DETECTION AND CLASSIFICATION OF KIDNEY STONES BASED ON DEEP LEARNING METHODS

2024
MASTER THESIS
BIOMEDICAL ENGINEERING

Aziz AYDIN

Thesis Advisor
Assist. Prof. Dr Eftâl ŞEHİRLİ

DETECTION AND CLASSIFICATION OF KIDNEY STONES BASED ON DEEP LEARNING METHODS

Aziz AYDIN

Thesis Advisor
Assist. Prof. Dr. Eftâl ŞEHİRLİ

T.C.
Karabuk University
Institute of Graduate Programs
Department of Biomedical Engineering
Prepared as
Master Thesis

KARABUK
January 2024

I certify that in my opinion the thesis submitted by Aziz AYDIN titled “DETECTION AND CLASSIFICATION OF KIDNEY STONES BASED ON DEEP LEARNING METHODS” is fully adequate in scope and in quality as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Eftâl ŞEHİRLİ ....
Thesis Advisor, Department of Biomedical Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Biomedical Engineering as a Master of Science thesis. January 16, 2024

Examining Committee Members (Institutions)		Signature
Chairman	: Assoc. Prof. Dr. Ahmet Reşit KAVSAOĞLU (KBU)	....
Member	: Assist. Prof. Dr. Kadir İLERİ (BANU)	ONLINE
Member	: Assist. Prof. Dr. Eftâl ŞEHİRLİ (KBU)	....

The degree of Master of Science by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabuk University.

Assoc. Prof. Dr. Zeynep ÖZCAN ....
Director of the Institute of Graduate Programs

“I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well.”

Aziz AYDIN

ABSTRACT

M.Sc. Thesis

DETECTION AND CLASSIFICATION OF KIDNEY STONES BASED ON DEEP LEARNING METHODS

Aziz AYDIN

Karabuk University
Institute of Graduate Programs
Department of Biomedical Engineering

Thesis Advisor:
Assist. Prof. Dr. Eftâl ŞEHİRLİ
January 2024, 154 pages

Kidney stones are a prevalent global health issue, leading numerous individuals to seek emergency care due to intense pain. Different imaging methods are employed in the diagnosis of kidney stone disease, requiring specialized expertise for the comprehensive interpretation and diagnosis of these images. Significant advancements in the medical field have been facilitated thanks to the application of machine learning and deep learning methods. This thesis aims to employ deep learning and object detection techniques to detect and classify kidney stones on CT images. The dataset employed in this thesis comprises a total of 1799 coronal CT scans. Among these, 1009 scans originate from individuals without kidney stones, while the remaining are collected from patients who have been diagnosed with kidney stones. This thesis involves implementing three different models as Faster R-CNN, YOLO, and a customized convolutional neural network (CNN). While Faster R-CNN performance was underwhelming, YOLO v5 achieved promising results, surpassing YOLO v7 with a mAP (0.5) of 84.6% and a mAP (0.5:0.95) of 39.0% for kidney stone detection. The customized CNN exhibited remarkable accuracy reaching 99.13%. Indicating its efficacy in classifying kidney stones, the model achieved an accuracy closely comparable to the leading studies in the literature firmly establishing itself as a noteworthy achievement.

**Key Words : ** Kidney stones, Deep learning, Object detection

**Science Code : ** 925118

ÖZET

Yüksek Lisans Tezi

DERİN ÖĞRENME YÖNTEMLERİYLE BÖBREK TAŞLARININ TESPİT EDİLMESİ VE SINIFLANDIRILMASI

Aziz AYDIN

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Biyomedikal Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Eftâl ŞEHİRLİ
Ocak 2024, 154 sayfa

Böbrek taşları, küresel bir sağlık sorunu olarak önem taşımakta ve şiddetli ağrı nedeniyle birçok kişinin acil yardım talep etmesine sebep olmaktadır. Böbrek taşı hastalığının teşhisi için farklı görüntüleme teknikleri kullanılmakta, bu görüntülerin detaylı bir şekilde değerlendirilmesi ve doğru teşhis konulabilmesi için uzmanlık gerekmektedir. Tıp alanında, makine öğrenmesi ve derin öğrenme yöntemlerinin entegrasyonu ile önemli ilerlemeler kaydedilmiştir. Bu tez, koronal BT görüntülerde böbrek taşlarını tespit etmek ve sınıflandırmak amacıyla derin öğrenme ve nesne tespiti tekniklerini kullanmayı amaçlamaktadır. Kullanılan veri seti, toplamda 1799 koronal BT görüntüleri içermektedir. Bu görüntülerden 1009'u böbrek taşı bulunmayan bireylerden alınmış olup, geri kalanı ise böbrek taşı teşhisi konmuş hastalardan elde edilmiştir. Bu tezde Faster R-CNN, YOLO ve özelleştirilmiş bir evrişimli sinir ağı (CNN) olmak üzere üç farklı model uygulanmıştır. Faster R-CNN'nin performansı beklenen seviyenin altında kalmıştır. Öte yandan, YOLO v5 umut verici sonuçlar elde etmiş ve özellikle böbrek taşı tespiti konusunda mAP (0,5)

%84,6 ve mAP (0,5:0,95) %39,0 başarı oranlarına ulaşarak YOLO v7'yi geçmiştir. Ayrıca, özelleştirilmiş bir CNN modeli de %99,13 doğrulukla dikkat çekmiştir. Bu model, böbrek taşlarını sınıflandırmadaki etkinliğini literatürdeki önde gelen çalışmalarla kıyaslanabilir bir doğruluk seviyesi elde etmiştir.

Anahtar Kelimeler: Böbrek taşları, Derin öğrenme, Nesne tespiti

Bilim Kodu: 925118

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my esteemed advisor, Asist. Prof. Dr. Eftâl ŞEHİRLİ, for his invaluable guidance and unwavering support throughout my master's journey. his vast wisdom and wealth of experience have inspired me throughout my thesis. I'd like to thank God, my family, and my friends. It would have been impossible to finish my studies without their unlimited support over the past few years.

CONTENTS

Page

Page

LIST OF FIGURES

Page

Page

Page

Page

LIST OF TABLES

SYMBOLS AND ABBREVIATIONS

ABBREVIATION

AI	: Artificial Intelligence
AUC	: Area Under Curve
AP	: Average Precision
APIs	: Application Programming Interfaces
ANN	: Artificial Neural Network
AS	: Average Sensitivity
CAD	: Computer-Aided Diagnosis
CCT	: Compact Convolutional Transformer
CKD	: Chronic Kidney Disease
CNN	: Convolutional Neural Network
CPU	: Central Processing Units
CT	: Computed Tomography
CV	: Cross-Validation
D1	: Original Dataset
D2	: Augmented Dataset
D3	: ROI Dataset
D4	: Augmented ROI Dataset
DKN	: Deep Kronecker Network
DBN	: Deep Belief Network
DQN	: Deep Q Network
DT	: Decision Tree
EANet	: External Attention Transformer
EPO	: Erythropoietin
E-ELAN	: Extended Efficient Layer Aggregation Network
FO-CHIO	: Fractional Order Coronavirus Herd Immunity Optimizer
GANs	: Generative Adversarial Networks
GFR	: Glomerular Filtration Rate
GPU	: Graphical Processing Units

INCA	: Iterative Neighborhood Component Analysis
IoU	: Intersection over Union
Knn	: k-Nearest Neighbor
KUB	: Kidney Ureter Bladder
LDA	: Linear Discriminant Analysis
LR	: Linear Regression
mAP	: mean Average Precision
NB	: Naïve Bayes
NMS	: Non-Maximum Suppression
PIQUE	: Perception-Based Image Quality Evaluator
RAM	: Read Access Memory
RF	: Random Forest
R-CNN	: Region-based Convolutional Neural Network
ResNet	: Residual Neural Network
ROI	: Region of Interest
RPN	: Region Proposal Network
SARSA	: State Action Reward State Action
SGD	: Stochastic Gradient Descent
SPP	: Spatial Pyramid Pooling
SVM	: Support Vector Machine
TP	: True Positive
TN	: True Negative
FP	: False Positive
FN	: False Negative
TPU	: Tensor Processing Units
ViT	: Vision Transform
VM	: Virtual Machine
VGG	: Visual Geometry Group
xAI	: Explainable Artificial Intelligence
YOLO	: You Only Look Once

CHAPTER 1

INTRODUCTION

Kidney stone disease stands as a prevalent health concern, affecting a substantial portion of the population with a reported prevalence ranging from 1% to 20% [1]. The formation of stones within the urinary system, encompassing the kidneys, ureters, and bladder, arises from the condensation of minerals and acid salts, eventually crystallizing over time. Statistics reveal that approximately 11% of men and 6% of women in the United States encounter kidney stones at least once during their lifetime [2]. Factors contributing to the development of kidney stones include middle or advanced age, a familial history of stone disease, low fluid intake, protein and salt-rich diets, sedentary lifestyles, overweight or obesity, and certain genetic or inflammatory conditions [3].

Visualization of kidney stones is achieved through medical imaging modalities such as ultrasound, MRI, and CT scans, with the increased use of CT contributing to improved detection rates. The categorization of kidney stones into calcium, uric acid, struvite, and cysteine stones is based on their composition and formation [4]. Treatment methods vary according to the type of kidney stone, with options including shock wave lithotripsy, ureteroscopy, percutaneous nephrolithotomy, percutaneous nephron lithotripsy, and open surgery [5].

Despite the evolution of medical imaging techniques, challenges such as low resolution, noise induced distortions, a high volume of patients, and a shortage of specialists can impede accurate evaluations. To address these challenges, artificial intelligence (AI)-based systems have emerged as a promising solution. Presently, deep learning methodologies, a subset of AI, demonstrate remarkable accuracy in diverse areas, including medical image processing and biomedical signal analysis [6,7].

Deep learning is a subfield of AI that deals with machine learning like algorithms that have one or more hidden layers. Feature extraction happens automatically in deep learning. This feature makes it possible to work with larger data sets without the need for human intervention. Important features may disappear from the data in machine learning because the features identified in the data are selected. While crucial information is automatically extracted from the input in deep learning, it also has the ability to independently extract new features [8]. Although the human factor is more effective in machine learning, it is less in deep learning. Deep learning has becoming more popular as a method for working with large amounts of data since it requires less human intervention in tasks like feature extraction [9].

The aim of this thesis is to detect and classify kidney stones, which play an important role in number of kidney diseases, by using deep learning and object detection methods. Under this main aim, it is also planned to implement and compare different methods. Within the scope of this thesis, Faster R-CNN, YOLO, and a customized CNN model were developed using open source data set.

In the first part of this thesis, titled "Introduction," a brief overview of the study is provided. The introduction outlines the background, importance, and purpose of the thesis. In the second chapter, general information about the kidney, including its anatomy and physiology, kidney stone disease, and methods of detecting kidney stones, is presented. The third chapter reviews the literature on studies related to the classification and detection of kidney stones. In the fourth chapter, theoretical information on AI, including supervised, unsupervised, and reinforcement learning, as well as deep learning and artificial neural networks (ANN), is provided. The fifth chapter covers details about the dataset, the programming language used, the platform utilized, and features of the computer. In the sixth chapter, information about data pre-processing, data augmentation, image annotation, and a detailed explanation of Faster R-CNN, YOLO, CNN structures, and the model performance evaluation metrics are given. In the seventh chapter, a discussion section is included, comparing the findings obtained with the developed models and the results of other studies related to the subject.

CHAPTER 2

KIDNEY

The kidneys are two vital organs that are important for survival. The kidneys are located on either side of the spine, at the bottom of the rib cage, and behind the abdomen. The kidneys are shaped like a bean or bean seed and are about the size of a fist [10]. The main functions of the kidneys are based on filtering the blood from accumulated impurities and toxins, balancing the levels of salts, minerals, and water, thus helping to regulate blood pressure. The kidneys are involved in the production of red blood cells and in revitalizing vitamin D into a form that the body can use to absorb calcium from food, thus maintaining bone strength [11]. Each kidney consists of about one million nephrons, which are comprised of two main components: the glomerulus and the tubule. Within the nephron, the glomerulus acts as a network of small blood vessels responsible for filtering the blood. At the same time, the tubule performs the essential functions of reabsorbing valuable substances and eliminating waste products, ultimately producing urine [12].

2.1. ANATOMY OF THE KIDNEY

Kidney has an approximate length of 11 cm, a width of 6 cm, and a thickness of 3 cm. The left kidney is typically longer than the right kidney. Due to the liver’s position, the right kidney also tends to sit lower than the left kidney. The mean kidney weight is 150 g in males and 135 g in females [13]. Based on the glomerular filtration rate (GFR), the kidneys filter more than 150 L of fluid per day, but less than 1% of the filtered fluid is actually excreted in the urine [14]. When examining a kidney through a frontal section, three well-defined areas can be observed: the cortex, medulla, and pelvis. The outermost layer, known as the renal cortex, appears light in color and displays a granular texture. Situated beneath the cortex is the renal medulla, which has a darker reddish-brown hue. Within the medulla, cone-shaped tissue masses referred to as medullary or renal pyramids can be observed [15]. The renal pelvis is situated at the superior end of the ureter, displaying a flattened structure [16]. A frontal section of the kidney is illustrated in Figure 2.1 [17].

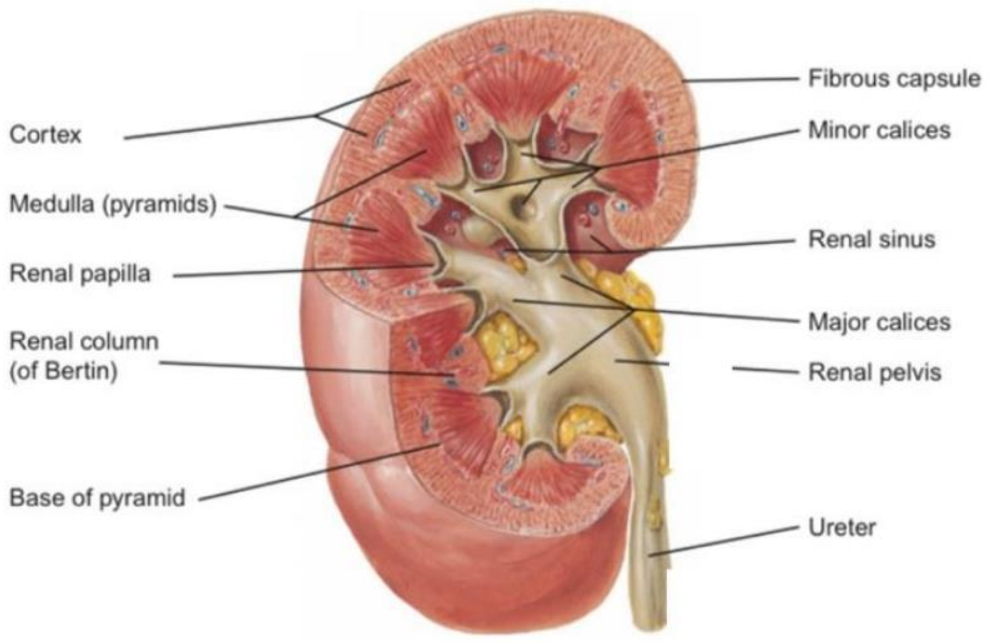


Figure 2.1. Kidney anatomy [17].

2.2. PHYSIOLOGY OF THE KIDNEY

The kidneys have four main functions regulating the body’s fluid and electrolyte balance, producing hormones, eliminating waste products generated during metabolism, and performing specific metabolic activities. One of the crucial roles of the kidneys is the excretion of nitrogenous waste substances, including urea, creatinine, and ammonia ions, through urine. Therefore, any notable changes in renal function lead to the accumulation of these waste products within the body [18].

Each nephron performs the task of filtering a small quantity of blood. Within the nephron, there is a filtering component called the glomerulus, along with a tubule. The nephrons operate through a two-step process. Initially, the glomerulus allows fluid and waste products to pass through it, while blocking the passage of blood cells and large molecules, particularly proteins. Subsequently, the filtered fluid moves through the tubule, which selectively reabsorbs necessary minerals back into the bloodstream while eliminating waste substances. Eventually, the production of urine is the outcome of this process [10].

The kidneys release several hormones, including erythropoietin (EPO), a peptide hormone that is crucial for the production of red blood cells in the bone marrow. Additionally, the kidneys play a role in the synthesis of 1,25-dihydroxyvitamin D3, the active form of vitamin D, which is essential for maintaining calcium homeostasis. This active form of vitamin D is produced by the proximal tubule cells through the action of specific enzymes [19]. Likewise, renin, an enzyme synthesized in the kidneys, serves a crucial function within the renin-angiotensin-aldosterone hormonal system, which aids in the regulation of blood pressure [20].

2.3. RENAL DISEASES

Renal diseases pose a significant threat to public health worldwide, with chronic kidney disease (CKD) affecting an estimated 8% to 16% of the global population. CKD is defined as a persistent impairment in kidney structure or function for a period exceeding three months. It is most commonly attributed to diabetes and hypertension [21]. If left untreated, CKD can progress to kidney failure, a critical condition necessitating either dialysis or a kidney transplant. Other common kidney diseases include diabetic nephropathy, glomerulonephritis, kidney stones, kidney tumors, pyelonephritis, and renal cell carcinoma [22].

2.3.1. Kidney Stones

Nephrolithiasis, also known as kidney stone formation, occurs when substances such as calcium or other minerals in the urine become excessively concentrated. As a result, these substances adhere to each other and form solid masses within the kidneys, leading to the development of kidney stones [23]. The term “nephrolithiasis” originates from the Greek words “nephros,” meaning kidney, and “lithos,” meaning stone [24]. Nephrolithiasis, following hypertension, is one of the most prevalent chronic kidney conditions and has been recognized since ancient times, with treatments documented in early medical texts. Kidney stones are a preventable source of illness. Annually, the United States faces a significant economic burden surpassing 5 billion dollars. This includes expenses associated with hospitalization, procedures to remove symptomatic stones, as well as the productivity loss due to missed work [25]. A sample of kidney stones is illustrated in Figure 2.2 [26].

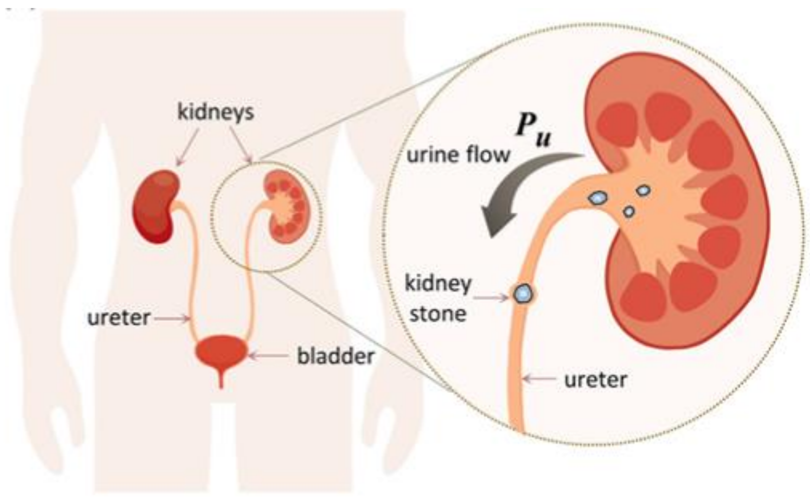


Figure 2.2. Kidney stone [26].

Generally, there are four types of stone formation: calcium, uric acid, struvite and cystine stones. Basically, kidney stones are categorized according to their primary crystalline composition. An illustration of types of kidney stones is shown in Figure 2.3 [27].

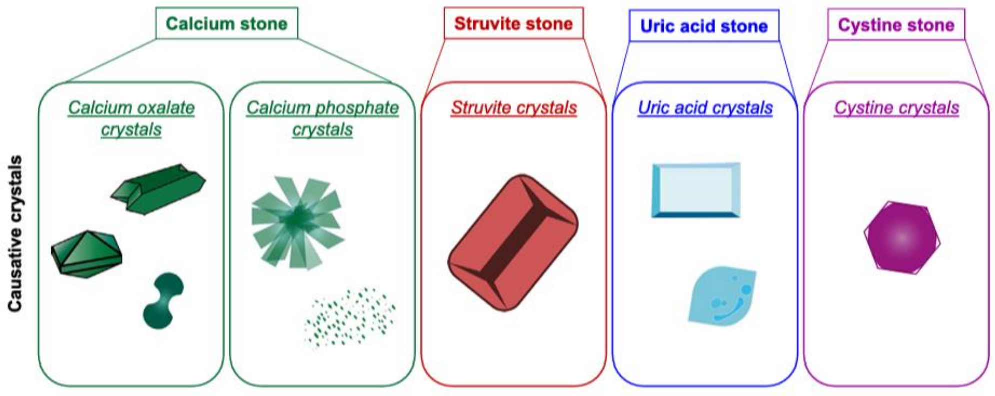


Figure 2.3. Types of kidney stones [27].

Kidney stones exhibit a wide range of sizes, ranging from as small as a grain of sand to as large as a pearl. However, it is important to note that the majority of kidney stones are typically quite small in size. The summer season is associated with a higher prevalence of kidney stones. Typically, smaller stones are expelled from the body, often accompanied by varying levels of discomfort. Conversely, larger stones can obstruct the normal flow of urine, leading to extreme pain when they become lodged in the ureters, bladder, or urethra [28]. Kidney stones are more prevalent in men compared to women, and there are slight variations in the types of stones found between the sexes [29].

Although establishing a direct link between climate and the formation of kidney stones is challenging, there is an increased prevalence of kidney stones in areas with high temperatures and during the summer season. In hot climates, increased water loss through sweating can lead to concentrated urine and reduced urine volume. This, in turn, raises urine acidity and the concentration of certain molecules, promoting the crystallization of these substances in individuals prone to kidney stone formation [30]. Most kidney stones are the result of a combination of genetic and environmental factors [31]. Dehydration resulting from inadequate fluid intake is a primary factor in the progress of kidney stones [32].

Diagnosis of nephrolithiasis requires confirmation of the presence of a kidney stone by observing its transition, removal, and destruction, or by imaging or surgery to confirm the presence of a stone in the urinary tract. As a part of the investigation, a comprehensive medical history and physical examination are performed on people with suspected kidney stones. However, to establish a clinical diagnosis, it is usually necessary to complement these evaluations with suitable imaging methods [33].

2.4. DIAGNOIS OF THE KIDNEY STONES

In the management of patients with renal stone disease, imaging plays a significant role in various aspects, including the initial diagnosis, development of treatment plans, and monitoring the effectiveness of medical therapy or urologic interventions during follow-up [34]. The use of imaging techniques is crucial in the evaluation of kidney stones, serving as a significant diagnostic tool and the first step in determining the most suitable therapeutic options for their treatment. The choice of the most suitable imaging modality for kidney stones involves considering several factors, including the clinical setting, patient’s body composition, expense implications, and the patient’s sensitivity to ionizing radiation. Among the available imaging modalities, Computed Tomography (CT) scans, ultrasonography, and kidney ureter bladder (KUB) plain film radiography are widely utilized in clinical practice for the evaluation of kidney stones [35].

Non-contrast CT of the abdomen and pelvis is considered the gold standard for accurately diagnosing kidney stones, providing highly accurate results. However, one drawback is that it exposes patients to ionizing radiation. On the other hand, ultrasonography, although traditionally having lower sensitivity and specificity compared to CT, offers the advantage of being a radiation-free imaging modality. When evaluating patients with a history of stone disease, KUB plain film radiography is most beneficial for detecting any growth or changes in the stones over time. However, its usefulness in diagnosing acute stones is limited. MRI provides the potential for radiation-free 3D imaging, but its ability to visualize stones is currently challenging, and it can be an expensive option [35].

CHAPTER 3

LITERATURE REVIEWS

Deep learning is an AI technique that enables computers to utilize provided data in order to make predictions. In recent times, there has been a remarkable progress in the field of computer vision and deep learning algorithms, which has led to their widespread adoption for analyzing medical images [36]. This section provides an overview of various studies conducted on the detection of kidney diseases, highlighting their key findings, and summarizing their outcomes.

Patro et al. [37] proposed a study in which they introduced an approach for automatic kidney stone diagnosis. They developed a custom CNN model that utilized a novel Kronecker product structure. During the experimentation phase, a database consisting of 1799 coronal CT scans was utilized. This database included scans from 433 individuals, with 790 scans showing kidney stones and 1009 scans representing normal healthy cases. To validate the proposed method, a 10-fold cross-validation (CV) technique was employed. the performance of the proposed Deep Kronecker Network (DKN) was evaluated and compared with traditional approaches such as CNN, Residual Neural Network (ResNet), and AlexNet. The automated model developed in this study achieved an accuracy of 98.56% in detecting kidney stones using CT images.

Razmjooy & Yan [38], introduced a novel automatic method aimed at accurately diagnosing kidney stones. The primary objective was to propose an improved version of a metaheuristic technique called Fractional Order Coronavirus Herd Immunity Optimizer (FO-CHIO), which was integrated into a modified version of a Deep Belief Network (DBN). They used a dataset comprised a total of 12446 images, which were categorized as follows: 5077 normal images, 3709 cysts, 2283 tumor images, and 1377 stone images. Finally, a comparison was conducted between the proposed DBN/FO-CHIO method an other state-of-the-art approaches. The results of the simulations revealed that the recommended DBN/FO-CHIO approach exhibited superior performance in terms of an accuracy of 97.98%. Additionally, the proposed DBN/FO-CHIO method demonstrated exceptional sensitivity with a value of 92.99%, surpassing the performance of the other comparison algorithms.

Caglayan et al. [39] conducted a study to evaluate the effectiveness of a deep learning model in detecting kidney stones of varying sizes in different planes using unenhanced CT images. A total of 455 patients who underwent CT scanning for kidney stones between January 2016 and January 2020 were included. Among these patients, 405 were diagnosed with kidney stones, while 50 patients did not have kidney stones. The patients were divided into different groups based on the size of their renal stones: group 1 included patients with stones measuring 0-1 cm, group 2 included patients with stones measuring 1-2 cm, and group 3 included patients with stones larger than 2 cm. A total of 2959 CT images from 455 patients were reviewed by two radiologists across three different planes. Among the different planes examined, the sagittal plane demonstrated the highest sensitivity and specificity in comparison to the other plane. The deep learning model achieved accuracy rates of 78%, 68%, and 70% in the axial plane for the testing group. In the coronal plane, the accuracy rates were 63%, 72%, and 64%. Lastly, in the sagittal plane, the accuracy rates were 85%, 89%, and 93% for the respective testing groups.

Gurkan et al. [40] The You Only Look Once (YOLO) architecture designs were employed to detect kidney, kidney cysts, and kidney stones, with the added support of explainable artificial intelligence (xAI) features. The performance analysis of these YOLO designs utilized CT images categorized into three classes: 72 images of kidney cysts, 394 images of kidney stones, and 192 images of healthy kidneys. The dataset was split into three sets, with 75% used for training, 10% for validation, and 15% for testing. Both of tiny Yolov7 and Yolov7 were utilized. The YOLOv7 architecture design attained the following results, with values of 0.85, 0.882, 0.829, and 0.854 for mAP (0.5), precision, sensitivity, and F1 score, respectively.

Baygin et al. [41] utilized a publicly available dataset consisting of 1799 CT images. These images were captured with dimensions of 512x512 pixels. The dataset comprised two classes: normal and kidney. As a part of the pre-processing steps, the CT images underwent several techniques to ensure their compatibility with the deep learning model. One of these techniques involved resizing the images to 224x224 pixels. A novel ExDark19 classification model was introduced to detect kidney stones in CT images. The proposed method, based on the concept of vision transform (ViT), demonstrated high classification performance in analyzing CT images. The primary objective of ExDark19 was to achieve accurate classification results while minimizing the computational time required for kidney stone detection. In addition to that, the iterative neighborhood component analysis (INCA) technique was utilized to select the most informative feature vectors. These selected feature vectors were then fed into a k-nearest neighbor (kNN) classifier for the purpose of kidney stone classification. The evaluation of the proposed ExDark19 model was carried out using a 10-fold CV strategy. The results demonstrated an accuracy of 99.22% with the 10-fold CV approach and 99.71% using the hold-out validation method.

The focus of the research [42] revolved around three significant categories of renal diseases: kidney stones, cysts, and tumors. To construct an AI based diagnostic system for kidney diseases, a comprehensive collection of 12446 whole abdomen and urogram CT images was gathered and annotated. Six models were developed for the purpose of kidney disease classification. Among these models, three were based on recent state-of-the-art variants of ViT, namely External Attention Transformer (EANet), Compact Convolutional Transformer (CCT), and Swin Transformer (Shifted Window Transformer). The other three models utilized well-known deep learning architectures: ResNet, Visual Geometry Group (VGG16), and Inception V3. These models were employed to leverage the strengths of both recent advancements and established deep learning techniques in the field of kidney disease classification. After testing the models, VGG16 and CCT exhibited good performance. However, the Swin Transformer model surpassed them all in terms of accuracy, achieving an accuracy rate of 99.30%.

In a study conducted by Yildirim et al. [43], the focus was on the detection of kidney stones. A total of 1799 non-contrast CT images of the brain were collected from 500 patients with urinary and kidney stone-related medical conditions. Out of these, 67 patients were excluded from the study based on a specific criterion. A total of 433 subjects were included in the study, consisting of 278 patients with kidney stones and 165 patients without stones (normal). The labeling procedure, where the presence or absence of stones was identified, was performed by experts, including a radiologist and a urologist. Notably, no segmentation was applied to the CT images during the labeling process. The researchers utilized rotation and zooming techniques for data augmentation. After augmenting the images, the next step involved feeding images them into a deep learning model for further analysis. Specifically, they employed the XResNet-50 model for the detection process, leveraging its capabilities to effectively classify and identify kidney stones. For parameter tuning of the model, Adam optimization and cross-entropy loss were employed. By utilizing these techniques, it was aimed to optimize the performance of the model. As a result, the model achieved an accuracy of 96.82%.

S. Sudharson & P. Kokil [44] proposed a paper focused on kidney disease detection and classification. The study utilized a dataset comprising 4940 ultrasound images acquired from various sources. The dataset was categorized into four distinct categories, namely cyst, tumor, stone, and normal. The researchers proposed a computer-aided diagnosis (CAD) system to address the issue of speckle noise. To ensure high image quality, a perception-based image quality evaluator (PIQUE) score was utilized as a part of the image selection process. Images with a PIQUE score of $P < 50$ were selected for further analysis. Subsequently, data augmentation techniques rotation, translation, and cropping were applied to the selected images. The dataset was then divided into train and test sets, with 90% of the images allocated to the training process and the remaining 10% reserved for testing the CAD system's performance. In the test dataset, different speckle noise levels were intentionally added by utilizing noise model. The proposed CAD system then performed the de-speckling process using a pre-trained network. Specifically, a pre-trained ResNet-101 model was utilized for the feature extraction process. This model played a crucial role in extracting informative features from the noisy images, aiding in the subsequent steps of the CAD system's analysis and diagnosis of kidney diseases. For the classification means Support Vector Machine (SVM) was employed. In conclusion, the model achieved an accuracy score of 87.31% when tested with a speckle noise ratio of 0.02. This indicates the model's ability to effectively classify and diagnose kidney diseases, even in the presence of a certain level of speckle noise.

Parakh et al. [45] the aim of this study was to examine the diagnostic accuracy of a cascading CNN for detecting urinary stones in unenhanced CT images. Additionally, the researchers sought to assess the performance of pretrained models when supplemented with labeled CT images acquired from various scanners. In this retrospective clinical study, unenhanced abdominopelvic CT scans from 535 adults who were suspected of having urolithiasis were utilized. 279 of them (comprising 165 men and 114 women) were diagnosed with stones, while the remaining 256 patients (including 140 men and 116 women) did not have stones. The effectiveness of a cascading CNN for the detection of urinary stones was demonstrated. In this approach, the urinary tract was first detected by the initial CNN model, while the subsequent CNN model was responsible for detecting the presence of stones. This cascading approach allowed for a more accurate and specific identification of urinary stones within the imaging data. The CNN utilized was initially pretrained with ImageNet, which consisted of 1.2 million natural images spanning 1000 categories. Following the ImageNet pretraining, the model was fine-tuned using an in-house dataset called GrayNet. Which contained labeled CT images specifically designed for human anatomy recognition. This fine-tuning process resulted in the generation of a pretrained model called the GrayNet pretrained model. The weights of this pretrained model were then utilized to initialize the CNN models employed for urinary tract identification and stone detection tasks. The network achieved Area Under Curve (AUC) of 0.954.

Längkvist et al [46] a CNN was employed to detect ureteral stones in thin-slice CT scans. The primary focus of this research was to develop an automatic detection method for ureteral stones that do not rely on specific feature selection or segmentation techniques. The complete dataset used in this study comprised 465 unenhanced abdominal CT scans that were clinically acquired. To train the CNN, the scans were randomly divided into a training set (80% of the dataset) and a testing set (20% of the dataset). Prior to the division, 28 scans that contained stones that were either too small or too large were removed from the dataset. The achieved sensitivity for the model was 100%.

Table 3.1 provides an overview of previous research studies, listing the algorithms utilized and the corresponding years of each study and additional information on the used datasets.

Table 3.1. Overview of previous research studies.

No	Authors	Reference	Year	Input images	Number of Images	Model	Results
1	Patro et al.	[37]	2023	CT	1799	DKN	Accuracy: 98.56%.
2	Razmjooy & Yan	[38]	2023	CT	12446	DBN/FO-CHIO	Accuracy: 97.98%.
3	Caglayan et al.	[39]	2022	CT	2959	xResNet50	Accuracy: 85%, 89%, and 93% in the sagittal plane.
4	Gurkan et al.	[40]	2022	CT	658	YOLO v7	mAP (0.5): 0.85, Precision: 0.882, Sensitivity: 0.829, F1 score: 0.854.
5	Baygin et al.	[41]	2022	CT	1799	ExDark19	10-fold CV Accuracy: 99.22%, hold-out Accuracy: 99.71%.
6	Islam et al.	[42]	2022	CT	12446	Swin transformers	Accuracy :99.30%.
7	Yildirim et al.	[43]	2021	CT	1799	xResNet50	Accuracy: 96.82 %
8	S. Sudharson & P. Kokil	[44]	2021	Ultrasound	4940	SVM	Accuracy: 87.31%
9	Parakh et al.	[45]	2019	CT	535	Dual CNN	AUC: 0.954
10	Längkvist et al.	[46]	2018	CT	465	CNN	Sensitivity: 100%

CHAPTER 4

ARTIFICIAL INTELLIGENCE

AI is an emerging field that utilizes computer technology to explore and progress theories, methods, techniques, and application systems aimed at simulating, extending, and amplifying human intelligence [47]. AI, as a scientific field, aims to enable machines to solve complex problems in a manner that resembles thinking and problem-solving capabilities of human [48]. Machine learning and deep learning are two integral components of the field of AI, with deep learning being the more recently introduced technique [49].

4.1. MACHINE LEARNING

Machine learning is the discipline that revolves around developing algorithms and statistical models, allowing computer systems to perform tasks by analyzing patterns and making inferences, without relying on explicit instructions. It is a branch of AI that aims to extract information from given inputs, recognize patterns, and make decisions with minimal human intervention [50]. Machine learning is a type of software that enhances its performance in the future by learning from past experiences. It falls under the umbrella of AI, aiming to simulate human intelligence within computer systems [51]. There are various types of machine learning, including supervised learning, unsupervised learning and reinforcement learning offering different approaches to the learning process [52]. Supervised learning involves the use of classification and regression techniques, while unsupervised learning utilizes clustering techniques. Diagram of machine learning algorithms is visually represented in Figure 4.1 [52].

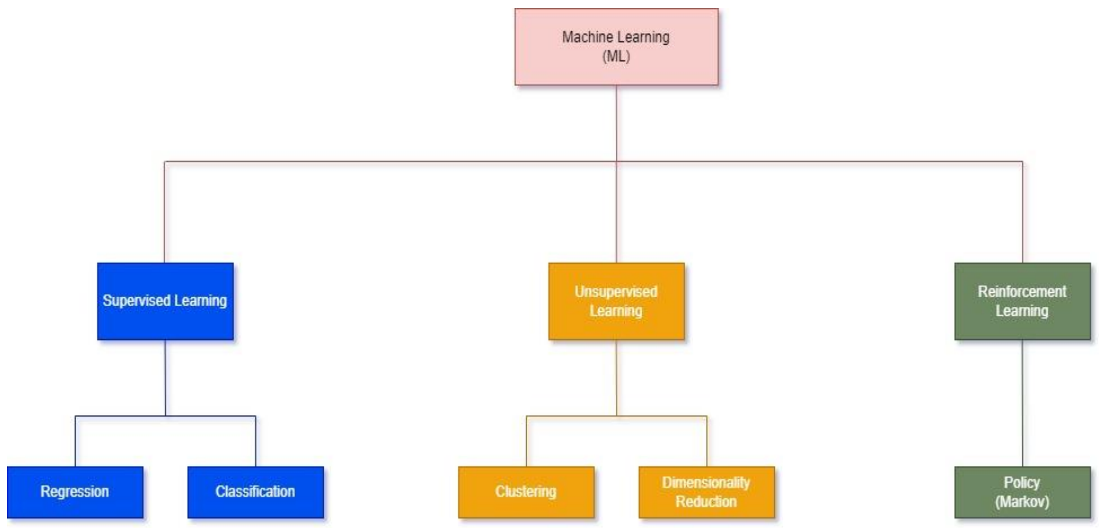


Figure 4.1. Diagram of machine learning algorithms [52].

4.1.1. Supervised Learning

Supervised learning, a subset of machine learning, relies on labeled data to train models for both prediction and detection tasks. In this approach, known outputs are assigned to each known input in the training data. Adequate availability of labeled input-output data enables supervised learning to achieve high performance in estimation. Machine learning can be classified into two main categories based on the types of model outputs in supervised learning. If the output is continuous, it is considered a regression problem, whereas if the output is discrete and represents a value from a finite set of predefined options, it is categorized as a classification problem [53].

In regression, the objective is to predict a continuous value label for an unlabeled sample using a trained model. The model makes predictions based on the patterns and information which were learned from the labeled dataset. While classification involves predicting the class to which a new test sample belongs, utilizing a labeled training set where each sample is associated with a known class [53].There is a range of supervised machine learning algorithms available, and some of the commonly used ones include Decision Tree (DT), Random Forest (RF), kNN, SVM, ANN, Naïve Bayes (NB), Linear Regression (LR), and Linear Discriminant Analysis (LDA) [54].

4.1.2. Unsupervised Learning

Unsupervised machine learning methods hold great significance as analytical tools for handling and interpreting high-dimensional data. By identifying and underlying both patterns and hidden structures in complex datasets, these techniques effectively simplify the understanding and analysis of high-dimensional data [55]. Clustering techniques, dimensionality reduction algorithms, autoencoders, and generative adversarial networks (GANs) are among the commonly utilized unsupervised techniques [56]. Unsupervised learning offers advantages over supervised learning in certain tasks by eliminating the need for annotated data guidance, making it more suitable for handling those tasks [57]. The differentiation between supervised learning and unsupervised learning is shown in Figure 4.2 [57].

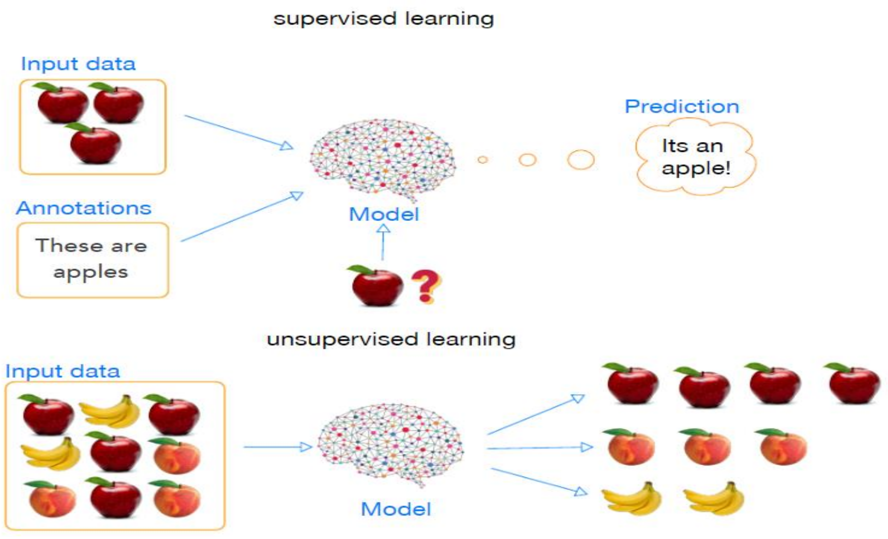


Figure 4.2. The difference between supervised learning and unsupervised learning [57].

The picture consists of two scenarios: supervised and unsupervised learning. In the supervised learning scenario, the model is trained using labeled samples of three apples with corresponding annotations. Its goal is to predict accurate annotations for new and unseen data. On the other hand, in the unsupervised learning scenario, a mix of apples, bananas, and peaches are presented without annotations. The model's objective is to discover patterns or structures within the unlabeled data and classify or group the fruits based on inherent similarities or relationships.

4.1.3. Reinforcement Learning

Reinforcement learning involves learning by actively interacting with an environment, making diverse actions, and encountering both successes and failures in the pursuit of maximizing the rewards obtained. Unlike supervised learning, where the correct actions are explicitly provided for each encountered situation, reinforcement learning aligns with natural learning processes where there is no available supervisor. Instead, the learning process evolves through trial and error, allowing the agent to learn from its own experiences [58]. Reinforcement learning can broadly be categorized into two main techniques: model-based and model-free approaches. Examples of model-based Reinforcement learning approaches include AlphaZero and AlphaGo. Examples of model-free Reinforcement learning algorithms include Q-learning, Deep Q Network (DQN), Monte Carlo Control, and State-Action-Reward-State-Action (SARSA) [59].

4.2. DEEP LEARNING

The evolution of information technologies has been accompanied by the parallel processing capabilities of computers, enabling the growth of AI technology, and facilitating the expansion of ANN architectures with a higher number of artificial neurons. Deep learning, specifically, has emerged as a specialized form of ANN, benefiting from these advancements [60]. The term deep in deep learning, in fact, refers to the series of consecutive intermediary layers known as hidden layers. By adopting an incremental approach and learning layer by layer, deep learning methods are able to construct sophisticated representations of the data [61].

Deep learning has demonstrated its effectiveness in challenging tasks and delivering remarkable solutions with high accuracy across various domains, including text, signal, image, and video. It has particularly shown a promise in the fields such as medical image analysis and is regarded as an essential methodology for the future applications in the healthcare sector [61].

ANN is an advanced system and computational methodology utilized in machine learning. At the same time, ANN is the fundamental component of deep learning algorithms. ANN serves the purpose of acquiring knowledge, demonstrating it, and ultimately applying it to maximize the output responses of complex systems, respectively [62].

ANN is similarly structured to the human brain, where neuron nodes are interconnected in a network-like manner. In the human brain, billions of cells called as neurons play a vital role. These neurons consist of a cell body responsible for processing information by sending and receiving inputs and outputs from the brain. The concept behind ANN is influenced by the functioning of the biological neural system. The aim is to process data and information in a manner that facilitates learning and the estimation of knowledge. Neurons within ANN are intricately interconnected and arranged in layers. The input layer is responsible for receiving data, while the output layer produces the final result. Between these layers, there are one or more than one hidden layer. These hidden layers serve as intermediate processing stages within the network [62]. A sample of ANN architecture is shown in the Figure 4.3 [62]. The depicted figure shows a sample of ANN architecture that includes an input layer with five interconnected neurons, two hidden layers with their respective connections, and one output layer with four interconnected neurons. This arrangement enables the network to effectively process input data, performs computations within the hidden layers, and produces meaningful outputs based on the processed information.

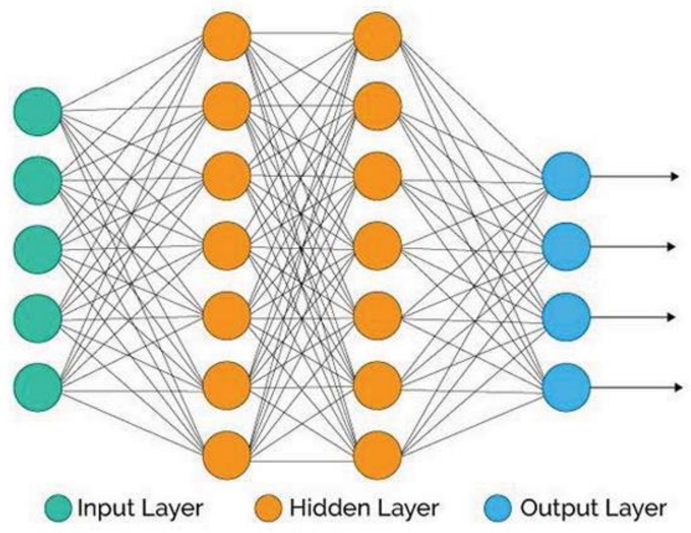


Figure 4.3. A sample of ANN architecture [62].

ANN offer numerous advantages compared to traditional machine learning algorithms. It possesses significant numerical capabilities, enabling them to perform multiple tasks simultaneously. Moreover, it is well-suited for systems that demand a high level of fault tolerance. Trained ANN exhibits the remarkable ability to generate output even when presented with incomplete information, showcasing their adaptability. However, it is important to note that solving complex problems often requires the utilization of multilayer and multi-neuron ANN models. Additionally, the training process of ANN can be time-consuming due to the extensive number of neurons and links within the network structure [63].

CHAPTER 5

MATERIALS

This section provides an overview of the materials utilized in this thesis as dataset, the used programming language, the used platform, and the properties of the used computer.

5.1. DATASET

The data used in this study was sourced from the kidney stone detection repository available on GitHub [64]. This repository contains a diverse collection of coronal CT scans obtained from various institutions and scanners, with the aim of developing robust and precise algorithms for automated kidney stone detection.

The researchers [43] collected Non-Contrast CT images from a total of 500 patients who were admitted to Elazığ Fethi Sekin City Hospital in Turkey for urinary system stone disease. However, certain criteria were applied to exclude specific patients from the study. These exclusions encompassed 67 patients who had double-J (pigtail) ureteral catheters, patients under the age of 18 or over the age of 80, individuals with a single kidney, patients with kidney anomalies, and those with atrophic kidneys.

The dataset utilized in this study consists of CT scans obtained from individuals of varying genders (both male and female) with ages ranging from 18 to 80 years, all of whom have received a diagnosis of kidney stones. The confirmation of kidney stone diagnoses was established through a thorough examination of the scans by radiologists or urologists. To ensure precise detection and meticulous annotation of kidney stones, each CT scan in the dataset underwent evaluation by at least two radiologists. Within the scope of this study, 268 participants presented positive results on a stone test, indicating the presence of kidney stones, while 165 participants reported normal results without any indication of kidney stones. The dataset used for this study comprises a total of 1799 coronal CT scans. Among these scans, 1009 correspond to normal subjects without kidney stones, while the remaining 790 scans belong to patients diagnosed with kidney stones. All CT scans were acquired with the patient in a supine position on a single scanner, and no contrast agent was administered during the imaging procedure [43]. Examples of the used dataset is shown in Figure 5.1 [43]. A description of the dataset utilized is illustrated in Table 5.1.

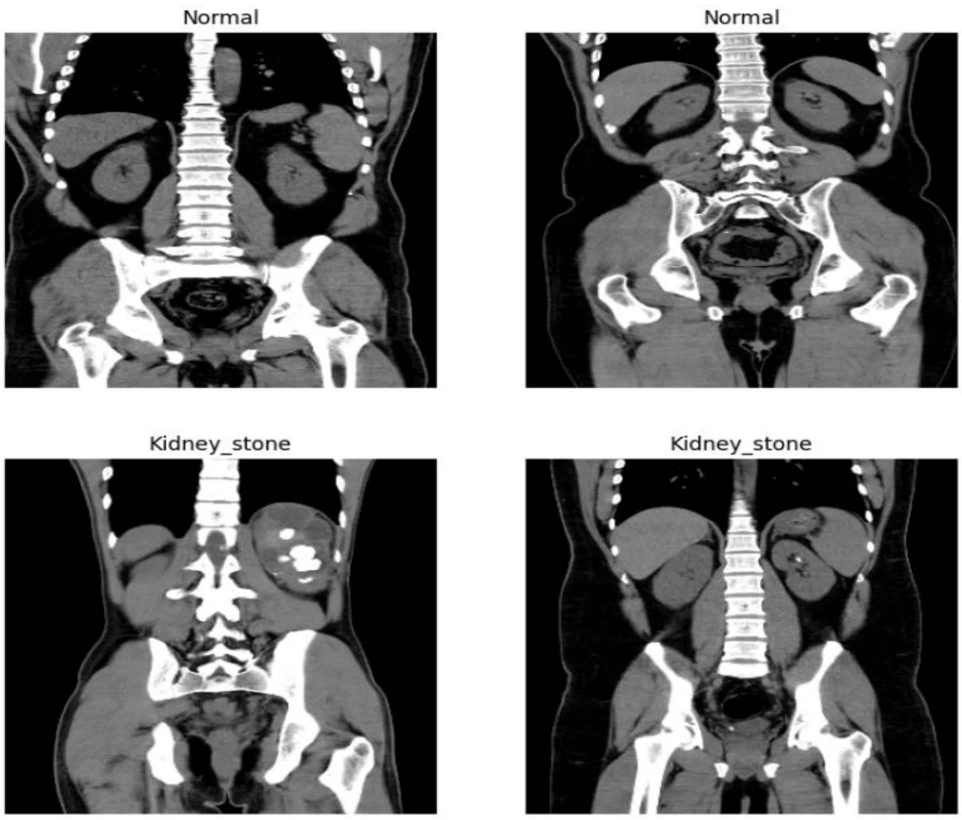


Figure 5.1. Samples of CT images in the dataset [43].

Table 5.1. Description of the dataset.

	Normal	Kidney stone	Total
Nubmer of patients	165	268	433
Coronal CT scans	1009	790	1799

5.2. PYTHON

Python is an incredibly robust programming language that excels in areas such as data science, scientific computing, and machine learning. It offers great flexibility, is known for its ease of learning, and boasts a wide range of libraries and packages to support these fields. Python remains the top choice for scientific computing, data science, and machine learning due to its ability to balance both performance and productivity. It achieves this by allowing the use of low-level libraries for optimized execution while also providing clean and intuitive high-level application programming interfaces (APIs) for seamless development [65].

5.2.1. Advantages of Python

Python is widely popular among AI developers for several reasons:

- Ease of use: Python has a simple and easy-to-understand syntax, making it accessible for new data scientists.
- Flexibility: Python is not only suitable for software development but also enables handling data analysis, numerical and logical computations, and web development. It is extensively used in web development frameworks like Django, TurboGears, and Tornado, making it a preferred choice for developers with application and web development skills.
- Building analytics tools: Python is well-suited for building data analytics tools, which are crucial for assessing performance in various business domains. It allows easy extraction of insights and correlation analysis from large datasets, making it significant in self-service analytics and data mining.
- Deep learning capabilities: Python offers a range of packages like TensorFlow, Keras, and Theano that assist data scientists in developing deep learning algorithms. These packages enable the creation of ANN that mimic human brain architecture, providing superior support for deep learning tasks.
- Strong community base: Python has a large and active community of developers and data scientists. This community provides a platform for sharing

ideas, discussing issues, and collaborating on projects. Platforms like Python.org, Fullstackpython.com, and realpython.com offer resources and support for Python developers, fostering continuous improvements and advancements in the language [66].

5.3. GOOGLE COLABORATORY

Google Colaboratory is a project aimed at promoting machine learning education and research. Colaboratory notebooks are built on the Jupyter platform and function similarly to Google Docs, allowing for easy sharing and collaboration among multiple users on the same notebook. Google Colaboratory comes equipped with pre-configured Python 2 and 3 runtimes, featuring essential machine learning and AI libraries such as TensorFlow, Matplotlib, and Keras. It’s noteworthy that the runtime’s virtual machine (VM) deactivates after a certain period, leading to data and configuration loss. However, the notebook itself remains intact, and files can be transferred from the VM’s hard disc to the user’s Google Drive account [67]. The execution of notebooks in Google Colaboratory occurs within Linux-based VMs that are provided and managed by Google. These VMs enable computation using central processing units (CPU) and can also leverage specialized graphical processing units (GPU) and tensor processing units (TPU) for accelerated computation [68].

5.4. ROBOFLOW

Roboflow is a computer vision platform designed for various tasks such as data collection, data training, and pre-processing. This platform offers a wide range of features, including support for both public datasets and custom datasets. Additionally, Roboflow provides multiple annotation techniques and employs pre-processing techniques such as image resizing, orientation adjustment, and contrast enhancement. Roboflow is an online platform that offers free labeling and annotation services, eliminating the need to download additional software onto your computer. Its primary objective is to provide a secure environment for managing and annotating datasets, with the added convenience of accessing the platform from various devices such as tablets or smartphones [69].

5.5. PROPERTIES OF THE COMPUTER

The properties of the computer utilized in this thesis are presented in Table 5.2.

Table 5.2. Properties of the used computer.

Product	Property
CPU	i7-1165G7 2.80GHz
Read access memory (RAM)	16.00 GB
Hard disc	512 GB
Cache memory	12.00 MB
Display card	2.00 GB GDDR6
Operating system	Windows 11 (64-bit)

CHAPTER 6

METHODS

This chapter provides an illustration of the procedure of the execution of the proposed techniques of work in this thesis, as shown in Figure 6.1.

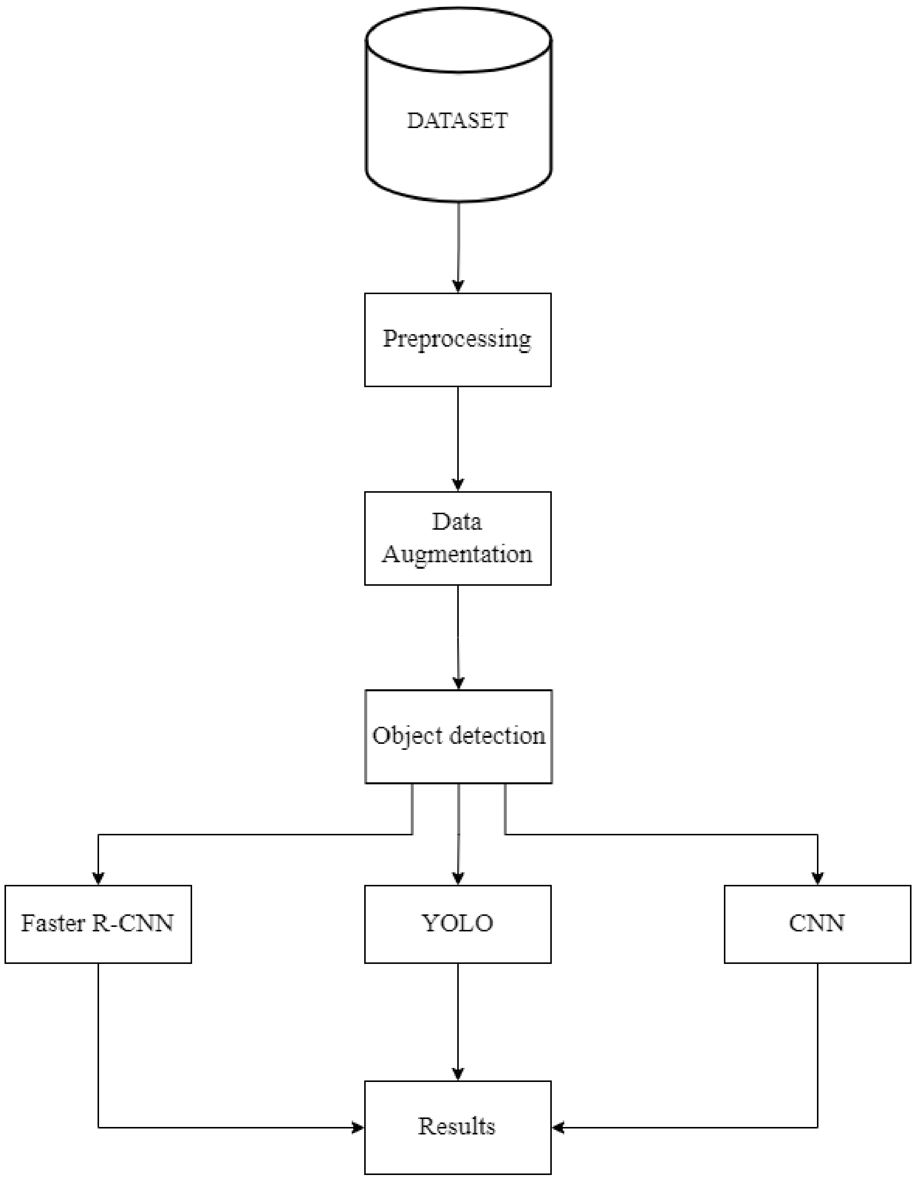


Figure 6.1. Flow chart of the proposed methods.

6.1. PREPROCESSING

The recognition or classification accuracy of objects is significantly influenced by the quality of the image. Superior image quality enhances recognition or classification accuracies compared to unprocessed images that contain noise. Extracting features from such unprocessed images becomes more arduous, resulting in a decrease in the object recognition or classification accuracy. To address issues caused by low-quality images, it is common to perform pre-processing before extracting features from the image [70]. Image preprocessing generally encompasses two primary types: data cleaning and data conversion. Data cleaning aims to minimize the presence of noise within the image, while data conversion focuses on resizing the image, converting it to color space like grayscale, and applying normalization techniques [71].

6.1.1. Resizing

Resizing images is an essential pre-processing step in computer vision, particularly when using deep learning models. Smaller images enable faster training of these models. When using larger images, the neural network needs to process and learn from a significantly larger number of pixels, leading to increased training time for the architecture [72].

In this thesis, the YOLO algorithms and CNN were utilized with resized images having dimensions of 640x640 and 300x300, respectively. As for the Faster R-CNN the original dimensions of images were utilized.

6.1.2. Detection of Region of Interest (ROI)

The region of interest (ROI) refers to a specific labeled area within an image, typically representing a small portion of the overall image [73]. By selecting the ROI based on its relevance to the medical condition or research question, medical professionals and researchers can direct their analysis, measurements, and diagnostic interpretations specifically to ROI. This targeted approach enables more precise and accurate analysis, as it eliminates the need to analyze the overall image [74]. A sample of the ROI is shown in Figure 6.2.

