AUTOMATED ANNOTATION USING OPENCV

A dissertation submitted to the Jawaharlal Nehru Technological University, Hyderabad in partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

L. HARIKA (19B81A05K3)

G. SNEHA JOYCE (19B81A05N4)

D. SRITHA REDDY (19B81A05P0)

Under the guidance of

N.N.S.S.S. ADITHYA

Assistant Professor

Department of Computer Science and Engineering

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH, Accredited by NBA, and NAAC) Vastunagar, Mangalpalli (V), Ibrahimpatnam (M), Ranga Reddy (Dist.) - 501510, Telangana State.

2022-23

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH, Accredited NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Ranga Reddy (Dist.) - 501510, Telangana State.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled **"Automated Annotation Using OpenCV"** is being submitted by **L. Harika (19B81A05K3)**, **G. Sneha Joyce (19B81A05N4), D. Sritha Reddy (19B81A05P0)** in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering,** during the academic year 2022-2023.

Project Guide
(N.N.S.S.S. Adithya)
Assistant Professor

Professor-in-charge projects
(Dr. R.K. Selvakumar)
Professor, Department of CSE

External Examiner

Professor and Head, CSE

(Dr. A. Vani Vathsala)

DECLARATION

We the undersigned solemnly declare that the project report titled 'DETECTION OF CARDIOVASCULAR DISEASES IN ECG IMAGES USING MACHINE LEARNING AND DEEP LEARNING METHODS' is based on our own work carried out during the course of our study under the supervision of Ch. Bhavani, CSE Dept., CVR College of Engineering.

We assert the statements made and conclusions are drawn are an outcome of our research work. We further certify that

1. The work contained in the report is original and has been done by us under the general supervision of our supervisor.

2. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or in the any other University of India or abroad

3. We have followed the guidelines provided by the university in writing the report.

ACKNOWLEDGEMENT

The satisfaction 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 has been a source of inspiration throughout the course of the project.

It is a great pleasure to convey our profound sense of gratitude to our principal **Dr. K. Ramamohan Reddy**, Vice-Principal **Prof. L. C. Siva Reddy**, **Dr. A. Vani Vathsala**, Head of CSE Department, CVR College of Engineering, for having been kind enough to arrange necessary facilities for executing the project in the college.

We deem it a pleasure to acknowledge our sense of gratitude to our project guide **Ch. Bhavani** under whom we have carried out the project work. His incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work.

We wish a deep sense of gratitude and heartful thanks to the management for providing excellent lab facilities and tools. Finally, we thank all those whose guidance helped us in this regard.

KISTIPATI PRIYATHAM REDDY(19B81A05F5)

BUSSA AADARSH(19B81A05F6)

NAVARU RISHI VARDHAN REDDY(19B81A05F9)

ABSTRACT

Cardiac disease is the leading cause of death worldwide. Cardiovascular diseases can be prevented if an effective diagnostic is made at the initial stages. (e ECG test is referred to as the diagnostic assistant tool for screening of cardiac disorder. (e research purposes of a cardiac disorder detection system from 12-lead-based ECG Images. (e healthcare institutes used various ECG equipment that present results in nonuniform formats of ECG images. (e research study proposes a generalized methodology to process all formats of ECG. Single Shoot Detection (SSD) MobileNet v2-based Deep Neural Network architecture was used to detect cardiovascular disease detection. (e study focused on detecting the four major cardiac abnormalities (i.e., myocardial infarction, abnormal heartbeat, previous history of MI, and normal class) with 98% accuracy results were calculated. (e work is relatively rare based on their dataset; a collection of 11,148 standard 12-lead-based ECG images used in this study were manually collected from health care institutes and annotated by the domain experts. (e study achieved high accuracy results to differentiate and detect four major cardiac abnormalities. Several cardiologists manually verified the proposed system's accuracy result and recommended that the proposed system can be used to screen for a cardiac disorder.

TABLE OF CONTENTS

List of Figures

List of Abbreviations

ML	Machine Learning
WEKA	Waikato Environment for Knowledge Analysis
CNN	Convolutional Neural Network
VR	Virtual Reality
NLP	National Language Processing
NLU	National Language Understanding
API	Application Program Interface
AI	Artificial Intelligence
UI	User Interface
IP	Internet Protocol
UML	Unified Modeling Language
GUI	Graphical User Interface
HTML	Hyper Text Markup Language

1.INTRODUCTION

According to the Centres for Disease Control and Prevention (CDC) and the American Health Monitoring Organization, the leading cause of death is cardiovascular disease . CDC revealed that 74% of the population is affected yearly by heart disease. Cardiovascular diseases can be prevented if an effective diagnostic is made at the initial stages . Modern medical science has shown substantial and potent solutions to cope with heart-related problems. In the era of technology, medical issues can be tackled with advanced techniques of Information technology. Most common heart disease detection technique is based on electrocardiogram (ECG), angiography screening, and blood test. ECG test is also referred to as the diagnostic assistant tool for screening heart diseases. ECG is a visual signal captured or measured by placing electrodes on the body's surface to detect voltage changes (Figure 1) ECG represents the possible cardiac abnormalities in their ST segments: normally, the rises in ST segments, changes in segments, or flipping of T waves, or new Q wave; these abnormal segments reflect cardiac disease symptoms. Researchers investigated many techniques for automatically detecting cardiovascular disease using machine and deep learning techniques, typically by using ECG in one or two dimensional voltage amplitude data represented as time-series signals. To classify ECG signals in time series, different methods are explored and show a substantial result. Ubeyli used an eigenvector method for feature extraction and to classify ECG beats. Sharma et al. used time frequency-based ECG signals for feature extraction of the eigenvalue decomposition of Hankel matrix and Hilbert transform and used the random forest to detect the cardiovascular disorder. In the current era, systems are being developed for the automatic detection of cardiac-related issues. Systems predict high accuracy results based on one-dimensional ECG beats signals but are still not adopted as tools in health care institutes. The main areas that affect the success of these approaches, i.e., selection of feature, extraction techniques, types of classification algorithms, and the most important, the use of imbalanced data for classification can reduce the recognition accuracy of the minority class. Moreover, state-of-the-art studies represents the classical methods to show high accuracy in classification and detection tasks on one lead-based ECG images instead of standard 12-lead-based ECG images. Furthermore, the healthcare institutes used various ECG equipment, presenting results in nonuniform formats of ECG images. State-of-the-art

research was unable to propose a generalized methodology for nonuniform formats of ECG images. The availability of 12-lead-based ECG source images is not publicly available for researchers. In this research, the main focus is to provide a novel automatic detection tool relatively similar and adaptable for the cardiac hospitals to process the 12-lead-based ECG images.

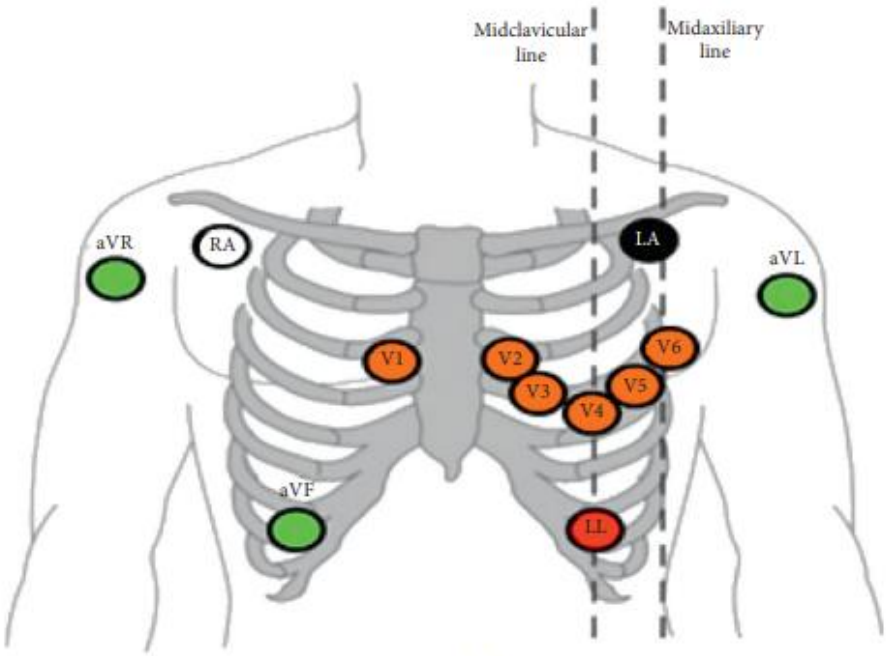


FIGURE 1: ECG electrodes.

Automated cardiac disorder detection via a deep neural network using 12-lead-based ECG image processing is critical. Deep neural network works in the same fashion as a human brain, founded on mathematical formulas/models. The mathematics of functional principles of deep neural networks targets to understand and recognize the pattern among different components. The deep neural network's fundamental unit is a neuron trained by repetitive tasks and gets experienced just like a human brain through acquired knowledge attained in training. The focus of training and acquiring knowledge is to establish a connection between input and output. After training, the system is capable of detecting the objects about what it has been trained. The primary motivation of this study is to develop an efficient automated model for the diagnosis of cardiac disorder on ECG that can feasibly be implemented on portable healthcare devices.

