

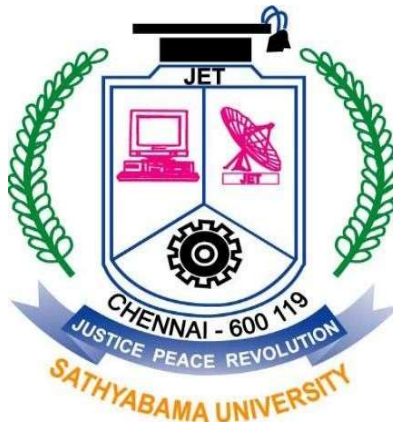
ANALYSIS OF OPTIMIZED HYPER PARAMETERS FOR TIME - FREQUENCY ECG SIGNALS IN CNN

Submitted in partial fulfillment of the requirement for the award of a Bachelor of Engineering Degree in Electronics and Communication Engineering

By

Kuruguntla Nishanth (39130251)

Kothapalli Sriram Varma (39130242)



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

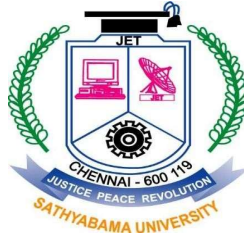
SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119**

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Final Year Project Report is the bonafide work of **Kuruguntla Nishanth (Reg No - 39130251), Kothapalli Sriram Varma (Reg No - 39130242)** who have done the project work as an individual who carried out the project entitled **"ANALYSIS OF OPTIMIZED HYPER PARAMETERS FOR TIME-FREQUENCY ECG SIGNALS IN CNN"** under my supervision from Jan 2023 to April 2023.

Internal Guide

**Dr.M.Sumathi M.E.,PhD,
Professor, Dept of ECE.**

Head of the Department

Dr.T.Ravi M.E., Ph.D.,

Submitted for Viva-voce Examination held on 20.04.2023

Internal Examiner

External Examiner

DECLARATION

I, **Kuruguntla Nishanth (39130251)** hereby declare that the Project Report on **“ANALYSIS OF OPTIMIZED HYPER PARAMETERS FOR TIME-FREQUENCY ECG SIGNALS IN CNN”** done by us under the guidance of **Dr. M. SUMATHI M.E., PhD** at Sathyabama Institute of Science and technology is submitted in partial fulfilment of the requirements of the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

DATE : 20/4/2023

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE



ACKNOWLEDGMENT

We are pleased to acknowledge our sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them

We convey our thanks to **Dr. N. M. NANDHITHA, M.E., Ph.D., Dean**, School of Electrical and Electronics and **Dr. T. RAVI, Ph.D.**, Head of Department, Department of Electronics and Communication Engineering for providing me/us necessary support and details at the right time during the progressive reviews.

We would like to express my sincere and deep sense of gratitude to our Project Guide **Dr. M. SUMATHI M.E., PhD. Professor**, Dept of Electronics and Communication Engineering for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The electrocardiogram (ECG) signal is utilized to assess electrical abnormalities and gives imperative data about human heart well being. There are two sorts of ECG signals normal and abnormal .In this proposed project, conversion of ECG signal to the images are transformed and the new technique introduced was a new time-frequency analysis (TFA) method to study the trend and instantaneous frequency (IF) of nonlinear and non stationary data of ECG signal .The proposed method is termed as synchro extracting transform (SET).The ECG signals are transformed into images and directly given into one of the most popular deep neural networks technique which is the Resnet Convolution Neural Network (CNN). It takes this name from the numerical direct activity between frameworks called convolution .ECG signals are transformed into images and directly updated to the CNN and it trains the input images and then it passes to the hyper- parameter tuning of the convolution neural network (CNN) and results are obtained. Result of the model state that activation function with relu and learning rate 0.001 got a highest accuracy that is 100 percentage and also the least accuracy was 38 percentage which has the activation function as Tanh with learning rate 0.1 so after measuring the accuracy of different parameters by this table we came to know that activation function relu with learning rate 0.001 has a highest accuracy. The key idea of the creating this project is to measure the performance of best hyper tuning parameter for the model.

TABLE OF CONTENTS

S NO.	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	VI
	LIST OF TABLES	VII
	LIST OF ABBREVIATIONS	VIII
1	INTRODUCTION	1
	1.1. ECG	1
	1.2. Time frequency approach	3
	1.3. Convolution Neural Network	4
	1.4. Types of CNN	5
	1.5. Hyper Tuning Parameter	8
2	LITERATURE SURVEY	20
3	METHODOLOGY	22
	3.1. Working	22
	3.2. Transformation of 2D images	23
	3.3. CNN Model	24
4	SOFTWARE DESCRIPTION	25
	4.1. Matlab	25
	4.2. Jupyter Notebook	28
5	RESULT AND ANALYSIS	32
6	CONCLUSION	39
7	Appendix	43
	a. Coding	43
	i. Synchro Extracting Transform using Matlab	43
	ii. CNN using Hyperparameters	45

LIST OF FIGURES

S No	TITLE	PAGE NO
1.	Characteristic's of ECG wave	2
2.	CNN Model	4
3.	Lenet CNN Model	6
4.	Alexnet CNN Model	13
5.	Resnet Expression	14
6.	Architecture of resnet	15
7.	Googlenet CNN inception module	16
8.	Mobilenet CNN architecture	16
9.	Learning rate graph	16
10.	Good enough graph for learning rate	17
11.	DropOut Parameter	17
12.	Number of epoch's Parameter	18
13.	Linear Activation function	18
14.	Sigmoid Activation function	18
15.	Tanh Activation function	19
16.	Relu Activation function	19
17.	Number of Hidden layers and units Parameter	19
1.	Block diagram of system	22
2.	Image OF Normal signal	23
3.	Image OF Abnormal signal	23
4.	CNN Model flow diagram	24
1.	Accuracy graph of Relu activation function with learning rate 0.001	33
2.	Accuracy graph of Relu activation function with learning rate 0.01	33
3.	Accuracy graph of Relu activation function with learning rate 0.1	34
4.	Accuracy graph of Tanh activation function with learning rate 0.1	34
5.	Accuracy graph of Tanh activation function with learning rate 0.01	35
6.	Accuracy graph of Tanh activation function with learning rate 0.001	35
7.	Accuracy graph of Sigmoid activation function with learning rate 0.001	36
8.	Accuracy graph of Sigmoid activation function with learning rate 0.01	36
9.	Accuracy graph of Sigmoid activation function with learning rate 0.1	37

LIST OF TABLES

S.No.	TITLE	PAGE NO
5.1	Accuracy table	37

CHAPTER 1

INTRODUCTION

1.1 ECG

The function of the heart is to contract rhythmically and pump blood to the lungs for oxygenation and then pump this oxygenated blood into the general circulation. This perfect rhythm is continuously maintained and signaled by the spread of electrical signals generated by the heart pacemaker, the Sinoatrial (SA) node. Detecting such electrical activity of the heart can help identify many heart disorders. [1] This is the main concept behind using an ECG (Electrocardiogram), tracing the electrical activity of the heart. By measuring and tracing the potential difference between two points on the outer surface of the body we obtain the simplest ECG chart. Two typical measuring points are between left arm and the right arm. By defining two points and setting up conventional positive direction for measuring the voltage what created is called a 'Lead'. The first phase of cardiac muscle activation is the stimulation of the right and left atria by an electrical signal generated from SA node. This phase appears as the P-wave on the ECG chart. The electrical signal generated, originally generated by the SA node, then spreads through the Atrioventricular node (AV) Junction, the bundle of HIS, and the Purkinje fibers to finally reach and stimulate the ventricles. The spread of electrical signal through ventricles causes ventricular contraction. The phase of ventricular contraction appears as the characteristic QRS complex on ECG chart. Finally, with the relaxation of two ventricles, a depolarization signal is generated and appears as the T-wave on ECG chart. The characteristics of ECG was mentioned at figure 1.1 and following figure shows the characteristics of the ECG wave.

The common noise sources are Baseline wander and ECG amplitude modulation with respiration, Power line interference,[2] Muscle contraction noise, Electrosurgical noise, Motion artifacts, Noise due to variation of electrode skin contact impedance and Noise generated by electronic devices used in signal processing circuits. Power line interference noise is

electromagnetic field from the powerline which causes 50/60 Hz sinusoidal interference. This noise causes problem in interpreting low amplitude waveform like ECG. Although modern instrumentation amplifiers/differential amplifiers have high common mode rejection ratio and other measures like shielding and grounding are taken care of while recording ECG, even then the recordings are often contaminated by powerline interference. In order to avoid the wrong identification of the characteristics of ECG signals and its impact on analysis and diagnostic accuracy, it is necessary to remove the power-line interference effectively.

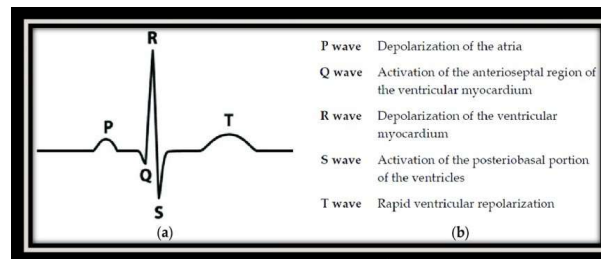


Figure 1.1: Characteristic's of ECG wave

1.1.1 P-WAVE

The P wave represents depolarization of the atria. Atrial depolarization spreads from the SA node towards the AV node, and from the right atrium to the left atrium. The P wave is typically upright in most leads except for aVR; an unusual P wave axis (inverted in other leads) can indicate an ectopicatrial pacemaker. If the P wave is of unusually long duration, it may represent atrial enlargement. Typically a large right atrium gives a tall, peaked P wave while a large left atrium gives a two-humped bifid P wave.

1.1.2 PR INTERVAL

The PR interval is measured from the beginning of the P wave to the beginning of the QRS complex. This interval reflects the time the electrical impulse takes to travel from the sinus node through the AV node. A PR interval shorter than 120 ms suggests that the electrical impulse is bypassing the AV node, as in Wolf- Parkinson-White syndrome. A PR interval consistently longer than 200 ms diagnoses first degree atrioventricular block. The PR segment (the portion of the tracing after the P wave and before the QRS complex) is typically completely flat, but may be depressed in pericarditis.

1.1.3 QRS COMPLEX

The QRS complex represents the rapid depolarization of the right and left ventricles. The ventricles have a large muscle mass compared to the atria, so the QRS complex usually has a much larger amplitude than the P wave. If the QRS complex is wide (longer than 120 ms) it suggests disruption of the heart's conduction system, such as in LBBB, RBBB, or ventricular rhythms such as ventricular tachycardia. Metabolic issues such as severe hyperkalemia, or tricyclic antidepressant overdose can also widen the QRS complex. An unusually tall QRS complex may represent left ventricular hypertrophy while a very low-amplitude QRS complex may represent a pericardial effusion or infiltrative myocardial disease.

1.1.4 J-POINT

The J-point is the point at which the QRS complex finishes and the ST segment begins. The J-point may be elevated as a normal variant. The appearance of a separate J wave or Osborn wave at the J-point is pathognomonic of hypothermia or hypercalcemia.

1.1.5 ST - SEGMENT

The ST segment connects the QRS complex and the T wave; it represents the period when the ventricles are depolarized. It is usually isoelectric, but may be depressed or elevated with myocardial infarction or ischemia. ST depression can also be caused by LVH or digoxin. ST elevation can also be caused by pericarditis, Brugada syndrome, or can be a normal variant (J-point elevation).

1.1.6 T - WAVE

The T wave represents the repolarization of the ventricles. It is generally upright in all leads except aVR and lead V1. Inverted T waves can be a sign of myocardial ischemia, left ventricular hypertrophy, high intracranial pressure, or metabolic abnormalities. Peaked T waves can be a sign of hyperkalemia or very early myocardial infarction.

1.1.7 U - WAVE

The U wave is hypothesized to be caused by the repolarization of the interventricular septum. It normally has a low amplitude, and even more often is completely absent. A very prominent U wave can be a sign of hypokalemia, hypercalcemia or hyperthyroidism.

1.2 Time frequency approach

In signal handling, the time-recurrence investigation involves those strategies that concentrate on a sign in both the time and recurrence areas while utilizing different time-recurrence portrayals. Instead of reviewing a 1-layered signal (a capacity, genuine or complex-esteemed, whose area is the genuine line) and some change (another capacity whose space is the genuine line, acquired from the first through some change), time-recurrence investigation concentrates on a two-layered signal a capacity whose area is the two-layered genuine plane, got from the sign by means of a period recurrence change. The numerical inspiration for this study is that capacities and their change portrayal are firmly associated, and they can be seen better by concentrating on them mutually, as a two-layered object, as opposed to independently. A straightforward model is that the 4-overlay periodicity of the Fourier change and the way that two-overlap Fourier change switches course - can be deciphered by considering the Fourier change as a 90° turn in the related time-recurrence plane: 4 such revolutions yield the character, and 2 such pivots just converse bearing (reflection through the beginning).

The viable inspiration for time-recurrence examination is that old-style Fourier investigation expects that signs are endless in time or intermittent, while many signs by and by are of brief-term and change significantly over their span. For instance, customary instruments don't create limitless length sinusoids, however rather start with an assault, then, at that point, slowly rot. This is inadequately addressed by conventional techniques, which persuades time-recurrence examination. One of the most fundamental types of time-recurrence investigation is the brief time frame Fourier change (STFT), however more complex procedures have been grown, quiet wavelets and least-squares otherworldly examination techniques for unevenly separated information.