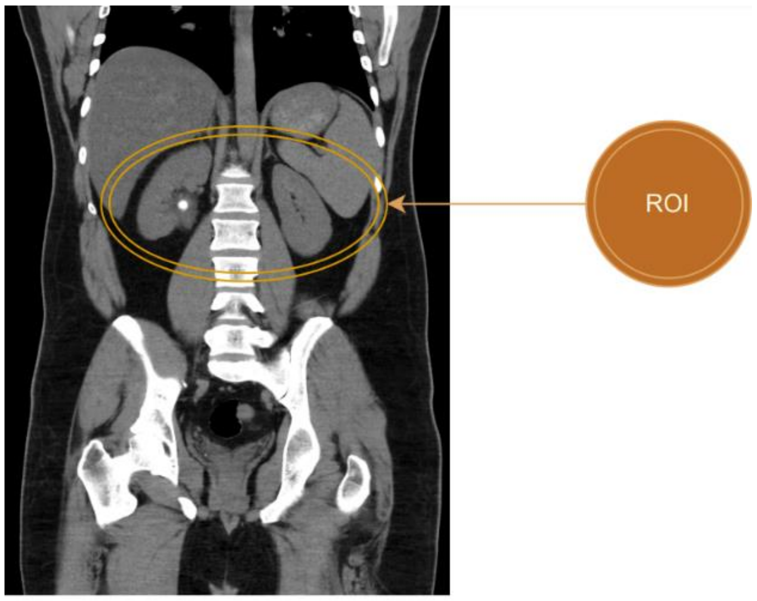


Figure 6.2. ROI of CT coronal image.

The image is a CT coronal image of a patient with kidney stones. ROI of the sample image has been marked as the kidneys which possesses the kidney stones within the specific region.

6.2. DATA AUGMENTATION

To achieve optimal performance, modern machine learning models generally rely on a large amount of high-quality annotated data. However, the collection and annotation processes for such data are usually manual and time-consuming, requiring significant resources. Obtaining an adequate amount of training data is often challenging in real-world application. Data augmentation has emerged as the most effective approach to address this issue. The primary objective of data augmentation is to increase the volume, quality, and diversity of the training data. Data augmentation involves the application of diverse techniques, including geometric transformations, noise injection, and generative models, to expand a dataset. This process enhances models' capacities to generalize and improve performances [75].

In this thesis, data augmentation techniques were utilized to improve the performances of CNN and YOLO algorithms. The CNN employed horizontal flip, zoom, shear, and rotation augmentations. Besides, points outside the boundaries of the images were filled according to the nearest pixel. On the other side, the YOLO algorithm utilized both horizontal flipping and 10-degree rotation. Table 6.1 displays the augmentation techniques utilized for the CNN and YOLO with their respective ranges.

Table 6.1. Augmentation Techniques for CNN and YOLO with respective ranges.

Type	**CNN**	**YOLO**
Horizontal Flip	True	True
Zoom	0.2	-
Shear	0.2	-
Rotation	20^{o}	10^{o}

6.3. IMAGE ANNOTATIONS

Over the past decade, object detection has experienced significant advancements and has emerged as a rapidly evolving sub-field of deep learning. This progress has led to increased complexity and broader applications of object detection models. Consequently, the demand for larger datasets and multi-format labeled annotations for training and testing these models has also risen. Image annotation is a specific type of data labeling that pertains to the labeling process of visual digital data. Generally, image annotation involves manual work [76].

LabelImg is a locally installed software that runs on the user’s machine. It provides a visual interface built with the Qt toolkit and is implemented in Python. This software supports multiple annotation formats such as PASCAL VOC (XML), YOLO, and CreateML. It is compatible with Windows, macOS, and Linux operating systems [76].

In this thesis, the annotation process was carried out by utilizing LabelImg. For the Faster R-CNN model, the resulting annotations were saved in XML files, while for the

YOLO model, the annotations were saved in TXT files. A sample of the annotated image from the dataset is shown in Figure 6.3. The annotated image is coronal CT scan including kidney stones. The annotations highlight the presence and location of kidney stones within the image.

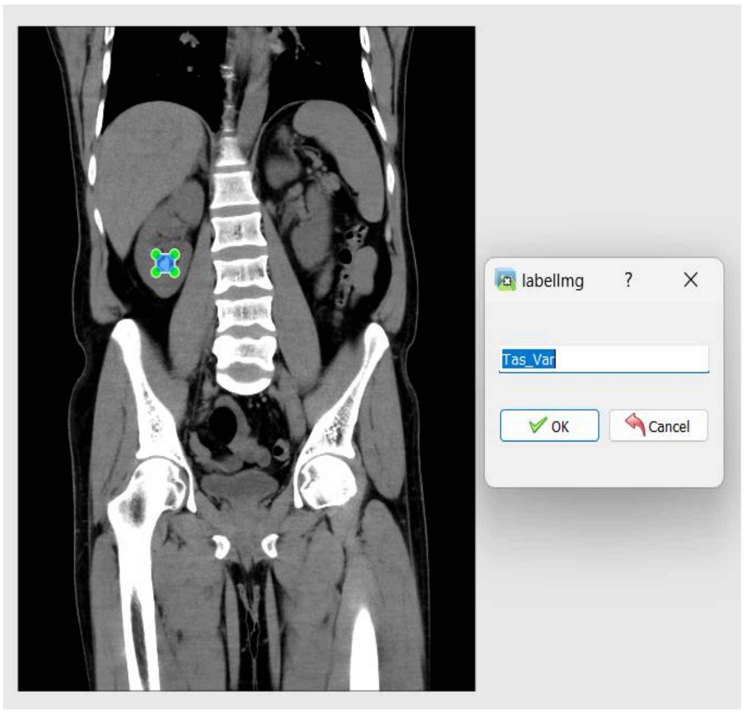


Figure 6.3. A Sample annotated image in LabelImg.

6.4. OBJECT DETECTION

In recent years, there has been a significant expansion in the field of computer vision research. Utilizing machine learning techniques has proven to be an effective approach for addressing various computer vision tasks. Object detection, a fundamental problem in computer vision, focuses on identifying and localizing instances of specific objects within digital images or videos. Object detection consists of two primary tasks: object localization and classification. Object localization involves determining the precise location and scale of one or multiple object instances by enclosing them with bounding boxes. On the other hand, classification refers to the process of assigning a class label for the detected object based on its visual characteristics and features [77]. Single-class object detection refers to the task of detecting and localizing a specific type of object within an image, where there is only one object of interest. In contrast, multi-class object detection involves detecting and localizing multiple objects belonging to same or different classes within an image [77]. This process is shown in Figure 6.4 which provides a comprehensive illustration of classification, localization, and segmentation for object detection [78].

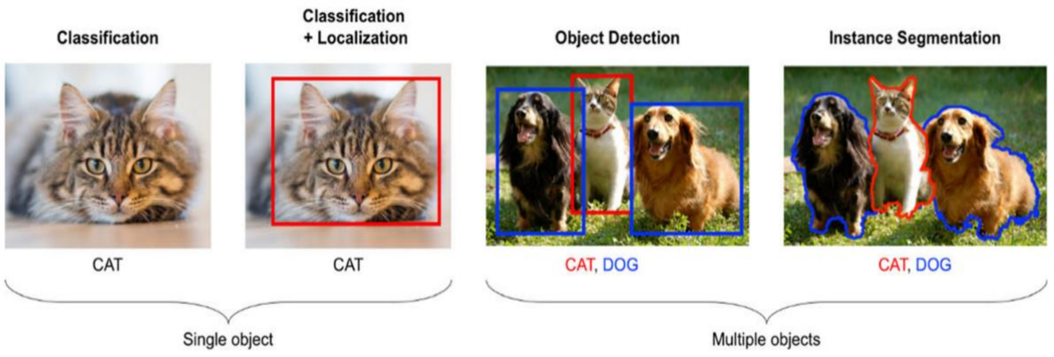


Figure 6.4. Classification, localization, and segmentation in object detection [78].

Object detection algorithms are broadly classified into two main categories: one-stage and two-stage algorithms, both predominantly relying on deep learning techniques. The key difference between these approaches lies in the generation of region proposals. One-stage object detection algorithms do not require a separate region proposal generation step. Instead, they directly predict object's class label and its corresponding bounding box coordinates. On the other hand, two-stage object detection algorithms follow a two-step process. Initially, they generate region proposals that are likely to contain objects of interest. These proposals are then classified and refined to obtain the final object detections [79]. One-stage object detection algorithms include YOLO, SSD, RetinaNet, DSSD, M2Det, and RefineDet, while two-stage object detection algorithms include R-CNN, Fast R-CNN, Faster R-CNN and Mask R-CNN [80]. Architectures of one-stage and two-stage object detection algorithms are shown in Figure 6.5.

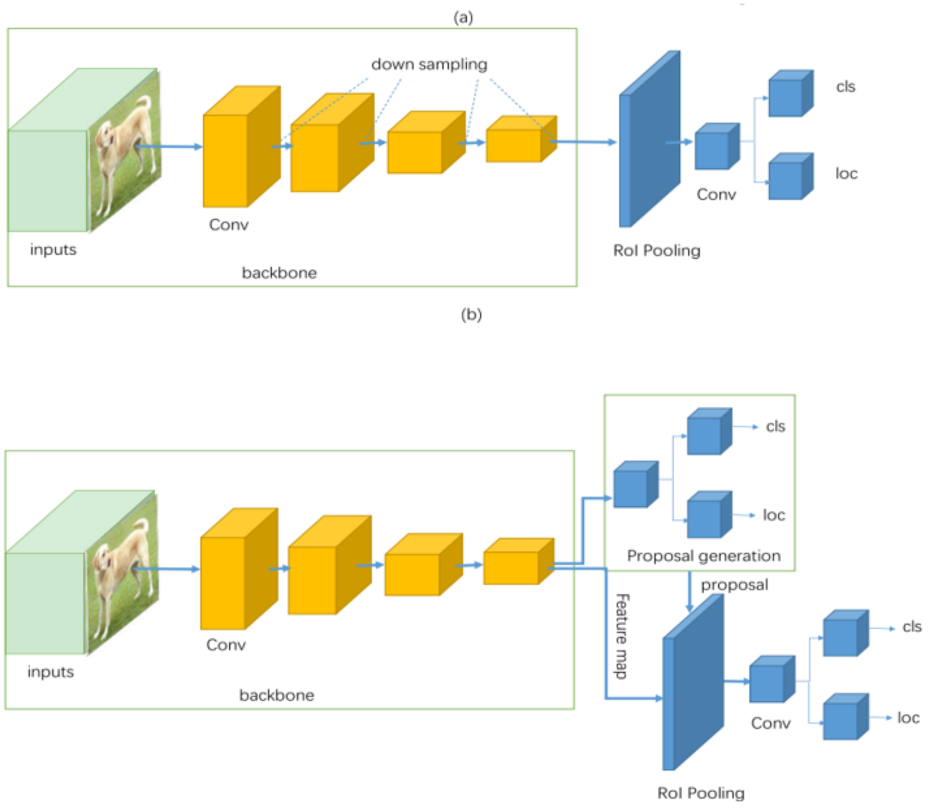


Figure 6.5. Architectures of object detection algorithms. (a) one-stage (b) two-stage [80].

Where in (a) the basic architecture of one-stage detectors involves predicting bounding boxes directly from input images without the need for a separate region proposal network. In (b) the fundamental architecture of two-stage detectors comprises a region proposal network that generates region proposals, which are then fed into a classifier and regressor for further processing [80].

In this thesis, both Faster R-CNN, YOLO and CNN were utilized for the detection of the kidney stones and image classification in CT coronal scan images. Labels were specified as with kidney stones and without kidney stones.

6.4.1. Faster R-CNN

R-CNN, or Region-based Convolutional Neural Network, emerged as one of the earliest successful methods for object detection. It introduced a multi-step approach that combined selective search for generating region proposals and a convolutional neural network for classification. Although R-CNN showed promising results, its speed was a limiting factor. This prompted the development of Fast R-CNN, a significant improvement that addressed the efficiency issues. Fast R-CNN was developed as an enhancement to R-CNN. Instead of generating region proposals before the convolutional network, Fast R-CNN generates proposals by applying the selective search algorithm directly on the convolutional feature map. Both R-CNN and Fast R-CNN failed to overcome a particular challenge which was the generation of many invalid regions using methods such as selective search. This drawback not only hampers efficiency but also leads to wasteful utilization of computational resources [81]. The difference in the architectures of R-CNN and Fast R-CNN is shown in Figure 6.6 [82].

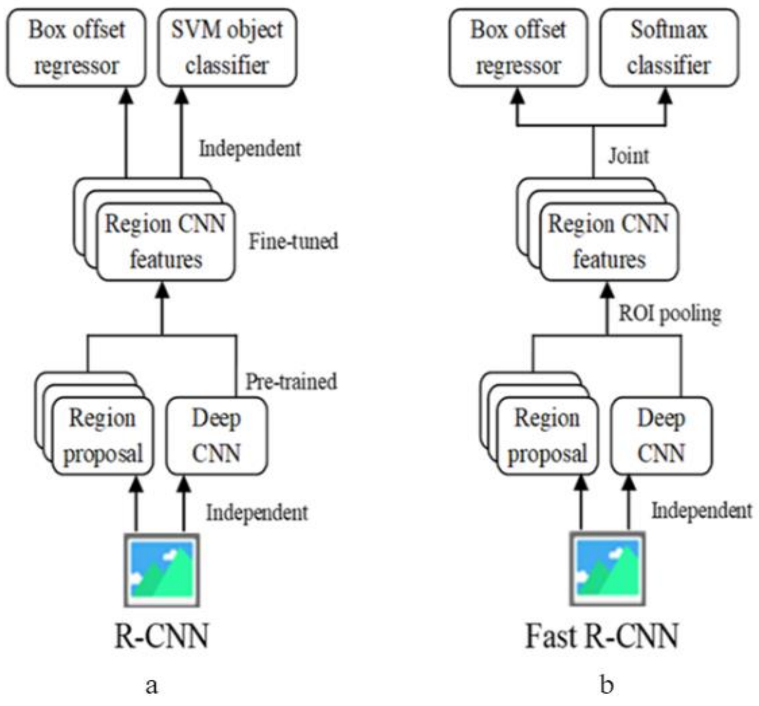


Figure 6.6. The difference in the architectures of R-CNN algorithms. (a) R-CNN (b) Fast R-CNN [82].

However, the pursuit of even greater speed and accuracy gave rise to Faster R-CNN, a revolutionary innovation in the field of object detection [81]. Faster R-CNN combined the strengths of its predecessors while introducing a novel element known as the Region Proposal Network (RPN). RPN replaces computationally expensive methods like selective search with the aim of significantly speeding up the object detection process. By sharing convolutional layers with the detection network, the RPN generated region proposals directly, eliminating the need for external region proposal algorithms. This architectural change significantly accelerated the object detection process and achieved state-of-the-art performance. Figure 6.7 illustrates the flowchart of the Faster R-CNN process [81].

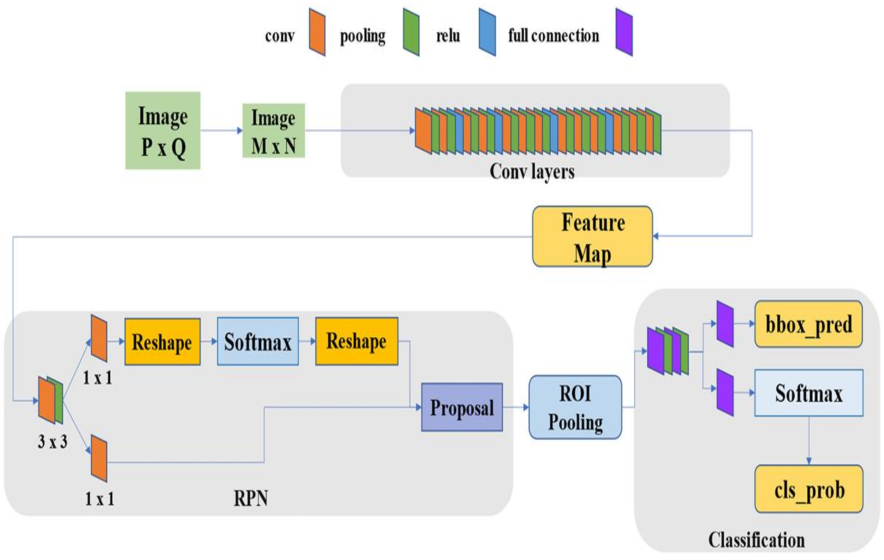


Figure 6.7. Flowchart of the Faster R-CNN [81].

According to the description provided by Ren et al. in their original paper, Faster R-CNN can be divided into four main components: Conv layers, RPN, Roi Pooling, and Classification. The process begins with resizing an input image P*Q of any size to a fixed size M*N, which is then fed into the Convolution layers to extract the feature map. This feature map is subsequently utilized by both the following RPN layer and the fully connected layer in a shared manner. The RPN utilizes SoftMax to determine the positivity or negativity of anchors and applies bounding box regression to refine the proposals for more accurate results. The ROI Pooling layer combines the feature maps and proposals, extracting proposal feature maps that are then passed to the fully connected layer for object category determination. The Classification component utilizes the proposal feature maps to calculate the category of the proposal and simultaneously obtains the precise position of the detection frame through bounding box regression [81].

6.4.1.1. Implementation of Faster R-CNN

In this thesis, the Faster-RCNN implementation was carried out by using TensorFlow object detection API. This API is a toolkit designed for object detection applications. It represents an advanced methodology for real-time object detection. TensorFlow Serving simplifies the deployment of novel methods, algorithms, and experiments while preserving the same server architecture and APIs [83]. The below flowchart in Figure 6.8 represents the steps of implementing Faster R-CNN.

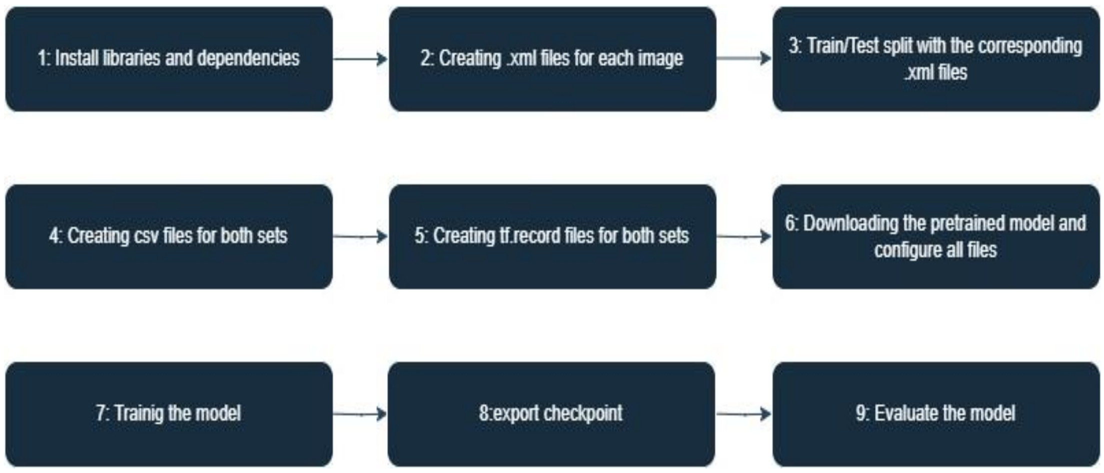

1: Install libraries and dependencies
2: Creating .xml files for each image
3: Train/Test split with the corresponding .xml files
4: Creating csv files for both sets
5: Creating tf.record files for both sets
6: Downloading the pretrained model and configure all files
7: Trainig the model
8:export checkpoint
9: Evaluate the model

Figure 6.8. Flowchart of implementing Faster-RCNN using TensorFlow object detection API.

The first step was to install and download all the required libraries and dependencies. The images in the dataset were labelled by using LabelImg. These annotated images are then used to train a model that can provide bounding box information. In order to train the model, both the image itself and the corresponding bounding box coordinates for all objects within the image are required. To store this annotation information, the program generates XML files in the Pascal Visual Object Classes format. These XML files contain various image details, such as the image name, size, and the coordinates of the bounding boxes. The dataset has been divided into two portions. Specifically, 80% of the total images have been allocated as training samples, while the remaining

20% of the images have been set aside as test samples. The training and testing samples are used to create two separate CSV files. A sample of the CSV files is shown in Table 6.2 where CSV files contain information such as the height and width of the images, as well as the coordinates of the bounding boxes and the class names associated with the objects in the images.

Table 6.2. A sample of CSV file.

filename	**width**	**height**	**class**	**xmin**	**ymin**	**xmax**	**ymax**
1.3.46.670589.33.png	1052	1266	Tas_Var	671	401	682	413
1.3.46.670589.33.png	956	1346	Tas_Var	363	596	368	602
1.3.46.670589.33.png	956	1346	Tas_Var	357	602	372	619
1.3.46.670589.33.png	956	1346	Tas_Var	359	635	365	641
1.3.46.670589.33.png	956	1346	Tas_Var	370	630	376	637
1.3.46.670589.33.png	956	1346	Tas_Var	364	632	371	640

Subsequently, TFRecord files are generated from these CSV files. TFRecord is a binary storage format used by TensorFlow. By using a binary format, the data occupies less disk space compared to other formats. Additionally, binary data can be copied more quickly and read efficiently from the disk, leading to improved performance when working with large datasets. The next step was to download and configure the pre-trained models. In this thesis, both Resnet-50 and Resnet-101 were utilized for the detection of kidney stones. In order to train the model effectively, a labelmap text file is created. This file plays a significant role in mapping the class names to their corresponding IDs. It serves as an input during the training process. In this thesis, there is one ID related to kidney stone class. checkpoints are generated during training. These checkpoints contain the weights and parameter values learned by the model during the training process. In this thesis, the training process utilized specific values for the batch size, number of steps, and learning rate. The batch size was set to 8, the learning rate, on the other hand, was set to 0.001. The number of steps was set to 8000, indicating the total number of iterations or updates performed during the training process. To monitor and analyze the results of the training and evaluation stages, TensorFlow provides a visualization platform called TensorBoard. This powerful tool allows to observe various metrics and statistics related to the training process. Many metrics are tracked such as training time, total loss, number of steps, and more [84]. Table 6.3 provides additional information on the implementation of Faster R-CNN.

Table 6.3. Additional information on the utilized hyperparameters.

Type	**Value**
No. of Images	Train: 625 Test: 165
No. of Steps	8000
Batch Size	8
Learning Rate	0.001

6.4.2. YOLO

The YOLO algorithm stands out as an object detection algorithm, utilizing a single neural network to estimate both bounding boxes and class probabilities for objects within an image. YOLO takes a grid-based approach, splitting the input image into cells (W × H) and predicting precise bounding box details along with class probabilities for each cell. Every prediction for a bounding box entails five values: Pc, bx, by, bh, and bw. Here, Pc denotes the confidence score, indicating the model's certainty about the presence and accuracy of the object within the box. The coordinates bx and by specify the box center relative to the grid cell, while bh and bw represent the box's height and width relative to the entire image. [85].

The fundamental structure of YOLO involves three key elements: the backbone, the neck, and the head. Notably, the architecture of the backbone, the neck, and the head can undergo variations in different YOLO versions. Continuous refinements and innovations within these components have been instrumental in driving significant improvements in both accuracy and speed of the YOLO network [85]. A sample of the architecture of YOLO is shown in Figure 6.9 [85].

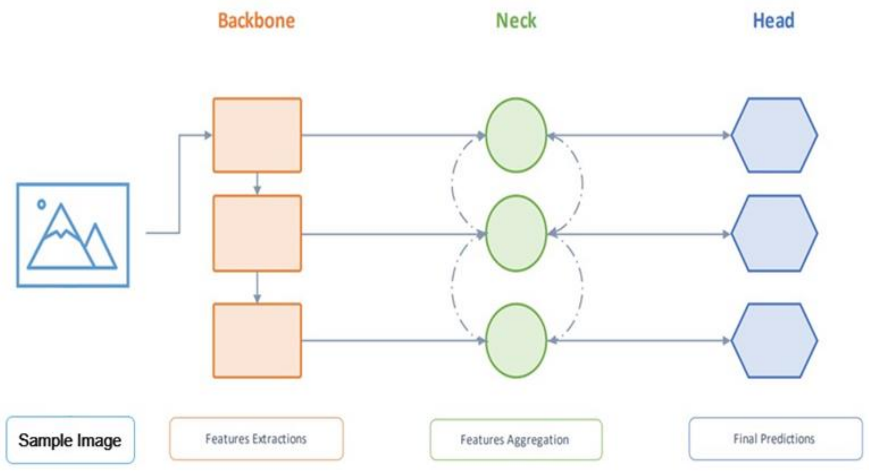


Figure 6.9. The architecture of YOLO [85].

The primary function of the Backbone is to extract essential features from the input image. Where the neck plays a crucial role in combining feature maps from diverse layers of the backbone network, forwarding them seamlessly to the. Finally, the head module takes charge of processing the combined features, predicting bounding boxes, objectness scores, and classification scores [86]. Table 6.4 shows the differences among YOLO versions.

Table 6.4. Primary features of YOLO versions [85].

Version	**Date**	**Framework**	**Backbone**
YOLO	2015	Darknet	Darknet24
YOLOv2	2016	Darknet	Darknet24
YOLOv3	2018	Darknet	Darknet53
YOLOv4	2020	Darknet	CSPDarknet53
YOLOv5	2020	Pytorch	Modified CSP v7
PP-YOLO	2020	PaddlePaddle	ResNet50-vd

Scaled-YOLOv4	2021	Pytorch	CSPDarknet
PP-YOLOv2	2021	PaddlePaddle	ResNet101-vd
YOLOR	2021	Pytorch	CSPDarknet
YOLOX	2021	Pytorch	Modified CSP v5
PP-YOLOE	2022	PaddlePaddle	CSPRepResNet
YOLOv6	2022	Pytorch	EfficientRep
YOLOv7	2022	Pytorch	RepConvN
DAMO-YOLO	2022	Pytorch	MAE-NAS
YOLOv8	2023	Pytorch	YOLO v8
YOLO-NAS	2023	Pytorch	YOLO-NAS

The YOLO output takes the form of a tensor with dimensions W × H × (B × 5 + C), where B represents the number of bounding boxes, and C represents the number of classes. This output is subject to non-maximum suppression (NMS) to remove redundant detections. The grid cells play a crucial role in handling operations related to bounding box estimation and class probabilities. Each grid cell in the model predicts bounding boxes and confidence scores for those boxes. These confidence scores indicate the model’s level of certainty that the box contains an object and convey its assessment of the accuracy of the predicted box as shown in Eq. 6.1. In essence, YOLO calculates the probability of the detection element’s bounding box center residing within the grid cell, as expressed by Eq. 6.2 [86,87].

$$Confidence(P) = Probability(p) \times IOU(prediction, target) \tag{6.1}$$

$$IoU = \left|\frac{B \cap B^{gt}}{B \cup B^{gt}}\right| \tag{6.2}$$

Within the YOLO algorithm framework, the target box is labeled as B^{gt} , and the predicted box is denoted as B. The probability (p) indicates the likelihood of an object’s presence within the identified bounding box. The Intersection over Union (IoU) metric, articulated by Eq. 6.2, computes the shared area between the ground truth and predicted bounding boxes. It establishes a threshold for an acceptable region for each identified object in the input image, influencing decision-making processes. After the estimation, the confidence value is then applied to determine the most suitable bounding box [87,88]. The process of IoU is shown in Figure 6.10.

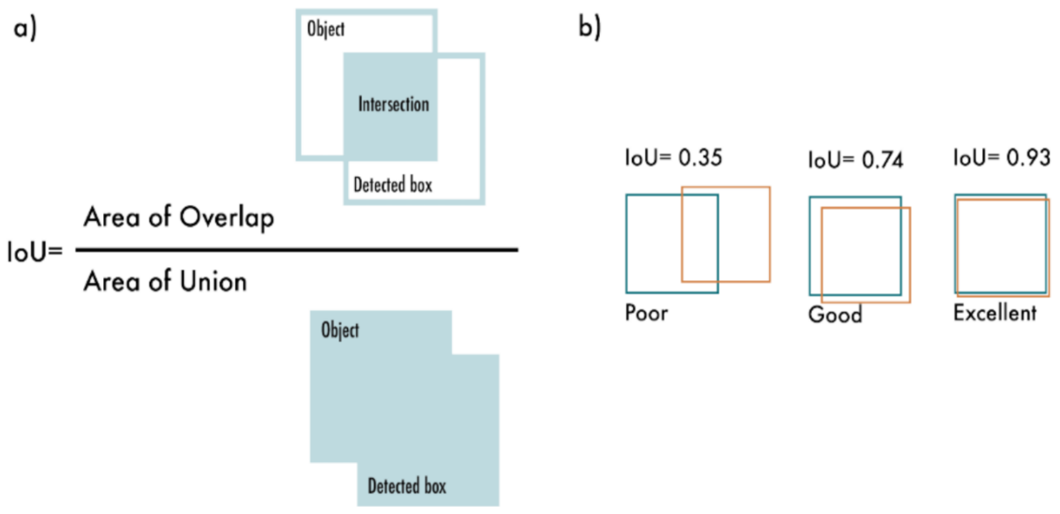


Figure 6.10. a) The process of calculating IoU [89]. b) Examples of different IoU values.

In this thesis, both YOLO v5 and YOLO v7 were utilized for the detection of the kidney stones.

6.4.2.1. YOLO v5

In June 2020, Ultralytics LLC introduced the YOLOv5 algorithm, presenting a more compact and convenient alternative compared to YOLOv2, YOLOv3, and YOLOv4. YOLOv5's smaller size facilitates flexible deployment and enhances detection accuracy [80]. YOLOv5 introduces five scaled versions: YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large). These variants are tailored with varying widths and depths in their convolution modules to fit specific applications and adhere to diverse hardware requirements [89]. Table 6.5 shows the differences between the versions of yolo in respect to the mean average precision (mAP) of the models based on COCO val2017 dataset, the inference speed on CPU and GPU, and the number of the parameters of each model [90].

Table 6.5. Overview of all the models, including the mAP, inference speed on CPU, GPU, and the number of parameters.

Model	**mAP^{val} 50-95**	**GPU Speed V100 b1 (ms)**	**CPU speed b1 (ms)**	**Params (M)**
YOLO v5n	28.0	6.3	45	1.9
YOLO v5s	37.4	6.4	98	7.2
YOLO v5m	45.4	8.2	224	21.2
YOLO v5l	49.0	10.1	430	46.5
YOLO v5x	50.7	12.1	766	86.7

Where $\mathrm{mAP}^{\mathrm{val}}$ stands for mean average precision over different IoU thresholds, from 0.5 to 0.95, step 0.05 in COCO val2017 dataset, V100 b1 is an inference time which is the speed at which a neural network can make predictions in milliseconds on the Nvidia V100 GPU where the batch size is set to 1. Params stand for the number of parameters in the model. Which is calculated by summing the number of elements in each layer of the model. The M stands for Million. The architecture of YOLO v5 is shown in Figure 6.11 [91].

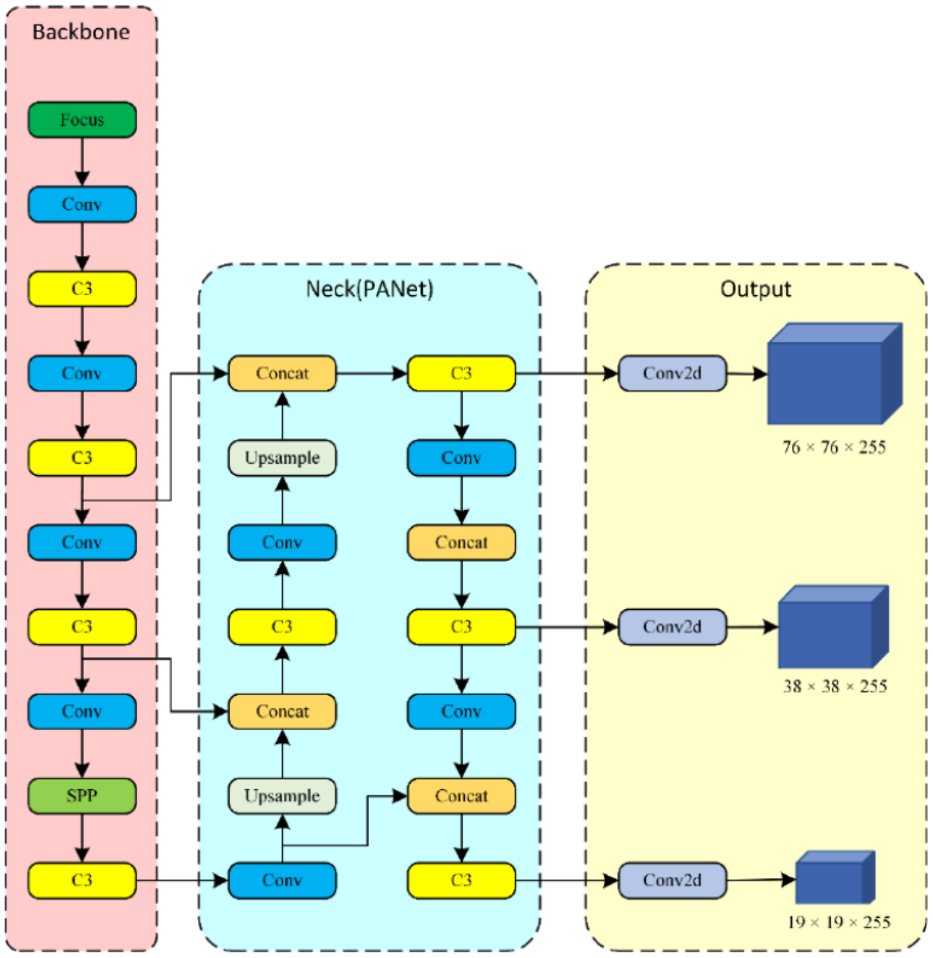


Figure 6.11. YOLO v5 architecture [91].

As Illustrated in Figure 6.11, the YOLOv5 network is composed of three primary components: the backbone, neck, and output. In the YOLOv5 architecture, the backbone network is primarily comprised of the focus module, the wrapped convolution module, the C3 module, and the spatial pyramid pooling (SPP) module. The focus module improves the network's ability to extract features from images by performing slicing operations on the input images. The C3 module, on the other hand, enhances the feature representation capabilities of the network while preserving the accuracy of feature extraction. This is achieved by reducing the memory consumption and number of parameters required by the network. SPP is a pooling layer that eliminates the network's fixed size restriction, allowing it to effectively process inputs of varying sizes. The neck network is designed to optimize the utilization of features extracted from the backbone network. predicting confidence, class probability, and object box coordinates across three diverse feature maps are carried out using three 1×1 convolutional layers as shown in the output layer [91,92].

6.4.2.1.1 Implementation of YOLO v5

In this thesis, the YOLO implementation was carried out by using Google colaboratory. The first step was to setup the YOLO environment by cloning the repository of YOLO in GitHub. After the cloning was done, folders such as data, model, and hyperparameters configuration files were created. Both images and the labels were uploaded into data folders to initiate the training process. The labels are in .TXT extension which contains number of classes, the center position of x and y, the height, and the width. In addition to that, the hyperparameters can be determined and customized as needed. In this thesis, the value of class is zero since there is only one class. Out of total 790 images in the dataset 80% of them were divided as train ,10% as validation and 10% as test. The used dataset and the hyperparameters are shown in Figure 6.12.

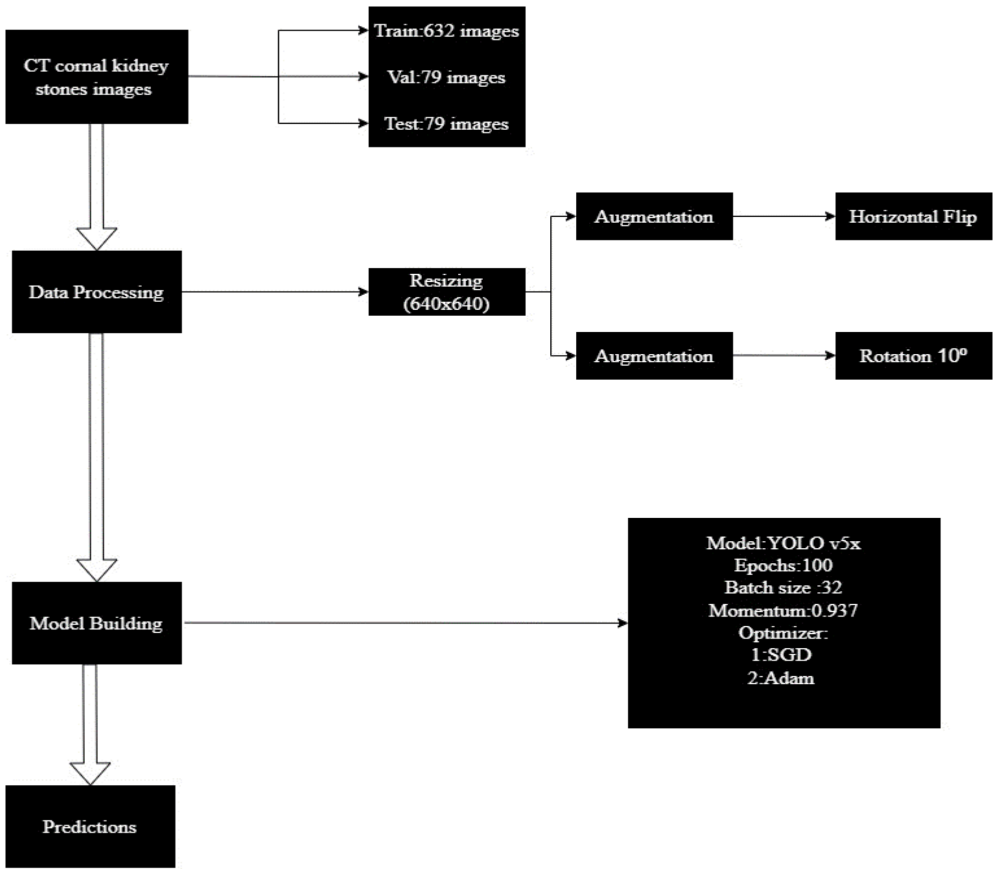


Figure 6.12. Schematic of implementation of YOLO v5.

In this thesis, four datasets were used to perform the detection process of kidney stones. A brief description of the utilized datasets is shown in Figure 6.13.

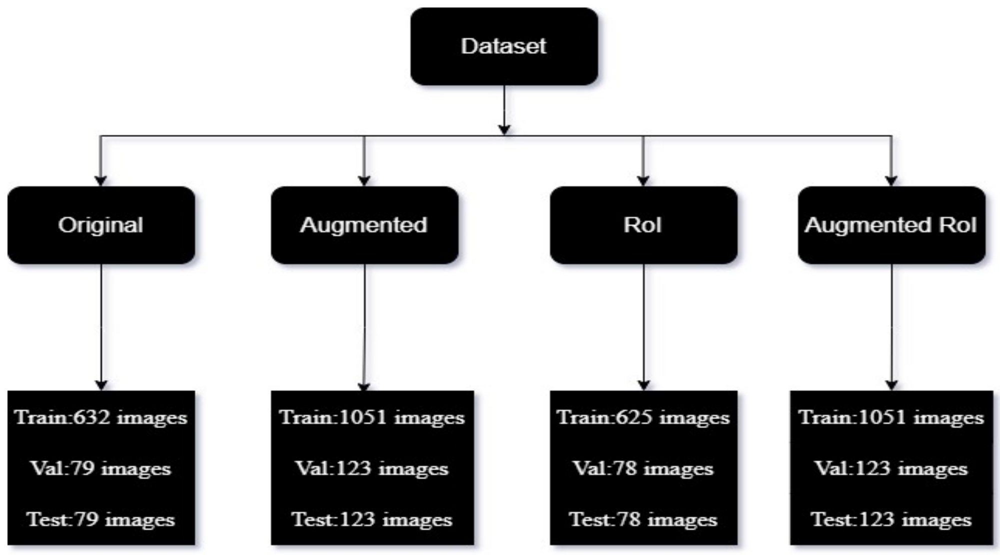


Figure 6.13. The utilized datasets in YOLO v5 and YOLO v7.

To detect kidney stones using the YOLO algorithm, four datasets were used. Original Dataset (D1) which was the main set of original images. Augmented Dataset (D2) was the second dataset which had additional images created with variations. The third dataset was RoI Annotated Dataset (D3) which focused on specific areas relevant to kidney stones, marked for the model’s attention. The last one was combined augmented RoI Dataset (D4) which contained a mix of augmented images within the identified areas of interest.

For this thesis, Adam and SGD optimizers were used. Stochastic Gradient Descent (SGD) stands out as a prevalent optimizer in deep learning, primarily employed to minimize the cost function. SGD algorithms are widely utilized when dealing with massive datasets. SGD is an optimization algorithm commonly employed in linear regression, as illustrated in Eq. 6.3., Eq. 6.4, and Eq. 6.5 [93].

$$W = \omega - \eta \nabla Qi(\omega) \tag{6.3}$$

$$W \leftarrow \eta \nabla Q(\omega) \tag{6.4}$$

$$Q(\omega) = \boldsymbol{ln}\sum iQi(\omega) \Rightarrow \nabla\, Q(\omega) = \boldsymbol{ln}\sum i\, \nabla\, Qi(\omega) \tag{6.5}$$

in Eq. 6.3, ω represents the initial weight during the training of a neural network. η denotes the learning rate. Qi signifies the presently observed data, where Q generally represents an error function. SGD determines the optimal weight W by minimizing the error function Q. Subsequently, the derivative of the actual and predicted values is calculated to obtain the loss function. W undergoes updates during the neural network training process. Eq. 6.4. represents the parameter update in the context of SGD. The final expression in Eq. 6.5. is the objective of minimizing the error along with its gradient [93].

Adam optimization, introduced by Diederik P. Kingma and Jimmy Ba in 2014, is a gradient descent-based algorithm. The name “Adam” stands for Adaptive Moment Estimation, reflecting the optimizer’s approach to weight updates during training. The mathematical representation of the Adam optimization algorithm is shown in Eq. 6.6 [93].

$$\triangle\, \omega_t = -\eta \frac{x_t}{\sqrt{y_t + \epsilon}} * g_t \tag{6.6}$$

$$x_t = \delta_1 * x_{t-1} - (1 - \delta 1) * g_t \tag{6.7}$$

$$y_t = \delta_2 * y_{t-1} - (1 - \delta 2) * g_t^2 \tag{6.8}$$

$$\omega_{t+1} = \omega_t + \triangle\, \omega_t \tag{6.9}$$

In Eq. 6.6., η represents the learning rate, and g_t is the gradient at time t. x_t signifies the exponential average of gradients along the parameter w_j, while y_t represents the exponential average of the squares of gradients along w_j. The decay rates of moment estimates are controlled by the hyperparameters $\delta_1$and δ_2 [93]. ϵ represents a fixed value to ensure numerical stability [94]. Eq. 6.7 explains the process for updating the first moment (mean) of the gradient, x_t. This involves a weighted combination of the previous first moment x_{t-1} and the current gradient , g_t with the rate of decay determined by the hyperparameter δ_1. Simultaneously, Eq. 6.8 articulates the update mechanism for the second moment (uncentered variance) of the gradient, y_t. It incorporates the previous second moment y_{t-1} and the squared gradient, g_t^2 with the decay rate modulated by the hyperparameter δ_2. Eq. 6.9 depicts the adjustment of the internal parameter, ω, within the optimizer. This adjustment entails adding the change, represented as $\Delta\omega_t$, into the current value of ω_t [95].

The default value of the learning rate of SGD and ADAM is 0.01 and 0.001 respectively [94]. The values of epochs, batch size and momentum were set as 100, 32, and 0.937, respectively. After the training process was carried out the validation and test phases were done to validate the model's performance. The last step was to show the detected kidney stones alongside with the confidence value, bounding box, and class label.

6.4.2.2. YOLO v7

Released as the successor to YOLOv6, YOLOv7 is a recent model that significantly elevates object detection performance. Notably, YOLOv7 achieves enhanced accuracy without imposing additional computational and inference costs. Wang et al. introduced the YOLOv7 version in July 2022. The developers of YOLOv7 aimed to establish a state-of-the-art standard in object detection by devising a network architecture capable of more accurate bounding box predictions at comparable inference rates to its old YOLO versions [96]. The novel design of this network architecture is illustrated in Figure 6.14 [96].

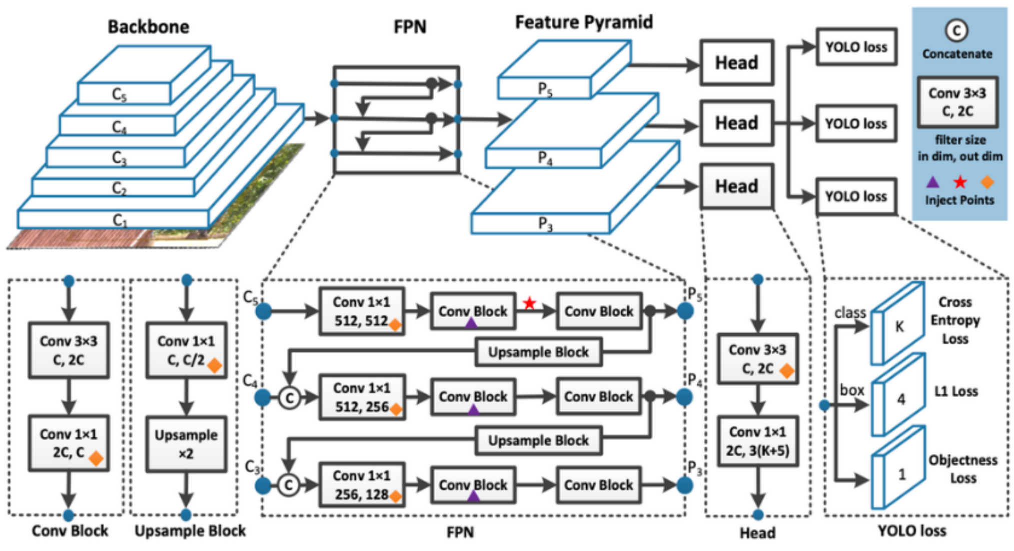


Figure 6.14. Architecture of YOLO v7 [96].

The computational block within the YOLOv7 backbone is denoted as Extended Efficient Layer Aggregation Network (E-ELAN). The YOLOv7 Neck module plays a crucial role by executing feature fusion on the previously generated effective feature layers. The YOLOv7 Head, encompassing essential elements for both classification and regression, serves as the pivotal component in charge of these tasks [97].

6.4.3. CNN

CNN is a versatile deep learning model engineered to handle diverse data types, including 1D for signals or sequences, 2D for images or sound spectrograms, and 3D for video or volumetric images. CNN accepts images for the input layer. It can consist of one or more layers, with matrix multiplication or convolution applied in at least one layer. CNN architectures comprise layers with distinct functions and characteristics, such as convolution layers, activation layers, pooling layers, flatten layers, and fully connected layers. The output layer represents the final stage in the neural network, where the model produces its final predictions based on the acquired knowledge and training. [98]. A general CNN representation structure is given in Figure 6.15 [99].

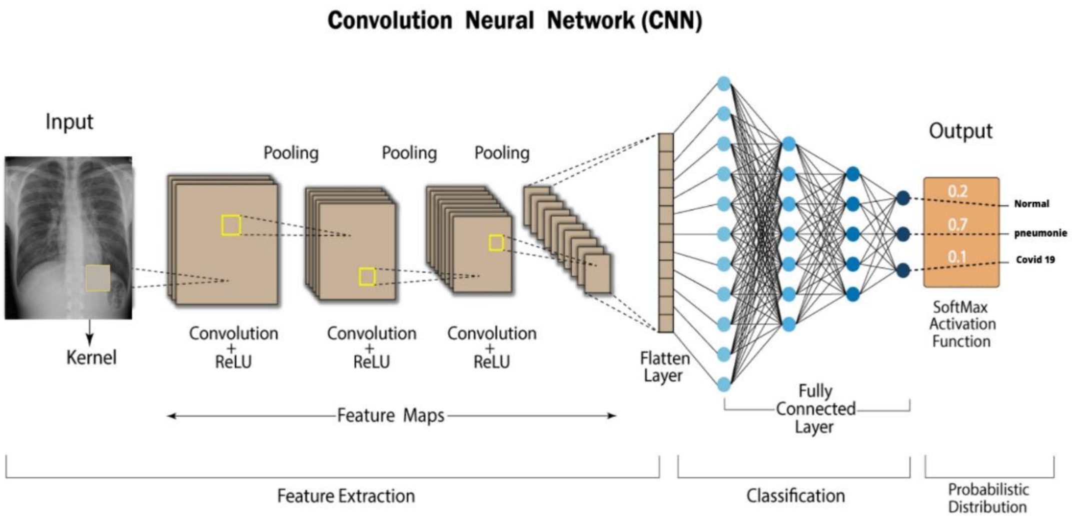


Figure 6.15. CNN general structure [99].

Convolution Layer: This layer is very important for the CNN structure. This layer allows the detection of features in the image. It contains low or high frequency features in the image data. To detect these features, a sample matrix called a filter or kernel is applied on the image. The dimensions of the kernel matrix are generally values such as 3x3,5x5, 7x7.The kernel matrix starts from the upper left corner of the image and scrolls throughout the lower right corner. As the kernel matrix moves along the image, the values of the image and filter matrices undergo multiplication according to their respective indices, and the products are summed. The total result is then recorded in an output matrix. This process is continued in the same way throughout the entire image. The matrix formed as a result of the values recorded in the output matrix is called Feature Map. As a result of the kernels applied on the image, there will be changes in the original dimensions of the image. Even if a filter is applied, pixel padding can be done to avoid losing important information on the image. For this purpose, the process performed to ensure that the dimensions of the input image and the output image are the same is the padding process. The logic of padding is to increase the size by adding zeros around the image [100]. The process of convolution and padding is shown in Figure 6.16 [101].

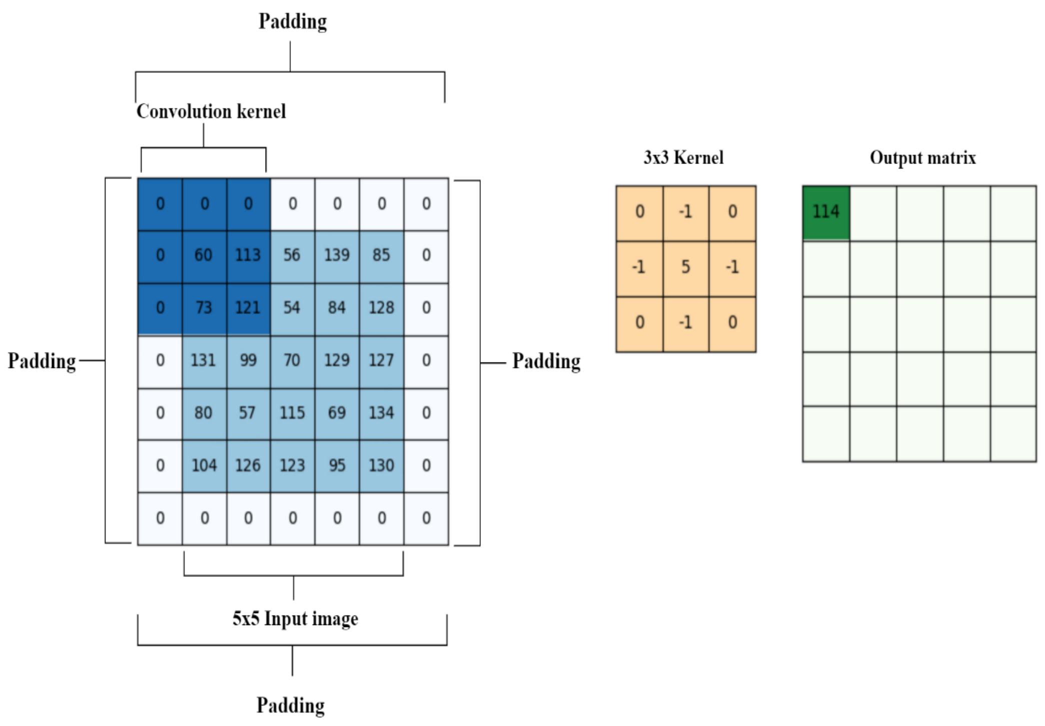


Figure 6.16. Convolution and padding process in CNN [101].

Activation Layer: This layer can also be called the non-linear layer. sigmoid, tanh, SoftMax and Relu are frequently used activation functions in this layer. Activation functions determine what action a neuron should apply to the incoming input and thus create the output. In general, the Relu activation function is used in the layers of CNN models. Additionally, in the context of binary classification tasks, the final layer often utilizes the sigmoid function, whereas for multi-class classification, the SoftMax function is commonly applied to the last layer [100,102].

Pooling Layer: The pooling layer operates similarly to the convolution layer and reduces the burden of data calculation by reducing image dimensions. There are two types of pooling: Max Pooling and Average Pooling. In Max Pooling, the highest value within the kernel region is selected and stored in the output matrix. Average Pooling, on the other hand, takes the average of the part covered by the filter/kernel and stores it in the output matrix. Both pooling methods iteratively apply this process across the image, generating the output matrix [100,102]. Although Average Pooling method was used more widely in the past years, Max Pooling has been used more widely recently thanks to its noise reduction ability [103]. Figure 6.17 shows Max and Average pooling operations. Max Pooling method was used in this thesis.

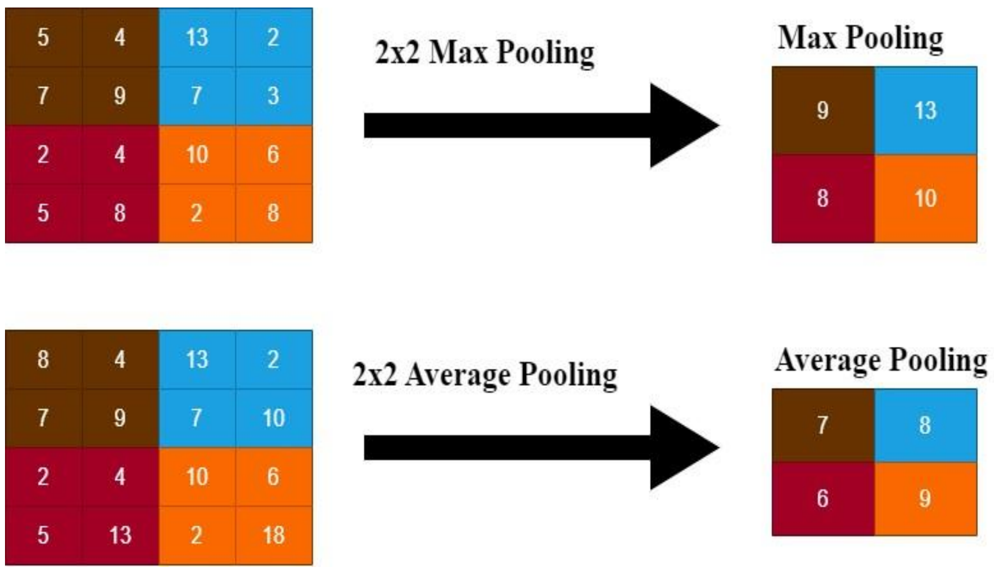


Figure 6.17. Max and Average pooling operations.

Flatten Layer: Within CNN models, the flatten layer serves the purpose of transforming the output from the preceding convolution layer, which consists of two-dimensional feature maps, into a one-dimensional array. This step is essential to facilitate the transition to the fully connected layer, as fully connected layers operate exclusively on one-dimensional arrays [104].

Fully connected layer: It is used at the end of the network after feature extraction is performed by the convolution and pooling layers. In this layer, each neuron is connected to all neurons of the previous layer. It is used by the network to make predictions [104].

6.4.3.1 Optimization of Hyperparameters

Determining hyperparameters and appropriately preprocessing data are crucial aspects in training deep learning models. These parameters have a direct impact on the model’s performance and contribute significantly to achieving generalized results.

To enhance model performance, various combinations were explored, including (16-16-32-32-64-64), (32-32-64-64-256-256), (32-64-128-256), (128-128-256-512), and (16-32-32-64) for the number of convolution layers and neurons. Additionally, different neuron sizes as 16, 32, 64, 128, and 256 were experimented with for the fully connected layer. The dropout rates of 0.2, 0.3, 0.4, and 0.5 were tested to further fine-tune the model. In addition, Batch normalization layer also was implemented as part of the tests carried out.

6.4.3.2 Implementation of CNN

In this thesis, the CNN implementation was carried out by using Google colaboratory. The first step was to setup the CNN environment by importing the necessary files such as Keras and TensorFlow packages. The next step was to upload the dataset files into the Google Colaboratory environment where there are two classes in the dataset named as Normal and Kidney stone. Later the image pre-processing step was initiated by resizing the images to 300x300. Out of total 1799 images in the dataset 80% of them were divided as train and 20% as test. Further information about the used dataset and the hyperparameters are shown in Figure 6.18.

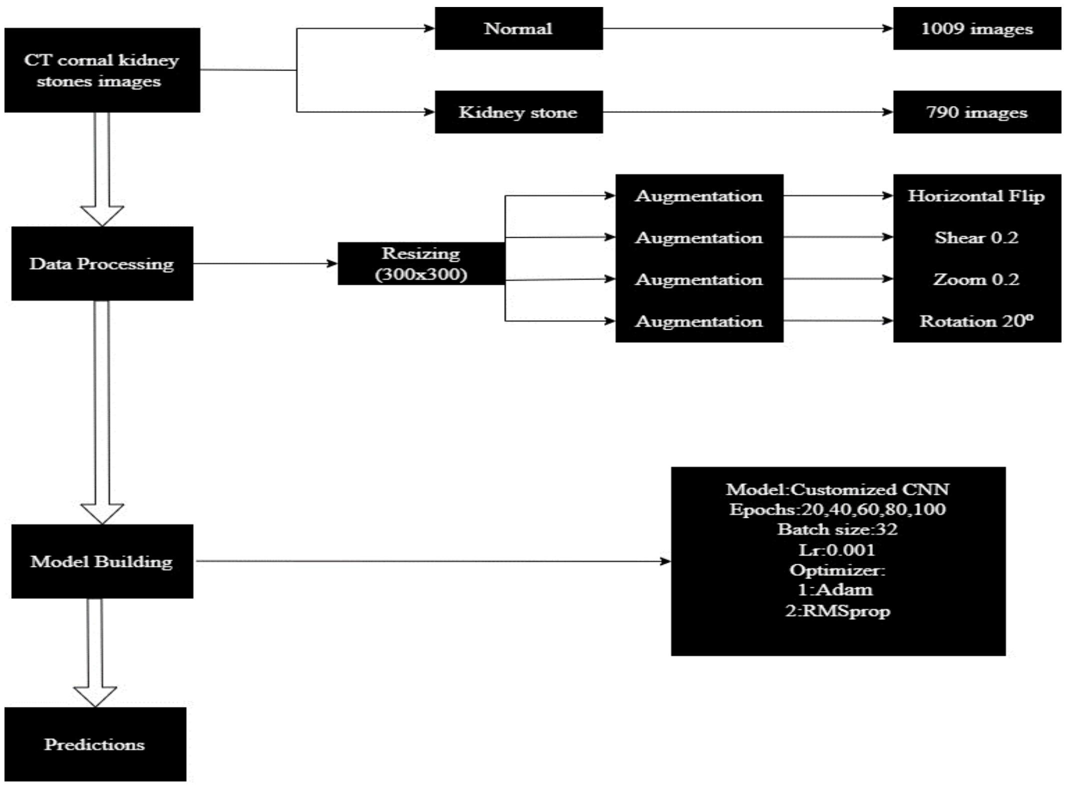


Figure 6.18. Schematic of implementation of CNN.

next step was to build the CNN model by using Convolution, Max Pooling, Flatten and Fully connected layers. A total of 23 layers were utilized. The architecture of the utilized CNN is shown in Figure 6.19.