2. LITERATURE REVIEW

In the past few decades, the research community has focused on artificial intelligence by working in digital image processing, computer vision, and machine learning to provide a platform between human and machine theory. This work is widely recognized by several companies and medical fields for classification, detection, and identification of cardiac disorder, which play a vital role in the health community. Techniques used for the identification of cardiac disorder using deep learning have been focused on for many years. In recent studies, the state-of-the-art research on deep learning gained the potential growth with satisfactory results in detection and classification tasks on medical images . Most studies treated dimensional ECG signals as time-series classification by keeping in the view of deep learning. For example, PCGs and ECGs are used to diagnose heart disease. PCGs (also known as heart sound auscultation) is commonly listened to or recorded by practitioners through a stethoscope, identifying the heart irregularities. For this reason, heart signals have been critically studied to make a diagnosis. CNN is considered a state-of-the-art tool for detecting and classification heart signals and has been studied with several variations like 1-dimensional, 2-dimensional, or the combination of both. Similarly, Noman et al. proposed a framework based on 1-dimensional CNN for direct feature learning from raw heart signals and 2-dimensional CNN, which takes 2-dimensional time-frequency feature maps. Xia et al. proposed a model for automatic wearable ECG classification. Huang et al. transformed five types of heartbeats' signals into time-frequency spectrograms and then trained a 2D-CNN for classifying arrhythmia types. Lu et al. transformed the 1D signals into 2D images by joining the dots of 1D signals. Ji et al. also used a 1D signal, converted all signals into 2D, and used R-CNN for ECG classification. In the state-of-the-art research studies, these time-series data used feature selection manually, which is not adaptable in different application environments. Moreover, ECG's time-series data with signal leads are not appropriate for stable baseline wanders, muscle contraction, and power line interface. In general, the practical methods normally adopted by a cardiologist for screening cardiac patients are 12-lead-based ECG images. In standard 12-lead-based ECG images, six leads are considered "limb leads" because they are placed on the individual's arms and/or legs. (e other six leads are considered "precordial leads" because they are placed on the torso (precordium). (e six limb leads are lead I, II, III,

aVL, aVR, and aVF. The six precordial leads are called leads V1, V2, V3, V4, V5, and V6. Figure 3 shows sample 12-lead-based ECG image used in this study. As concerned with ECG images, this research study is unique. In the real, collection of ECG images is done by recording or screening from Telehealth ECG diagnostic tool used in healthcare institutes. It is hard to gain these 12-leadbased on ECG images. Moreover, these images are used as a standard tool to detect the cardiac disorder in real-time applications. In this research, the main focus is to provide a novel automatic detection tool relatively similar and adaptable for cardiac hospitals by using a deep neural network (DNN). (e DNN has been widely recognized as a reliable approach to directly detect the characteristics of features from medical images. DNN is best suited for medical images problem; that is the main reasons authors used SSD MobileNet V2, a deep neural network-based architecture to detect the cardiac disorder on 12-lead-based ECG images

2.1 EXISTING WORK

Many researchers are working in the field of heart disease prediction. Shen et al. initially, proposed a self applied questionnaire (SAQ) based study to predict heart disease. This study is based on the analysis of the common risk features of the disease and other data collected in SAQ. Dundee rank factor score is used to validate their study. This study is based on statistically 3 risk factors (blood pressure, smoking, and blood cholesterol) together with sex and age to determine the risk of having heart disease.

Chen et al. Proposed an idea of heart disease prediction. They used learning Vector Quantization which is one of the artificial intelligence technique for classification and prediction purpose. Initially, 13 clinical features i.e. age, cholesterol, chest pain type, exercise, induced angina, max heart rate, fasting blood sugar, number of vessels colored, old peak, resting ECG, sex, slope, thal, and trestbps are identified for prediction. Secondly, Artificial Neural Network (ANN) is applied for classification purpose. Finally, training of neural networks is performed using back propagation to evaluate the heart disease prediction system. Nearly 80% accuracy is achieved on testing set for heart disease prediction by the given system.

2.2 LIMITATIONS OF EXISTING WORK

There are several limitations to the existing models of detection of cardiovascular diseases using ECG images using machine learning and deep learning methods, some of which are:

1. Limited training data: The availability of large-scale annotated ECG datasets is limited, which limits the potential for the development of highly accurate models.
2. Variability in ECG signal quality: The quality of ECG signals can vary due to factors such as electrode placement, motion artifacts, and noise, which can affect the accuracy of the model.
3. Lack of interpretability: Deep learning models are often considered "black boxes," making it difficult to understand how the model makes predictions, which can limit its clinical utility.
4. Limited generalization: Models trained on specific populations may not generalize well to other populations, as ECG signals can vary depending on demographic factors such as age, sex, and race.
5. Imbalanced datasets: Imbalanced datasets, where the number of positive and negative cases are not balanced, can lead to biased models that perform poorly on underrepresented classes.
6. Lack of explainability: The models may not provide explanations for the predictions made, which is crucial in a clinical setting where explanations for decisions are necessary.
7. Limited accuracy: The current models are not highly accurate, and false positives and false negatives are common. This can lead to missed diagnoses or unnecessary interventions.

Overall, while machine learning and deep learning methods have shown promise in the detection of cardiovascular diseases using ECG images, there are several limitations that must be addressed before these models can be integrated into clinical practice.

3.REQUIREMENT ANALYSIS

3.1 FUNCTIONAL REQUIREMENTS

Functional requirement should include function performed by a specific screen outline work-flows performed by the system and other business or compliance requirement the system must meet.

Functional requirements specify which output file should be produced from the given file they describe the relationship between the input and output of the system, for each functional requirement a detailed description of all data inputs and their source and the range of valid inputs must be specified.

The functional specification describes what the system must do, how the system does it is described in the design specification.

If a user requirement specification was written, all requirements outlined in the user requirements specifications should be addressed in the functional requirements.

- The user should be able to register and manage his appointments online at any time.
- Database has to store all the information efficiently without any information loss.
- The user shall be able to search for the doctors by specialty, name, working time and/or gender.
- The user can change his profile info at any time
- Hospital can manage all appointments made with him on his account.

3.2 NON-FUNCTIONAL REQUIREMENTS

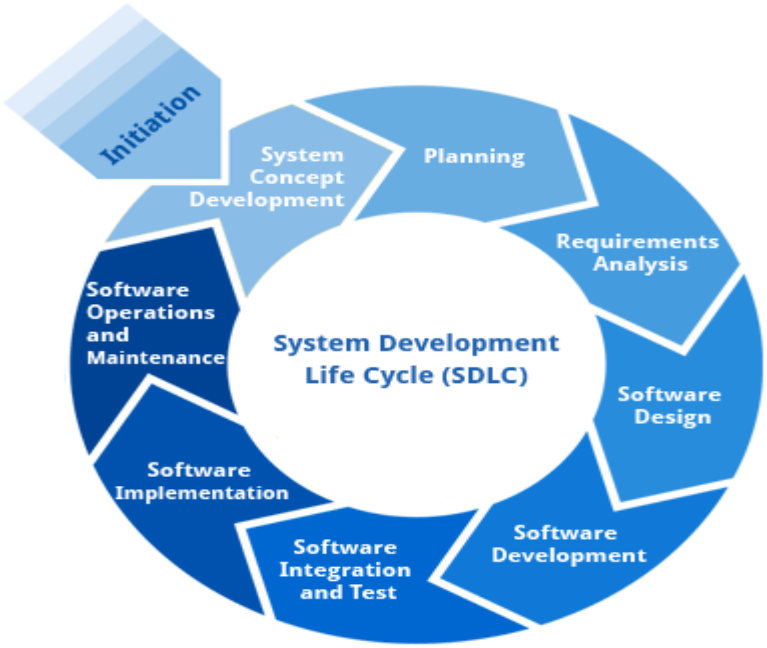
Describe user-visible aspects of the system that are not directly related with the functional behaviour of the system. Non-Functional requirements include quantitative constraints, such as response time (i.e. how fast the system reacts to user commands.) or accuracy i.e. how precise are the systems numerical answers.).

- Portability
- Reliability
- Usability

- Time Constraints
- Error messages
- Actions which cannot be undone should ask for confirmation
- Responsive design should be implemented
- Space Constraints
- Performance
- Standards
- Ethics
- Interoperability
- Security
- Privacy
- Scalability

3.3 Software Development Life Cycle:

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies use to develop these systems.



Requirement Analysis and Design

Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered. Design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. The logical system of the product is developed in this phase.

Implementation

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like PYTHON 3.6, Anaconda Cloud are used for coding. With respect to the type of application, the right programming language is chosen.

Testing

In this phase the system is tested. Normally programs are written as a series of individual modules, this subject to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system. The system is tested to ensure that interfaces between modules work (integration testing), the system works on the intended platform and with the expected volume of data (volume testing) and that the system does what the user requires (acceptance/beta testing).

Maintenance

Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

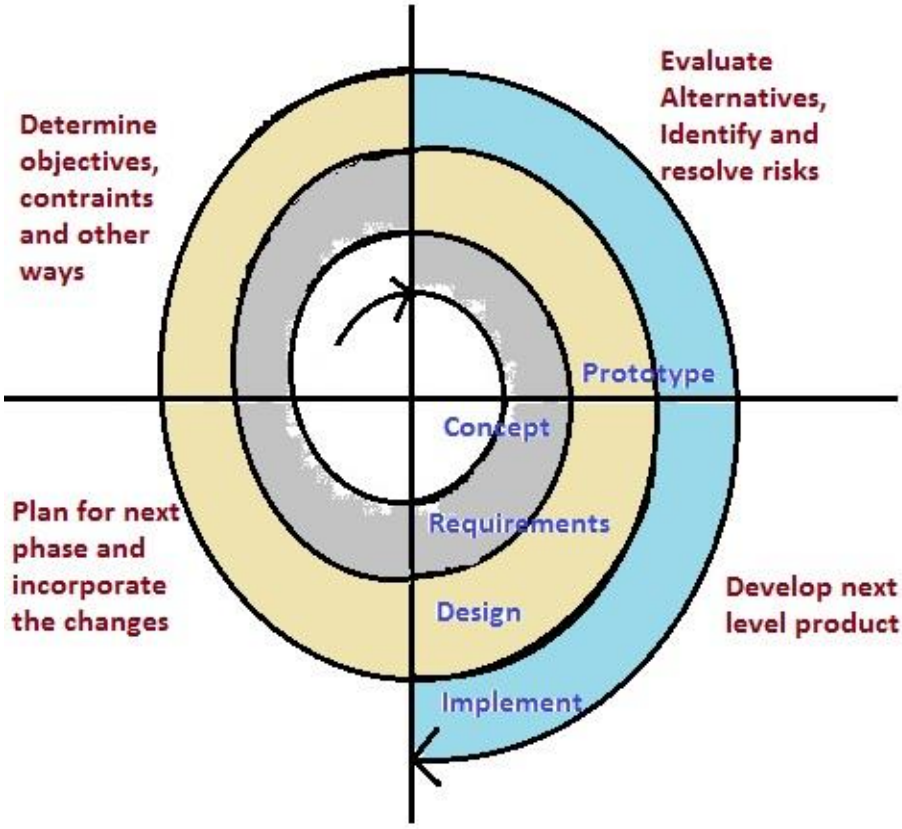
SDLC METHDOLOGIES

This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

SPIRAL MODEL was defined by Barry Boehm in his 1988 article, "A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The following diagram shows how a spiral model acts like:



The steps for Spiral Model can be generalized as follows:

- The new system requirements are defined in as much details as possible.
- This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.

- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
- A second prototype is evolved by a fourfold procedure:

1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
2. Defining the requirements of the second prototype.
3. Planning a designing the second prototype.
4. Constructing and testing the second prototype.

- At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer's judgment, result in a less-than-satisfactory final product.
- The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
- The final system is constructed, based on the refined prototype.
- The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

4. PROPOSED SYSTEM DESIGN

The proposed system is built around conventional three-tier architecture. The three-tier architecture for web development allows programmers to separate various aspects of the solution design into modules and work on them separately. That is, a developer who is best at one part of development, say UI development need not worry about the implementation levels so much. It also allows for easy maintenance and future enhancements. The three-tiers of the solution include:

- The Layout**:**

  This tier is at the uppermost layer and is closely bound to the user, i.e., the users of the system interact with it through this tier.

- The business-tier**:**

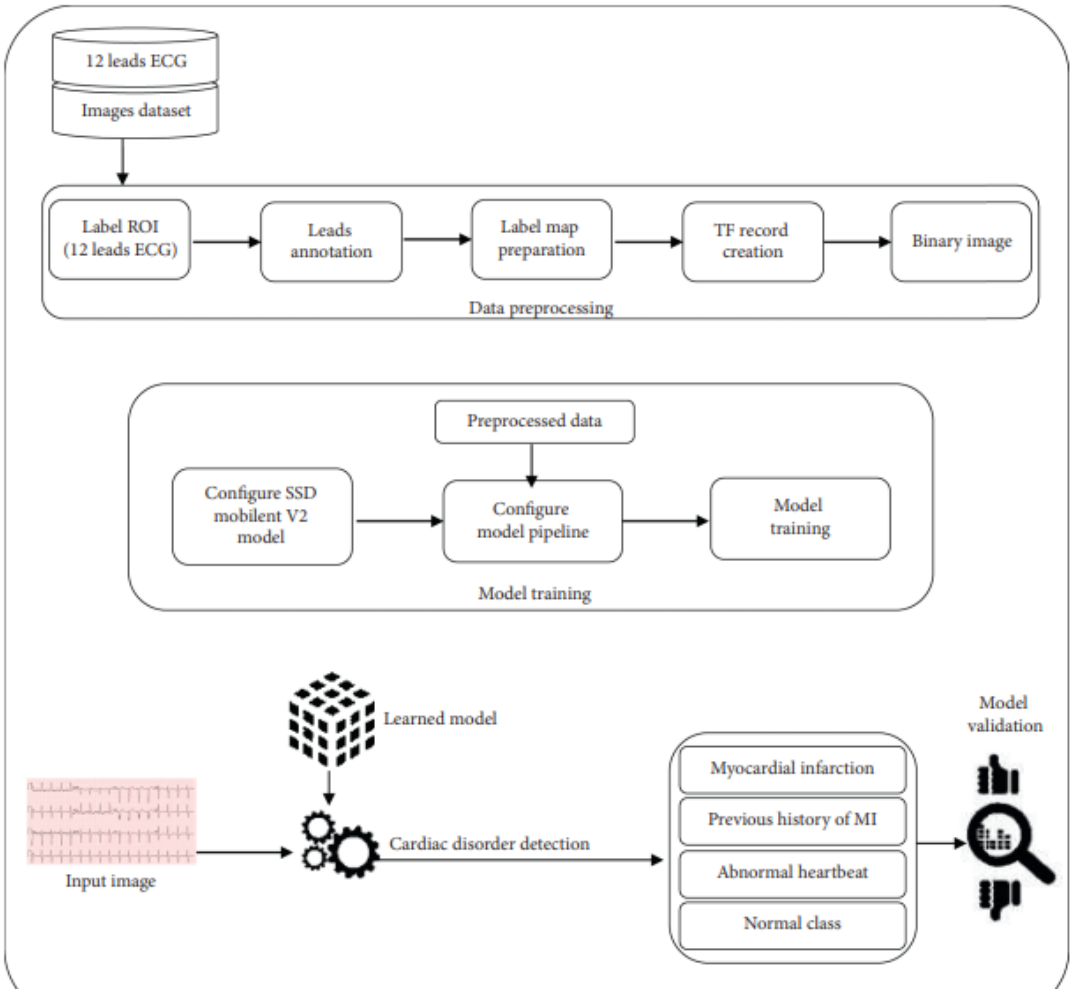
  This tier is responsible for implementing all the business rules of the organization. It operates on the data provided by the users through the web-tier and the data stored in the underlying data-tier. So in a way this tier works on data from the web-tier and the data-tier in order to perform task for the users in agreement with the business rules of the organization.

- The data-tier**:**

  This tier contains the persist able data that is required by the business tier to operate on. Data plays a very important role in the functioning of any organization. Thus, persisting of such data is very important. The data tier performs the job of persisting the data.

4.2 Methodology and Framework

Methodology used in this paper is to process the 12-leadbased ECG images to detect a cardiac abnormality. Single shot detector (SSD) MobileNet v2 is used in this study for model design. SSD is used to detect the objects that can classify and locate the objects in one step. (e block diagram of the model used in this research study is shown in Figure 6. 5.1. Model Configuration. In this study, the model is implemented using Tensorflow API and Google Colab to



build and train a cardiac disorder detector, using pretrained SSD MobileNet V2. In this study authors used the batch size of 24 and 200K training steps for model training. Complete detail of the implementation with working source code is uploaded and can be seen on the GitHub of this project [23]. 5.1.1. Dataset Splitting. (e dataset used in this study is split so that 80% of the images are used for training the algorithms, while the remaining 20% is used for testing each class. 5.1.2. Number of Classes. Classes mean the number of objects that the model should learn and detect after training. In this case, the algorithms are responsible for detecting 4 classes (i.e., normal, abnormal heartbeat, myocardial infarction, and previous history of MI), and each class contains 12 leads; therefore, the number of objects is set to 48. 5.1.3. Learning Rate. A default learning rate of 0.0002 has been used to train the algorithms. 5.2. Training. Once all hyperparameters are configured, then the training process starts with an average step speed of 0.50 s. Training will take almost 4 days to complete the desired steps. The model will validate itself on the test dataset on each iteration and then display the

prediction accuracy and loss. A sufficient number of steps are required for a better prediction of the model. Figure 7 shows the average loss (price paid for the wrong detection) of the model

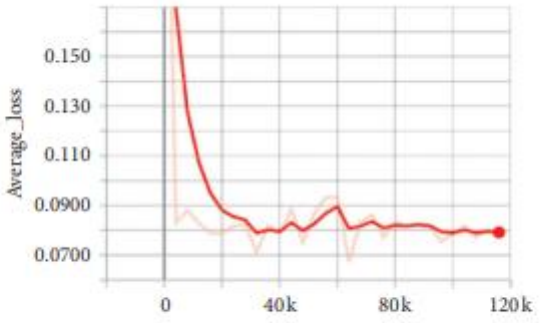


FIGURE 7: Average loss.

TABLE 2: Model evaluation.

Class	Accuracy
Myocardial infarction (MI)	98.33
Abnormal heartbeat	97.22
Previous history of MI	98.28
Normal	96.18

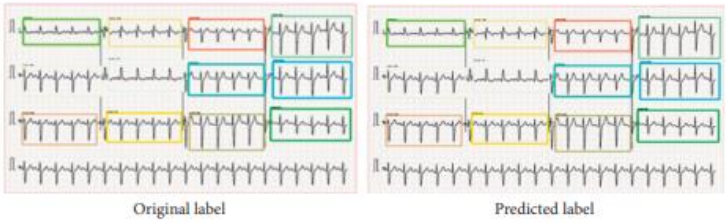


FIGURE 8: Predicted results.

UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business

modelling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.2 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

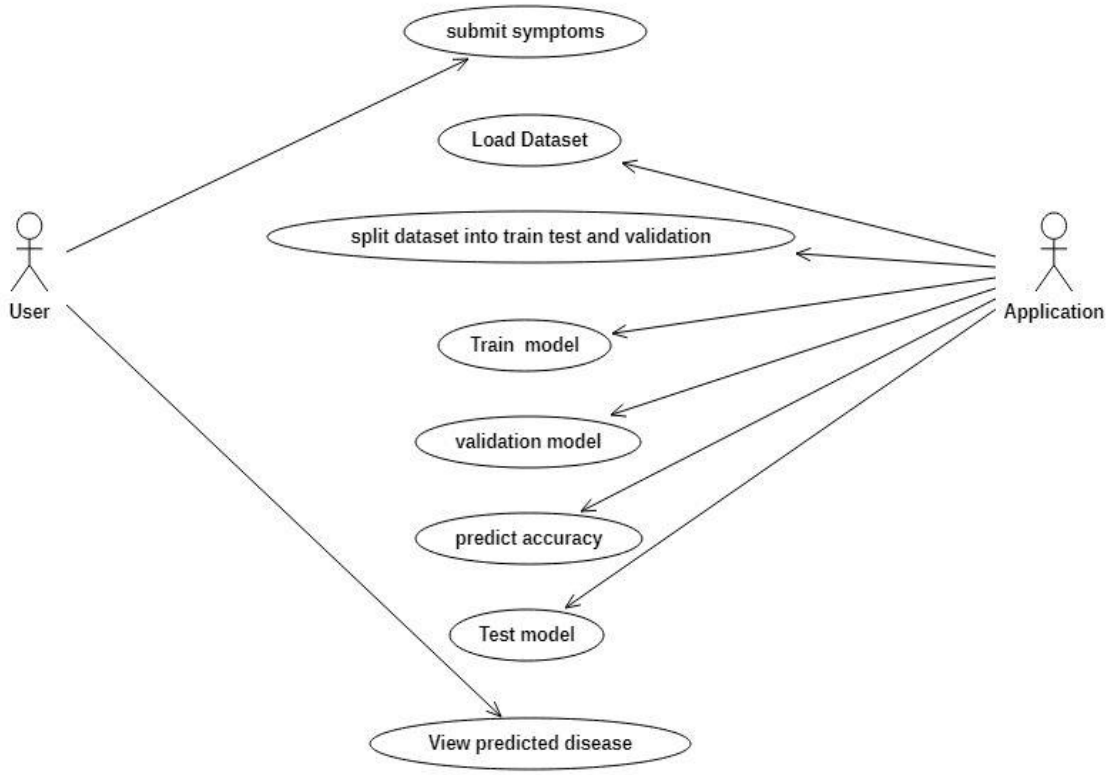


Figure 4.2: Use Case Diagram

4.3 SEQUENCE DIAGRAM:

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioural classifier that represents a declaration of an offered behaviour. Each use case specifies some behaviour, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behaviour of the subject without reference to its internal structure. These behaviours, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment.

