A

Mini Project Report on

# A Recurrent CNN for Automatic Detection and Classification of Coronary Artery Plaque and Stenosis in Coronary CT Angiography

*Submitted for partial fulfilment of the requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**in**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**by**


| | |
|---|---|
| **BHUPENDRA KUMAR** | **19K81A05C4** |
| **MOHAMMAD ANWAR** | **19K81A05F8** |
| **SAGAR SHARMA** | **19K81A05G7** |


Under the Guidance of

**Mrs. E. Soumya**

**ASSOCIATE PROFESSOR**


## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
Affiliated to JNTUH, Approved by AICTE,
Accredited by NBA & NAAC A+, ISO 9001:2008 Certified
Dhulapally, Secunderabad - 500 100


**NOVEMBER - 2022**

# St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
NBA & NAAC A+ Accredited
Dhulapally, Secunderabad - 500 100
www.smec.ac.in

## Certificate

This is to certify that the project entitled **"A recurrent CNN for automatic detection and classification of coronary artery plaque and stenosis in coronary CT angiography"** is being submitted By **BHUPENDRA KUMAR (19K81A05C4), MOHAMMED ANWAR (19K81A05F8), SAGAR SHARMA (19K81A05G7),** in fulfilment of the requirement for the award of degree of **BACHELOR OF TECHNOLOGY in DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** is recorded of bonafide work carried out by them. The result embodied in this report have been verified and found satisfactory.

**Guide**
**Mrs. E. Soumya**
**Associate Professor**
**Department of CSE**

**Head of the Department**
**Dr. R. SANTHOSHKUMAR**
**Professor & Head**
**Department of CSE**

**Internal Examiner** **External Examiner**

**Date:**

**Place:**

# St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
NBA & NAAC A+ Accredited
Dhulapally, Secunderabad - 500 100
www.smec.ac.in


## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## DECLARATION

We, the students of '**Bachelor of Technology in Department of Computer Science and Engineering'**, session: 2019 - 2023**, St. Martin's Engineering College, Dhulapally, Kompally, Secunderabad,** hereby declare that the work presented in this Project Work entitled "**A recurrent CNN for automatic detection and classification of coronary artery plaque and stenosis in coronary CT angiography"** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. This result embodied in this project report has not been submitted in any university for award of any degree.

| | |
|---|---|
| **BHUPENDRA KUMAR** | **19K81A05C4** |
| **MOHAMMED ANWAR** | **19K81A05F8** |
| **SAGAR SHARMA** | **19K81A05G7** |

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowded our efforts with success.

We extend our deep sense of gratitude to Principal, **Dr. P. SANTOSH KUMAR PATRA**, St. Martin's Engineering College Dhulapally, for permitting us to undertake this project.

We are also thankful to **Dr. R.SANTHSOHKUMAR**, Head of the Department, Department of Computer Science and Engineering, St. Martin's Engineering College, Dhulapally, Secunderabad. for his support and guidance throughout our project as well as our Project Coordinators **Dr. B. RAJALINGAM**, Associate Professor and **Mrs. E. SOUMYA**, Associate Professor, Department of Computer Science and Engineering for their valuable support.

We would like to express our sincere gratitude and indebtedness to our project supervisor **Dr. B.Rajalingam**, Associate Professor, Department of Computer Science and Engineering, St. Martins Engineering College, Dhulapally, for his support and guidance throughout our project.

Finally, we express thanks to all those who have helped us successfully completing this project. Furthermore, we would like to thank our family and friends for their moral support and encouragement. We express thanks to all those who have helped us in successfully completing the project.



**BHUPENDRA KUMAR** **19K81A05C4**

**MOHAMMED ANWAR** **19K81A05F8**

**SAGAR SHARMA** **19K81A05G7**

# ABSTRACT

Various types of atherosclerotic plaque and varying grades of stenosis could lead to different management of patients with coronary artery disease. Therefore, it is crucial to detect and classify the type of coronary artery plaque, as well as to detect and determine the degree of coronary artery stenosis. This study includes retrospectively collected clinically obtained coronary CT angiography (CCTA) scans of 163 patients. In these, the centrelines of the coronary arteries were extracted and used to reconstruct multi-planar reformatted (MPR) images for the coronary arteries. To define the reference standard, the presence and the type of plaque in the coronary arteries (no plaque, non-calcified, mixed, calcified), as well as the presence and the anatomical significance of coronary stenosis (no stenosis, non-significant i.e. < 50% luminal narrowing, significant i.e. ≥ 50% luminal narrowing) were manually annotated in the MPR images by identifying the start- and end-points of the segment of the artery affected by the plaque. To perform automatic analysis, a multi-task recurrent convolutional neural network is applied on coronary artery MPR images. First, a 3D convolutional neural network is utilized to extract features along the coronary artery. Subsequently, the extracted features are aggregated by a recurrent neural network that performs two simultaneous multiclass classification tasks. In the first task, the network detects and characterizes the type of the coronary artery plaque. In the second task, the network detects and determines the anatomical significance of the coronary artery stenosis. The network was trained and tested using CCTA images of 98 and 65 patients, respectively. For detection and characterization of coronary plaque, the method achieved an accuracy of 0.77. For detection of stenosis and determination of its anatomical significance, the method achieved an accuracy of 0.80. The results demonstrate that automatic detection and classification of coronary artery plaque and stenosis are feasible. This may enable automated triage of patients to those without coronary plaque and those with coronary plaque and stenosis in need for further cardiovascular workup.

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ACRONYMS AND DEFINITIONS

| S.NO | ACRONYM | DEFINITION |
|---|---|---|
| 01. | CNN | Convolutional Neural Network |
| 02. | RNN | Recurrent Neural Network |
| 03. | CT | Computed Tomography |
| 04. | CCTA | Coronary CT Angiography |
| 05. | CAD | Coronary Artery Disease |
| 06. | RCNN | Recurrent Convolutional Neural Network |
| 07. | MPR | Multi-Planar Reformation |
| 08. | CAC | Coronary Artery Calcification |
| 09. | CVD | Cardiovascular Disease |
| 10. | MDCT | Multidetector Computed Tomography |

# CONTENTS

# CHAPTER 1

# INTRODUCTION

## 1.1 Objective of the Project

Various types of atherosclerotic plaque and varying grades of stenosis could lead to different management of patients with coronary artery disease. Therefore, it is crucial to detect and classify the type of coronary artery plaque, as well as to detect and determine the degree of coronary artery stenosis. This study includes retrospectively collected clinically obtained coronary CT angiography (CCTA) scans of 163 patients. In these, the centrelines of the coronary arteries were extracted and used to reconstruct multi-planar reformatted (MPR) images for the coronary arteries. To define the reference standard, the presence and the type of plaque in the coronary arteries (no plaque, non-calcified, mixed, calcified), as well as the presence and the anatomical significance of coronary stenosis (no stenosis, no significance i.e. < 50% luminal narrowing, significant i.e. ≥ 50% luminal narrowing) were manually annotated in the MPR images by identifying the start- and end-points of the segment of the artery affected by the plaque. To perform automatic analysis, a multi-task recurrent convolutional neural network is applied on coronary artery MPR images. First, a 3D convolutional neural network is utilized to extract features along the coronary artery. Subsequently, the extracted features are aggregated by a recurrent neural network that performs two simultaneous multiclass classification tasks. In the first task, the network detects and characterizes the type of the coronary artery plaque. In the second task, the network detects and determines the anatomical significance of the coronary artery stenosis. The network was trained and tested using CCTA images of 98 and 65 patients, respectively. For detection and characterization of coronary plaque, the method achieved an accuracy of 0.77. For detection of stenosis and determination of its anatomical significance, the method achieved an accuracy of 0.80. The results demonstrate that automatic detection and classification of coronary artery plaque and stenosis are feasible. This may enable automated triage of patients to those without coronary plaque and those with coronary plaque and stenosis in need for further cardiovascular workup

# CHAPTER 2

# LITERATURE SURVEY

### 1. "Heart disease and stroke statistics - 2016 update,"

The amount of coronary artery calcification (CAC) is a strong and independent predictor of cardiovascular disease (CVD) events. In clinical practice, CAC is manually identified and automatically quantified in cardiac CT using commercially available software. This is a tedious and time-consuming process in large-scale studies. Therefore, a number of automatic methods that require no interaction and semiautomatic methods that require very limited interaction for the identification of CAC in cardiac CT have been proposed. Thus far, a comparison of their performance has been lacking. The objective of this study was to perform an independent evaluation of (semi)automatic methods for CAC scoring in cardiac CT using a publicly available standardized framework.

### 2. "SCCT guidelines for the interpretation and reporting of coronary computed tomographic angiography,"

The majority of patients with acute coronary syndromes (ACS) present with unstable angina, acute myocardial infarction, and sudden coronary death. The most common cause of coronary thrombosis is plaque rupture followed by plaque erosion, whereas calcified nodule is infrequent. If advances in coronary disease are to occur, it is important to recognize the precursor lesion of ACS. Of the three types of coronary thrombosis, a precursor lesion for acute rupture has been postulated. The non-thrombosed lesion that most resembles the acute plaque rupture is the thin cap fibroatheroma (TCFA), which is characterized by a necrotic core with an overlying fibrous cap measuring <65 microm, containing rare smooth muscle cells but numerous macrophages. Thin cap fibroatheromas are most frequently observed in patients dying with acute myocardial infarction and least common in plaque erosion. They are most frequently observed in proximal coronary arteries, followed by mid and distal major coronary arteries. Vessels demonstrating TCFA do not usually show severe narrowing but show positive remodeling. In TCFAs the necrotic core length is approximately 2 to 17 mm (mean 8 mm) and the underlying cross-sectional area narrowing in over 75% of cases is <75% (diameter stenosis <50%). The area of the necrotic core in at least 75% of cases is < or =3 mm2. These lesions have lesser degree of calcification than plaque ruptures. Thin cap fibroatheromas are common in

patients with high total cholesterol (TC) and high TC/high-density lipoprotein cholesterol ratio, in women >50 years, and in those patients with elevated high levels of high sensitivity C-reactive protein. It has only recently been recognized that their identification in living patients might help reduce the incidence of sudden coronary death.

**3. "An evaluation of automatic coronary artery calcium scoring methods with cardiac CT using the orca Score framework,"**

The amount of coronary artery calcification (CAC) is a strong and independent predictor of cardiovascular disease (CVD) events. In clinical practice, CAC is manually identified and automatically quantified in cardiac CT using commercially available software. This is a tedious and time-consuming process in large-scale studies. Therefore, a number of automatic methods that require no interaction and semiautomatic methods that require very limited interaction for the identification of CAC in cardiac CT have been proposed. Thus far, a comparison of their performance has been lacking. The objective of this study was to perform an independent evaluation of (semi)automatic methods for CAC scoring in cardiac CT using a publicly available standardized framework.

**4. "Pathology of the vulnerable plaque"**

The majority of patients with acute coronary syndromes (ACS) present with unstable angina, acute myocardial infarction, and sudden coronary death. The most common cause of coronary thrombosis is plaque rupture followed by plaque erosion, whereas calcified nodule is infrequent. If advances in coronary disease are to occur, it is important to recognize the precursor lesion of ACS. Of the three types of coronary thrombosis, a precursor lesion for acute rupture has been postulated. The non-thrombosed lesion that most resembles the acute plaque rupture is the thin cap fibroatheroma (TCFA), which is characterized by a necrotic core with an overlying fibrous cap measuring <65 microm, containing rare smooth muscle cells but numerous macrophages. Thin cap fibroatheromas are most frequently observed in patients dying with acute myocardial infarction and least common in plaque erosion. They are most frequently observed in proximal coronary arteries, followed by mid and distal major coronary arteries. Vessels demonstrating TCFA do not usually show severe narrowing but show positive remodeling. In TCFAs the necrotic core length is approximately 2 to 17 mm (mean 8 mm) and the underlying cross-sectional area narrowing in over 75% of cases is <75% (diameter stenosis <50%). The area of the necrotic core in at least 75% of cases is < or =3 mm2. These lesions have lesser

degree of calcification than plaque ruptures. Thin cap fibroatheromas are common in patients with high total cholesterol (TC) and high TC/high-density lipoprotein cholesterol ratio, in women >50 years, and in those patients with elevated high levels of high sensitivity C-reactive protein. It has only recently been recognized that their identification in living patients might help reduce the incidence of sudden coronary death.

**5. "Chronic coronary artery disease: diagnosis and management,"**

Coronary artery disease (CAD) is the single most common cause of death in the developed world, responsible for about 1 in every 5 deaths. The morbidity, mortality, and socioeconomic importance of this disease make timely accurate diagnosis and cost-effective management of CAD of the utmost importance. This comprehensive review of the literature highlights key elements in the diagnosis, risk stratification, and management strategies of patients with chronic CAD. Relevant articles were identified by searching the PubMed database for the following terms: *chronic coronary artery disease* or *stable angina*. Novel imaging modalities, pharmacological treatment, and invasive (percutaneous and surgical) interventions have revolutionized the current treatment of patients with chronic CAD. Medical treatment remains the cornerstone of management, but revascularization continues to play an important role. In the current economic climate and with health care reform very much on the horizon, the issue of appropriate use of revascularization is important, and the indications for revascularization, in addition to the relative benefits and risks of a percutaneous vs a surgical approach, are discussed.

