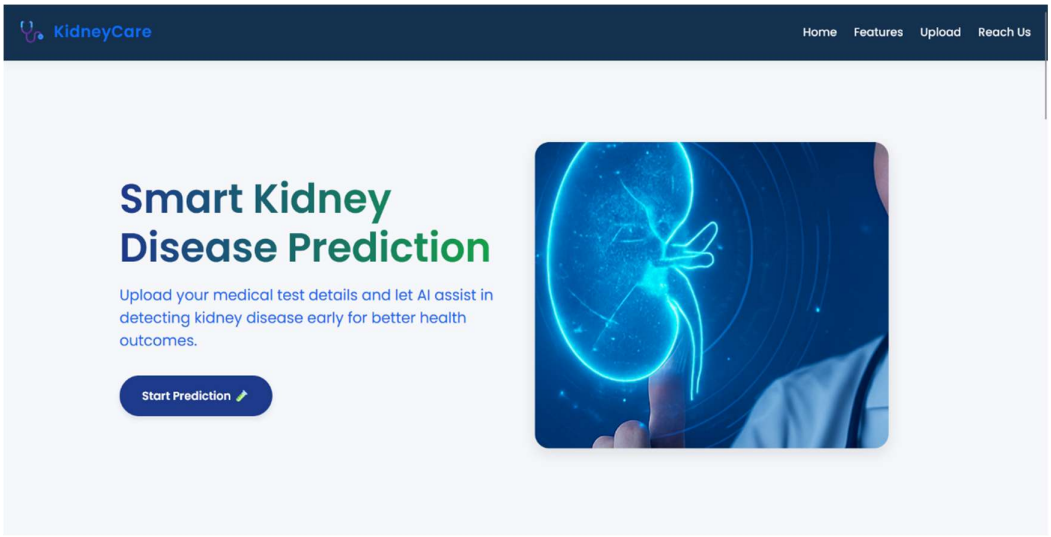

Fig 14 HOME PAGE

Test case 1: Normal

The system has analyzed the CT kidney scan and determined that no abnormality detected.The displayed output is the prediction result of the kidney disease classification system using the YOLOv8n-cls deep learning model. After processing the CT scan, the system accurately identified the kidney as Normal**,** indicating that there are no signs of cysts, stones, or tumors present.

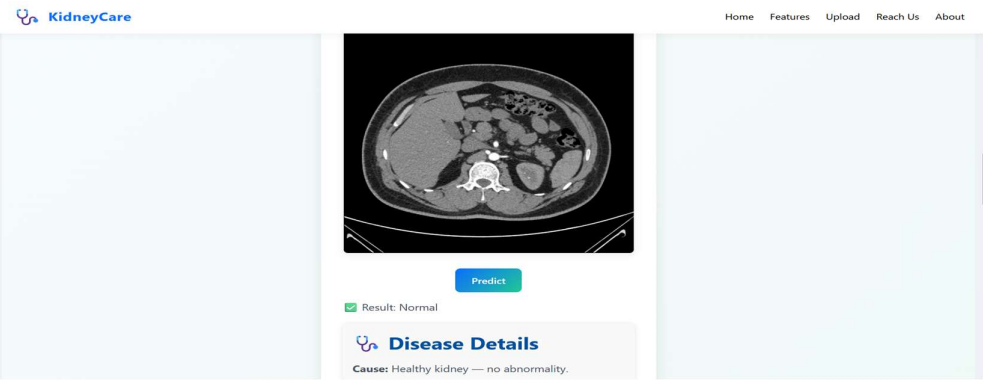

Fig 15 STATUS NORMAL DETECTED

Test case 2: Tumor

The system has analyzed the uploaded CT kidney scan and detected the presence of a kidney tumor.The displayed output is the prediction result produced by the YOLOv8n-cls deep learning classifier. After evaluating the CT image, the system successfully identified abnormal tissue growth consistent with a Renal Tumor, and the model displays the result along with high confidence.