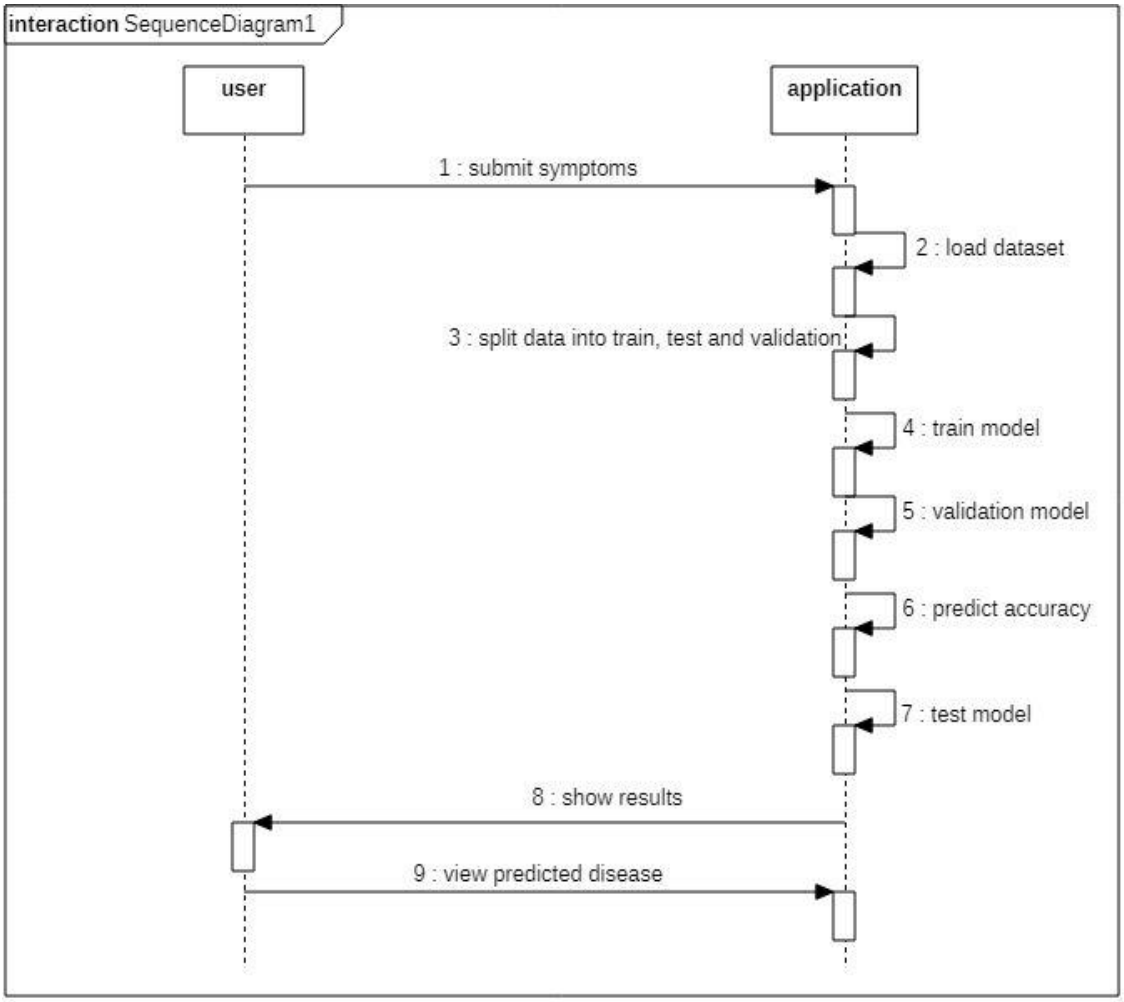


Figure 4.3: Sequence Diagram

4.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

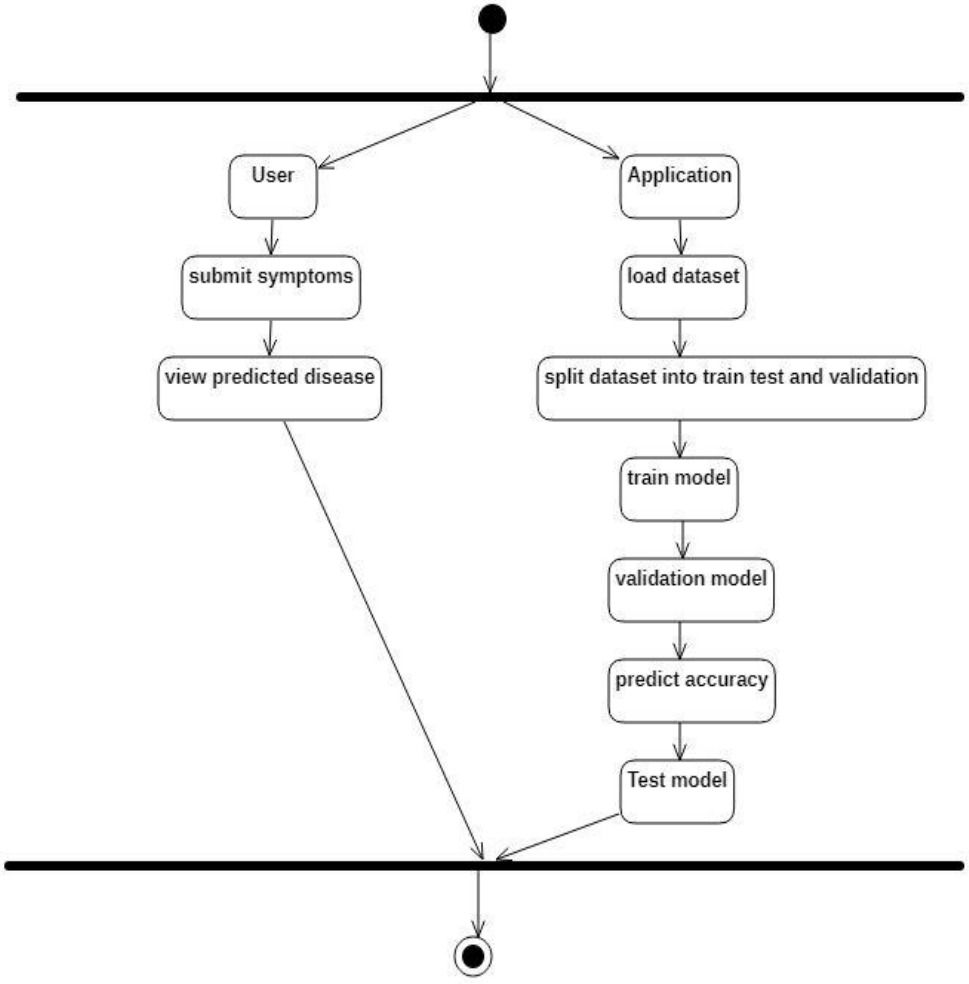


Fig 4.4 Activity Diagram

4.5 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

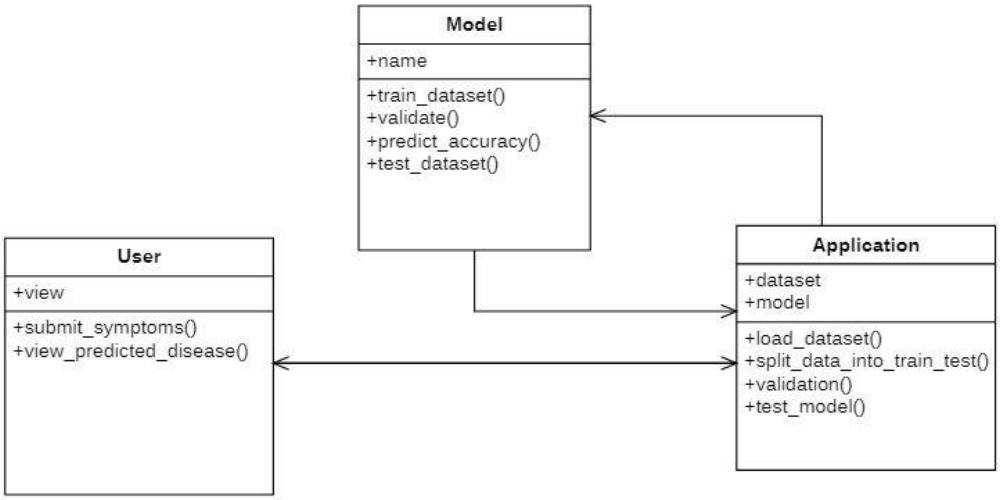


Fig 4.5 Class Diagram

4.6 DEPLOYMENT DIAGRAM:

There may be more steps involved, depending on what specific requirements you have, but below are some of the main steps:

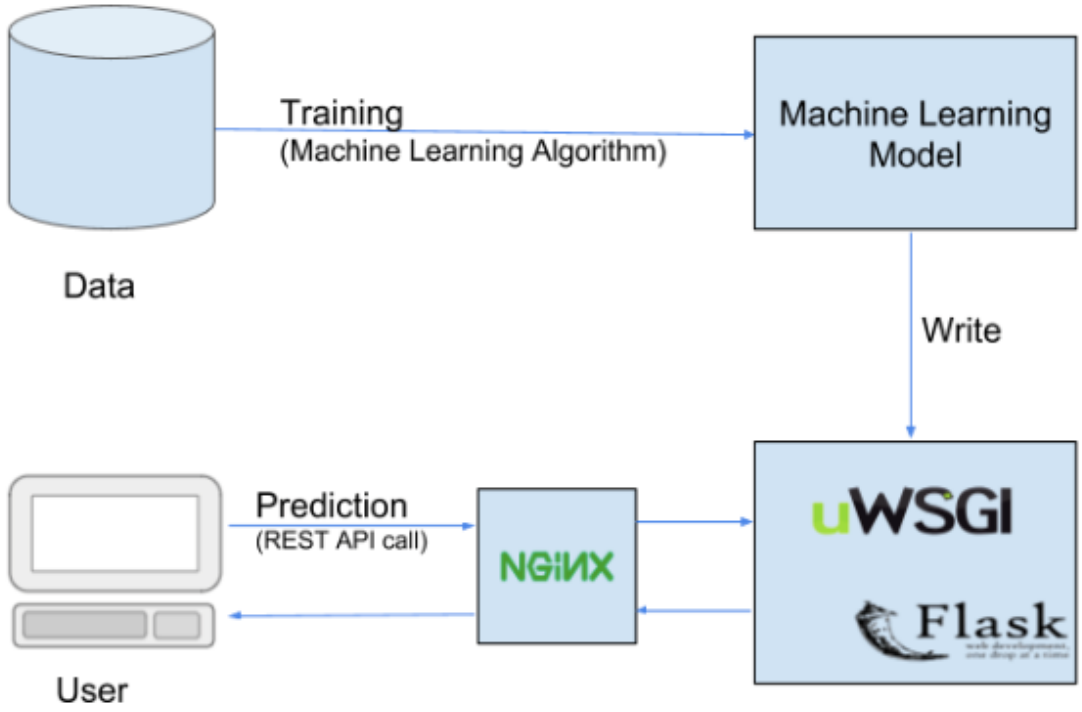


Fig 4.6 Deployment Diagram

4.7 DATAFLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional details.

   Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

   Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

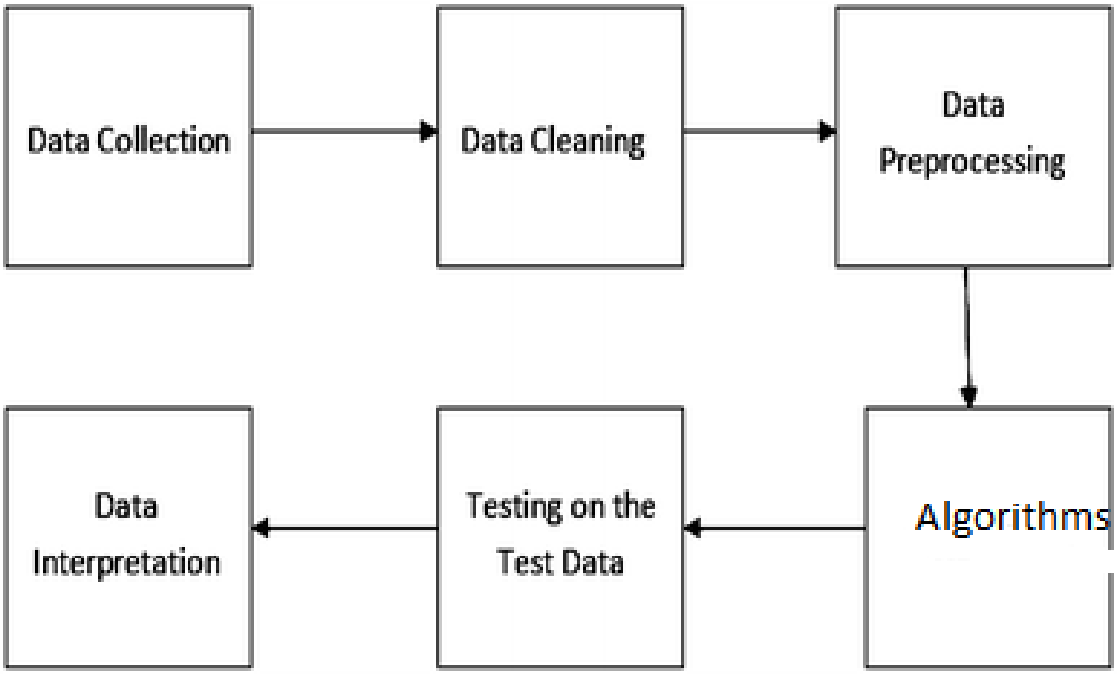


Fig 4.7 Data Flow Diagram

4.7. SYSTEM ARCHITECTURE:

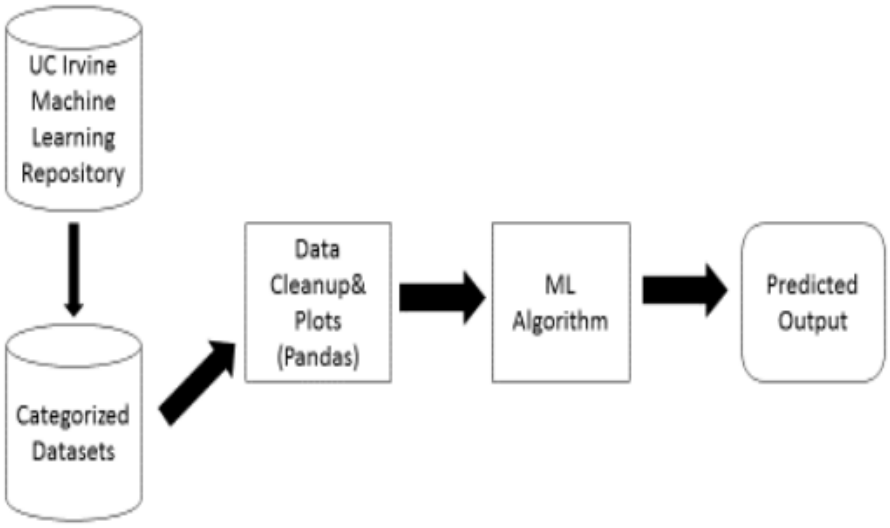


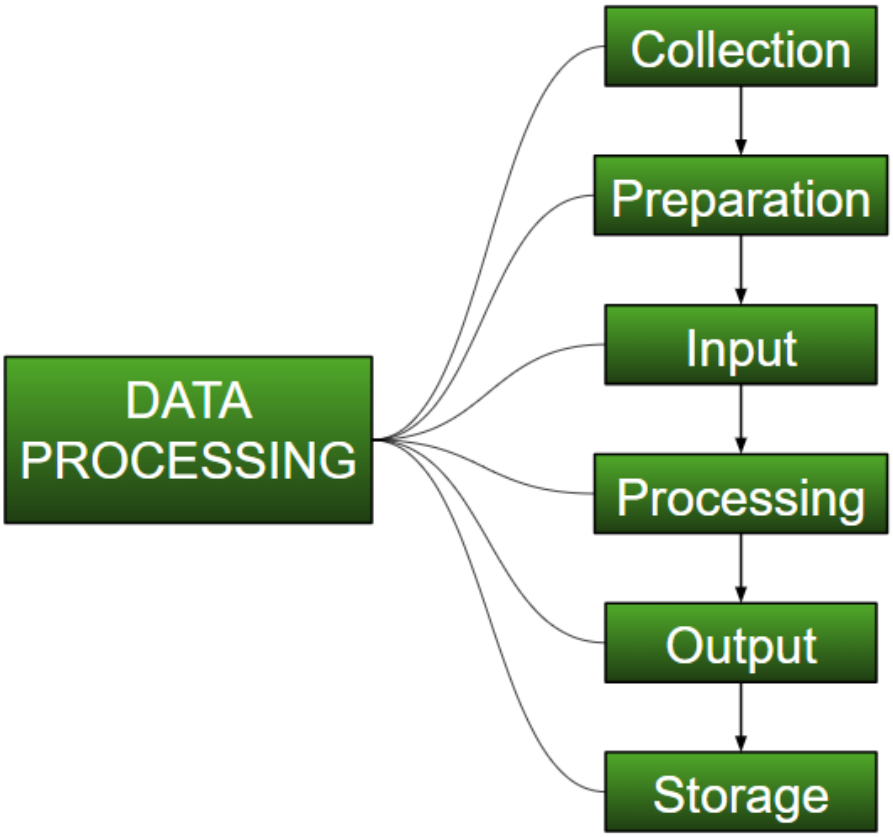
Fig 4.8 System Architecture

5. IMPLEMENTATION AND TESTING

5.1 Implementation

We proposed as an alternative to the user-based neighbourhood approach. We first consider the dimensions of the input and output of the neural network. In order to maximize the amount of training data we can feed to the network, we consider a training example to be a user profile (i.e. a row from the user-item matrix R) with one rating withheld. The loss of the network on that training example must be computed with respect to the single withheld rating. The consequence of this is that each individual rating in the training set corresponds to a training example, rather than each user. As we are interested in what is essentially a regression, we choose to use root mean squared error (RMSE) with respect to known ratings as our loss function. Compared to the mean absolute error, root mean squared error more heavily penalizes predictions which are further off. We reason that this is good in the context of recommender system because predicting a high rating for an item the user did not enjoy significantly impacts the quality of the recommendations. On the other hand, smaller errors in prediction likely result in recommendations that are still useful—perhaps the regression is not exactly correct, but at least the highest predicted rating are likely to be relevant to the user.

Data Processing is a task of converting data from a given form to a much more usable and desired form i.e. making it more meaningful and informative. Using Machine Learning algorithms, mathematical modelling and statistical knowledge, this entire process can be automated. The output of this complete process can be in any desired form like graphs, videos, charts, tables, images and many more, depending on the task we are performing and the requirements of the machine. This might seem to be simple but when it comes to really big organizations like Twitter, Facebook, Administrative bodies like Parliament, UNESCO and health sector organizations, this entire process needs to be performed in a very structured manner.



Collection:

The most crucial step when starting with ML is to have data of good quality and accuracy. Data can be collected from any authenticated source like data.gov.in, Kaggle or UCI dataset repository. For example, while preparing for a competitive exam, students study from the best study material that they can access so that they learn the best to obtain the best results. In the same way, high-quality and accurate data will make the learning process of the model easier and better and at the time of testing, the model would yield state of the art results.
A huge amount of capital, time and resources are consumed in collecting data. Organizations or researchers have to decide what kind of data they need to execute the tasks.
Example: Working on the Facial Expression Recognizer, needs a large number of images having a variety of human expressions. Good data ensures that the results of the model are valid and can be trusted upon.

Preparation:

The collected data can be in a raw form which can't be directly fed to the machine. So, this is a process of collecting datasets from different sources, analyzing these datasets and then constructing a new dataset for further processing and exploration. This preparation can be performed either manually or from the automatic approach. Data can also be prepared in numeric forms also which would fasten the model's learning. **Example:** An image can be converted to a matrix of N X N dimensions, the value of each cell will indicate image pixel.

Input:

Now the prepared data can be in the form that may not be machine-readable, so to convert this data to readable form, some conversion algorithms are needed. For this task to be executed, high computation and accuracy is needed. Example: Data can be collected through the sources like MNIST Digit data(images), twitter comments, audio files, video clips.

Processing:

This is the stage where algorithms and ML techniques are required to perform the instructions provided over a large volume of data with accuracy and optimal computation.