**6. "Standardized evaluation framework for evaluating coronary artery stenosis detection, stenosis quantification and lumen segmentation algorithms in computed tomography angiography,"**

Though conventional coronary angiography (CCA) has been the standard of reference for diagnosing coronary artery disease in the past decades, computed tomography angiography (CTA) has rapidly emerged, and is nowadays widely used in clinical practice. Here, we introduce a standardized evaluation framework to reliably evaluate and compare the performance of the algorithms devised to detect and quantify the coronary artery stenoses, and to segment the coronary artery lumen in CTA data. The objective of this evaluation framework is to demonstrate the feasibility of dedicated algorithms to: (1) (semi-)automatically detect and quantify stenosis on CTA, in

comparison with quantitative coronary angiography (QCA) and CTA consensus reading, and (2) (semi-)automatically segment the coronary lumen on CTA, in comparison with expert's manual annotation. A database consisting of 48 multicenter multivendor cardiac CTA datasets with corresponding reference standards are described and made available. The algorithms from 11 research groups were quantitatively evaluated and compared. The results show that (1) some of the current stenosis detection/quantification algorithms may be used for triage or as a second-reader in clinical practice, and that (2) automatic lumen segmentation is possible with a precision similar to that obtained by experts.

**7. Quantification of coronary arterial stenosis by multidetector CT angiography in comparison with conventional angiography: methods, caveats, and implications,"**

Multidetector computed tomography (MDCT) is a rapidly evolving technology for performing noninvasive coronary angiography. Despite good sensitivity and specificity for detecting significant coronary artery disease in patients, disagreement on individual coronary arterial stenosis severity is common between MDCT and the current gold standard, conventional angiography. The reasons for such disagreement are numerous, but are at least partly inherent to MDCT's modest spatial and temporal resolution at present. Less well acknowledged, however, is the fact that MDCT and conventional angiography are fundamentally different technologies, rendering good agreement on the degree of lumen narrowing rather unrealistic, given both of their respective limitations. Discrepant stenosis assessment by MDCT and conventional angiography receives remarkable attention, whereas its significance for patient outcome is less certain. On the other hand, the ability to noninvasively assess coronary arterial plaque characteristics and composition in addition to lumen obstruction shows strong promise for improved risk assessment and may at last enable us to move beyond mere coronary stenosis assessment for the management of patients with coronary artery disease.

**8. "Automatic coronary artery calcium scoring in cardiac CT angiography using paired convolutional neural networks,"**

The amount of coronary artery calcification (CAC) is a strong and independent predictor of cardiovascular events. CAC is clinically quantified in cardiac calcium

scoring CT (CSCT), but it has been shown that cardiac CT angiography (CCTA) may also be used for this purpose. We present a method for automatic CAC quantification in CCTA. This method uses supervised learning to directly identify and quantify CAC without a need for coronary artery extraction commonly used in existing methods. The study included cardiac CT exams of 250 patients for whom both a CCTA and a CSCT scan were available. To restrict the volume-of-interest for analysis, a bounding box around the heart is automatically determined. The bounding box detection algorithm employs a combination of three ConvNets, where each detects the heart in a different orthogonal plane (axial, sagittal, coronal). These ConvNets were trained using 50 cardiac CT exams. In the remaining 200 exams, a reference standard for CAC was defined in CSCT and CCTA. Out of these, 100 CCTA scans were used for training, and the remaining 100 for evaluation of a voxel classification method for CAC identification. The method uses ConvPairs, pairs of convolutional neural networks (ConvNets). The first ConvNet in a pair identifies voxels likely to be CAC, thereby discarding the majority of non-CAC-like voxels such as lung and fatty tissue. The identified CAC-like voxels are further classified by the second ConvNet in the pair, which distinguishes between CAC and CAC-like negatives. Given the different task of each ConvNet, they share their architecture, but not their weights. Input patches are either 2.5D or 3D. The ConvNets are purely convolutional, i.e. no pooling layers are present and fully connected layers are implemented as convolutions, thereby allowing efficient voxel classification. The performance of individual 2.5D and 3D ConvPairs with input sizes of 15 and 25 voxels, as well as the performance of ensembles of these ConvPairs, were evaluated by a comparison with reference annotations in CCTA and CSCT. In all cases, ensembles of ConvPairs outperformed their individual members. The best performing individual ConvPair detected 72% of lesions in the test set, with on average 0.85 false positive (FP) errors per scan. The best performing ensemble combined all ConvPairs and obtained a sensitivity of 71% at 0.48 FP errors per scan. For this ensemble, agreement with the reference mass score in CSCT was excellent (ICC 0.944 [0.918-0.962]). Aditionally, based on the Agatston score in CCTA, this ensemble assigned 83% of patients to the same cardiovascular risk category as reference CSCT. In conclusion, CAC can be accurately automatically identified and quantified in CCTA using the proposed pattern recognition method. This might obviate the need to acquire a dedicated CSCT scan for CAC scoring, which is regularly acquired prior to a CCTA, and thus reduce the CT radiation dose received by patients.

**9. "Automatic calcium scoring in low-dose chest CT using deep neural networks with dilated convolutions,"**

Heavy smokers undergoing screening with low-dose chest CT are affected by cardiovascular disease as much as by lung cancer. Low-dose chest CT scans acquired in screening enable quantification of atherosclerotic calcifications and thus enable identification of subjects at increased cardiovascular risk. This paper presents a method for automatic detection of coronary artery, thoracic aorta, and cardiac valve calcifications in low-dose chest CT using two consecutive convolutional neural networks. The first network identifies and labels potential calcifications according to their anatomical location and the second network identifies true calcifications among the detected candidates. This method was trained and evaluated on a set of 1744 CT scans from the National Lung Screening Trial. To determine whether any reconstruction or only images reconstructed with soft tissue filters can be used for calcification detection, we evaluated the method on soft and medium/sharp filter reconstructions separately. On soft filter reconstructions, the method achieved $F_1$ scores of 0.89, 0.89, 0.67, and 0.55 for coronary artery, thoracic aorta, aortic valve, and mitral valve calcifications, respectively. On sharp filter reconstructions, the $F_1$ scores were 0.84, 0.81, 0.64, and 0.66, respectively. Linearly weighted kappa coefficients for risk category assignment based on per subject coronary artery calcium were 0.91 and 0.90 for soft and sharp filter reconstructions, respectively. These results demonstrate that the presented method enables reliable automatic cardiovascular risk assessment in all low-dose chest CT scans acquired for lung cancer screening.

**10. "Fully convolutional deep-learning based system for coronary calcium score prediction from non-contrast chest CT,"**

The amount of calcium deposits in the coronary arteries is an important biomarker of cardiovascular disease. Coronary calcium has traditionally been quantified as an Agatston score using ECG-synchronized cardiac CT. Coronary calcium is rarely quantified from general chest CT scans, of which nearly 10 Million are performed in the US annually. We present an automatic method based on fully-convolutional deep neural network to segment coronary calcium and predict Agatston score from any non-contrast chest CT. We experimented with an internal dataset acquired through partnership with a large health organization in Israel. The dataset is composed of 1054 Chest CTs and reflects a variety of originating institutions, acquisition devices and

manufacturers. In comparison to expert manual annotations, our algorithm achieved a Pearson correlation coefficient of 0.98. Bland-Altman analysis demonstrated a bias of 0.4 with 95% limits of agreement of [-189.9-190.7]). Our linearly weighted Kappa results are 0.89 for Agatston risk category assignment. We also applied our method on a very large (14,365 subjects) cohort from the National Lung Screening Trial (NLST). We demonstrate correlation of the algorithm predictions with cardiovascular-related clinical outcomes.

**11. "An evaluation of automatic coronary artery calcium scoring methods with cardiac CT using the orca Score framework,"**

Coronary artery centerline extraction in cardiac CT angiography (CCTA) images is a prerequisite for evaluation of stenoses and atherosclerotic plaque. In this work, we propose an algorithm that extracts coronary artery centerlines in CCTA using a convolutional neural network (CNN). In the proposed method, a 3D dilated CNN is trained to predict the most likely direction and radius of an artery at any given point in a CCTA image based on a local image patch. Starting from a single seed point placed manually or automatically anywhere in a coronary artery, a tracker follows the vessel centerline in two directions using the predictions of the CNN. Tracking is terminated when no direction can be identified with high certainty. The CNN is trained using manually annotated centerlines in training images. No image preprocessing is required, so that the process is guided solely by the local image values around the tracker's location. The CNN was trained using a training set consisting of 8 CCTA images with a total of 32 manually annotated centerlines provided in the MICCAI 2008 Coronary Artery Tracking Challenge (CAT08). Evaluation was performed within the CAT08 challenge using a test set consisting of 24 CCTA test images in which 96 centerlines were extracted. The extracted centerlines had an average overlap of 93.7% with manually annotated reference centerlines. Extracted centerline points were highly accurate, with an average distance of 0.21 mm to reference centerline points. Based on these results the method ranks third among 25 publicly evaluated methods in CAT08. In a second test set consisting of 50 CCTA scans acquired at our institution (UMCU), an expert placed 5448 markers in the coronary arteries, along with radius measurements. Each marker was used as a seed point to extract a single centerline, which was compared to the other markers placed by the expert. This showed strong correspondence between extracted centerlines and manually placed markers. In a third test set containing 36 CCTA scans from the MICCAI 2014

Challenge on Automatic Coronary Calcium Scoring (orCaScore), fully automatic seeding and centerline extraction was evaluated using a segment-wise analysis. This showed that the algorithm is able to fully-automatically extract on average 92% of clinically relevant coronary artery segments. Finally, the limits of agreement between reference and automatic artery radius measurements were found to be below the size of one voxel in both the CAT08 dataset and the UMCU dataset. Extraction of a centerline based on a single seed point required on average 0.4 ± 0.1 s and fully automatic coronary tree extraction required around 20 s. The proposed method is able to accurately and efficiently determine the direction and radius of coronary arteries based on information derived directly from the image data. The method can be trained with limited training data, and once trained allows fast automatic or interactive extraction of coronary artery trees from CCTA images.

# CHAPTER 3

# SYSTEM ANALYSIS

## 3.1 EXISTING SYSTEM

Your doctor may diagnose CAD in a number of ways. These may include blood tests, blood pressure tests, and EKG. If these tests indicate CAD, further imaging tests may be necessary. Not all chest pain and shortness of breath are related to heart disease. However, since CAD is serious, it's important to get it checked out to be sure. Other conditions.

**Disadvantage**

1.Time taken process

2. less accuracy

## 3.2 PROPOSED SYSTEM

The care of patients with coronary artery disease may differ depending on the types of atherosclerotic plaque present and the degree of stenosis. Therefore, it is imperative to identifyas well as to categorise the different types of coronary artery plaque.find coronary artery stenosis and assess its severity.This study involves clinical data that was retrospectively gathered andimages of 163 patients' coronary CT angiography (CCTA). those areWe retrieved and utilised the coronary artery centerlines.for the purpose of reassembling multi-planar reformatted (MPR) images,heart arteries. To specify the benchmark, the presence ofRegarding the presence or absence of plaque in the coronary arteries,(calcified, mixed, and non-calcified), as well as the availability and theanatomy of coronary stenosis (no stenosis, nonsignificant, 50% luminal narrowing,

**Advantage**

1.More Accuracy.

2.Quick response.

**MODULES**

To implement this project we have designed following modules

1) **Upload CCTA Scan Plaque Dataset**: using this module we will upload CCTA images to application

2) **Dataset Pre processing:** using this module we will read all images and then resize all images to equal size and then normalize all images values and then split dataset images into train and test where application using 80% images for training 20% for testing
3) **Generate & Load RCNN Model**: using this module we will train RCNN using 80% images and then test RCNN performance by applying 20% test images prediction
4) **Plaque Classification**: using this module we will upload test image and then RCNN will predict presence of Plaque and stenosis.
5) **RCNN Training Graph:** using this module we will plot RCNN training accuracy and loss graph