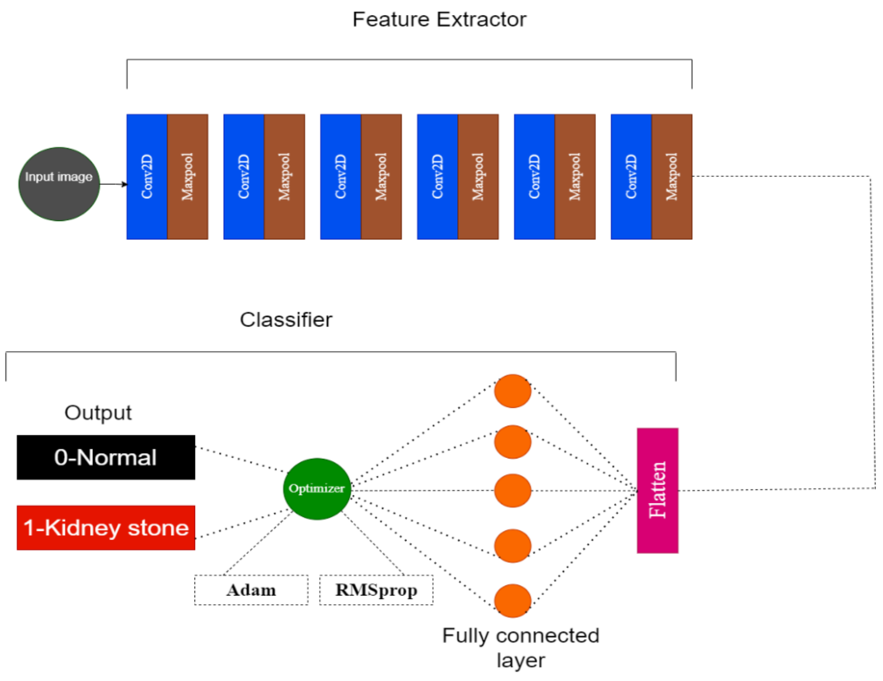


Figure 6.19. Architecture of the CNN model.

Afterward, data augmentation techniques such as horizontal flip, shear, zoom, and rotation were applied to the dataset. Following that, to initiate the training phase the batch size and the learning rate were set as 32, 0.001, respectively.

In this thesis, different epochs such as 20, 40, 60, 80, and 100 were utilized. In addition to that, Adam and RMSprop optimizers were selected to perform the training of the CNN model. Binary cross-entropy loss function was selected. Sigmoid was chosen as the activation function in order to perform binary classification in the output layer. It is expressed in Eq. 6.10.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{6.10}$$

Here, x represents the input value to the sigmoid function. The sigmoid function produces output values that fall within the range of 0 to 1 [105].

Following that, the test set were utilized to validate the model’s performance. Table 6.6 shows information on the utilized CNN architecture for the feature extraction process. Table 6.7 shows information about the Flatten and Fully connected layers utilized in CNN architecture.

Table 6.6. Additional information on the architecture of CNN for the feature extraction process.

Block	**Type**	**Kernel size**	**Filters**	**Stride**	**Padding**
1	Conv2D	3x3	64	1	same
	Relu	-	-	-	-
	MaxPooling2D	2x2	-	2	-
2	Conv2D	3x3	64	1	same
	Relu	-	-	-	-

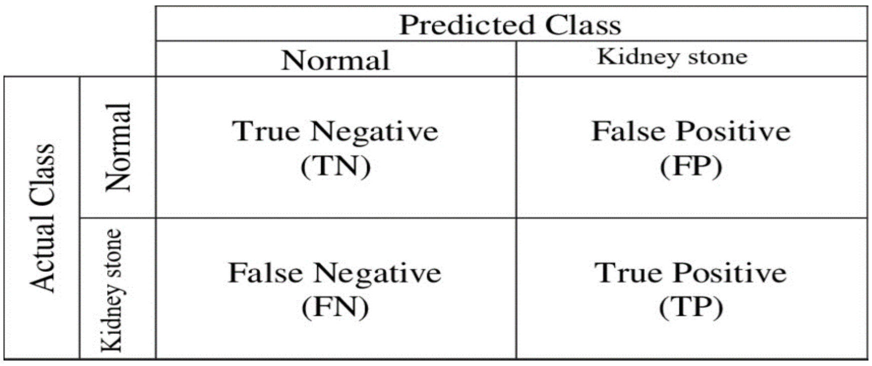
	MaxPooling2D	2x2	-	2	-
3	Conv2D	3x3	128	1	same
	Relu	-	-	-	-
	MaxPooling2D	2x2	-	2	-
4	Conv2D	3x3	128	1	same
	Relu	-	-	-	-
	MaxPooling2D	2x2	-	2	-
5	Conv2D	3x3	512	1	same
	Relu	-	-	-	-
	MaxPooling2D	2x2	-	2	-
6	Conv2D	3x3	512	1	same
	Relu	-	-	-	-
	MaxPooling2D	2x2	-	2	-

Table 6.7. Architecture of CNN in classifier process.

Layer	**Type**	**No. of Neurons**
1	Flatten	-
2	Dense	128
	Relu	-
3	Output	1

6.5. EVALUATION METRICS

To assess classification models, multiple performance evaluation metrics are employed, and these metrics rely on values within the confusion matrix, also known as the error matrix. This matrix is instrumental in visualizing the performance of algorithms or models through a tabular representation. It encompasses four key parameters: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Here, TP refers to the number of images classified as kidney stones and actually identified as kidney stones, while TN refers to the number of images classified as Normal and actually identified as Normal. On the other hand, FP refers to the number of images classified as kidney stones, but actually identified as Normal, while FN refers to the number of images classified as Normal, but actually identified as kidney stones. The components of confusion matrix are shown in Figure 6.20 [106]. A practical example illustrating the process of TP, FP, FN, and FN is given in Figure 6.21.



		Predicted Class	
		Normal	Kidney stone
Actual Class	Normal	True Negative (TN)	False Positive (FP)
	Kidney stone	False Negative (FN)	True Positive (TP)

Figure 6.20. Confusion matrix [106].

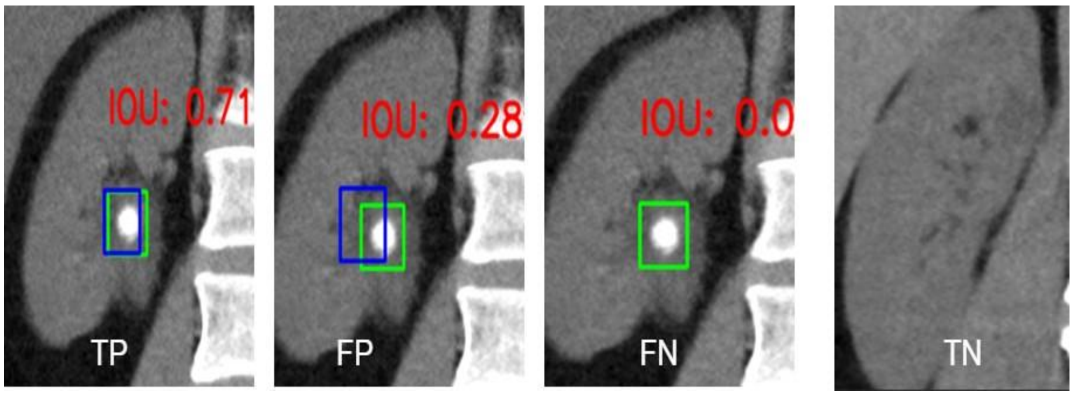


Figure 6.21. An example showing the process of TP, FP, FN, and TN in object detection.

The provided example illustrates predictions with the IoU value set at α=0.5. The initial prediction is classified as a TP, given that the IoU value is higher than 0.5. The second prediction in the example is classified as a FP because it does not meet the IoU value criteria. The third prediction is FN since the model failed to predict ground truth bounding box. The last prediction is TN due to the absence of kidney stone.

Sensitivity (Recall): It measures the proportion of true positive samples relative to the total number of positive samples (both TP and FN), as expressed by Eq. 6.11 [107].

$$Sensitivity = \frac{TP}{TP + FN} \tag{6.11}$$

Specificity: It serves as a performance metric used to assess a model's ability in accurately classifying negative examples, as expressed by Eq. 6.12 [107].

$$Specificity = \frac{TN}{TN + FP} \tag{6.12}$$

Precision: It measures the proportion of true positive samples among the total number of samples classified as positive. (both TP and FP), as expressed by Eq. 6.13 [107].

$$Precision = \frac{TP}{TP + FP} \tag{6.13}$$

Accuracy: It evaluates the ratio of correctly classified samples to the total number of samples and stands as the most utilized performance measure for classification models, as expressed by Eq. 6.14 [107].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.14}$$

F1 Score: This metric used in performance evaluation at the end of the classification It is a metric that offers insights into the accuracy of a test, obtained through the calculation of the harmonic mean between sensitivity and precision. The F1 score equation is shown in Eq. 6.15 [107].

$$F1 = \frac{2 * (Sensitivity * Precision)}{Sensitivity + Precision} \tag{6.15}$$

Average Precision (AP): A metric frequently used alongside with precision and sensitivity, to offer a comprehensive evaluation of a model's performance. It enhances our understanding by averaging precision across various sensitivity levels. Furthermore, mAP is another metric used to evaluate object detection models. Similar to average precision, mAP involves averaging precision, but instead of considering precision at different sensitivity levels, it averages precision across various confidence score thresholds. To calculate mAP, precision and sensitivity values are assessed at different confidence value thresholds. A graph depicting the precision-sensitivity curve is generated, and the area under this curve is determined. The mAP score is then obtained by averaging the area under the curve across different object classes. A higher mAP value indicates a more effective detection performance of the target detection model on a given dataset. The equation of mAP is given in Eq. 6.16 [108-110].

$$mAP = \frac{1}{N}\sum_{n=1}^{N} APi \tag{6.16}$$

In this context, where APi represents the AP of the i^{th} class and N is the total number of classes. Since there is only kidney stone class in our thesis, the number of classes was taken as 1.

CHAPTER 7

RESULTS & DISCUSSION

In this part of the thesis, the analysis of the performance evaluation metrics obtained by Faster R-CNN, YOLO, and CNN models is explained in detail. The models were first compared with each other in various combinations and then evaluated by comparing them with state-of-the-art studies that have achieved significant results.

7.1. Faster R-CNN

To address the objective of kidney stone detection, three models of Faster R-CNN namely Faster R-CNN ResNet50 V1 800x1333, Faster R-CNN ResNet101 V1 800x1333, and Faster R-CNN ResNet101 V1 1024x1024 were utilized.

7.1.1. Results of ResNet50 V1 800x1333 model

The results obtained by the training process of Faster R-CNN ResNet50 V1 800x1333 model such as total loss, classification loss, localization loss, RPN's localization loss, and RPN's objectness loss are shown in Figures 7.1-7.3

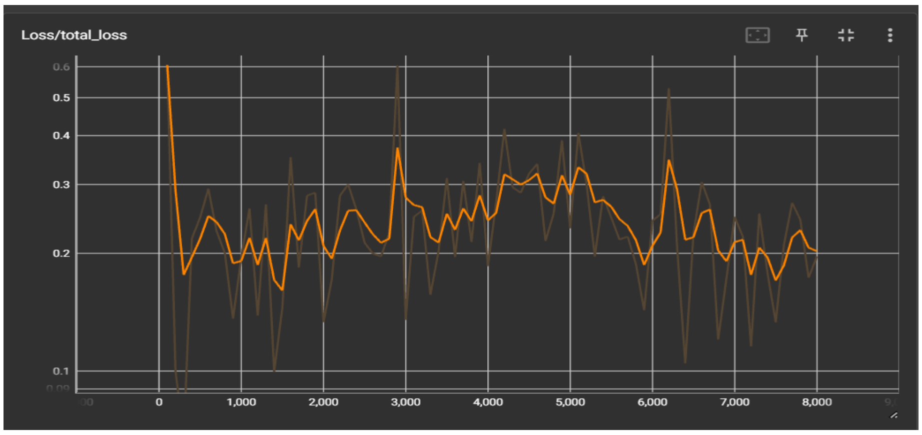


Figure 7.1. Total loss graph of the ResNet50 V1 800x1333 model.

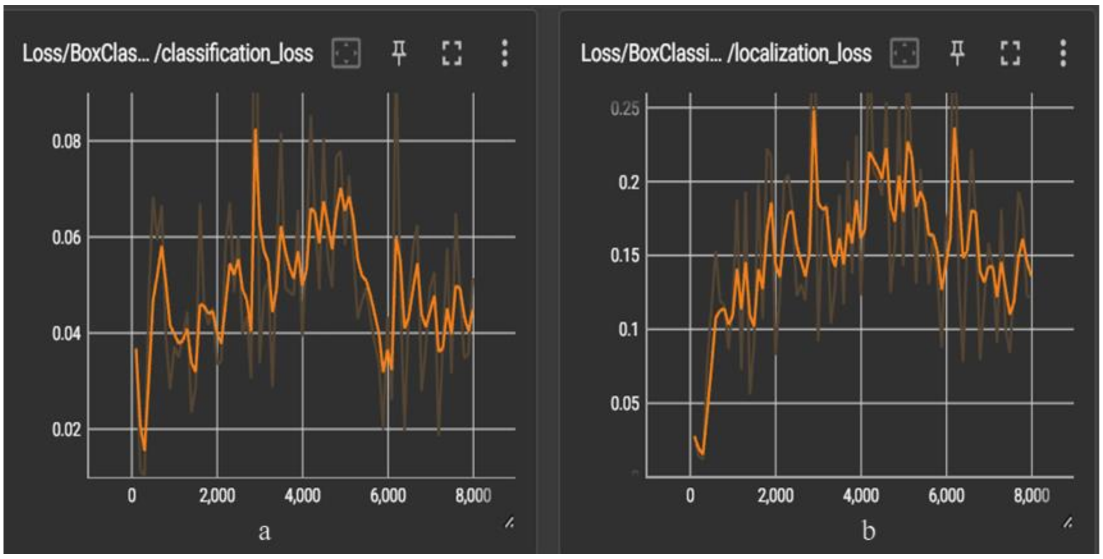


Figure 7.2. Loss graphs of the model. (a) Classification (b) localization.

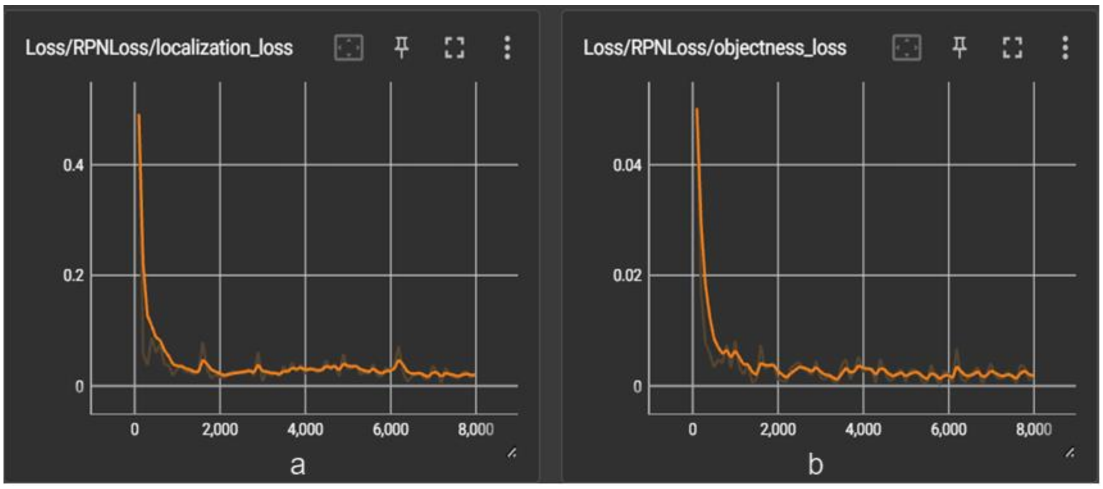


Figure 7.3. Loss graphs of RPN. (a) Localization (b) objectness.

As mentioned earlier, AP stands out as a crucial metric for assessing the accuracy of the object detection model. The obtained results of AP in the test phase are shown in Table 7.1.

Table 7.1. The obtained results based on AP.

Model	**AP (0.5:0.95) (%)**	**AP (0.50) (%)**	**AP (0.75) (%)**	**Total loss**
ResNet50 V1 800x1333	24.8	62.9	14.0	0.164

The Average Sensitivity (AS) is similar to AP, but instead it measures the sensitivity. The obtained results of AS in the test phase are shown in Table 7.2.

Table 7.2. The obtained results based on AS.

Model	**AS (1) (%)**	**AS (10) (%)**	**AS (100) (%)**
ResNet50 V1 800x1333	20.5	34.2	35.4

AS (1) denotes a scenario where the model provides one detection per image. AS (10) signifies the model offering ten detections per image. AS (100) indicates the model producing one hundred detections per image.

The learning rate is a fundamental hyperparameter that affects the model's performance. In this thesis, the learning rate value was set as 0.001 and the number of steps was set as 8000 . Figure 7.4 shows the change of learning rate during the training process.

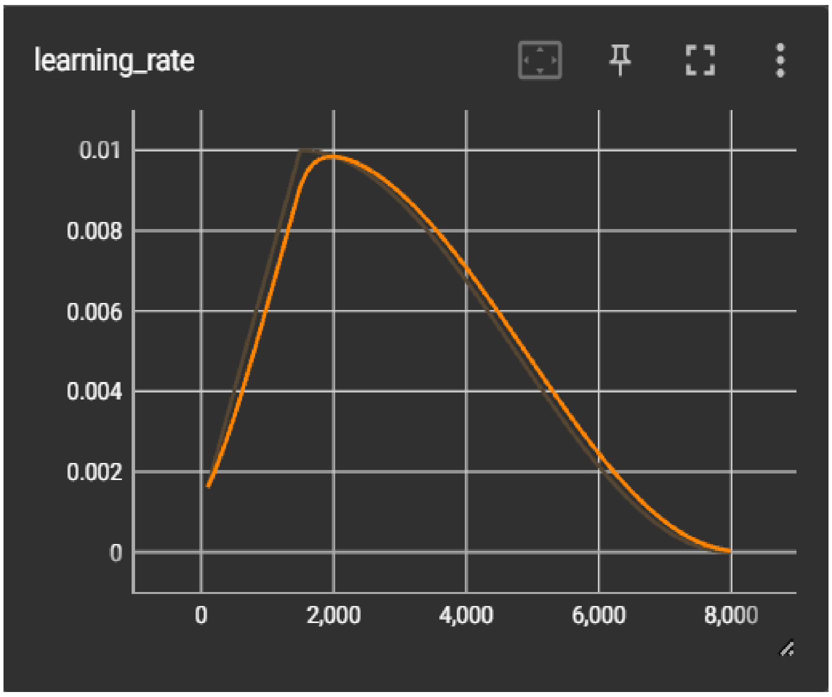


Figure 7.4. The change of learning rate during the training process.

7.1.2. Results of ResNet101 V1 800x1033 model

The results obtained by the training process of Faster R-CNN ResNet101 V1 800x1333 model such as total loss, classification loss, localization loss, RPN's localization loss, and RPN's objectness loss are shown in Figures 7.5-7.7.

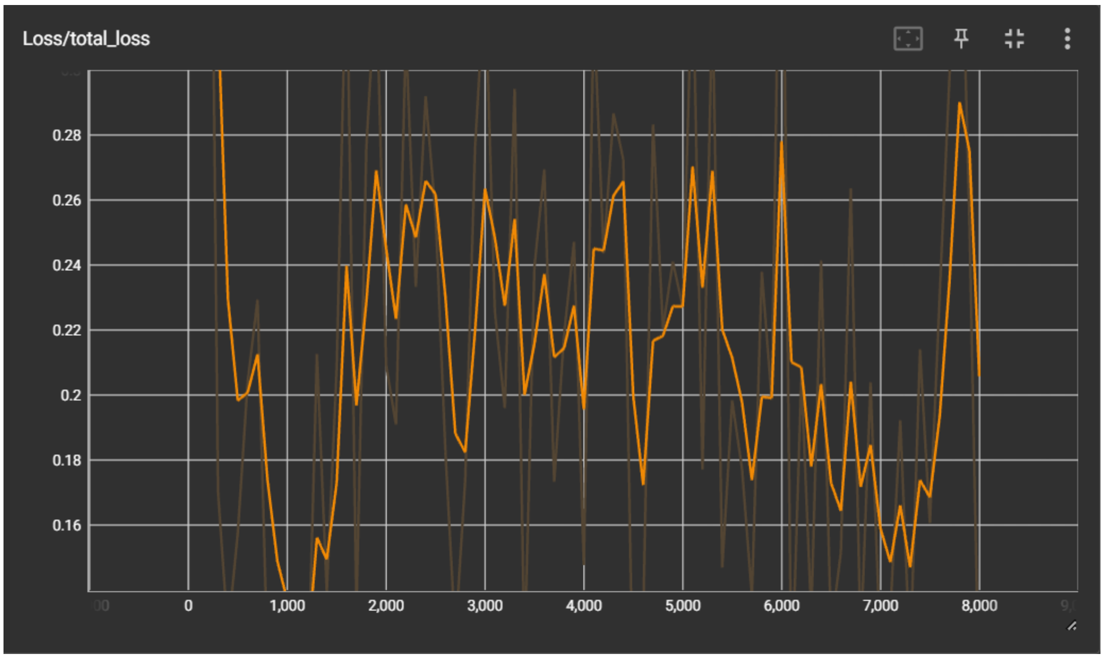


Figure 7.5. Total loss graph of the ResNet101 V1 800x1333 model.

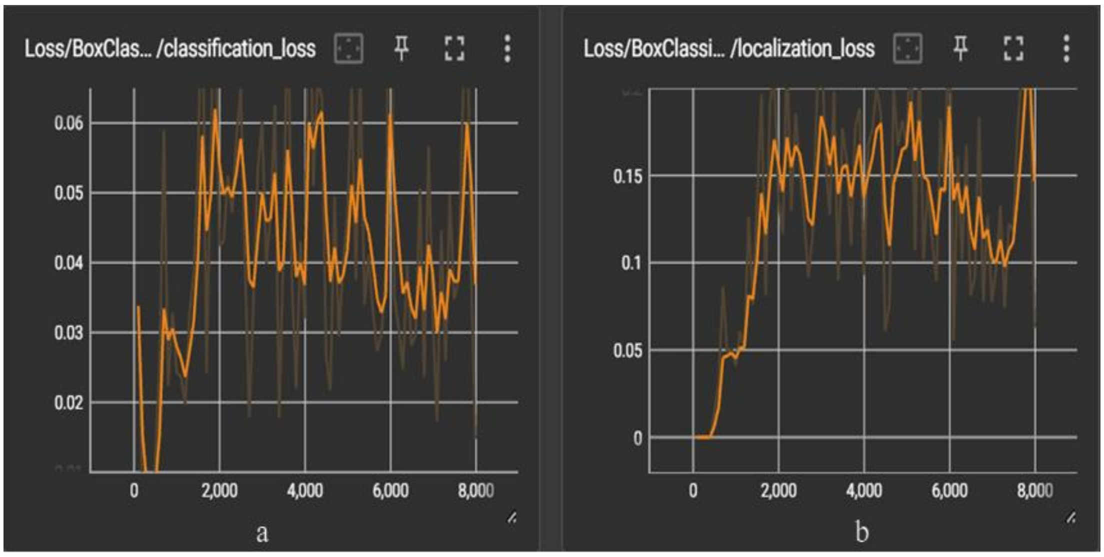


Figure 7.6. Loss graphs of the model. (a) Classification (b) localization.

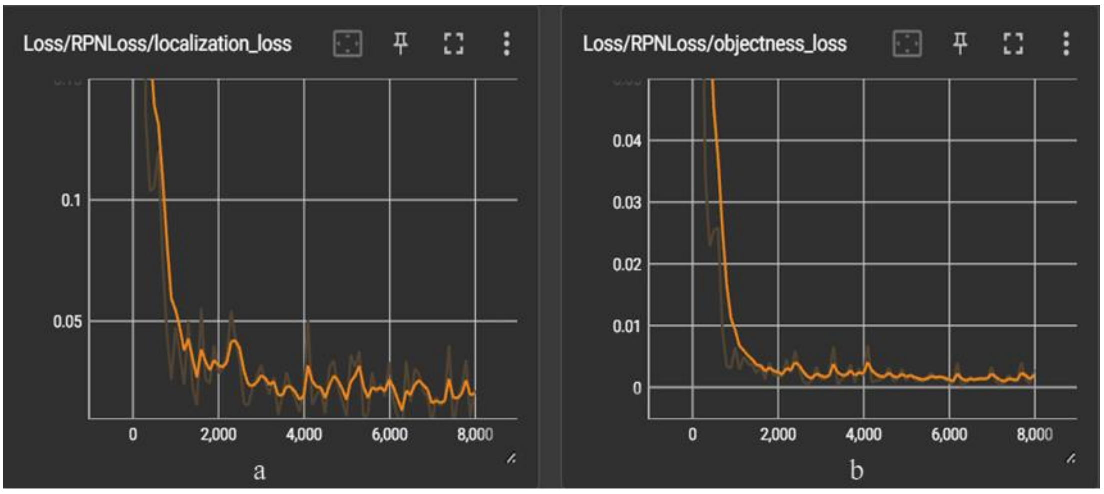


Figure 7.7. Loss graphs of RPN. (a) Localization (b) objectness.

The obtained results based on AP in the test phase are shown in Table 7.3.

Table 7.3. The obtained results based on AP.

Model	**AP (0.5:0.95) (%)**	**AP (0.50) (%)**	**AP (0.75) (%)**	**Total loss**
ResNet101 V1 800x1333	25.0	62.5	14.7	0.193

The obtained results based on AS in the test phase are shown in Table 7.4.

Table 7.4. The obtained results based on AS.

Model	**AS (1) (%)**	**AS (10) (%)**	**AS (100) (%)**
ResNet101 V1 800x1333	19.5	34.9	36.1

In this thesis, the learning rate value was set as 0.001 and the number of steps was set as 8000. Figure 7.8 shows the change of learning rate during the training process.

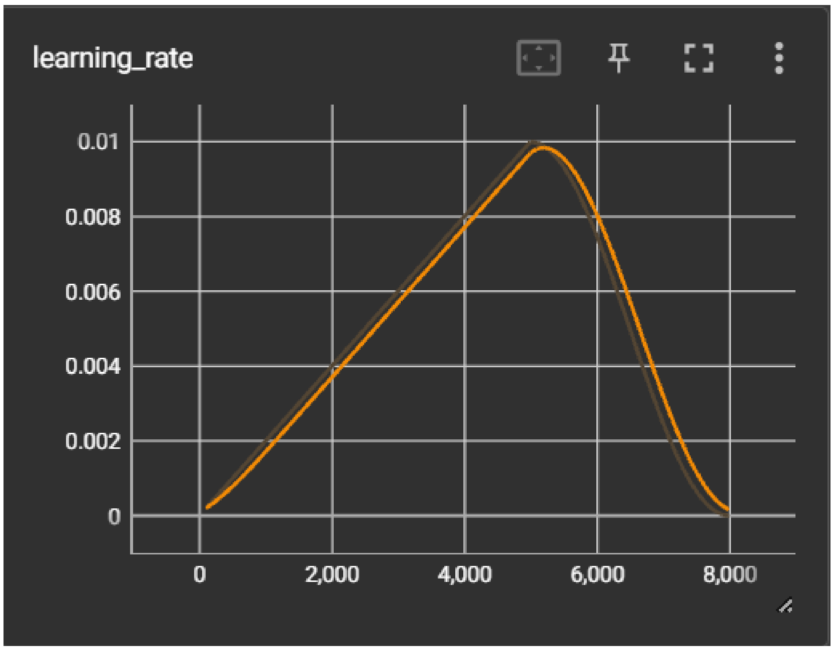


Figure 7.8. The change of learning rate during the training process.

7.1.3. Results of ResNet101 V1 1024x1024 model

The results obtained by the training process of Faster R-CNN ResNet101 V1 1024x1024 model such as total loss, classification loss, localization loss, RPN's localization loss, and RPN's objectness loss, and are shown in Figures 7.9-7.11.

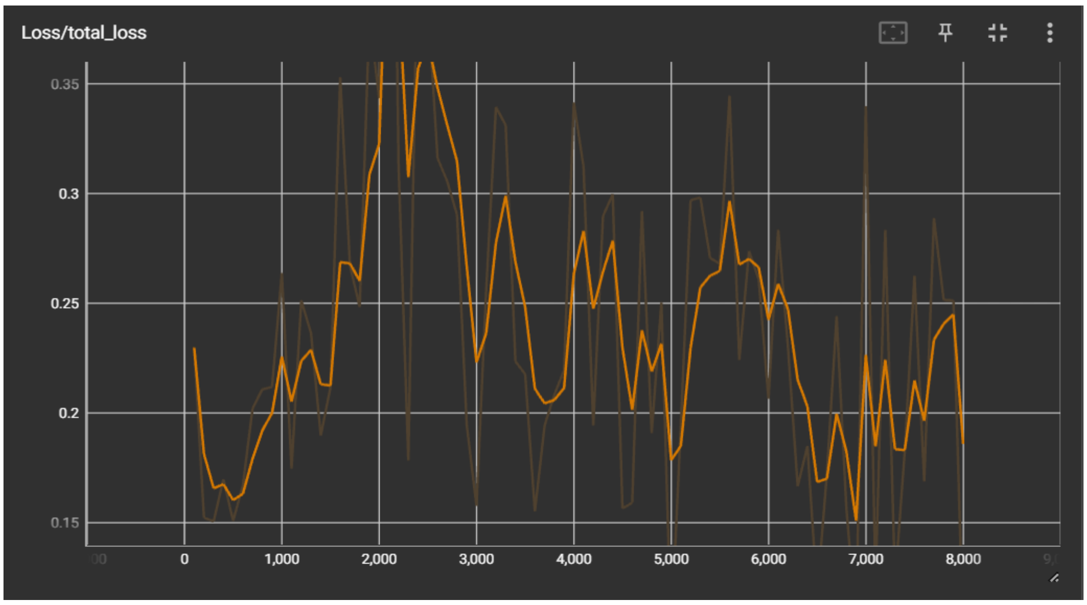


Figure 7.9. Total loss graph of the ResNet101 V1 1024x1024 model.

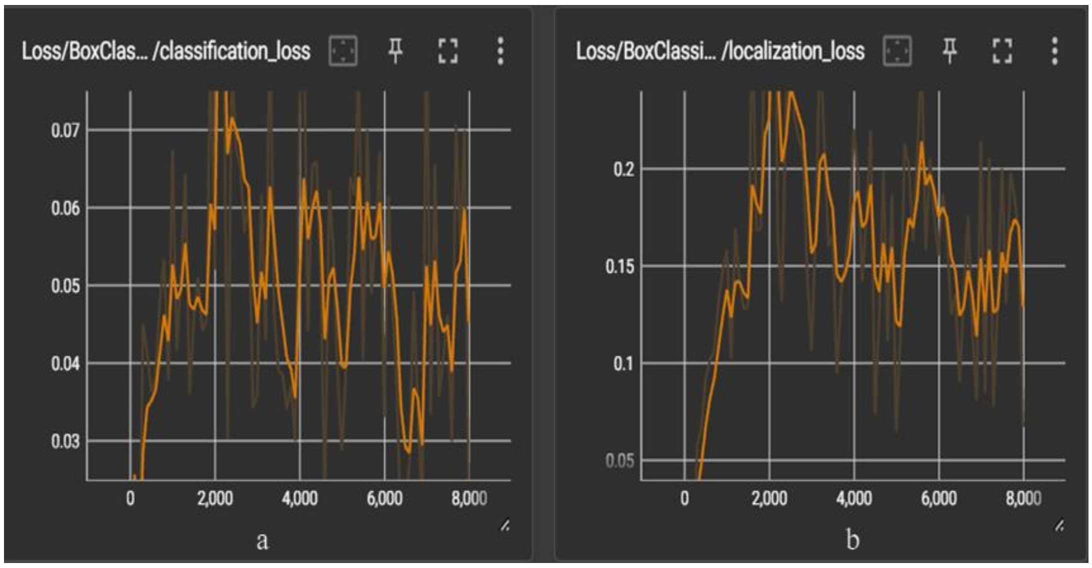


Figure 7.10. Loss graphs of the model. (a) Classification (b) localization.

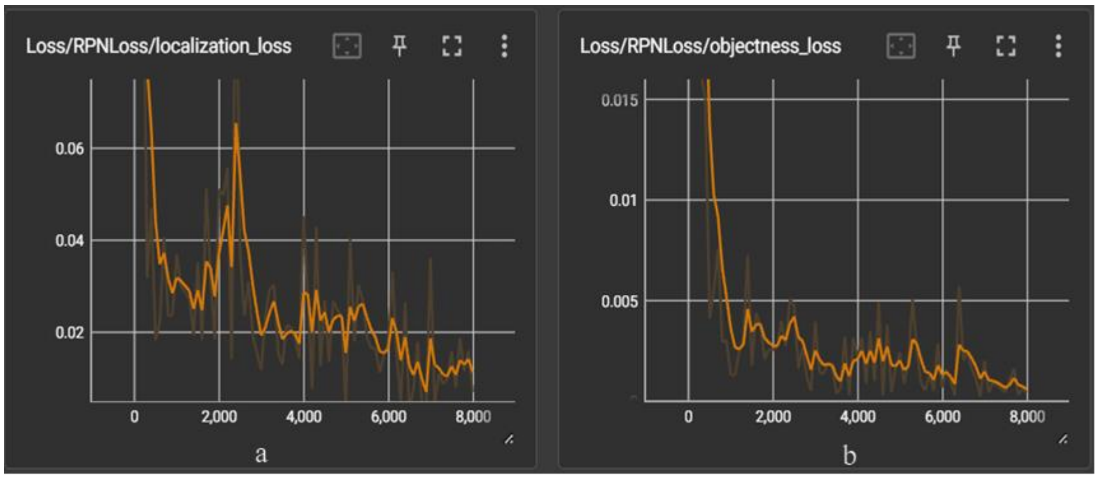


Figure 7.11. Loss graphs of RPN. (a) Localization (b)objectness.

The obtained results based on AP in the test phase are shown in Table 7.5.

Table 7.5. The obtained results based on AP.

Model	**AP (0.5:0.95) (%)**	**AP (0.50) (%)**	**AP (0.75) (%)**	**Total loss**
ResNet101 V1 1024x1024	23.5	63.7	12.0	0.138

The obtained results based on AS in the test phase are shown in Table 7.6.

Table 7.6. The obtained results based on AS.

Model	**AS (1) (%)**	**AS (10) (%)**	**AS (100) (%)**
ResNet101 V1 1024x1024	19.1	33.1	34.3

In this thesis, the learning rate value was set as 0.001 and the number of steps was set as 8000.Figure 7.12 shows the change of learning rate during the training process.

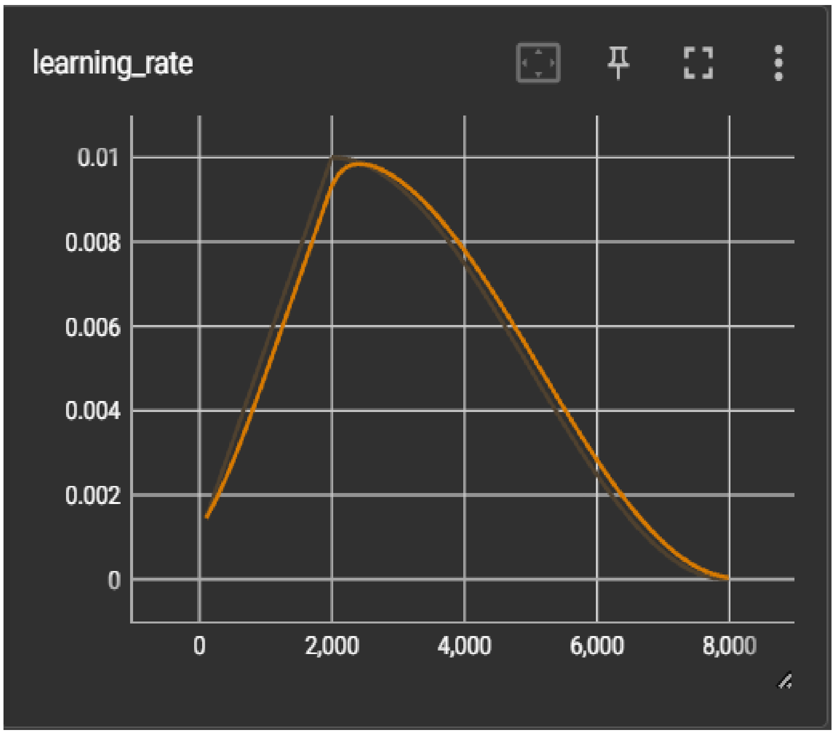


Figure 7.12. The change of learning rate.

Table 7.7 shows a summary of the obtained results among the models in respect to AP and AS.

Table 7.7. Summary of the obtained results of AP

Model	AP (0.5:0.95) (%)	AP (0.50) (%)	AP (0.75) (%)	AS (1) (%)	AS (10) (%)	AS (100) (%)
Faster R-CNN ResNet50 V1 800x1333	24.8	62.9	14.0	20.5	34.2	35.4
Faster R-CNN ResNet101 V1 800x1333	25.0	62.5	14.7	19.5	34.9	36.1
Faster R-CNN ResNet101 V1 1024x1024	23.5	63.7	12.0	19.1	33.1	34.3

7.1.4. Experimental results of ResNet50 V1 800x1333 model

Visual representations of the detection results achieved by the Faster R-CNN ResNet50 V1 800x1333 architecture during the test phase of kidney stones in the dataset alongside with the ground truth bounding boxes are given in Figures 7.13-7.15.

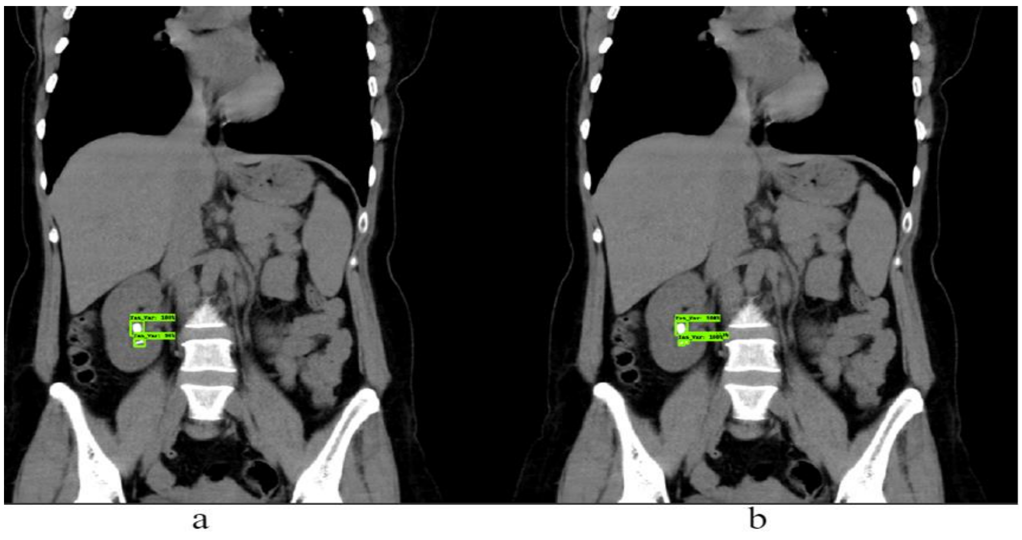


Figure 7.13. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

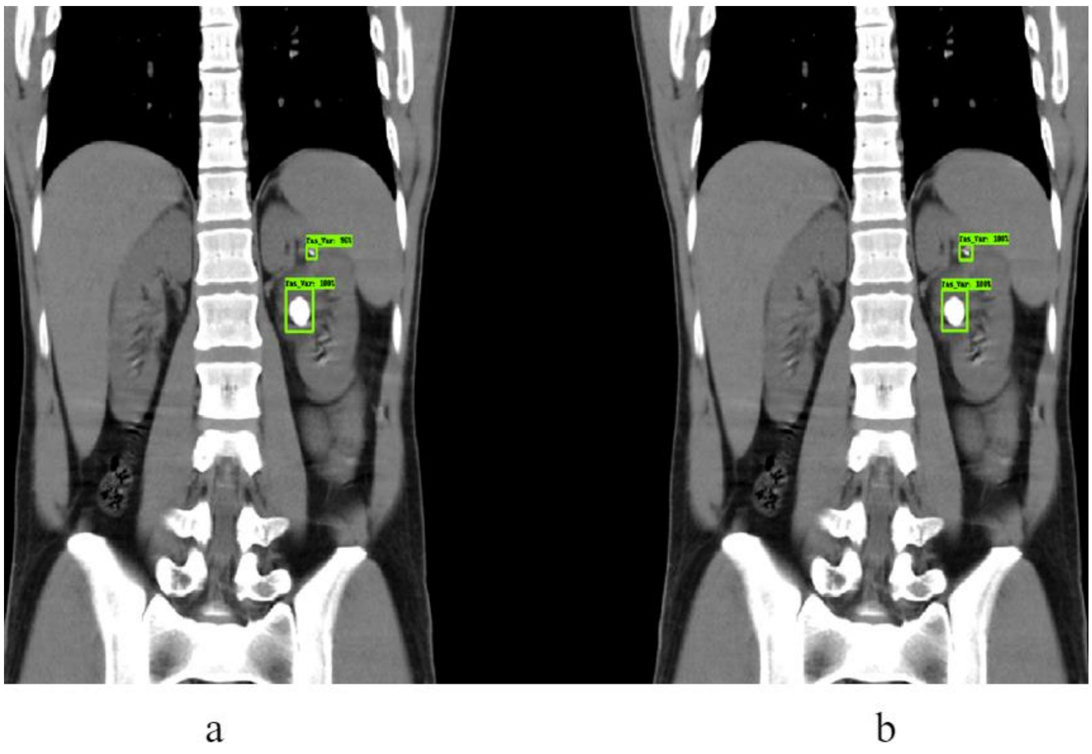


Figure 7.14. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

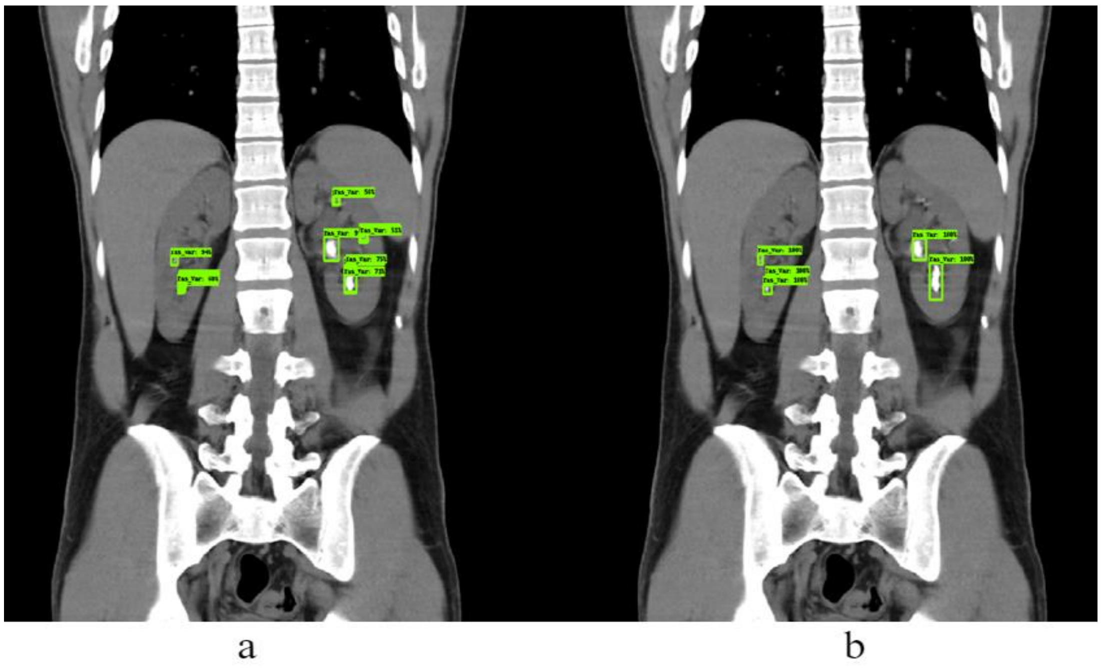


Figure 7.15. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

7.1.5. Experimental results of ResNet101 V1 800x1333 model

Visual representations of the detection results achieved with the Faster R-CNN ResNet101 V1 800x1333 architecture during the test phase of kidney stones in the dataset alongside with the ground truth bounding boxes are given in Figure 7.16, and 7.17, respectively.

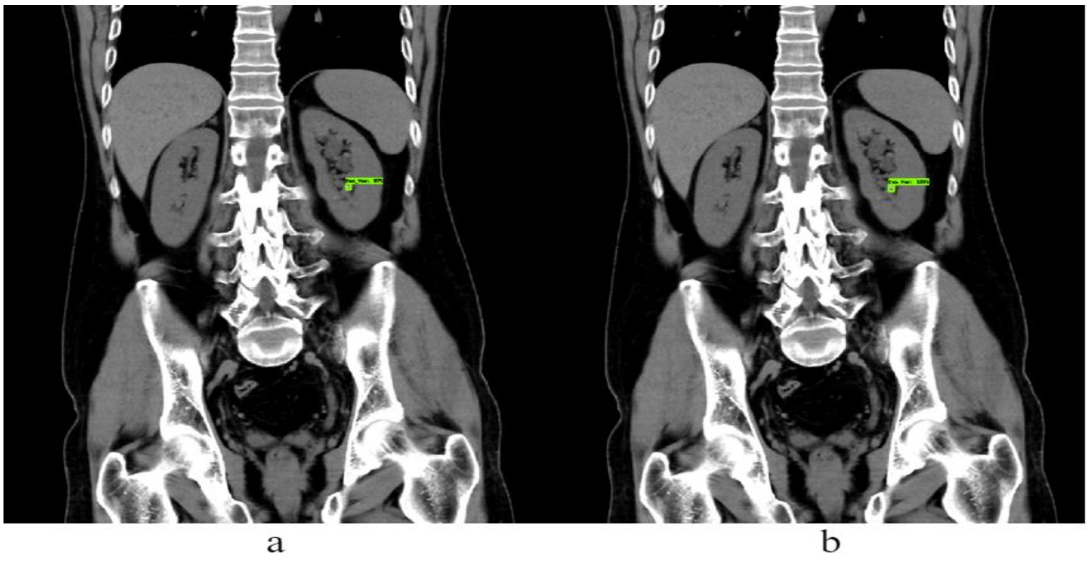


Figure 7.16. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

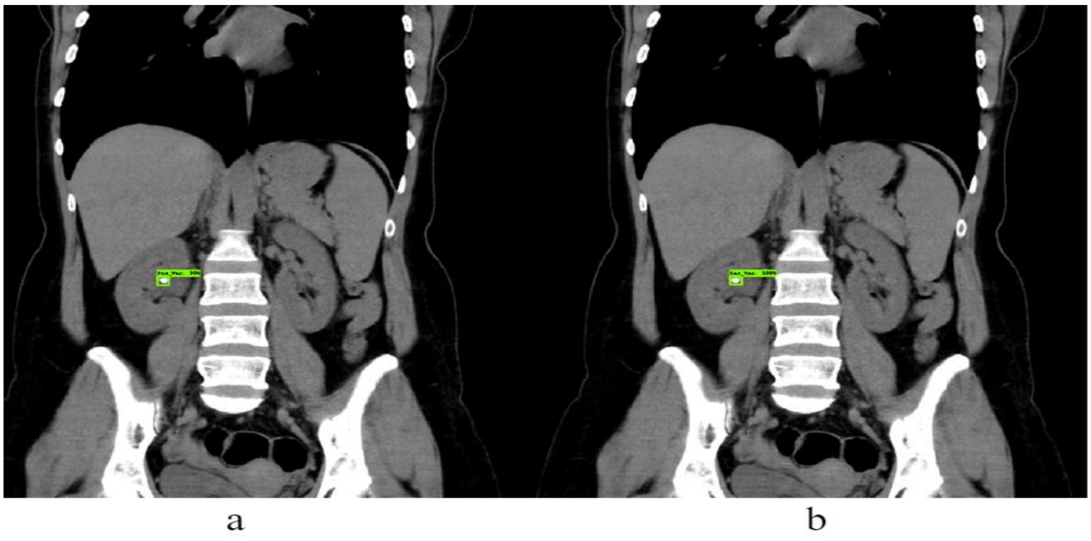


Figure 7.17. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes

7.1.6. Experimental results of ResNet101 V1 1024x1024 model

Visual representations of the detection results achieved with the Faster R-CNN ResNet101 V1 1024x1024 architecture during the test phase of kidney stones in the dataset alongside with the ground truth bounding boxes are given in Figure 7.18, and 7.19, respectively.

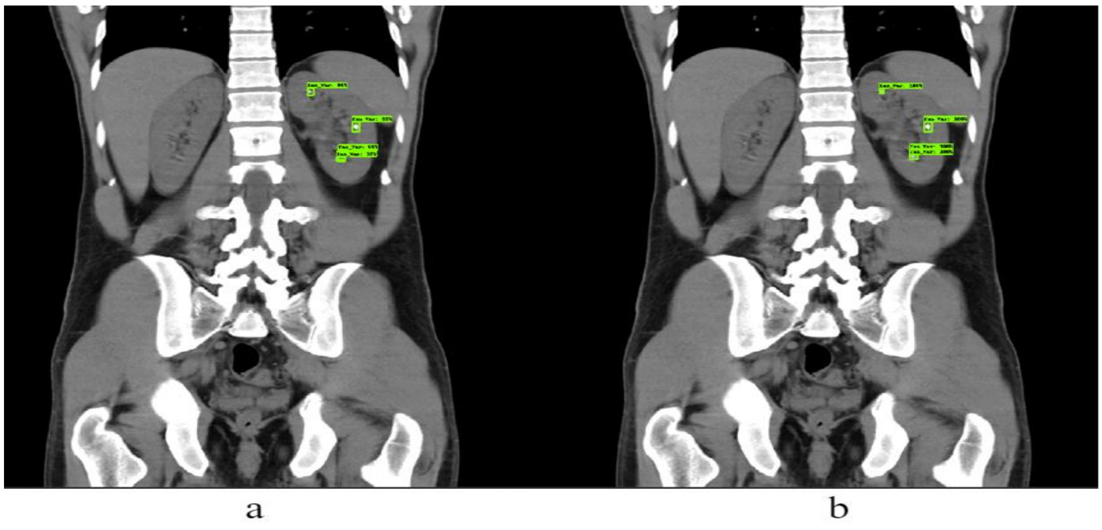


Figure 7.18. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

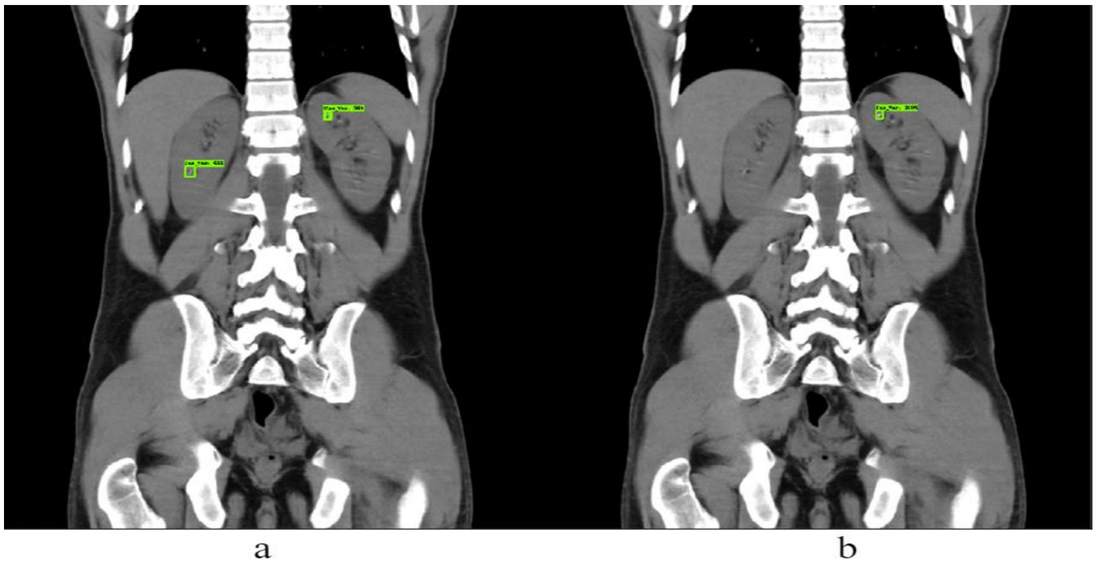


Figure 7.19. Evaluation of kidney stone. (a) Model’s predictions (b) ground truth bounding boxes.

7.2. YOLO

In this thesis, as stated before for the purpose of detection of kidney stone using YOLO algorithm, four datasets named as D1, D2, D3, and D4 were utilized.

Both YOLO v5 and YOLO v7 models were utilized for the detection of kidney stones, in addition to that both SGD and Adam optimizers were selected to perform the detection process. Furthermore, the epoch size and batch size values were set as 100 and 32, respectively.

7.2.1. YOLO v5

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the SGD optimizer on D1 dataset are shown in Figures 7.20-7.22.

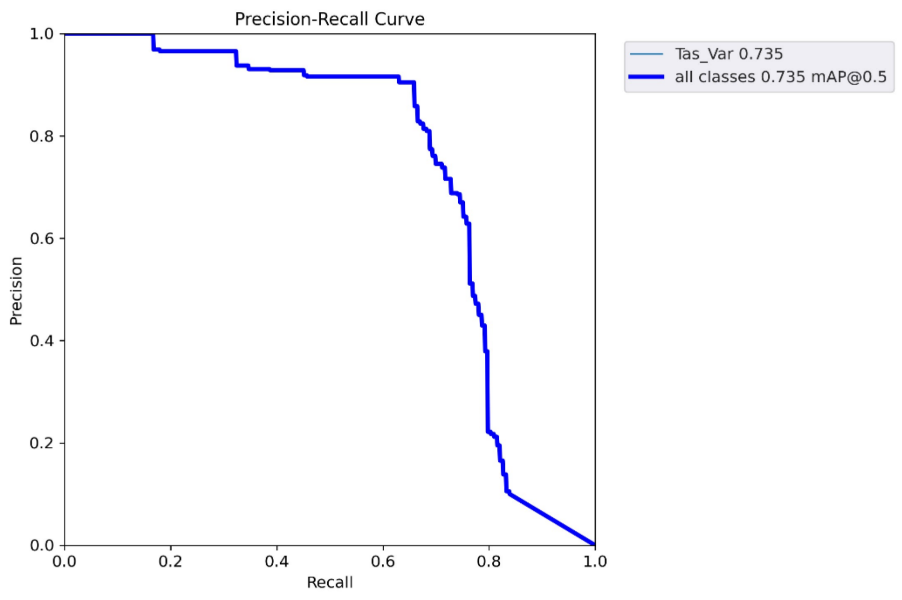


Figure 7.20. Precision-Sensitivity curve for D1 dataset (Train Phase).

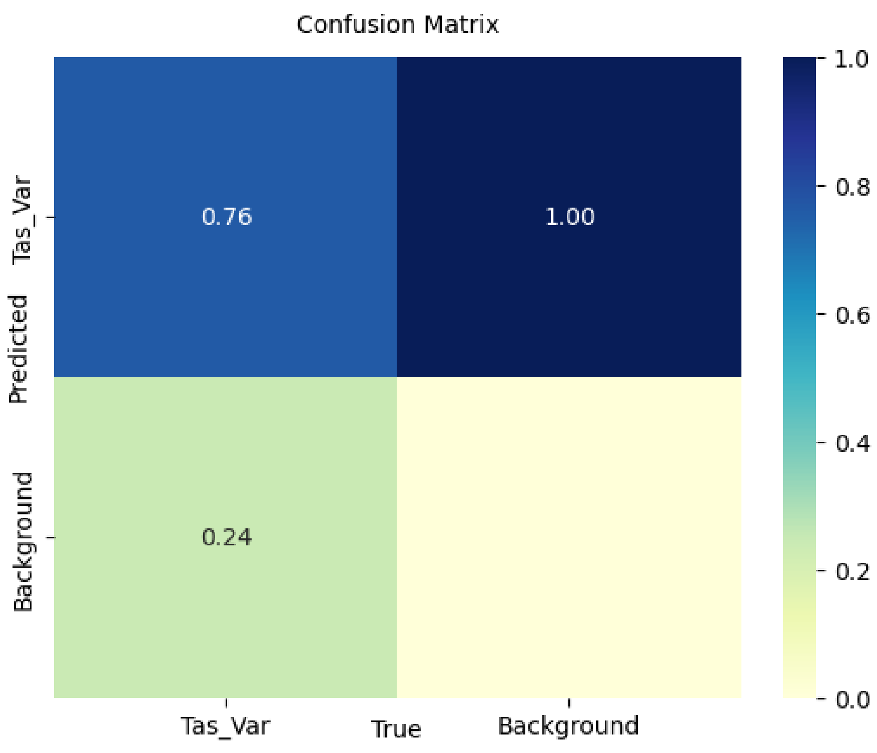


Figure 7.21. Confusion matrix for D1 dataset (Train Phase).