1.3 Convolution Neural Network

A convolutional brain organization (CNN) is a kind of fake brain network utilized in picture acknowledgment and handling that is explicitly intended to deal with pixel information. A Convolutional brain organization (CNN) is a brain network that has at least one convolutional layers and are utilized

chiefly for picture handling, arrangement, division and furthermore for other auto connected information. CNNs are strong picture handling, computerized reasoning (AI) that utilization pro- found figuring out how to perform both generative and illustrative assignments, frequently utilizing machine vision that incorporates picture and video acknowledgment, alongside recommender frame- works and regular language handling (NLP). A brain network is an arrangement of equipment or potentially programming designed after the activity of neurons in the human mind.

Customary brain networks are not great for picture handling and should be taken care of pictures in decreased goal pieces. A CNN utilizes a framework similar as a multi-facet perceptron that has been intended for decreased handling necessities. CNN is a kind of profound learning model for handling information that has a framework design, for example, pictures, which is enlivened by the association of creature visual cortex [3] and intended to consequently and adaptively learn spatial ordered progressions of highlights, from low-to significant level examples. A CNN engineering is framed by a heap of unmistakable layers that change the info volume into a result volume (for example holding the class scores) through a differentiable capacity.

A couple of unmistakable sorts of layers are generally utilized. Convolutional networks were propelled by natural cycles in that the availability design between neurons looks like the association of the creature visual cortex. Individual cortical neurons answer boosts just in a confined district of the visual field known as the open field.

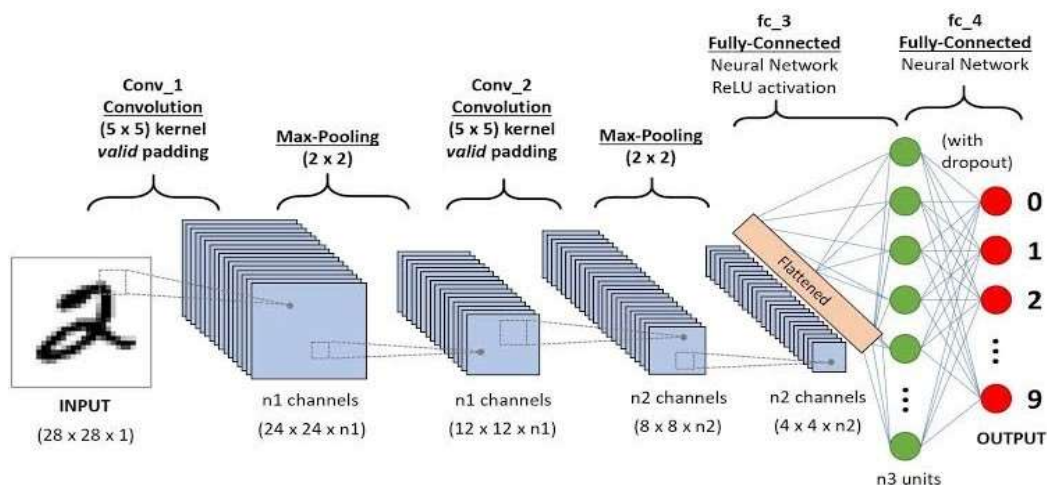


Figure 1.2: CNN Model

1.4 Types of CNN

- LENET
- ALEXNET
- RESNET
- GOOGLNET
- MOBILENET

1.4.1 LENET

LeNet is the most well known CNN engineering it is likewise the primary CNN Model which came in the year 1998. LeNet was initially evolved to order written by hand digits from 0-9 of the MNIST Dataset. It is comprised of seven layers, each with its own arrangement of teachable boundaries. It acknowledges a 32×32 pixel picture, which is somewhat immense in contrast with the pictures in the informational indexes used to prepare the organization.[4]

- The First Convolutional Layer is comprised of 6 5×5 channels with a step of 1.
- The Second Layer is a 2×2 normal pooling or "sub-testing" layer with a step of 2.
- The Third Layer is similarly a Convolutional layer, with 16 5×5 filters and a stride of 1.
- The Fourth Layer is another 2×2 average-pooling layer with a stride of 2.
- The fifth layer is basically connecting the output of the fourth layer with a fully connected layer consisting of 120 nodes.
- The Sixth Layer is a similarly fully-connected layer with 84 nodes that derives from the output of the Fifth Layer's 120 nodes.

The seventh layer consists of categorising the output of the previous layer into ten classifications based on the 10-digits it was trained to identify.

It was one of the principal powerful digit-acknowledgment calculations for ordering transcribed digits. In any case, this organization was inadequate as

far as figuring cost and precision when it came to handling enormous pictures and arranging among countless article classes. Figure 1.3 shows the Lenet CNN model flow we can show it below.

1.4.2 ALEXNET

Beginning with a 11x11 bit, Alexnet is developed of 5 conv layers. For the three monstrous direct layers, it was the primary plan to utilize max-pooling layers, ReLu enactment capacities, and dropout. The organization was utilized to characterize pictures into 1000 distinct classes. The organization is like the LeNet Architecture, yet it incorporates much a greater number of channels than the first

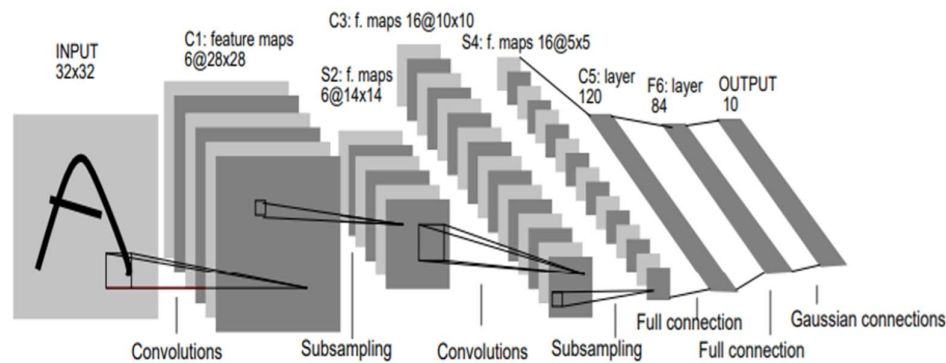


Figure 1.3: Lenet CNN Model

LeNet, permitting it to order significantly more items. Moreover, it manages overfitting by utilizing "dropout" as opposed to regularization.[5]

- Initial, a Convolution Layer (CL) with 96 11 X 11 channels and a step of 4. From that point onward, a Max-Pooling Layer (M-PL) with a channel size of 3 X 3 and a step of 2 is applied.
- A CL of 256 channels of size 5 X 5 and step = 4 is utilized again. Then a CL with a channel size of 3 X 3 and a step of 2 was utilized.
- A CL of 384 channels of size 3 X 3 and step = 4 was utilized. Once more, a CL of 384 channels of size 3 X 3 and step = 4.
- A CL of 256 channels of size 3 X 3 and step = 4 was utilized once more. At

the point when the result of the last layer is changed to an information layer, as in the Fully Linked Block, it has 9261 hubs, which are all completely associated with a secret layer with 4096 hubs.

- The primary secret layer is by and by completely connected to a 4096-hub stowed away layer.

Two GPUs were utilized to prepare the underlying organization. It contains eight layers, each with own arrangement of settings might be learned. RGB photographs are utilized as contribution to the Model. Relu is the enactment work used in all levels.[20] Two Dropout layers were utilized. Softmax is the initiation work used in the result layer..Figure 1.4 shows the Alexnet CNN model flow we can show it below.[6]

1.4.3 RESNET

ResNet is a notable profound learning model that was first presented in a paper by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang. In 2015, a review named "Profound Residual Learning for Image Recognition" was distributed. ResNet is one of the most broadly utilized and viable profound learning models to date. ResNets[23] are comprised of what's known as a lingering block. This is based on the idea of "skip-associations" and uses a great deal of cluster standardization to allow it to prepare many layers effectively without forfeiting speed over the long run..Figure 1.5 shows the Resnet CNN model flow we can show it below.[7]

The primary thing we note in the above outline is that there is an immediate connection that avoids a few of the model's levels. The 'skip association,' as it is known, lies at the center of leftover squares. On account of the skip association, the result isn't something similar. Without the skip association, input 'X' is duplicated by the layer's loads, then, at that point, adding of an inclination term. The Architecture configuration is motivated on VGG-19 and has a 34-layer plain organization to which alternate way and skip associations are added. As found in the chart underneath, these skip associations or remaining squares change the plan into a leftover organization. Figure 1.6 shows the Resnet Architecture model we can show it below.

After the primary CNN-based design (AlexNet) that success the ImageNet 2012 contest, Every result- ing winning engineering involves more layers in a

profound brain organization to diminish the mistake rate. This works for less number of layers, yet when we increment the quantity of layers, there is a typical issue in profound learning related with that called Vanishing/Exploding slope. This makes the angle become 0 or excessively enormous. In this way when we builds number of layers, the preparation and test mistake rate additionally increments.

To take care of the issue of the evaporating/detonating slope, this engineering presented the idea called Residual Network. In this organization we utilize a method called skip associations . The skip association skips preparing from a couple of layers and interfaces straightforwardly to the result. The methodology behind this organization is rather than layers get familiar with the hidden planning, we permit network fit the leftover planning.

1.4.4 GOOGLNET

The ILSVRC 2014 rivalry was won by the GoogleNet or Inception Network, which had a main 5 blunder pace of 6.67 percent, which was essentially human level execution. Google made the model, which fuses a better execution of the first LeNet plan. This depends on the origin module idea. GoogLeNet is a variety of the Inception Network, which is a 22-layer profound convolutional brain organization.[8] GoogLeNet is presently used for an assortment of PC vision applications, including face discovery and distinguishing proof, antagonistic preparation, etc. The inception Module looks like fig no 1.7 The InceptionNet/GoogLeNet configuration is comprised of nine origin modules stacked on top of one another, with max-pooling layers between them (to split the spatial aspects). It is comprised of 22 layers (27 with the pooling layers). After the last beginning module, it utilizes worldwide normal pooling.

1.4.5 MOBILENET

Nine starting modules are placed on top of each other in the InceptionNet/GoogLeNet design, with max- pooling layers between them (to divide the spatial perspectives). There are 22 layers overall (27 with the pooling layers). It uses overall ordinary pooling after the last initiation

module. The MobileNet model depends on depthwise recognizable convolutions, which are a sort of factorized convolution that parcels a standard convolution into a depthwise convolution and a pointwise convolution. The depthwise convolution used by MobileNets applies a singular channel to every information channel. The depthwise convolution's outcomes are then joined using an 11 convolution by the point-wise convolution. In one phase, a standard convolution channels and mixes commitments to make one more game plan of results. The depthwise separable convolution segments this into two layers: one for isolating and the other for combining. Figure 1.8 shows the Lenet CNN model flow we can show it below.

Besides the essential layer, which is a completed convolution, the MobileNet structure relies upon depthwise separable convolutions, as discussed in the main region. Aside from the last totally related layer, which has no nonlinearity and feeds into a softmax layer for portrayal, all layers are followed by a batch norm and ReLU nonlinearity.

1.5 Hyper Tuning Parameter

Hyperparameters are boundaries whose values control the learning system and decide the upsides of model boundaries that a learning calculation winds up learning. The prefix 'hyper' recommends that they are 'high level' boundaries that control the gaining system and the model boundaries that outcome from it. As an AI engineer planning a model, you pick and set hyperparameter values that your learning calculation will use before the preparation of the model even starts. In this light, hyperparameters are supposed to be outside to the model on the grounds that the model can't change its qualities during getting the hang of/ preparing. Hyperparameters are utilized by the learning calculation when it is advancing yet they are not piece of the subsequent model. Toward the finish of the learning system, we have the prepared model boundaries which successfully is what we allude to as the model. The hyperparameters that were utilized during preparing are not piece of this model. We can't for example know what hyperparameter values were utilized to prepare a model from the actual model, we just know the model boundaries that were learned. Essentially, anything in AI and profound discovering that you choose their qualities or pick their arrangement prior to preparing starts and whose qualities or setup will continue as before

while preparing closes is a hyperparameter.

Types of Hyper Tuning parameters:

- Number of epochs
- Batch Size
- Activation Function
- Number of hidden layers and units
- Dropout for regularization

1.5.1 Learning rate

The layers of a brain network are aggregated and an enhancer is appointed. The streamlining agent is mindful to change the learning rate and loads of neurons in the brain organization to arrive at the base misfortune work. The analyzer is vital to accomplish the conceivable most noteworthy precision or least misfortune. One of the hyperparameters in the enhancer is the learning rate. We will likewise tune the learning rate. Learning rate controls the progression size for a model to arrive at the base misfortune work. A higher learning rate causes the model to learn quicker, however, it might miss the base misfortune work and just arrive at its encompassing.[6] A lower learning rate allows a superior opportunity to find a base misfortune work. As a tradeoff lower learning rate needs higher ages, or additional time and memory limit assets.

The sum that the loads are refreshed during preparing is alluded to as the progression size or the "learning rate." In particular, the learning rate is a configurable hyperparameter utilized in the preparation of brain networks that has a little sure worth, frequently in the reach somewhere in the range of 0.0 and 1.0. learning rate, a positive scalar deciding the size of the progression. The learning rate is many times addressed utilizing the documentation of the lowercase Greek letter estimated time of arrival (η).

During preparation, the backpropagation of mistake assesses how much blunder for which loads of a hub in the organization are mindful. Rather than refreshing the load with everything, it is scaled by the learning rate. This implies that a learning pace of 0.1, generally normal default esteem, would

imply that loads in the organization are refreshed $0.1 \times$ (assessed weight blunder) or 10 of the assessed weight mistake each time the loads are refreshed.