### 3.3. PROCESS MODEL USED WITH JUSTIFICATION

**SDLC (Umbrella Model):**

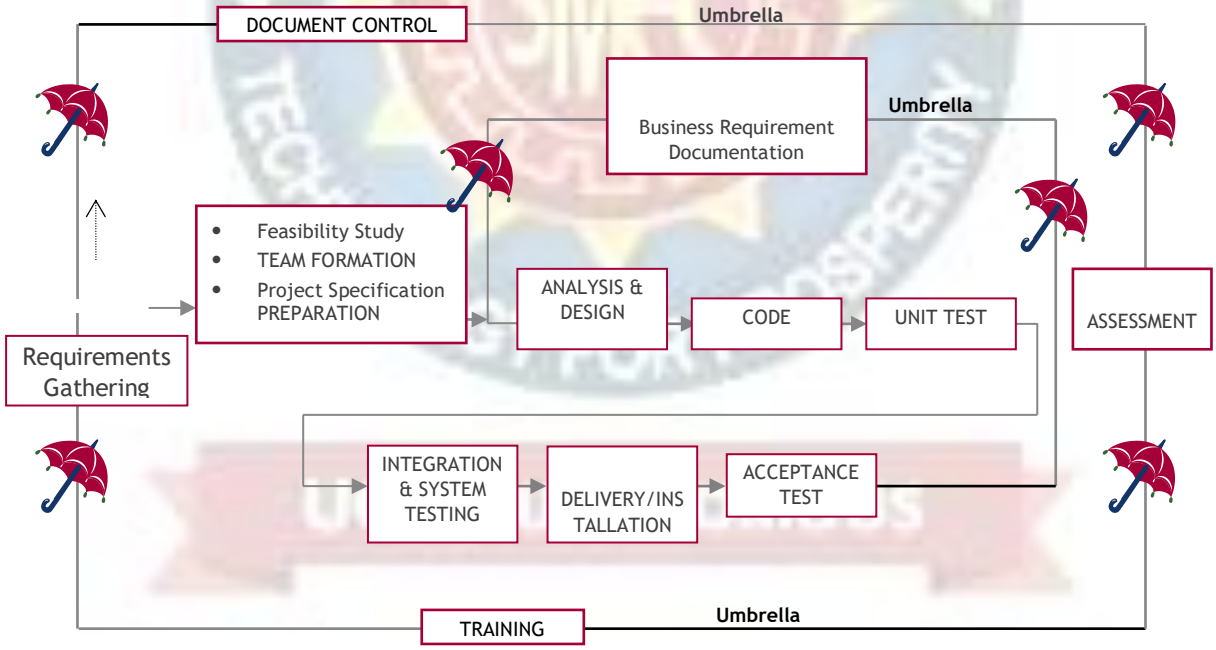


Fig3.1 Umbrella Model

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

- Requirement Gathering
- Analysis
- Designing
- Coding
- Testing
- Maintenance

**Requirements Gathering stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

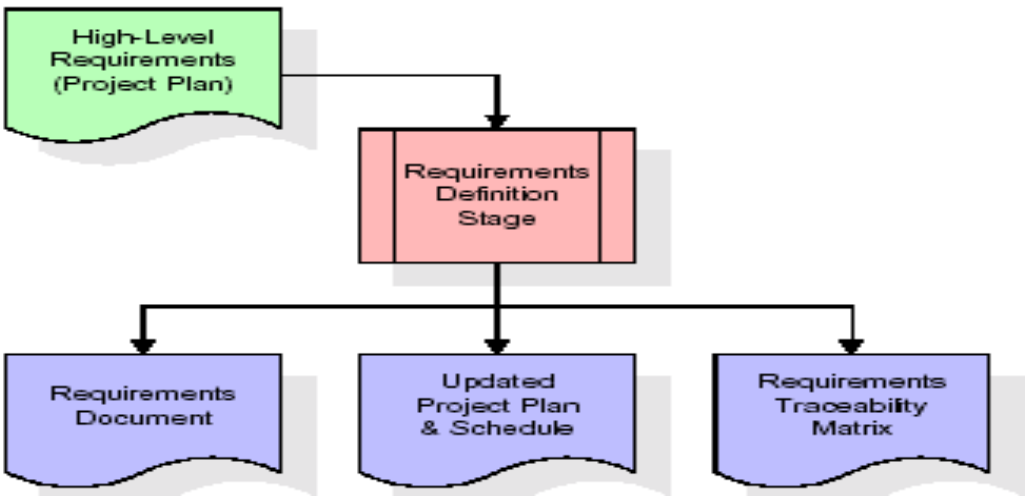


Fig 3.2 Requirement Gathering Stage

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- Feasibility study is all about identification of problems in a project.
- No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
- Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

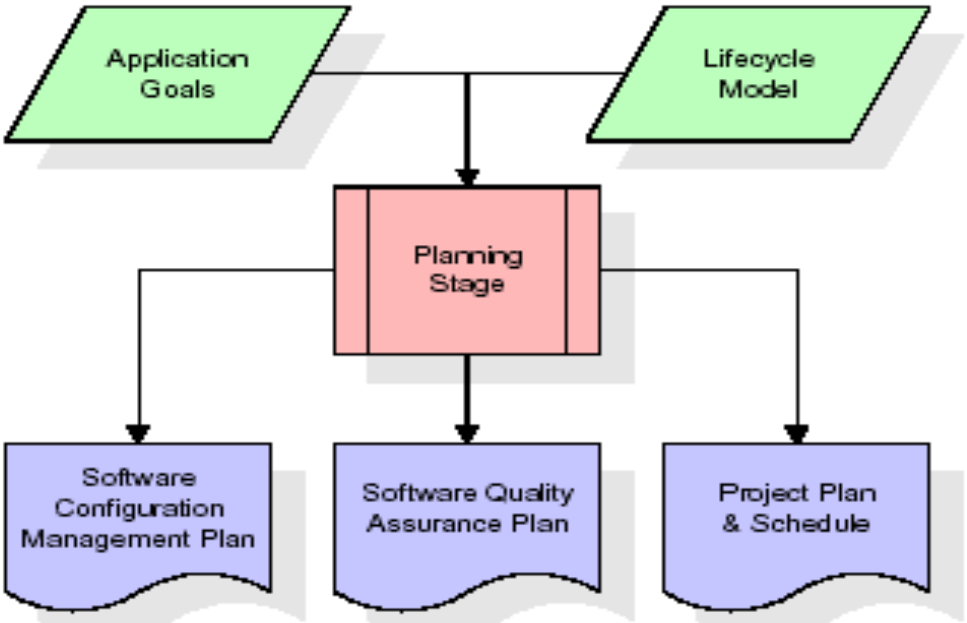


Fig 3.3 Analysis Stage

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

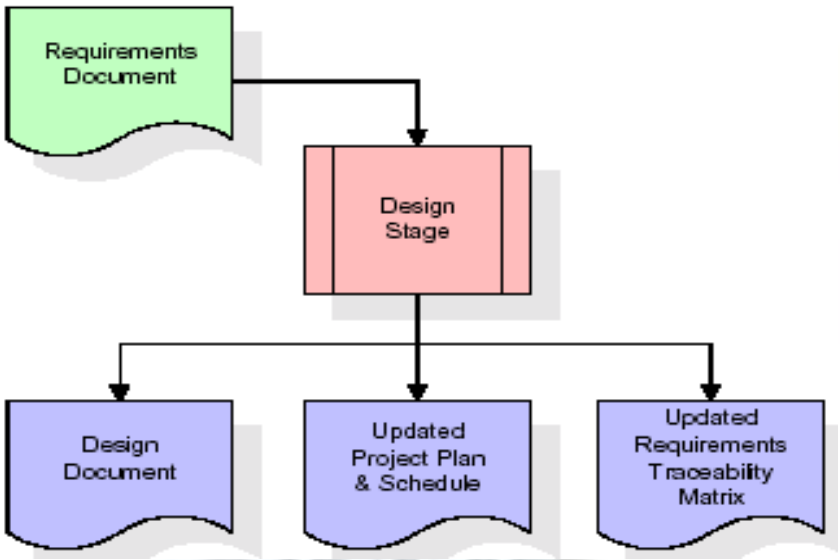


Fig 3.4 Designing Stage

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artefacts will be produced. Software artefacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artefacts, and an online help system will be developed to guide users in their interactions with the software.

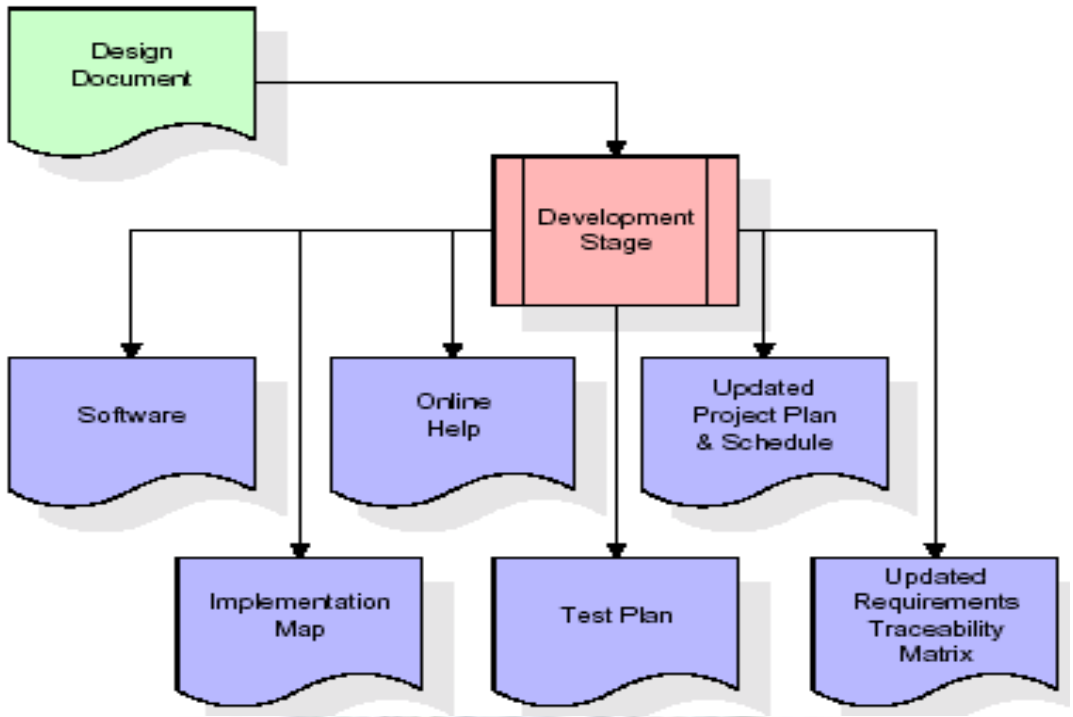


Fig 3.5 Development Stage

The RTM will be updated to show that each developed artefact is linked to a specific design element, and that each developed artefact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artefacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

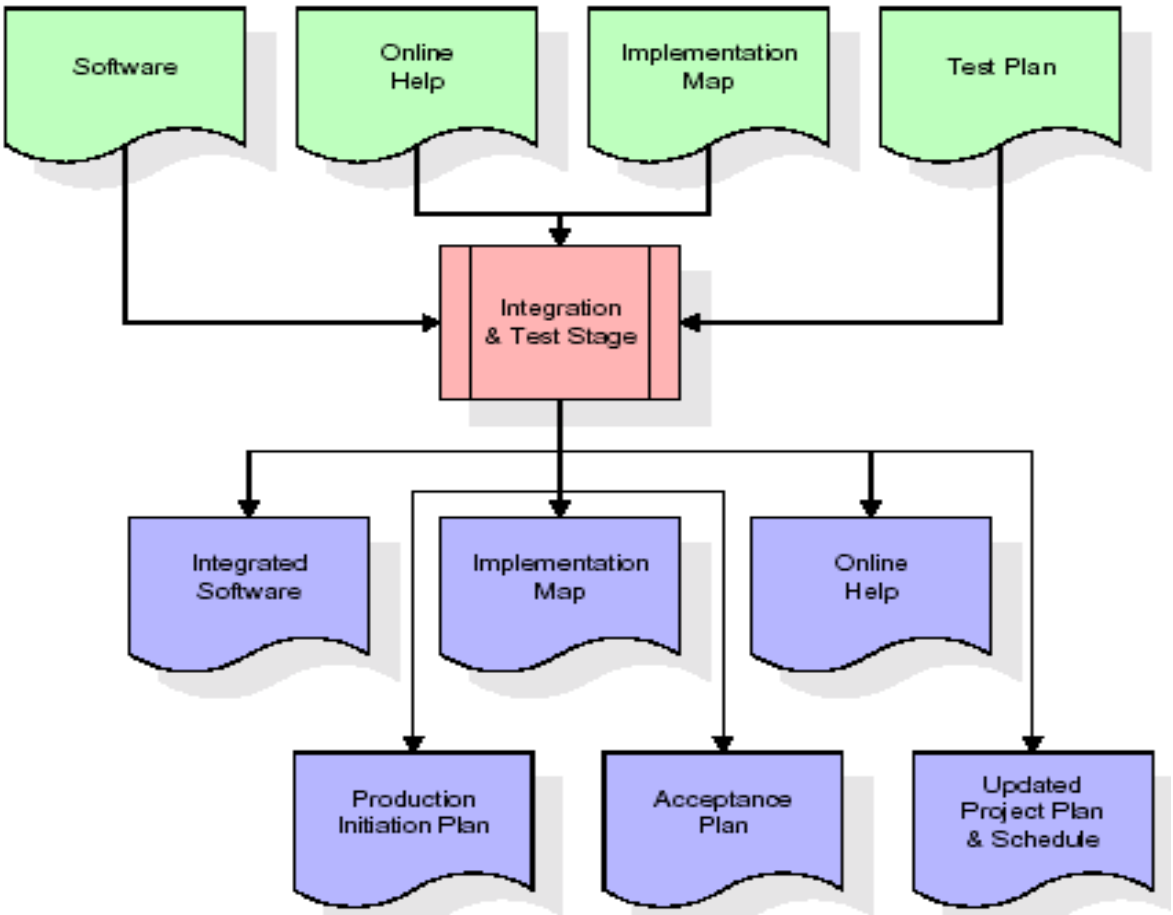


Fig 3.6 Integration & Test Stage

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

♦ **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

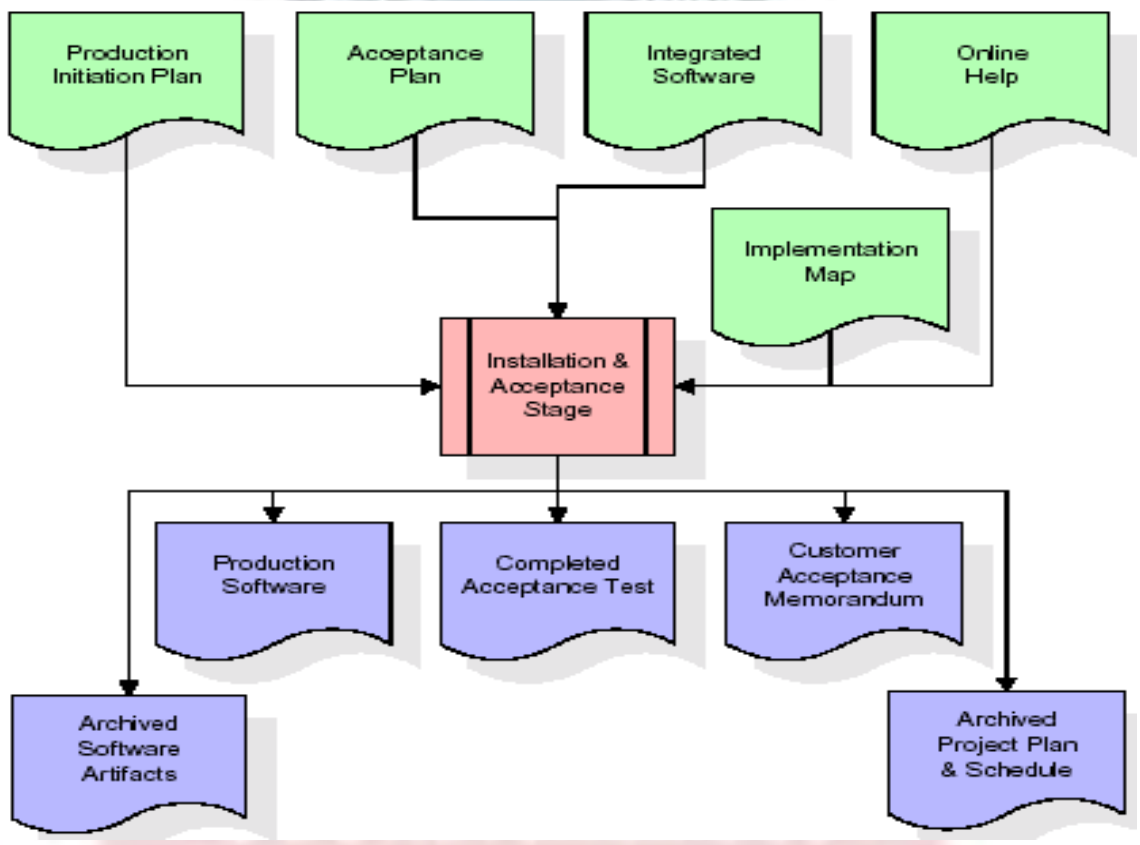


Fig 3.7 Installation & Acceptance Test

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

### 3.4. Software Requirement Specification

#### 3.4.1. Overall Description

A Software Requirements Specification (SRS) – a requirements specification for a software system is a complete description of the behaviour of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.
- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite

time. There are aspects in the feasibility study portion of the preliminary investigation:

