International Journal on Emerging Technologies **11**(3): 897-903(2020)

ISSN No. (Print): 0975-8364
ISSN No. (Online): 2249-3255

An Approach of Convolution Neural Network to Detect Cancer from Medical Radiology Images

***Abhilash Shukla*[1] *and Atul Patel*[2]**
[1]*Assistant Professor (Faculty of Computer Science and Applications, CHARUSAT, Changa, India.*
[2]*Dean & Principal (Faculty of Computer Science and Applications, CHARUSAT, Changa, India.*

(Corresponding author: Abhilash Shukla)

(Received 11 March 2020, Revised 13 May 2020, Accepted 15 May 2020)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Convolution Neural Network (CNN) is the new and efficient way to process on image in this modern era of computer vision. Basic approach of CNN with its all layers input, hidden and output is discussed in detail in this paper. Different models of CNN with all its parameter also discussed in this paper. This Paper also contains the study of tools, technologies, libraries and frameworks required to build a CNN model. Finally this paper shows to implement CNN model and train it on a dataset of cancer. Paper also shows model accuracy and loss after the training.

Keywords: Convolution Neural Network (CNN), Image Processing, Bone cancer.

I. INTRODUCTION

A Convolution Neural Network (CNN) is one of the emerging techniques of deep learning to study and analyses digital image in the field of computer vision. Deep Q-network and Deep belief networks architecture is based on to the concepts of CNN. CNN basically consists of three categories of layers called input layer, hidden layer and output layer. Hidden layer playing a vital role in Image recognition and analysis, Video analysis, Natural Language Processing, Anomaly Detection and so many other things. This Hidden layer typically consists of Convolution layer, Activation layer, Pooling layer, Normalization layer, Fully Connected layer and Softmax layer. These layers are the main building blocks of any CNN models. Different model can be developed such as LeNet, AlexNet, ZFNet, GoogLeNet, VGGNet and Microsoft ResNet using the main building blocks of CNN. Many deep learning libraries like Tenserflow, Keras, Caffes, Microsoft Congnitive, PyTorch, Theano etc. provide a facility to develop and implement various CNN model to build an application which take a form of product of any research done in the area of CNN. The same things can be apply for medical images to examine, analyze and resulting various kind of cancer detection like brain cancer, bone cancer, breast cancer, lung cancer etc. from X-Ray, computer tomography (CT), magnetic resonance imaging(MRI) and ultrasound. Previously various Image segmentaion techniques are used to detect cancer from X-Ray images but there is no deterministics technique available for it. Its depend on the manual features extracted from image. CNN uses Deep Learning approach in which features are automatically exatracted and give more accurate result.

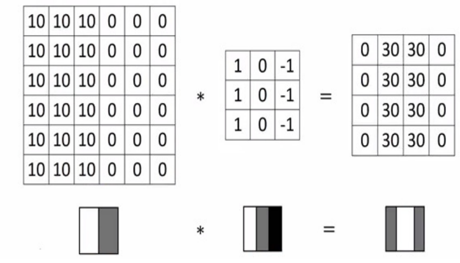
II. BASIC APPROACH OF CNN

As mention into introduction, CNN contain various hidden layer. This all layers worked as follow.

A. Convolution Layer

Convolution layer is used to extract a feature from an image. Convolution layer basically doing the convolution filtering on image using the mathematical formula. $y = x * w$. Here x is an image, w is a filter applies on image, $*$ is called convolution operator and y is resulting image of a convolution layer. Now mathematic formula for convolution operation is

$\int_o^n x.w\,dx.$



$$\begin{array}{|c|c|c|c|c|c|}\hline 10&10&10&0&0&0\\\hline 10&10&10&0&0&0\\\hline 10&10&10&0&0&0\\\hline 10&10&10&0&0&0\\\hline 10&10&10&0&0&0\\\hline 10&10&10&0&0&0\\\hline\end{array} * \begin{array}{|c|c|c|}\hline 1&0&-1\\\hline 1&0&-1\\\hline 1&0&-1\\\hline\end{array} = \begin{array}{|c|c|c|c|}\hline 0&30&30&0\\\hline 0&30&30&0\\\hline 0&30&30&0\\\hline 0&30&30&0\\\hline\end{array}$$

Fig. 1. Convolution Operation.

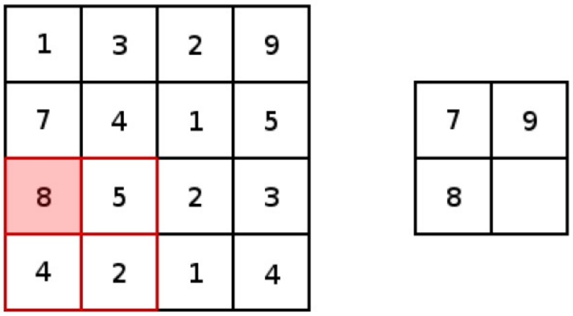
As a filter w, various filters can be used as Sobel, Prewitt, Schorr and many more but in CNN coefficients of filter are automatically estimated. Coefficients of different filters are as follow.