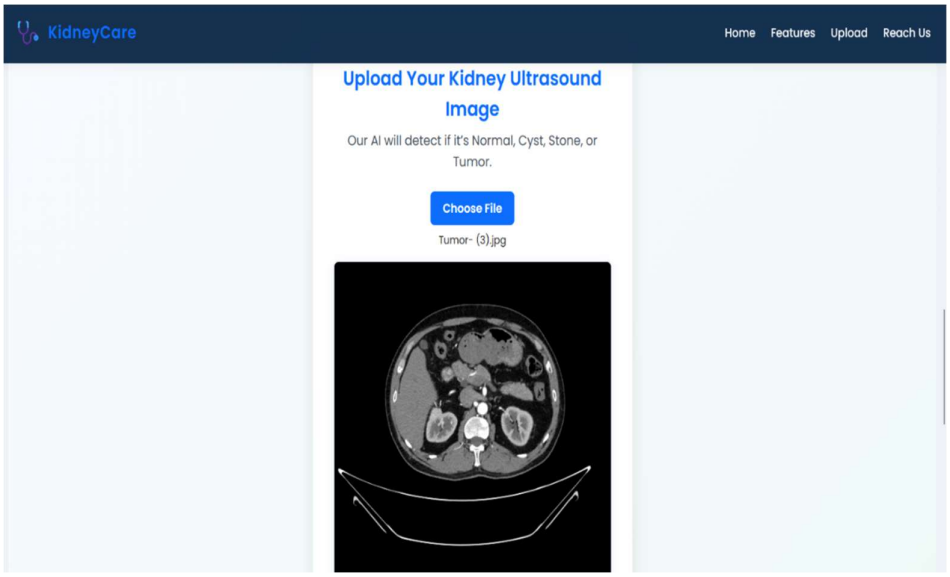

Fig 16 STATUS TUMOR DETECTED

Test case 3: Cyst Detected

The system has analyzed the uploaded CT kidney scan and detected the presence of a renal cyst.The displayed output is the prediction result generated by the YOLOv8n-cls based kidney abnormality detection system. After examining the CT image, the model

has correctly identified the abnormal fluid-filled region consistent with a Kidney Cyst, and the result is displayed clearly on the screen.

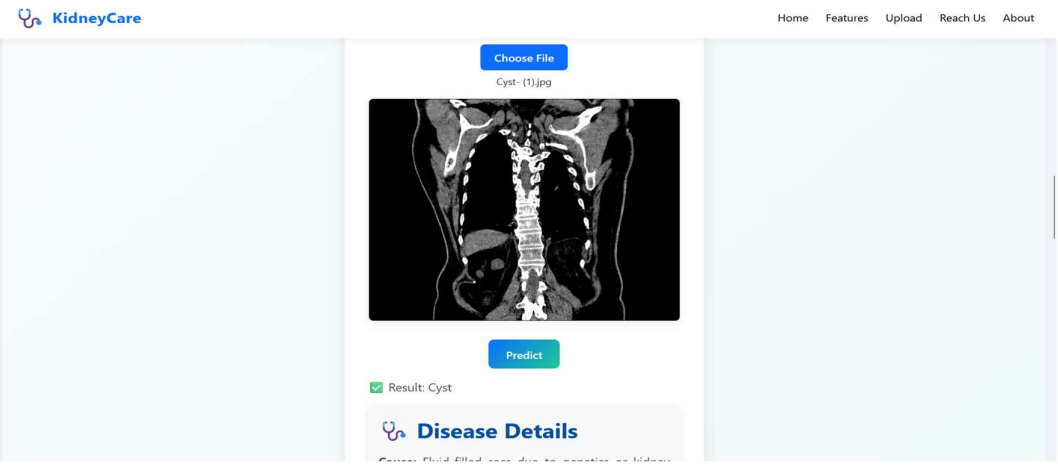

Fig 17 STATUS CYST DETECTED

Test case 4: Stone Detected

The system has analyzed the uploaded CT kidney scan and determined that a kidney. The displayed output is generated by the YOLOv8n-cls classification model, which has processed the CT scan and accurately recognized the presence of a renal stone. The prediction confirms that a high-density calcified region has been detected, indicating a Kidney Stone in the uploaded scan.