- **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

- **OPERATIONAL FEASIBILITY**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

- **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

### 3.4.2. External Interface Requirements

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**HARDWARE REQUIREMENTS:**

- Processor - Intel i3(min)
- Speed - 1.1 Ghz
- RAM - 4GB DDR4(min)
- Hard Disk - 500GB(min)
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

- Operating System - Windows10/above
- Programming Language - Python 3.7/above

# CHAPTER 4

# SYSTEM DESIGN

### 4.1 UML Diagram:

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- **User Model View**
  i. This view represents the system from the users perspective.
  ii. The analysis representation describes a usage scenario from the end-users perspective.
- **Structural Model view**
  i. In this model the data and functionality are arrived from inside the system.
  ii. This model view models the static structures.
- **Behavioural Model View**

  It represents the dynamic of behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.
- **Implementation Model View**

  In this the structural and behavioural as parts of the system are represented as they are to be built.

- **Environmental Model View**

  In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

### 4.1.1 Class Diagram:

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

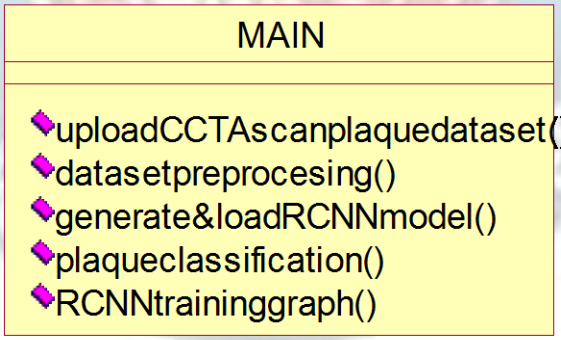


Fig 4.1 Class Diagram

### 4.1.2 Use case Diagram:

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

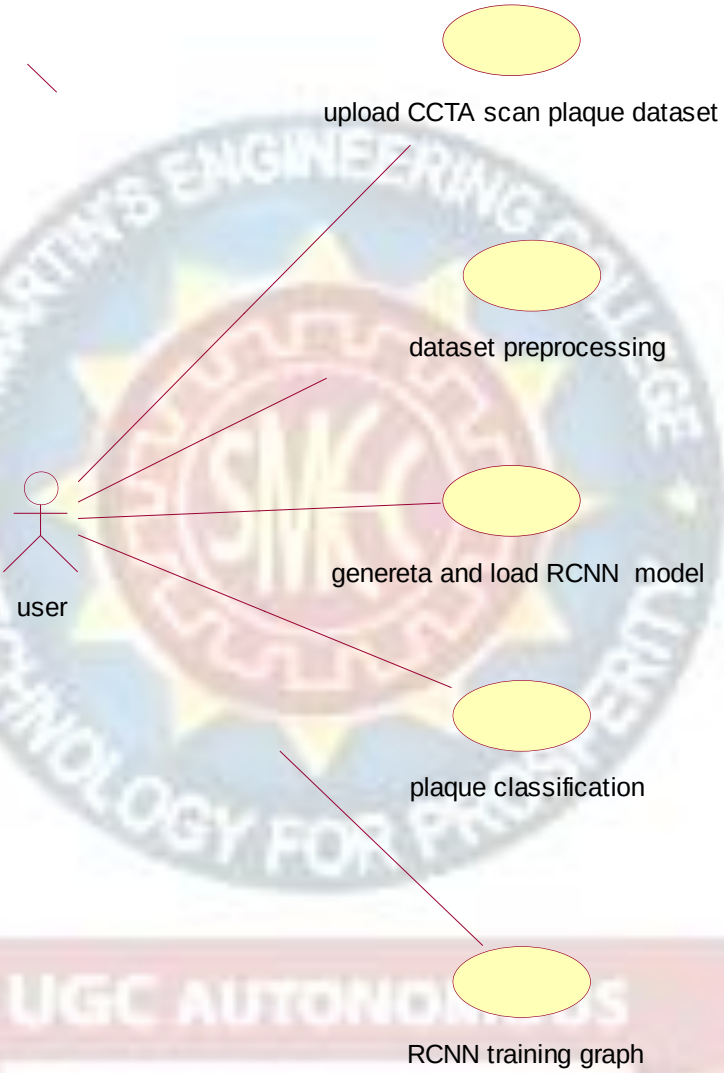


Fig 4.2 Use Case Diagram

### 4.1.3 Sequence diagram:

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.

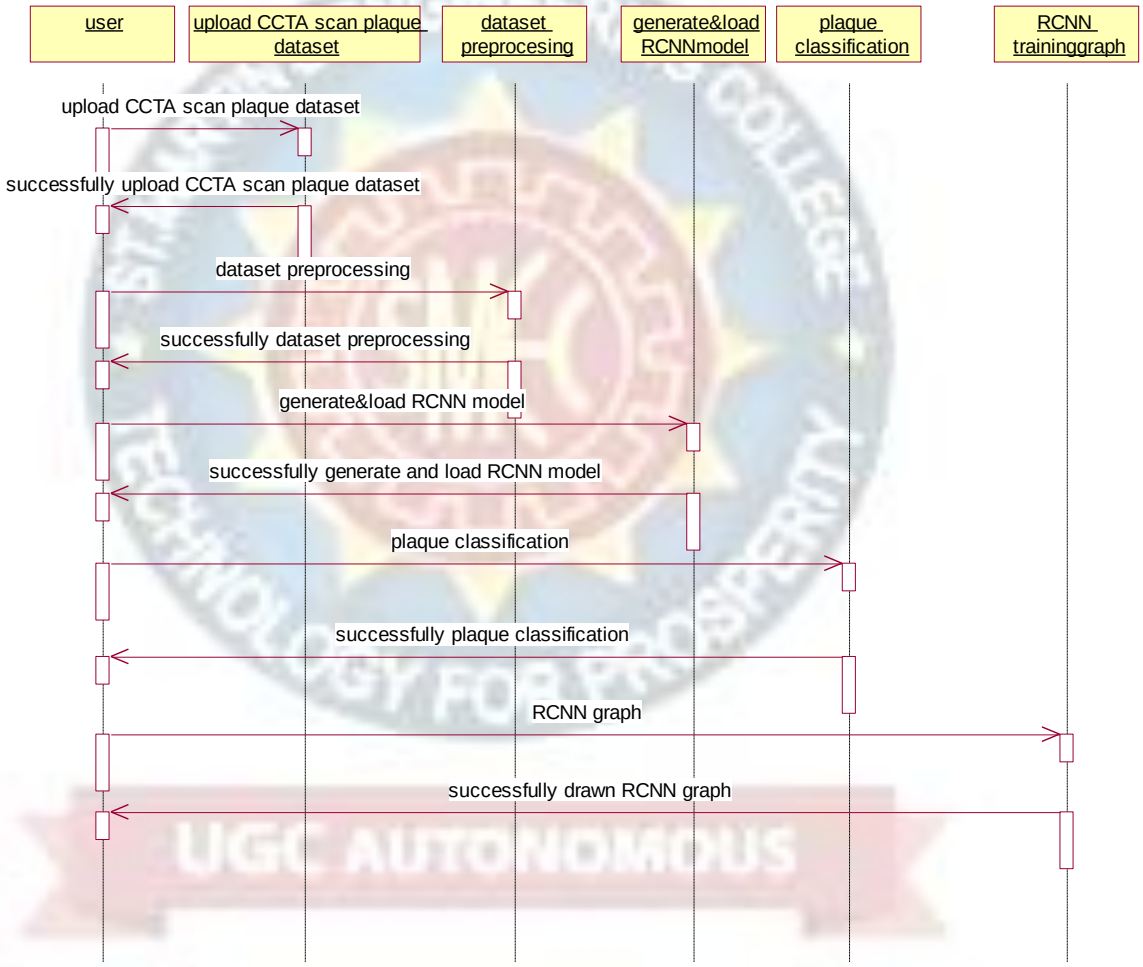


Fig 4.3 Sequence Diagram

### 4.1.4 Collaboration diagram:

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.