Table 1: Coefficients of different filters.

Filter	Vertical mask			Horizontal mask		
Sobel	1	0	-1	1	2	1
	2	0	-2	0	0	0
	1	0	-1	-1	-2	-1
Prewitt	1	0	-1	1	0	1
	0	0	0	0	0	0
	1	0	-1	-1	0	-1
Schorr	3	0	-3	3	10	3
	10	0	-10	0	0	0
	3	0	-3	-3	-10	-3
CNN	w1	w2	w3	**CNN automatically estimates the coefficients of the filter**		
	w4	w5	w6			
	w7	w8	w9			

B. Pooling and Padding Layer

This both layers are used to maintain the size of an original image. Pooling layer is used to reduce the size of an image. There are different techniques like max pooling, min pooling, average pooling etc are used to shrink the image. Example of max pooling is as under.

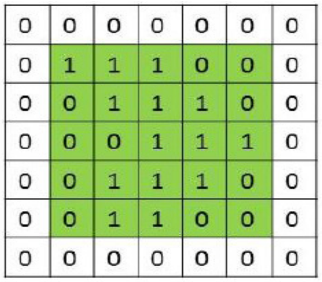


1	3	2	9
7	4	1	5
8	5	2	3
4	2	1	4

7	9
8	

Fig. 2. Max Pooling.

Padding layer does the reverse work than pooling layer. It is used to enhance the size of image by padding the image with the value 0 or 1. Example of padding is as under



0	0	0	0	0	0	0
0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0

Fig. 3. Padding.

Now normally after the convolution operation the size of image reduced based on the filter size taken. To overcome this, padding layer is used after the convolution operation. After the activation layer, pooling layer is used for object detection in image. Pooling layer also eliminate some parameter which result into less computation in CNN model.

C. Activation layer

In neural network Image interpretation, Speech Recognition, Text Recognition, etc are considered as unstructured data. Unstructured data are hard to learn compare to learn and interpreted the structured data. Activation function is used to learn this unstructured data. In deep learning as well as in CNN three activation functions: SIGMOID, TANH and ReLU are used in activation layer.

Sigmoid is a mathematical function which results into "S" shaped curved and range the output value of y from 0 to 1.

$$\hat{Y} = \frac{1}{1+e^{-(Z)}} = \frac{1}{1+e^{-(W^{T}X+b\ )}}$$

Here Z is the representation of neural network model

$$Z = W^{T}X + b$$

This formula derived from the formula of straight line $y = mx + b$ where m is the slop value which multiplied to the given value x and b is the bias which the value of y intercept. It can be represent in graph as

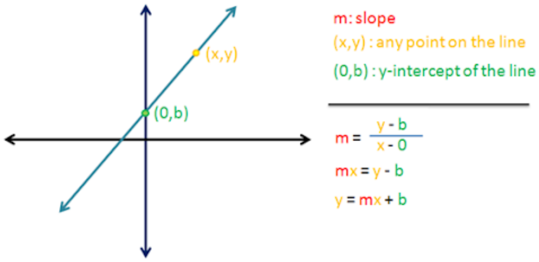

(x,y)
(0,b)
m: slope
(x,y) : any point on the line
(0,b) : y-intercept of the line
$m = \frac{y - b}{x - 0}$
$mx = y - b$
$y = mx + b$

Fig. 4. Straight Line Graph.

With the concept of straight line formula and sigmoid formula, the result of y range from 0 to 1 and represent in graph as below.

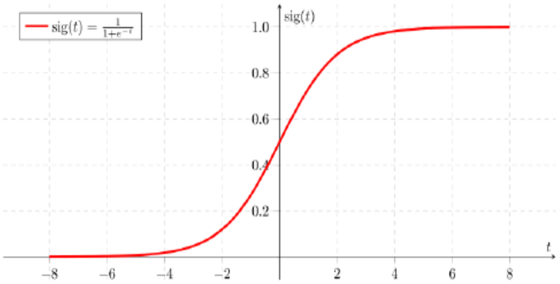


Fig. 5. Result of Sigmoid Function.

TanH (Hyperbolic Tangent Function) is the rescaling of logistic sigmoid function which ranges the value of y from -1 to 1. TanH work as

Here interpreta $\tanh = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$ as given in sigmoid function.

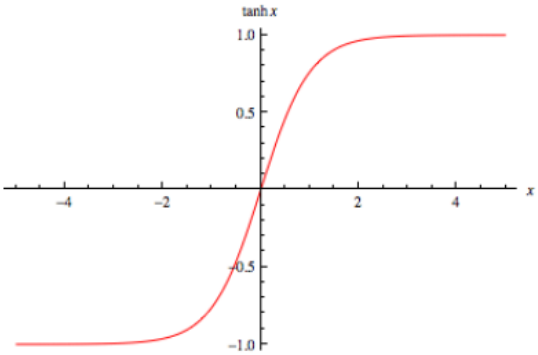


Fig. 6. Result of TanH Function.