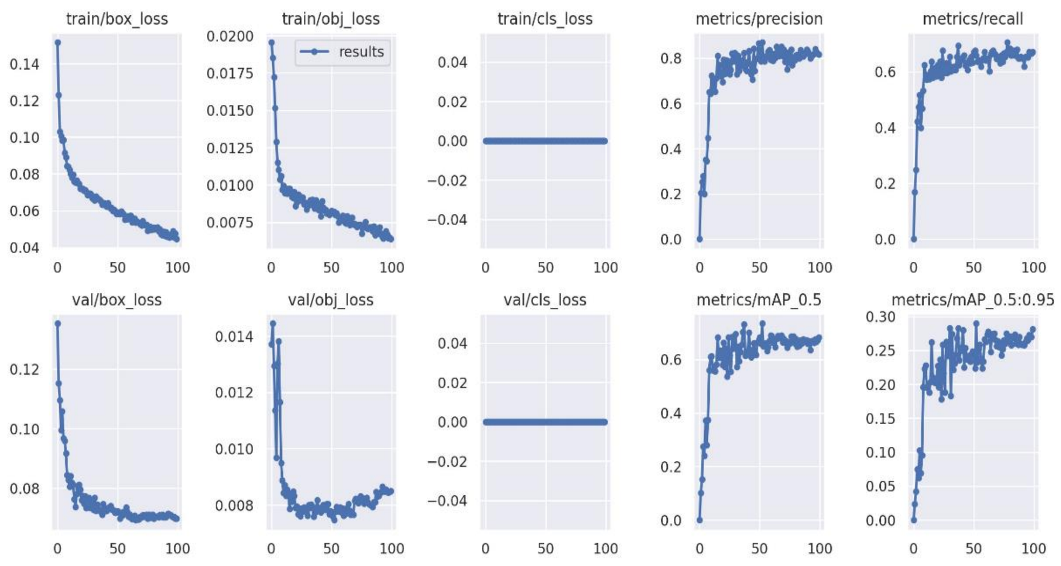


Figure 7.22. Train and validation losses for D1 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the SGD optimizer on D1 dataset are shown in Figures 7.23, and 7.24, respectively.

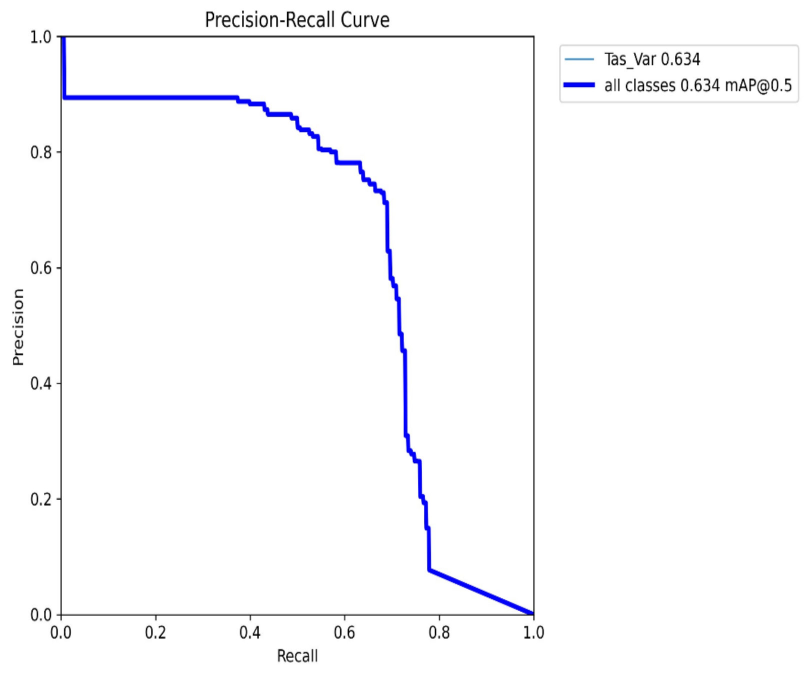


Figure 7.23. Precision-Sensitivity curve for D1 dataset (Test phase).

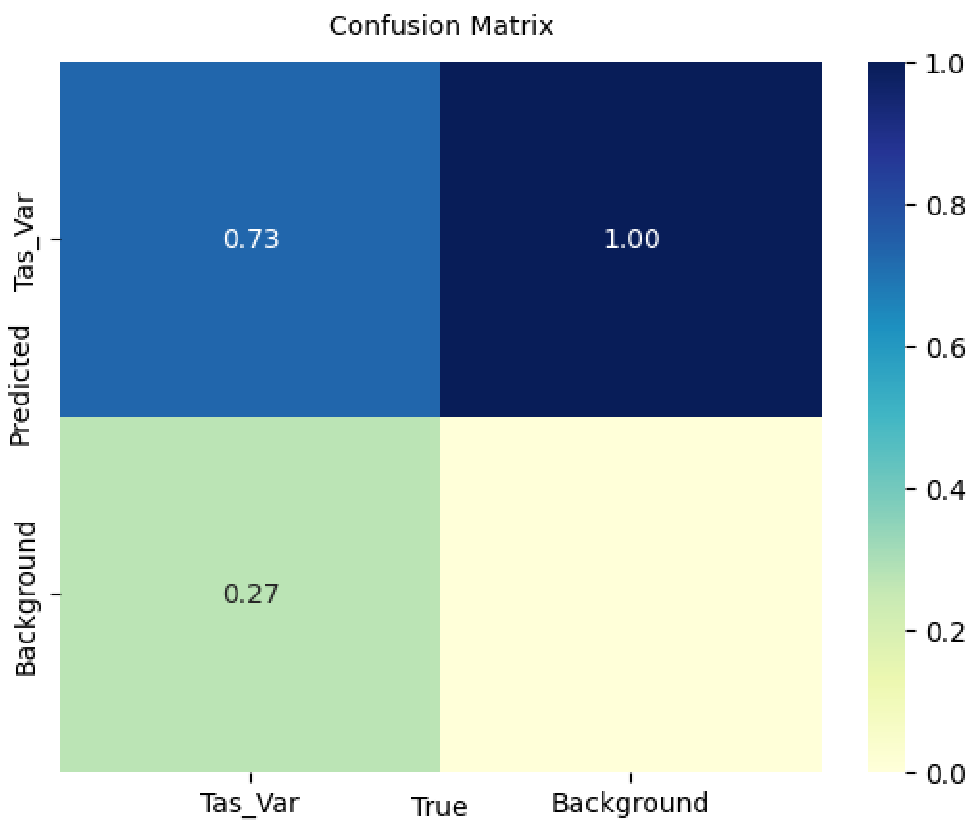


Figure 7.24. Confusion matrix for D1 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the SGD optimizer on D2 dataset are shown in Figures 7.25-7.27.

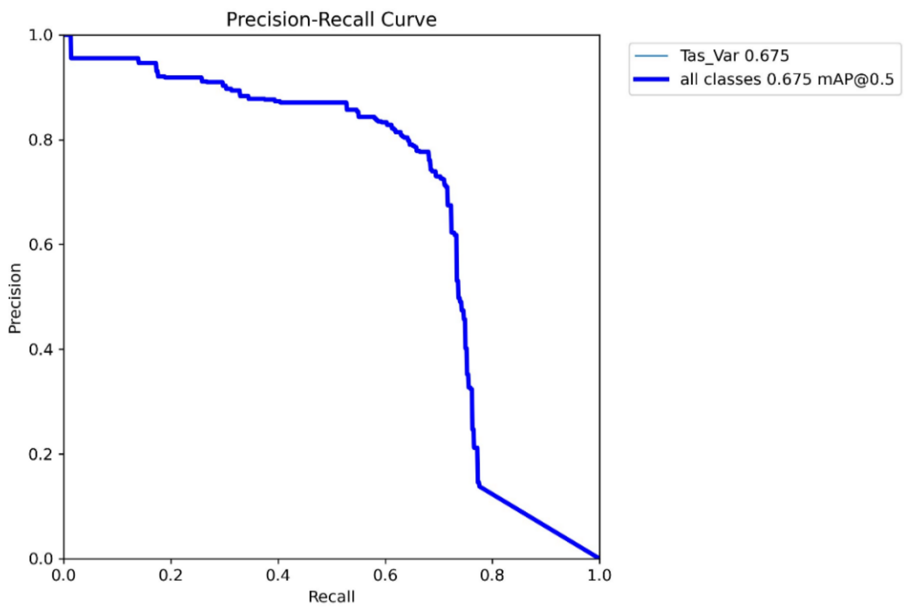


Figure 7.25. Precision-Sensitivity curve for D2 dataset (Train Phase).

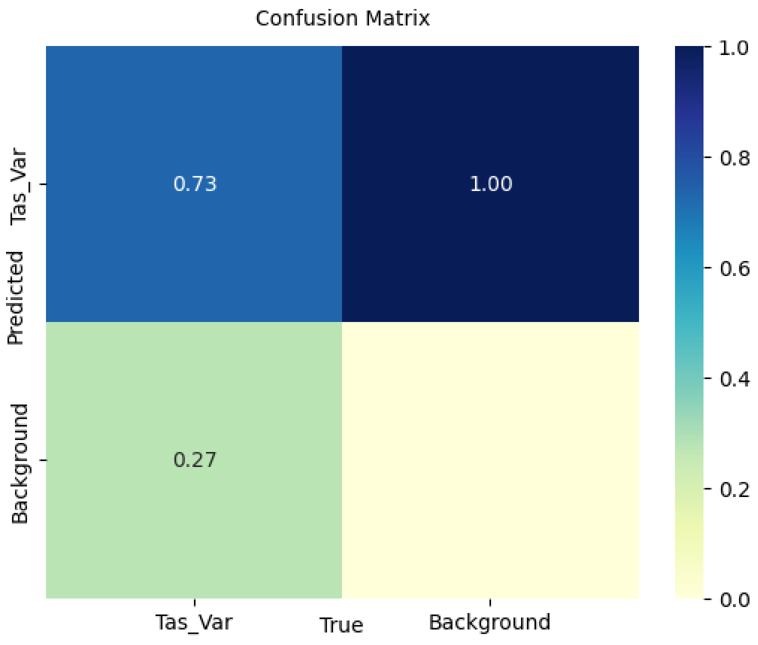


Figure 7.26. Confusion matrix for D2 dataset (Train phase).

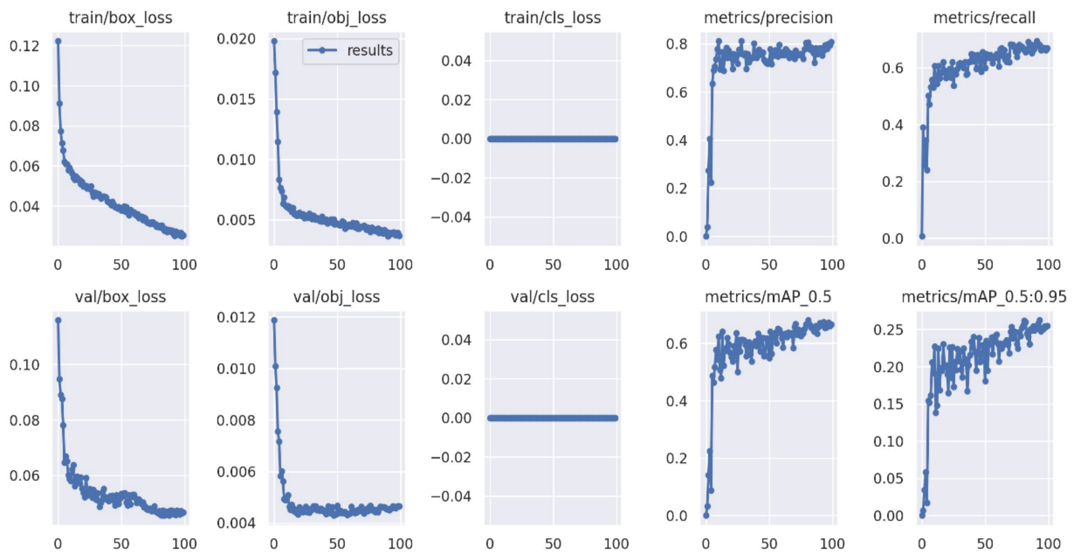


Figure 7.27. Train and validation losses for D2 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the SGD optimizer on the D2 dataset are shown in Figures 7.28, and 7.29, respectively.

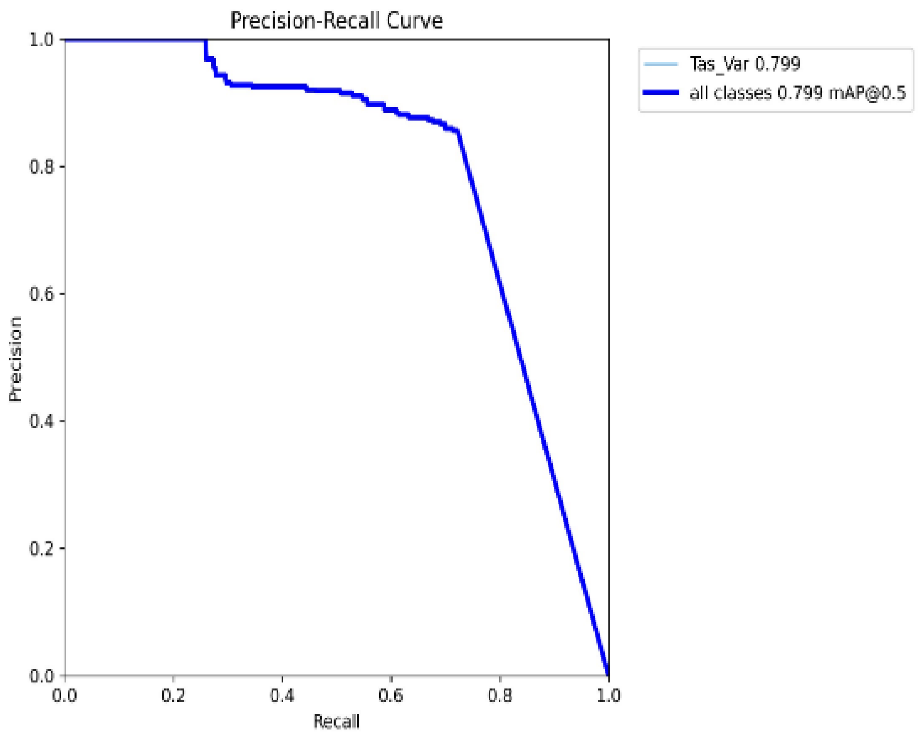


Figure 7.28. Precision-Sensitivity curve for D2 dataset (Test phase).

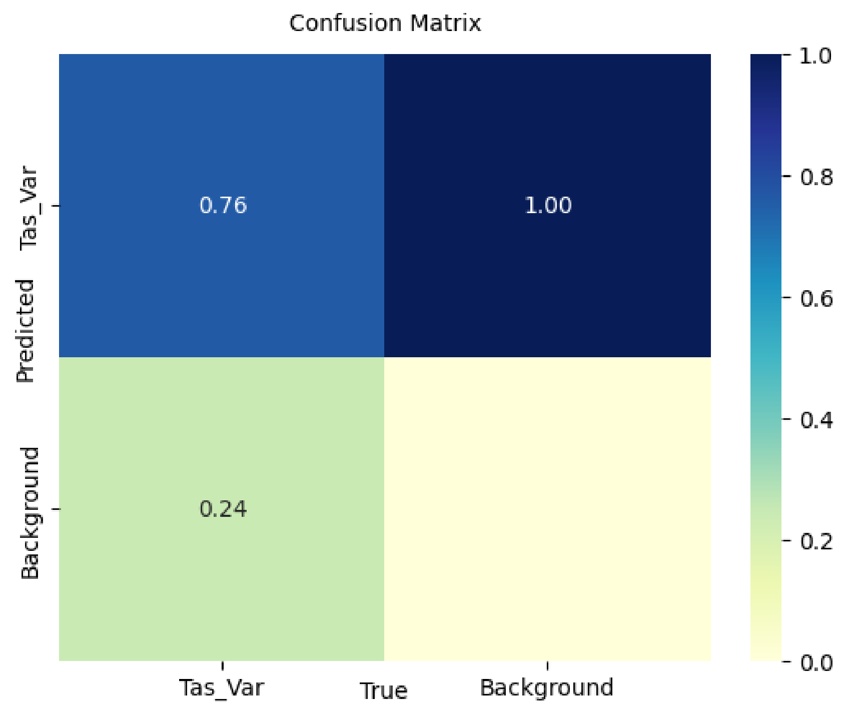


Figure 7.29. Confusion matrix for D2 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the SGD optimizer on D3 dataset are shown in Figures 7.30-7.32.

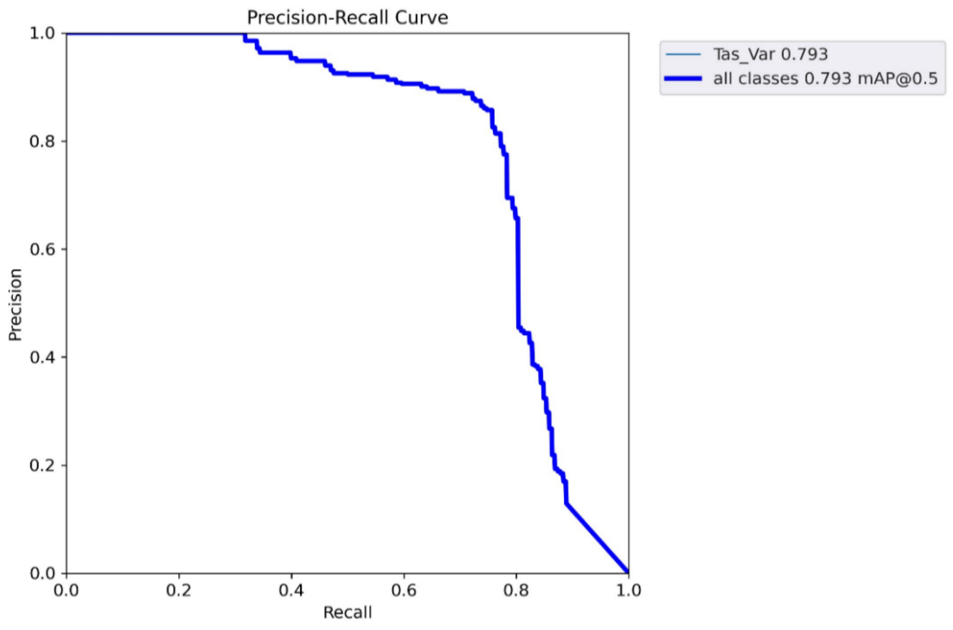


Figure 7.30. Precision-Sensitivity curve for D3 dataset (Train Phase).

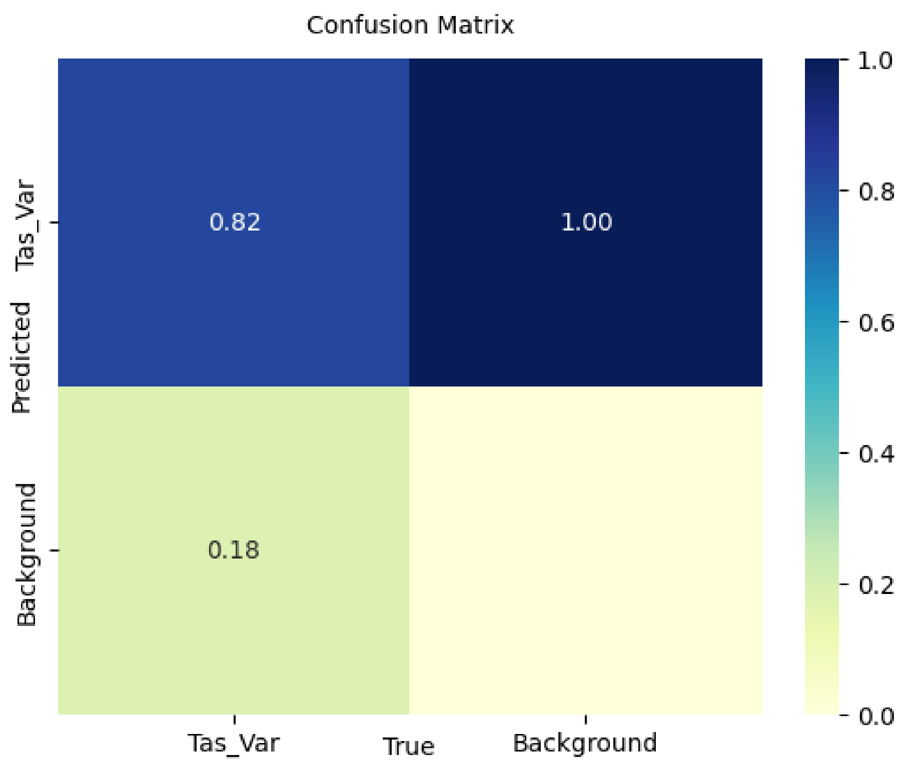


Figure 7.31. Confusion matrix for D3 dataset (Train Phase).

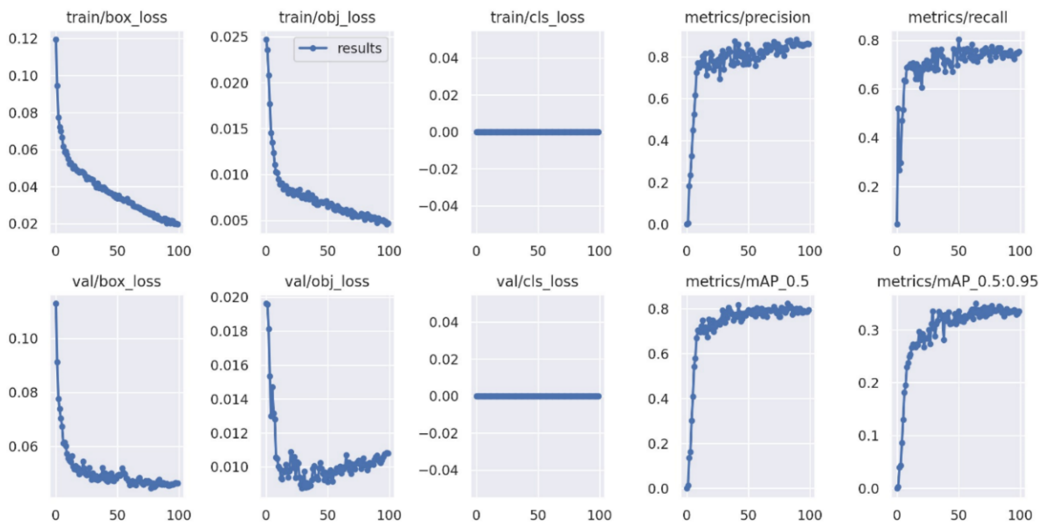


Figure 7.32. Train and validation losses for D3 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the SGD optimizer on the D3 dataset are shown in Figures 7.33, and 7.34, respectively.

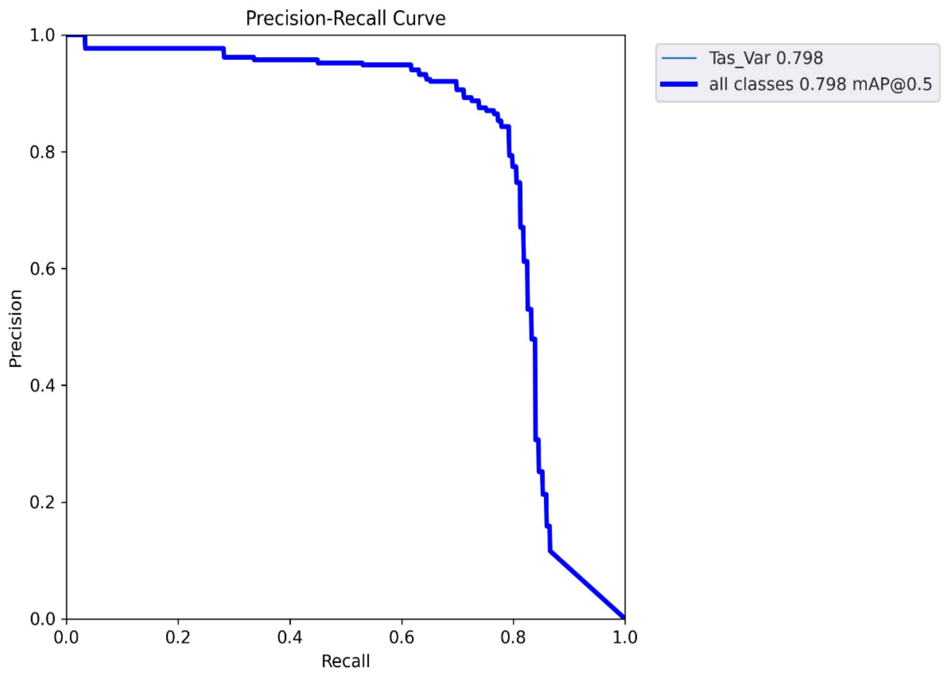


Figure 7.33. Precision-Sensitivity curve for D3 dataset (Test phase).

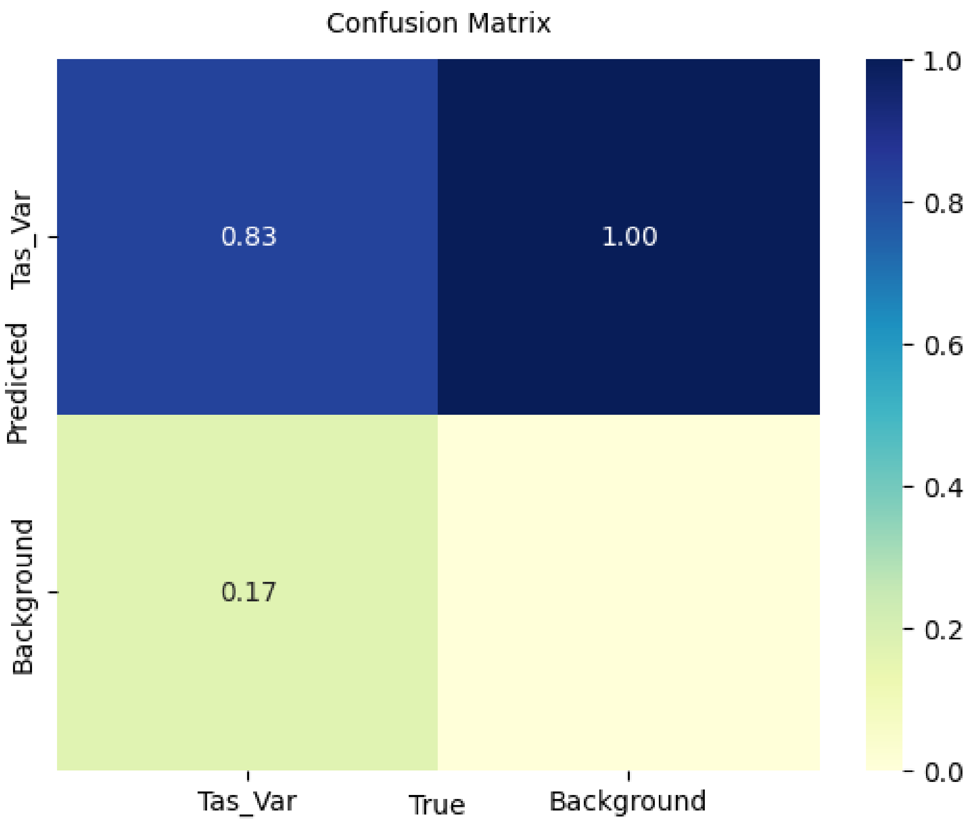


Figure 7.34. Confusion matrix for D3 dataset (Test Phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the SGD optimizer on D4 dataset are shown in Figure 7.35-7.37.

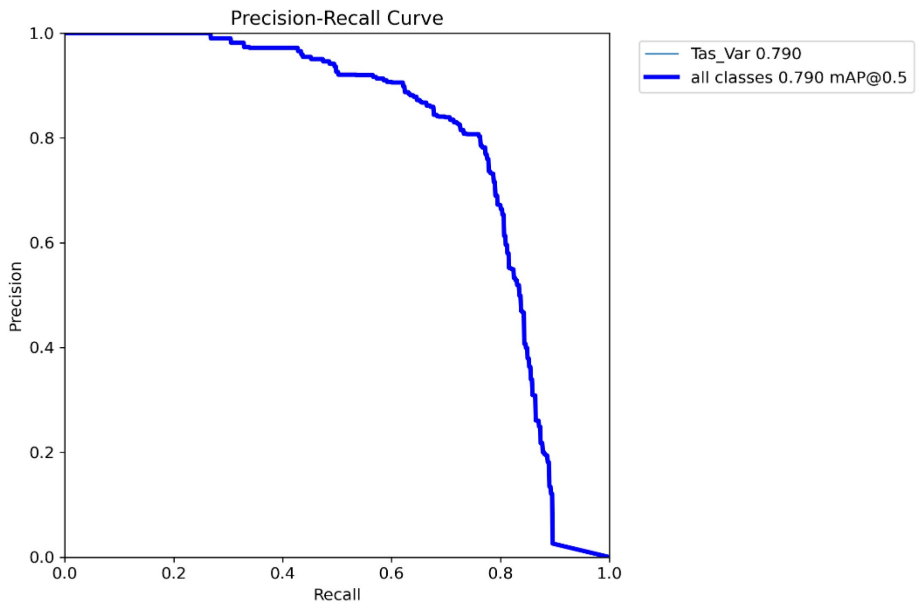


Figure 7.35. Precision-Sensitivity curve for D4 dataset (Train Phase).

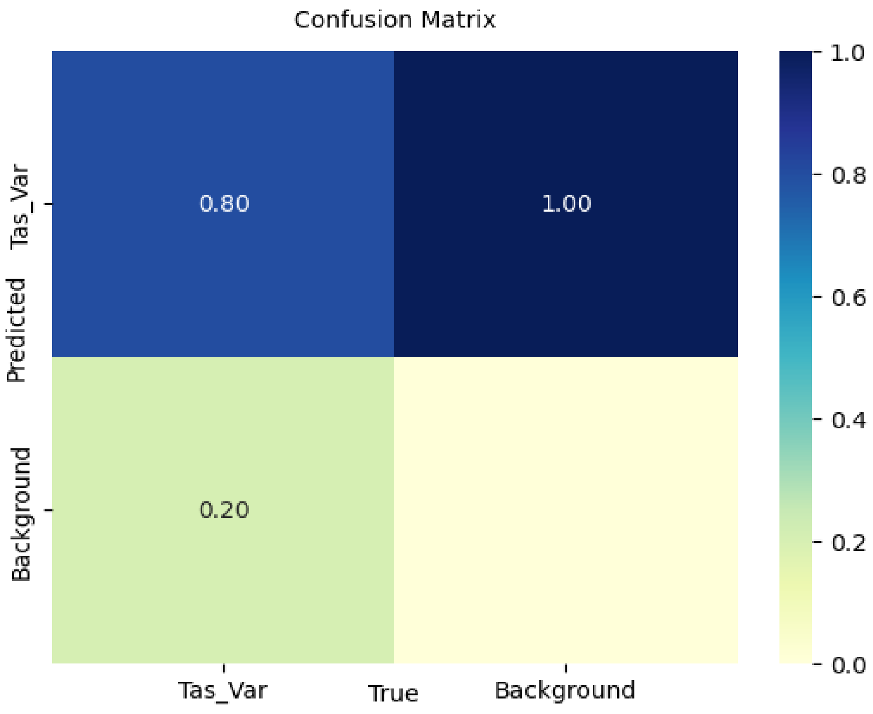


Figure 7.36. Confusion matrix for D4 dataset (Train Phase).

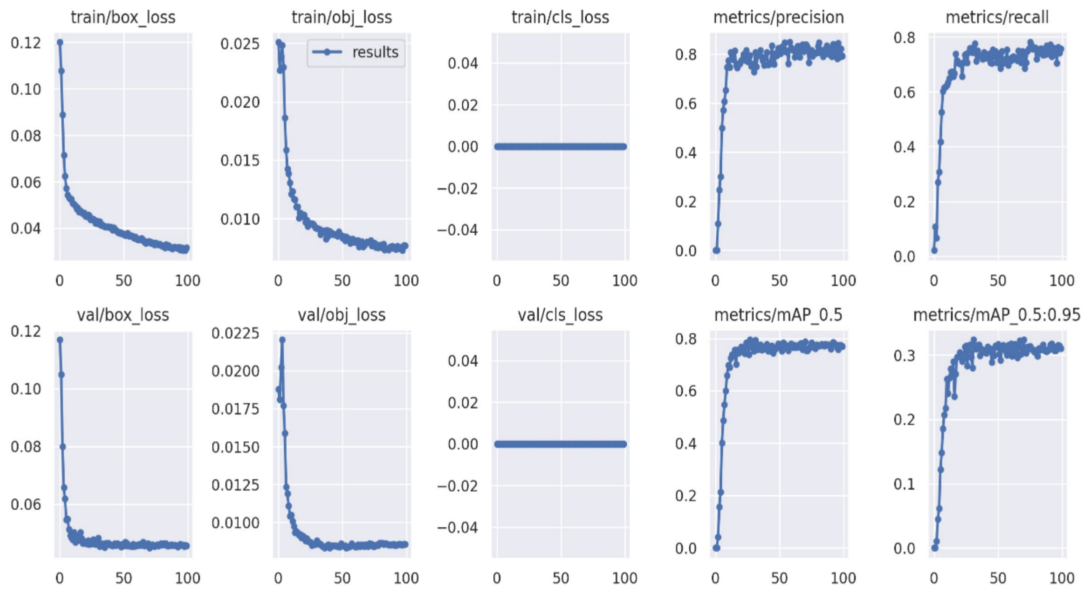


Figure 7.37. Train and validation losses for D4 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the SGD optimizer on the D4 dataset are shown in Figures 7.38, and 7.39, respectively.

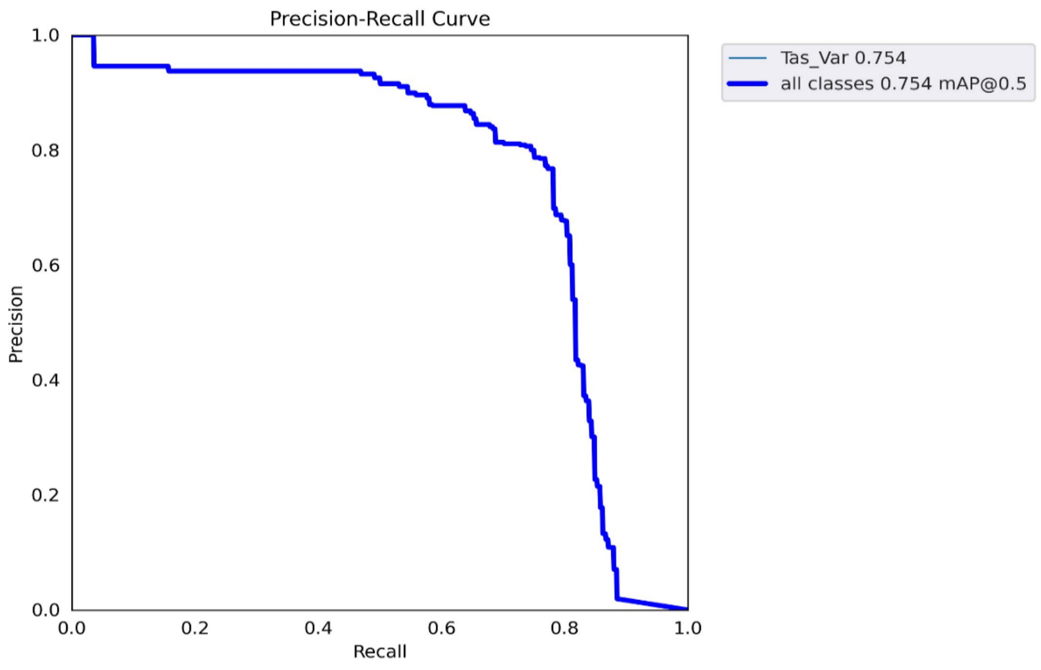


Figure 7.38. Precision-Sensitivity curve for D4 dataset (Test phase).

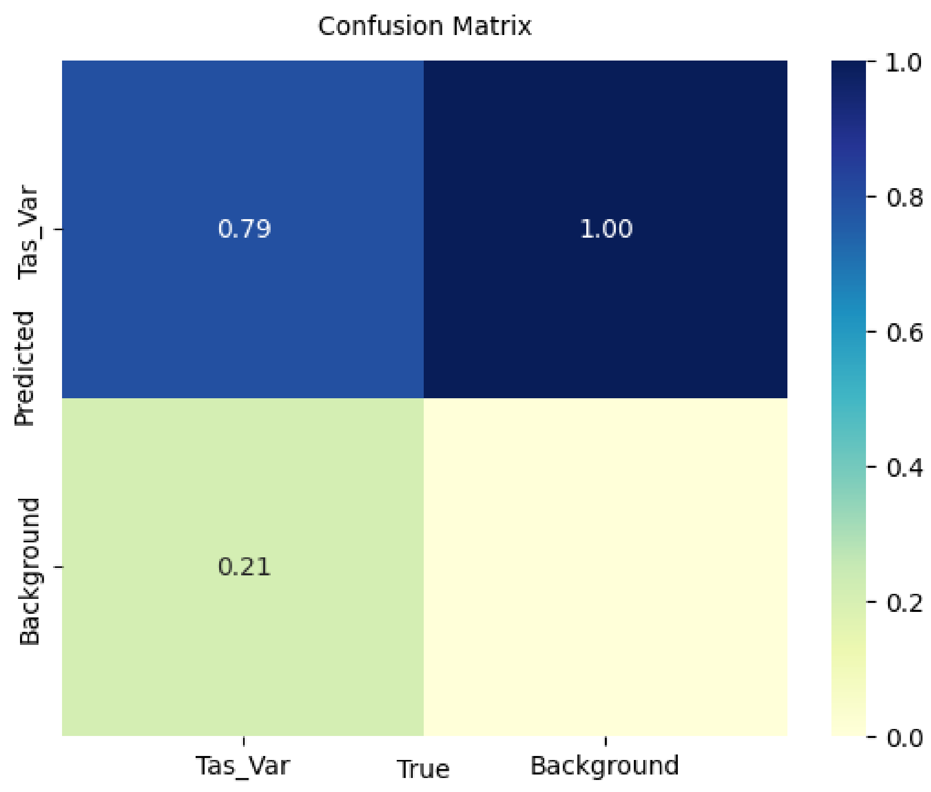


Figure 7.39. Confusion matrix for D4 dataset (Test phase).

Table 7.8 shows a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, and mAP in both of Train and Test phases for YOLO v5 model with SGD optimizer.

Table 7.8. Train and test results of the model.

	Train				**Test**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	76.0	73.0	82.0	80.0	73.0	76.0	83.0	79.0
Precision (%)	87.0	77.3	85.7	80.2	73.3	85.6	85.3	78.1
Sensitivity (%)	65.9	68.1	75.7	76.0	66.5	72.2	77.7	76.8

F1 score (%)	74.9	72.4	80.3	78.0	69.7	78.3	81.3	77.4
mAP (0.5) (%)	73.5	67.5	79.3	79.0	63.4	79.9	79.8	75.4
mAP (0.5:0.95) (%)	29.0	26.2	35.0	32.4	25.2	39.0	37.2	31.5

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the Adam optimizer on D1 dataset are shown in Figures 7.40-7.42.

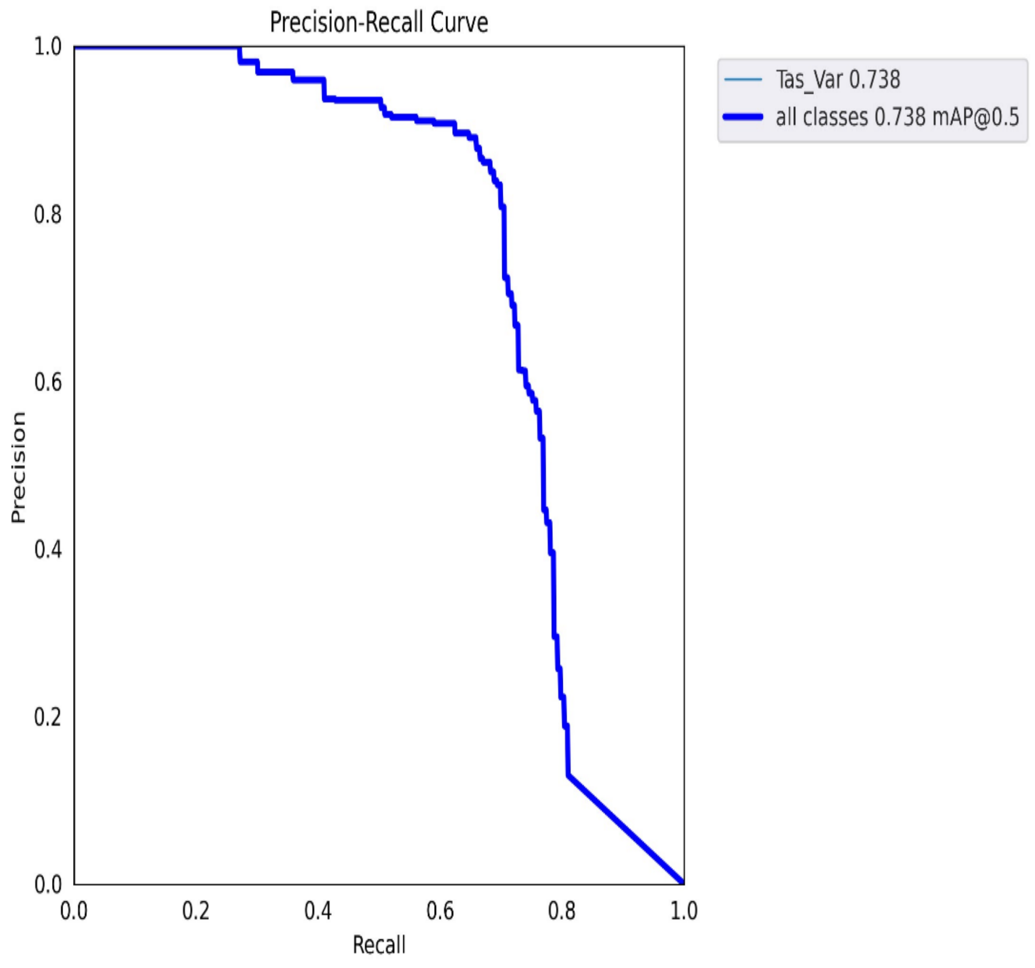


Figure 7.40. Precision-Sensitivity curve for D1 dataset (Train Phase).

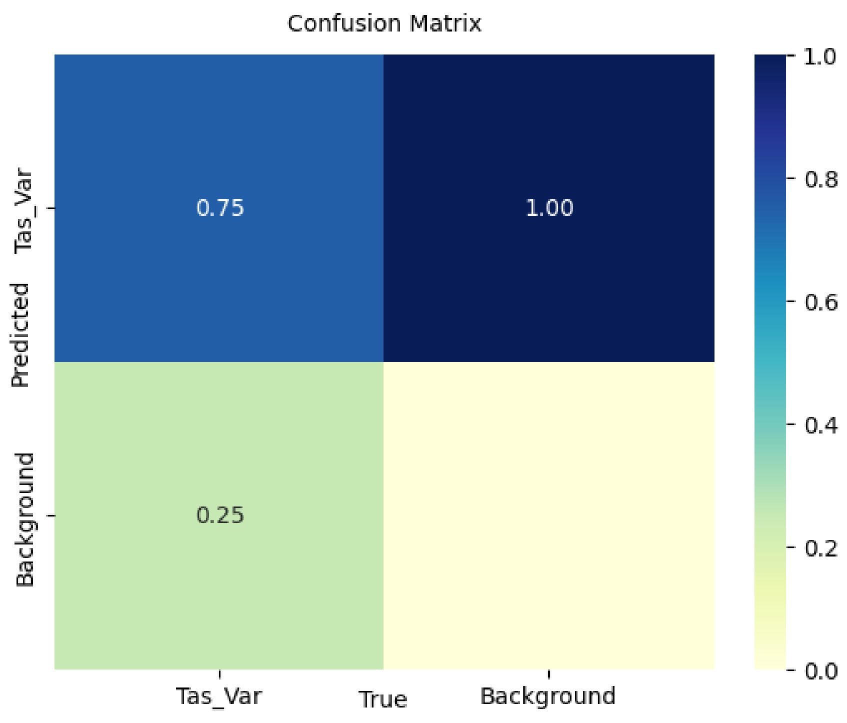


Figure 7.41. Confusion matrix for D1 dataset (Train Phase).

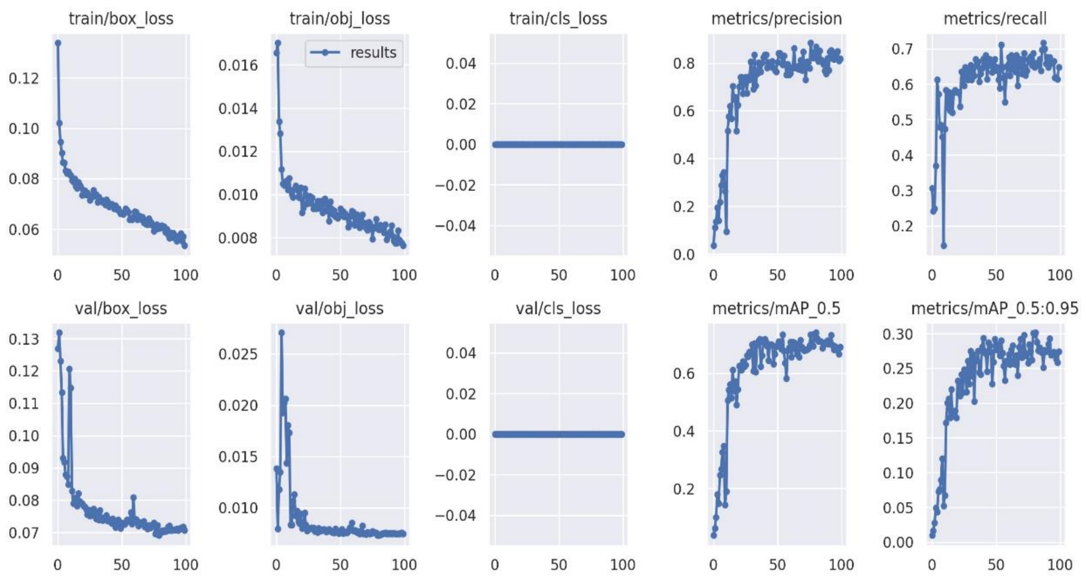


Figure 7.42. Train and validation losses for D1 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the Adam optimizer on D1 dataset are shown in Figures 7.43, and 7.44, respectively.

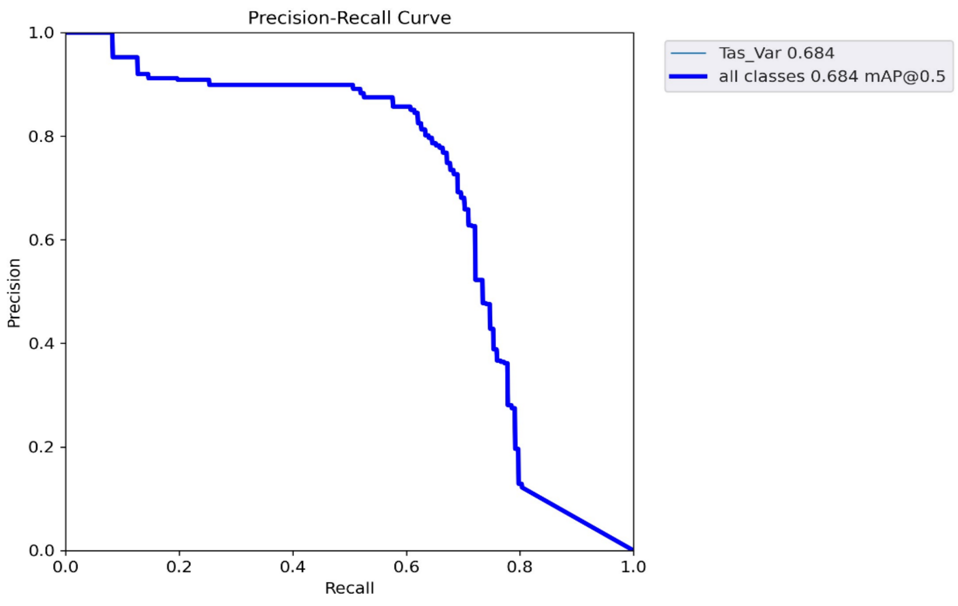


Figure 7.43. Precision-Sensitivity curve for D1 dataset (Test phase).

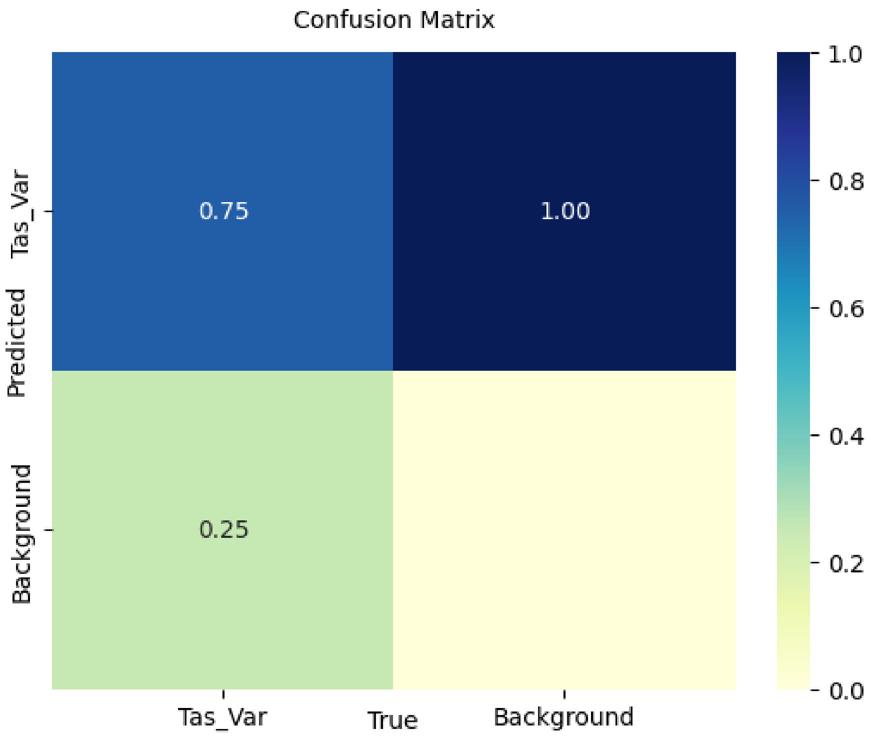


Figure 7.44. Confusion matrix for D1 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the Adam optimizer on D2 dataset are shown in Figures 7.45-7.47.

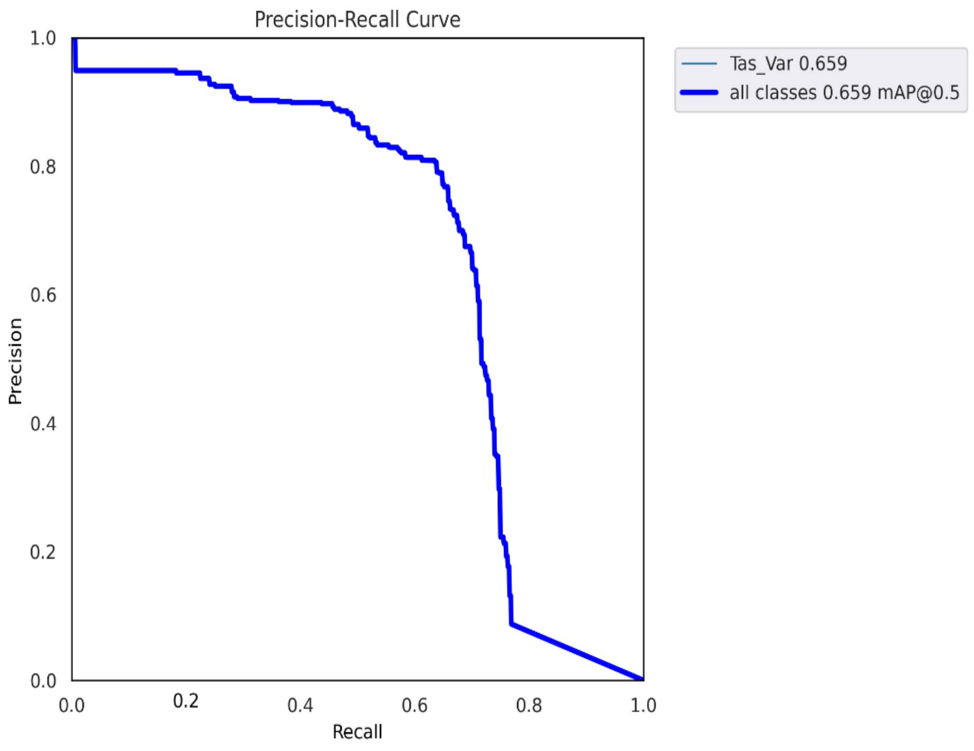


Figure 7.45. Precision-Sensitivity curve for D2 dataset (Train Phase).

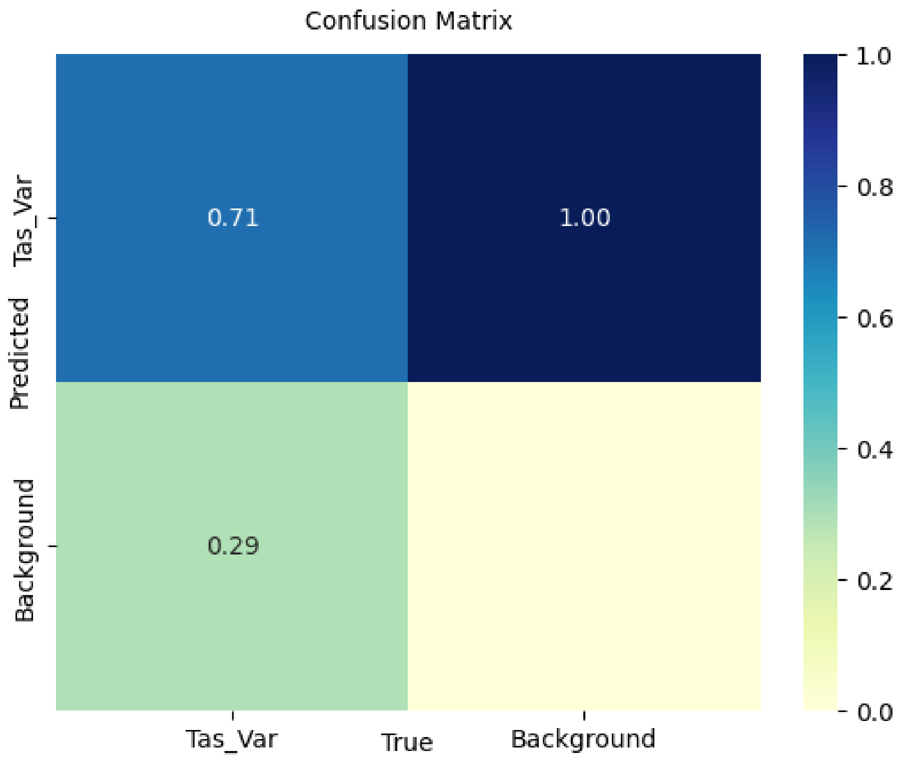


Figure 7.46. Confusion matrix for D2 dataset (Train Phase).

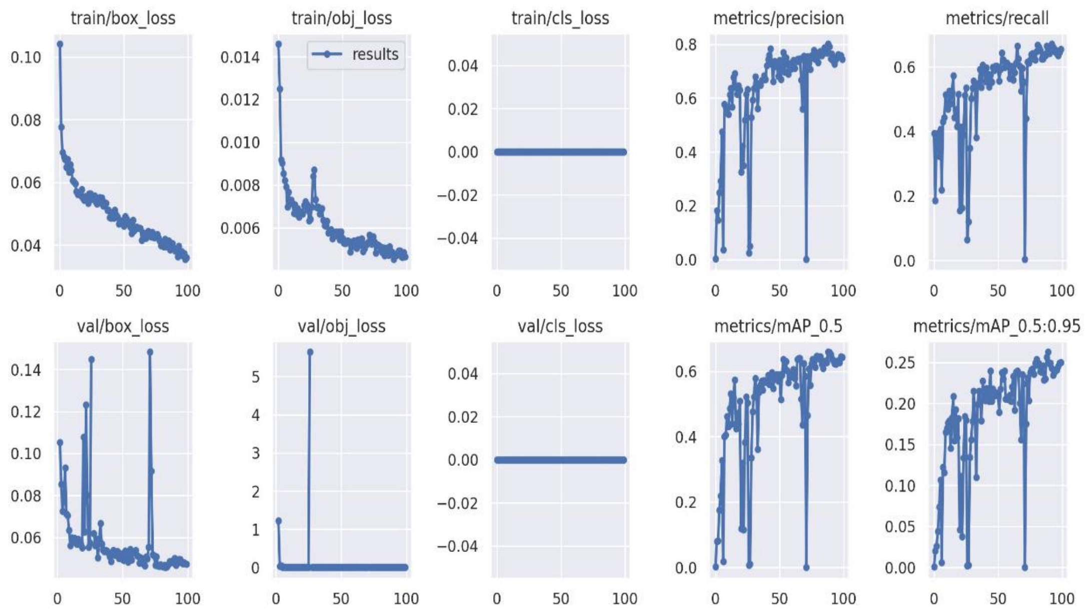


Figure 7.47. Train and validation losses for D2 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the Adam optimizer on D2 dataset are shown in Figures 7.48, and 7.49, respectively.