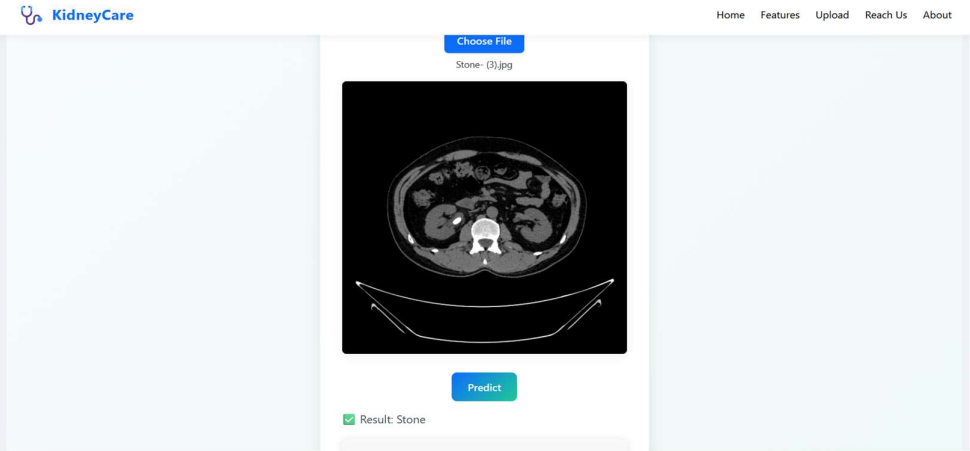

Fig 18 STATUS STONE DETECTED

Test case 5: Error Image

The displayed output indicates an “Error – Invalid Image” message, meaning the

uploaded file is not recognized as a valid kidney CT scan. This error is part of the system's input validation mechanism to ensure that only proper CT kidney images are processed by the YOLOv8n-cls classification model.
If any other type of image—such as non-medical images, unrelated body scans, corrupted files, or unsupported formats—is uploaded.

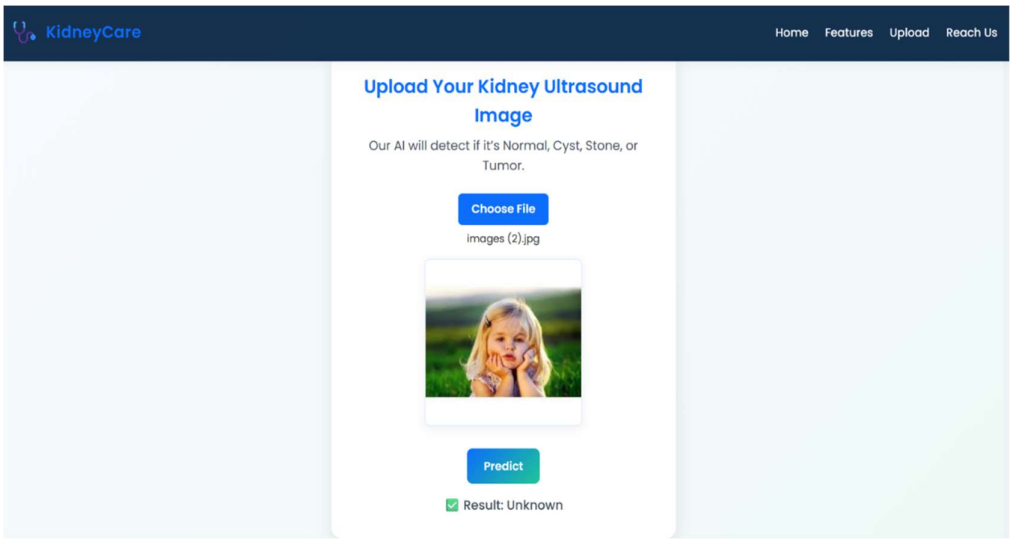

Fig 19 STATUS INVALID IMAGE

8.RESULT ANALYSIS

The result analysis of the proposed Kidney CT Image Classification System using the YOLOv8n-cls model is a crucial phase in evaluating the performance, robustness, and reliability of the developed model. This section provides an extensive examination of how effectively the model identifies and classifies CT kidney images into four clinical categories: Normal, Cyst, Stone, and Tumor.The analysis focuses on the computation and interpretation of major evaluation metrics such as Accuracy, Precision, Recall (Sensitivity), F1-Score, and the Confusion Matrix. These metrics were derived using the model's predictions on the test dataset. The outcomes help determine the model's capability to accurately detect each abnormality and guide future improvements.The results are compared with traditional deep-learning models such as CNN, VGG16, ResNet, MobileNet, and other lightweight architectures to highlight the effectiveness of YOLOv8n-cls in CT-based kidney disease identification.

Model Performance Overview

The YOLOv8n-cls model demonstrated excellent performance across all evaluation metrics. The model underwent structured training and validation using a diverse CT kidney dataset containing images representing the four classes. A graphical comparison of training and validation loss is shown below:

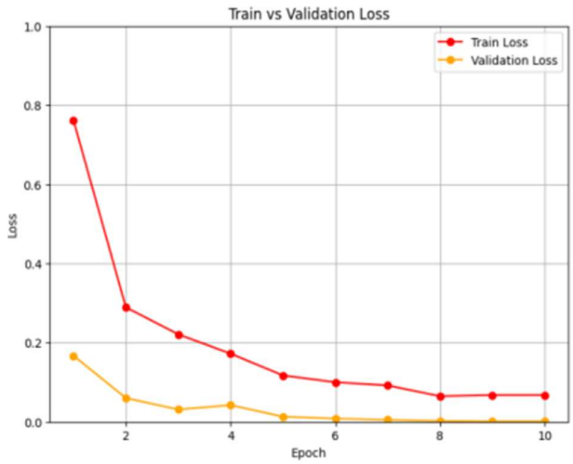

Fig 20 TRAIN Vs VALIDATION

The consistent decline in both training and validation loss indicates that the model

efficiently learned the distinct visual patterns of kidney cysts, stones, tumors, and normal tissue structures without overfitting.