Output:

In this stage, results are procured by the machine in a meaningful manner which can be inferred easily by the user. Output can be in the form of reports, graphs, videos, etc

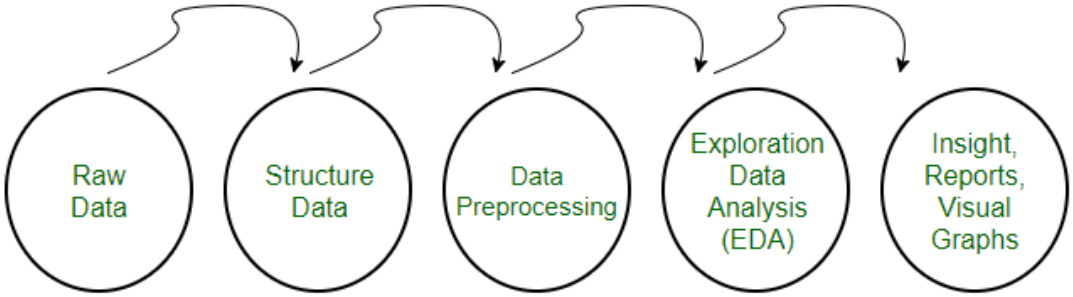
Storage:

This is the final step in which the obtained output and the data model data and all the useful information are saved for the future use.

Data Preprocessing for Machine learning in Python

• Preprocessing refers to the transformations applied to our data before feeding it to the algorithm.

• Data preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.



Need of Data Preprocessing

• For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set.

• Another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen.

Rescale Data

- When our data is comprised of attributes with varying scales, many machine learning algorithms can benefit from rescaling the attributes to all have the same scale.
- This is useful for optimization algorithms in used in the core of machine learning algorithms like gradient descent.
- It is also useful for algorithms that weight inputs like regression and neural networks and algorithms that use distance measures like K-Nearest Neighbors.
- We can rescale your data using scikit-learn using the MinMaxScaler class.

Binarize Data (Make Binary)

- We can transform our data using a binary threshold. All values above the threshold are marked 1 and all equal to or below are marked as 0.
- This is called binarizing your data or threshold your data. It can be useful when you have probabilities that you want to make crisp values. It is also useful when feature engineering and you want to add new features that indicate something meaningful.
- We can create new binary attributes in Python using scikit-learn with the Binarizer class.**Standardize Data**

- Standardization is a useful technique to transform attributes with a Gaussian distribution and differing means and standard deviations to a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.
- We can standardize data using scikit-learn with the StandardScaler class.

Data Cleansing

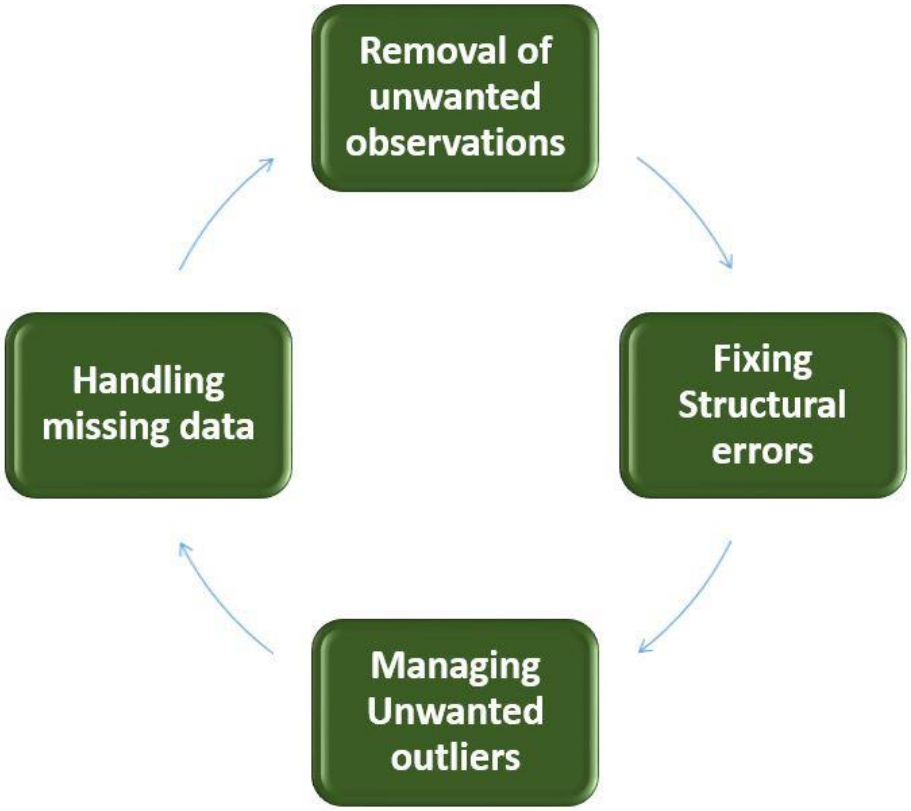
Introduction:

Data cleaning is one of the important parts of machine learning. It plays a significant part in building a model. Data Cleaning is one of those things that everyone does but no one really talks about. It surely isn't the fanciest part of machine learning and at the same time, there aren't any hidden tricks or secrets to uncover. However, proper data cleaning can make or break your project. Professional data scientists usually spend a very large portion of their time on this step.

Because of the belief that, "Better data beats fancier algorithms".

If we have a well-cleaned dataset, we can get desired results even with a very simple algorithm, which can prove very beneficial at times.

Steps involved in Data Cleaning



1. Removal of unwanted observations

This includes deleting duplicate/ redundant or irrelevant values from your dataset. Duplicate observations most frequently arise during data collection and irrelevant observations are those that don't actually fit the specific problem that you're trying to solve.

- Redundant observations alter the efficiency by a great extent as the data repeats and may add towards the correct side or towards the incorrect side, thereby producing unfaithful results.
- Irrelevant observations are any type of data that is of no use to us and can be removed directly.

2. Fixing Structural errors

The errors that arise during measurement transfer of data or other similar situations are called structural errors. Structural errors include typos in the name of features, same attribute with different name, mislabelled classes, i.e. separate classes that should really be the same or inconsistent capitalization.

- For example, the model will treat America and America as different classes or values, though they represent the same value or red, yellow and red-yellow as different classes or attributes, though one class can be included in other two classes. So, these are some structural errors that make our model inefficient and gives poor quality results.

3. Managing Unwanted outliers

Outliers can cause problems with certain types of models. For example, linear regression models are less robust to outliers than decision tree models. Generally, we should not remove outliers until we have a legitimate reason to remove them. Sometimes, removing them improves performance, sometimes not. So, one must have a good reason to remove the outlier, such as suspicious measurements that are unlikely to be the part of real data.

4. Handling missing data

Missing data is a deceptively tricky issue in machine learning. We cannot just ignore or remove the missing observation. They must be handled carefully as they can be an indication of something important. The two most common ways to deal with missing data are:

1. **Dropping observations with missing values.**

Dropping missing values is sub-optimal because when you drop observations, you drop information.

- The fact that the value was missing may be informative in itself.
- Plus, in the real world, you often need to make predictions on new data even if some of the features are missing!

2. Imputing the missing values from past observations.

Imputing missing values is sub-optimal because the value was originally missing but you filled it in, which always leads to a loss in information, no matter how sophisticated your imputation method is.

- Again, "missingness" is almost always informative in itself, and you should tell your algorithm if a value was missing.
- Even if you build a model to impute your values, you're not adding any real information. You're just reinforcing the patterns already provided by other features.
- Both of these approaches are sub-optimal because dropping an observation means dropping information, thereby reducing data and imputing values also is sub-optimal as we fil the values that were not present in the actual dataset, which leads to a loss of information.
- Missing data is like missing a puzzle piece. If you drop it, that's like pretending the puzzle slot isn't there. If you impute it, that's like trying to squeeze in a piece from somewhere else in the puzzle.
- So, missing data is always informative and indication of something important. And we must aware our algorithm of missing data by flagging it. By using this technique of flagging and filling, you are essentially allowing the algorithm to estimate the optimal constant for missingness, instead of just filling it in with the mean.

Some data cleansing tools

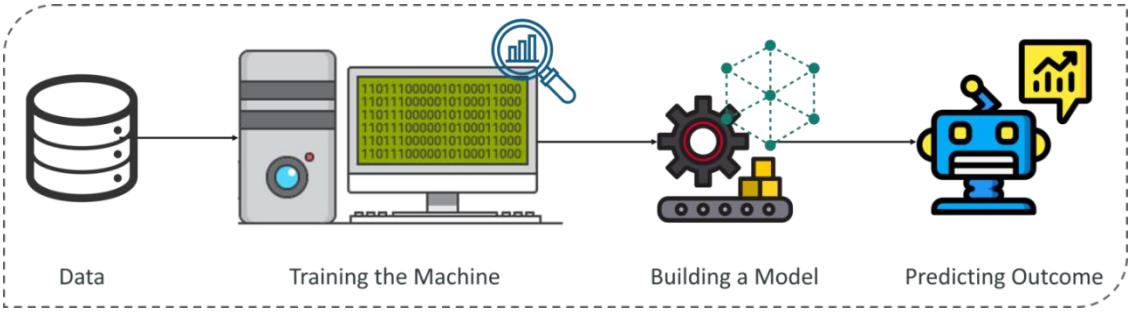
- Openrefine
- Trifacta Wrangler
- TIBCO Clarity
- Cloudingo
- IBM Infosphere Quality Stage

Conclusion

So, we have discussed four different steps in data cleaning to make the data more reliable and to produce good results. After properly completing the Data Cleaning steps, we'll have a robust dataset that avoids many of the most common pitfalls. This step should not be rushed as it proves very beneficial in the further process.

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing.