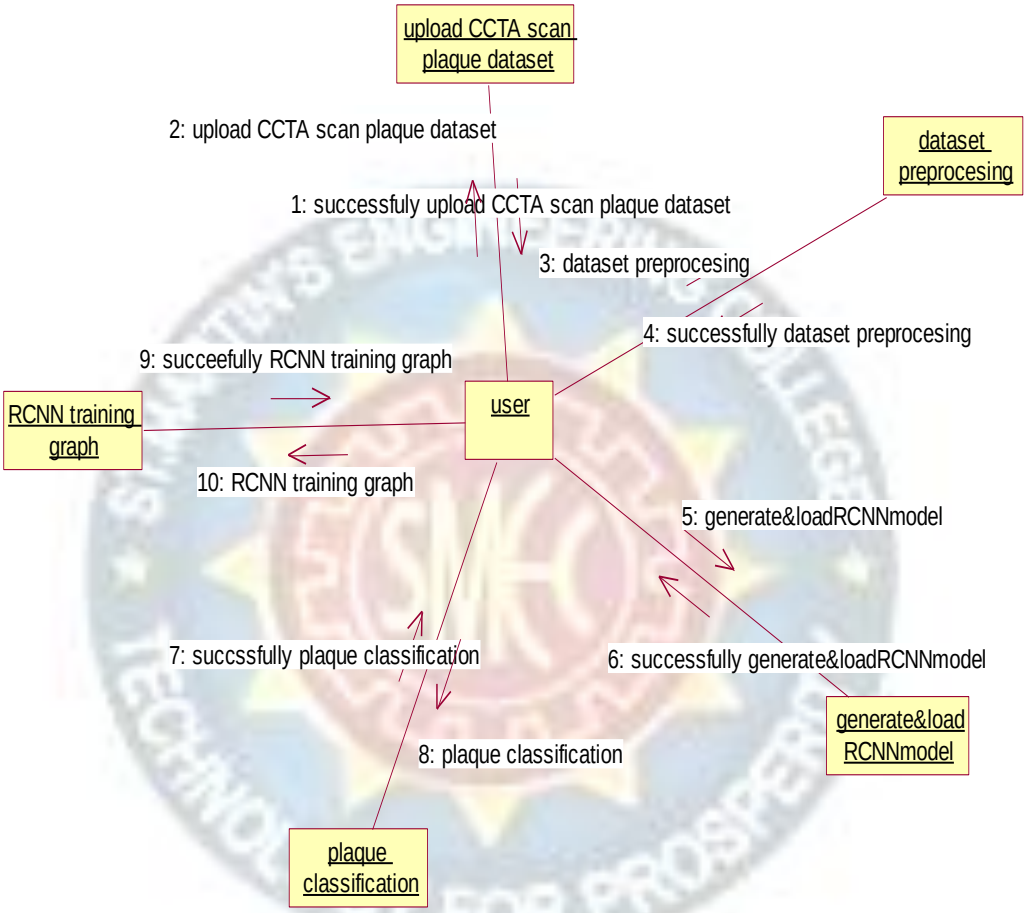


Fig 4.4 Collaboration Diagram

### 4.1.5 Component Diagram:

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

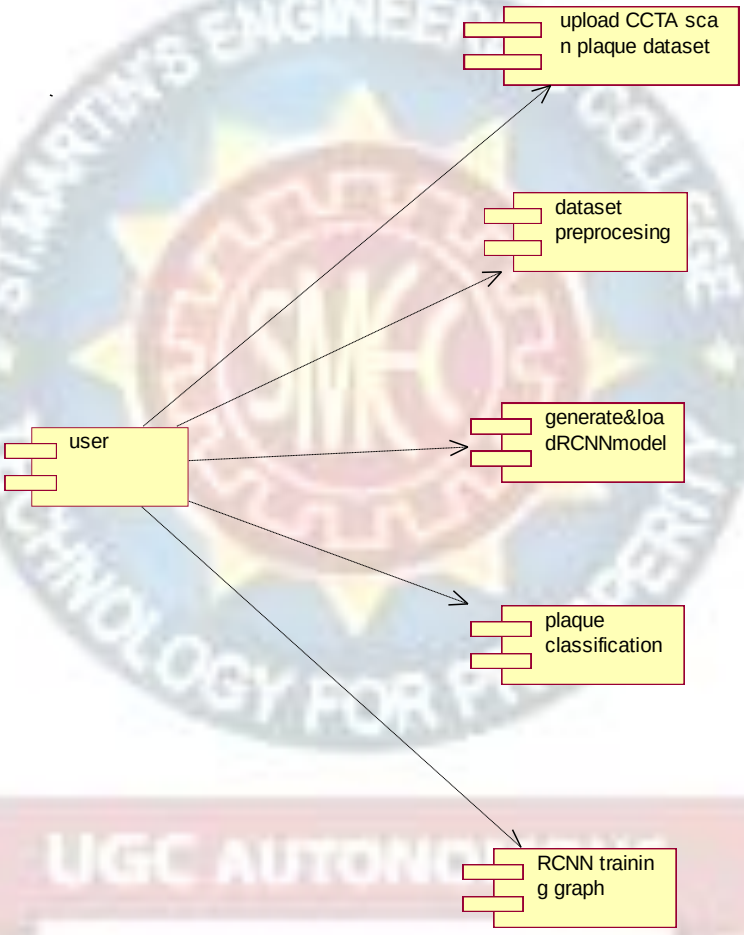


Fig 4.5 Component Diagram

### 4.1.6 Deployment Diagram:

A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

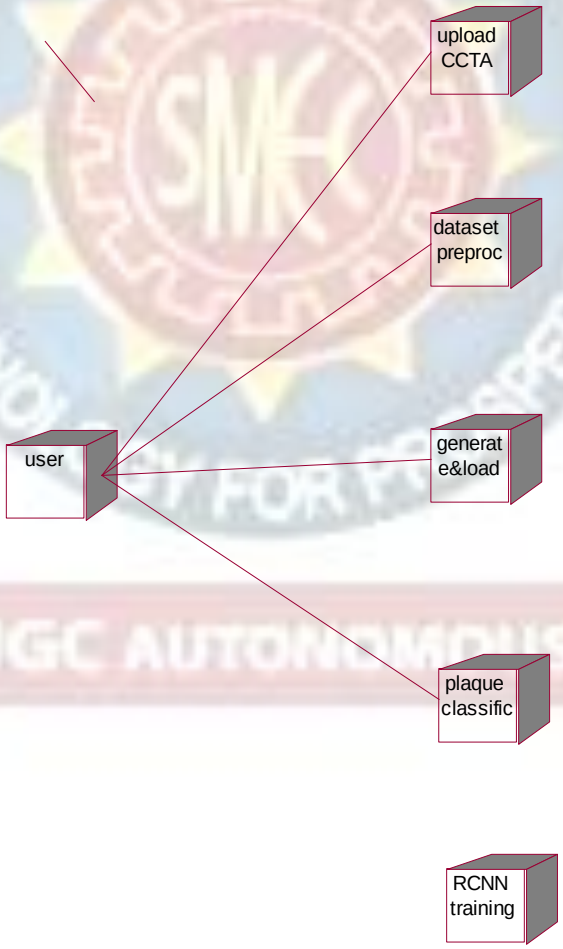


Fig 4.6 Deployment Diagram

### 4.1.7 Activity Diagram:

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent

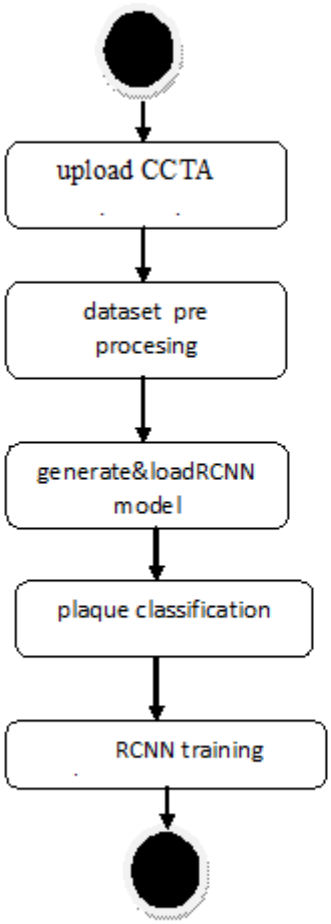


Fig 4.7 Activity Diagram

### 4.1.8 Data Flow Diagram:

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

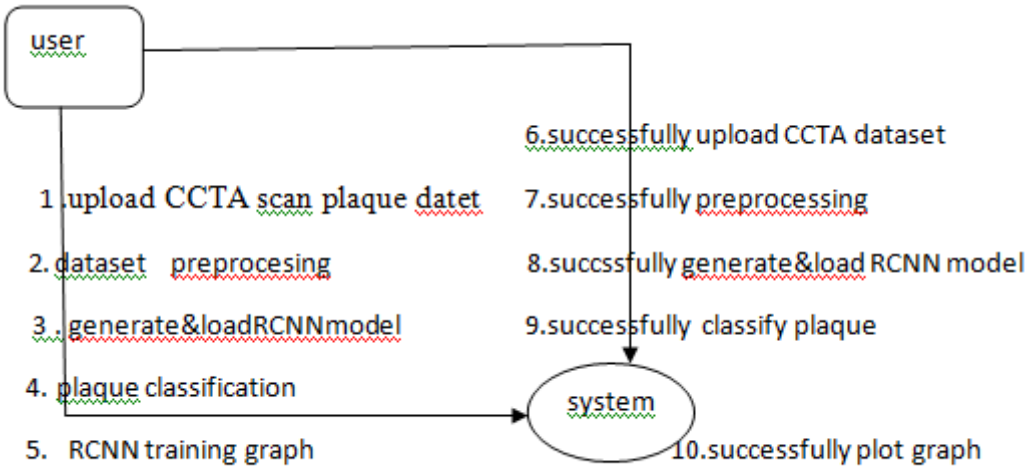


Fig 4.8 Data Flow Diagram

# CHAPTER 5

# IMPLEMETATION

## 5.1 Python

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software's written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

```
a = 2

b = 3

sum = a + b

print(sum)
```

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

```
from tkinter import messagebox

from tkinter import *

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import accuracy_score

from sklearn.model_selection import train_test_split

import os

from sklearn.metrics import confusion_matrix

from sklearn.metrics import precision_score

from sklearn.metrics import recall_score

from sklearn.metrics import f1_score

import seaborn as sns

from skimage.metrics import structural_similarity

import cv2

from keras.utils.np_utils import to_categorical

from keras.layers import  MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D
```

```
from keras.models import Sequential

from keras.models import model_from_json

import pickle

global filename

global X,Y

global X_train, X_test, y_train, y_test

global classifier, text

main = tkinter.Tk()

main.title("A Recurrent CNN for Automatic Detection and Classification of Coronary Artery Plaque and Stenosis in Coronary CT Angiography") #designing main screen
main.geometry("1300x1200")

#traffic names VPN and NON-VPN

class_labels = ['No Plaque','Plaque']

#fucntion to upload dataset

def uploadDataset():

    global filename, text

    text.delete('1.0', END)

    filename = filedialog.askdirectory(initialdir=".")

    text.insert(END,filename+" loaded\n\n")

def DataPreprocessing():

    global X, Y

    global filename, text

    global X_train, X_test, y_train, y_test

    text.delete('1.0', END)
```

```
if os.path.exists("model/X.txt.npy"):
    X = np.load("model/X.txt.npy")
    Y = np.load("model/Y.txt.npy")
else:
    X = []
    Y = []
    for root, dirs, directory in os.walk(filename):
        for j in range(len(directory)):
            name = os.path.basename(root)
            if 'Thumbs.db' not in directory[j]:
                img = cv2.imread(root+"/"+directory[j])
                img = cv2.resize(img, (64,64))
                im2arr = np.array(img)
                im2arr = im2arr.reshape(64,64,3)
                X.append(im2arr)
                lbl = 0
                if name == 'Plaque':
                    lbl = 1
                Y.append(lbl)
                print(name+" "+root+"/"+directory[j]+" "+str(lbl))
    X = np.asarray(X)
    Y = np.asarray(Y)
    X = X.astype('float32')
```

```
        X = X/255
        indices = np.arange(X.shape[0])
        np.random.shuffle(indices)
        X = X[indices]
        Y = Y[indices]
        Y = to_categorical(Y)
        np.save('model/X.txt',X)
        np.save('model/Y.txt',Y)
    text.insert(END,"Total classes found in dataset : "+str(class_labels)+"\n")
    text.insert(END,"Total images found in dataset : "+str(X.shape[0])+"\n\n")
    text.insert(END,"Dataset train & test split. 80% images used for training and 20% images used for testing\n\n")
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
    text.insert(END,"Training Images Size : "+str(X_train.shape[0])+"\n")
    text.insert(END,"Testing Images Size : "+str(X_test.shape[0])+"\n")
    text.update_idletasks()
    test = X[3]
    cv2.imshow("Sample Processed Image",cv2.resize(test,(300,300)))
    cv2.waitKey(0)
def runRCNN():
    global X, Y
    global text, classifier
    global X_train, X_test, y_train, y_test
```

```
    text.delete('1.0', END)
    if os.path.exists('model/model.json'):
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)
        json_file.close()
        classifier.load_weights("model/model_weights.h5")
        classifier._make_predict_function()
    else:
        classifier = Sequential()
        classifier.add(Convolution2D(32, 3, 3, input_shape = (64, 64, 3), activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (2, 2)))
        classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (2, 2)))
        classifier.add(Flatten())
        classifier.add(Dense(output_dim = 256, activation = 'relu'))
        classifier.add(Dense(output_dim = Y.shape[1], activation = 'softmax'))
        classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
        hist = classifier.fit(X_train, y_train, batch_size=16, epochs=10, shuffle=True, verbose=2, validation_data=(X_test,y_test))
        classifier.save_weights('model/model_weights.h5')
        model_json = classifier.to_json()
```

```
    with open("model/model.json", "w") as json_file:
        json_file.write(model_json)
    json_file.close()
    f = open('model/history.pckl', 'wb')
    pickle.dump(hist.history, f)
    f.close()
predict = classifier.predict(X_test)
predict = np.argmax(predict, axis=1)
for i in range(0,3):
    predict[i] = 0
y_test = np.argmax(y_test, axis=1)
p = precision_score(y_test, predict,average='macro') * 100
r = recall_score(y_test, predict,average='macro') * 100
f = f1_score(y_test, predict,average='macro') * 100
a = accuracy_score(y_test,predict)*100
text.insert(END,'RCNN Plaque Classification Accuracy  : '+str(a)+"\n")
text.insert(END,'RCNN Plaque Classification Precision : '+str(p)+"\n")
text.insert(END,'RCNN Plaque Classification Recall    : '+str(r)+"\n")
text.insert(END,'RCNN Plaque Classification FSCORE    : '+str(f)+"\n\n")
text.update_idletasks()
LABELS = class_labels
conf_matrix = confusion_matrix(y_test, predict)
plt.figure(figsize =(6, 6))
```

```
    ax = sns.heatmap(conf_matrix, xticklabels = LABELS, yticklabels = LABELS, annot = True, cmap="viridis" ,fmt ="g");
    ax.set_ylim([0,2])
    plt.title("RCNN Plaque Classification Confusion matrix")
    plt.ylabel('True class')
    plt.xlabel('Predicted class')
    plt.show()
def checkStenosis(filename):
    first = cv2.imread(filename)
    first = cv2.resize(first,(100,100))
    first_gray = cv2.cvtColor(first, cv2.COLOR_BGR2GRAY)
    result = "Non Stenosis Detected"
    for root, dirs, directory in os.walk("stent"):
        for j in range(len(directory)):
            if 'Thumbs.db' not in directory[j]:
                second = cv2.imread(root+"/"+directory[j])
                second = cv2.resize(second,(100,100))
                second_gray = cv2.cvtColor(second, cv2.COLOR_BGR2GRAY)
                score, diff = structural_similarity(first_gray, second_gray, full=True)
                if score >= 0.12:
                    result = "Significant Stenosis Detected"
    return result
```

```
def classify():
    global text, classifier
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="testImages")
    image = cv2.imread(filename)
    img = cv2.resize(image, (64,64))
    im2arr = np.array(img)
    im2arr = im2arr.reshape(1,64,64,3)
    img = np.asarray(im2arr)
    img = img.astype('float32')
    img = img/255
    preds = classifier.predict(img)
    predict = np.argmax(preds)
    img = cv2.imread(filename)
    img = cv2.resize(img, (600,400))
    result = checkStenosis(filename)
    cv2.putText(img, 'Coronary Artery Classified as : '+class_labels[predict], (10, 25), cv2.FONT_HERSHEY_SIMPLEX,0.7, (255, 0, 0), 2)
    cv2.putText(img, result, (10, 75),  cv2.FONT_HERSHEY_SIMPLEX,0.7, (255, 0, 0), 2)
    cv2.imshow('Flower Classified as : '+class_labels[predict], img)
    cv2.waitKey(0)
```

```
def graph():
    f = open('model/history.pckl', 'rb')
    data = pickle.load(f)
    f.close()
    accuracy = data['acc']
    loss = data['loss']
    plt.figure(figsize=(10,6))
    plt.grid(True)
    plt.xlabel('Training Epoch')
    plt.ylabel('Accuracy/Loss')
    plt.plot(loss, 'ro-', color = 'red')
    plt.plot(accuracy, 'ro-', color = 'green')
    plt.legend(['Loss', 'Accuracy'], loc='upper left')
    plt.title('RCNN Plaque Classification Training Accuracy & Loss Graph')
    plt.show()
def GUI():
    global main
    global text
    font = ('times', 16, 'bold')
    title = Label(main, text='A Recurrent CNN for Automatic Detection and Classification of Coronary Artery Plaque and Stenosis in Coronary CT Angiography')
    title.config(bg='darkviolet', fg='gold')
    title.config(font=font)
```

```
title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=30,width=110)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=100)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload CCTA Scan Plaque Dataset", command=uploadDataset, bg='#ffb3fe')

uploadButton.place(x=900,y=100)

uploadButton.config(font=font1)

processButton = Button(main, text="Dataset Preprocessing", command=DataPreprocessing, bg='#ffb3fe')

processButton.place(x=900,y=150)

processButton.config(font=font1)

rcnnButton = Button(main, text="Generate & Load RCNN Model", command=runRCNN, bg='#ffb3fe')

rcnnButton.place(x=900,y=200)

rcnnButton.config(font=font1)

classifyButton = Button(main, text="Plaque Classification", command=classify, bg='#ffb3fe')

classifyButton.place(x=900,y=250)

classifyButton.config(font=font1)
```

```
    graphButton = Button(main, text="RCNN Training Graph", command=graph, bg='#ffb3fe')

    graphButton.place(x=900,y=300)

    graphButton.config(font=font1)

    main.config(bg='forestgreen')

    main.mainloop()

if __name__ == "__main__":

    GUI()
```