1.5.2 Dropout Regularization

Dropout is a strategy where arbitrarily chosen neurons are overlooked during preparing. They are "exited" haphazardly. This implies that their commitment to the enactment of downstream neurons is transiently eliminated on the forward pass and any weight refreshes are not applied to the neuron on the regressive pass. As a brain network learns, neuron loads sink into their setting inside the organization. Loads of neurons are tuned for explicit highlights giving some specialization. Adjoining neurons become to depend on this specialization, which whenever taken excessively far can result in a delicate model excessively concentrated on the preparation of information. This dependent on set for a neuron during preparation alludes to complex co-transformations. You can envision that assuming neurons are arbitrarily exited the organization during preparation, different neurons should step in and handle the portrayal expected to make forecasts for the missing neurons. This is accepted to bring about numerous free inner portrayals being advanced by the organization. The impact is that the organization turns out to be less touchy to the particular loads of neurons. This thus brings about an organization that is prepared to do better speculation and is less inclined to overfit the preparation information. Figure 1.9 shows the Dropout parameters we can show it below.

1.5.3 Number of epochs and Batch Size

The quantity of ages indicates how many times the whole arrangement of information preparing information is gone through the brain network during preparation. It is important to use numerous ages during preparation in light of the fact that this permits the organization to change its loads during preparation.[5] Utilizing too not many or such a large number of ages can make the organization underfit or overfit; both of these situations make the organization have a horrible showing. Essentially, managed AI endeavors to surmise an objective capacity that guides its given authentic contributions to

verifiable results. Since a capacity is being approximated, the capacity ought to simply be speculation as we just have a fragmented, loud example of information. This summed-up work is alluded to as inductive getting the hang of, meaning the AI calculation is attempting to learn general ideas that are fundamental in the particular preparation of information.

This thought of summed up learning is significant on the grounds that it permits an AI calculation to utilize these overall ideas to make expectations on future information. For the calculation to learn summed up ideas, we need to stay away from over-fitting, and that implies that the calculation is too thoroughly prepared on the preparation information. Over-fitting implies that the model has advanced a lot of insight concerning the preparation information (like commotion), so the model will not be able, to sum up. Subsequently, the model can not make precise forecasts on inconspicuous information. On the other hand, under-fitting implies that the model doesn't represent the fundamental connections in the given information, which additionally will bring about unfortunate model execution. The objective for choosing the number of ages is to stay away from over-fitting and under fitting to upgrade model execution. The lattice search was performed on ten stocks from the IT area.

The table beneath shows an example of the examinations that were run. In the arrangement of examinations. the clump size was kept consistent at size 50, and the number of ages was shifted. The best outcomes were accomplished utilizing 10 ages. A number of ages is the times the entire preparation information is displayed to the organization while preparing. Increment the number of ages until the approval exactness begins diminishing in any event, while preparing precision is increasing (over-fitting). Figure 1.10 shows the Number of epoch's Parameter model we can show it below.

1.5.4 Activation function

An activation function is a parameter in each layer. Input data are fed to the input layer, followed by hidden layers, and the final output layer. The output layer contains the output value.

The input values moving from a layer to another layer keep changing according to the activation function. The activation function decides how to compute the input values of a layer into output values.

The output values of a layer are then passed to the next layer as input values again. The next layer then computes the values into output values for another layer again.[22] There are 9 activation functions to tune in to this demonstration. Each activation function has its own formula (and graph) to compute the input values. It will not be discussed in this article.

Types of Activation function

- Linear Function
- Sigmoid Function
- Tanh Function
- Relu Function

1.5.5 Linear function

On the off chance that we don't have any significant bearing on an actuation work then the resultsignwould be straightforward direct capacity. A direct capacity is not difficult to tackle however they are restricted in their intricacy and have less power t gain complex useful mappings from information.

Likewise without enactment work our brain organizations wouldn't have the option to learn and different sorts of information, for example, pictures, recordings, sound, discourse and so forth.Figure

1.11 shows the Linear Activation function we can show it below. Linear $f(x) = x$

1.5.6 Sigmoid function

This capacity accepts any genuine worth as info and results in values in the scope of 0 to 1. The bigger the info (more sure), the nearer the result worth will be to 1, though the more modest the information (more regrettable), the nearer the result will be to 0.Figure 1.12 shows the Sigmoid Activation

function we can show it below.

1.5.7 Tanh function

The tanh function is very similar to the sigmoid function. It is actually just a scaled version of the sigmoid function.

$$\tanh(x) = 2\text{sigmoid}(2x) - 1$$

It can be directly written as $\tanh(x) = 2/(1 + e^{-2x}) - 1$. Tanh works similar to the sigmoid function but is symmetric over the origin. It ranges from -1 to 1. Figure 1.13 shows the Tanh Activation function we can show it below.

1.5.8 Relu function

The Relu function performs a threshold operation on each input element where all values less than zero are set to zero. Thus,

The Relu is represented as $\text{Relu} \max(0, x)$. Figure 1.14 shows the Relu Activation function we can show it below.

1.5.9 Number of Hidden layers and units

The quantity of secret layers d is a really basic boundary for deciding the general design of brain organizations, which affects the last result.[11] I have seen quite often that profound learning networks with additional layers frequently acquire more perplexing elements and somewhat higher exactness making this a standard way to deal with accomplishing improved results.

The quantity of units addresses the number of neurons of a layer[19]. A unit takes inputs from every one of the hubs in the layer beneath, does its estimations, and afterward yields them to the layer above. Accordingly, the quantity of neurons in a layer controls the authentic limit of the organization.

Since 50 neurons have been the default in this testing system, I chose to test the number of neurons above and underneath this number to check whether there were any upgrades in forecasts. This setup utilizing 50 neurons actually had a slight edge in prescient precision.

Diminishing the number of neurons prompted an extreme lessening in the capacity of the calculation to foresee the right directional development of the cost, an expansion in the number of neurons still somewhat failed to meet expectations of the 50-neuron design. Figure 1.15 shows the Number of Hidden layers and units Parameter we can show it below.

Since 50 neurons have been the default in this testing system, I chose to test the number of neurons above and underneath this number to check whether there were any upgrades in forecasts

The quantity of layers d is a really basic boundary for deciding the general design of brain organizations, which affects the last result.[11] I have seen quite often that profound learning networks with additional layers frequently acquire more perplexing elements and somewhat higher exactness making this a standard way to deal with accomplishing improved results.

The quantity of units addresses the number of neurons of a layer[19]. A unit takes inputs from every one of the hubs in the layer beneath, does its estimations, and afterward yields them to the layer above. Accordingly, the quantity of neurons in a layer controls the authentic limit of the organization.

The quantity of secret layers d is a really basic boundary for deciding the general design of brain organizations, which affects the last result.[11] I have seen quite often that profound learning networks with additional layers frequently acquire more perplexing elements and somewhat higher exactness making this a standard way to deal with accomplishing improved results.

Diminishing the number of neurons prompted an extreme lessening in the capacity of the calculation to foresee the right directional development of the cost, an expansion in the number of neurons still somewhat failed to meet expectations of the 50-neuron design. Figure 1.15 shows the Number of Hidden layers and units Parameter we can show it below.