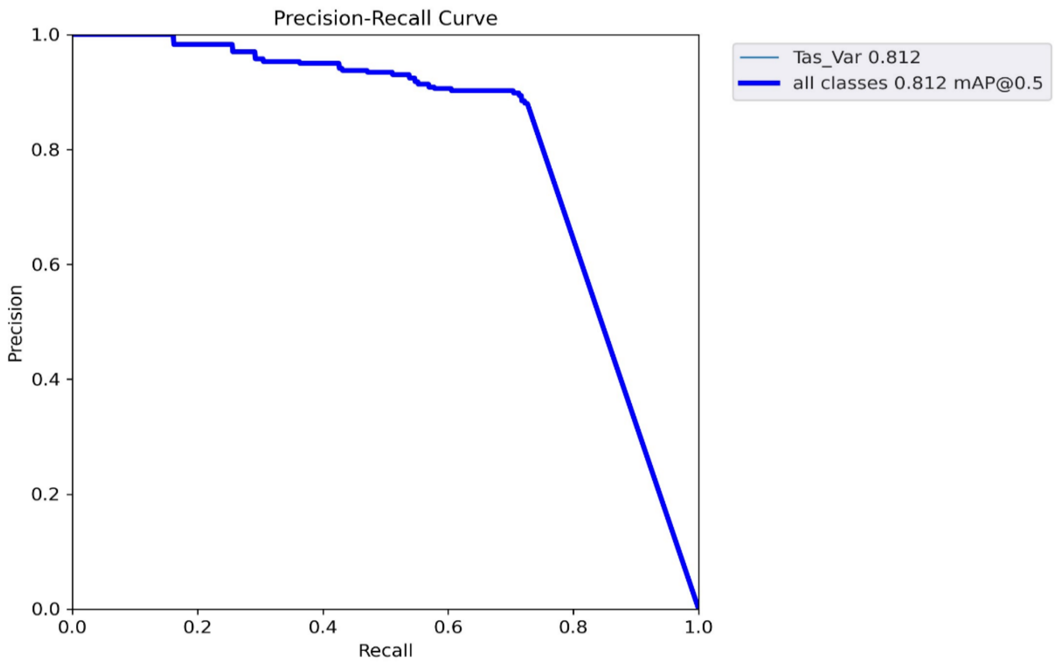


Figure 7.48. Precision-Sensitivity curve for D2 dataset (Test phase).

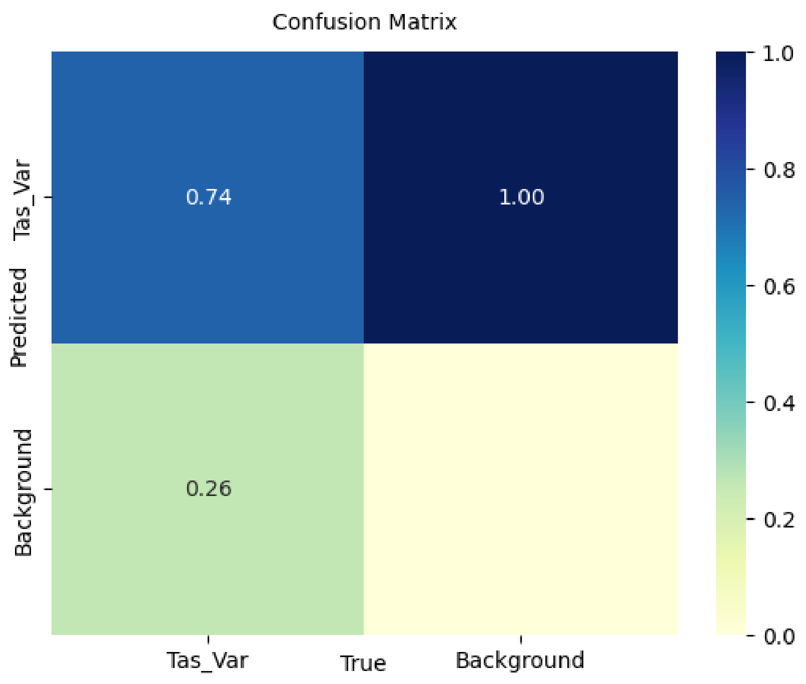


Figure 7.49. Confusion matrix for D2 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the Adam optimizer on D3 dataset are shown in Figures 7.50-7.52

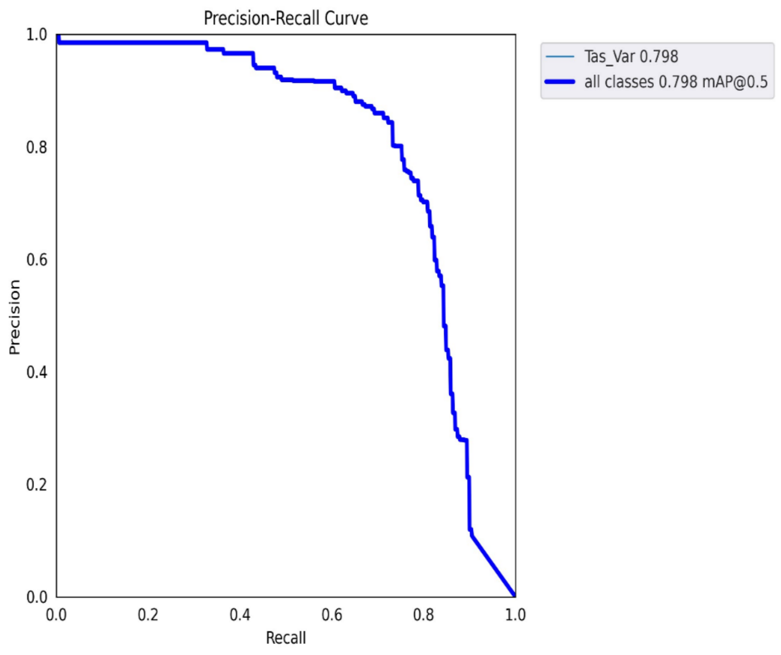


Figure 7.50. Precision-Sensitivity curve for D3 dataset (Train Phase).

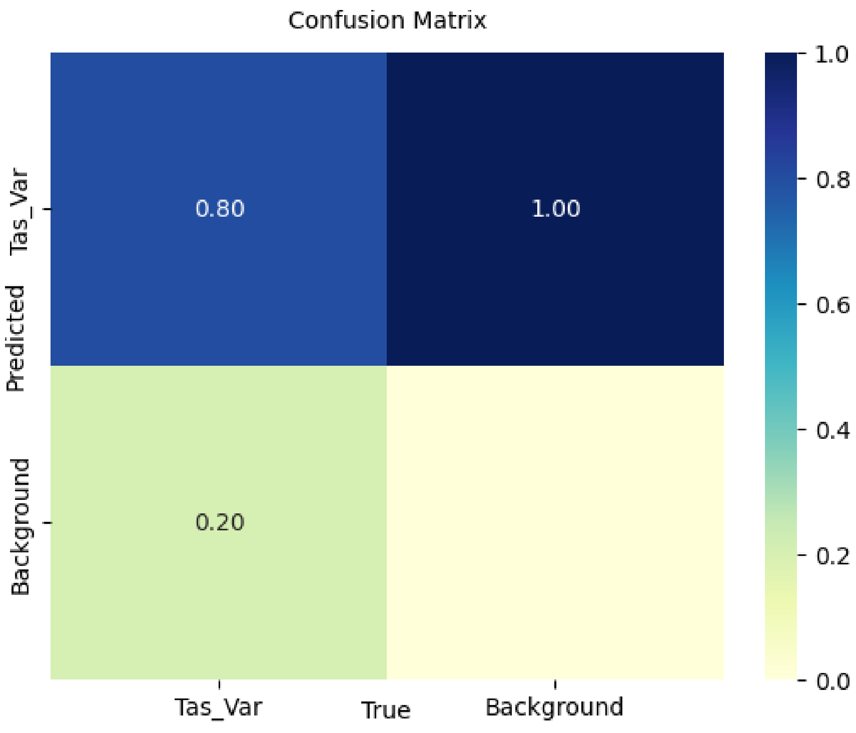


Figure 7.51. Confusion matrix for D3 dataset (Train Phase).

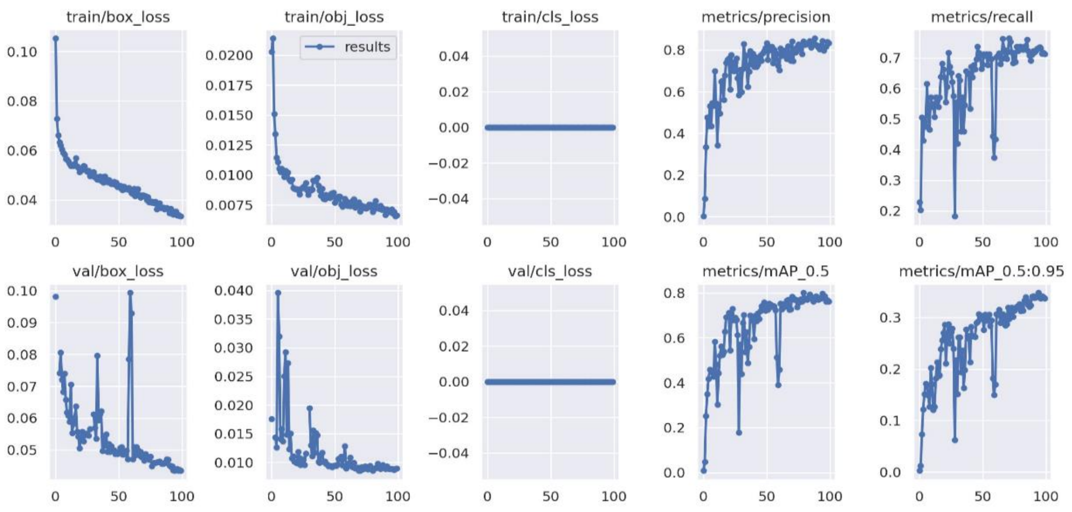


Figure 7.52. Train and validation losses for D3 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the Adam optimizer on D3 dataset are shown in Figures 7.53, and 7.54, respectively.

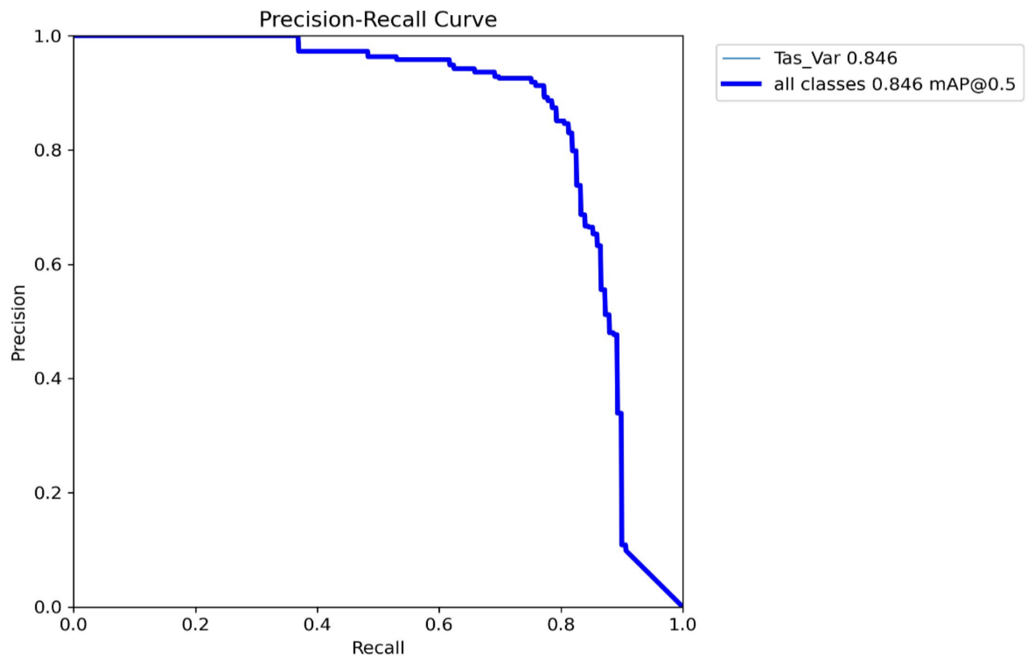


Figure 7.53. Precision-Sensitivity curve for D3 dataset (Test phase).

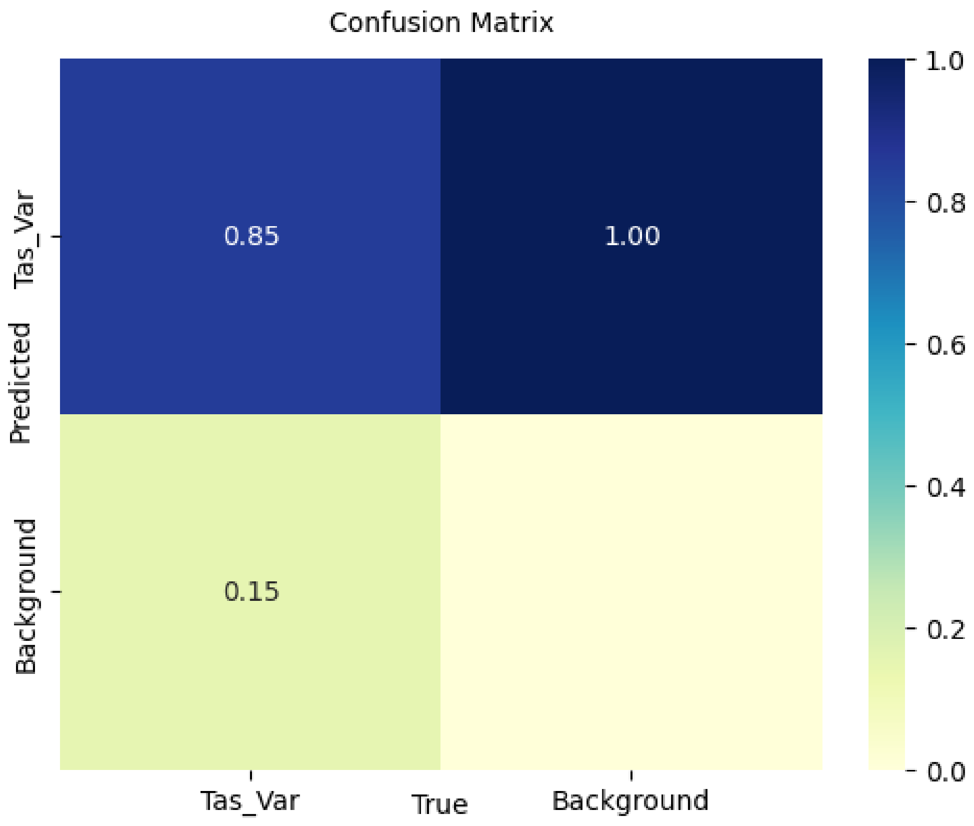


Figure 7.54. Confusion matrix for D3 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v5 with the Adam optimizer on D4 dataset are shown in Figures 7.55-7.57.

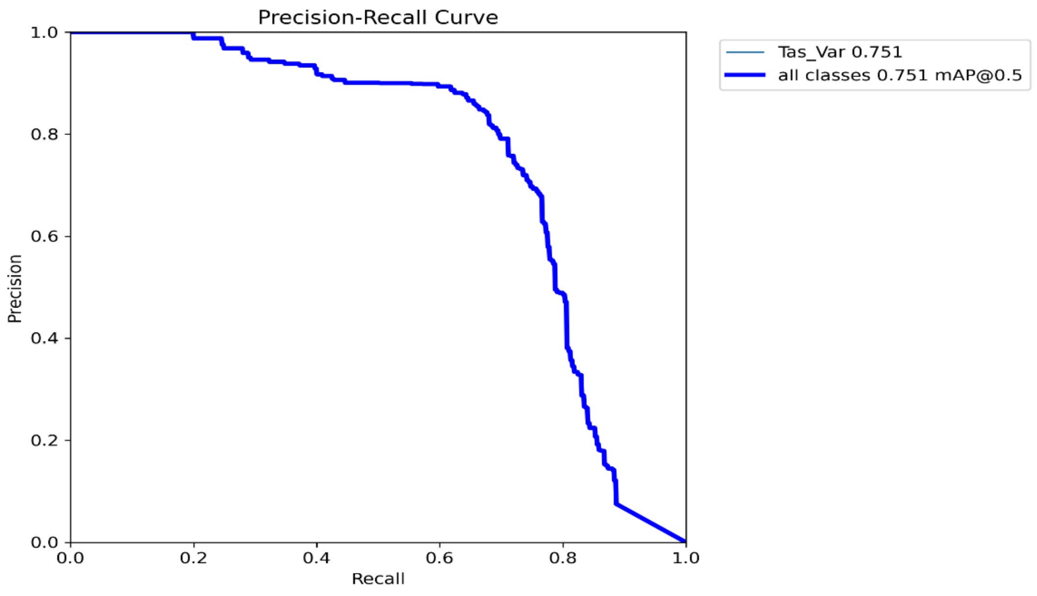


Figure 7.55. Precision-Sensitivity curve for D4 dataset (Train Phase).

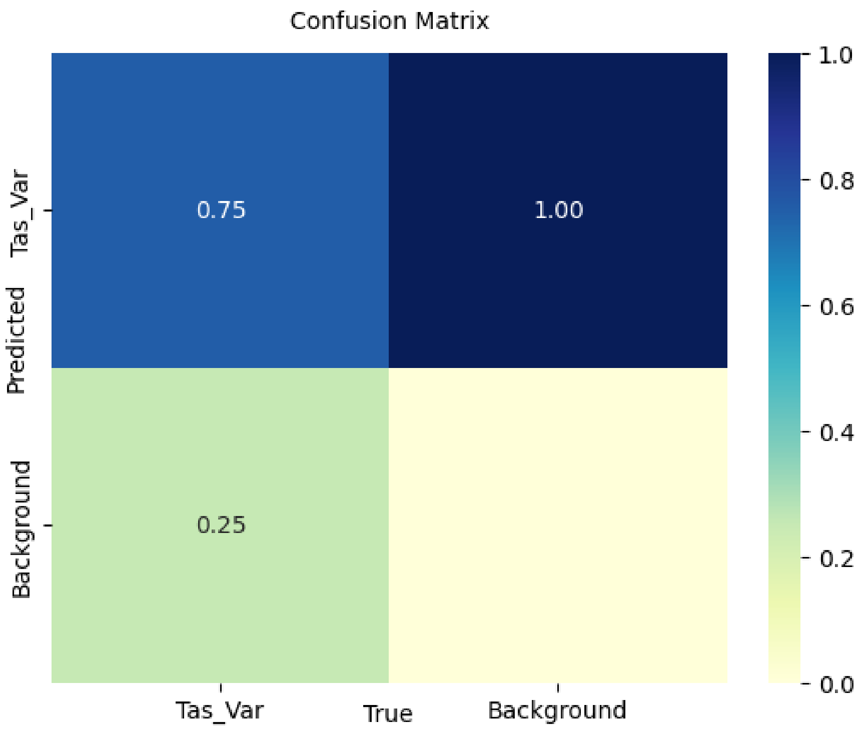


Figure 7.56. Confusion matrix for D4 dataset (Train Phase).

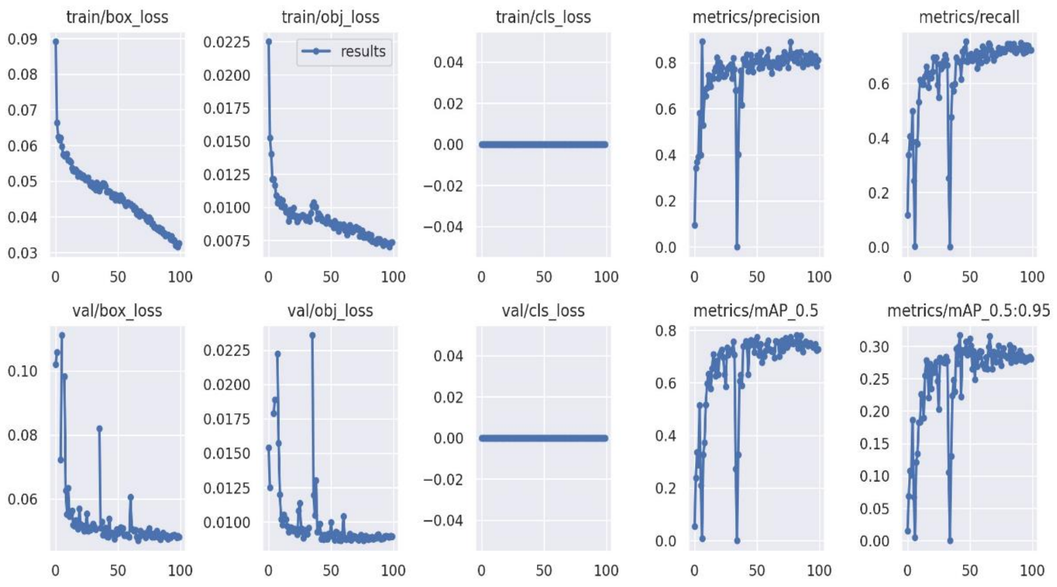


Figure 7.57. Train and validation losses for D4 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v5 with the Adam optimizer on D4 dataset are shown in Figures 7.58, and 7.59, respectively.

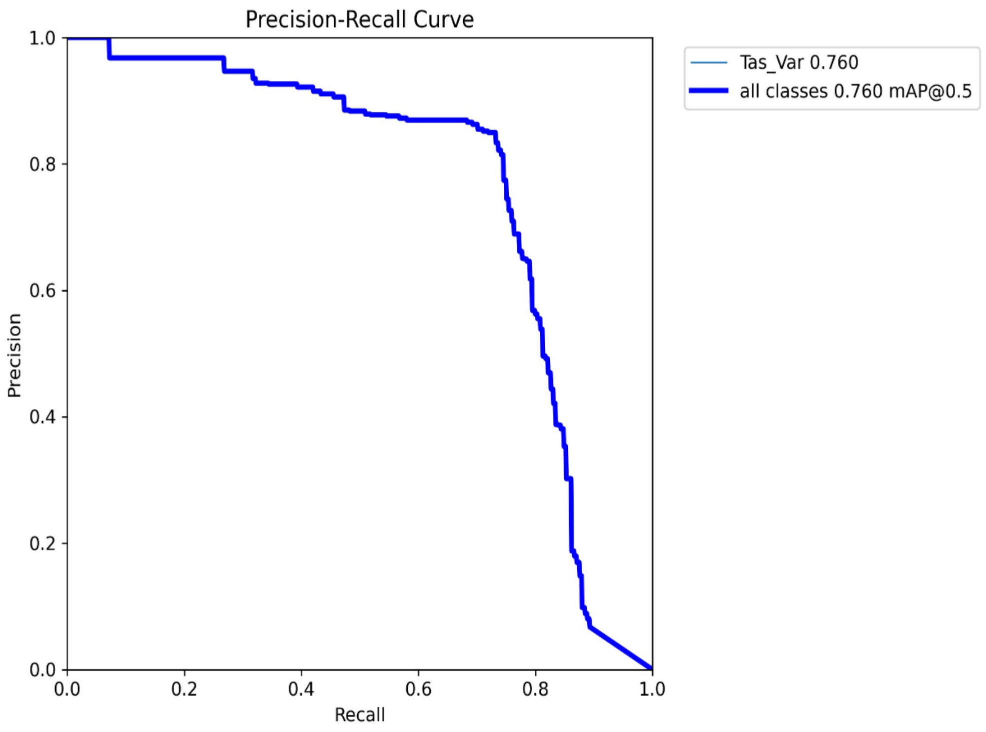


Figure 7.58. Precision-Sensitivity curve for D4 dataset (Test phase).

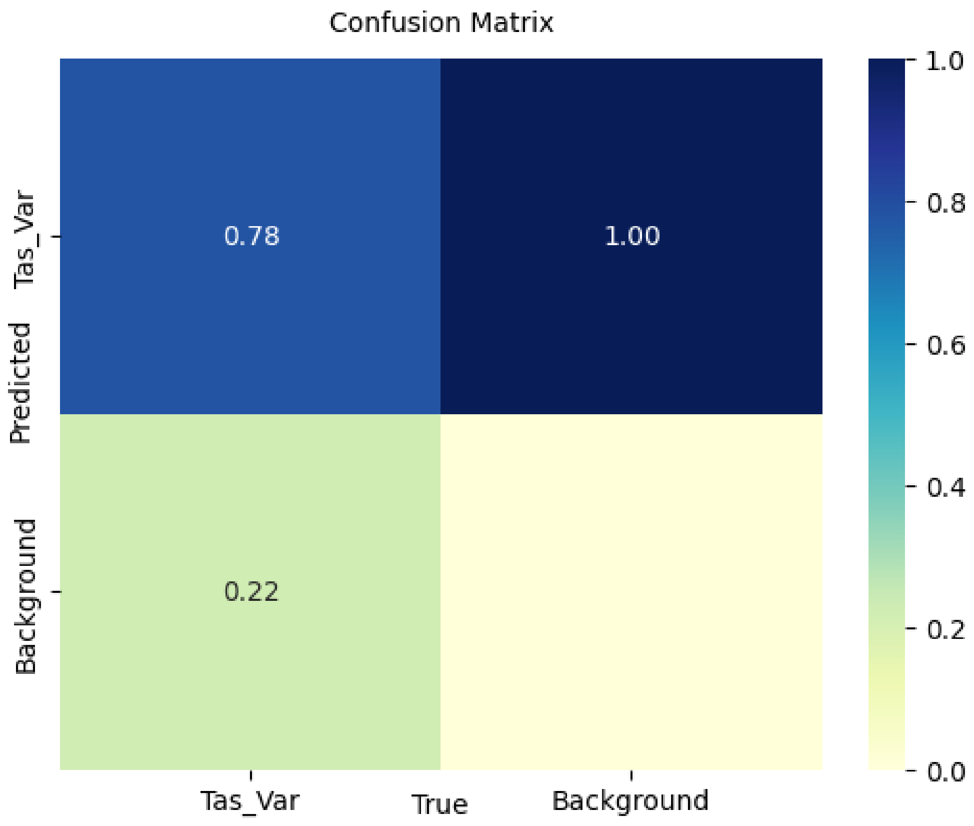


Figure 7.59. Confusion matrix for D4 dataset (Test phase).

Table 7.9 shows a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, and mAP in both of Train and Test phases for YOLO v5 model with Adam optimizer.

Table 7.9. Train and test results of the model.

	Train				**Test**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	75.0	71.0	80.0	75.0	75.0	74.0	85.0	78.0
Precision (%)	84.9	79.1	84.3	81.9	77.6	88.0	89.7	83.3

Sensitivity (%)	68.8	64.1	73.1	68.3	65.9	72.6	77.2	73.2
F1 score (%)	76.0	70.8	78.3	74.4	71.2	79.5	82.9	77.9
mAP (0.5) (%)	73.8	65.9	79.8	75.1	68.4	81.2	84.6	76.0
mAP (0.5:0.95) (%)	30.1	26.2	34.8	31.7	25.2	40.5	39.0	33.6

Table 7.10 and Table 7.11 show a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, and mAP in both of Train and Test phases for YOLO v5 model between SGD and Adam optimizers.

Table 7.10. Train results of the model between SGD and Adam.

	SGD				**Adam**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	76.0	73.0	82.0	80.0	75.0	71.0	80.0	75.0
Precision (%)	87.0	77.3	85.7	80.2	84.9	79.1	84.3	81.9
Sensitivity (%)	65.9	68.1	75.7	76.0	68.8	64.1	73.1	68.3
F1 score (%)	74.9	72.4	80.3	78.0	76.0	70.8	78.3	74.4
mAP (0.5) (%)	73.5	67.5	79.3	79.0	73.8	65.9	79.8	75.1
mAP (0.5:0.95) (%)	29.0	26.2	35.0	32.4	30.1	26.2	34.8	31.7

Table 7.11. Test results of the model between SGD and Adam.

	SGD				Adam			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	73.0	76.0	83.0	79.0	75.0	74.0	85.0	78.0
Precision (%)	73.3	85.6	85.3	78.1	77.6	88.0	89.7	83.3
Sensitivity (%)	66.5	72.2	77.7	76.8	65.9	72.6	77.2	73.2
F1 score (%)	69.7	78.3	81.3	77.4	71.2	79.5	82.9	77.9
mAP (0.5) (%)	63.4	79.9	79.8	75.4	68.4	81.2	84.6	76.0
mAP (0.5:0.95) (%)	25.2	39.0	37.2	31.5	25.2	40.5	39.0	33.6

7.2.1.2. Experimental results of YOLO v5

Visual representations of the detection results achieved with the YOLO v5 during the test phase of kidney stones in the datasets alongside with the ground truth bounding boxes are given in Figures 7.60-7.63.

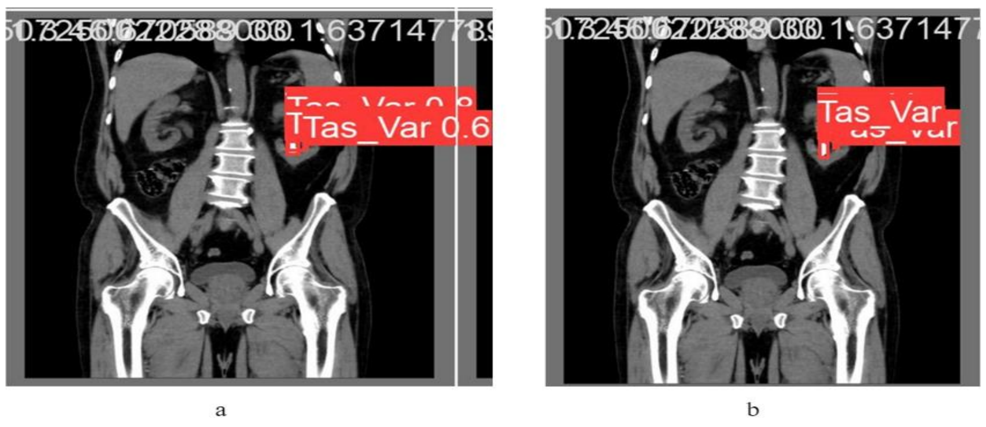


Figure 7.60. Evaluation of kidney stones. (a) Model's predictions (b) ground truth bounding boxes.

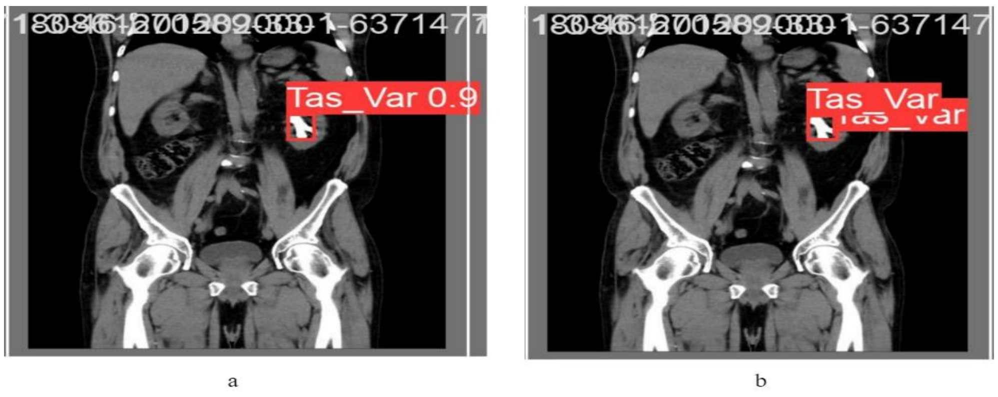


Figure 7.61. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

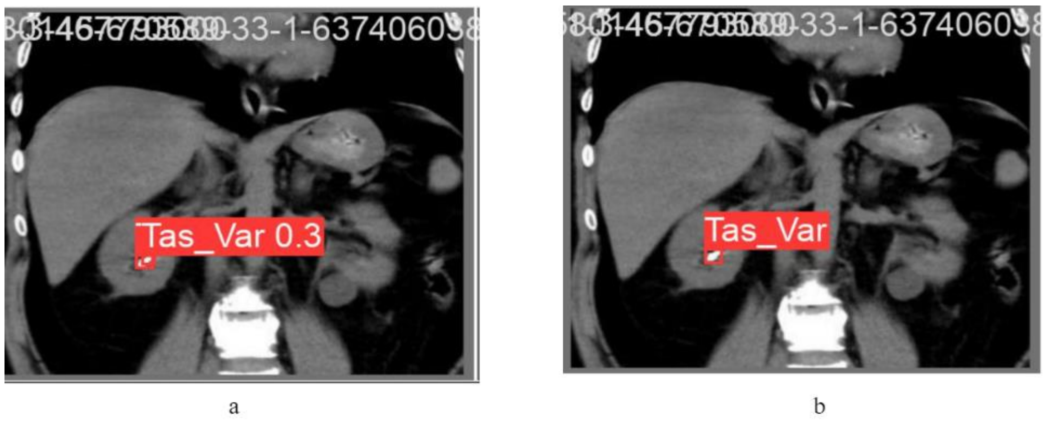


Figure 7.62. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

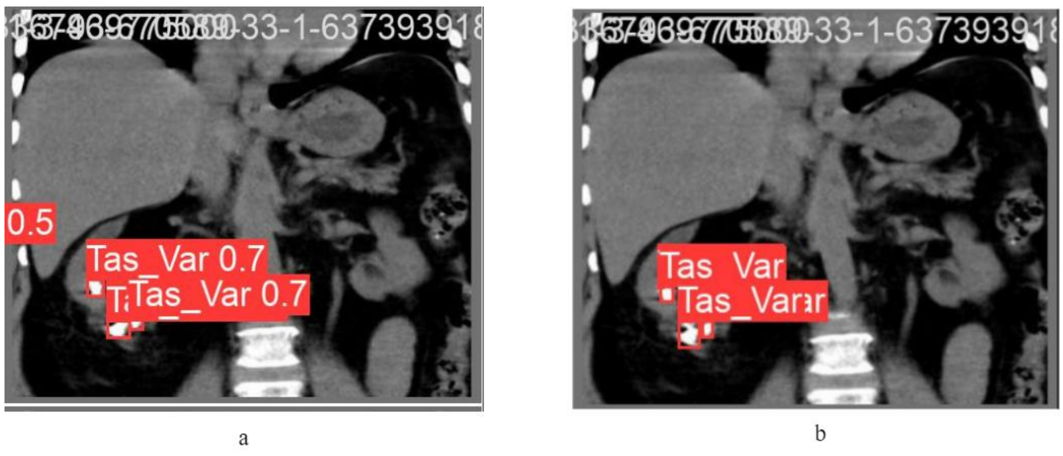


Figure 7.63. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

7.2.2. YOLO v7

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the SGD optimizer on D1 dataset are shown in Figures 7.64-7.66.

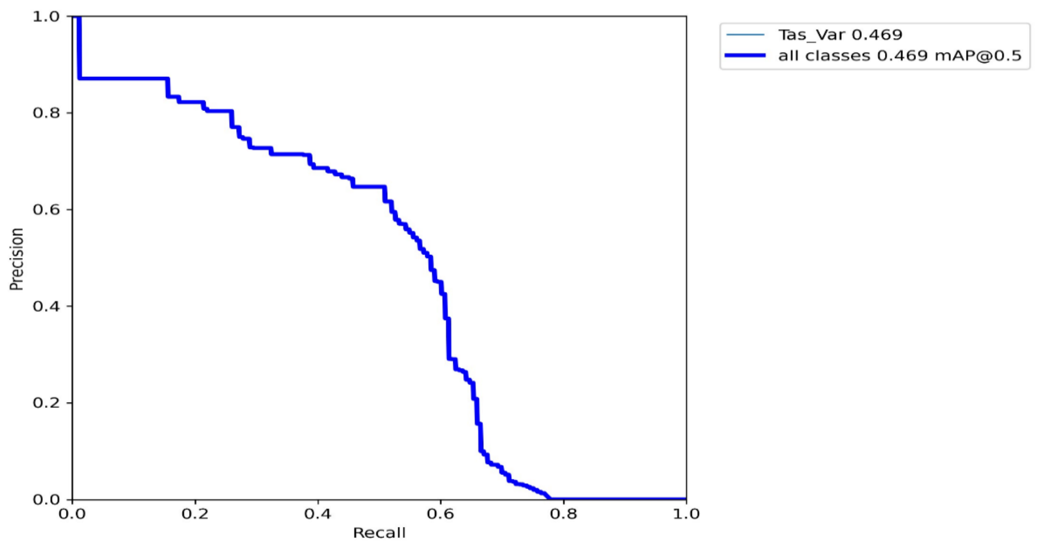


Figure 7.64. Precision-Sensitivity curve for D1 dataset (Train Phase).

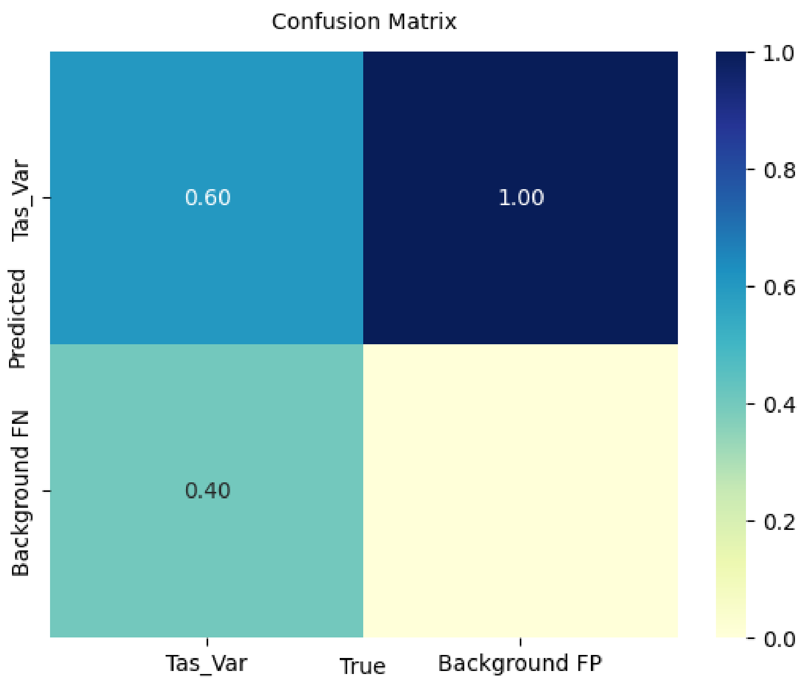


Figure 7.65. Confusion matrix for D1 dataset (Train Phase).

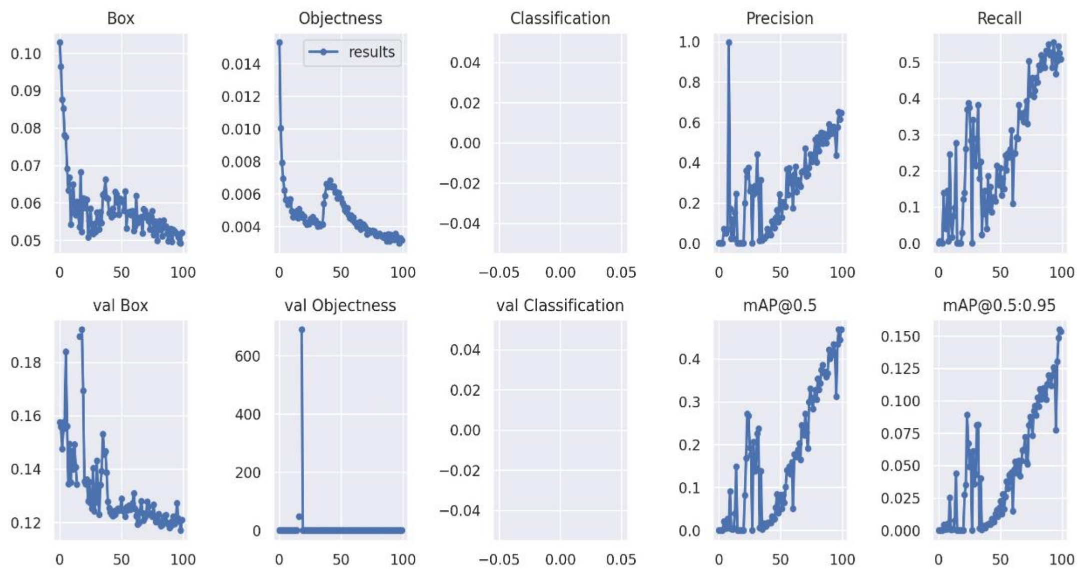


Figure 7.66. Train and validation losses for D1 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the SGD optimizer on D1 dataset are shown in Figures 7.67, and 7.68, respectively.

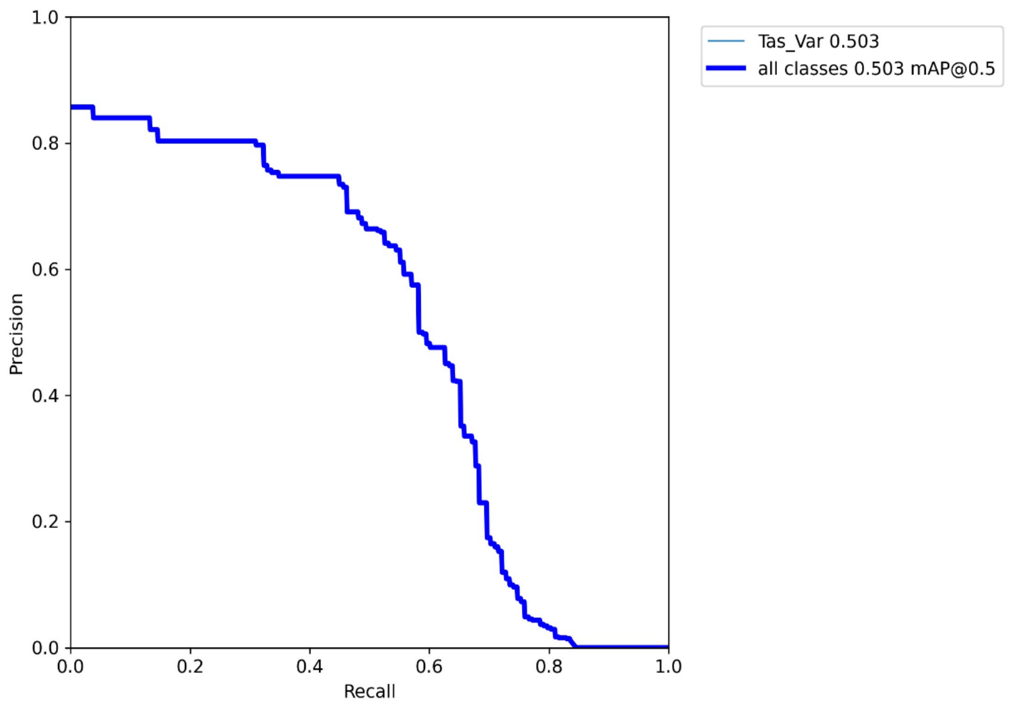


Figure 7.67. Precision-Sensitivity curve for D1 dataset (Test phase).

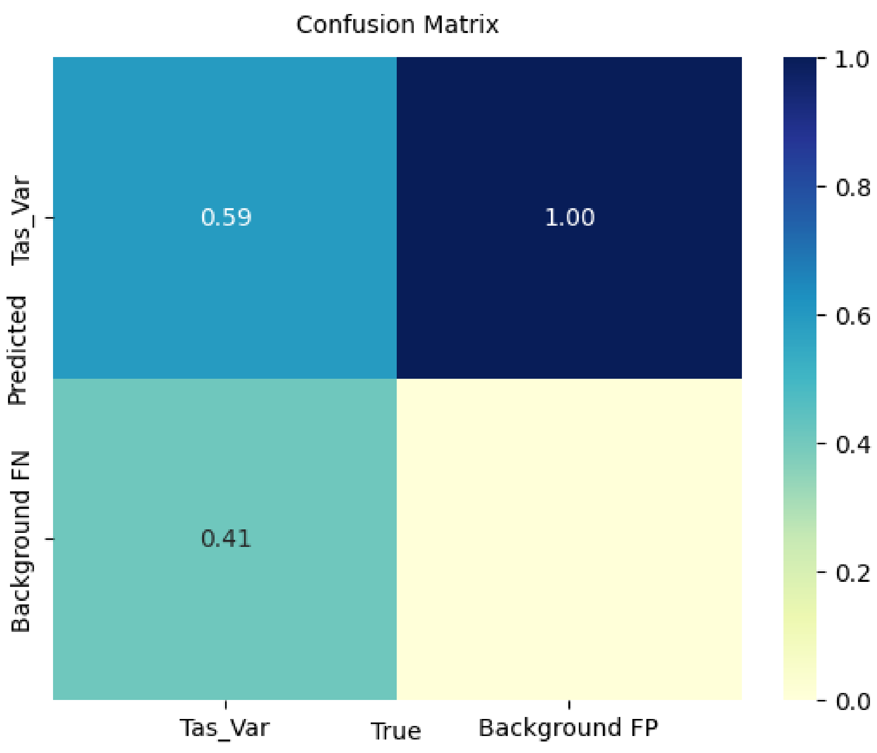


Figure 7.68. Confusion matrix for D1 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the SGD optimizer on D2 dataset are shown in Figures 7.69-7.71.

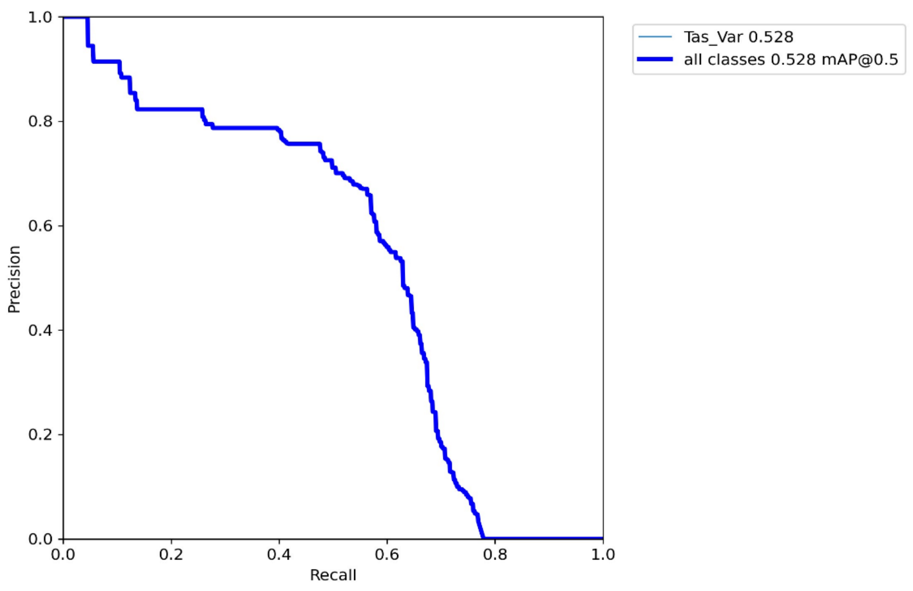


Figure 7.69. Precision-Sensitivity curve for D2 dataset (Train Phase).

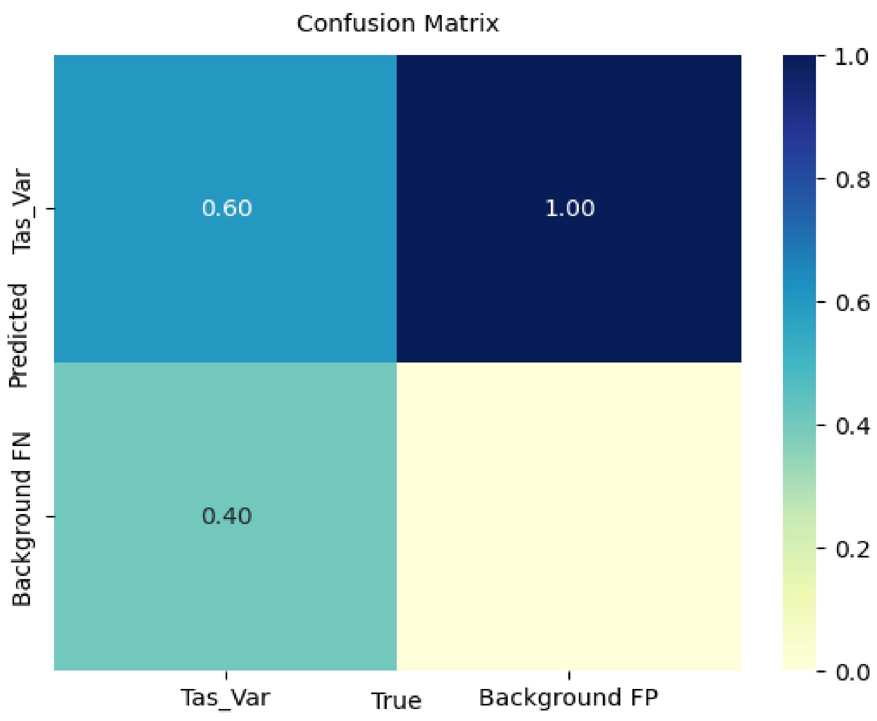


Figure 7.70. Confusion matrix for D2 dataset (Train Phase).

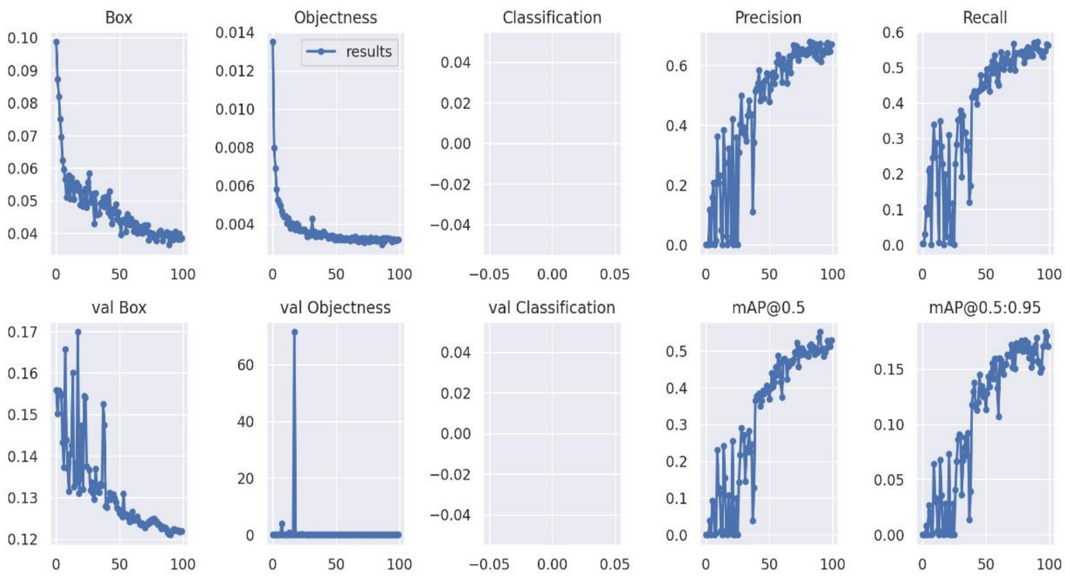


Figure 7.71. Train and validation losses for D2 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the SGD optimizer on D2 dataset are shown in Figures 7.72, and 7.73, respectively.

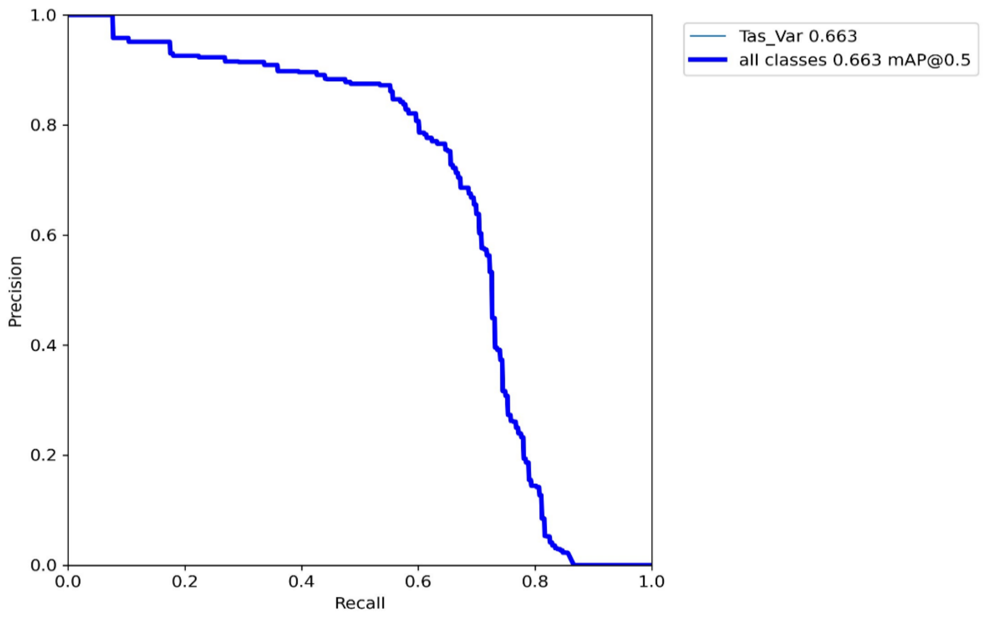


Figure 7.72. Precision-Sensitivity curve for D2 dataset (Test phase).

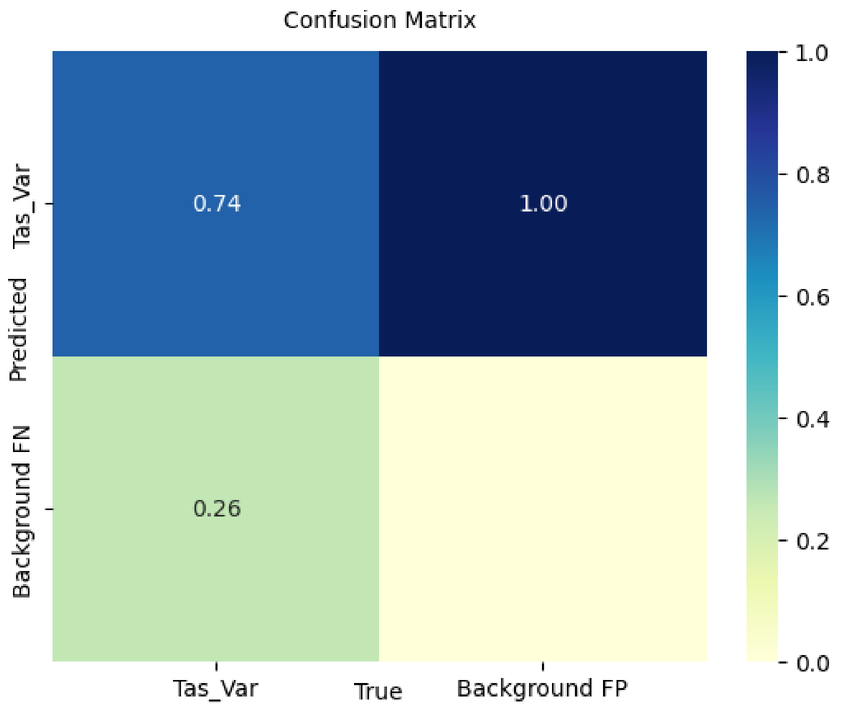


Figure 7.73. Confusion matrix for D2 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the SGD optimizer on D3 dataset are shown in Figures 7.74-7.76.

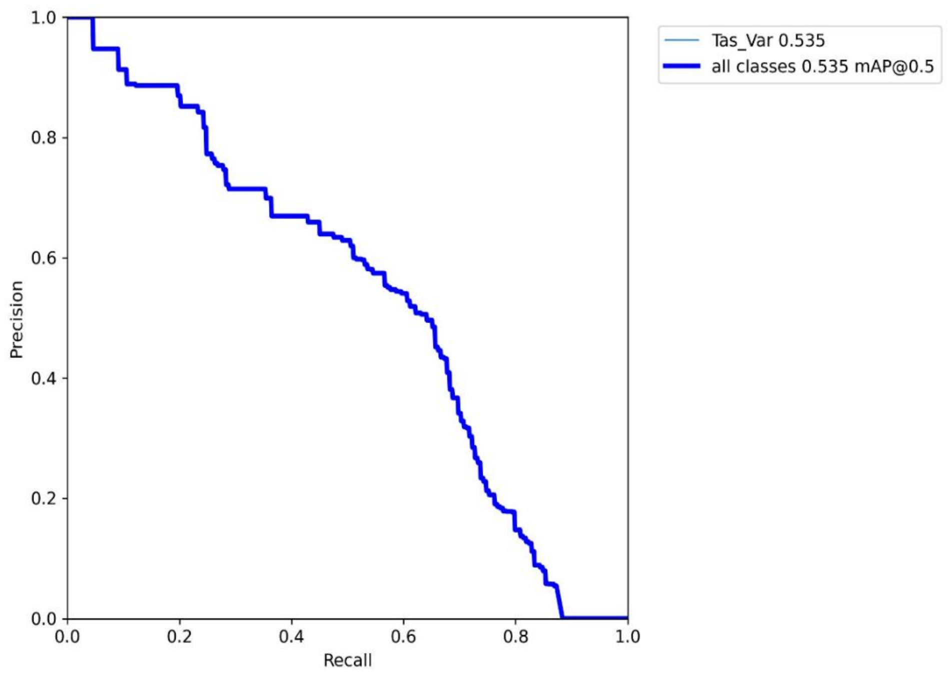


Figure 7.74. Precision-Sensitivity curve for D3 dataset (Train phase).