Machine Learning Process

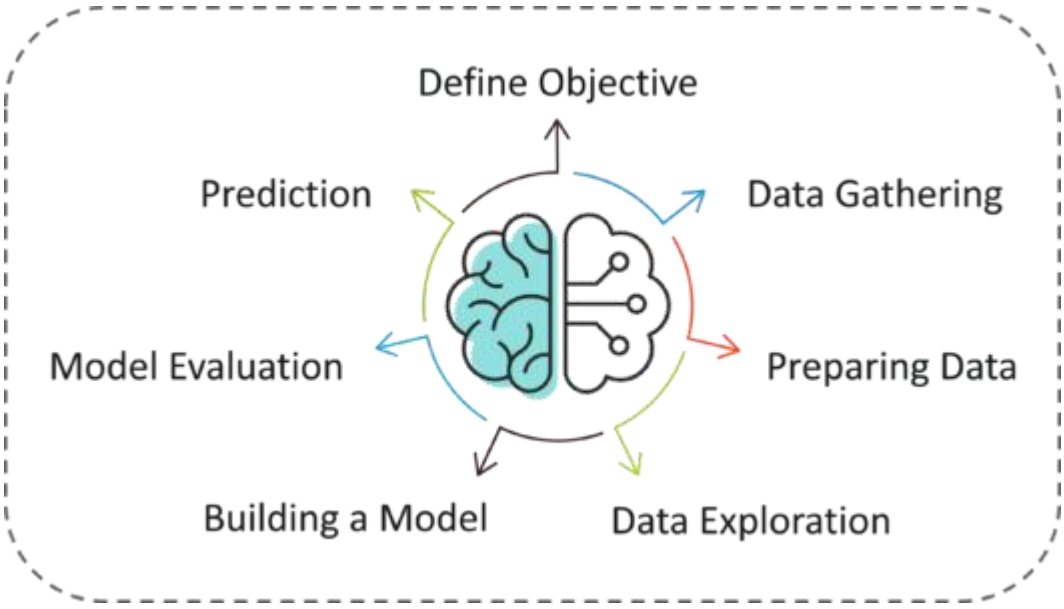


How does Machine Learning Work?

Machine Learning algorithm is trained using a training data set to create a model. When new input data is introduced to the ML algorithm, it makes a prediction on the basis of the model.

The prediction is evaluated for accuracy and if the accuracy is acceptable, the Machine Learning algorithm is deployed. If the accuracy is not acceptable, the Machine Learning algorithm is trained again and again with an augmented training data set.

The Machine Learning process involves building a Predictive model that can be used to find a solution for a Problem Statement. To understand the Machine Learning process let's assume that you have been given a problem that needs to be solved by using Machine Learning.



The below steps are followed in a Machine Learning process:

Step 1: Define the objective of the Problem Statement

At this step, we must understand what exactly needs to be predicted. In our case, the objective is to predict the possibility of rain by studying weather conditions. At this stage, it is also essential to take mental notes on what kind of data can be used to solve this problem or the type of approach you must follow to get to the solution.

Step 2: Data Gathering

At this stage, you must be asking questions such as,

- What kind of data is needed to solve this problem?
- Is the data available?
- How can I get the data?

Once you know the types of data that is required, you must understand how you can derive this data. Data collection can be done manually or by web scraping. However, if you're a beginner and you're just looking to learn Machine Learning you don't have to worry about getting the data. There are 1000s of data resources on the web, you can just download the data set and get going.

Coming back to the problem at hand, the data needed for weather forecasting includes measures such as humidity level, temperature, pressure, locality, whether or not you live in a hill station, etc. Such data must be collected and stored for analysis.

Step 3: Data Preparation

The data you collected is almost never in the right format. You will encounter a lot of inconsistencies in the data set such as missing values, redundant variables, duplicate values, etc. Removing such inconsistencies is very essential because they might lead to wrongful computations and predictions. Therefore, at this stage, you scan the data set for any inconsistencies and you fix them then and there.

Step 4: Exploratory Data Analysis

Grab your detective glasses because this stage is all about diving deep into data and finding all the hidden data mysteries. EDA or Exploratory Data Analysis is the brainstorming stage of Machine Learning. Data Exploration involves understanding the patterns and trends in the data. At this stage, all the useful insights are drawn and correlations between the variables are understood.

For example, in the case of predicting rainfall, we know that there is a strong possibility of rain if the temperature has fallen low. Such correlations must be understood and mapped at this stage.

Step 5: Building a Machine Learning Model

All the insights and patterns derived during Data Exploration are used to build the Machine Learning Model. This stage always begins by splitting the data set into two parts, training data, and testing data. The training data will be used to build and analyze the model. The logic of the model is based on the Machine Learning Algorithm that is being implemented.

Choosing the right algorithm depends on the type of problem you're trying to solve, the data set and the level of complexity of the problem. In the upcoming sections, we will discuss the different types of problems that can be solved by using Machine Learning.

Step 6: Model Evaluation & Optimization

After building a model by using the training data set, it is finally time to put the model to a test. The testing data set is used to check the efficiency of the model and how accurately it can predict the outcome. Once the accuracy is calculated, any further improvements in the model can be implemented at this stage. Methods like parameter tuning and cross-validation can be used to improve the performance of the model.

Step 7: Predictions

Once the model is evaluated and improved, it is finally used to make predictions. The final output can be a Categorical variable (eg. True or False) or it can be a Continuous Quantity (eg. the predicted value of a stock).

In our case, for predicting the occurrence of rainfall, the output will be a categorical variable.

5.3 Results

SSD MobileNet V2 was used to detect cardiovascular disease detection. High accuracy results were calculated for cardiac disease detection. (e mean average precision against all four classes is evaluated. (e study showed high accuracy results to differentiate and detect four cardiac abnormalities. “Myocardial Infarction” and “Previous History of MI” both classes achieved high accuracy among the results of all four classes. Table 2 shows the mean average precision (mAP) of all classes. Figure 8 shows the original label and the predicted output results from the proposed model and their accuracy results. (e authors compare the cardiac disorder detection system performance with existing research studies related to ECG cardiac classification works. However, the mentioned work has a different dataset based on time-series data and different types of classes, so it is not justified to directly compare their results. (is study achieved remarkable results on the detection of four cardiac abnormalities and achieved high accuracy results. Table 3 shows the comparative results related to the study. 6.1. Confusion Matrix. (e confusion matrix presents true positive, false positive, false negative, and true negative categories in each class (i.e., myocardial infarction, abnormal heartbeat, previous history of MI, and normal class). Table 4 displays the confusion matrix of the related study. (e class “myocardial infarction” generates the highest average score. The two major aspects of good prediction are low variance and large datasets. The training dataset has an important and effective role while learning high-level attributes, but the variance plays a vital role in emphasizing key features.

Table 3: Result comparison.

Classifier	Work	Data set	Publicly available	Accuracy	Total classes
SSD MobileNet V2	Proposed	12-lead ECG images	Yes (now available)	98.33	4
CNN	Acharya et al. [24]	Time series	No	94.90	2
1D CNN	Kiranyaz et al. [25]	Time series	Yes	94.73	2
NN	Inan et al. [26]	ECG beats	Yes	95.20	2
Logistic regression	Sadhukhan et al. [27]	Multilead ECG	No	95.60	2
CNN	Lodhi et al. [28]	12-lead ECG	Yes	93.53	2

Table 4: Confusion matrix.

	Myocardial infarction (MI)	Abnormal heartbeat	Previous history of MI	Normal
Myocardial infarction (MI)	96.22	1.89	1.89	0.00
Abnormal heartbeat	3.70	93.59	1.35	1.36
Previous history of MI	0.00	1.61	96.78	1.61
Normal	0.00	2.75	3.51	93.74

DATA DESCRIPTION :

Table 1 shows *the total* numbers of Images used for cardiac disorder detection used for each class.

Table 1: Data description.

Sr.	Class	12-lead ECG	Total images
1	Myocardial infarction (MI)	240	2880
2	Abnormal heartbeat	233	2796
3	Previous history of MI	172	2064
4	Normal	284	3408

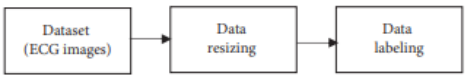


Figure 4: Data preprocessing.

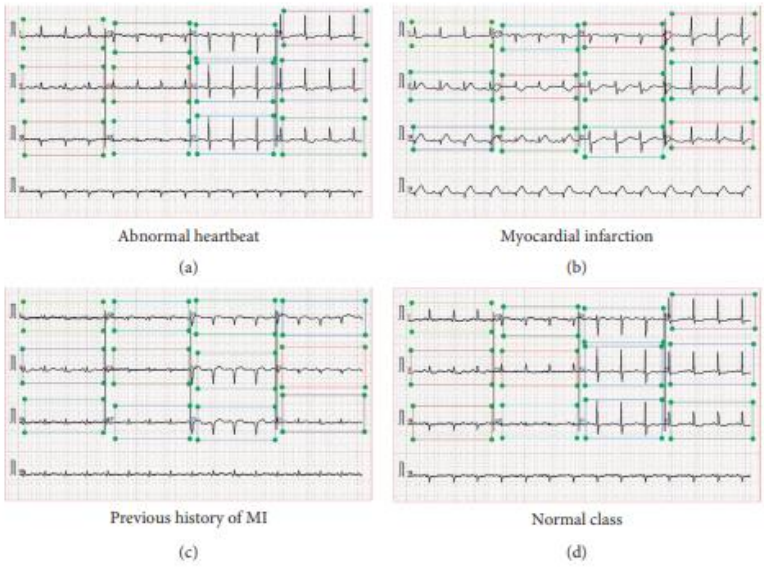


Figure 5: (a) Abnormal heartbeat; (b) myocardial infarction; (c) previous history of MI; (d) normal Class.

Test cases :

Tested	**Test name**	**Inputs**	**Expected output**	**Actual Output**	**status**
1	Load Dataset	Csv file	Read dataset	Load dataset	success
2	Split dataset	Train80% and test20%	Divide the training set and Testing set	Split train and Test	success
	Train Model	Train dataset, random value, predicted class	Train with best accuracy	Train with best accuracy	success
4	Validate Model	No .of Epochs	Validate the Model with best fit	Model Generated	success
5	Predict accuracy and Error Rate	Accuracy	Plot expected accuracy and predicted accuracy	Plot expected predicted accuracy	success
6	Test Data	Test column	Predicted accuracy	Predicted accuracy	success

6.CONCLUSION

6.1 Conclusion

Cardiac disorder detection plays a significant role in medical and health science. (e study focuses on processing the ECG images to detect cardiac abnormalities. (e deep neural network has proven its capabilities in various applications of image processing and computer vision. (is paper critically discusses the related work and analysis, specifically on cardiac disease detection. (e study proposes a generalized methodology to process all formats of ECG. Single shoot detection (SSD) MobileNet v2 based Deep Neural Network architecture was used to detect cardiovascular disease detection. (e study presented high accuracy results in differentiating and detecting four major cardiac abnormalities and showed remarkable results with an average accuracy of 98%. Several cardiologists manually verified the proposed system's accuracy result and recommended that the proposed system be used to screen for cardiac disorder. (e work is relatively rare based on their dataset using standard 12-lead-based ECG images used by cardiac professors in health care institutes. (is work can be extended by training a larger dataset, particularly on more cardiac abnormalities, validating the recognition ratio using DNNs. (e extraction of advanced features on ECG images with image acquisition, adaptive image enhancement, and various boundary detection algorithms on various cardiac-related issues can be detected with medical experts' help with new developing tools. (e way of learning domain adaptation is another big task for researchers to do in the future

6.2 Future Scope

In future, the proposed system can be used in making a software application which could be used in various sections of the society; be it healthcare services, educational institutions, etc. Many services which are externally available to the users could be incorporated in a single application with the help of Google Dialogflow and various other NLU Platforms available in the market and thereby making the application a multifunctioned software. We would encourage the readers to gain a deeper insight of Natural Language Platforms and use to develop applications which will cater the different needs of the society.