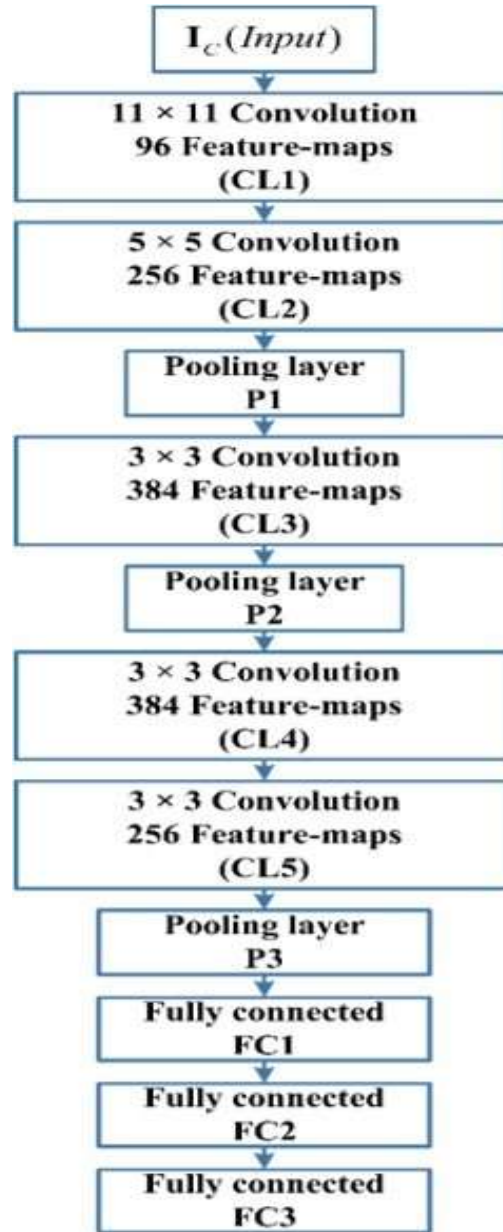


Figure 1.4: Alexnet CNN Model

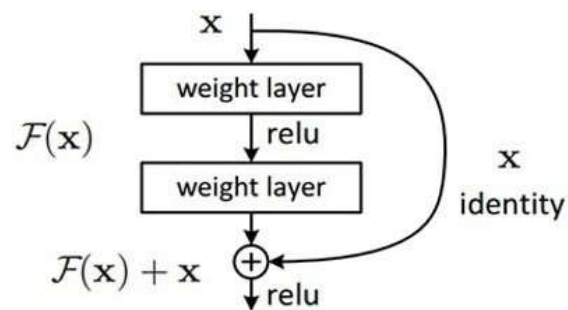


Figure 1.5: Resnet Expression

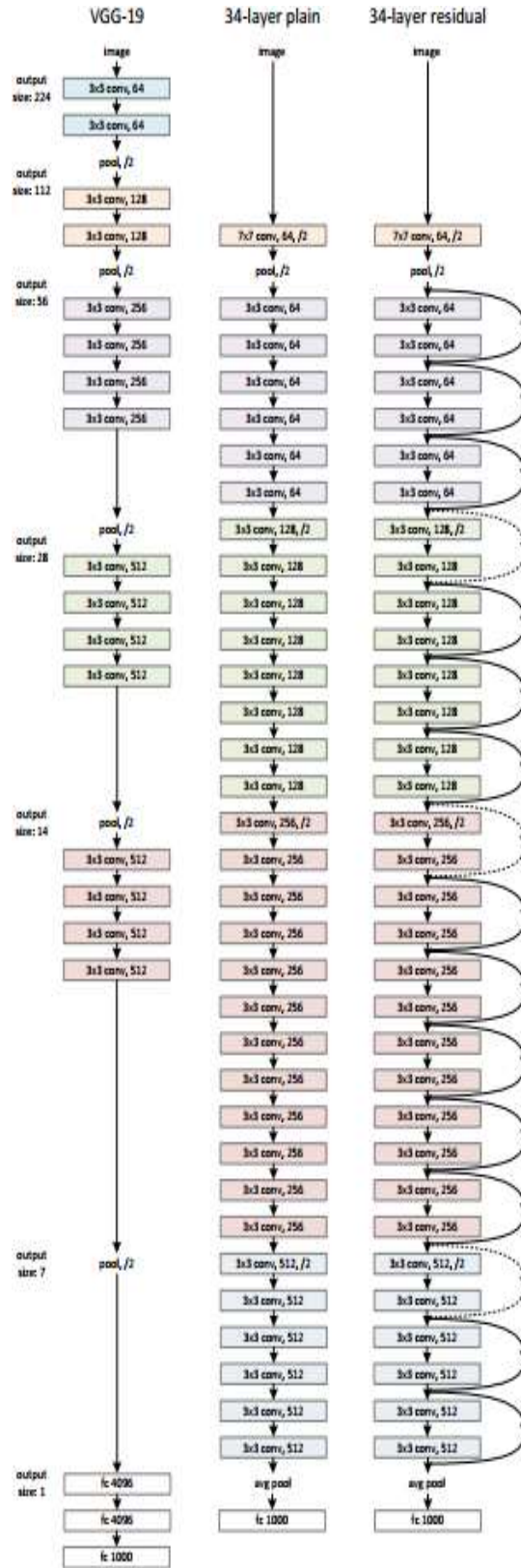


Figure 1.6: Architecture of resnet

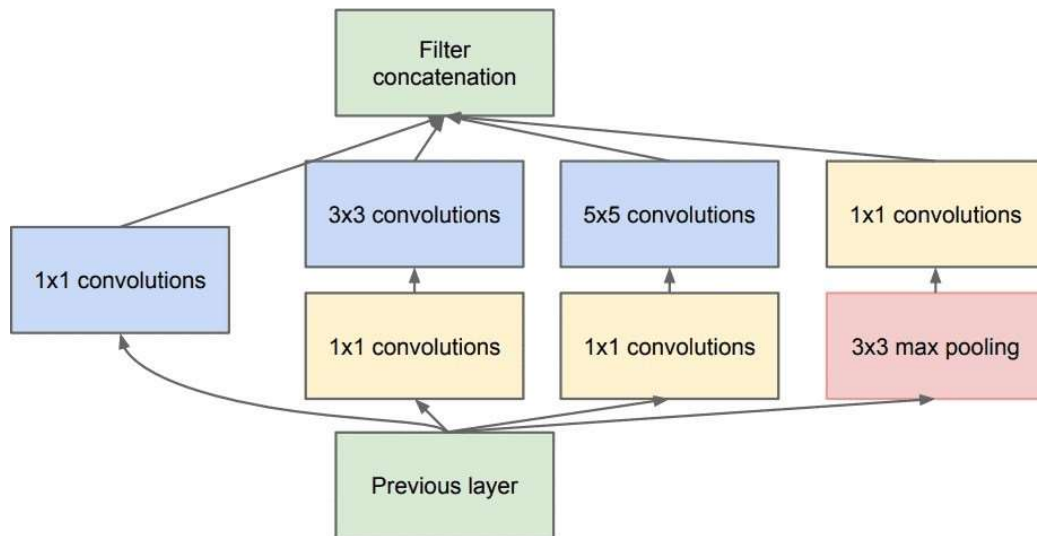


Figure 1.7: Googlenet CNN inception module

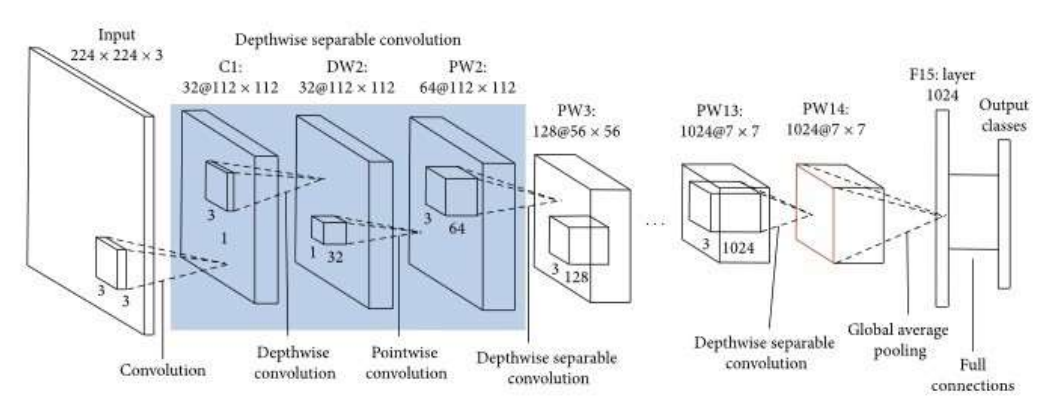


Figure 1.8: Mobilenet CNN architecture

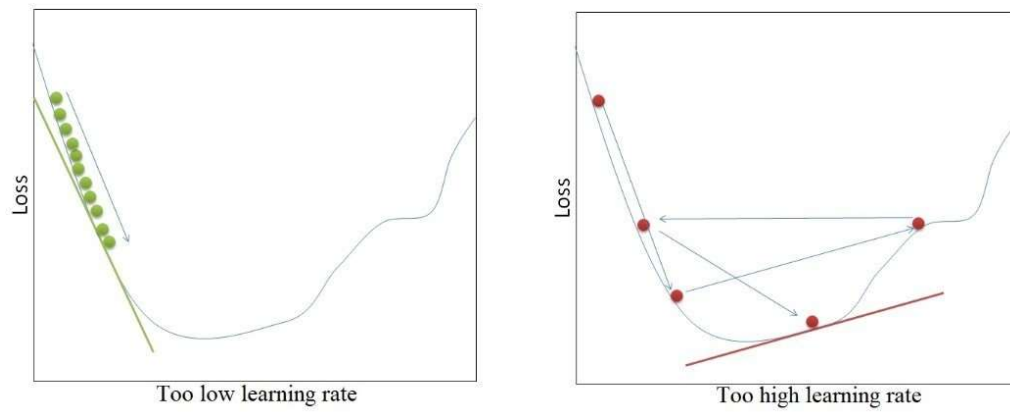


Figure 1.9: Learning rate graph

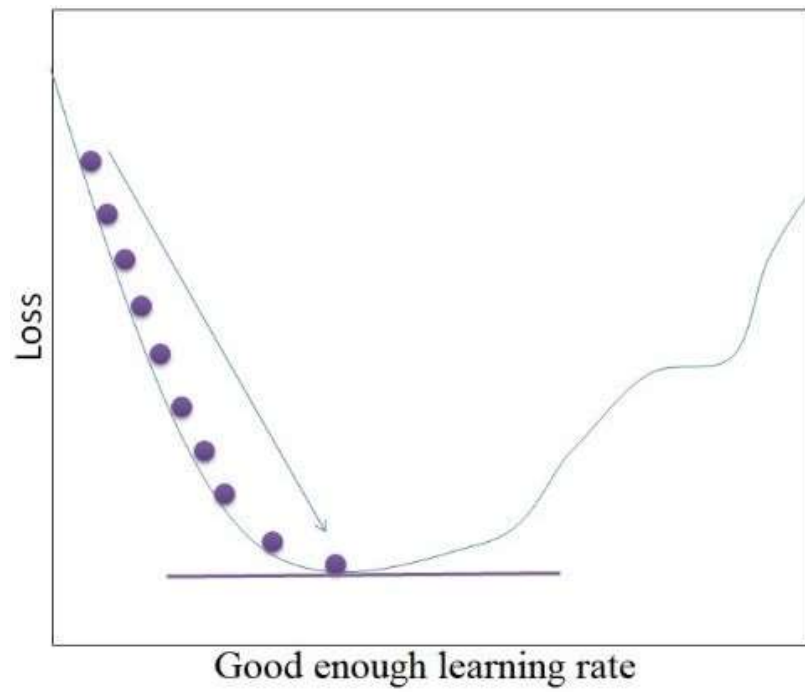


Figure 1.10: Good enough graph for learning rate

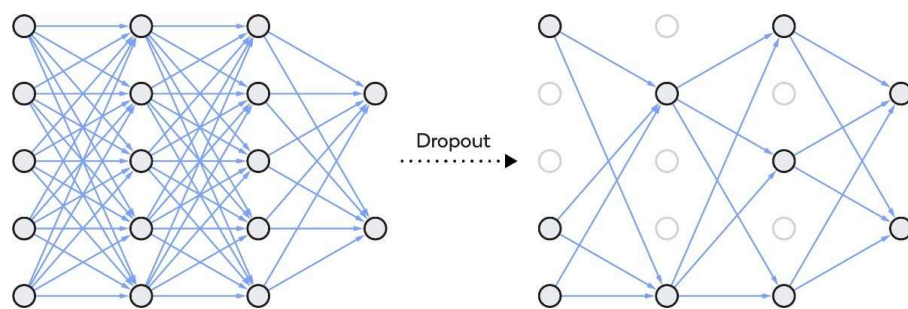


Figure 1.11: DropOut Parameter

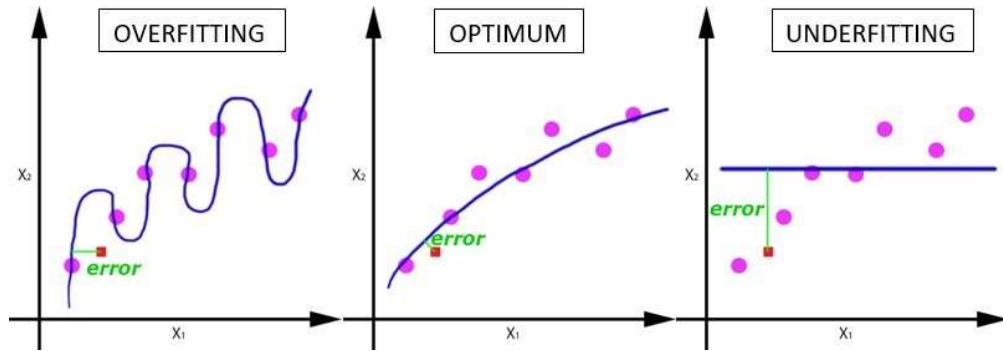


Figure 1.12: Number of epoch's Parameter

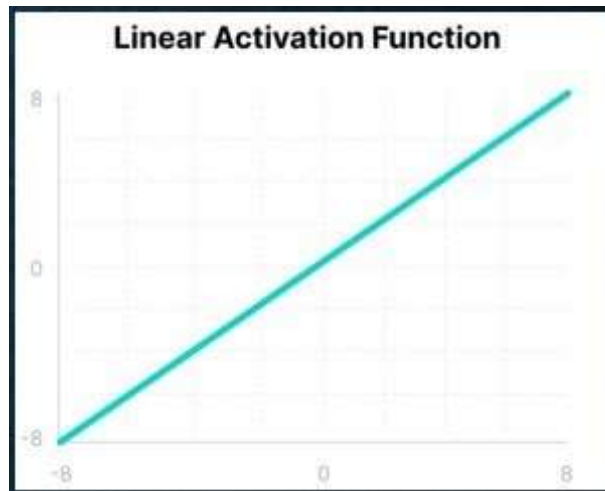


Figure 1.13: Linear Activation function

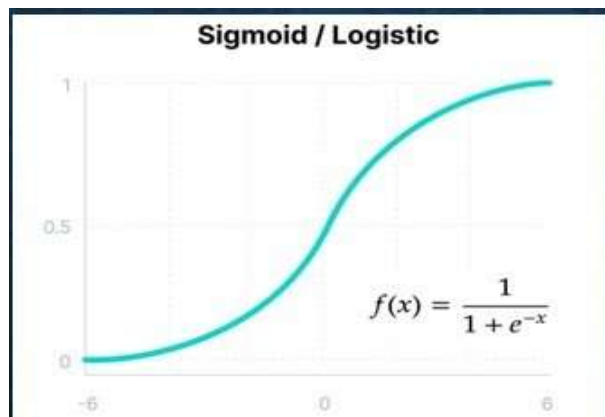


Figure 1.14: Sigmoid Activation function

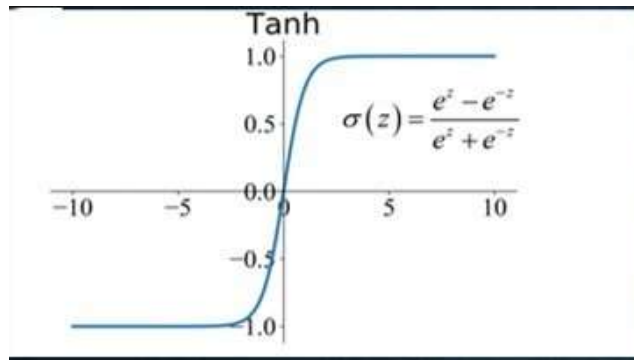


Figure 1.15: Tanh Activation function

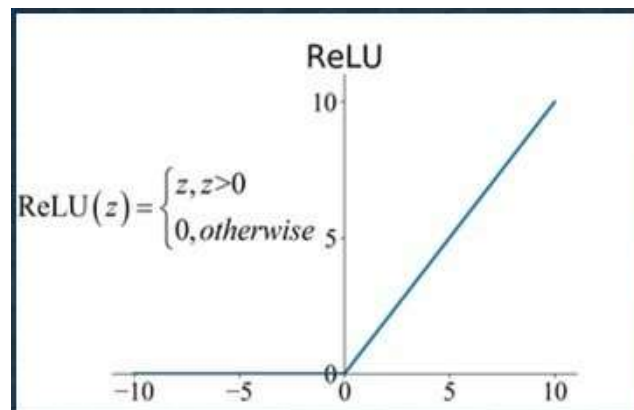


Figure 1.16: Relu Activation function

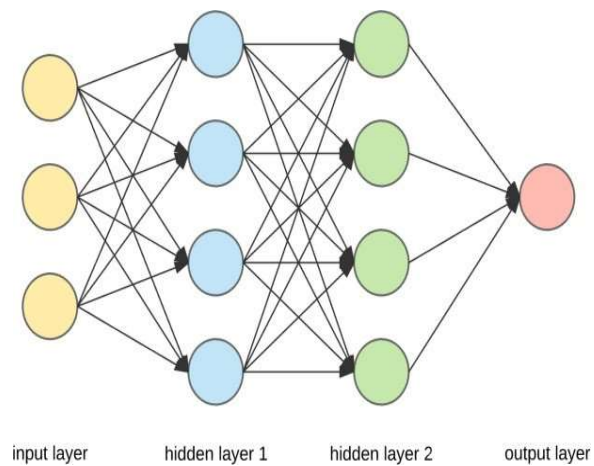


Figure 1.17: Number of Hidden layers and units Parameter

CHAPTER 2

LITERATURE SURVEY

M. Azad , F. Khaled and M.I. Pavel [9] have represent that a novel approach to transform one dimension (1- D) signals into two dimension (2-D) grayscale image and a feature extraction process to extricate detail texture data of this 2D image to classify signals utilizing multi-class support vector machine. In all previous approaches of the signal processing strategies, the signal is continuously processed in one dimension (1-D) representation.

Dan Li; Jianxin Zhang; Qiang Zhang; Xiaopeng Wei [10] gave us a deep understanding in this innova- tion,They introduced a conspicuous expanding number of cardiovascular illnesses, and the programmed grouping exploration of Electrocardiogram signals (ECG) has been having an essentially significant influence in the clinical finding of cardiovascular infection. In this paper,they have examined a 1D con- volution brain organization (CNN)[3] based technique is proposed to characterize ECG signals. The proposed CNN Model comprises of five layers notwithstanding the information layer and the result layer, i.e., two convolution layers, two down testing layers and one full association layer, extricatingthe viable elements from the first information and arranging the highlights naturally.

Shenghui Yang,Yu Tan[11] said that the synchroextracting change is a strong postprocessing strategy that hones the energy appropriation of the spectrogram. In any case, its limit is the unit utilizationof time or recurrence factors that can't precisely depict signals containing both 'slow-differing' and 'quick evolving' parts. The proposed technique applies the synchroextracting methodology in both time and recurrence aspects to further develop the energy focus.