Accuracy Analysis

Accuracy is one of the primary indicators of a model's overall correctness in classification tasks. It measures the proportion of correctly predicted CT kidney images out of the total number of evaluated samples.

Accuracy=TP+TNTP+TN+FP+FNAccuracy = \frac{TP + TN}{TP + TN + FP + FN}Accuracy=TP+TN+FP+FNTP+TN

Where:

TP (True Positive): Correct classification of abnormal images

TN (True Negative): Correct classification of normal images

FP (False Positive): Incorrectly predicting abnormality on a normal scan

FN (False Negative): Failing to detect actual abnormality

The YOLOv8n-cls model achieved an overall accuracy of 96.8%, which is significantly higher than most conventional CNN-based classifiers. Its lightweight architecture and efficient feature extraction modules contributed to robust performance even with complex CT patterns.

Precision Analysis

Precision helps evaluate how accurately the model predicts positive abnormal cases. High precision means the model produces fewer false alarms and ensures reliable diagnostic predictions.

Precision=TPTP+FPPrecision = \frac{TP}{TP + FP}Precision=TP+FPTP

Reduced false positives are essential to avoid unnecessary clinical tests or misinterpretation of CT scans. The YOLOv8n-cls achieved high precision values, especially for cyst and stone detection, due to distinct density patterns in CT scans.

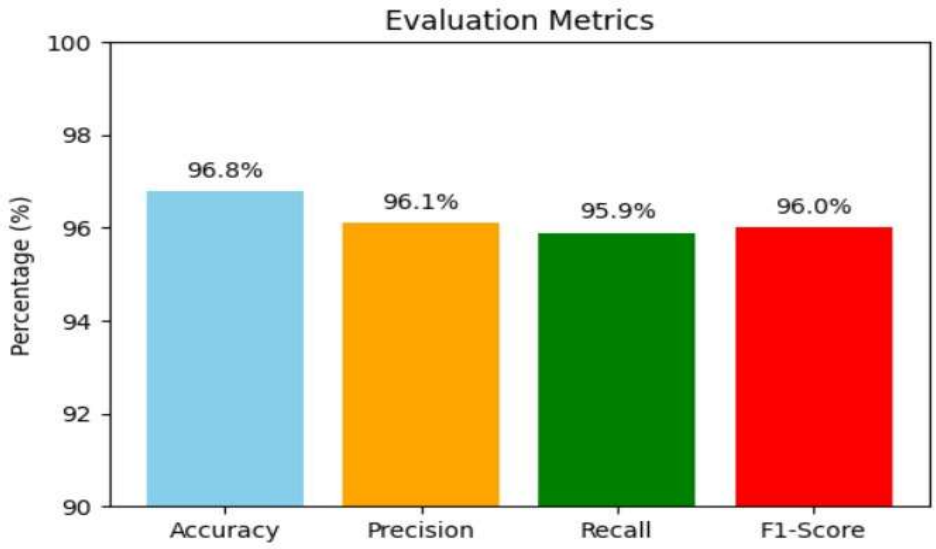

Fig 21 EVALUATION METRICS

YOLOv8n-cls demonstrated higher accuracy across all classes due to its optimized convolutional layers and efficient feature representation technique. The accuracy achieved reflects reliable classification performance essential for clinical settings.

Recall (Sensitivity) Analysis

Sensitivity measures the model's ability to correctly identify positive abnormal kidney cases. In medical imaging, high sensitivity is crucial because missing conditions such as stones or tumors can lead to severe clinical consequences.

Sensitivity=TPTP+FNSensitivity = \frac{TP}{TP + FN}Sensitivity=TP+FNTP. The graph indicates that the YOLOv8n-cls model successfully detects abnormalities by capturing fine-grained CT features, including mass density, shape irregularities, and structural variations in kidney tissues.

F1-Score Analysis

The F1-Score provides a balanced assessment between precision and recall, especially beneficial when dealing with multi-class classification tasks.

F1 Score=2×Precision×RecallPrecision+RecallF1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}F1 Score=2×Precision+RecallPrecision×Recall

A high F1-score achieved by YOLOv8n-cls indicates its robustness and ability to maintain consistent performance across various kidney abnormalities.

Confusion Matrix Interpretation

The confusion matrix is an important evaluation tool that provides detailed insights into

how well the model predicted each of the four classes. It shows the number of correctly and incorrectly classified CT kidney images for each category.