# CHAPTER 6

# TESTING

**Implementation and Testing:**

is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

Implementation

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**6.1 System Testing**

Testing has become an integral part of any system or project especially in the field of

information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

### 6.2 Module Testing

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

### 6.3 Integration Testing

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

### 6.4 Acceptance Testing

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

| Test Case Id | Test Case Name | Test Case Desc. | Test Steps | | | Test Case Status | Test Priority |
|---|---|---|---|---|---|---|---|
| | | | Step | Expected | Actual | | |
| 01 | Upload CCTA SCAN DATASET | Test whether the Personality Dataset is uploaded or not into the system | If the Personality Dataset may not uploaded | We cannot do further operations | Personality Dataset uploaded we will do further operations | High | High |
| 02 | DATASET pre processing | Test whether the data set is pre processing or not | If the Model may not uploaded | we cannot do pre process Model | Up Load model we will do further | High | High |
| 03 | generate& load RCNN model | Test whether RCNN generate and run or not | If the RCNN is Not sent | We cannot do RUN RCNN | Sent the cloud operations we will do further | High | High |
| 04 | plaque classification | Verify plaque classified or not | Without classified | we cannot detected | we can fall detected | High | High |
| 05 | RCNN training graph | Verify thegraph plotted or not | Without plot | we cannot closed | we can closed | High | High |

**Table 6.4.1**

# CHAPTER 7

# SCREENSHOTS

To implement this project, we have used below dataset

In the given dataset we have 2 folder and just go inside any folder to view CCTA images:

- Plaque
- No Plaque

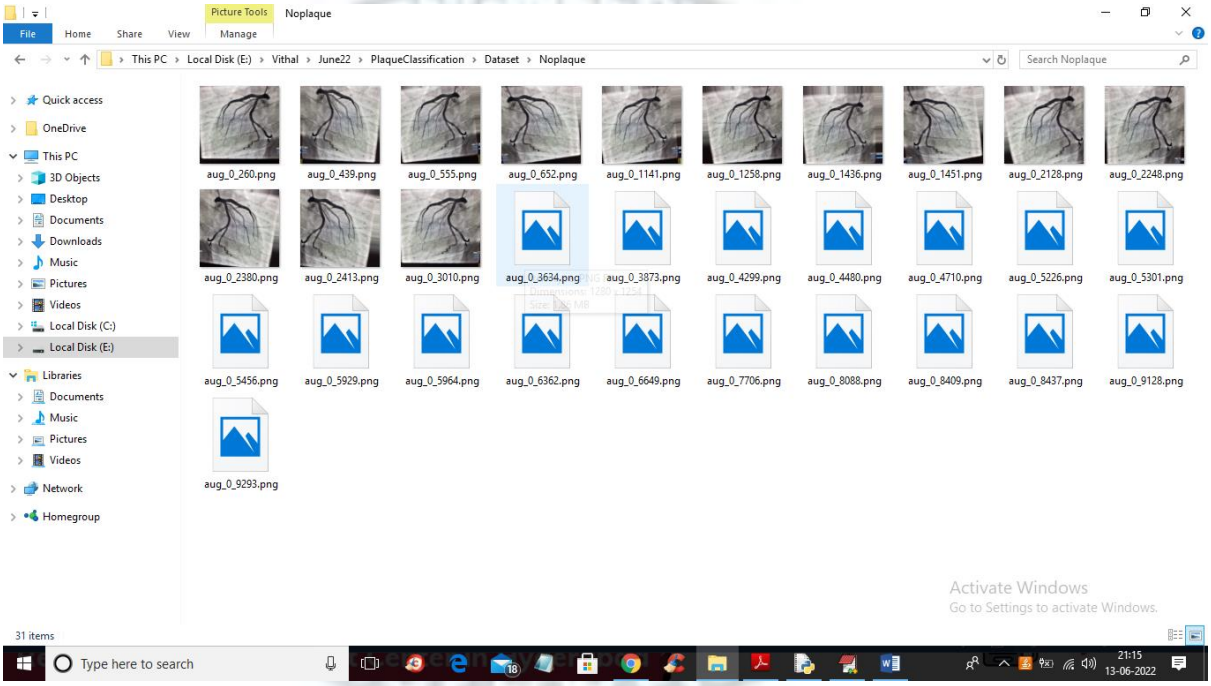


Fig 7.1 Dataset Images

By using above image, we are training CNN algorithm

SCREEN SHOTS

To run project double click on 'run.bat' file to get below screen

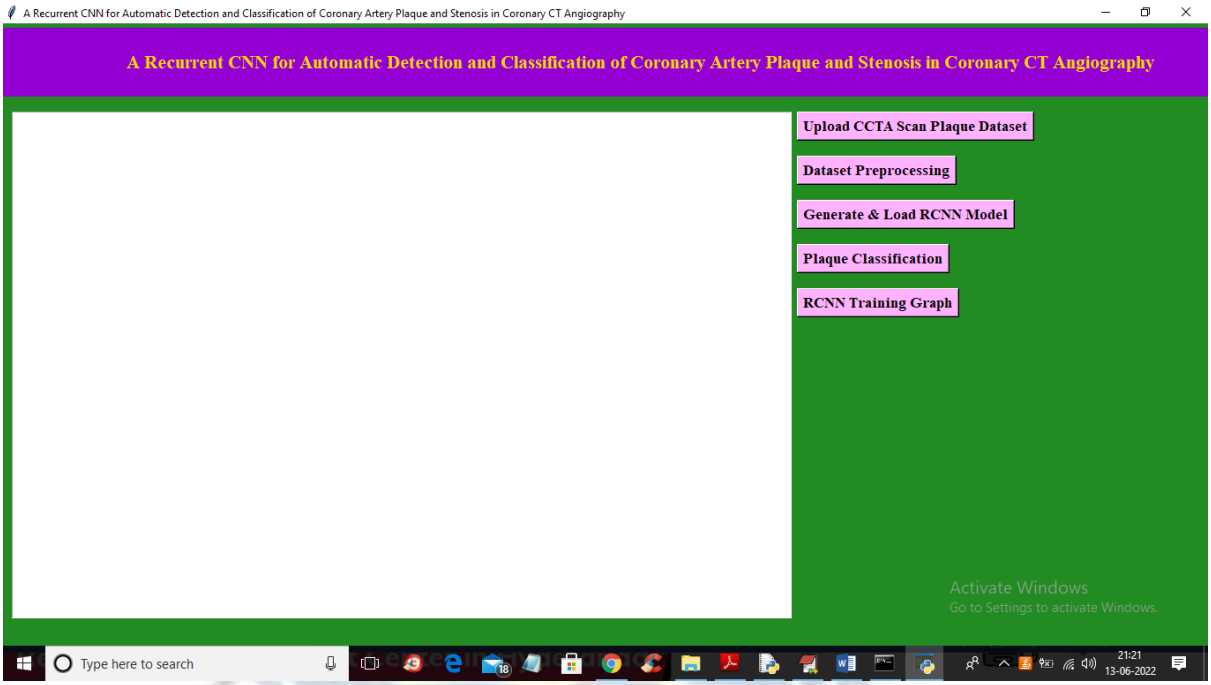


Fig 7.2 GUI Home Page

In above screen click on 'Upload CCTA Scan Plaque Dataset' button to upload dataset and get below output

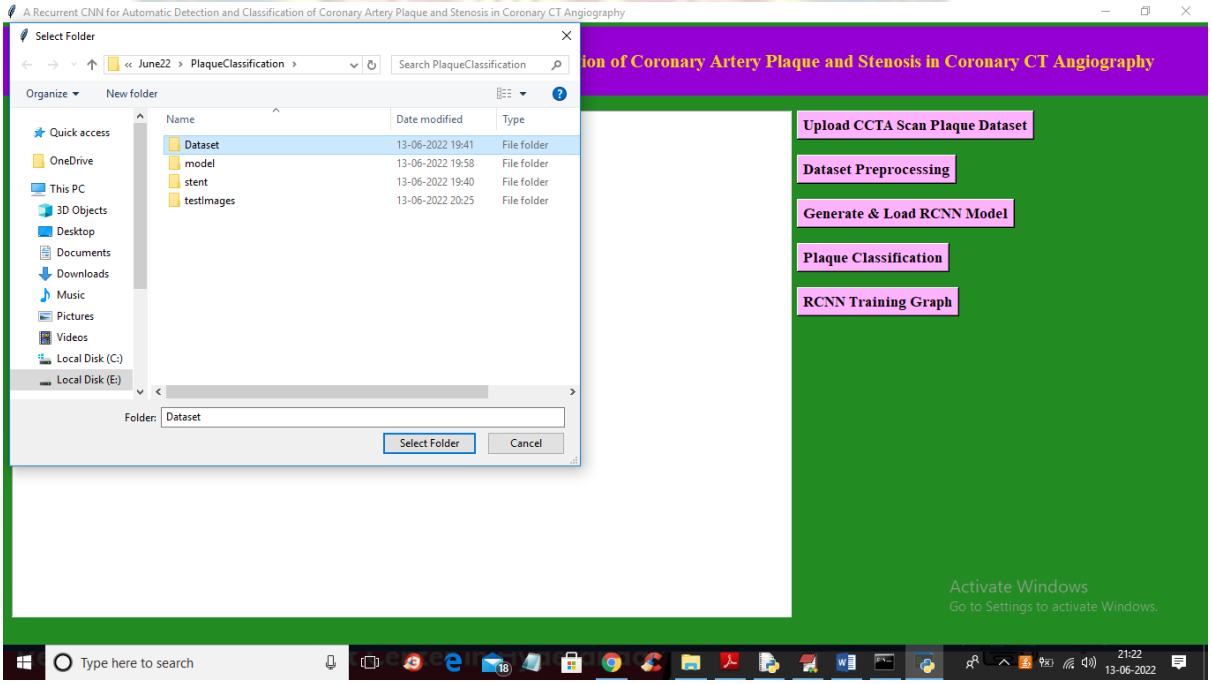


Fig 7.3 Upload Dataset

In above screen selecting and uploading dataset and then click on 'Select Folder' button to load dataset and get below output

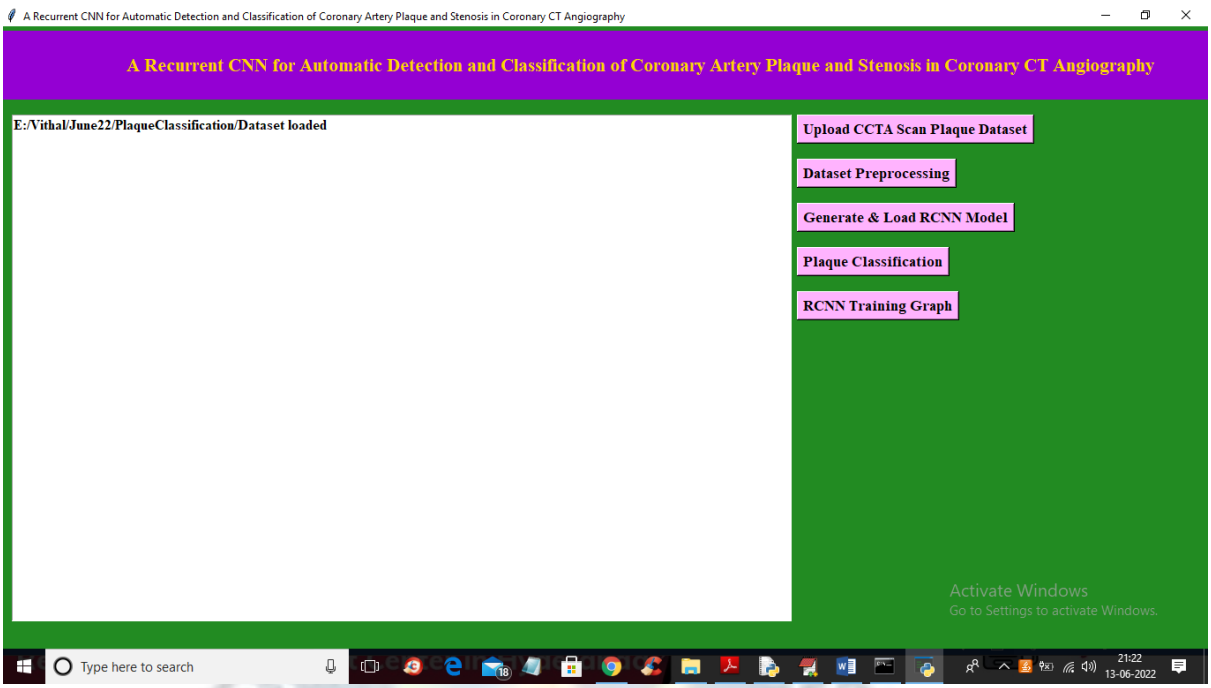


Fig 7.4 Data Preprocessing

In above screen dataset loaded and now click on 'Dataset Preprocessing' button to process dataset and get below output