M. Hadi Amini [12] researched about capacity to precisely anticipates understudies' prosperity settles on it as an ideal device for choice-making errands connected with advanced education. The precision of AI models relies upon choosing legitimate hyper-parameters. In any case, it's anything but a simple undertaking since it demands investment and skill to tune the hyper-tuning to fit the AI model. In this paper, he analyze the adequacy of mechanized hyper-parameter tuning methods to the domainof understudies'

prosperity.

Raja Majid Mehmood [13] done research on the electrocardiogram (ECG) is quite possibly the most broadly utilized signal utilized in the conclusion and expectation of cardiovascular illnesses (CVDs).The ECG signs can catch the heart's cadenced anomalies, regularly known as arrhythmias. A cautious investigation of ECG signals is pivotal for exact judgments of patients' intense and ongoing heart conditions.

Then again Meria M George et al, [14] focused on ceaselessly checking utilizing unique sensors which is associated with the Arduino board. Also, the obtained information is ship off the server utilizing Ethernet safeguard joined to the Arduino board. They utilized An ECG Sensor,Accelerometer and a Temperature sensor.

Paola Pierleoni et al [15]recommended a clever framework. This work proposes a technique dependent on the Backing Vector Machine procedure and it is addressed to minimal expense smartphones.This technique begins from the information that is procured from accelerometer and magnetometer, which is presently accessible taking all things together the low-end gadgets,. After an underlying preparing, the order of fall occasions and non-fall occasions is performed by the Support Vector Machine calculation.

CHAPTER 3

METHODOLOGY

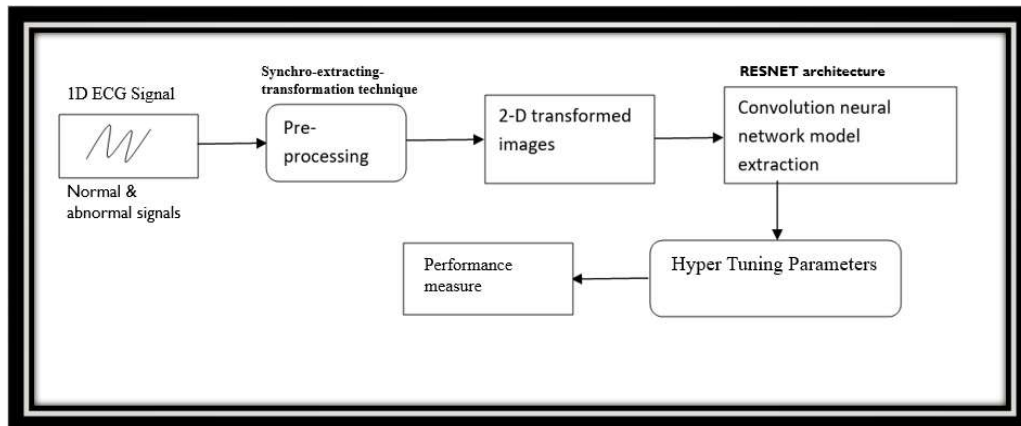


Figure 3.1: Block diagram of system

3.1 Working

First we have to collect ECG signals of normal data and abnormal data, we have used Kaggle platform to collect ECG data set :: <https://www.kaggle.com/code/gregoiredc/arrhythmia-on-ecg-classification-using-CNN/data> Now we have to convert dataset .csv files to excel format and dump the data as input to the synchro-extracting-transform model. Collect the 2D images that are produced by the method and store it as a jpeg format images. Software used for this method is MATLAB 2021b and images were stored in separate folders for both normal and abnormal. Now this 2D images were sent to Convolution neural network model, the type of CNN Model we have taken was keras sequential model because to create models layer-by-layer. Functional API is an alternative approach of creating more complex models. Functional model, you can define multiple input or output that share layers. The output of these CNN Model were passed through different hyper tuning parameters and thus we can measure accuracy of those parameters and pick the best performance parameter. Hyper parameters we have used were learning rate, activation function, number of epochs to calculate the accuracy.

3.2 Transformation of 2D images

Firstly transformation of 1d ECG signals to 2D images was done using a technique synchro extracting transform it was done using an online software tool Matlab which helps to convert signals to images so in this way we have took totally 60 images for both normal and abnormal data-sets at last we have saved all the images in the JPEG format and now the images were ready and they can be tested by using CNN Model. images of normal and abnormal are shown in fig3.2 fig3.3.

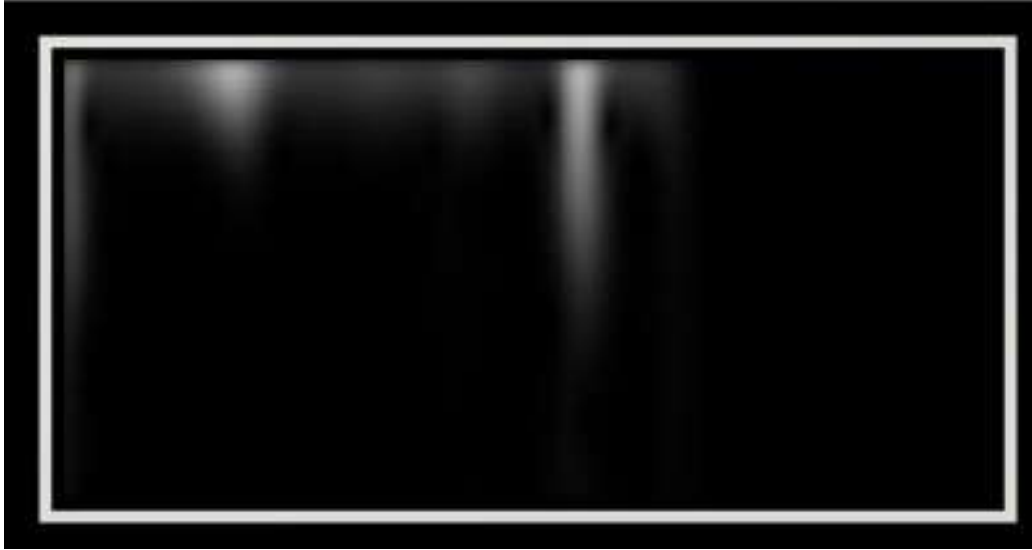


Figure 3.2: Image OF Normal signal

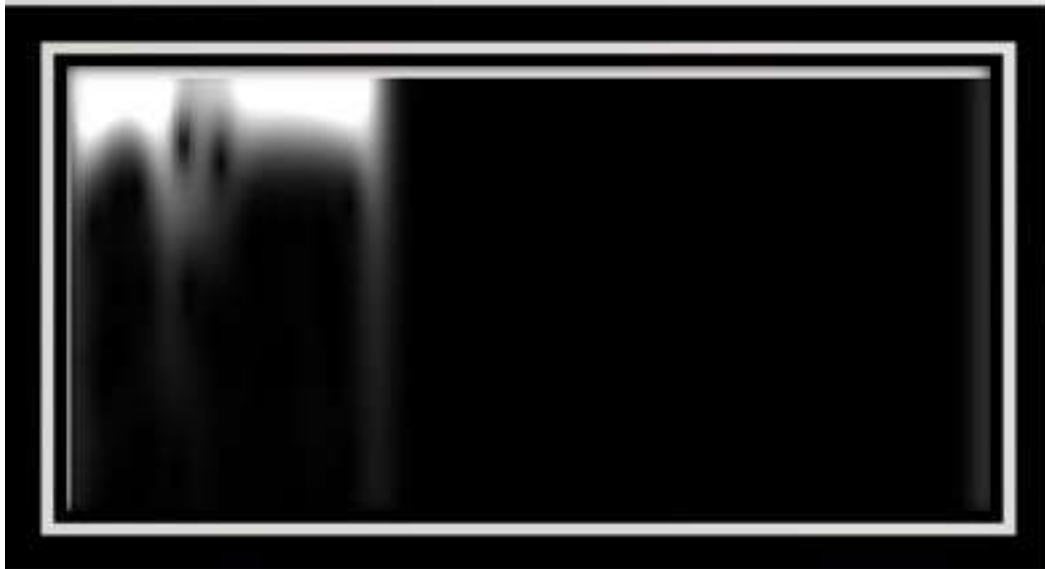


Figure 3.3: Image OF Abnormal signal

3.3 CNN Model

Now at second stage 2D images are fed into our CNN Model here we use sequential keras model as our CNN Model here we had given 3 layers for our CNN Model and passed different activation functions like relu, tanh, sigmoid and also while compiling we have given different hyper tuning parameters to measure accuracy of model. fig 4.4 shows the flow diagram of keras sequential model which will have 3 layers and also max pooling layer followed by flatten layer and dense layer.

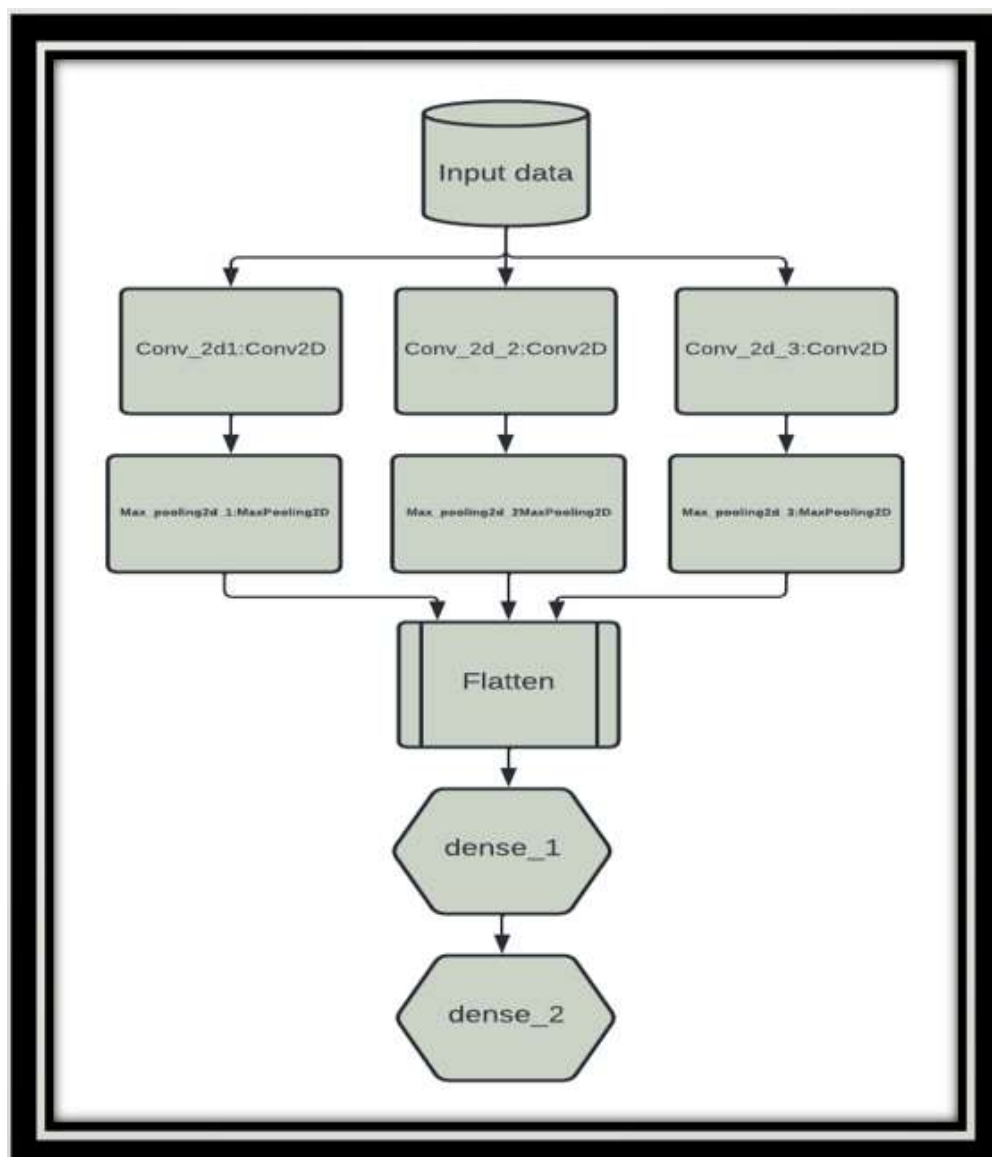


Figure 3.4: CNN Model flow diagram

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 Matlab

MATLAB is an elite presentation language for specialized processing. It incorporates calculation, perception, and programming in a simple to-utilize climate where issues and arrangements are communicated in recognizable numerical documentation. Matrix laboratory or MATLAB for short is a multi-worldview mathematical processing environment and exclusive programming language created by Mathworks. It consolidates calculation, perception, and programming in an effectively usable climate and is totally communicated in numerical conditions. Written in C, C++, and Java, Matlab was at first delivered in 1984. The most recent adaptation has been delivered on March 2018. Applications of Matlab is built around the Matlab scripting language and revolves based on the following mathematical concepts:

- Variables
- Vectors and matrices
- Structures
- Functions
- Function handles
- Classes and object-oriented programming Typical uses include:
- Math and computation
- Algorithm development
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

MATLAB is an interactive framework whose fundamental information component is a cluster that doesn't need dimensioning. This permits you to tackle numerous specialized figuring issues, particularly those with matrix and vector details, in a small portion of the time it would take to compose a program in a scalar noninteractive language like C or Fortran.