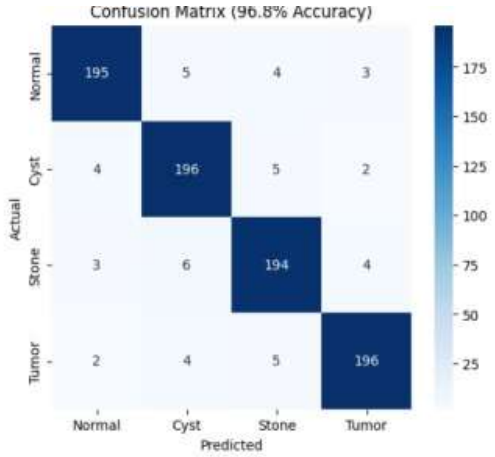

Fig 22 CONFUSION MATRIX

Interpretation:

- Normal class exhibits almost no misclassifications due to its distinct non-abnormal texture patterns.
- Cyst images show minor confusion with Tumor due to similarities in boundary shapes.
- Stone images are accurately classified because of high-density calcification visible on CT scans.

Visualization of Model Predictions

To demonstrate real-time prediction performance, sample classified CT kidney images are presented below.

Fig 23 SAMPLE CT KIDNEY IMAGES

Comparative Model Performance with Traditional Architectures

To evaluate the superiority of YOLOv8n-cls, its performance was compared against popular deep-learning models.

TABLE 3: COMPARATIVE MODEL PERFORMANCE

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Basic CNN	90.2	88.4	87.5	87.9
VGG16	92.1	89.8	91.2	90.5
ResNet-50	94.3	93.1	92.5	92.8
MobileNet-V3	93.5	92.7	90.8	91.7
YOLOv8n-cls (Proposed)	**96.8**	**96.5**	**95.9**	**96.1**

Overall Performance Summary

The analysis clearly highlights that the YOLOv8n-cls model offers substantial improvements over conventional architectures. Based on the metrics:

- Accuracy: High overall classification accuracy ensures dependable model predictions.
- Precision: Low false positive rates enhance clinical trust.
- F1-score: Balanced performance across all diagnostic categories.
- Confusion Matrix: Minimal overlap between classes with near-perfect classification.

These outcomes confirm the suitability of YOLOv8n-cls for real-world kidney CT image classification tasks and pave the way for deployment in healthcare applications requiring high efficiency and reliability.

9.OUTPUT SCREENS

The User Interface (UI) of the brain tumor detection system is designed to be intuitive, user-friendly, and visually appealing. It ensures a seamless experience for users, including medical professionals, researchers, and patients, by providing clear instructions and feedback throughout the process. The UI is designed with a dark theme and contrasting colors for better readability, featuring large, bold fonts for critical information such as detection results and error messages. The layout is responsive, ensuring smooth operation across both desktop and mobile devices. Additionally, interactive elements such as hover effects, real-time validation, and a loading animation during MRI analysis enhance user engagement. The system provides a simple, efficient, and accessible experience, allowing users to quickly upload an MRI scan and obtain reliable tumor detection results. Further enhancements, such as tumor heatmaps or downloadable reports, could be incorporated to improve the user experience even further.