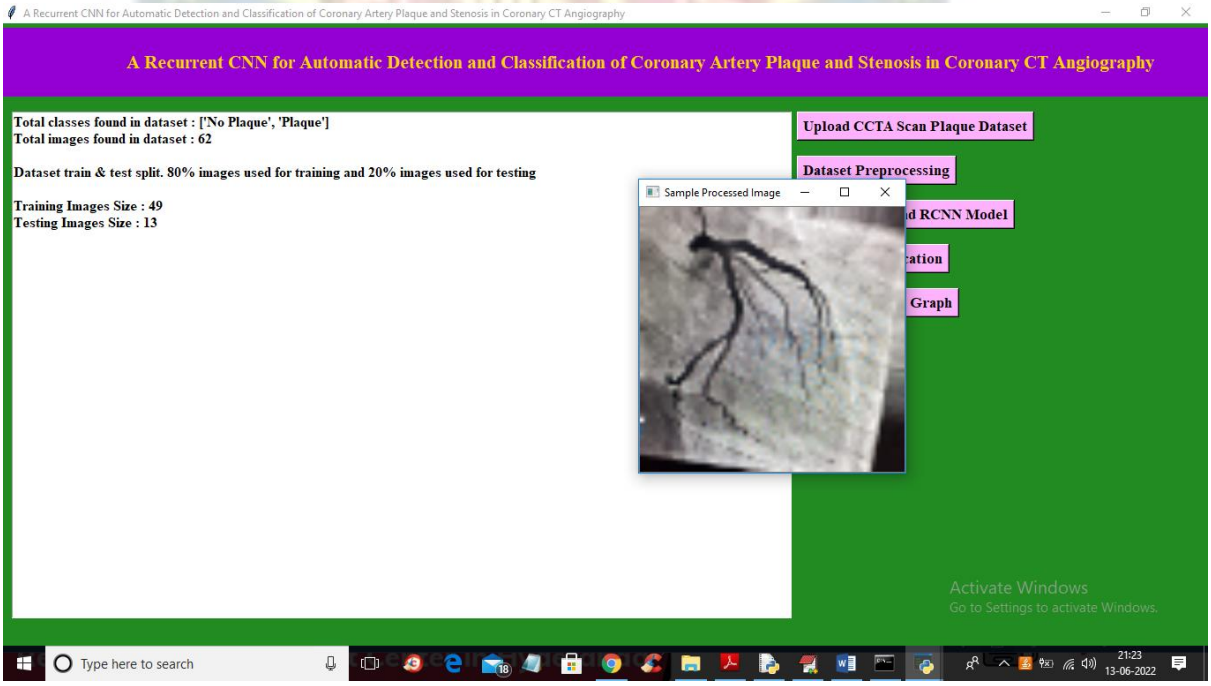


Fig 7.5 Generate & Load RCNN

In above screen we can see class labels found in dataset and total images available in dataset with train and test images size and in above screen displaying sample process image to check all images are loaded and process properly and now close above image and then click on 'Generate & Load RCNN Model' button to build RCNN model and get below output

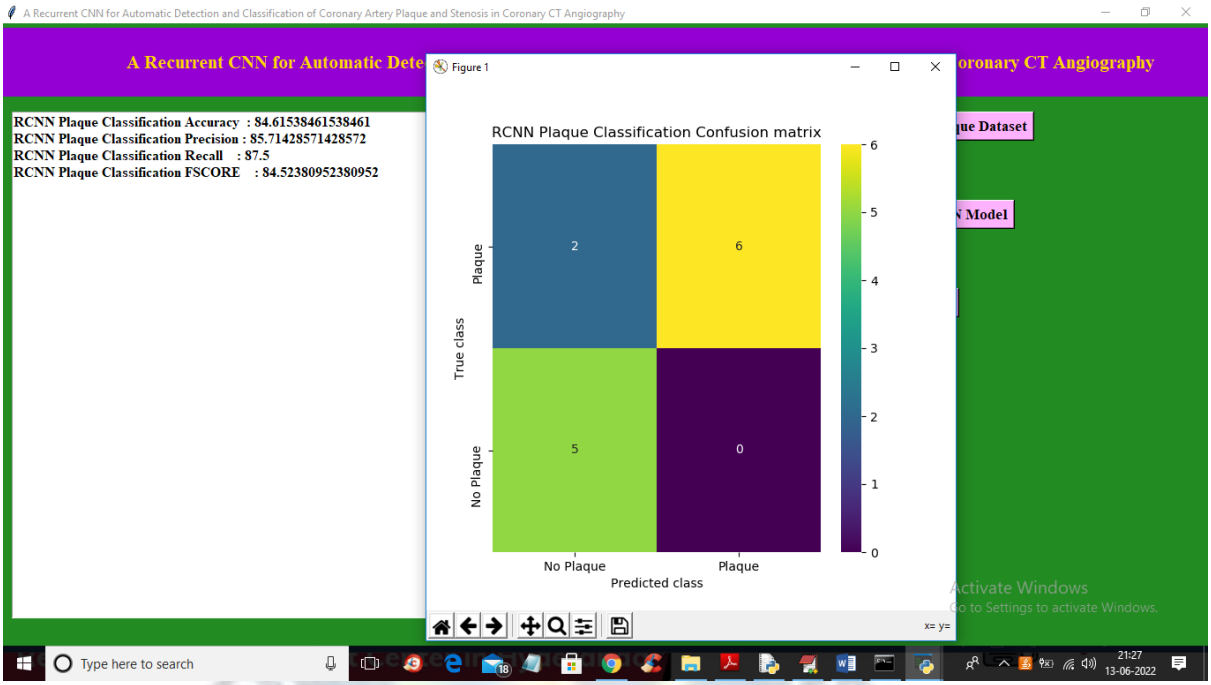


Fig 7.6 Confusion Matrix

In above screen with RCNN we got accuracy as 84% and in confusion matrix graph x-axis represents Predicted Classes and y-axis represents True classes and in above graph we can see only 2 images of 'No Plaque' is incorrectly predicted as 'Plaque'. Now close above graph and then click on 'Plaque Classification' button to upload test image and get below output

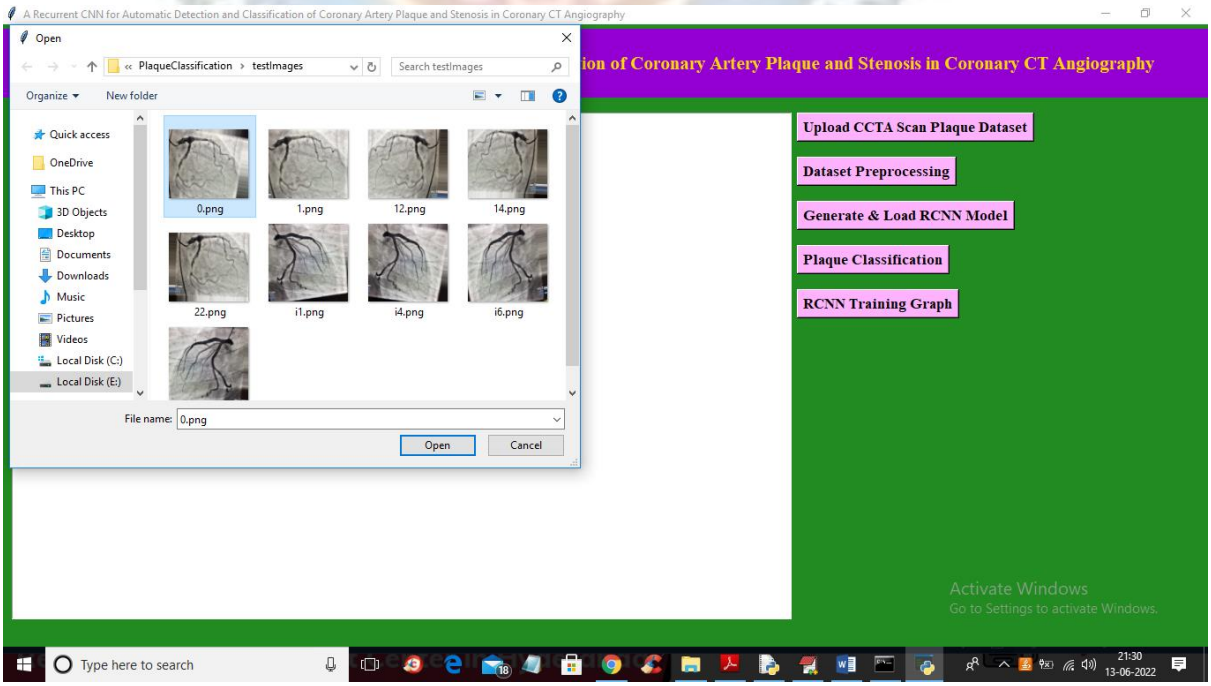


Fig 7.7 Upload CCTA Image to Model

In above screen selecting and uploading '0.png' and then click on 'Open' button to get below output

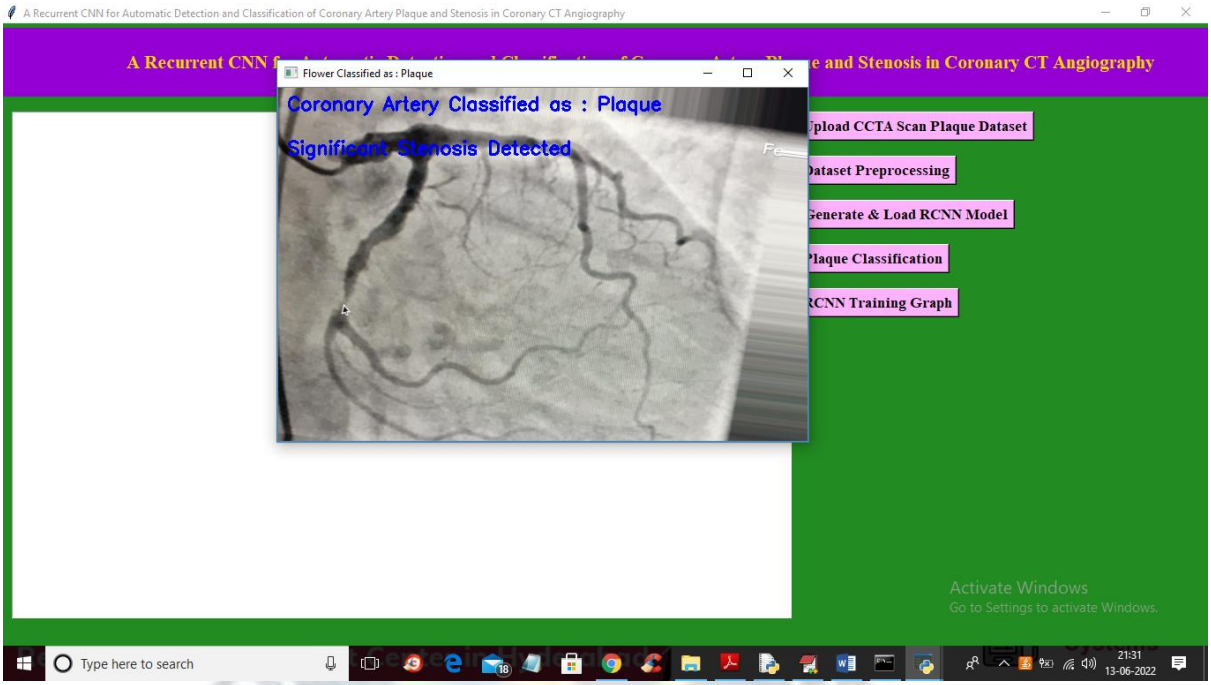


Fig 7.8 Plaque Detected

In above screen in image blue colour text we can see classification result as 'Plaque Detected' and Stenosis also detected and now upload and test other images

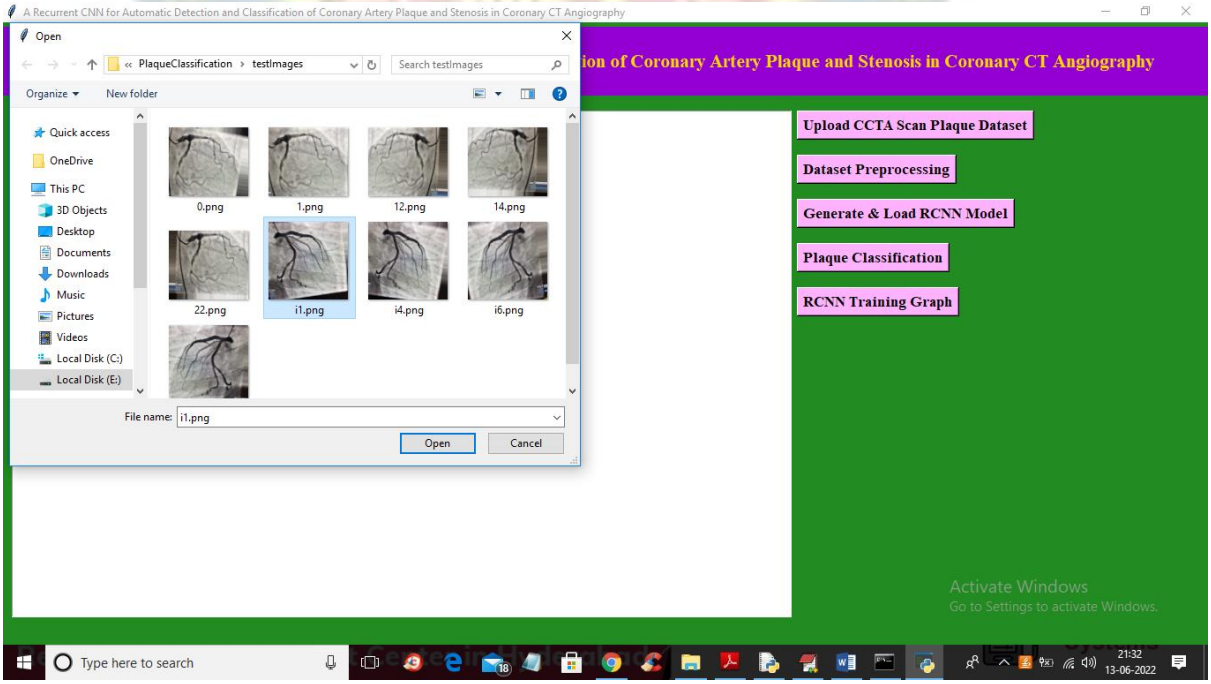


Fig 7.9 Upload CCTA Image to Model

In above screen uploading 'i1.png' and then click on 'Open' button to get below output