The name MATLAB represents a network research facility. MATLAB was initially composed to give simple admittance to network programming created by the LINPACK and EISPACK projects, which together address the cutting edge in programming for grid calculation. MATLAB has advanced over a time of years with input from numerous clients. In college conditions, it is the standard educational instrument for starting and progressing courses in arithmetic, designing, and science. In industry, MATLAB is the apparatus of decision for high-efficiency exploration, advancement, and examination.

MATLAB highlights a group of utilization explicit arrangements called tool stash. Vital to most clients of MATLAB, tool kits permit you to learn and apply particular innovations. Tool compartments are complete assortments of MATLAB capacities (M-records) that stretch out the MATLAB climate to take care of specific classes of issues. Regions in which tool compartments are accessible incorporate sign handling, control frameworks, brain organizations, fluffy rationale, wavelets, recreation, and numerous others.

The MATLAB system consists of five main parts:

4.2 The MATLAB language

This is an undeniable level grid/cluster language with control stream explanations, capacities, information structures, input/result, and item situated programming highlights. It permits both "programming in the little" to quickly make down to business discard projects, and "programming in the huge" to make total enormous and complex application programs. The MATLAB working environment

This is the arrangement of apparatuses and offices that you work with as the MATLAB client or software engineer. It incorporates offices for dealing with the factors in your work area and bringing in and sending out information. It additionally incorporates apparatuses for creating, making due, investigating, and profiling M-documents, MATLAB's applications.

4.3 Handle Graphics

This is the MATLAB illustrations framework. It incorporates significant level orders for two-layered and three-layered information representation, picture handling, activity, and show illustrations. It additionally incorporates low-level orders that permit you to completely tweak the presence of designs as well as to construct total Graphical User Interfaces on your MATLAB applications.

4.4 The MATLAB mathematical function library

This is a tremendous assortment of computational calculations going from rudimentary capacities like total, sine, cosine, and complex math, to additional refined capacities like network reverse, framework eigenvalues, Bessel capacities, and quick Fourier changes.

4.5 The MATLAB Application Program Interface (API)

This is a library that permits you to compose C and Fortran programs that communicate with MATLAB. It incorporates offices for calling schedules from MATLAB (dynamic connecting), calling MATLAB as a computational motor, and for perusing and composing MATLAB-documents.

4.6 Origins

MATLAB was concocted by mathematician and software engineer Cleve Moler. The thought for MATLAB depended on his 1960s PhD theory. Moler turned into a numerical teacher at the University of New Mexico and begun creating MATLAB for his understudies as a side interest. He fostered MATLAB's underlying direct polynomial math programming in 1967 with his one-time proposition counselor, George Forsythe. This was trailed by Fortran code for direct conditions in 1971.

At the outset (before form 1.0) MATLAB "was not a programming language; it was a basic intelligent grid mini-computer. There were no projects, no tool kits, no designs. Furthermore, no ODEs or FFTs." The main early form of MATLAB was finished in the last part of the 1970s. The product was revealed to general society without precedent for February 1979 at the Naval Postgraduate School in California. Early renditions of MATLAB were basic grid adding machines with 71 pre-fabricated capacities. At that point, MATLAB was disseminated for nothing to colleges. Moler would leave

duplicates at colleges he visited and the product fostered a solid continuing in the mathematical branches of college grounds.

4.7 Control systems

One more normal motivation behind why the utilization of Matlab is significant is that it gives control to gadgets and frameworks. A control framework is answerable for making due, providing orders, and managing the way of behaving of different gadgets or frameworks. It depends on control circles. The gadgets or frameworks being controlled can go from basic home warmers to huge modern control frameworks that control the cycles or the machines. The control framework tool stash of Matlab gives calculations and applications to deliberately dissecting, planning, and tuning direct control frameworks.

4.8 Digital signal processing

Digital signal processing is the usage of digital processing such as by computers or specialized digital signal processors to perform a range of signal processing operations. Uses of Matlab products make it easy to use signal processing techniques to analyze time series data and also provides a unified workflow for the development of embedded systems and streaming applications.

4.9 Wireless communications

Wireless communication is a broad term used for connecting two devices using a wireless signal. Teams in wireless Engineering use Matlab too reduce development time, eliminate design problems early, and streamline testing and verification.

4.10 Image processing and computer vision

Image processing is mainly focused on the processing of raw images and making them ready for other tasks such as computer vision. Computer vision, on the other hand, looks at images like the human eye. It involves understanding and predicting the visual output. Algorithm building is vital for image processing and computer vision. Uses of Matlab provides a comprehensive environment to analyze images and build algorithms.

4.11 Internet of Things

Internet of Things is the network of devices, vehicles and home appliances

and others embedded with electronics, software, sensors, actuators, and connectivity which enables the exchange of data. uses of Matlab help in the design, prototype, and deployment of IOT applications such as predictive maintenance, operations optimization, supervisory control etc.

4.12 FPGA Design and Codesign

Matlab enables hardware-software codesign by providing C/C++ and HDL code generation with targeted support for programmable SoC devices.

4.13 Jupyter Notebook

Jupyter Notebook is an open-source web application that allows a user, scientific researcher, scholar or analyst to create and share the document called the Notebook, containing live codes, documentation, graphs, plots, and visualizations. Jupyter Notebook provides support for 40+ programming languages to the users including the most frequently used programming languages – Python, R, Julia to name a few. It allows the user to download the notebook in various file formats like PDF, HTML, Python, Markdown or an .ipynb file. Due to Jupyter Notebook's multi-programming support, huge feature availability and rapidly growing popularity among the community, it has become a standard for all sorts of analysis, visualizations, rapid prototyping, ML and various code practices.

A Jupyter Notebook is an open source web application that allows data scientists to create and share documents that include live code, equations, and other multimedia resources. Any individual who is an information researcher, information engineer, information expert, AI researcher, research researcher, logical specialists or an overall client who needs to do any kind of logical calculation, information handling or representation related work can utilize the Jupyter Notebook. We, at Elucidata, are chipping away at this task to foster new highlights and administrations on top of a customary Jupyter Notebook, to work with our end clients to have the best client experience. We have dealt with making a Jupyter Notebook with a pristine and rich UI, and new custom functionalities. We are not leaving any stone unturned to make it the best journal experience a client can have. We are committed to this task to make it, what an information researcher, information architect or information researcher would need on our foundation.

We presented the Jupyter Notebooks in the eco-arrangement of our foundation - Polly™, to help the control, perception, and the programming of the outcome of the implicit work processes. Afterward, we utilized the usefulness of the Jupyter Notebook and joined it with the JupyterHub design to expand the functionalities for the accompanying use cases:

- **PROJECT:** In Polly™, a user can go to his/her project and from there, they can open their saved Jupyter Notebooks, or create new Jupyter Notebooks, or even upload an existing Jupyter Notebook. Projects are the most widely used feature for opening, creating, modifying or even deleting the Jupyter Notebook in Polly™
- **TEMPLATES:** We have assembled a nonexclusive format note pads also to help all the utilization cases
- **ANALYSIS:** The analysis is a central issue in tracking down important bits of knowledge from the gigantic genomics and metabolomics datasets transferred to our foundation for running different in-house constructs and work processes. By coordinating the Jupyter Notebook we offer our end clients and our in-house information researchers an advantageous connection point for intuitively running code, investigating yield, and imagining information - all from a solitary cloud-based advancement climate. Alongside this, we have added the custom usefulness of our foundation API which turns out consistently for bringing datasets from our cloud-oversaw climate straightforwardly into the Notebook.
- **Our in-house build workflows:** As we extended the stage, we presented new abilities called work processes. With the hard endeavors of our designing group and the coordinated effort of the information science group, we prevailed with regard to building work processes which are a progression of calculations running in a set way to accomplish a particular objective. Information Scientists can code the analysis utilizing our facilitated Jupyter Notebook. Programmers can code different functionalities utilizing the Jupyter Notebook.
- **COMPUTE:** The user's virtual machine instances support the computation of a core, with 2 GB RAM and 100 GB block storage, but as per availability and cluster usage the computation power would increase. Our cluster is AutoScale

enabled which allows spawning of the new user pods on the fly based on high requests. We have deployed our whole notebook infrastructure on Google Cloud.

- **CLUSTER MANAGEMENT SOFTWARE:** We are involving Kubernetes for dealing with our calculation examples and bunch. Kubernetes guarantees that the cases in running state don't close down because of a blunder, keeping up with high accessibility. With Kubernetes, we can oversee 1000+ client cases without losing any information.
- **DEPLOYMENT:** We use Helm (a bundle administrator) for Kubernetes to mechanize our arrangement interaction. Rudder guarantees the right docker picture is sent and saved for later use to try not to pull the picture once more and lessen bringing forth time.
- **STORAGE:** We are using Amazon S3 as a storage system for the users' Notebook and their reusable scripts across the Jupyter Notebook. Thus each user project has a directory structure at S3 for storing, managing, creating or deleting their notebook. They can launch its interactive notebook from within our platform. Following is the snapshot of the users' project storage on S3.

- **PYTHON**

Python is much of the time utilized as a help language for programming engineers, for fabricating control and the executives, for testing, and in numerous alternate ways. SCons for fabricates control. Buildbot and Apache Gump for computerized constant assemblage and testing. Roundup or TrackK-ing for bug following and task the board. Python is a deciphered, object-arranged, significant-level programming language with dynamic semantics.

- **NUMPY:** One of the most principal bundles in Python, NumPy is a universally useful cluster handling bundle. It gives elite execution complex cluster items and instruments to work with the exhibits. NumPy is a productive holder of nonexclusive multi-faceted information. NumPy's principal object is the homogeneous multi-faceted cluster. It is a table of components or quantities of the equivalent data type, ordered by a tuple of positive numbers. In NumPy, aspects are called tomahawks and the quantity of tomahawks is called a rank. NumPy's exhibitclass is called array otherwise known as a cluster. When to utilize? NumPy is utilized to deal with exhibits that store upsides of the

equivalent data. NumPy works with math procedures on exhibits and their vectorization. This fundamentally improves execution and velocities up the execution time correspondingly.

- **SCIPY:** The SciPy library is one of the center bundles that make up the SciPy stack. Presently, there is a contrast between SciPy Stack and SciPy, the library. SciPy expands on the NumPy cluster object and is essential for the stack which incorporates instruments like Matplotlib, Pandas, and SymPy with extra apparatuses, SciPy library contains modules for productive numerical schedules such as straight variable-based math, insertion, improvement, incorporation, and insights. The primary use of the SciPy library is based upon NumPy and its exhibits. SciPy utilizes NumPy. When to utilize? SciPy involves clusters as its essential information structure. It has different modules to perform normal logical programming undertakings such as straight polynomial math, coordination, analytics, customary differential conditions, and sign handling.
- **MATPLOTLIB:** This is without a doubt my number one and quintessential Python library. You can make stories with the information pictured with Matplotlib. One more library from the SciPy Stack, Matplotlib plots 2D figures. When to utilize? Matplotlib is the plotting library for Python that gives an article arranged API to implant plots into applications. It is a nearby similarity to MATLAB inserted in the Python programming language.
- **SEABORN:** So when you read the authority documentation on Seaborn, it is characterized as the information representation library in light of Matplotlib that gives a significant level point of interaction to drawing alluring and instructive factual illustrations. Laying it out plainly, seaborn is an augmentation of Matplotlib with cutting-edge highlights. All in all, what is the contrast between Matplotlib and Seaborn? Matplotlib is utilized for essential plotting; bars, pies, lines, disperse plots, and stuff though, seaborn furnishes an assortment of perception designs with less intricate and less language structure.
- **TENSORFLOW:** Back in 2017, I got a TensorFlow USB as an appreciation for being an astounding speaker at a Google WTM occasion, haha. The USB was stacked with true documentation of TensorFlow. With no hint by then of what TensorFlow was, I Googled it. Tensor-Flow is an AI library that assists engineers with making huge scope brain networks with many layers utilizing

information stream diagrams. TensorFlow likewise works with the structure of Deep Learning models, pushing the cutting edge in ML/AI and permitting simple sending of ML-controlled applications.

- **KERAS:** Keras is TensorFlow's significant level API for building and preparing Deep Neural Network code. It is an open-source brain network library in Python. With Keras, measurable demonstrating, working with pictures and text is much simpler with improved on coding for profound learning. What is the contrast between Keras and TensorFlow all things considered? Keras is a brain network Python library while TensorFlow is an open-source library for different AI assignments. TensorFlow gives both undeniable level and low-level APIs while Keras gives just significant level APIs. Keras is worked for Python which makes it way more easy to use, particular and composable than TensorFlow

CHAPTER 5

RESULT AND ANALYSIS

By using different hyper-tuning parameters to our CNN mode we got different-different accuracy levels and also image classification was done where model can predict whether it is a normal image or abnormal image.