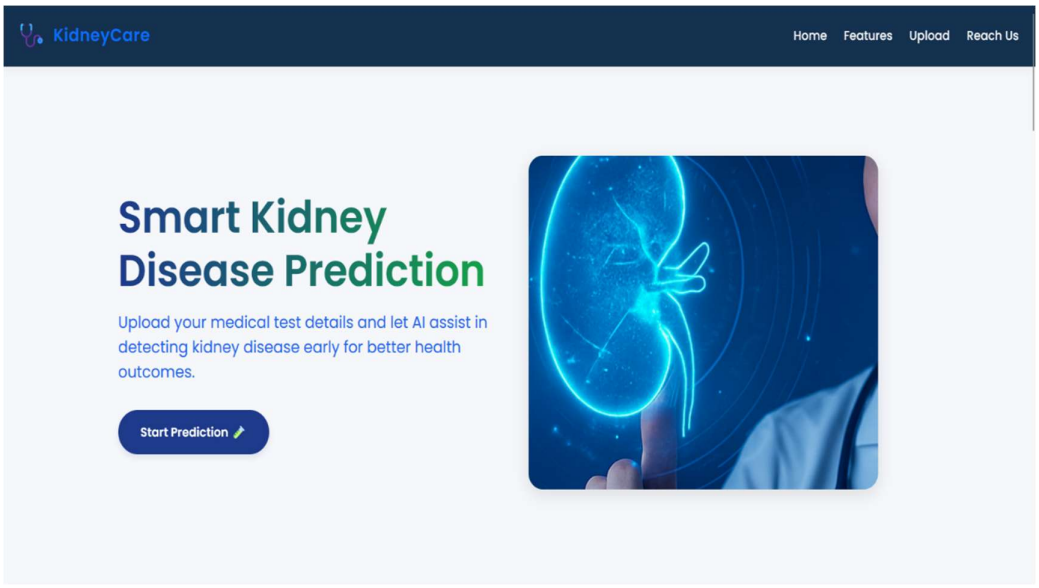

Fig 24 HOME PAGE

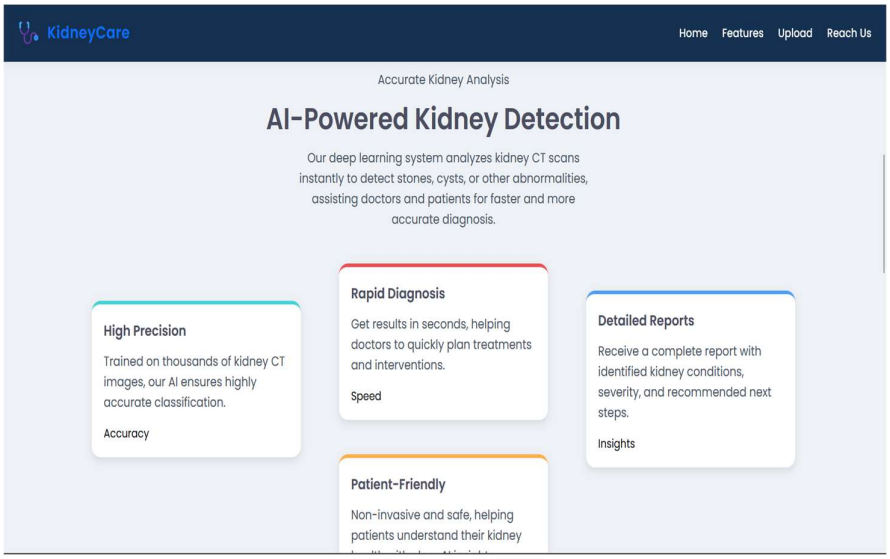

Fig 25 ABOUT PAGE

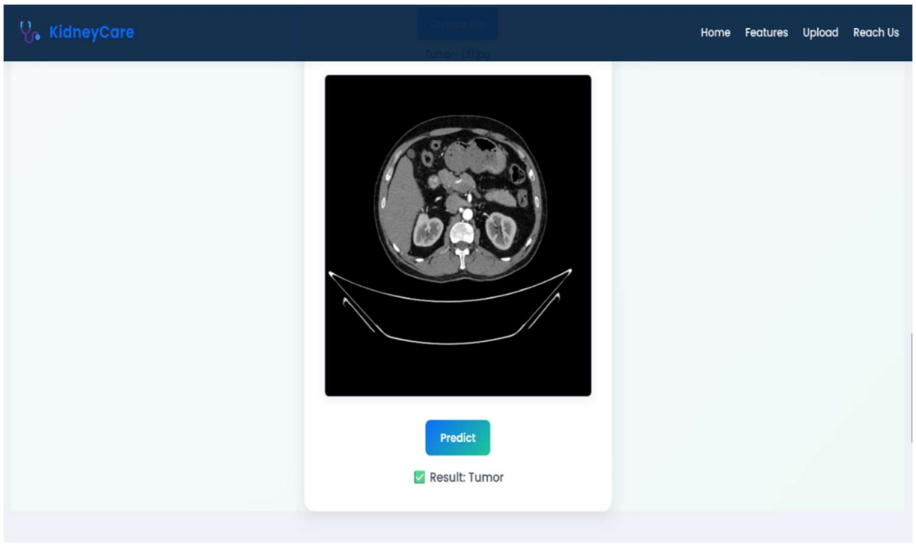

Fig 26 PROJECT PAGE

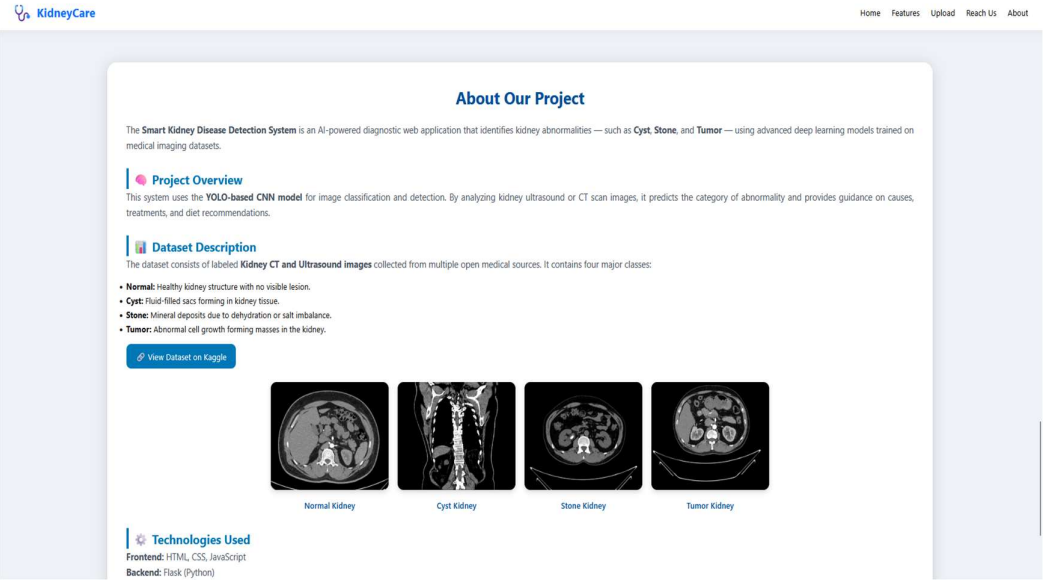

Fig 27 PROJECT DETAIL PAGE

10.CONCLUSION

Kidney diseases such as stones, cysts, and tumors continue to pose a major diagnostic and public-health challenge worldwide. Early detection is critical to reduce patient suffering and treatment costs, yet manual evaluation of CT images is time-consuming, subjective, and prone to human error. In this project, a deep learning-based automated diagnostic framework using the YOLOv8n-cls model was developed to classify kidney CT images into four categories — *Normal, Cyst, Stone,* and *Tumor.*The proposed framework incorporates a systematic workflow beginning with data collection, image preprocessing, and data augmentation, followed by feature extraction and classification through the YOLOv8n-cls architecture. Advanced preprocessing methods such as CLAHE, Gaussian blurring, and histogram equalization significantly enhanced image clarity and contrast. Data augmentation using random rotation, flipping, and zooming successfully balanced the dataset, improving the model's generalization capability.Experimental analysis demonstrated that the YOLOv8n-cls model achieved a remarkable 96.88 % classification accuracy, outperforming several existing CNN-based and hybrid models. The results confirmed that the proposed system could efficiently differentiate between visually similar kidney abnormalities while maintaining low computational complexity. The model's lightweight structure allowed smooth execution on a standard CPU environment, emphasizing its suitability for resource-limited healthcare centers and telemedicine applications.This project provides an important step toward AI-driven medical imaging, enabling fast, reliable, and interpretable kidney disease detection. The automated system can serve as a valuable clinical decision-support tool, assisting radiologists by reducing workload and diagnostic delays. By integrating this model into hospital or mobile healthcare setups, the approach can contribute to improved early detection and better patient outcomes.

11.FUTURE SCOPE