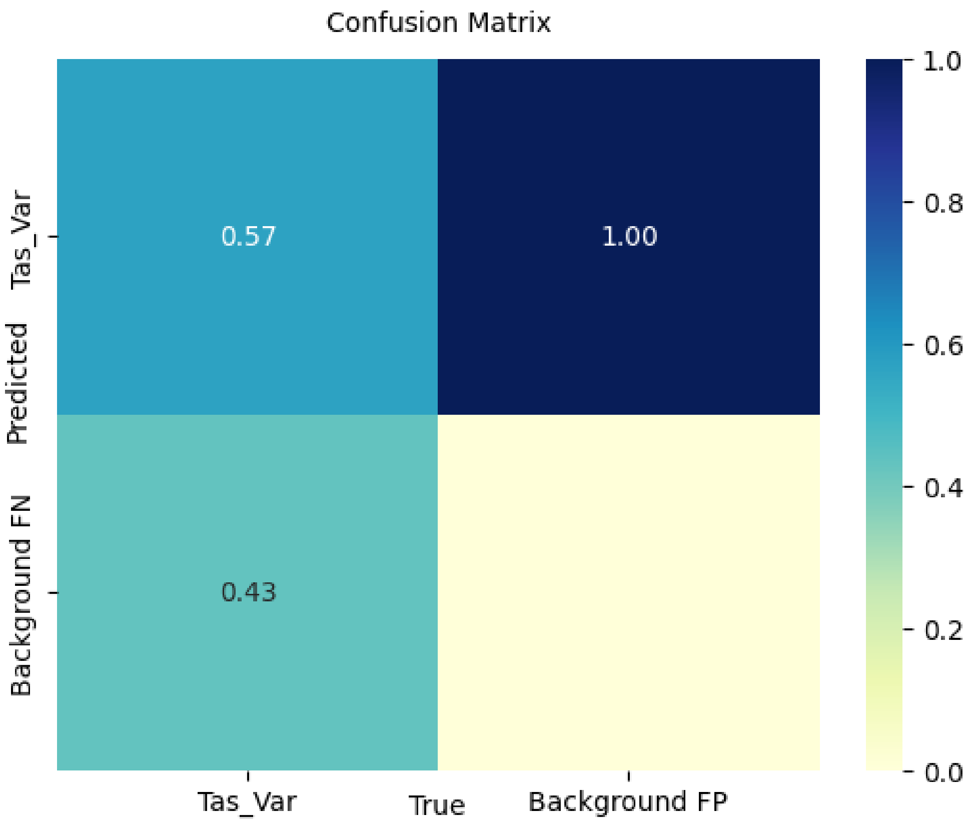


Figure 7.75. Confusion matrix for D3 dataset (Train phase).

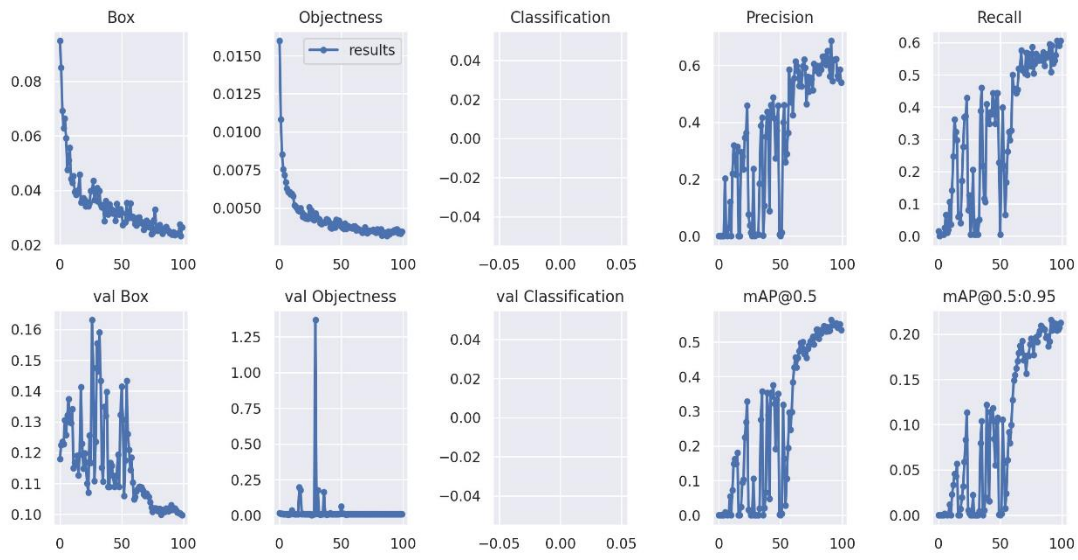


Figure 7.76. Train and validation losses for D3 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the SGD optimizer on D3 dataset are shown in Figures 7.77, and 7.78, respectively.

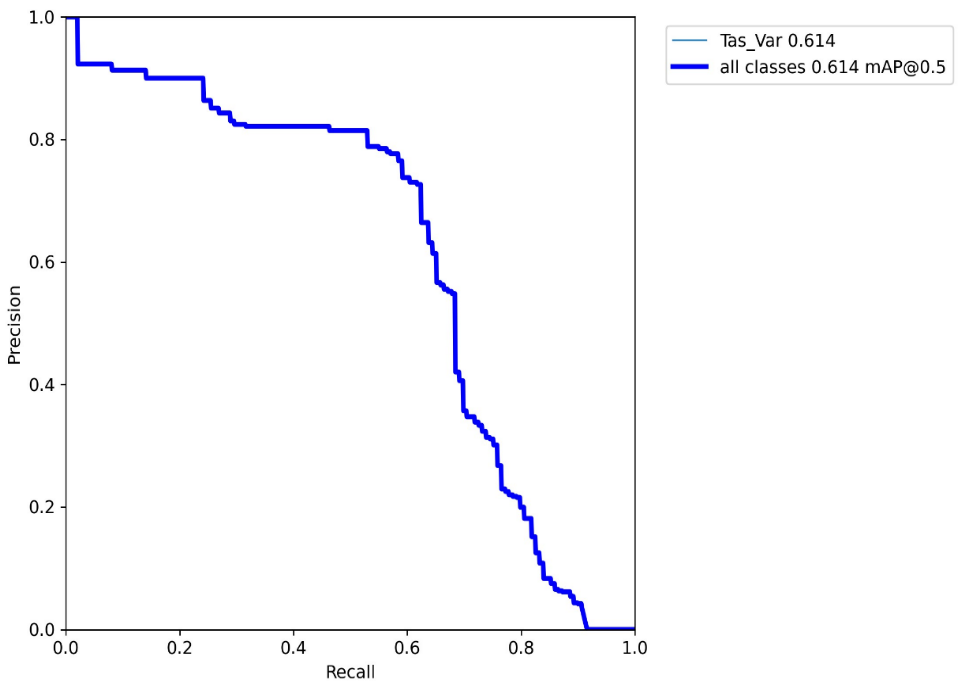


Figure 7.77. Precision-Sensitivity curve for D3 dataset (Test phase).

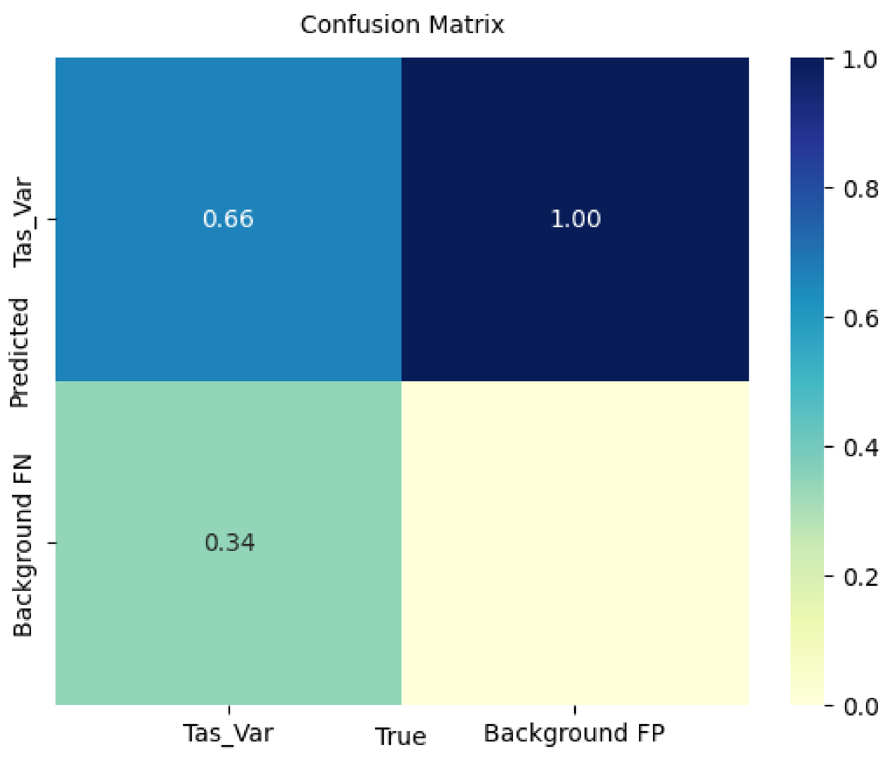


Figure 7.78. Confusion matrix for D3 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the SGD optimizer on D4 dataset are shown in Figures 7.79-7.81.

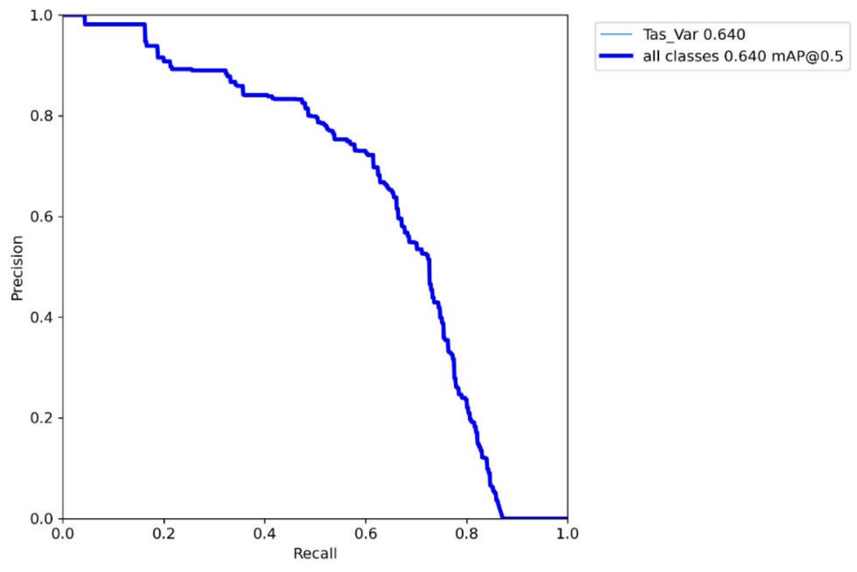


Figure 7.79. Precision-Sensitivity curve for D4 dataset (Train Phase).

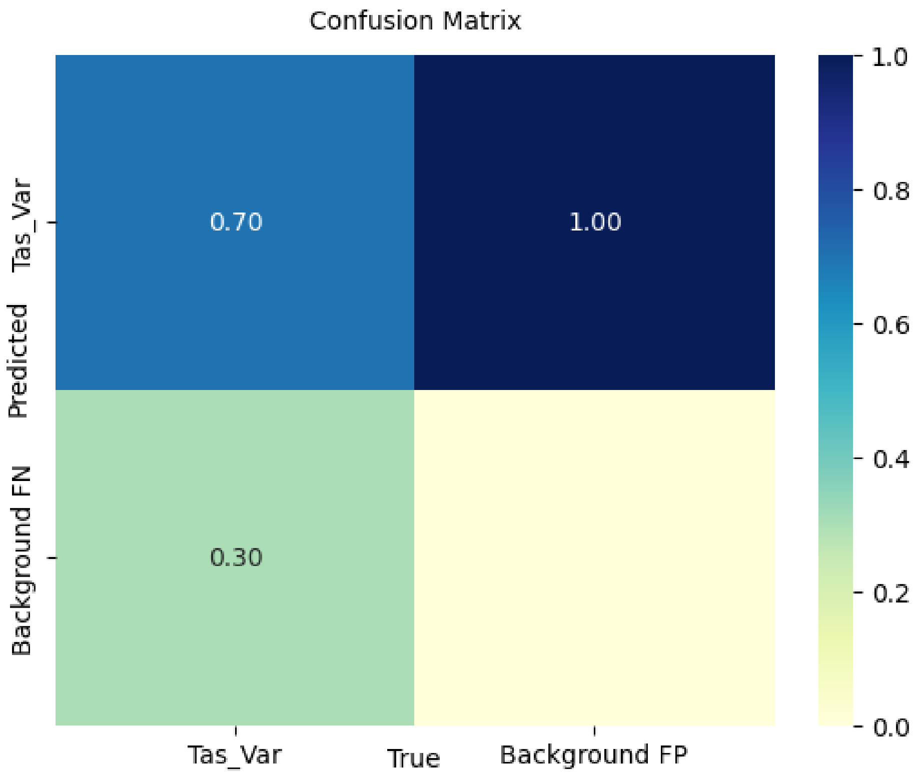


Figure 7.80. Confusion matrix for D4 dataset (Train Phase).

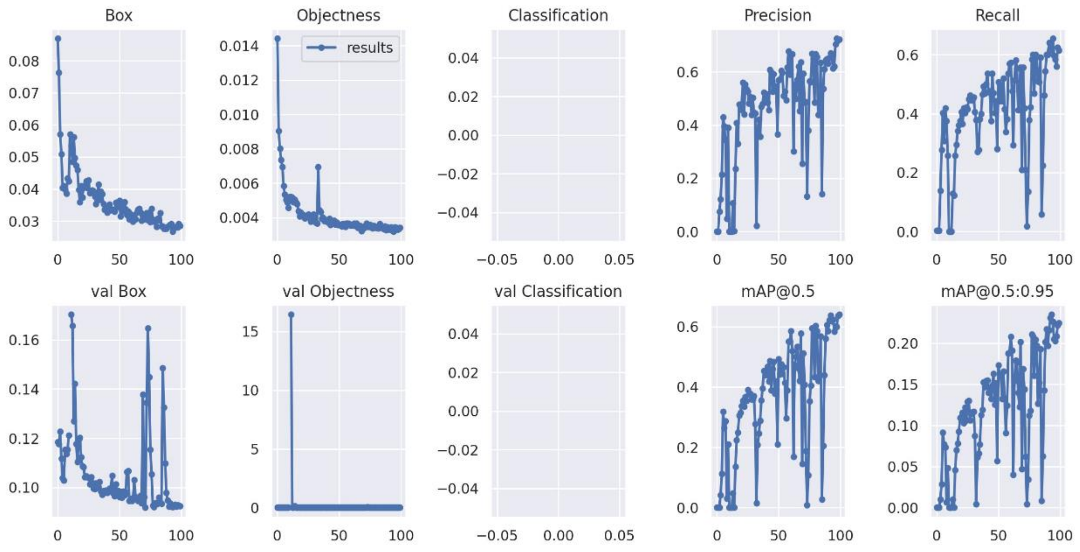


Figure 7.81. Train and validation losses for D4 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the SGD optimizer on D4 dataset are shown in Figures 7.82, and 7.83, respectively.

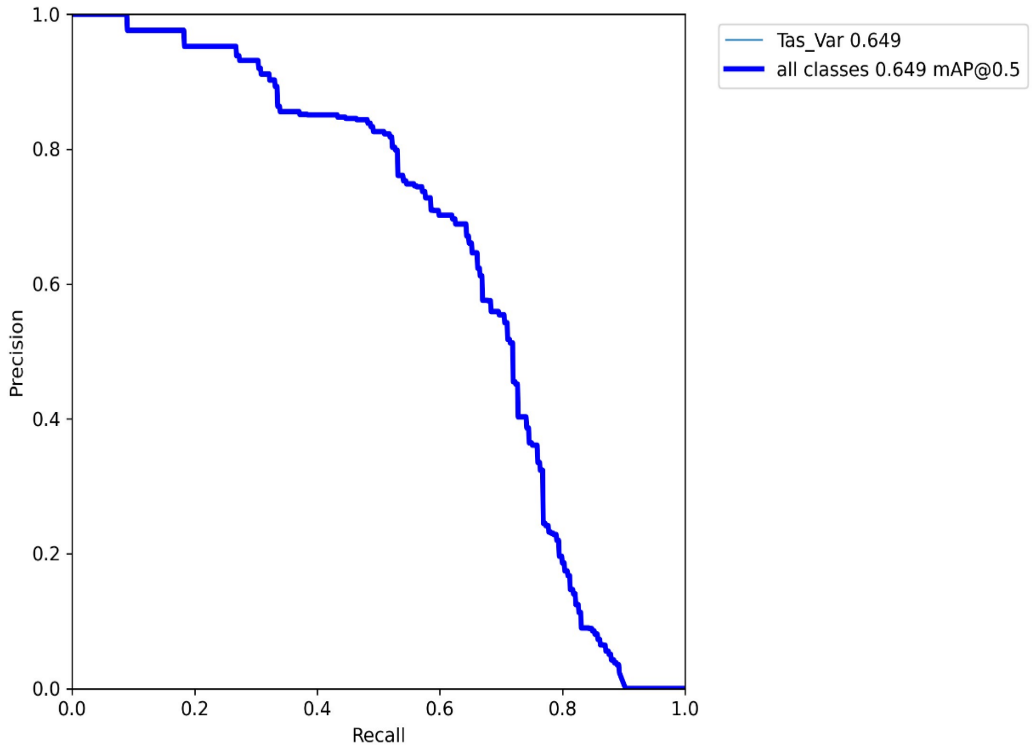


Figure 7.82. Precision-Sensitivity curve for D4 dataset (Test phase).

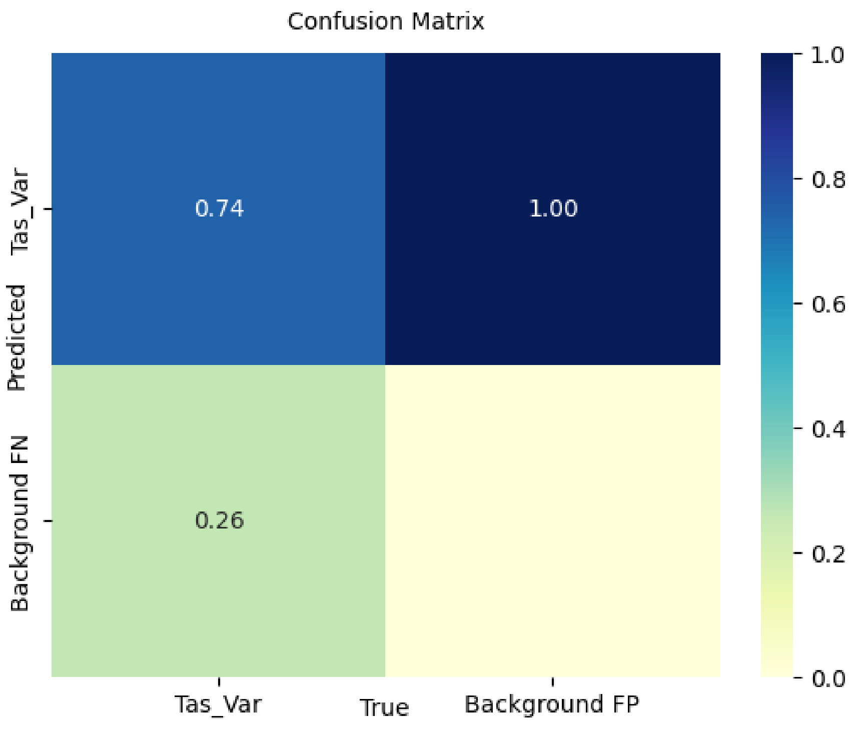


Figure 7.83. Confusion matrix for D4 dataset (Test phase).

Table 7.12 shows a summary of the obtained results for the four datasets in respect to Accuracy and mAP in both of Train and Test phases for YOLO v7 model with SGD optimizer. Table 7.13 show a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, mAP in Test phase for YOLO v7 model with SGD optimizer.

Table 7.12. Summary of the obtained results in Train and Test.

	Train				**Test**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	60.0	60.0	57.0	70.0	59.0	74.0	66.0	74.0
mAP (0.5) (%)	46.9	52.8	53.5	64.0	48.3	66.3	61.4	64.9

Table 7.13. Summary of the obtained results in Test.

	Test			
	D1	**D2**	**D3**	**D4**
Accuracy (%)	59.0	74.0	66.0	74.0
Precision (%)	61.9	76.6	72.6	68.9
Sensitivity (%)	52.5	64.5	62.4	64.3
F1 score (%)	56.8	70.0	67.1	66.5
mAP (0.5) (%)	48.3	66.3	61.4	64.9
mAP (0.5:0.95) (%)	17.3	26.7	26.7	26.1

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the Adam optimizer on D1 dataset are shown in Figures 7.84-7.86.

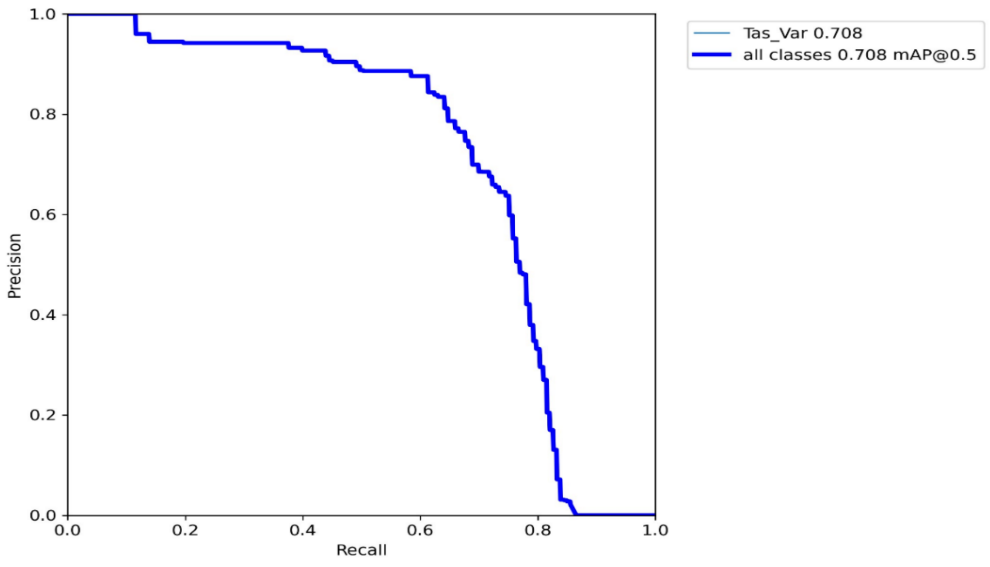


Figure 7.84. Precision-Sensitivity curve for D1 dataset (Train Phase).

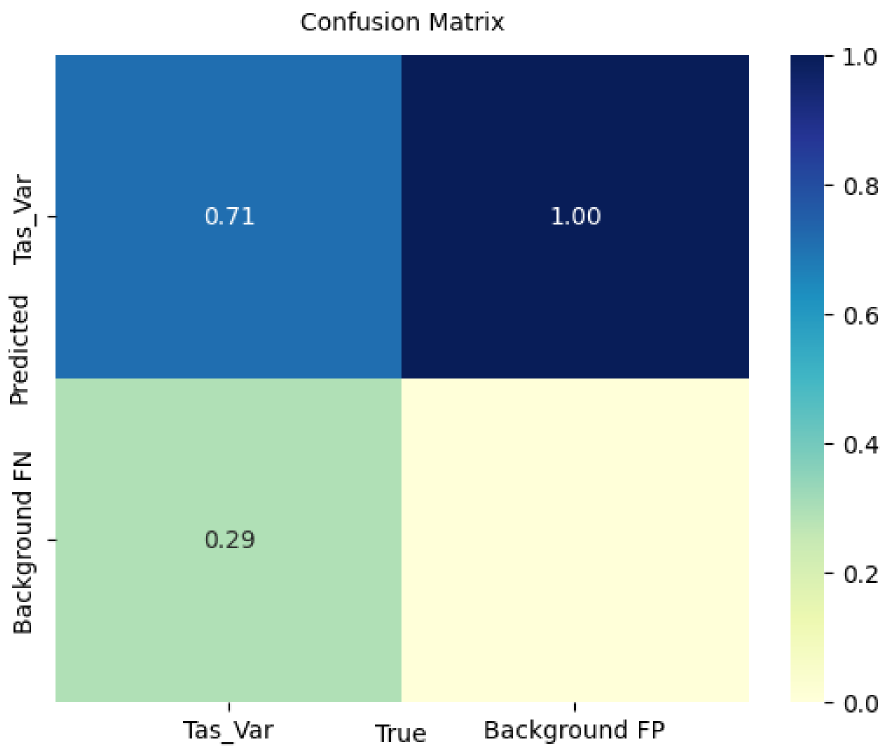


Figure 7.85. Confusion matrix for D1 dataset (Train Phase).

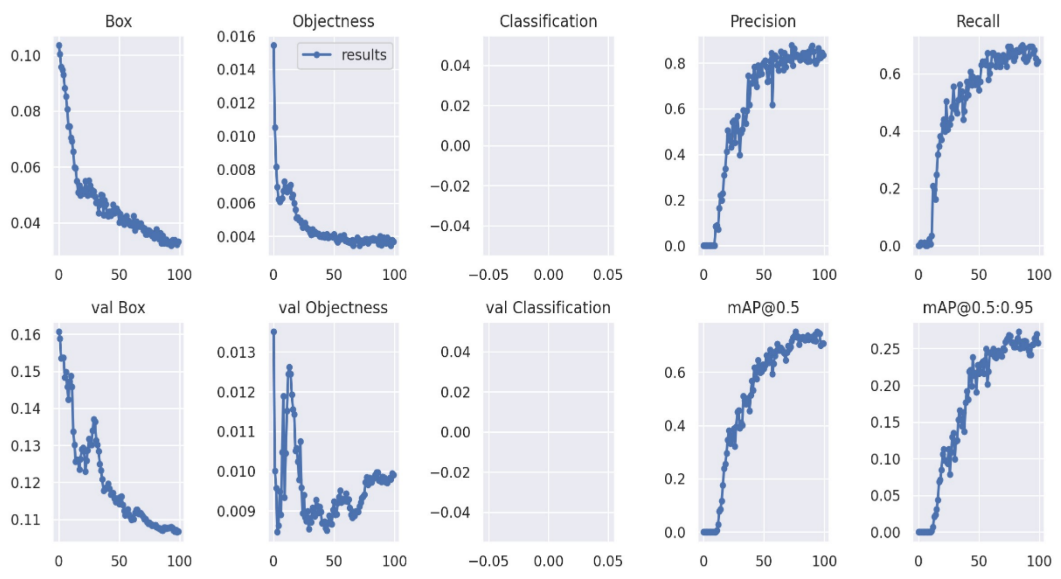


Figure 7.86. Train and validation losses for D1 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the Adam optimizer on D1 dataset are shown in Figures 7.87, and 7.88, respectively.

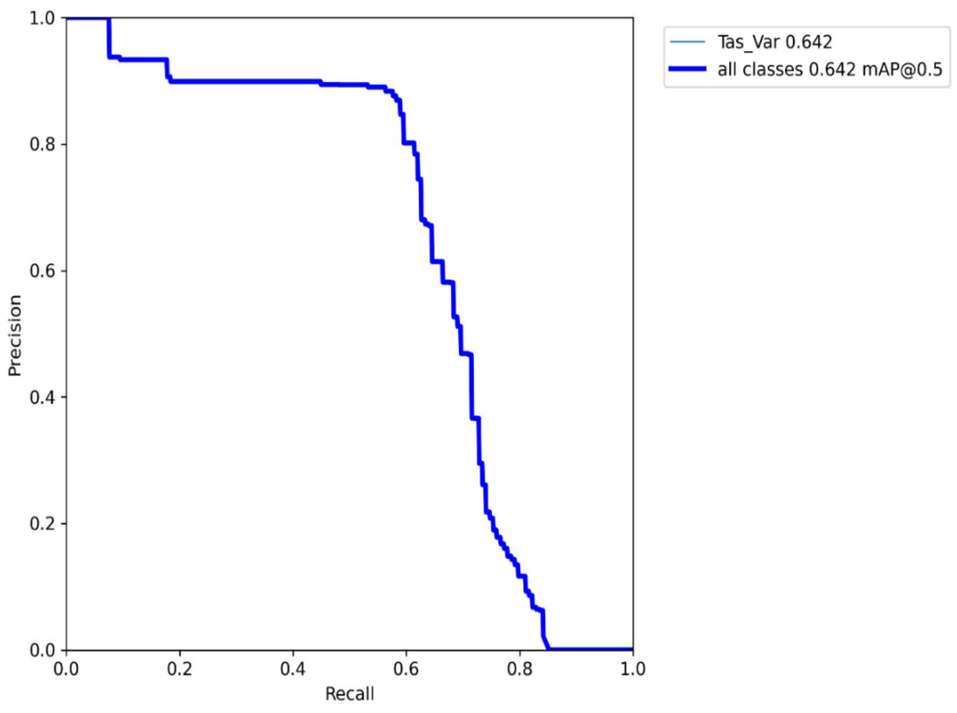


Figure 7.87. Precision-Sensitivity curve for D1 dataset (Test phase).

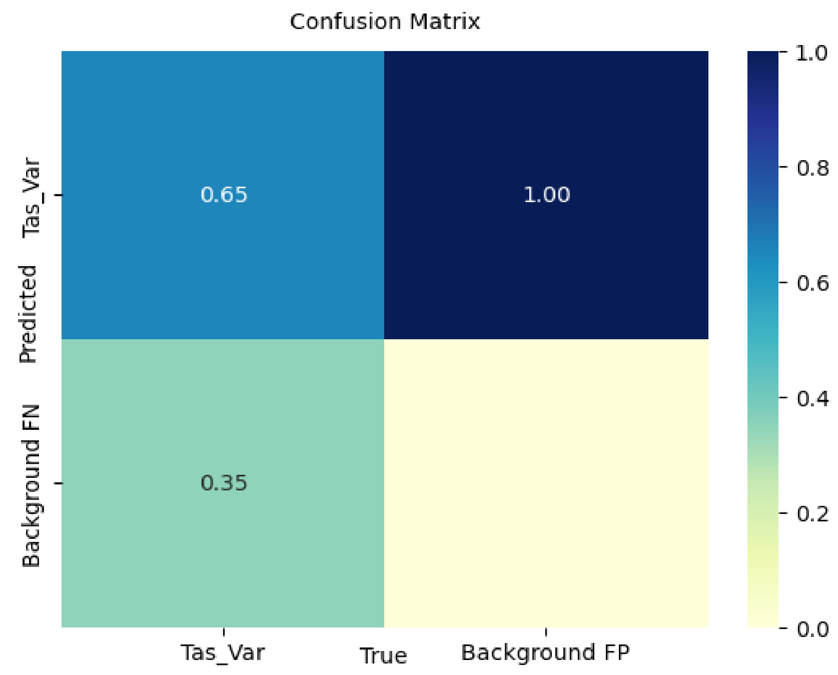


Figure 7.88. Confusion matrix for D1 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the Adam optimizer on D2 dataset are shown in Figures 7.89-7.91.

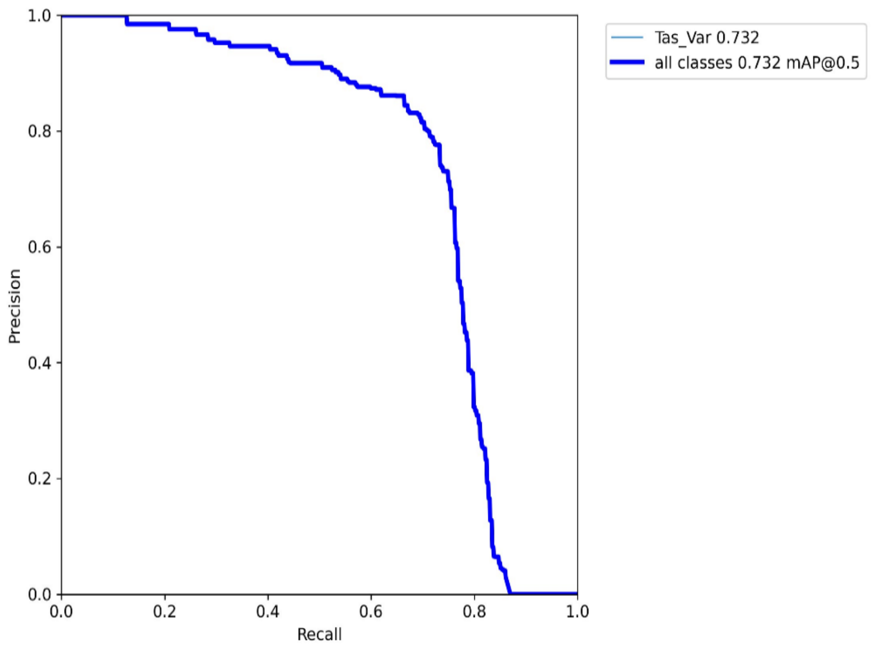


Figure 7.89. Precision-Sensitivity curve for D2 dataset (Train Phase).

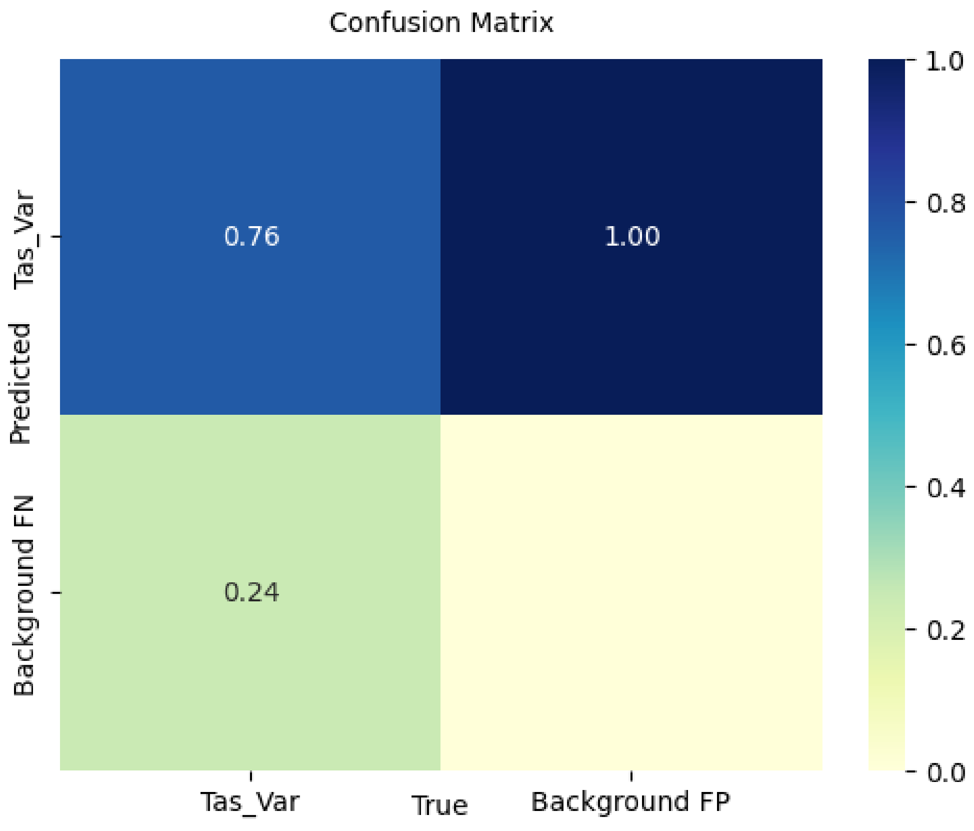


Figure 7.90. Confusion matrix for D2 dataset (Train Phase).

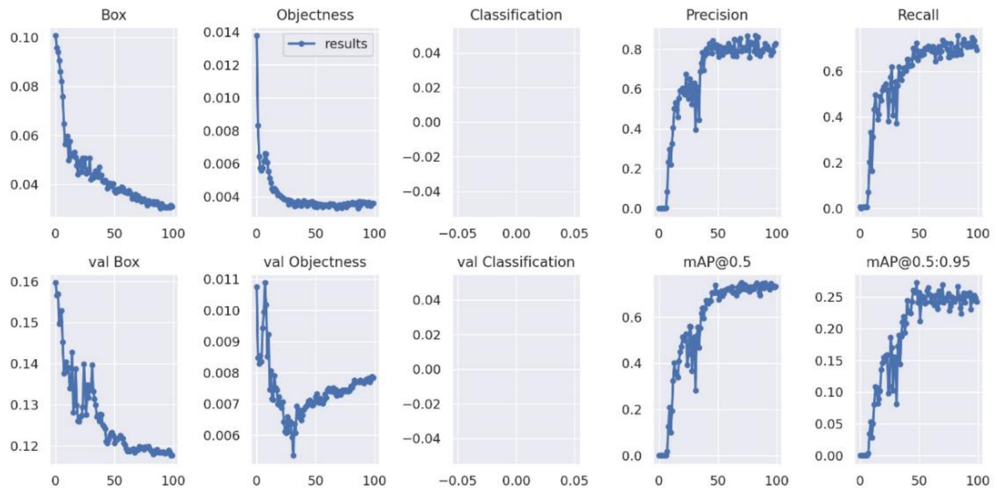


Figure 7.91. Train and validation losses for D2 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the Adam optimizer on D2 dataset are shown in Figures 7.92, and 7.93, respectively.

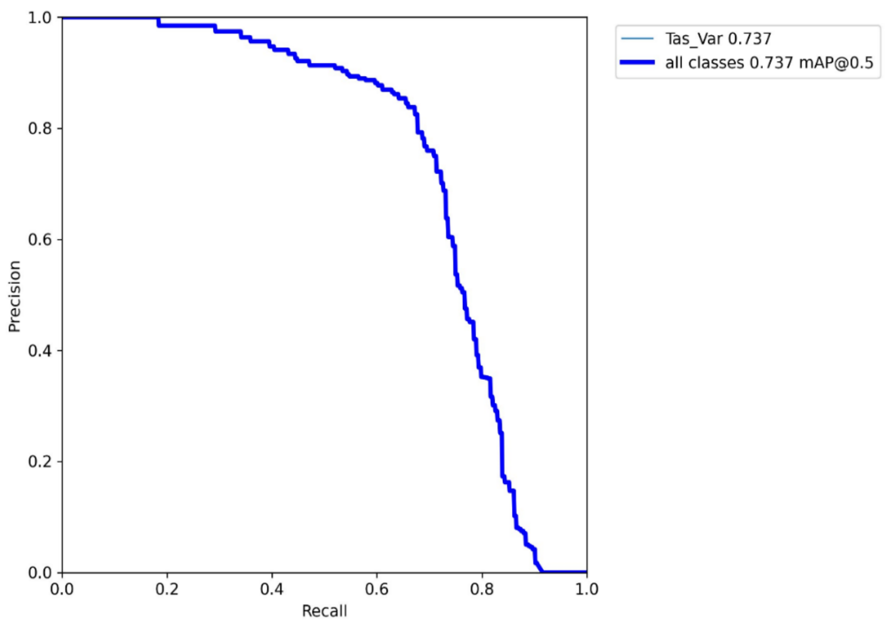


Figure 7.92. Precision-Sensitivity curve for D2 dataset (Test phase).

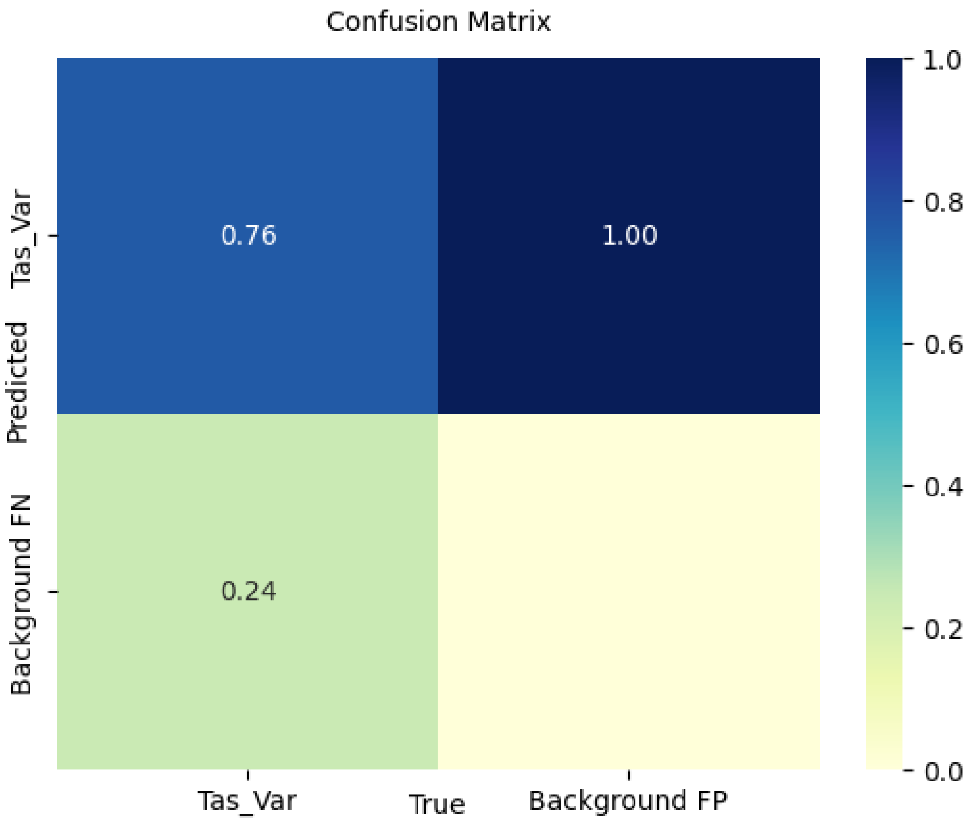


Figure 7.93. Confusion matrix for D2 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the Adam optimizer on D3 dataset are shown in Figures 7.94-7.96.

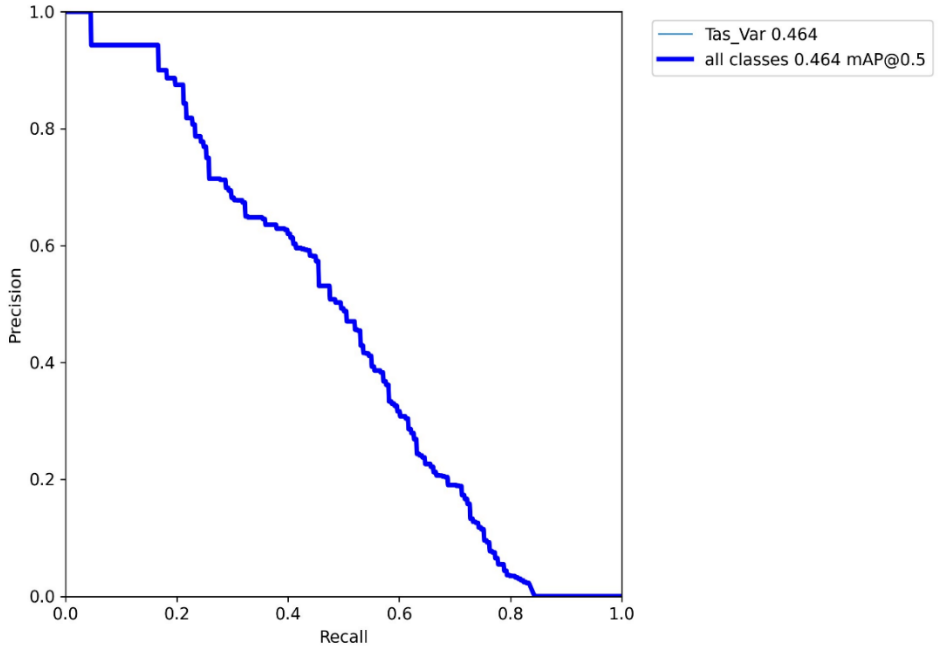


Figure 7.94. Precision-Sensitivity curve for D3 dataset (Train phase).

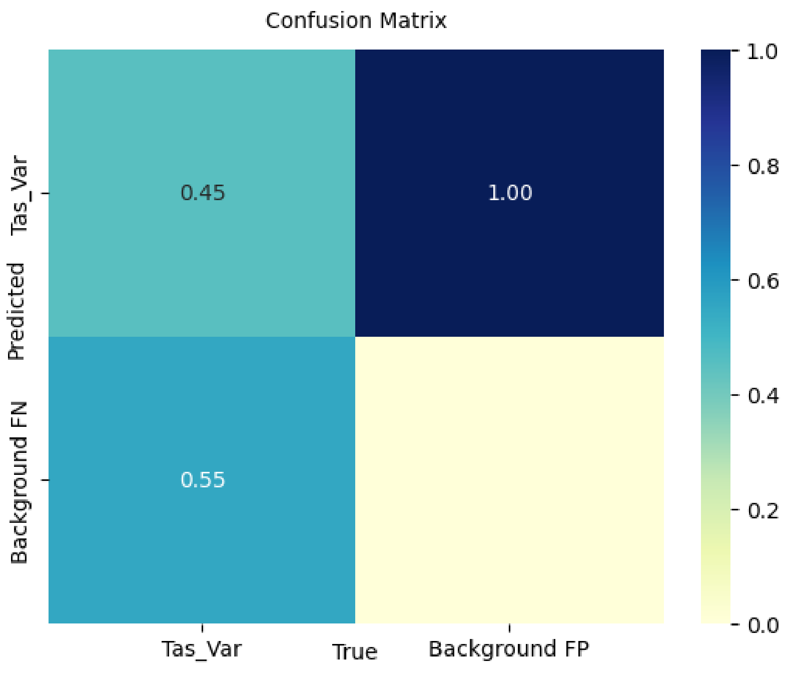


Figure 7.95. Confusion matrix for D3 dataset (Train phase).

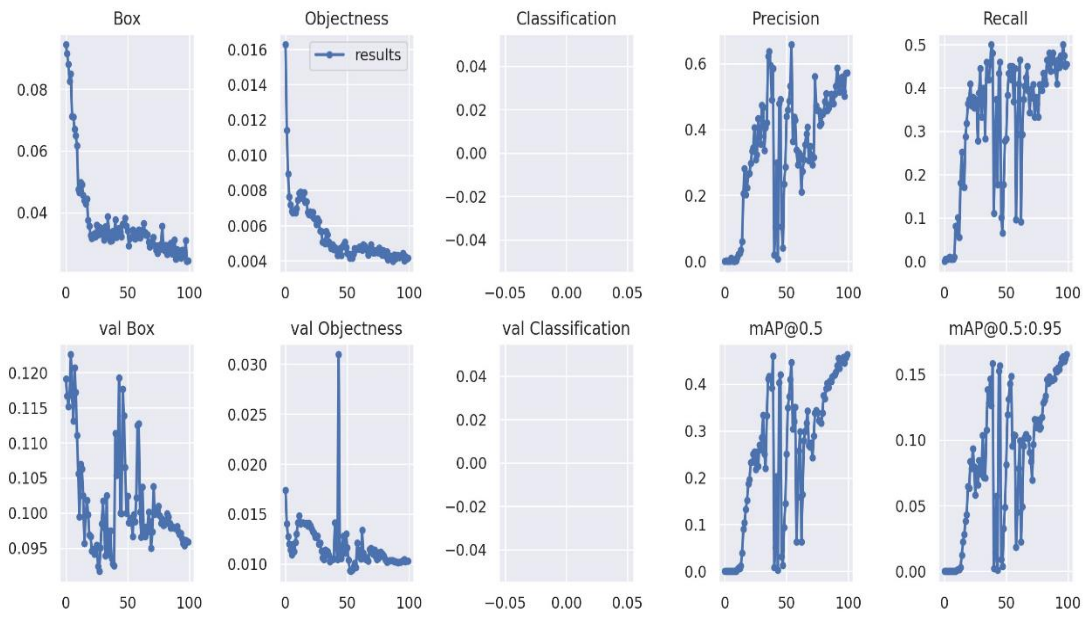


Figure 7.96. Train and validation losses for D3 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the Adam optimizer on D3 dataset are shown in Figures 7.97, and 7.98, respectively.

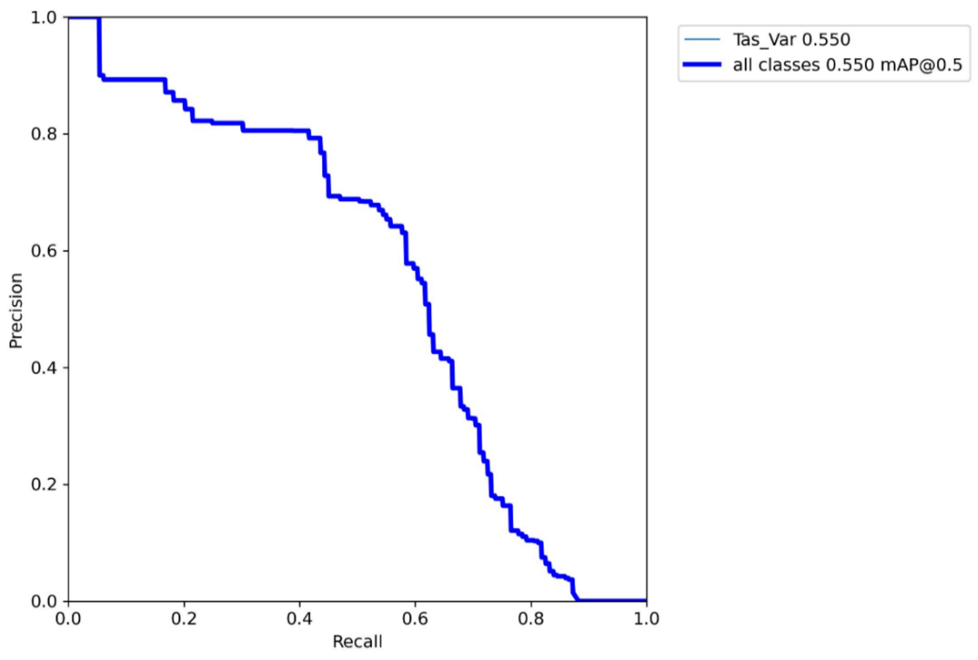


Figure 7.97. Precision-Sensitivity curve for D3 dataset (Test phase).

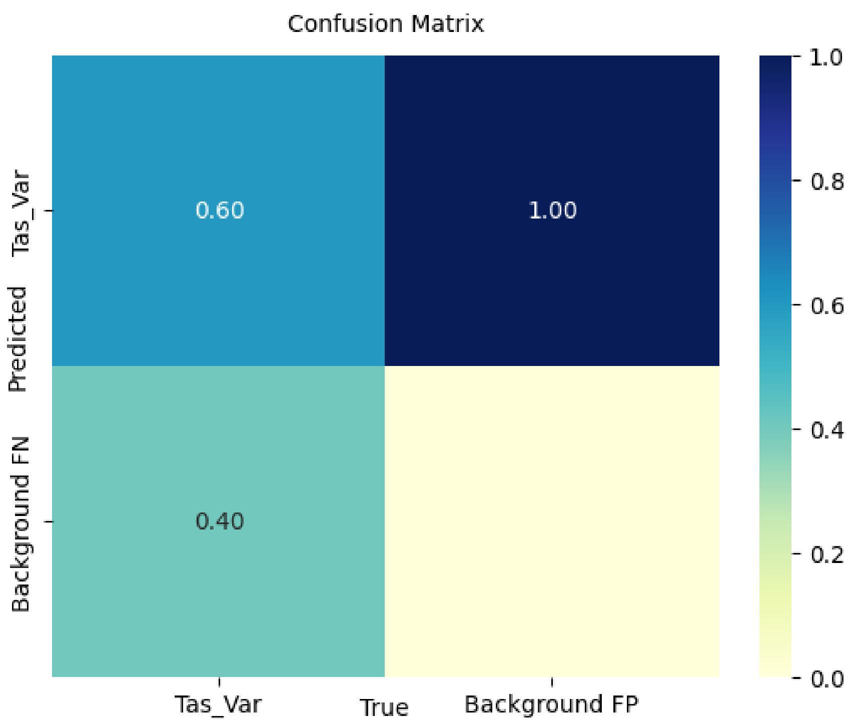


Figure 7.98. Confusion matrix for D3 dataset (Test phase).

The obtained results such as precision-sensitivity curve, confusion matrix, train, and validation loss graphs from training YOLO v7 with the Adam optimizer on D4 dataset are shown in Figures 7.99-7.101.

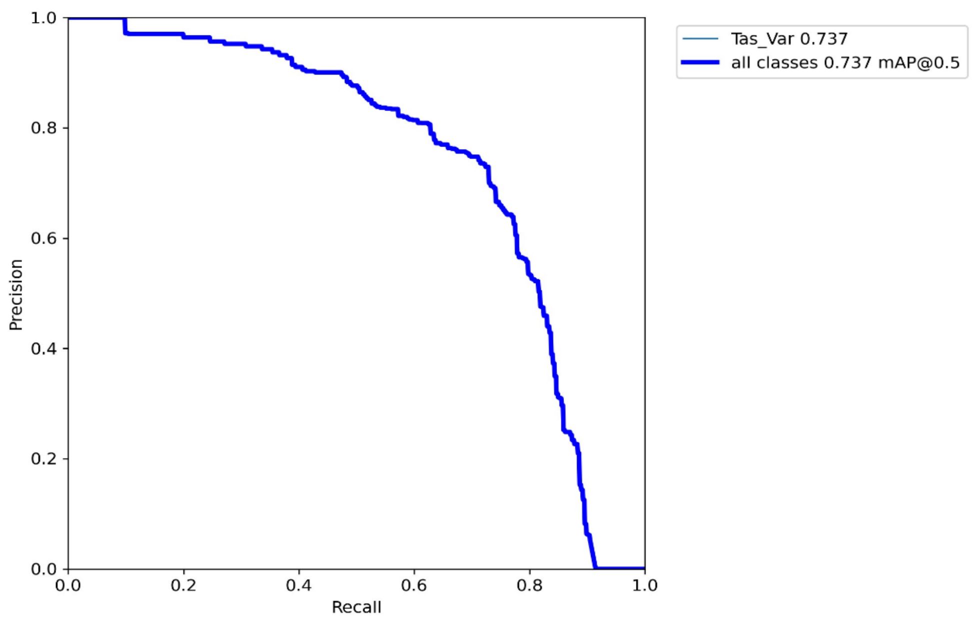


Figure 7.99. Precision-Sensitivity curve for D4 dataset (Train Phase).

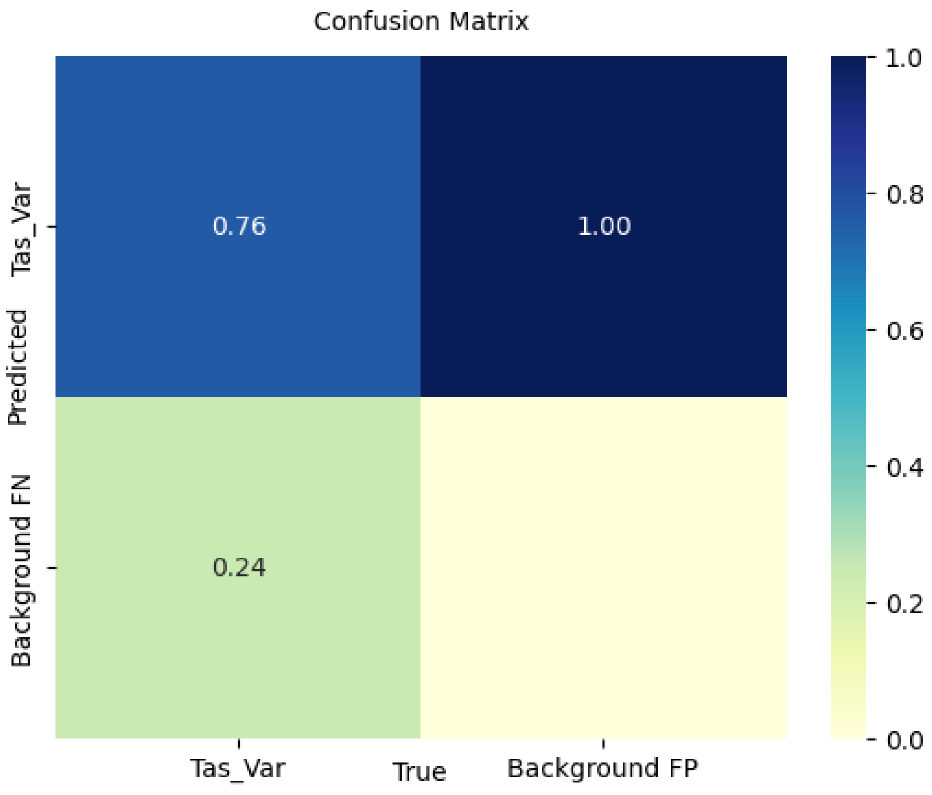


Figure 7.100. Confusion matrix for D4 dataset (Train Phase).

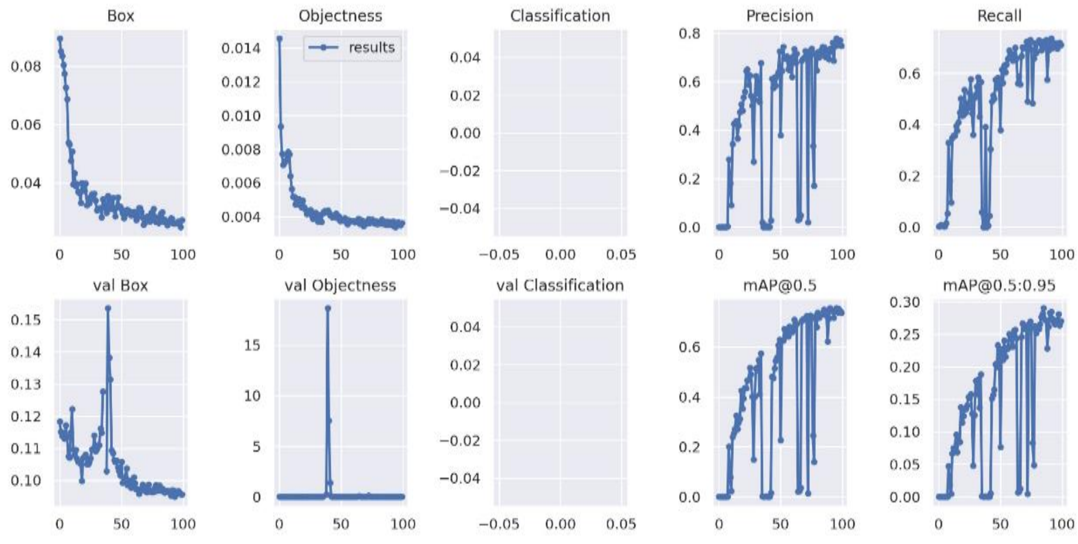


Figure 7.101. Train and validation losses for D4 dataset (Train Phase).

The obtained results such as precision-sensitivity curve and confusion matrix from the test phase of YOLO v7 with the Adam optimizer on D4 dataset are shown in Figures 7.102, and 7.103, respectively.