The two huge advancement boundaries in the proposed 2-D CNN Model are the learning rate also, the activation function of the information utilized. To work on the presentation, these two advancement boundaries should be chosen cautiously to acquire the best exactness in the programmed grouping of arrhythmia utilizing the ECG signals. The proposed model was assessed in various trials with different upsides of learning boundaries. For a more modest worth of the learning rate (i.e., under 0.0005), the speed of the assembly was exceptionally sluggish. Nonetheless, when the worth of the learning rate was enormous (i.e., more noteworthy than 0.001), the speed of intermingling moved along. Simultaneously, awry changes were noticed in the precision rate. Hence, we chose an ideal worth of 0.001 for the learning rate, as this worth can achieve better precision for the proposed model

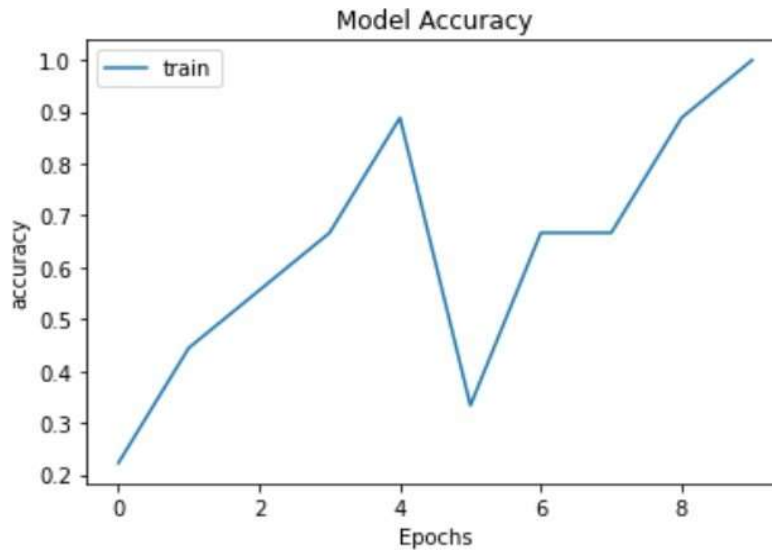


Fig 5.1: Accuracy graph of Relu activation function with learning rate 0.001

Figure 5.1 shows the accuracy of CNN model whose activation function is “RELU” and learning rates “0.001” and number of epochs is “10”.

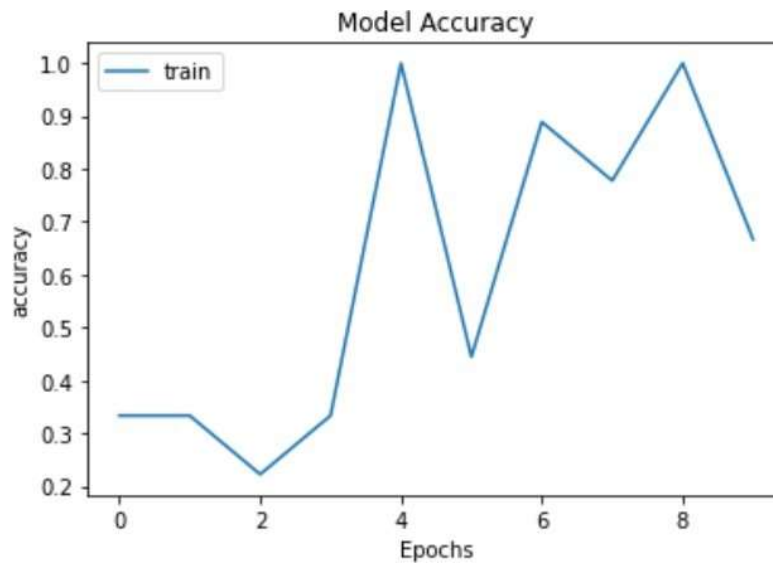


Fig 5.2: Accuracy graph of Relu activation function with learning rate 0.01

Figure 5.2 shows the accuracy of CNN model whose activation function is “RELU” and learning rates “0.01” and number of epochs is “10”.

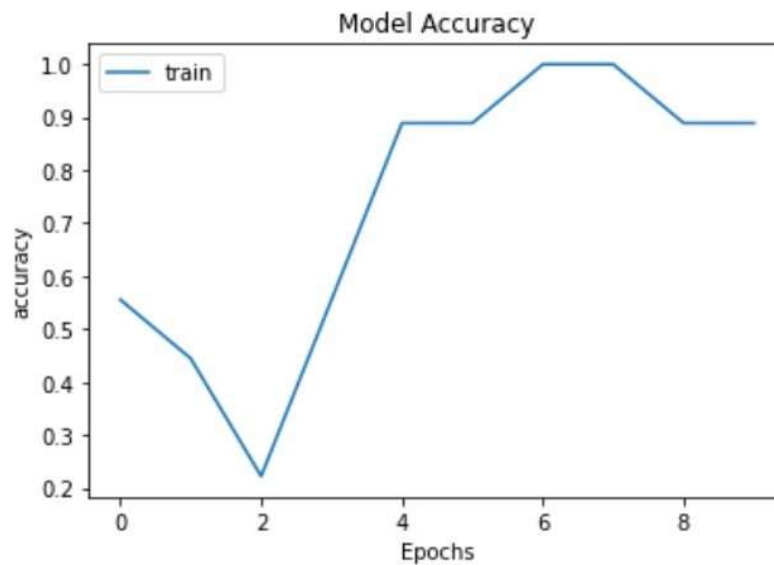


Fig 5.3: Accuracy graph of Relu activation function with learning rate 0.1

Figure 5.3 shows the accuracy of CNN model whose activation function is “RELU” and learning rate is “0.1” and number of epochs is “10”.

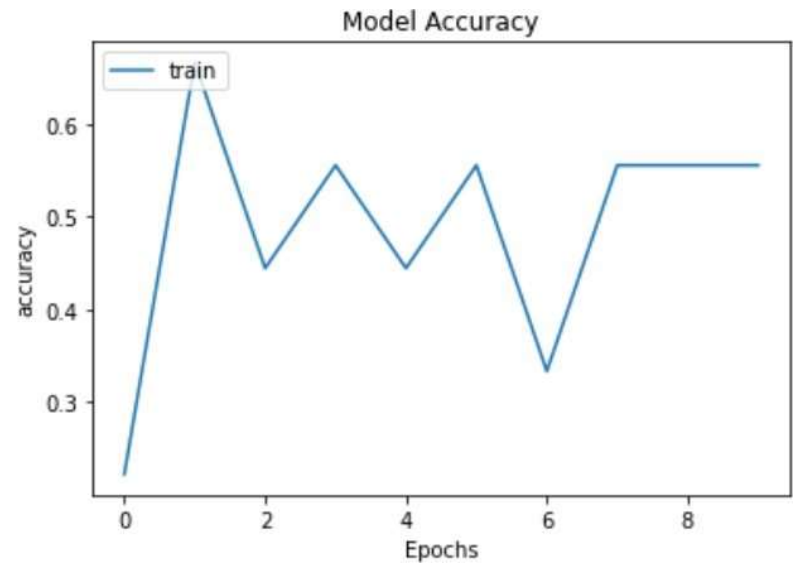


Fig 5.4: Accuracy graph of Tanh activation function with learning rate 0.1

Figure 5.4 shows the accuracy of CNN model whose activation function is “Tanh” and learning rate is “0.001” and number of epochs is “10”.

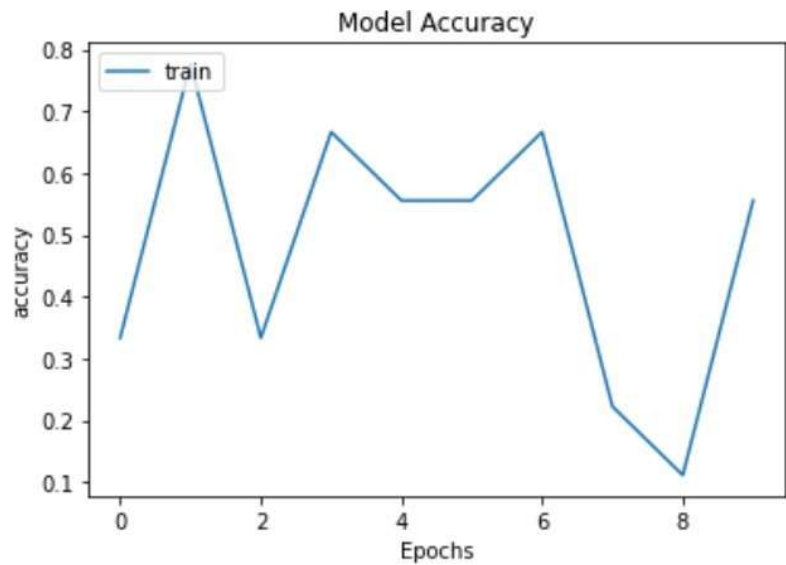


Fig 5.5: Accuracy graph of Tanh activation function with learning rate 0.0

Figure 5.5 shows the accuracy of CNN model whose activation function is “Tanh” and learning rate is “0.01” and number of epochs is “10”.

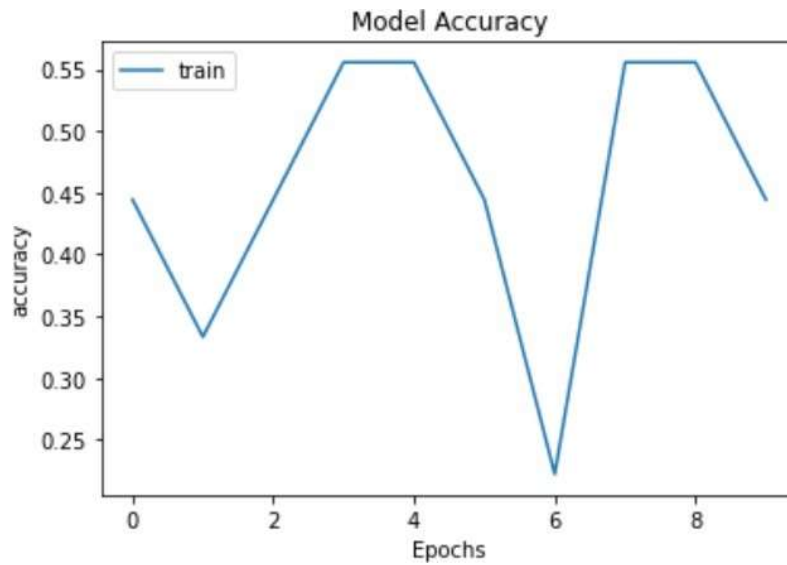


Fig 5.6: Accuracy graph of Tanh activation function with learning rate 0.001

Figure 5.6 shows the accuracy of CNN model whose activation function is “Tanh” and learning rate is “0.1” and number of epochs is “10”.

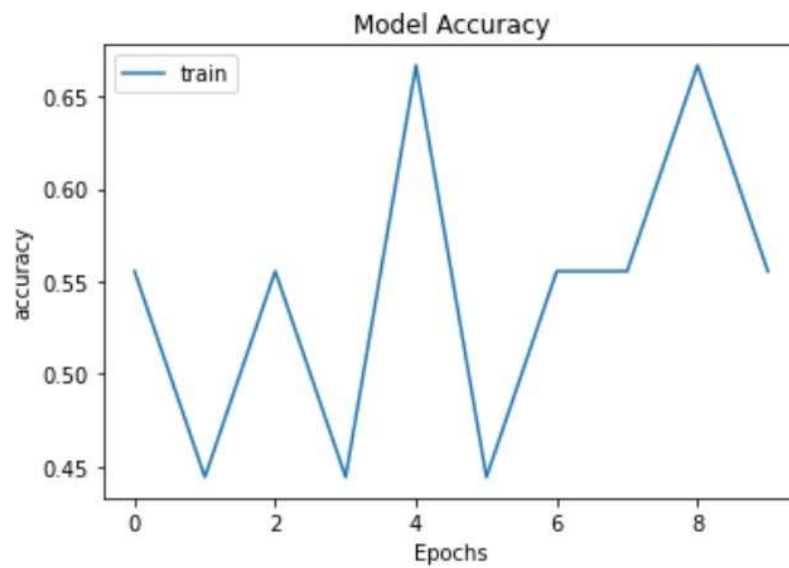


Figure 5.7: Accuracy graph of Sigmoid activation function with learning rate 0.001

Figure 5.7 shows the accuracy of CNN model whose activation function is “Sigmoid” and learningrateis “0.001” and number of epochs is “10”.

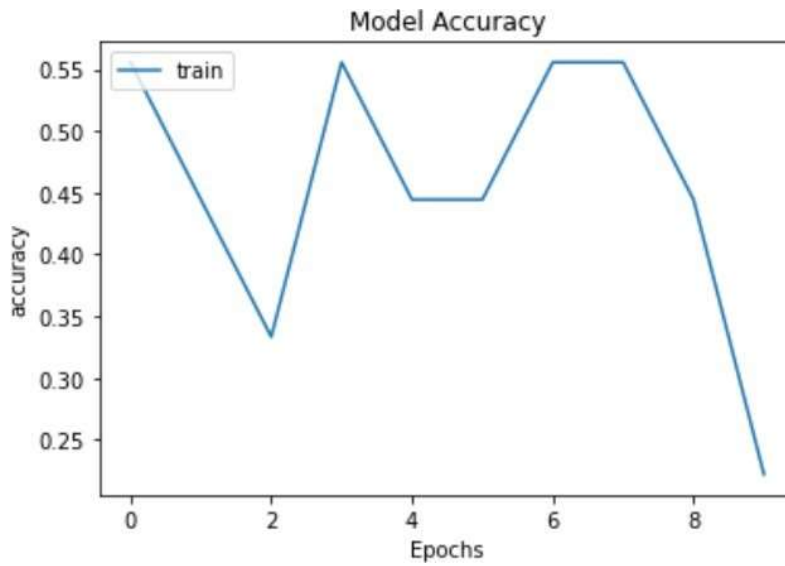


Figure 5.8: Accuracy graph of Sigmoid activation function with learning rate 0.01

Figure 5.8 shows the accuracy of CNN model whose activation function is “Sigmoid” and learningrateis “0.01” and number of epochs is “10”.