7. REFERENCES

[1] Centers for Disease Control and Prevention, Heart Disease Fats, Centers for Disease Control and Prevention, Atlanta, GA, USA, 2020, https://www.cdc.gov/heartdisease/facts.htm.

[2] World Health Organization, Cardiovascular Diseases, Who, Geneva, Switzerland, 2020, http://www.who.int/mediacentre/ factsheets/fs317/en/..

[3] L. Wang, H. Zhang, K. C. Wong, H. Liu, and P. Shi, "Physiological-model-constrained noninvasive reconstruction of volumetric myocardial transmembrane potentials," IEEE Transactions on Bio-Medical Engineering, vol. 57, no. 2, pp. 296–315, 2010.

[4] Q. Zhang, D. Zhou, and X. Zeng, "HeartID: a multiresolution convolutional neural network for ECG-based biometric human identification in smart health applications," IEEE Access, vol. 5, pp. 11805–11816, 2017.

[5] U. R. Acharya, S. L. Oh, Y. Hagiwara et al., "A deep convolutional neural network model to classify heartbeats," Computers in Biology and Medicine, vol. 89, pp. 389–396, 2017.

[6] C. Potes, S. Parvaneh, A. Rahman, and B. Conroy, "Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds," in Proceedings of the 2016 Computing in Cardiology Conference (CinC), Vancouver, Canada, September 2016.

[7] E. D. Ubeyli, "Combining recurrent neural networks with ¨ eigenvector methods for classification of ECG beats," Digital Signal Processing, vol. 19, no. 2, pp. 320–329, 2009.

[8] R. R. Sharma, M. Kumar, and R. B. Pachori, "Automated cad identification system using time frequency representation Table 4: Confusion matrix. Myocardial infarction (MI) Abnormal heartbeat Previous history of MI Normal Myocardial infarction (MI) 96.22 1.89 1.89 0.00 Abnormal heartbeat 3.70 93.59 1.35 1.36 Previous history of MI 0.00 1.61 96.78 1.61 Normal 0.00 2.75 3.51 93.74 Table 3: Result comparison. Classifier Work Data set Publicly available Accuracy Total classes SSD MobileNet V2 Proposed 12-lead ECG images Yes (now available) 98.33 4 CNN Acharya et al. [24] Time series No 94.90 2 1D CNN Kiranyaz et al. [25] Time series Yes 94.73 2 NN Inan et al. [26] ECG beats Yes 95.20 2 Logistic regression Sadhukhan et al. [27] Multilead ECG No 95.60 2 CNN Lodhi et al. [28] 12-lead ECG Yes 93.53 2 Complexity 7 based

on eigenvalue decomposition of ECG signals," Machine Intelligence and Signal Analysis, vol. 748, pp. 597–608, 2019.

[9] P. Hao, K. You, H. Feng et al., "Lung adenocarcinoma diagnosis in one stage," Neurocomputing, vol. 392, pp. 245–252, 2020.

[10] O. Deperlioglu, "Segmentation of heart sounds by Re-sampled signal energy method, BRAIN," Broad Research in Artificial Intelligence and Neuroscience, vol. 9, no. 2, 2018.

[11] S. Ali, S. M. Adnan, T. Nawaz, M. Obaid Ullah, and S. Aziz, "Human heart sounds classification using ensemble methods," Technical Journal, University of Engineering and Technology (UET), vol. 22, no. No. I, pp. 113–120, 2017.

[12] L. Bahekar, A. Mishal, M. Bisen, D. Koche, and S. Alone, "Heart valve diseases detection using anfis and wavelet transform," International Journal of Research in Science & Engineering, vol. 3, no. 2, pp. 279–291, 2017.

[13] M. V. Shervegar and G. V. Bhat, "Automatic segmentation of Phonocardiogram using the occurrence of the cardiac events," Informatics in Medicine Unlocked, vol. 9, no. 1, pp. 6–10, 2017.

[14] B. Xiao, Y. Xu, X. Bi, J. Zhang, and X. Ma, "Heart sounds classification using a novel 1-D convolutional neural network with extremely low parameter consumption," Neurocomputing, vol. 392, pp. 153–159, 2020.

[15] F. Noman, C. M. Ting, S. H. Salleh, and H. Ombao, "Shortsegment heart sound classification using an ensemble of deep convolutional neural networks," in Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1318–1322, Brighton, UK, May 2019.

[16] Y. Xia, Z. Gao, H. Zhang, H. Zhang, and S. Li, "An automatic cardiac arrhythmia classification system with wearable electrocardiogram," IEEE Access, vol. 6, no. 99, p. 1, 2018.

[17] J. Huang, B. Chen, B. Yao, and W. He, "ECG arrhythmia classification using stft-based spectrogram and convolutional neural network," IEEE Access, vol. 7, pp. 92871–92880, 2019.

[18] W. Lu, H. Hou, and J. Chu, "Feature fusion for imbalanced ecg data analysis," Biomedical Signal Processing and Control, vol. 41, pp. 152–160, 2018.

[19] Y. Ji, S. Zhang, and W. Xiao, "Electrocardiogram classification based on faster regions with convolutional neural network," Sensors, vol. 19, no. 11, p. 2558, 2019.

[20] G. Litjens, T. Kooi, B. E. Bejnordi et al., “A survey on deep learning in medical image analysis,” Medical Image Analysis, vol. 42, pp. 60–88, 2017.

[21] C. Park, C. C. Took, and J.-K. Seong, “Machine learning in biomedical engineering,” Biomedical Engineering Letters, vol. 8, no. 1, pp. 1–3, 2018.

[22] A. H. Khan and M. Hussain, “ECG images dataset of cardiac and COVID-19 patients,” Mendeley Data, vol. 1, 2020.

[23] GitHub repository model code, ECG object detection, 2020, https://github.com/alijiskani/ECG_Object_Detection.

[24] U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan, and M. Adam, “Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network,” Information Sciences, vol. 405, pp. 81–90, Sep. 2017.

[25] S. Kiranyaz, T. Ince, and M. Gabbouj, “Real-Time PatientSpecific ECG Classification by 1-D Convolutional Neural Networksfic ECG classiffication by 1-D convolutional neural networks,” IEEE Transactions on Biomedical Engineering, vol. 63, no. 3, pp. 664–675, 2016.

[26] O. T. Inan, L. Giovangrandi, and G. T. A. Kovacs, “Robust neural-network-based classification of premature ventricular contractions using wavelet transform and timing interval features,” IEEE Transactions on Biomedical Engineering, vol. 53, no. 12, pp. 2507–2515, 2006.

[27] D. Sadhukhan, S. Pal, and M. Mitra, “Automated Identification of Myocardial infarction using harmonic phase distribution pattern of ECG data,” IEEE Transactions on Instrumentation and Measurement, vol. 67, no. 99, pp. 1–11, 2018.

[28] A. M. Lodhi, A. N. Qureshi, U. Sharif, and Z. Ashiq, “A novel approach using voting from ECG leads to detect myocardial infarction,” in Proceedings of the SAI Intelligent Systems Conference, pp. 337–352, London, UK, Sept

APPENDIX: SOURCE CODE

```
import numpy as np
import pandas as pd
import tensorflow as tf
import cv2
from glob import glob
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import os
from imutils import paths
import random
imagePaths = sorted(list(paths.list_images("Dataset\\Heart_Disease\\")))
random.seed(42)
random.shuffle(imagePaths)
img_paths=imagePaths
df=img_paths
len(df)
labels = []
import time
start = time.time()

import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model
from tensorflow.keras.applications.mobilenet import preprocess_input
import efficientnet.tfkeras as efc

base_model = efc.EfficientNetB0(weights = 'imagenet')
model=Model(inputs=base_model.input,
outputs=base_model.get_layer('avg_pool').output)

image_size = 224
```

```
features_array = np.zeros((928,1280))

for i, img_path in enumerate(img_paths):
    img = image.load_img(img_path, target_size=(image_size, image_size))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x = preprocess_input(x)
    features = model.predict(x)
    features = features.reshape(1280,)
    features_array[i,:] = features
    label = img_path.split(os.path.sep)[-2]
    #print(label)
    labels.append(label)

print('Running time: %.4f seconds' % (time.time()-start))
features_array.shape
df
len(labels)
df = pd.DataFrame(features_array)
df['label'] = labels
df.head(2)

from sklearn.model_selection import train_test_split
X = df.drop(['label'], axis=1)
y = df.label
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.2)
import time
start = time.time()

from sklearn.svm import SVC
clf = SVC()
clf.fit(Xtrain, ytrain)
preds = clf.predict(Xtest)
```

```
print('Running time: %.4f seconds' % (time.time()-start))

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
print(accuracy_score(ytest, preds))

cm = confusion_matrix(ytest, preds)
sns.heatmap(cm, annot=True, cbar=False)
plt.xlabel('Predicted label')
plt.ylabel('True label')
plt.title('Confusion Matrix')

from sklearn.metrics import classification_report
print(classification_report(ytest, preds))

img = image.load_img("test_images\\test1.jpg", target_size=(image_size, image_size))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)
features = model.predict(x)
features = features.reshape(1280,)

preds = clf.predict([features])

print(preds)

import pickle

with open("svm_efficientnet.pickle", "wb") as output_file:
    pickle.dump(clf, output_file)
with open("svm_efficientnet.pickle", "rb") as input_file:
    svm=pickle.load(input_file)
base_model = efc.EfficientNetB0(weights = 'imagenet')
```

```
model=Model(inputs=base_model.input,
outputs=base_model.get_layer('avg_pool').output)

image_size = 224
features_array = np.zeros((4062,1280))
```