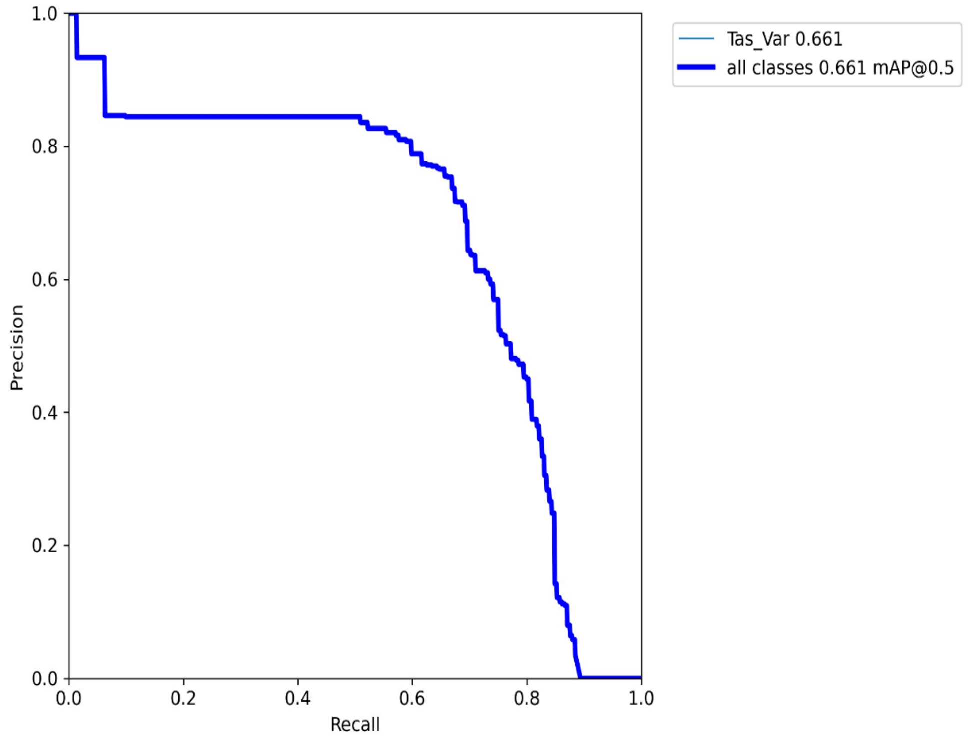


Figure 7.102. Precision-Sensitivity curve for D4 dataset (Test phase).

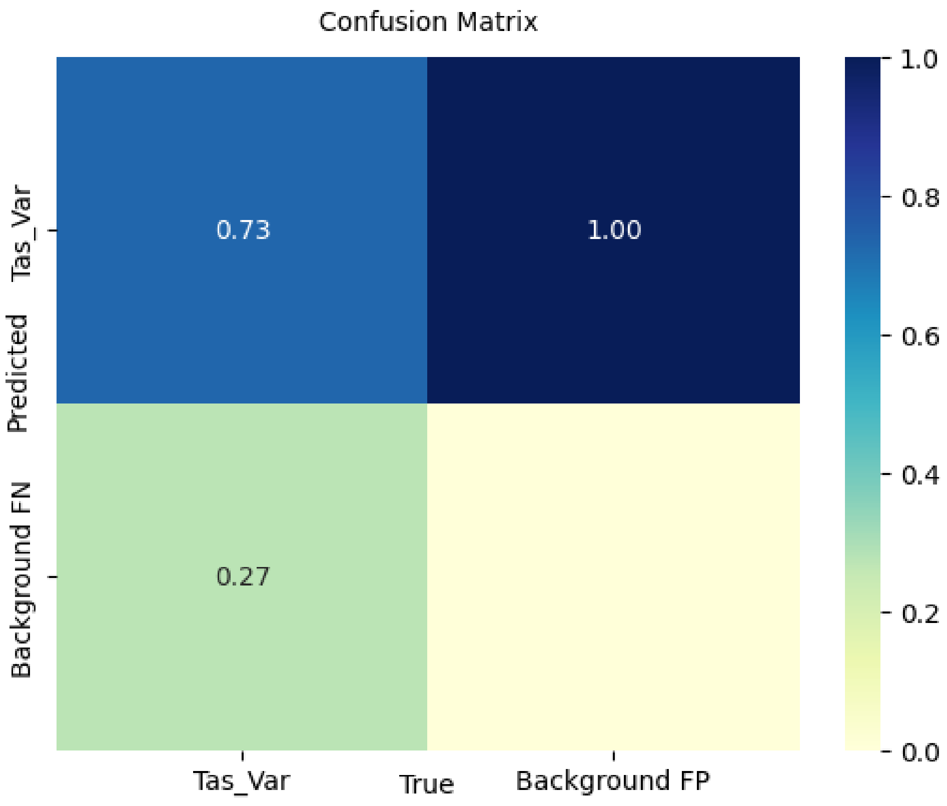


Figure 7.103. Confusion matrix for D4 dataset (Test phase).

Table 7.14 shows a summary of the obtained results for the four datasets in respect to Accuracy and mAP in both of Train and Test phases for YOLO v7 model with Adam optimizer. Table 7.15 shows a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, mAP in Test phase for YOLO v7 model with Adam optimizer.

Table 7.14. Summary of the obtained results in Train and Test.

	Train				**Test**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	71.0	76.0	45.0	76.0	65.0	76.0	60.0	73.0
mAP (0.5) (%)	70.8	73.2	46.4	73.7	64.2	73.7	55.0	66.1

Table 7.15. Summary of the obtained results in Test

	Test			
	D1	**D2**	**D3**	**D4**
Accuracy (%)	65.0	76.0	60.0	73.0
Precision (%)	86.8	83.7	64.2	75.4
Sensitivity (%)	58.9	67.3	57.7	66.9
F1 score (%)	70.1	74.6	60.7	70.8
mAP (0.5) (%)	64.2	73.7	55.0	66.1
mAP (0.5:0.95) (%)	23.4	31.6	21.9	25.7

Table 7.16 show a summary of the obtained results for the four datasets in respect to Accuracy and mAP in Train phase for YOLO v7 model between SGD and Adam optimizers. Table 7.17 shows a summary of the obtained results for the four datasets in respect to Accuracy, Precision, Sensitivity, F1 score, mAP in Test phase for YOLO v7 model between SGD and Adam optimizers.

Table 7.16. Train results of the model between SGD and Adam.

	SGD				**Adam**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	60.0	60.0	57.0	70.0	71.0	76.0	45.0	76.0
mAP (0.5) (%)	46.9	52.8	53.5	64.0	70.8	73.2	46.4	73.7

Table 7.17. Test results of the model between SGD and Adam.

	SGD				**Adam**			
	D1	**D2**	**D3**	**D4**	**D1**	**D2**	**D3**	**D4**
Accuracy (%)	59.0	74.0	66.0	74.0	65.0	76.0	60.0	73.0
Precision (%)	61.9	76.6	72.6	68.9	86.8	83.7	64.2	75.4
Sensitivity (%)	52.5	64.5	62.4	64.3	58.9	67.3	57.7	66.9
F1 score (%)	56.8	70.0	67.1	66.5	70.1	74.6	60.7	70.8
mAP (0.5) (%)	48.3	66.3	61.4	64.9	64.2	73.7	55.0	66.1
mAP (0.5:0.95) (%)	17.3	26.7	26.7	26.1	23.4	31.6	21.9	25.7

7.2.3.1. Experimental results of YOLO v7

Visual representations of the detection results achieved with the YOLO v7 during the test phase of kidney stones in the dataset alongside with the ground truth bounding boxes are given in Figure 7.104-7.108.

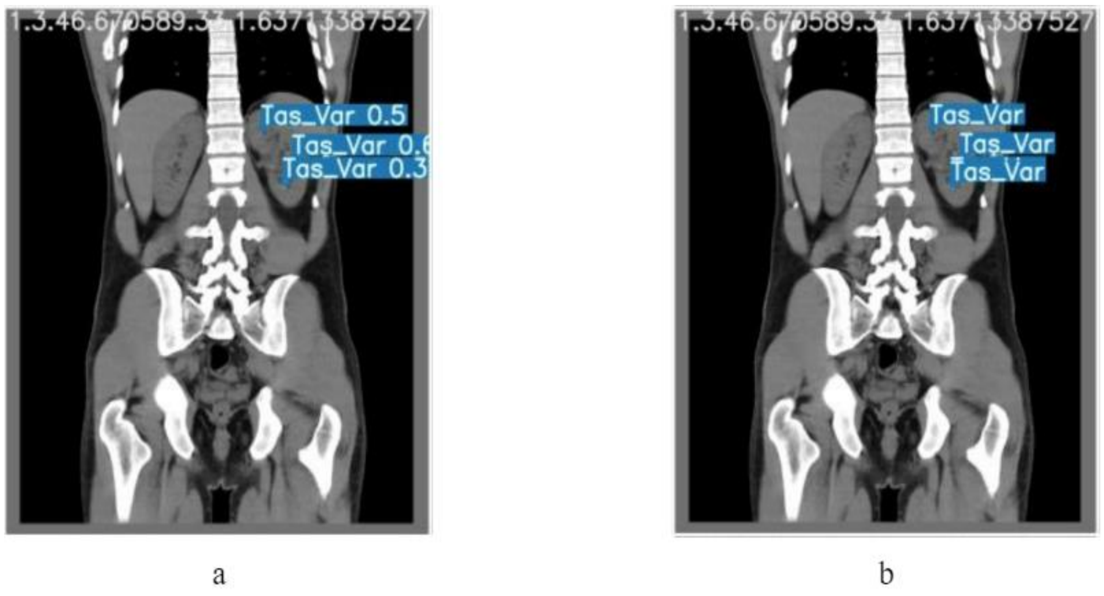


Figure 7.104. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

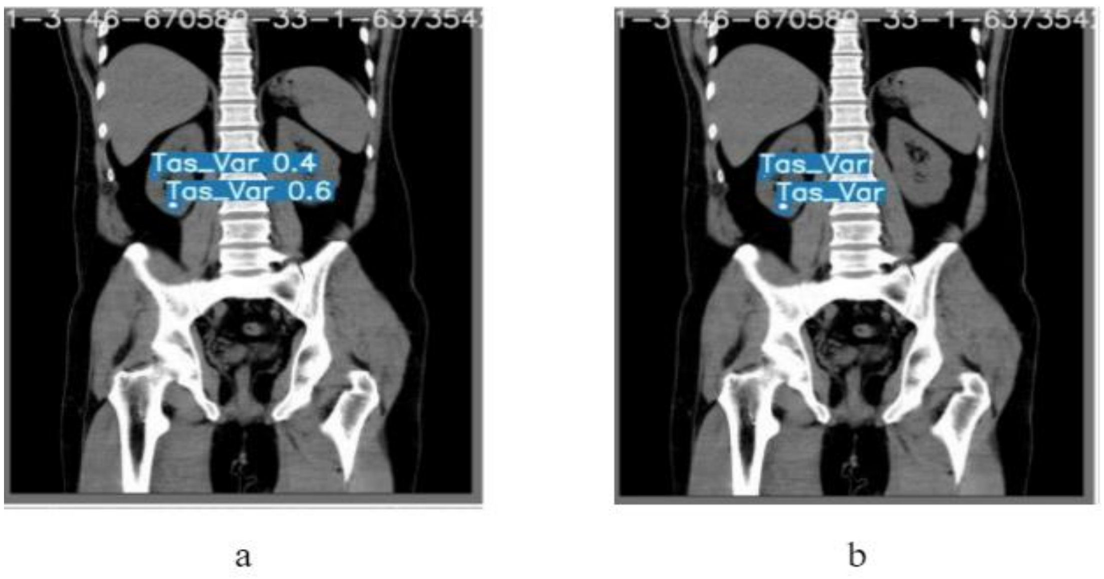
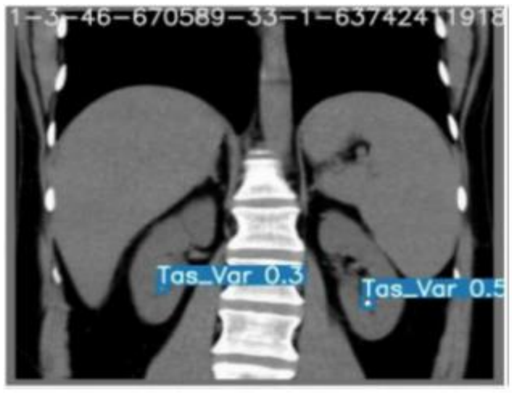
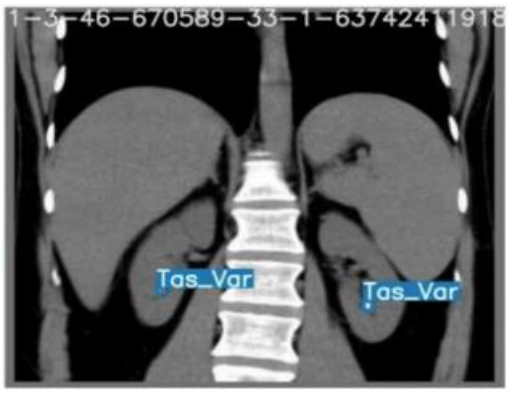


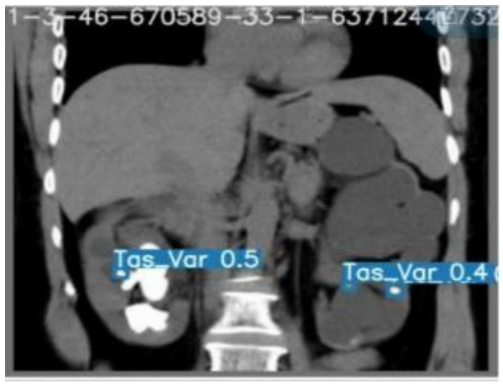
Figure 7.105. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

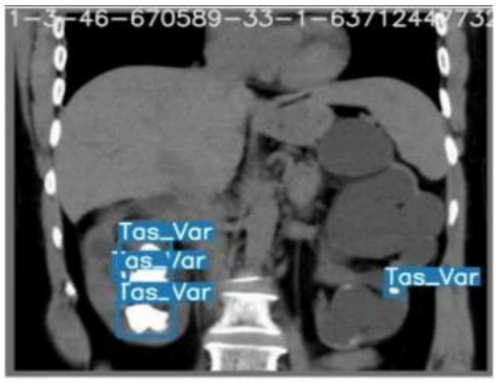




a b

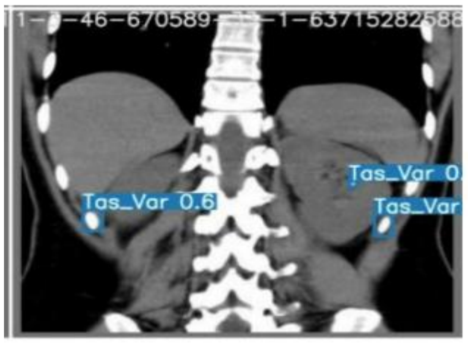
Figure 7.106. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

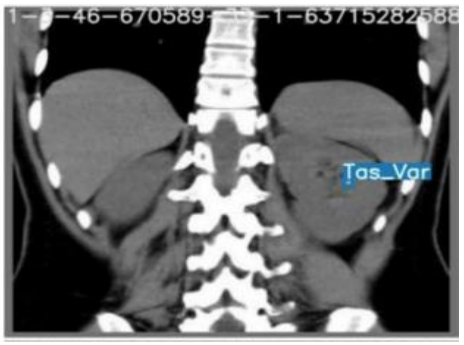




a b

Figure 7.107. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.





a b

Figure 7.108. Evaluation of kidney stones. (a) Model’s predictions (b) ground truth bounding boxes.

7.3. CNN

In this thesis, a deep learning system was created to classify kidney stones with the help of approximately 1799 CT scans images labeled as Normal and Kidney stone obtained from open source. Images used in the model were separated as 80% for training and 20% for testing. Additionally, two different optimizers, RMSprop and Adam, were used for classification purpose. The number of epochs was also assigned as 20, 40, 60, 80, and 100. The learning rate value was set as 0.001.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the RMSprop optimizer on D1 dataset for 20 epochs are shown in Figures 7.109-7.111.

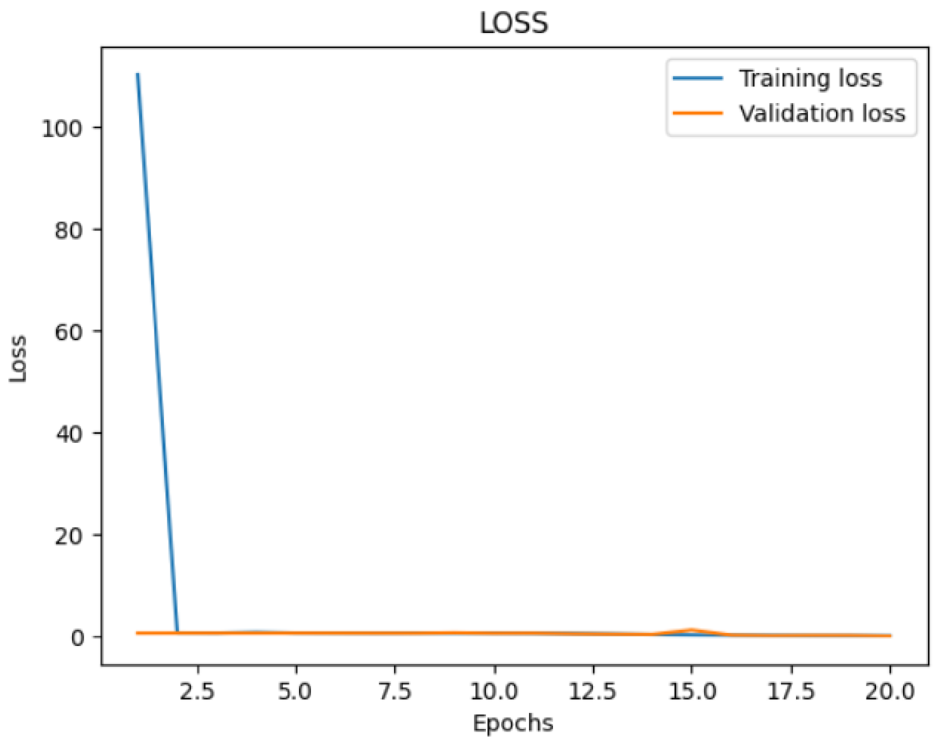


Figure 7.109. Training and validation losses graphs.

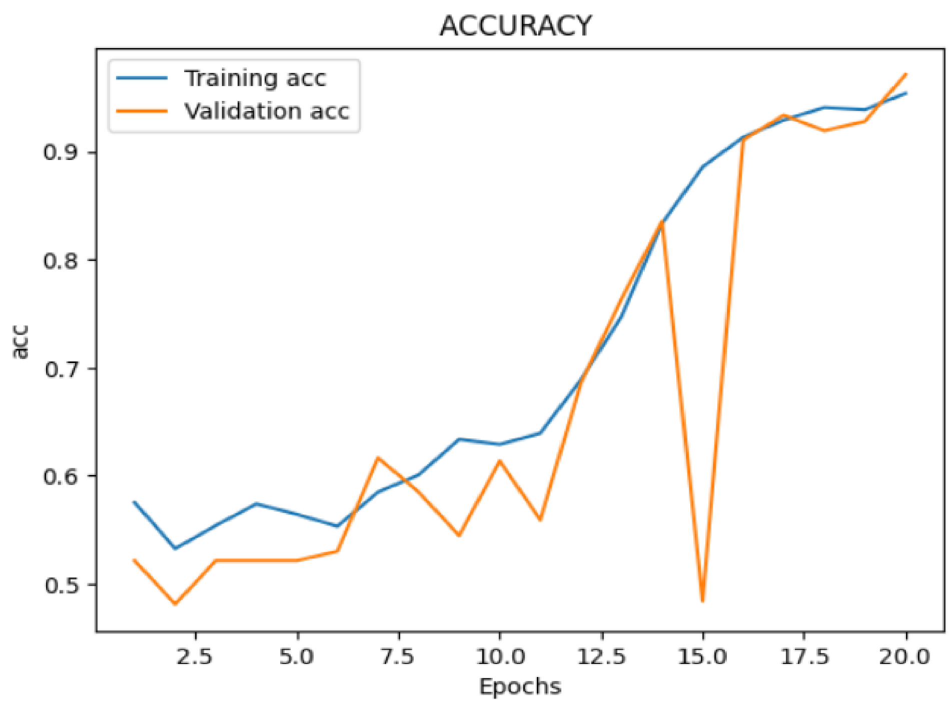


Figure 7.110. Training and validation accuracy graphs.

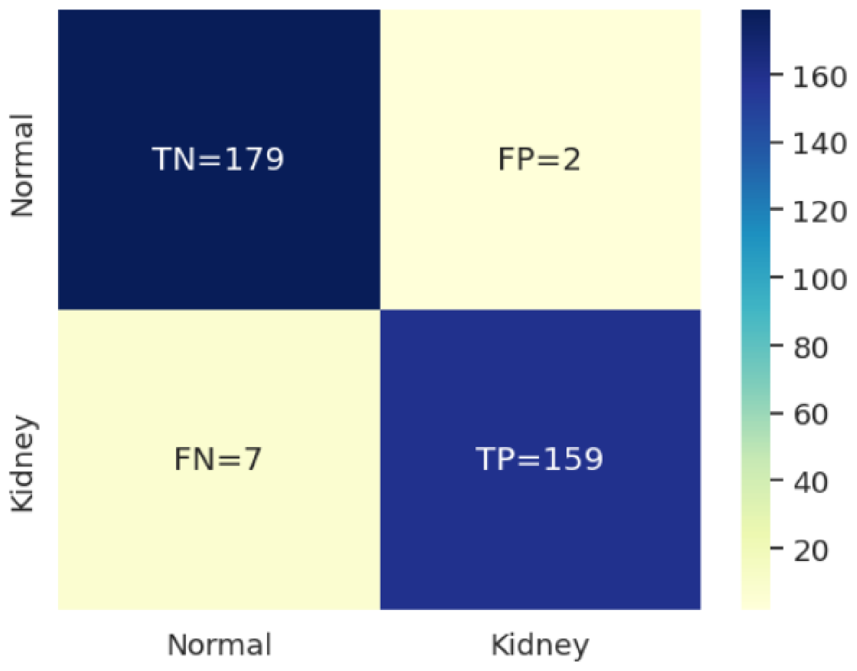


Figure 7.111. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the RMSprop optimizer on D1 dataset for 40 epochs are shown in Figures 7.112-7.114.

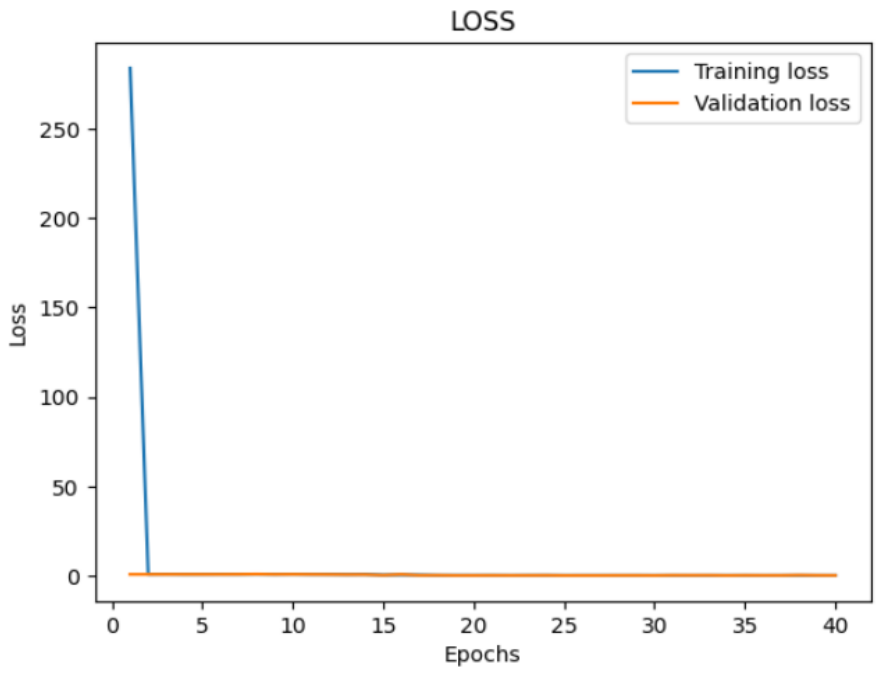


Figure 7.112. Training and validation losses graphs.

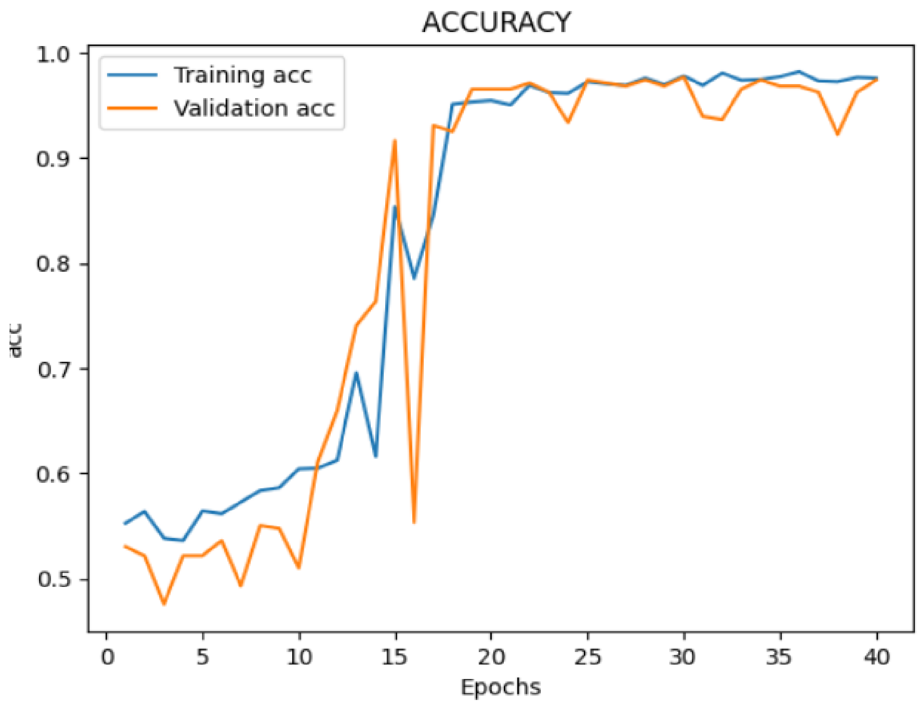


Figure 7.113. Training and validation accuracy graphs.

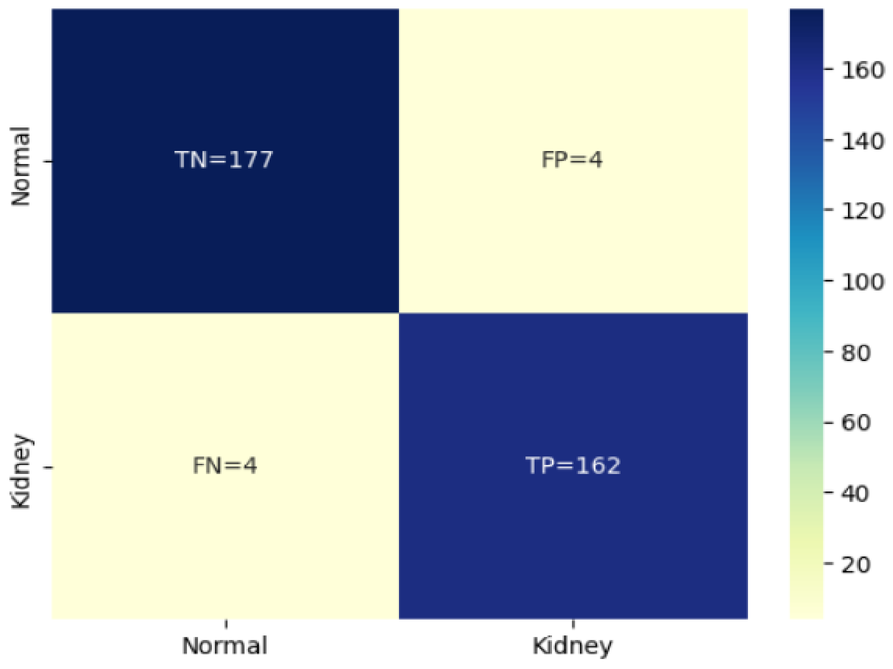


Figure 7.114. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the RMSprop optimizer on D1 dataset for 60 epochs are shown in Figures 7.115-7.117.

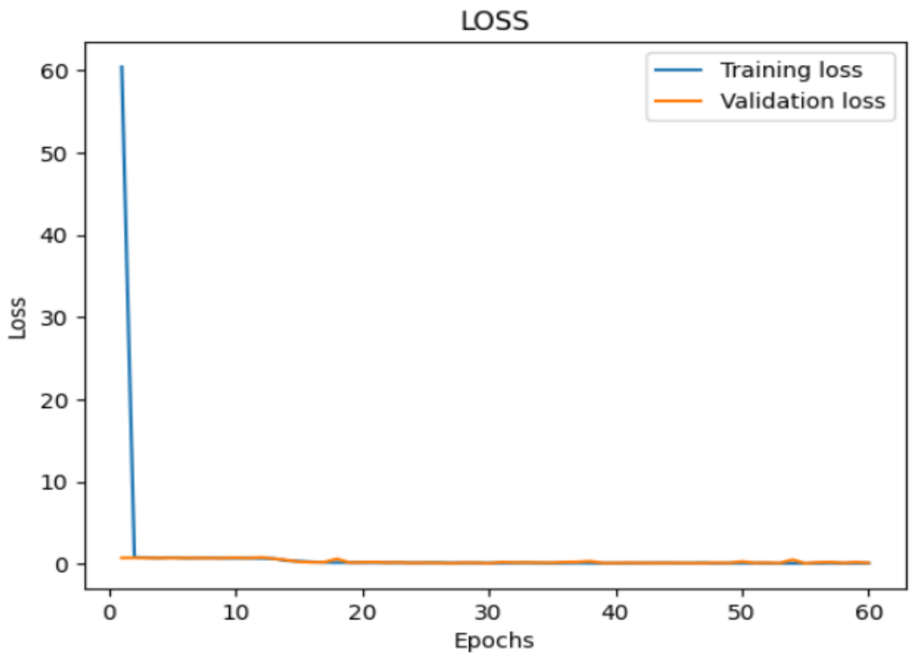


Figure 7.115. Training and validation losses graphs.

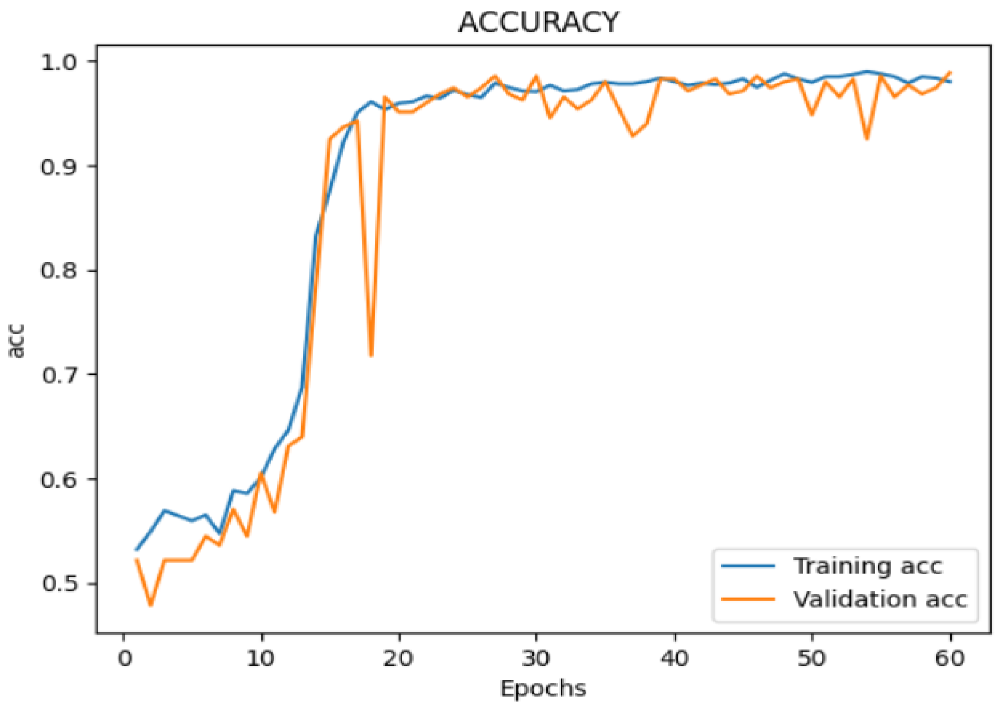


Figure 7.116. Training and validation accuracy graphs.

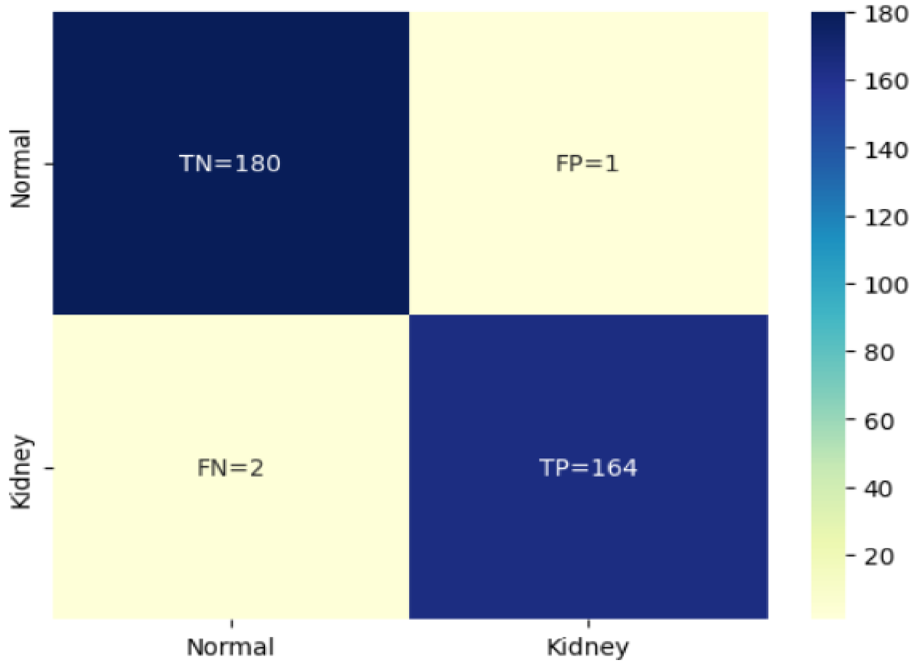


Figure 7.117. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the RMSprop optimizer on D1 dataset for 80 epochs are shown in Figures 7.118-7.120.

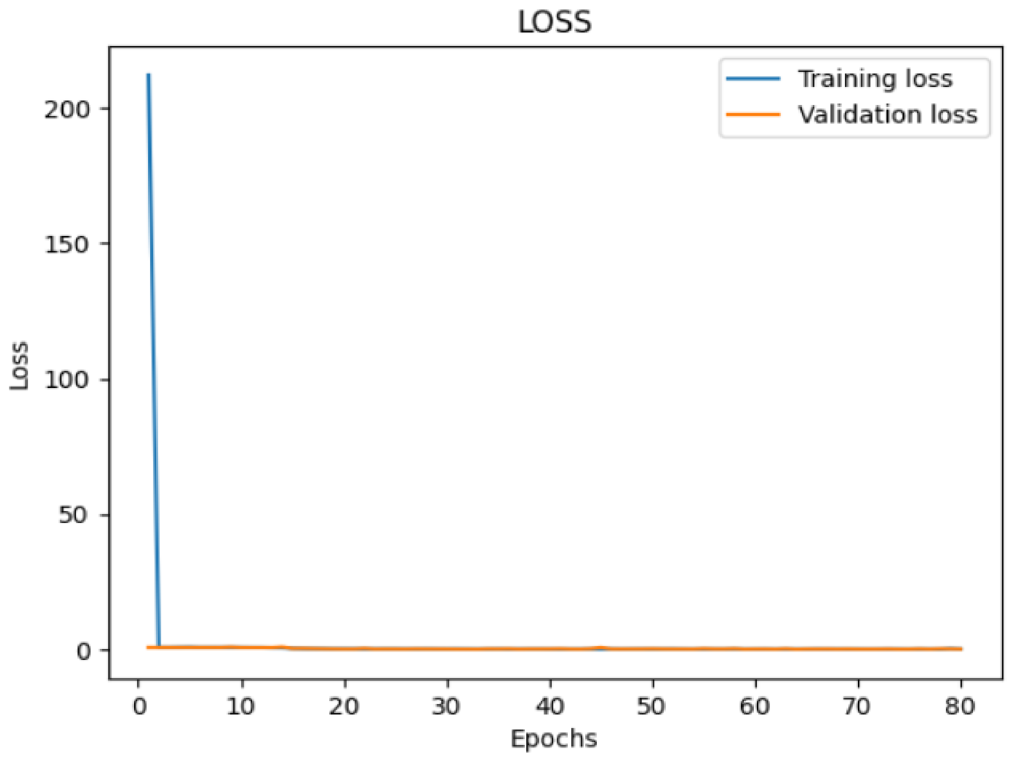


Figure 7.118. Training and validation losses graphs.

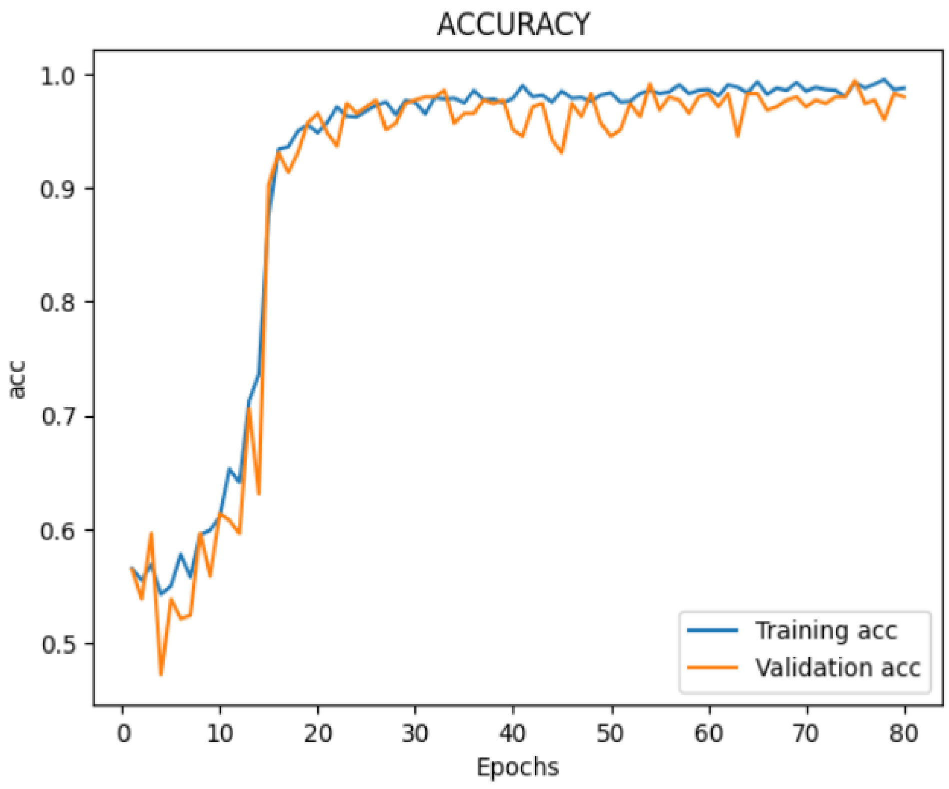


Figure 7.119. Training and validation accuracy graphs.

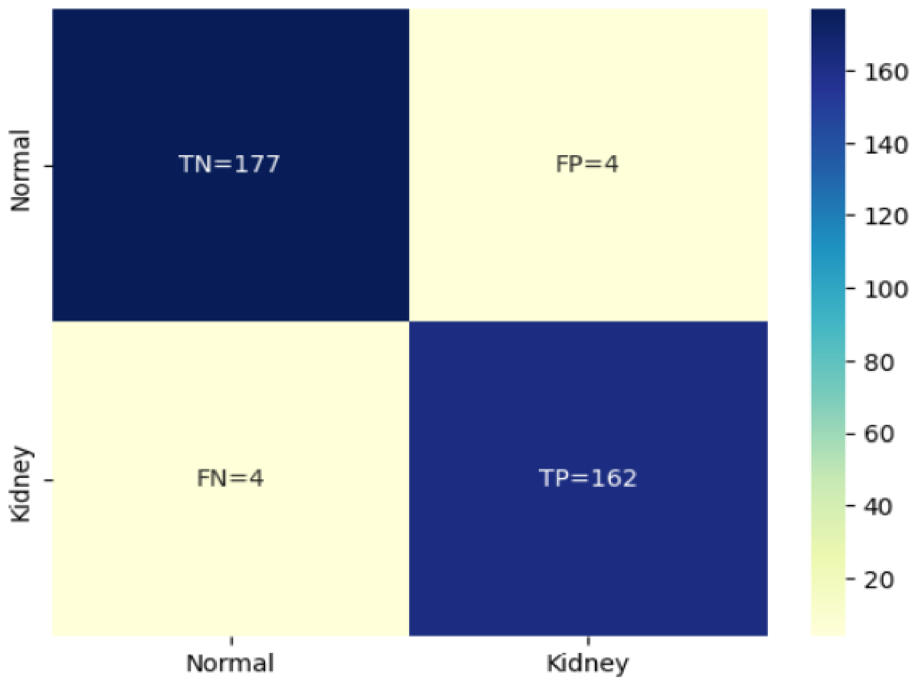


Figure 7.120. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the RMSprop optimizer on D1 dataset for 100 epochs are shown in Figures 7.121-7.123.

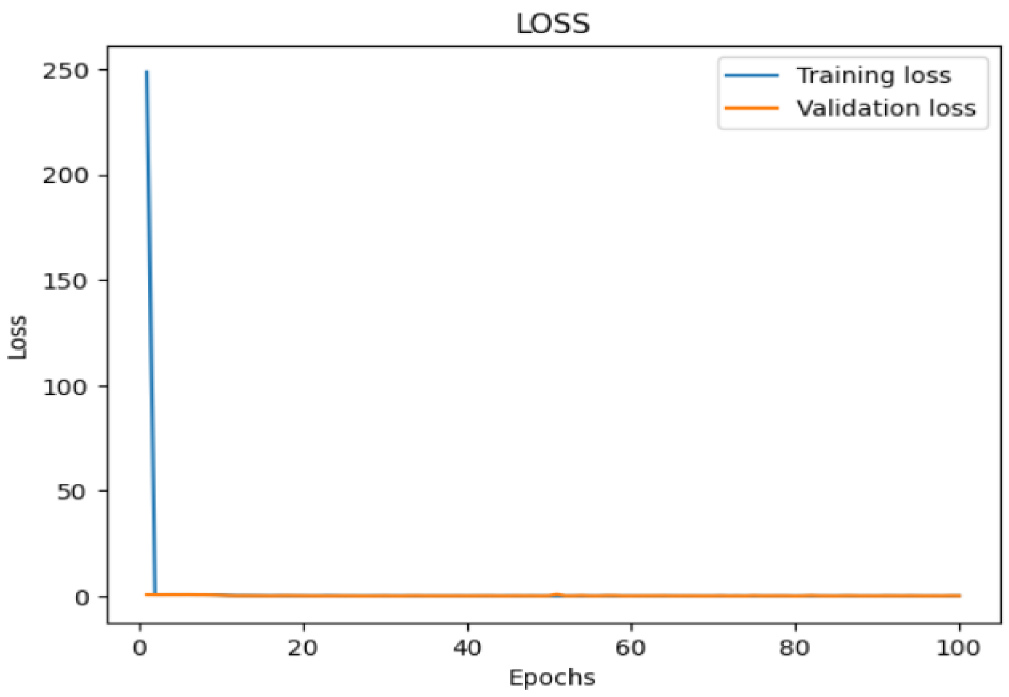


Figure 7.121. Training and validation losses graphs.

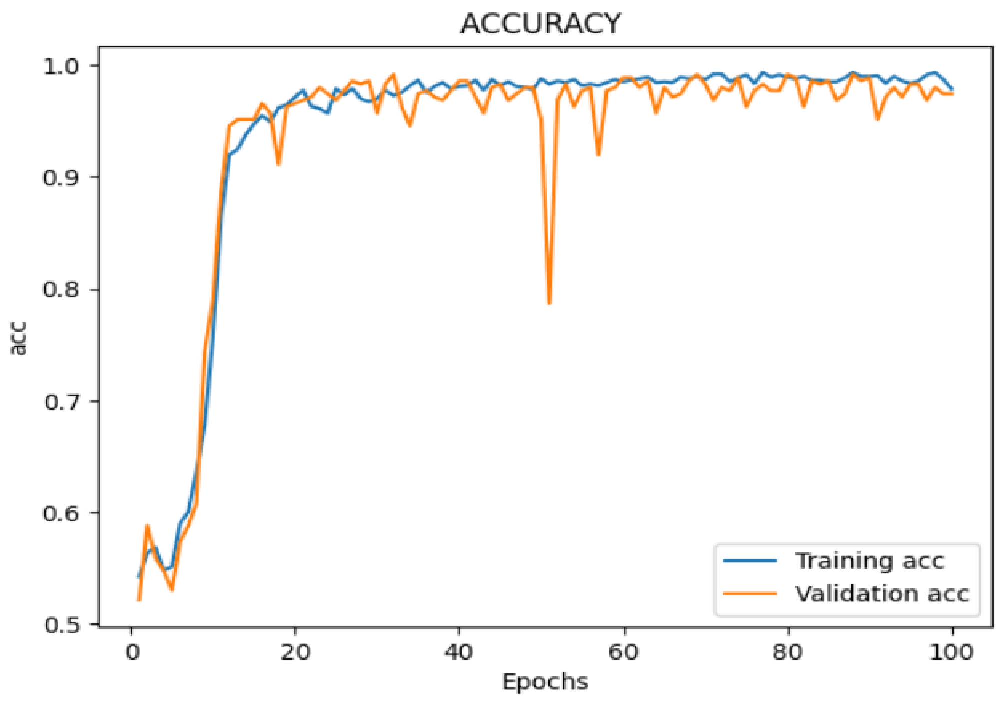


Figure 7.122. Training and validation accuracy graphs.

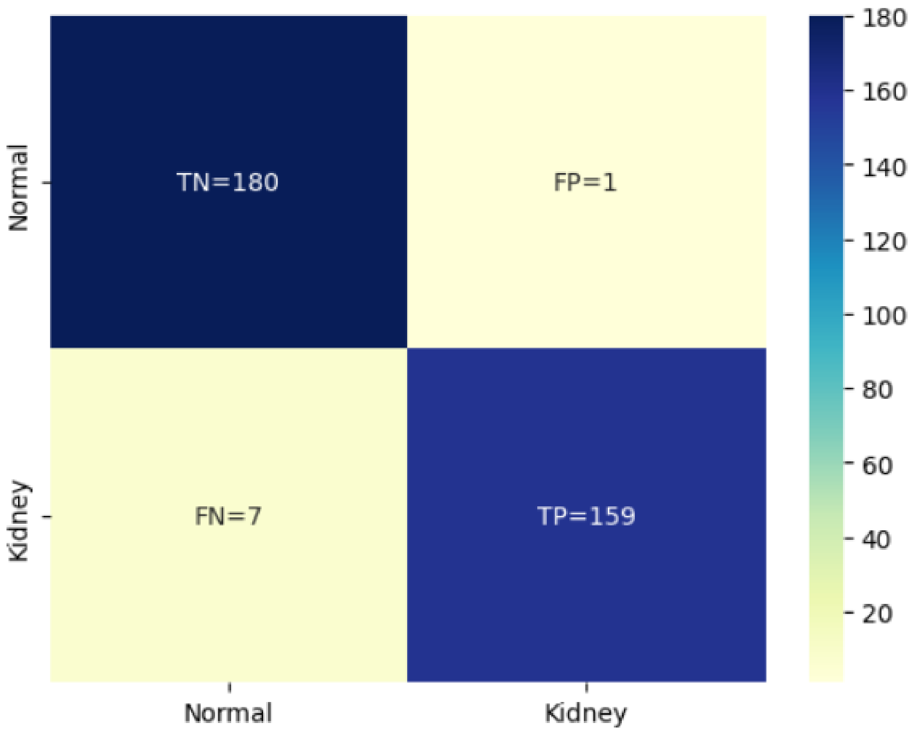


Figure 7.123. Confusion matrix.

Table 7.18 shows a summary of the obtained results in respect of Accuracy, Precision, Sensitivity, Specificity, and F1 score for CNN model with RMSprop optimizer alongside with different epochs.

Table 7.18. Summary of the obtained results.

Epochs	**20**	**40**	**60**	**80**	**100**
Accuracy (%)	97.40	97.69	99.13	97.69	97.69
Precision (%)	98.75	97.59	99.39	97.59	99.37
Sensitivity (%)	95.78	97.59	98.79	97.59	95.78
Specificity (%)	98.89	97.79	99.44	97.79	99.44
F1 Score (%)	97.24	97.59	99.08	97.59	97.54

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the Adam optimizer on D1 dataset for 20 epochs are shown in Figures 7.124-7.126.

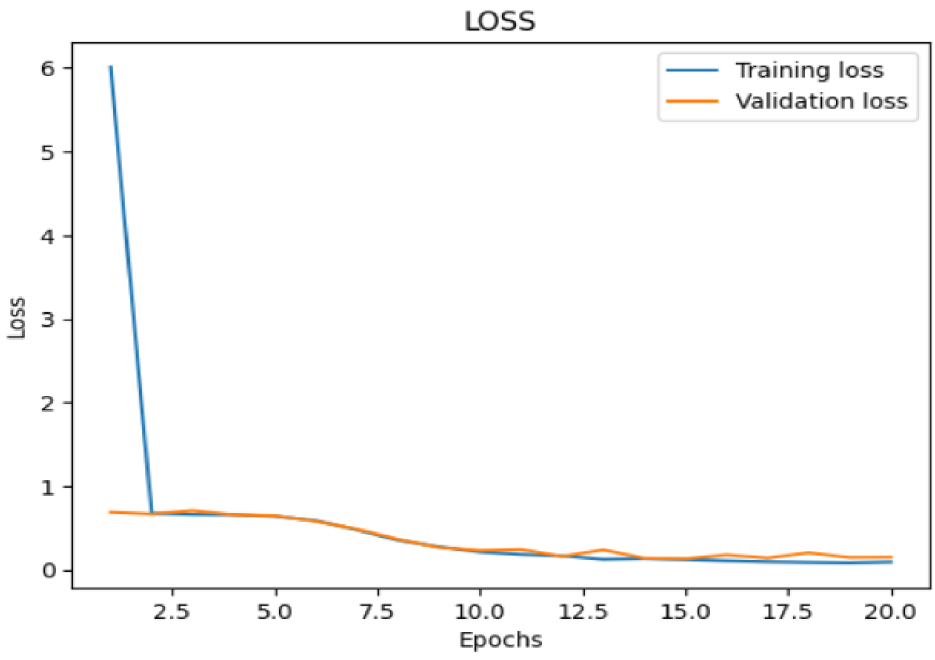


Figure 7.124. Training and validation losses graphs.

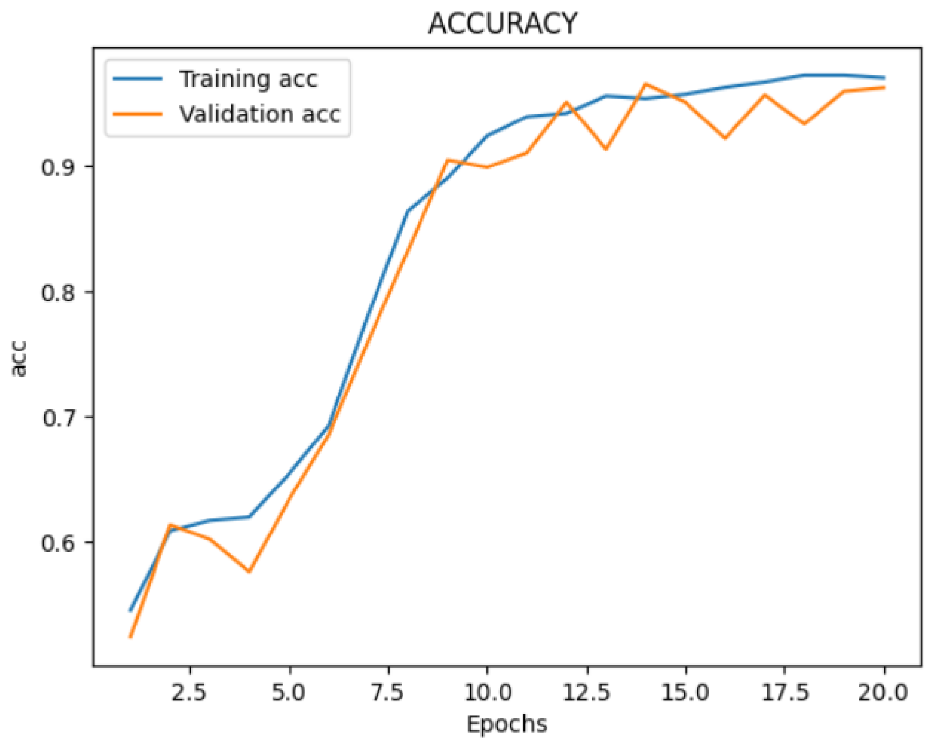


Figure 7.125. Training and validation accuracy graphs.

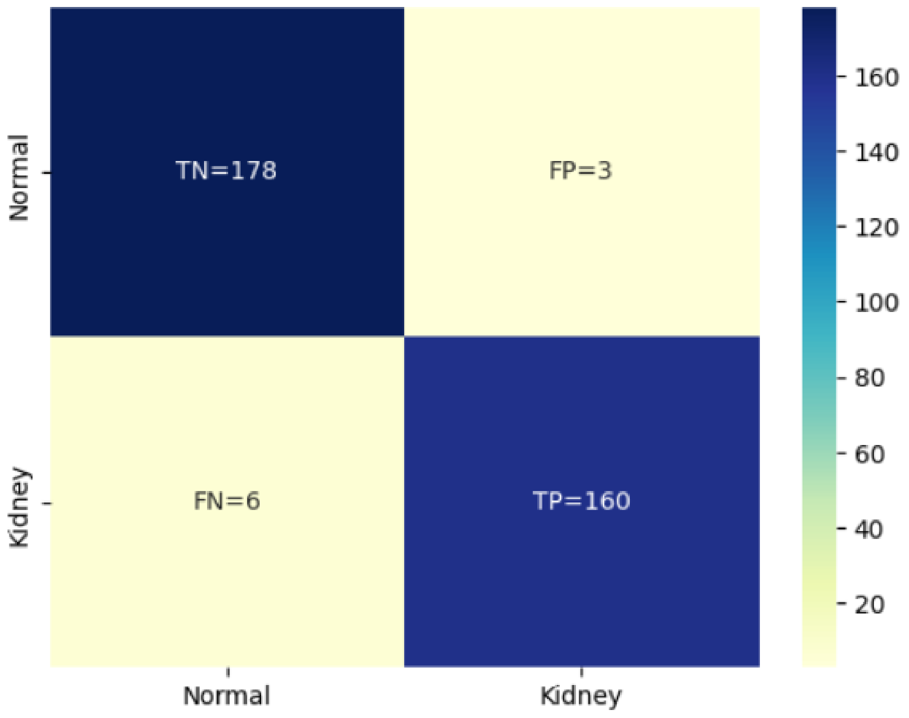


Figure 7.126. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the Adam optimizer on D1 dataset for 40 epochs are shown in Figures 7.127-7.129.

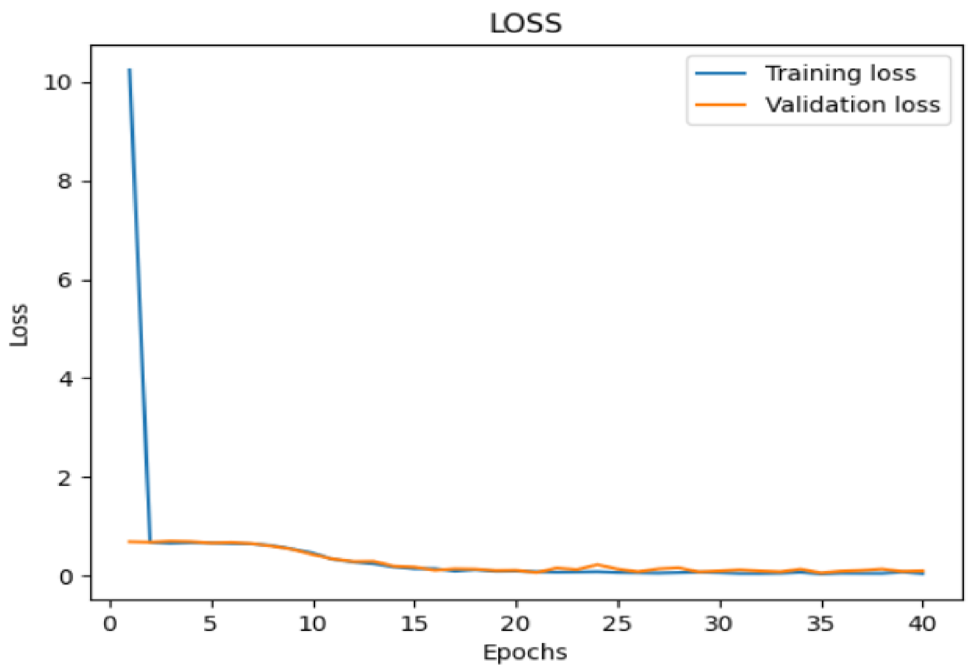


Figure 7.127. Training and validation losses graphs.