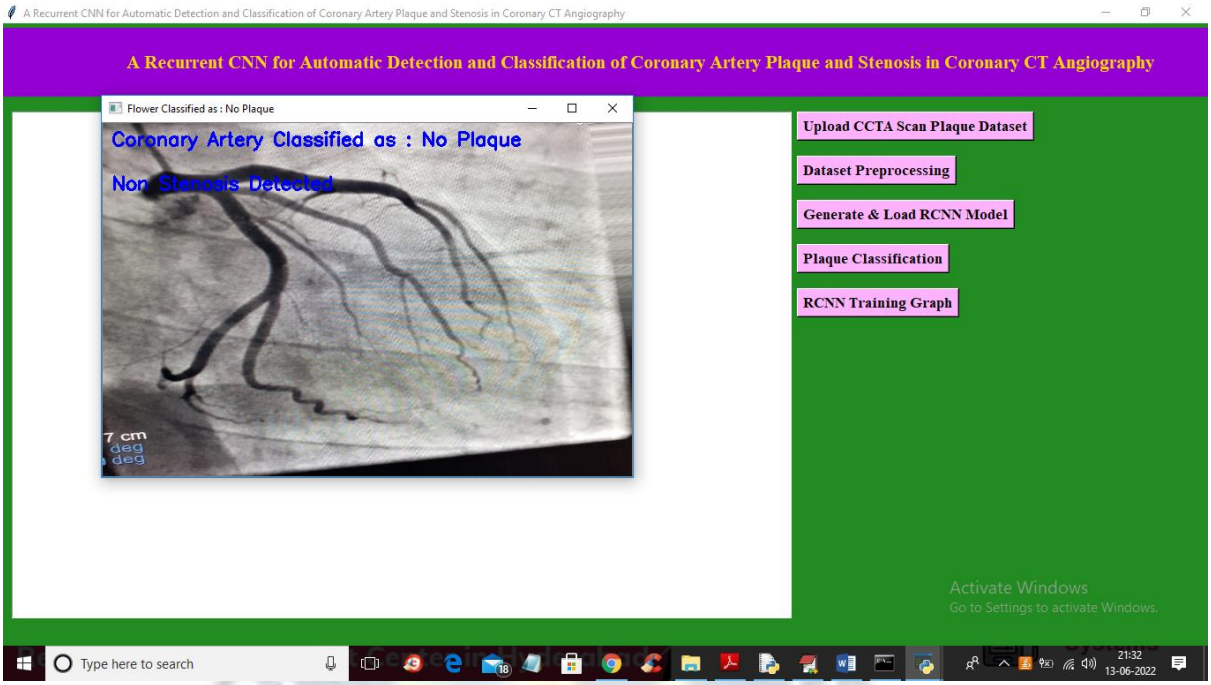


Fig 7.10 No Plaque Detected

In above screen no plaque detected and similarly you can upload and test other images and now click on 'RCNN Training Graph' button to get below graph

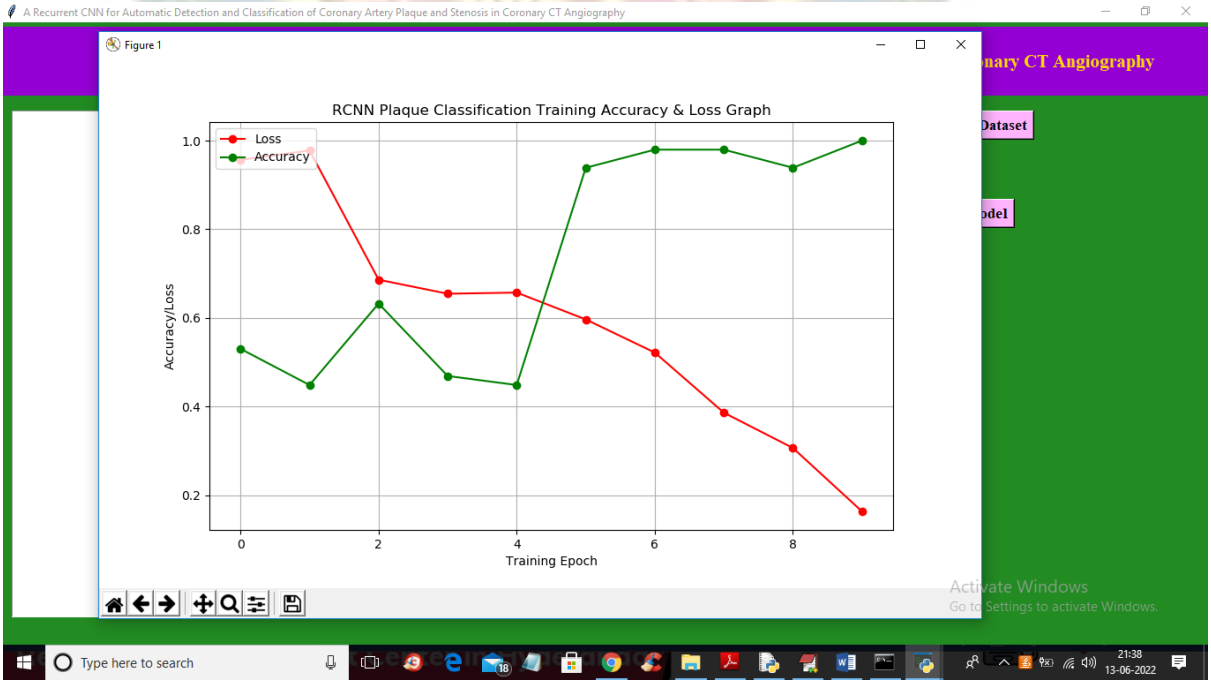


Fig 7.11 RCNN Training Graph

In above graph x-axis represents epochs and y-axis represents accuracy and loss values and in above graph red line represents LOSS and green line represents accuracy and in above graph we can see with each increasing epoch accuracy got increase and loss got decrease

# CHAPTER 8

# CONCLUSION

A method for automatic detection and characterization of the coronary artery plaque type, as well as detection and characterization of the anatomical significance of the coronary artery stenosis was presented. The method employs an RCNN that analyzes an MPR view of a coronary artery extracted from a CCTA scan using the coronary artery centerline. The RCNN utilizes a 3D CNN that computes image features from 3D volumes extracted along the coronary artery centerline. Subsequently, an RNN analyzes the computed image features to perform both classification tasks. Unlike most previous methods that detect and characterize coronary artery plaque and stenosis relying on the coronary artery lumen segmentation [9], the proposed method requires only the coronary artery centerline as an input along with the CCTA image.

# CHAPTER 9

# REFERENCES

[1] D. Mozaffarian, E. J. Benjamin, A. S. Go, D. K. Arnett, M. J. Blaha, M. Cushman, S. R. Das, S. de Ferranti, J.-P. Despres, H. J. Fullerton ´ et al., "Heart disease and stroke statistics - 2016 update," Circulation, vol. 133, no. 4, pp. e38–e360, 2022.

[2] G. Raff, A. Abidov, S. Achenbach, D. Berman, L. Boxt, M. Budoff, V. Cheng, T. DeFrance, J. Hellinger, R. Karlsberg et al., "SCCT guidelines for the interpretation and reporting of coronary computed tomographic angiography," Journal of Cardiovascular Computed Tomography, vol. 3, no. 2, pp. 122–136, 2022.

[3] J. M. Wolterink, T. Leiner, B. D. De Vos, J.-L. Coatrieux, B. M. Kelm, S. Kondo, R. A. Salgado, R. Shahzad, H. Shu, M. Snoeren et al., "An evaluation of automatic coronary artery calcium scoring methods with cardiac CT using the orCaScore framework," Medical Physics, vol. 43, no. 5, pp. 2361–2373, 2021.

[4] R. Virmani, A. P. Burke, A. Farb, and F. D. Kolodgie, "Pathology of the vulnerable plaque," Journal of the American College of Cardiology, vol. 47, no. 8 Supplement, pp. C13–C18, 2021.

[5] S. Achenbach, "Can CT detect the vulnerable coronary plaque?" The International Journal of Cardiovascular Imaging (formerly Cardiac Imaging), vol. 24, no. 3, pp. 311–312, 2019.

[6] A. Cassar, D. R. Holmes Jr, C. S. Rihal, and B. J. Gersh, "Chronic coronary artery disease: diagnosis and management," in Mayo Clinic Proceedings, vol. 84, no. 12. Elsevier, , pp. 1130–1146, 2019.

[7] R. C. Cury, S. Abbara, S. Achenbach, A. Agatston, D. S. Berman, M. J. Budoff, K. E. Dill, J. E. Jacobs, C. D. Maroules, G. D. Rubin et al., "CAD-RADSTM coronary artery disease–reporting and data system. an expert consensus document of the society of cardiovascular computed tomography (SCCT), the american college of radiology (ACR) and the north american society for cardiovascular imaging (NASCI). endorsed by the american college of cardiology," Journal of Cardiovascular Computed Tomography, vol. 10, no. 4, pp. 269–281, 2018.

[8] M. J. Budoff, D. Dowe, J. G. Jollis, M. Gitter, J. Sutherland, E. Halamert, M. Scherer, R. Bellinger, A. Martin, R. Benton et al., "Diagnostic performance of 64-multidetector row coronary computed tomographic angiography for evaluation of coronary artery stenosis in individuals without known coronary artery disease: results from the prospective multicenter ACCURACY trial," Journal of the American College of Cardiology, vol. 52, no. 21, pp. 1724–1732, 2018.

[9] H. Kiris¸li, M. Schaap, C. Metz, A. Dharampal, W. B. Meijboom, S.-L. Papadopoulou, A. Dedic, K. Nieman, M. De Graaf, M. Meijs et al., "Standardized evaluation framework for evaluating coronary artery stenosis detection, stenosis quantification and lumen segmentation algorithms in computed tomography angiography," Medical Image Analysis, vol. 17, no. 8, pp. 859–876, 2016.

[10] A. Arbab-Zadeh and J. Hoe, "Quantification of coronary arterial stenoses by multidetector CT angiography in comparison with conventional angiography: methods, caveats, and implications," JACC: Cardiovascular Imaging, vol. 4, no. 2, pp. 191–202, 2016.

[11] J. M. Wolterink, T. Leiner, B. D. de Vos, R. W. van Hamersvelt, M. A. Viergever, and I. Isgum, "Automatic coronary artery calcium scoring ˇ in cardiac CT angiography using paired convolutional neural networks," Medical Image Analysis, vol. 34, pp. 123–136, 2014.

[12] N. Lessmann, B. van Ginneken, M. Zreik, P. A. de Jong, B. D. de Vos, M. A. Viergever, and I. Isgum, "Automatic calcium scoring in low-dose ˇ chest CT using deep neural networks with dilated convolutions," IEEE Transactions on Medical Imaging, vol. 37, pp. 615–625, 2013.

[13] R. Shadmi, V. Mazo, O. Bregman-Amitai, and E. Elnekave, "Fullyconvolutional deep-learning based system for coronary calcium score prediction from non-contrast chest CT," in IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), , pp. 24–28, 2013.

[14] T. Schepis, M. Marwan, T. Pflederer, M. Seltmann, D. Ropers, W. G. Daniel, and S. Achenbach, "Quantification of noncalcified coronary atherosclerotic plaques with dual source computed tomography: comparison to intravascular ultrasound," Heart, pp. hrt–2009, 2018.

[15] D. Dey, T. Schepis, M. Marwan, P. J. Slomka, D. S. Berman, and S. Achenbach, "Automated three-dimensional quantification of noncalcified coronary plaque from

coronary CT angiography: comparison with intravascular US," Radiology, vol. 257, no. 2, pp. 516–522, 2012.

[16] S. Mittal, Y. Zheng, B. Georgescu, F. Vega-Higuera, S. K. Zhou, P. Meer, and D. Comaniciu, "Fast automatic detection of calcified coronary lesions in 3D cardiac CT images," in International Workshop on Machine Learning in Medical Imaging. Springer, , pp. 1–9, 2009.

[17] M. A. Zuluaga, I. E. Magnin, M. H. Hoyos, E. J. D. Leyton, F. Lozano, and M. Orkisz, "Automatic detection of abnormal vascular crosssections based on density level detection and support vector machines," International Journal of Computer Assisted Radiology and Surgery, vol. 6, no. 2, pp. 163–174, 2009.

[18] S. Sankaran, M. Schaap, S. C. Hunley, J. K. Min, C. A. Taylor, and L. Grady, "HALE: Healthy area of lumen estimation for vessel stenosis quantification," in International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 380–387, 2008.

[19] E. J. Halpern and D. J. Halpern, "Diagnosis of coronary stenosis with CT angiography: comparison of automated computer diagnosis with expert readings," Academic Radiology, vol. 18, no. 3, pp. 324–333, 2006.

[20] Y. Xu, G. Liang, G. Hu, Y. Yang, J. Geng, and P. K. Saha, "Quantification of coronary arterial stenoses in CTA using fuzzy distance transform," Computerized Medical Imaging and Graphics, vol. 36, no. 1, pp. 11–24, 2003.

[21] R. Shahzad, T. van Walsum, H. Kirisli, H. Tang, C. Metz, M. Schaap, L. van Vliet, and W. Niessen, "Automatic stenoses detection, quantification and lumen segmentation of the coronary arteries using a two point centerline extraction scheme," in MICCAI 2012 workshop proceedings, 2001.