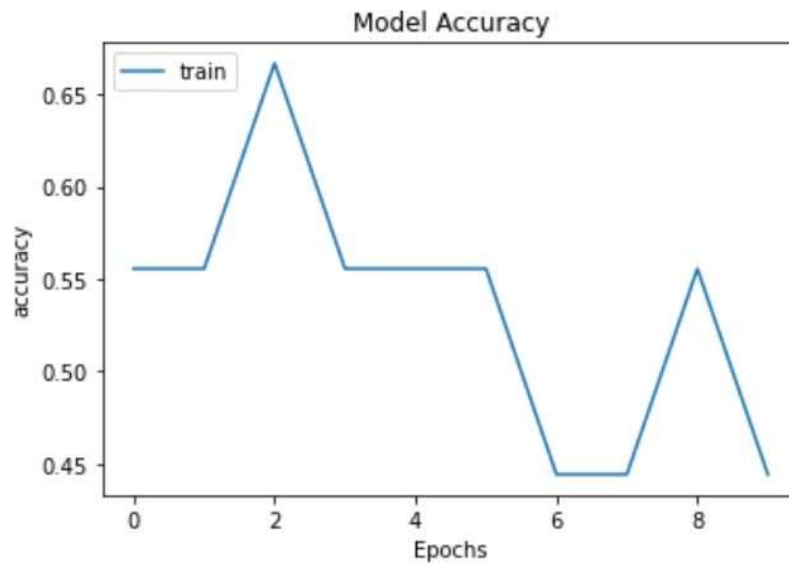


Figure 5.9: Accuracy graph of Sigmoid activation function with learning rate 0.1

125 Figure 5.9 shows the accuracy of CNN model whose activation function is “Sigmoid” and learningrateis “0.1” and number of epochs is “10’.

Activation function	Learning rate	Accurac y
Relu	0.001	100
Relu	0.01	92
Relu	0.1	64
Sigmoid	0.001	87
Sigmoid	0.01	53
Sigmoid	0.1	50
Tanh	0.001	68
Tanh	0.01	43
Tanh	0.1	38

Table
5.1: Accuracy
table

In this we have used three different Activation function that is Relu,Sigmoid,and Tanh. And also we have take three different Learning rate (0.001,0.01,0.1). With this three different activation function we got perfect accuracy with Relu, it got highest accuracy compare to other activation functions which we have used. so Relu got highest accuracy with learning rate 0.001 .

CHAPTER 6

CONCLUSION

Conversion of 1D ECG signals into transformed images were done successfully and also compilation of CNN Model using different activation functions and hypertuning parameters were done successfully. Here in the proposed report, conversion of ECG signal to the transformed images takes place for that, the proposed technique introduced was a new time-frequency analysis (TFA) method to study the trend and instantaneous frequency (IF) of nonlinear and nonstationary data. The proposed method is termed as synchro extracting transform (SET).

By this project we can conclude that different activation function has different results in accuracy and also we can observe that activation function with relu and learning rate 0.001 got a highest accuracy that is 100 percentage and also the least accuracy was 38 percentage which has the activation function as Tanh with learning rate 0.1 so after measuring the accuracy of different parameters by this table we came to know that activation function relu with learning rate 0.001 has a highest accuracy. Also classification of images that means whether it is a normal or abnormal image that was also classified successfully.

REFERENCES

1. Azad, M., F. Khaled, and M. Pavel. "A novel approach to classify and convert 1D signal to 2D grayscale image implementing support vector machine and empirical mode decomposition algorithm." *Int J Adv Res* 7 (2019): 328-335.
2. Li, Dan, et al. "Classification of ECG signals based on 1D convolution neural network." 2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom). IEEE, 2017.
3. Hammad, Mohamed, Shanzhuo Zhang, and Kuanquan Wang. "A novel two-dimensional ECG feature extraction and classification algorithm based on convolution neural network for human authentication." *Future Generation Computer Systems* 101 (2019): 180-196.
4. Narayan, Sridhar. "The generalized sigmoid activation function: Competitive supervised learning." *Information sciences* 99.1-2 (1997): 69-82.
5. Dubey, Arun Kumar, and Vanita Jain. "Comparative study of convolution neural network's relu and leaky-relu activation functions." *Applications of Computing, Automation and Wireless Systems in Electrical Engineering*. Springer, Singapore, 2019. 873-880.
6. Zahedi L, Mohammadi FG, Rezapour S, Ohland MW, Amini MH. Search algorithms for automated hyper-parameter tuning. *arXiv preprint arXiv:2104.14677*. 2021 Apr 29.
7. Zhang R, Liu X, Zheng Y, Lv H, Li B, Yang S, Tan Y. Time-frequency synchroextracting transform. *IET Signal Processing*. 2021 Nov 9.
8. Kiranyaz, Serkan, Turker Ince, and Moncef Gabbouj. "Real-time patient-specific ECG classification by 1-D convolutional neural networks." *IEEE Transactions on Biomedical Engineering* 63.3 (2018): 664-675.
9. Clifford, Gari D., and Francisco Azuaje. *Advanced methods and tools for ECG data analysis*. Ed. Patrick McSharry. Vol. 10. Boston: Artech house, 2006.
10. Raja Majid Mehmood. "Automated detection of diabetes using CNN and

- CNN-LSTM network and heart rate signals.” *Procedia computer science* 132 (2018): 1253-1262.
11. Da Silva, H. P., Lourenço, A., Fred, A., Raposo, N., Aires-de-Sousa, M. (2014). Check Your Biosignals Here: A new dataset for off-the-person ECG biometrics. *Computer methods and programs in biomedicine*, 113(2), 503-514.
 12. Mc Namara, K.; Alzubaidi, H.; Jackson, J.K. Cardiovascular disease as a leading cause of death: How are pharmacists getting involved? *Integr. Pharm. Res. Pract.* 2019, 8, 1. [CrossRef] [PubMed]
 13. Irmakci, I.; Anwar, S.M.; Torigian, D.A.; Bagci, U. Deep Learning for Musculoskeletal Image Analysis. *arXiv* 2020, arXiv:2003.00541.
 14. Wu, Y.; Yang, F.; Liu, Y.; Zha, X.; Yuan, S. A comparison of 1-D and 2-D deep convolutional neural networks in ECG classification. *arXiv* 2018, arXiv:1810.07088.
 15. Lorente, D.; Aleixos, N.; Gómez-Sanchis, J.; Cubero, S.; García-Navarrete, O.L.; Blasco, J. Recent Advances and Applications of Hyperspectral Imaging for Fruit and Vegetable Quality Assessment. *Food Bioprocess Technol.* 2011, 5, 1121–1142. [CrossRef]
 16. Mustaqeem, A.; Anwar, S.M.; Khan, A.R.; Majid, M. A statistical analysis based recommender model for heart disease patients. *Int. J. Med. Inform.* 2017, 108, 134–145. [CrossRef]
 17. Paola Pierleoni, S.M.; Gul, M.; Majid, M.; Alnowami, M. Arrhythmia Classification of ECG Signals Using Hybrid Features. *Comput. Math. Methods Med.* 2018, doi:10.1155/2018/1380348. [CrossRef]
 18. Bashir, A.K.; Arul, R.; Basheer, S.; Raja, G.; Jayaraman, R.; Qureshi, N.M.F. An optimal multi-tier resource allocation of cloud RAN in 5G using machine learning. *Trans. Emerg. Telecommun. Technol.* 2019, 30, 3627. [CrossRef]
 19. Sahu, Gupteswar, Sonali Dash, and Birendra Biswal. "Time-frequency analysis of power quality disturbances using synchroextracting transform." *International Transactions on Electrical Energy Systems* 30.4 (2020): e12278.

- 20.Li, Zhen, Jinghui Gao, and Zhiguo Wang. "A time-synchroextracting transform for the time–frequency analysis of seismic data." *IEEE Geoscience and Remote Sensing Letters* 17.5 (2019): 864-868.
- 21.Cho, Hyung-Ju, and Chin-Wan Chung. "An efficient and scalable approach to CNN queries in a road network." *International Conference on VLDB*. Vol. 2. International Conference on VLDB, 2005.
- 22.Cheng, Wentao, et al. "Jointly network: a network based on CNN and RBM for gesture recogni- tion." *Neural Computing and Applications* 31.1 (2019): 309-323.
- 23.Chauhan, Rahul, Kamal Kumar Ghanshala, and R. C. Joshi. "Convolutional neural network (CNN) for image detection and recognition." *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*. IEEE, 2018.
- 24.Vinayakumar, R., K. P. Soman, and Prabakaran Poornachandran. "Applying convolutional neu-ral network for network intrusion detection." *2017 International Conference on Advances in Com- puting, Communications and Informatics (ICACCI)*. IEEE, 2017.
- 25.Mai, Luo, et al. "Taming hyper-parameters in deep learning systems." *ACM SIGOPS Operating Systems Review* 53.1 (2019): 52-58.
- 26.Zhao, M., Li, J. (2018, March). Tuning the hyper-parameters of CMA-ES with tree-structured Parzen estimators. In *2018 Tenth International Conference on Advanced Computational Intelli-gence (ICACI)* (pp. 613-618). IEEE
- 27.M. Hadi Amini, et al.: A review of early fault diagnosis approaches and their applications in rotating machinery. *Entropy*. 21, 409 (2019).

Appendix

A. SOURCE CODE:

1.1. Synchro Extracting Transform using Matlab

```
function [IF Te tfr] = SET Y(x, hlength );[xrow ,  
  
xcol ] = size (x);  
  
N=xrow ;  
  
if ( xcol ~=1),  
    error ('X must be column vector ');end;  
  
if ( nargin  < 2),  
    hlength=round ( xrow /8 );end;  
  
t=1:N;  
ft = 1:round (N/2);  
  
[trow , tcol ] = size (t);  
  
hlength=hlength+1-rem ( hlength , 2 );  
ht = linspace ( -0.5,0.5, hlength ); ht=ht ' ;  
  
% Gaussian window  
h = exp(-pi /0.32 ^2 * ht .^ 2 );  
% derivative of window  
  
dh = -2*pi /0.32 ^2 * ht .* h; % g '  
[hrow , hcol]=size (h); Lh=(hrow-1)/2;tfr  
  
1= zeros (N, tcol );  
  
tfr2= zeros (N, tcol );
```

```

for icol=1:tcoll,
    ti=t(icol); tau=-min([round(N/2)-1,Lh,ti-1]):min([round(N/2)-1,Lh,xrow-ti]); indices=rem(N+tau,N)+1;
    rSig = x(ti+tau,1);
    tfr1(indices,icol)=rSig.*conj(h(Lh+1+tau)); tfr2
(indices,icol)=rSig.*conj(dh(Lh+1+tau)); end;

tfr1=fft(tfr1); tf
r2=fft(tfr2);

tfr1=tfr1(1:round(N/2),:);
tfr2=tfr2(1:round(N/2),:);

va=N/hlength;
IF=zeros(round(N/2),tcol); tf
r=zeros(round(N/2),tcol);
E=mean(abs(x));

for i=1:round(N/2)%frequency f
    for j=1:N%time
        if abs(tfr1(i,j))>0.8*E%if you are interested in weak signals, you
            can delete

            %if abs(1-real(va*i*tfr2(i,j)/2/pi./tfr1(i,j)))<0.5 if
            abs(-real(va*i*tfr2(i,j)/2/pi./tfr1(i,j)))<0.5
            IF(i,j)=1;

            end
        end
    end

end

end

tfr=tfr1/(sum(h)/2);%the amplitude of tfr result has been pre-rectified.Te=tfr.
*IF;

```

CNN using Hyperparameters

```
import tensorflow as tf
import matplotlib.pyplot as plt
import cv2
import os
import numpy as np

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image

from tensorflow.keras.optimizers import RMSprop

img = image.load_img('C:/Users/admin/Desktop/dsktp/major doc/train/Resizeabnor_')
cv2.imread('C:/Users/admin/Desktop/dsktp/major doc/train/_Resizeabnormal/Class')
train = ImageDataGenerator()

validation = ImageDataGenerator()

train_dataset = train.flow_from_directory('C:/Users/admin/Desktop/dsktp/major d',
target_size=(200,200),batch_size=3,
class_mode='binary')t
raindataset.class_indices

model = tf.keras.models.Sequential([tf.keras.layers.Conv2D(16,(3,3),activation='relu',
on tf.keras.layers.MaxPool2D(2,2),
#

tf.keras.layers.Conv2D(32,(3,3),activation='relu'),tf.keras.layers.MaxPool2D(
2,2),
#

tf.keras.layers.Conv2D(64,(3,3),activation='relu'),tf.keras.layers.MaxPool2D(
2,2),
##
tf.keras.layers.Flatten()),##
tf.keras.layers.Dense(512,activation='relu'),##
tf.keras.layers.Dense(1,activation='sigmoid')])model.compile(loss='binary_crossentropy',
optimizer=RMSprop(learning_rate=0.001),metrics=['accuracy'])
history = model.fit(train_dataset,
```

```

        steps per epoch = _
        3, epochs =10
    )

plt.plot(history.history['accuracy'])
plt.title('Model Accuracy')

plt.xlabel('Epochs')
plt.ylabel('accuracy')

plt.legend(['train'], loc='upper left')
plt.show()

dir path = '_C:/Users/admin/Desktop/dsktp/major doc/train/test' for i
in os.listdir(dir path):
    _
img = image.load_img(dir path+'/' + i, target_size=(200,200)) plt.
imshow(img)

plt.show()

X=image.img_to_array(img)
X = np.expand_dims(X, axis =0)
images = np.vstack([X])
val = model.predict(images)
if val == 0:
    print("
abnormal")else :
    print(" normal")

```

SCREENSHOT