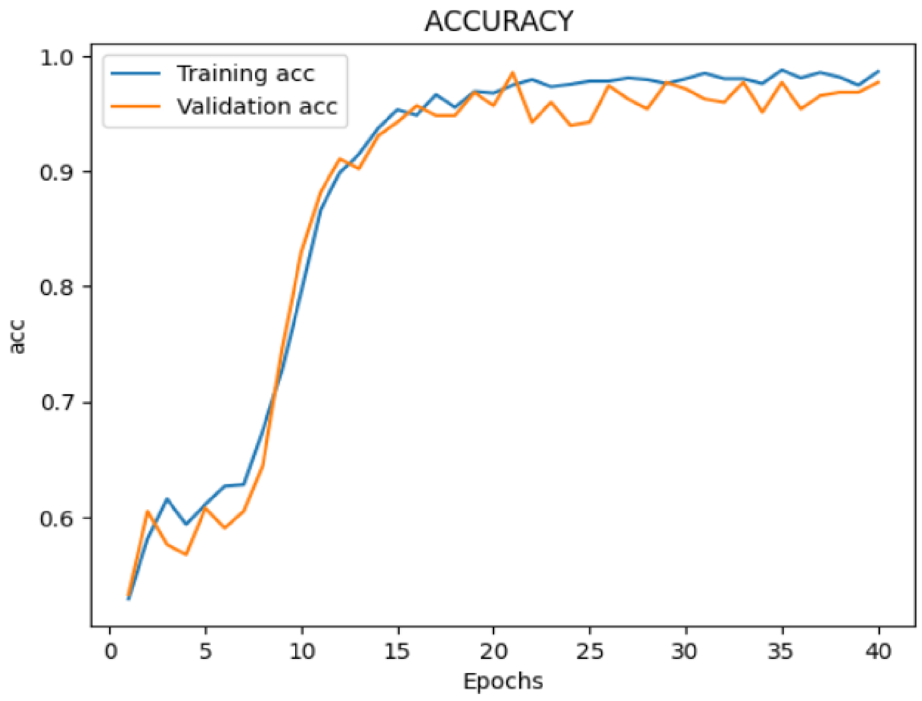


Figure 7.128. Training and validation accuracy graphs.

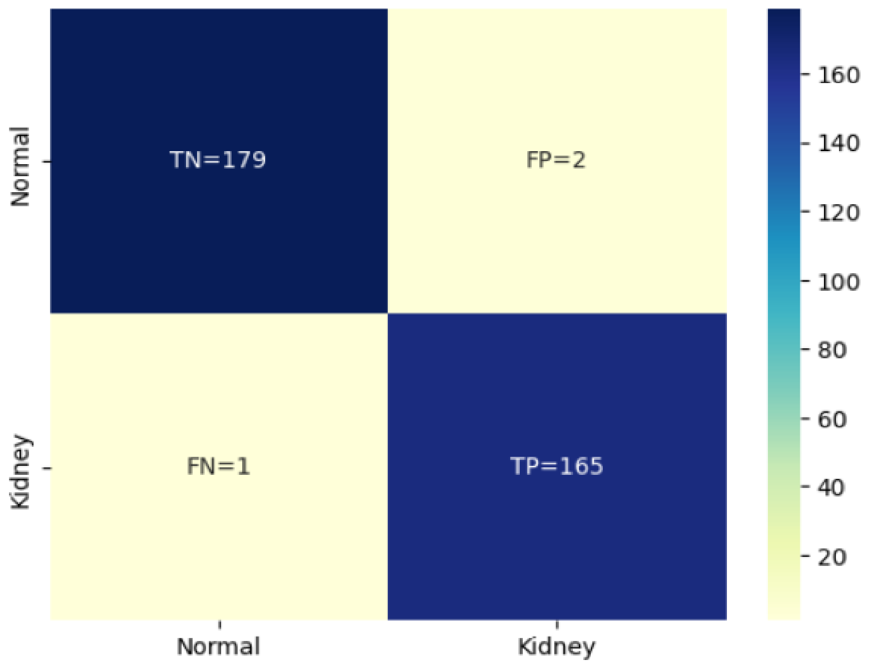


Figure 7.129. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the Adam optimizer on D1 dataset for 60 epochs are shown in Figures 7.130-7.132.

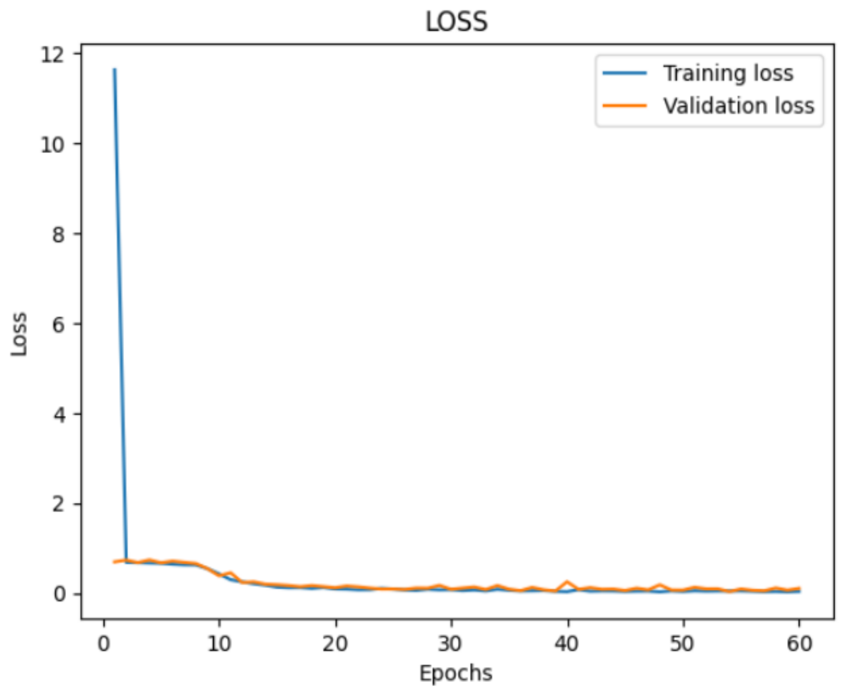


Figure 7.130. Training and validation losses graphs.

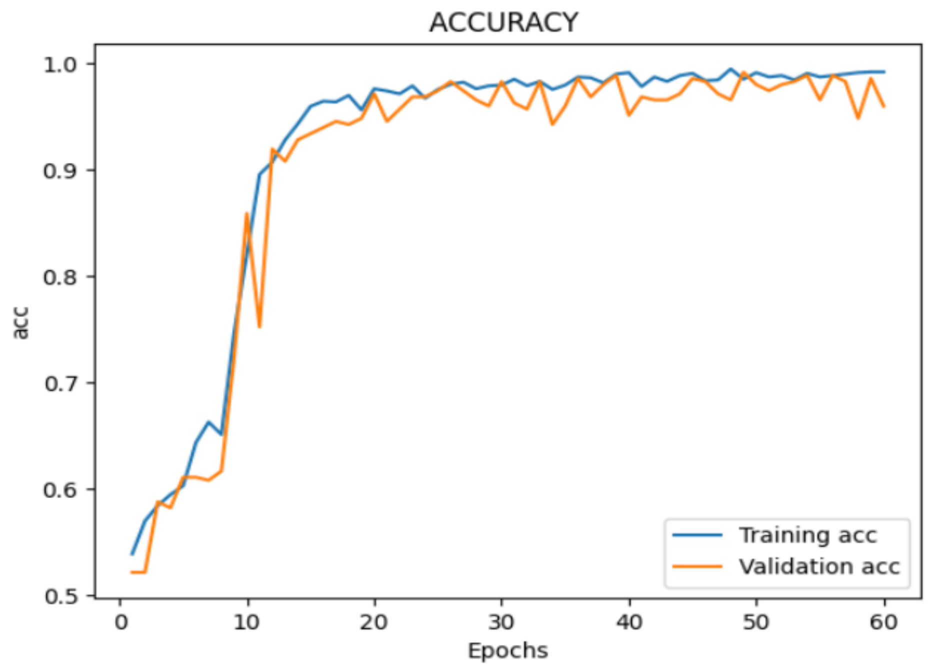


Figure 7.131. Training and validation accuracy graphs.

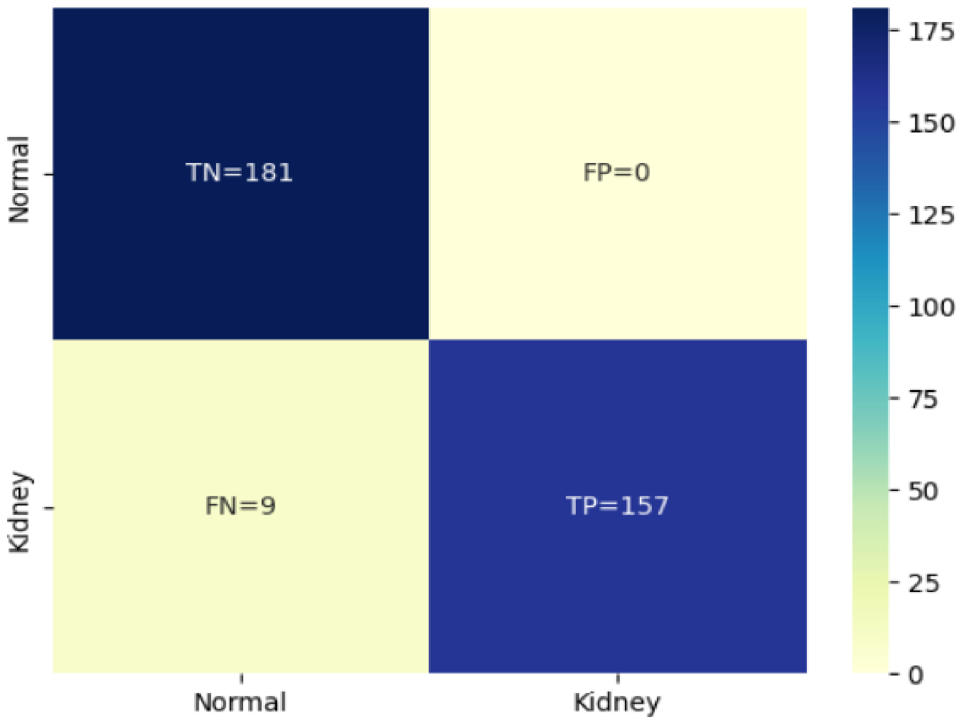


Figure 7.132. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the Adam optimizer on D1 dataset for 80 epochs are shown in Figures 7.133-7.135.

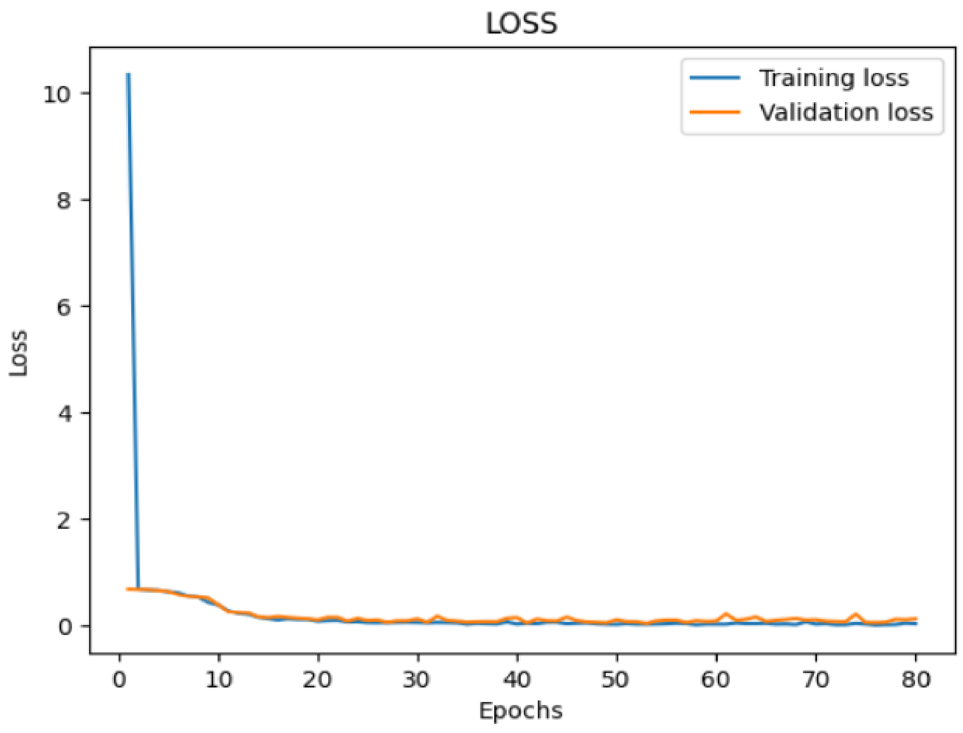


Figure 7.133. Training and validation losses graphs.

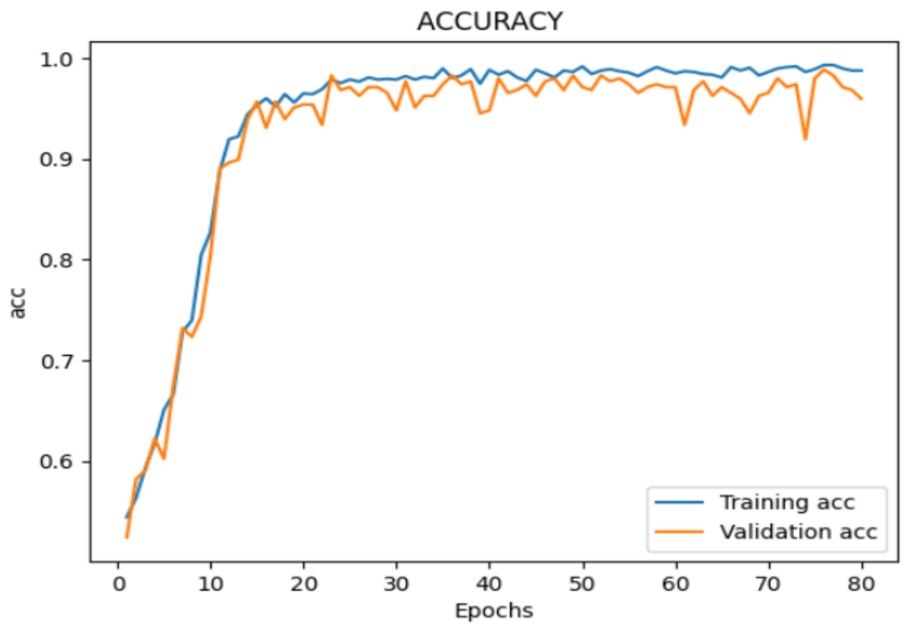


Figure 7.134. Training and validation accuracy graphs.

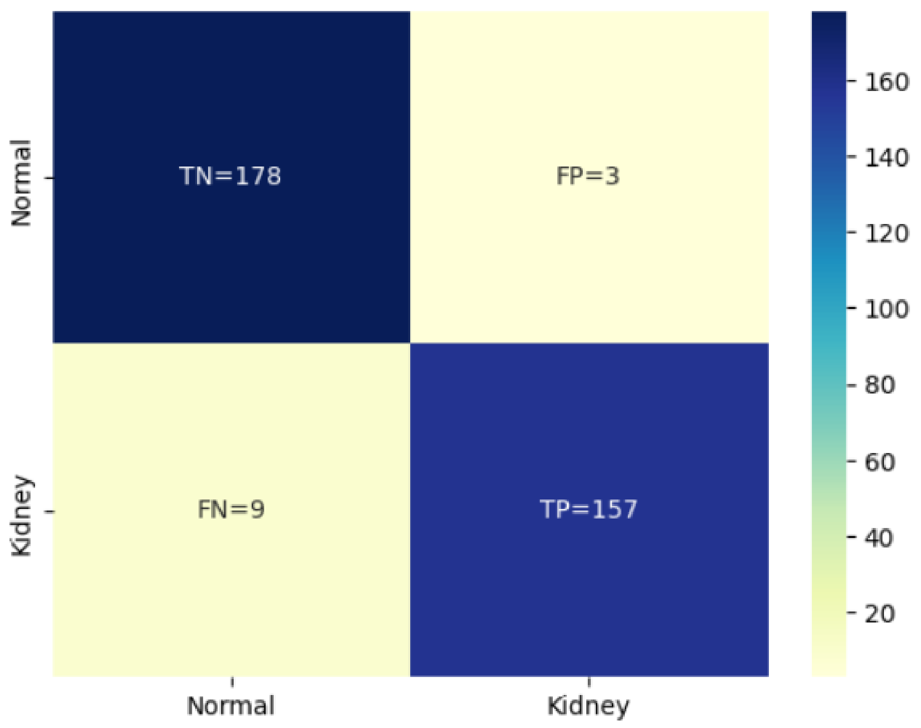


Figure 7.135. Confusion matrix.

The obtained results such as train, validation loss graphs, and confusion matrix of CNN with the Adam optimizer on D1 dataset for 100 epochs are shown in Figures 7.136-7.138.

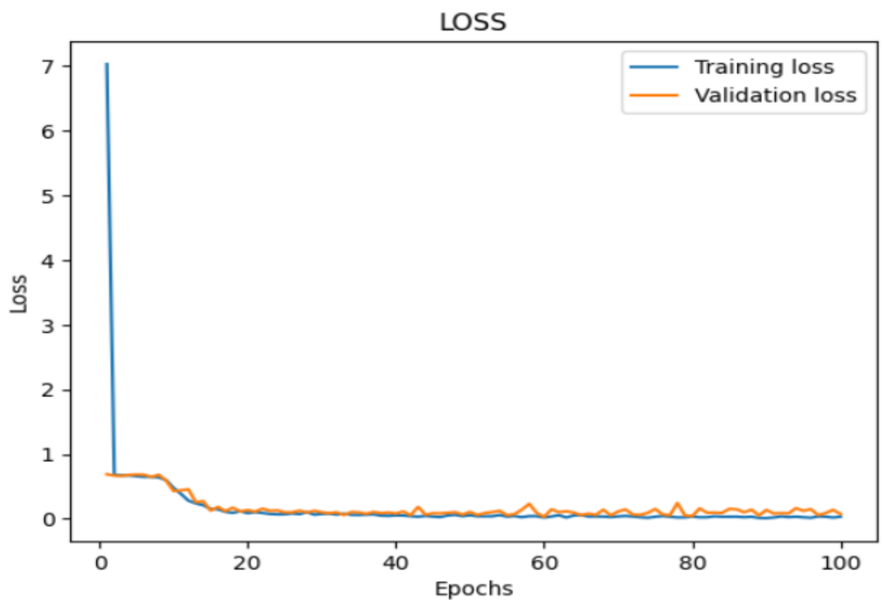


Figure 7.136. Training and validation losses graphs.

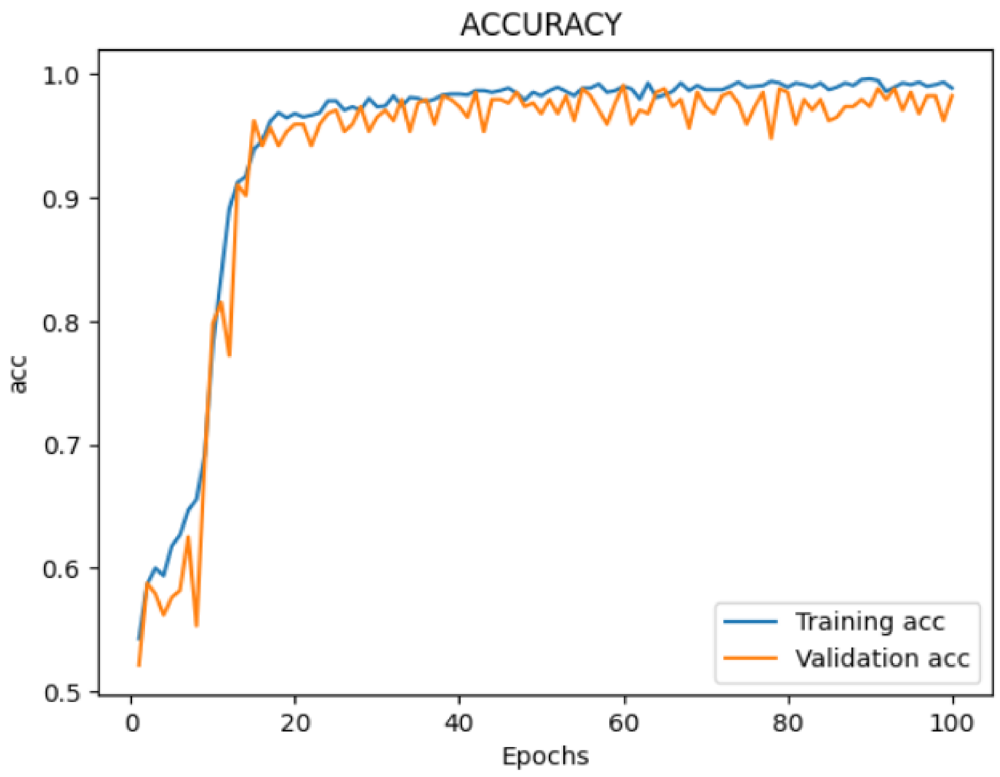


Figure 7.137. Training and validation accuracy graphs.

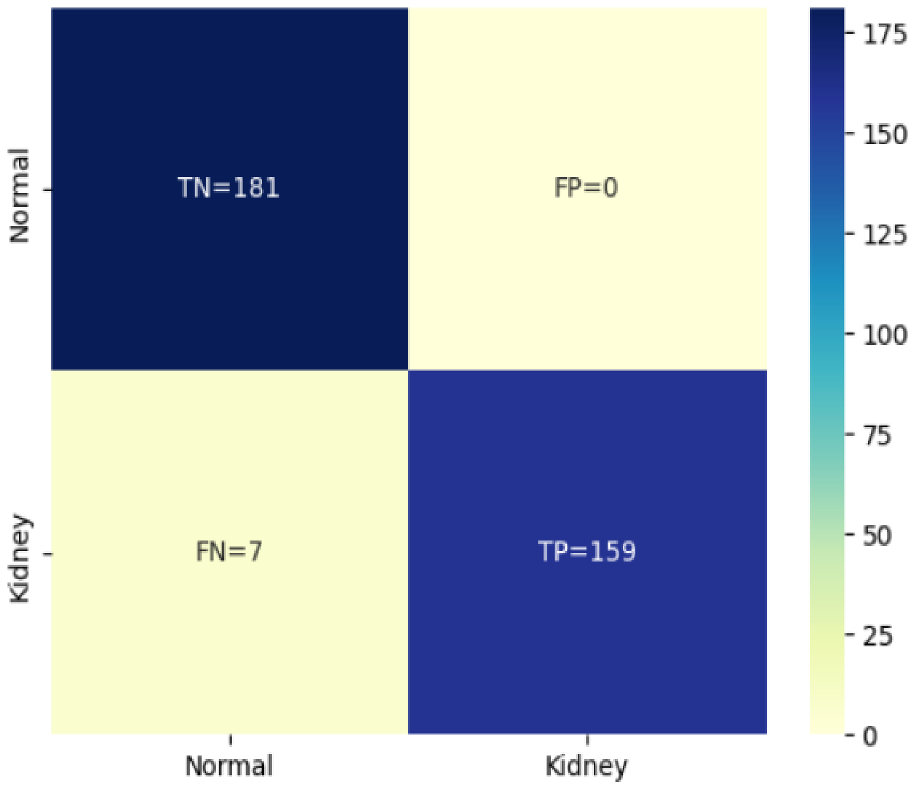


Figure 7.138. Confusion matrix.

Table 7.19 shows a summary of the obtained results in respect of Accuracy, Precision, Sensitivity, Specificity, and F1 score for CNN model with Adam optimizer alongside with different epochs.

Table 7.19. Summary of the obtained results.

Epochs	**20**	**40**	**60**	**80**	**100**
Accuracy (%)	97.40	99.13	97.40	96.54	97.98
Precision (%)	98.15	98.80	100	98.12	100
Sensitivity (%)	96.38	99.39	94.57	94.57	95.78
Specificity (%)	98.34	98.89	100	98.34	100
F1 Score (%)	97.25	99.09	97.20	96.31	97.84

Table 7.20 shows a summary of the obtained results in respect of Accuracy, Precision, Sensitivity, Specificity, and F1 score for CNN model with RMSprop and Adam optimizer alongside with different epochs.

Table 7.20. Summary of the obtained results based on RMSprop and Adam.

	RMSprop					**Adam**				
Epochs	**20**	**40**	**60**	**80**	**100**	**20**	**40**	**60**	**80**	**100**
Accuracy (%)	97.40	97.69	99.13	97.69	97.69	97.40	99.13	97.40	96.54	97.98
Precision (%)	98.75	97.59	99.39	97.59	99.37	98.15	98.80	100	98.12	100
Sensitivity (%)	95.78	97.59	98.79	97.59	95.78	96.38	99.39	94.57	94.57	95.78
Specificity (%)	98.89	97.79	99.44	97.79	99.44	98.34	98.89	100	98.34	100
F1 score (%)	97.24	97.59	99.08	97.59	97.54	97.25	99.09	97.20	96.31	97.84

7.4. DISCUSSION

In this thesis, publicly available CT kidney stone dataset was utilized, and three distinct deep learning models namely Faster R-CNN, YOLO, and a custom CNN were trained and tested. The primary objective of the thesis is to achieve both kidney stone detection and classification within the CT image dataset.

The first model employed was the Faster R-CNN, designed for precise object detection in computer vision. Complementing the Faster R-CNN model, feature extraction backbones ResNet50 and ResNet101 were incorporated. Three different variants of Faster R-CNN as ResNet50 V1 800x1333, ResNet101 V1 800x1333, and ResNet101 V1 1024x1024 were utilized. Considering the obtained results, the three models exhibited a very similar values in terms of AP and AS. The highest performance was achieved by ResNet101 V1 1024x1024 model which yielded a value of 63.7% in respect to AP (0.50). The significance of the number of steps parameter on the training results of Faster R-CNN has been demonstrated by researchers Oluibukun Gbenga Ajayi and John Ashi in their article [111]. This study investigated the impact of varying training epochs on the accuracy of a Faster R-CNN. They trained the Faster R-CNN model with five different epochs. They have concluded that increasing the number of training epochs significantly enhances the model's performance; however, a loss of efficiency occurs after a certain number of epochs. The accuracy for the five classes increased across epochs: 52.6% (10,000 epochs), 67.9% (20,000 epochs), 97.3% (100,000 epochs), 98.4% (200,000 epochs), and 97% (242,000 epochs). Taking these important findings into account, it's clear that in this thesis, the choice to limit the number of steps to 8000 due to computational limits noticeably affected the results. The constraint on training steps, driven by the limitations in computational resources, probably played a role in the obtained results.

The second utilized model was YOLO, with two different versions: YOLO v5 and YOLO v7. Additionally, two different optimizers, named SGD and Adam, were employed. In light of the obtained results, YOLO v5 with the Adam optimizer outperformed all other models with an 84.6% mAP (0.5) and an accuracy value of

85.0%. Furthermore, the model achieved a precision of 89.7%, sensitivity of 77.2%, F1 score of 82.9%, and a mAP (0.5:0.95) of 39.0%.

YOLO v7 yielded less favorable results, with various issues becoming apparent during the model evaluation. One of the observed issues was overfitting. Notably, there was a substantial decline in mAP curve during the training phase, which lead to a potential issue with the model's generalization. In addition to that, some of the validation loss curves were increasing, pointing to a possible overfitting. A noteworthy observation emerged during the evaluation where the model exhibited a tendency to detect objects in close proximity to the intended targets, leading to a decrease in its overall comprehension and performance. the observation described above is depicted in detail in Figure 7.139.

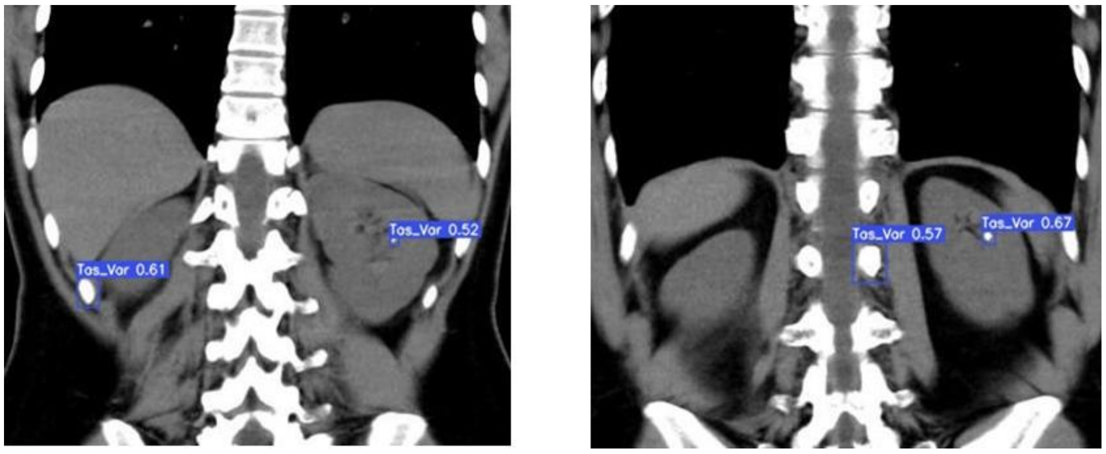


Figure 7.139. Model confusion between kidney stones and spinal cord.

In Figure 7.139, the model identified objects resembling kidney stones, including parts of the spinal cord. This misidentification led to numerous false predictions, resulting in a decline in overall performance. These false predictions align with FP in the confusion matrix. The second observation was that it's important to mention that the model struggled with accurate predictions for larger objects, contributing to an overall decrease in performance as shown in Figure 7.140.

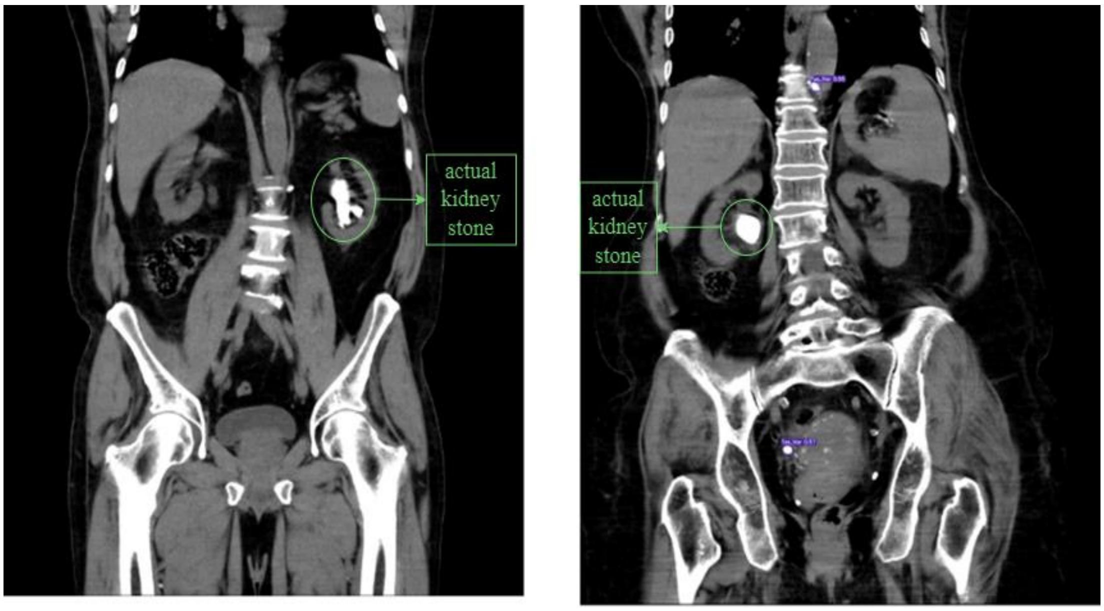


Figure 7.140. Model difficulty predicting larger kidney stones.

In Figure 7.140, the model faced difficulty predicting larger kidney stones, leading to an increase in FN due to its inability to draw bounding boxes for these objects. While these observations were made based on the current dataset, it’s crucial to acknowledge that they may differ depending on the quality and diversity of the dataset employed.

The third model used was CNN with RMSprop and Adam optimizers. It was trained with different epochs as 20, 40, 60, 80, and 100. The learning rate and batch size were set at 0.001 and 32. Among all the trained models using CNN, the best performing model achieved an accuracy of 99.13%. This result was attained after 40 epochs of training, utilizing the Adam optimizer. However, it was noticed that increasing the epochs may have led to an increase in FPs and FNs as shown in training curves for different epochs. As stated in section 6.4.3.1 for the optimization of the hyperparameters, too many tests were employed in order to get the best results of the model.

In the future work, increasing the dataset size for Faster R-CNN and YOLO is recommended to enhance their performance. Gathering data from diverse sources can contribute to better model generalization. Additionally, implementing CNN classification based on the size of kidney stones can be explored. For segmentation and classification, different deep learning models like U-Net and Mask R-CNN can be considered.

Table 7.21 presents a comparison of results between state-of-the-art studies and the proposed method based on kidney stone classification using CNN, with the same dataset.

Table 7.21. Comparative results of CNN based kidney stone classification

Authors	**Reference**	**Model**	**Accuracy (%)**
Patro et al.	[37]	DKN	98.56
Baygin et al.	[41]	ExDark19	99.71
Yildirim et al.	[43]	xResNet50	96.82
Proposed method		CNN	99.13

Table 7.22 shows a comparison of results between state-of-the-art studies and the proposed method using different datasets.

Table 7.22. Comparison of the proposed method with literature studies.

Authors	**Input images**	**Number of Images**	**Model**	**Results**
Razmjooy & Yan	CT	12446	DBN/FO-CHIO	Accuracy: 97.98%.
Caglayan et al.	CT	2959	xResNet50	Accuracy: 85%, 89%, and 93% in the sagittal plane.
Gurkan et al.	CT	658	YOLO v7	mAP (0.5): 0.85, Precision: 0.882, Sensitivity: 0.829, F1 score: 0.854.
Islam et al.	CT	12446	Swin transformers	Accuracy: 99.30%.
S. Sudharson & P. Kokil	Ultrasound	4940	SVM	Accuracy: 87.31%
Parakh et al.	CT	535	Dual CNN	AUC: 0.954
Längkvist et al.	CT	465	CNN	Sensitivity: 100%
Proposed method	CT	1799	CNN	Accuracy: 99.13 %

CHAPTER 8

CONCLUSION

In this thesis, advanced deep learning models were developed to detect and classify kidney stones in coronal CT images. The utilized dataset consisting of 1799 coronal CT scans. Out of these scans, 1009 were from individuals without kidney stones, and the remaining were from patients diagnosed with kidney stones. In addition to that, this thesis involved the training and testing of three distinct deep learning models, namely Faster R-CNN, YOLO, and the customized CNN.

According to the obtained results in this thesis, the Faster R-CNN results were less favorable. On the other hand, the YOLO v5 model exhibited promising outcomes in identifying kidney stones, surpassing the performance of YOLOv7. The YOLO v5 model demonstrated reasonable accuracy, detecting kidney stones, with a mAP (0.5) of 84.6% and a mAP (0.5:95) of 39.0% on the test set. In assessing the performance of the CNN, it’s noteworthy that the customized CNN model, trained over 40 epochs using the Adam optimizer with a learning rate of 0.001 and a batch size of 32, demonstrated the highest accuracy as 99.13%. This metric indicates the model’s effectiveness in correctly classifying instances within the dataset. The proposed model, employing a customized CNN, achieved an accuracy closely approaches to the top performing studies in the literature, marking it as a noteworthy achievement. Nevertheless, certain limitations and challenges were identified, emphasizing the need for future enhancements.

REFERENCES

1. Chewcharat, A. and Curhan, G., "Trends in the prevalence of kidney stones in the United States from 2007 to 2016", ***Urolithiasis***, 49 (1): 27–39 (2021).

2. Internet: "Definition & Facts for Kidney Stones - NIDDK", **https://www.niddk.nih.gov/health-information/urologic-diseases/kidney-stones/definition-facts** (2023).

3. Shin, S., Srivastava, A., Alli, N. A., and Bandyopadhyay, B. C., "Confounding risk factors and preventative measures driving nephrolithiasis global makeup", ***World Journal Of Nephrology***, 7 (7): 129 (2018).

4. Al-Shawi, M. M., Aljama, N. A., Aljedani, R., Alsaleh, M. H., Atyia, N., Alsedrah, A., and Albardi, M., "The Role of Radiological Imaging in the Diagnosis and Treatment of Urolithiasis: A Narrative Review", ***Cureus***, 14 (12): (2022).

5. Wood, K., Keys, T., Mufarrij, P., and Assimos, D. G., "Impact of Stone Removal on Renal Function: A Review", ***Reviews In Urology***, 13 (2): 73 (2011).

6. Shen, D., Wu, G., and Suk, H. Il, "Deep Learning in Medical Image Analysis", ***Annurev Bioeng***, 19: 221–248 (2017).

7. Zemouri, R., Zerhouni, N., and Racoceanu, D., "Deep learning in the biomedical applications: Recent and future status", ***Applied Sciences (Switzerland)***, 9 (8): (2019).

8. Shaheen, F., Verma, B., and Asafuddoula, M., "Impact of Automatic Feature Extraction in Deep Learning Architecture", ***International Conference On Digital Image Computing: Techniques And Applications***, (2016).

9. Chen, X. W. and Lin, X., "Big data deep learning: Challenges and perspectives", ***IEEE Access***, 2: 514–525 (2014).

10. Internet: "Your Kidneys & How They Work - NIDDK", **https://www.niddk.nih.gov/health-information/kidney-disease/kidneys-how-they-work** (2018).

11. Internet: "7 Things to Know About Kidney Function | National Kidney Foundation", **https://www.kidney.org/kidneydisease/howkidneyswrk** (2023).

12. Hall, J. E. and Hall, M. E., "Guyton and Hall Textbook of Medical Physiology E-Book", 14th. Ed., ***Elsevier***, 331–360 (2020).

13. Abboudi, H., Khan, M. S., Dasgupta, P., and Ahmed, K., "Simulation in Urology", Blandy’s Urology, 3. Ed., ***John Wiley & Sons, Ltd***, 27–38 (2019).

14. Choi, H. Y., Park, H. C., and Ha, S. K., "High Water Intake and Progression of Chronic Kidney Diseases", ***Electrolytes & Blood Pressure*** , 13 (2): 51 (2015).

15. Zulkar Nain, "(PDF) Genetic Anomalies in Kidney: Common Malformations and Dysfunctions", ***CreateSpace Independent Publishing Platform***, 2–3 (2016).

16. RM, S., D, P., and SW, L., "Anatomy, Abdomen and Pelvis, Kidneys", ***StatPearls***, (2023).

17. Delaney, M. A., Kowalewska, J., and Treuting, P. M., "Urinary System", Comparative Anatomy and Histology: A Mouse, Rat, and Human Atlas, Second Edition, ***Academic Press***, 275–301 (2018).

18. Khan, K. N. M., Hard, G. C., and Alden, C. L., "Kidney", Haschek and Rousseaux’s Handbook of Toxicologic Pathology, Third Edition: Volume 1-3, 3rd. Ed., ***Academic Press***, 1667–1773 (2013).

19. Mukoyama, M. and Nakao, K., "Hormones of the kidney", Endocrinology: Basic and Clinical Principles: Second Edition, ***Humana Press***, 353–365 (2005).

20. Shankar, A. S., Du, Z., Mora, H. T., van den Bosch, T. P. P., Korevaar, S. S., Van den Berg-Garrelds, I. M., Bindels, E., Lopez-Iglesias, C., Clahsen-van Groningen, M. C., Gribnau, J., Baan, C. C., Danser, A. H. J., Hoorn, E. J., and Hoogduijn, M. J., "Human kidney organoids produce functional renin", ***Kidney International***, 99 (1): 134–147 (2021).

21. Chen, T. K., Knicely, D. H., and Grams, M. E., "Chronic Kidney Disease Diagnosis and Management: A Review", ***JAMA***, 322 (13): 1294–1304 (2019).

22. Divatia, M., Ozcan, A., Guo, C. C., and Ro, J. Y., "Kidney Cancer: Recent Advances in Surgical and Molecular Pathology", First Edition., ***Springer Cham***, (2020).

23. Worcester, E. M. and Coe, F. L., "Nephrolithiasis", ***Primary Care***, 35 (2): 369–391 (2008).

24. Shah, J. and Whitfield, H. N., "Urolithiasis through the ages", ***BJU International***, 89 (8): 801–810 (2002).

25. Saigal, C. S., Joyce, G., and Timilsina, A. R., "Direct and indirect costs of nephrolithiasis in an employed population: opportunity for disease management?", ***Kidney International***, 68 (4): 1808–1814 (2005).

26. Liu, Y., Li, M., Qiang, L., Sun, X., Liu, S., and Lu, T. J., "Critical size of kidney stone through ureter: A mechanical analysis", ***Journal Of The Mechanical Behavior Of Biomedical Materials***, 135: (2022).

27. Peerapen, P. and Thongboonkerd, V., "Kidney Stone Prevention", ***Advances In Nutrition***, 14 (3): 555–569 (2023).

28. Asha, S. and Sunitha, J., "Kidney Stones Benefit of Natural Products", ***LAP LAMBERT Academic Publishing,*** 12–20 (2012).

29. Gillams, K., Juliebø-Jones, P., Juliebø, S. Ø., and Somani, B. K., "Gender Differences in Kidney Stone Disease (KSD): Findings from a Systematic Review", ***Current Urology Reports***, 22 (10): 3 (2021).

30. Stamatelou, K. and Goldfarb, D. S., "Epidemiology of Kidney Stones", ***Healthcare (Basel, Switzerland)***, 11 (3): (2023).

31. Curhan, G. C., Willett, W. C., Rimm, E. B., and Stampfer, M. J., "Family history and risk of kidney stones", ***Journal Of The American Society Of Nephrology : JASN***, 8 (10): 1568–1573 (1997).

32. Gamage, K. N., Jamnadass, E., Sulaiman, S. K., Pietropaolo, A., Aboumarzouk, O., and Somani, B. K., "The role of fluid intake in the prevention of kidney stone disease: A systematic review over the last two decades", ***Turkish Journal Of Urology***, 46 (Supp. 1): 92–103 (2020).

33. Khan, S. R., Pearle, M. S., Robertson, W. G., Gambaro, G., Canales, B. K., Doizi, S., Traxer, O., and Tiselius, H. G., "Kidney stones", ***Nature Reviews. Disease Primers***, 2: (2016).

34. McCarthy, C. J., Baliyan, V., Kordbacheh, H., Sajjad, Z., Sahani, D., and Kambadakone, A., "Radiology of renal stone disease", ***International Journal Of Surgery***, 36: 638–646 (2016).

35. Brisbane, W., Bailey, M. R., and Sorensen, M. D., "An overview of kidney stone imaging techniques", ***Nature Reviews. Urology***, 13 (11): 654–662 (2016).

36. Dogan, S., Akbal, E., Tuncer, T., and Acharya, U. R., "Application of substitution box of present cipher for automated detection of snoring sounds", ***Artificial Intelligence In Medicine***, 117: (2021).

37. Patro, K. K., Allam, J. P., Neelapu, B. C., Tadeusiewicz, R., Acharya, U. R., Hammad, M., Yildirim, O., and Pławiak, P., "Application of Kronecker convolutions in deep learning technique for automated detection of kidney stones with coronal CT images", ***Information Sciences***, 640: (2023).

38. Yan, C. and Razmjooy, N., "Kidney stone detection using an optimized Deep Believe network by fractional coronavirus herd immunity optimizer", ***Biomedical Signal Processing And Control***, 86: (2023).

39. Caglayan, A., Horsanali, M. O., Kocadurdu, K., Ismailoglu, E., and Guneyli, S., "Deep learning model-assisted detection of kidney stones on computed tomography", ***International Braz J Urol : Official Journal Of The Brazilian Society Of Urology***, 48 (5): 830–839 (2022).

40. BAYRAM, A. F., GURKAN, C., BUDAK, A., and KARATAŞ, H., "A Detection and Prediction Model Based on Deep Learning Assisted by Explainable Artificial Intelligence for Kidney Diseases", ***Avrupa Bilim Ve Teknoloji Dergisi***, 40: 67–74 (2022).

41. Baygin, M., Yaman, O., Barua, P. D., Dogan, S., Tuncer, T., and Acharya, U. R., "Exemplar Darknet19 feature generation technique for automated kidney stone detection with coronal CT images", ***Artificial Intelligence In Medicine***, 127: (2022).

42. Islam, M. N., Hasan, M., Hossain, M. K., Alam, M. G. R., Uddin, M. Z., and Soylu, A., "Vision transformer and explainable transfer learning models for auto detection of kidney cyst, stone and tumor from CT-radiography", ***Scientific Reports***, 12 (1): (2022).

43. Yildirim, K., Bozdag, P. G., Talo, M., Yildirim, O., Karabatak, M., and Acharya, U. R., "Deep learning model for automated kidney stone detection using coronal CT images", ***Computers In Biology And Medicine***, 135: (2021).

44. Sudharson, S. and Kokil, P., "Computer-aided diagnosis system for the classification of multi-class kidney abnormalities in the noisy ultrasound images", ***Computer Methods And Programs In Biomedicine***, 205: (2021).

45. Parakh, A., Lee, H., Lee, J. H., Eisner, B. H., Sahani, D. V., and Do, S., "Urinary Stone Detection on CT Images Using Deep Convolutional Neural Networks: Evaluation of Model Performance and Generalization", ***Radiology. Artificial Intelligence***, 1 (4): (2019).

46. Längkvist, M., Jendeberg, J., Thunberg, P., Loutfi, A., and Lidén, M., "Computer aided detection of ureteral stones in thin slice computed tomography volumes using Convolutional Neural Networks", ***Computers In Biology And Medicine***, 97: 153–160 (2018).

47. Liu, P. ran, Lu, L., Zhang, J. yao, Huo, T. tong, Liu, S. xiang, and Ye, Z. wei, "Application of Artificial Intelligence in Medicine: An Overview", ***Current Medical Science***, 41 (6): 1115 (2021).

48. Bansla, A. and Bansla, N., "Artificial intelligence", ***International Journal Of Applied Engineering Research***, 7 (11): (2012).

49. Sultan, A. S., Elgharib, M. A., Tavares, T., Jessri, M., and Basile, J. R., "The use of artificial intelligence, machine learning and deep learning in oncologic histopathology", ***Journal Of Oral Pathology & Medicine***, 49 (9): 849–856 (2020).

50. El Naqa, I. and Murphy, M. J., "What Is Machine Learning?", Machine Learning in Radiation Oncology, ***Springer International Publishing***, 3–11 (2015).

51. Yang, X. S., "Introduction to Algorithms for Data Mining and Machine Learning", Introduction to Algorithms for Data Mining and Machine Learning, 1st. Ed., ***Academic Press***, 1–173 (2019).

52. Rashidi, H. H., Tran, N. K., Betts, E. V., Howell, L. P., and Green, R., "Artificial Intelligence and Machine Learning in Pathology: The Present Landscape of Supervised Methods", ***Academic Pathology***, 6: (2019).

53. Jain, V. and Chatterjee, J. M., "Machine Learning with Health Care Perspective: Machine Learning and Healthcare", First Edition., ***Springer***, 1–25 (2020).

54. Osarogiagbon, A. U., Khan, F., Venkatesan, R., and Gillard, P., "Review and analysis of supervised machine learning algorithms for hazardous events in drilling operations", ***Process Safety And Environmental Protection***, 147: 367–384 (2021).

55. Eckhardt, C. M., Madjarova, S. J., Williams, R. J., Ollivier, M., Karlsson, J., Pareek, A., and Nwachukwu, B. U., "Unsupervised machine learning methods and emerging applications in healthcare", ***Knee Surgery, Sports Traumatology, Arthroscopy***, 31: 376–381 (2023).

56. McAlpine, E. D., Michelow, P., and Celik, T., "The Utility of Unsupervised Machine Learning in Anatomic Pathology", ***American Journal Of Clinical Pathology***, 157 (1): 5–14 (2022).

57. Ma, Y., Liu, K., Guan, Z., Xu, X., Qian, X., and Bao, H., "Background Augmentation Generative Adversarial Networks (BAGANs): Effective Data Generation Based on GAN-Augmented 3D Synthesizing", ***Symmetry,*** 10 (12): 734 (2018).

58. Hammoudeh, Ahmad. "A Concise Introduction to Reinforcement Learning" (2018).

59. Sarker, I. H., "Machine Learning: Algorithms, Real-World Applications and Research Directions", ***SN Computer Science***, 2: 1–21 (2021).

60. Hinton, G. E., Osindero, S., and Teh, Y. W., "A fast learning algorithm for deep belief nets", ***Neural Computation***, 18 (7): 1527–1554 (2006).

61. Lee, J. G., Jun, S., Cho, Y. W., Lee, H., Kim, G. B., Seo, J. B., and Kim, N., "Deep Learning in Medical Imaging: General Overview", ***Korean Journal Of Radiology***, 18 (4): 570–584 (2017).

62. Dastres, R. and Soori, M., "Artificial Neural Network Systems", ***International Journal Of Imaging And Robotics (IJIR)***, 21 (2): 13–25 (2021).

63. Maad M. Mijwil, " Artificial Neural Networks Advantages and Disadvantages", 2: (2018).

64. Internet: Ozal, Y., "GitHub - Yildirimozal/Kidney_stone_detection", **https://github.com/yildirimozal/Kidney_stone_detection** (2023).

65. Nabeel, S., " Research on Machine Learning in Python: Main Developments and Technology Trends in DS, ML, and AL", (2022).

66. Teoh, T. T. and Rong, Z., "Python for Artificial Intelligence. In: Artificial Intelligence with Python", Machine Learning: Foundations, Methodologies, and Applications, ***Springer Singapore***, Singapore, 3–7 (2022).

67. Carneiro, T., Da Nobrega, R. V. M., Nepomuceno, T., Bian, G. Bin, De Albuquerque, V. H. C., and Filho, P. P. R., "Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications", ***IEEE Access***, 6: 61677–61685 (2018).

68. Nelson, M. J. and Hoover, A. K., "Notes on Using Google Colaboratory in AI Education", ***Annual Conference On Innovation And Technology In Computer Science Education, ITiCSE***, 533–534 (2020).

69. Hidayah, A. H. N., Syafeeza, A. R., Razak, N. A., Saad, W. H. M., Wong, Y. C., and Naja, A. A., "Disease Detection of Solanaceous Crops Using Deep Learning for Robot Vision", ***Journal Of Robotics And Control (JRC)***, 3 (6): 790–799 (2022).

70. Shahriar, M. T. and Li, H., "A Study of Image Pre-processing for Faster Object Recognition", ***arXiv***, (2020).

71. Kottath, A. V. and Shri Bharathi, S. V., "Image Preprocessing Techniques in Skin Diseases Prediction using Deep Learning: A Review", ***4th International Conference On Inventive Research In Computing Applications, ICIRCA 2022 - Proceedings***, 1–6 (2022).

72. Saponara, S. and Elhanashi, A., "Impact of Image Resizing on Deep Learning Detectors for Training Time and Model Performance", ***Lecture Notes In Electrical Engineering***, 866: 10–17 (2022).

73. Galea, R. R., Diosan, L., Andreica, A., Popa, L., Manole, S., and Bálint, Z., "Region-of-Interest-Based Cardiac Image Segmentation with Deep Learning", ***Applied Sciences*** , 11 (4): 1965 (2021).

74. Sun, S. and Zhang, R., "Region of Interest Extraction of Medical Image based on Improved Region Growing Algorithm", ***Proceedings of the 2017 International Conference on Material Science, Energy and Environmental Engineering*** , (2017).

75. Mumuni, A. and Mumuni, F., "Data augmentation: A comprehensive survey of modern approaches", ***Array***, 16: (2022).

76. Doppala, B. P., Vamsi, B., Bhattacharyya, D., and Rao, J. N., "A Review of Image Annotation Tools for Object Detection", ***Proceedings - International Conference On Applied Artificial Intelligence And Computing, ICAAIC 2022***, 976–982 (2022).

77. Patel, S. and Patel, A., "Object Detection with Convolutional Neural Networks", Machine Learning for Predictive Analysis, Proceedings of ICTIS 2020, ***Springer Nature Singapore***, 529–539 (2021).

78. Diwan, T., Anirudh, G., and Tembhurne, J. V., "Object detection using YOLO: challenges, architectural successors, datasets and applications", ***Multimedia Tools And Applications***, 82 (6): 9243–9275 (2023).

79. Chen, Y., Goorden, M. C., Beekman, F. J., Du, L., Zhang, R., and Wang, X., "Overview of two-stage object detection algorithms", ***Journal Of Physics: Conference Series***, 1544 (1): 012033 (2020).

80. Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., and Qu, R., "A survey of deep learning-based object detection", ***IEEE Access***, 7: 128837–128868 (2019).

81. Chen, Y., Goorden, M. C., Beekman, F. J., and Li, W., "Analysis of Object Detection Performance Based on Faster R-CNN", ***Journal Of Physics: Conference Series***, 1827 (1): 012085 (2021).

82. Aniyan, A. K., Thorat, K., Hakim, H., and Fadhil, A., "Survey: Convolution Neural networks in Object Detection", ***Journal Of Physics: Conference Series***, 1804 (1): 012095 (2021).

83. Krishna Sai, B. N. and Sasikala, T., "Object Detection and Count of Objects in Image using Tensor Flow Object Detection API", ***2019 International Conference On Smart Systems And Inventive Technology (ICSSIT)***, 542–546 (2019).

84. Internet: "TensorBoard | TensorFlow", **https://www.tensorflow.org/tensorboard** (2023).

85. Qureshi, R., RAGAB, M. G., ABDULKADER, S. J., muneer, amgad, ALQUSHAIB, A., SUMIEA, E. H., and Alhussian, H., "A Comprehensive Systematic Review of YOLO for Medical Object Detection (2018 to 2023)", ***TechRxiv***, (2023).

86. Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You Only Look Once: Unified, Real-Time Object Detection", ***Proceedings Of The IEEE Computer Society Conference On Computer Vision And Pattern Recognition***, 2016-December: 779–788 (2015).

87. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A., "The pascal visual object classes (VOC) challenge", ***International Journal Of Computer Vision***, 88 (2): 303–338 (2010).

88. M. Hussain, ‘‘Yolo-v1 to yolo-v8, the rise of yolo and its complementary nature toward digital manufacturing and industrial defect detection,’’ ***Machines***, vol. 11, no. 7, p. 677, 2023

89. Terven, J. R. and Cordova-Esparza, D. M., "A Comprehensive Review of YOLO: From YOLOv1 and Beyond", ***ArXiv***, (2023).

90. Internet: Glenn, Jocher, " Ultralytics/Yolov5 · GitHub", **https://github.com/ultralytics/yolov5/blob/master/README.md** (2023).

91. Wu, B., Pang, C., Zeng, X., and Hu, X., "ME-YOLO: Improved YOLOv5 for Detecting Medical Personal Protective Equipment", ***Applied Sciences,*** 12 (23): 11978 (2022).

92. Antonakakis, M., Tzavaras, A., Tsakos, K., Spanakis, E. G., Sakkalis, V., Zervakis, M., and Petrakis, E. G. M., "Real-Time Object Detection using an Ultra-High-Resolution Camera on Embedded Systems", ***IST 2022 - IEEE International Conference On Imaging Systems And Techniques, Proceedings***, (2022).

93. Mehmood, F., Ahmad, S., and Whangbo, T. K., "An Efficient Optimization Technique for Training Deep Neural Networks", ***Mathematics 2023, Vol. 11, Page 1360***, 11 (6): 1360 (2023).

94. Internet: “Optimizers", **https://keras.io/api/optimizers/** (2023).

95. Kingma, D. P. and Ba, J. L., "Adam: A Method for Stochastic Optimization", ***3rd International Conference On Learning Representations, ICLR 2015 - Conference Track Proceedings***, (2014).

96. Kaya, Ö., Çodur, M. Y., and Mustafaraj, E., "Automatic Detection of Pedestrian Crosswalk with Faster R-CNN and YOLOv7", ***Buildings,*** 13 (4): 1070 (2023).

97. Li, K. ;, Wang, Y. ;, Hu, Z., Fischer, S., Li, K., Wang, Y., and Hu, Z., "Improved YOLOv7 for Small Object Detection Algorithm Based on Attention and Dynamic Convolution", ***Applied Sciences*** , 13 (16): 9316 (2023).

98. LeCun, Y., Bengio, Y., & Hinton, G., “Deep learning”, **Nature**, 521(7553), 436-444 (2015).

99. Rguibi, Z., Hajami, A., Zitouni, D., Elqaraoui, A., and Bedraoui, A., "CXAI: Explaining Convolutional Neural Networks for Medical Imaging Diagnostic", ***Electronics,*** 11 (11): 1775 (2022).

100. Varlı, M., “Derin Öğrenme Tabanli Epileptik Nöbet Teşhisi”, (master’s dissertation) (2022)

101. Internet: Dharmaraj, "Zero-Padding in Convolutional Neural Networks | by Dharmaraj | Medium", **https://medium.com/@draj0718/zero-padding-in-convolutional-neural-networks-bf1410438e99** (2023).

102. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions", ***Journal Of Big Data***, 8 (1): 53 (2021).

103. Indolia, S., Goswami, A. K., Mishra, S. P., and Asopa, P., "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach", ***International Conference On Computational Intelligence And Data Science***, 132: 679–688 (2018).

104. Aurélien Géron, "Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems", O’Reilly Media, ***O’Reilly Media***, 587–657 (2019).

105. KILIÇARSLAN, S., Kemal, A. D. E. M., & Çelik, M., “An overview of the activation functions used in deep learning algorithms”, ***Journal of New Results in Science***, 10(3), 75-88 (2021).

106. Jemshi, K. M., Sreelekha, G., Sathidevi, P. S., Mohanachandran, P., and Vinekar, A., "Plus disease classification in Retinopathy of Prematurity using transform based features", ***Multimedia Tools And Applications***, 1–31 (2023).

107. M, H. and M.N, S., "A Review on Evaluation Metrics for Data Classification Evaluations", ***International Journal Of Data Mining & Knowledge Management Process***, 5 (2): 01–11 (2015).

108. Han, X., Zhong, Y., Zhang, L., Wang, L., Di, L., Du, Q., Liu, P., and Thenkabail, P. S., "An Efficient and Robust Integrated Geospatial Object Detection Framework for High Spatial Resolution Remote Sensing Imagery", ***Remote Sesing*** , 9 (7): 666 (2017).

109. Padilla, R., Netto, S. L., and Da Silva, E. A. B., "A Survey on Performance Metrics for Object-Detection Algorithms", ***International Conference On Systems, Signals, And Image Processing***, 237–242 (2020).

110. Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., and Da Silva, E. A. B., "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit", ***Electronics***, 10 (3): 279 (2021).

111. Ajayi, O. G. and Ashi, J., "Effect of varying training epochs of a Faster Region-Based Convolutional Neural Network on the Accuracy of an Automatic Weed Classification Scheme", ***Smart Agricultural Technology***, 3: 100128 (2023).

BIOGRAPHY

Aziz AYDIN completed his high school studies at al Najah secondary school in Kuwait in 2015. He completed his undergraduate education in Medical Engineering at Karabuk University between 2016 and 2021, including the first year of preparation. In September 2021, he started his master's degree in Biomedical Engineering at Karabuk University. His current research interests are machine learning, deep learning, image processing, Python, MATLAB, and programming languages.