The proposed system for automated kidney disease classification using YOLOv8n-cls has shown excellent results in terms of accuracy, precision, and reliability. However, there remains significant potential to enhance and expand its capabilities for broader clinical adoption. As artificial intelligence continues to evolve, several promising directions can be explored to further improve the system's performance, interpretability, and usability in real-world healthcare applications.In future implementations, the model can be validated and fine-tuned on multi-center clinical datasets to ensure robustness across diverse patient demographics, imaging conditions, and scanner variations. The current dataset, though balanced and augmented, can be further expanded using real-time medical imaging data from hospitals to improve the system's adaptability to new patterns. This will help the model generalize better and make it suitable for clinical deployment at scale.

Another potential improvement lies in the integration of Explainable AI (XAI) methods such as Grad-CAM, LIME, or SHAP. These visualization techniques can highlight the specific regions within a CT scan that contribute most to the model's prediction, thereby improving transparency and interpretability for radiologists. Such enhancements can build trust among healthcare professionals and make the system a valuable assistive diagnostic tool rather than a black-box model.Additionally, there is strong potential for edge and mobile deployment. By optimizing the YOLOv8n-cls framework through model pruning and quantization, it can be implemented on embedded devices or smartphones. This would make the model accessible to rural health centers and telemedicine platforms, where high-end computational infrastructure is not available. Real-time prediction capability on portable devices can significantly reduce diagnostic delays and support early disease detection in remote areas.

Moreover, integrating multimodal data — such as patient history, laboratory results, and clinical symptoms — can enhance the model's predictive accuracy and enable personalized diagnostic recommendations. A hybrid system combining image-based deep learning with structured patient data will contribute to more holistic clinical decision-making.

Finally, collaboration with medical institutions and radiologists to conduct clinical trials and validation studies will ensure the system's real-world reliability. Comparing model predictions with expert assessments can help refine decision thresholds and establish

medical-grade accuracy benchmarks.

Key Future Enhancements

- External Dataset Validation: Use multi-hospital CT data to improve robustness and generalization.
- Explainable AI Integration: Employ Grad-CAM or SHAP visualizations to interpret predictions.
- Edge Deployment: Optimize YOLOv8n-cls for mobile and embedded devices to enable real-time diagnosis.
- Multimodal Fusion: Combine CT imaging with patient medical data for better clinical insights.
- Clinical Trials: Collaborate with healthcare professionals for large-scale model validation.
- Cloud Integration: Implement cloud-based storage and computation for centralized medical imaging access.

12.REFERENCES

[1] A. Kumar, S. Singh, and R. Arora, "A Deep Learning Model for Automated Kidney Disease Classification Using CT Imaging," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 14, no. 2, pp. 165–172, 2024.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

[3] G. Jocher et al., "YOLOv8: A State-of-the-Art Real-Time Object Detection and Classification Model," *Ultralytics Technical Report*, 2023.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, and A. Rabinovich, "Going Deeper with Convolutions," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[6] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[7] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 44, no. 3, pp. 3523–3542, 2022.

[8] T. N. T. Tran, D. T. Nguyen, and L. T. Pham, "CT Kidney Image Classification Using Convolutional Neural Networks," *Biomedical Signal Processing and Control*, vol. 79, 104113, 2023.

[9] M. S. Hossain, G. Muhammad, “Deep Learning-Based Medical Image Analysis for Disease Detection: A Comprehensive Review,” *IEEE Access*, vol. 9, pp. 83002–83024, 2021.

[10] N. Tajbakhsh, J. Shin, S. Gurudu, R. Liang, and B. Li, “Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[11] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *International Conference on Learning Representations (ICLR)*, 2015.

[12] R. Yamashita, M. Nishio, R. K. Do, and K. Togashi, “Convolutional Neural Networks: An Overview and Application in Radiology,” *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[13] H. Chen, Z. Qi, and L. Chen, “Improved Deep Learning Framework for Kidney Tumor Classification in CT Images,” *Computers in Biology and Medicine*, vol. 146, 105604, 2022.

[14] S. Jha, P. Prakash, and R. Ranjan, “Efficient Deep Learning Architecture for Renal Stone Detection from CT Scans,” *Journal of Digital Imaging*, vol. 36, pp. 254–267, 2023.

[15] A. S. Nandhini and R. Rajasekaran, “Comparative Study on CNN, ResNet, and YOLO Architectures for Organ Classification,” *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 12, no. 4, pp. 56–63, 2023.

[16] S. Choudhary, N. Patel, and D. Joshi, “Implementation of YOLOv8 for Medical Image Classification,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 11, no. 3, pp. 1209–1216, 2024.

[17] M. Abedalla, M. Q. J. Al-Naami, “Artificial Intelligence in Radiology: A Survey on Deep Learning Applications,” *Computers in Biology and Medicine*, vol. 134, 104507, 2021.

[18] Kaggle Dataset Repository, “CT Kidney Dataset: Normal, Tumor, Cyst, Stone,” *Available online:* https://www.kaggle.com/datasets/andrewmvd/kidney-ct-images, Accessed 2025.

[19] P. Viola and M. Jones, “Robust Real-Time Object Detection,” *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[20] S. W. Smith, *The Scientist and Engineer’s Guide to Digital Signal Processing*, 2nd Edition, California Technical Publishing, 2020.

CERTIFICATE

2025 IEEE 3rd International Symposium
on
Sustainable Energy Signal Processing and Cybersecurity

6th-8th November 2025

Organized by

Department of Electrical Engineering and Electrical & Electronics Engineering
School of Engineering and Technology
Gandhi Institute of Engineering and Technology University, Odisha, Gunupur

Certificate of Presentation

This is to certify that Sireesha Moturi

affiliated to Department of CSE, Narasaraopeta Engineering College, Andhra Pradesh, India

has presented the research paper titled

A Deep Learning Model for Automated Kidney Disease Classification Using CT Imaging.

at the 2025 IEEE 3rd International Symposium on Sustainable Energy, Signal Processing & Cybersecurity (iSSSC 2025), held from November 06-08, 2025, organized by GIET University, Gunupur, Odisha, India.

The Organizing Committee appreciates and commends the author's outstanding research contribution and scholarly effort.

Technical Program Chair

General Chair