ReLU (Rectified Linear Unit) is most commonly used activation function in neural network and especially in CNN. Mathematical representation of ReLU function is $y = \max(0, Z)$ which range the value of y from 0 to any positive value.

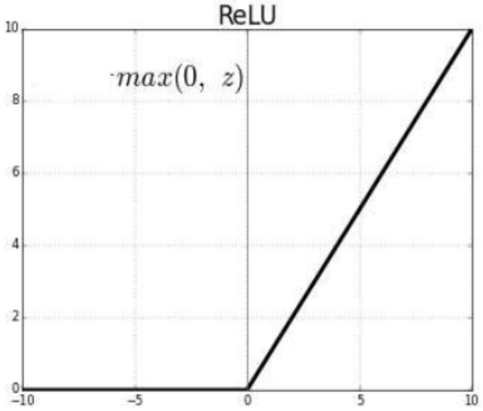


Fig. 7. Result of ReLU Function.

D. Normalization and Dropout Layer

Sometime at the time of training, CNN model may be over fit with the training dataset means model will work accurate with the data which used in training dataset or similar to it but not work properly for the data which have different behavior than the data provided to trained the model. This problem is called the overfitting problem in neural network and to overcome this problem in CNN, Normalization or Dropout techniques are used.

Normalization process ensures that all the input features of the next layer learned by the network at the same extend and make the learning more flexible. Mathematically it can be represent as

$$yi = \Upsilon xi^\wedge + \beta$$

Here $\Upsilon\ and\ \beta$ are considered as parameters to be learned. $xi^\wedge$ normalize as formula

$$xi^\wedge = \frac{xi - \mu\beta}{\sqrt{\sigma^2\,\beta + \varepsilon}}$$

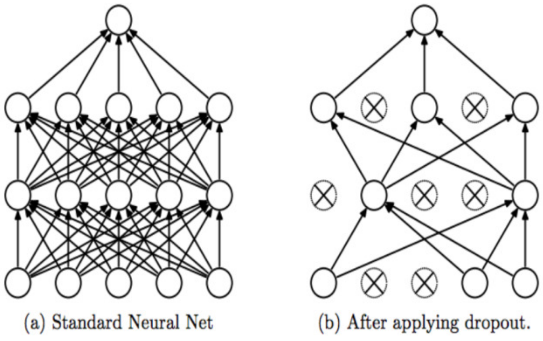
Here, $\mu\beta$ is mean and $\sigma^2\,\beta$ consider as variance which calculate as

$$\mu\beta = \frac{1}{m}\sum_{i=1}^{m} xi$$

$$\sigma^2\,\beta = \frac{1}{m}\sum_{i=1}^{m}(xi - \mu\beta)^2$$

m represents total values of x taken in calculation.

Dropout technique is one of easiest techniques to overcome overfitting issue in which some of the neuron randomly eliminated by the neural network for the input of next layer. Result of dropout can be understood from following figure.


(a) Standard Neural Net

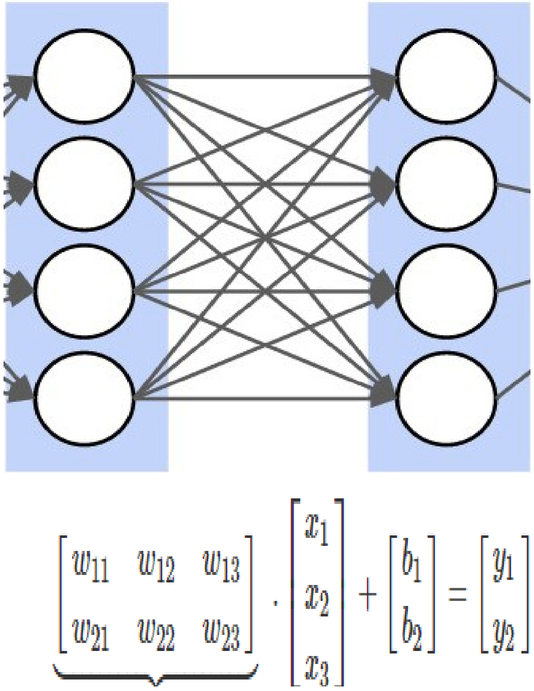
(b) After applying dropout.

Fig. 8. Dropout Technique.

It has been proved in paper titled "Determining Optimum Drop-out Rate for Neural Networks" that 50% neuron can be dropped to obtain better result.[3].

E. Fully Connected Layer

As discussed Convolution and Pooling layer are used to extract the features which now should be converted into form where image can be classify into the various classes. This can be achieved by applying fully connected layer after the result of normalization and dropout layer. In this layer all the inputs of one layer are connected to every activation unit of the next layer through which the result of previous layer flattened into a single vector on which next layer Softmax can be apply.



$$\underbrace{\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix}} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

Fig. 9. Fully Connected Layer.

F. SOFTMAX Layer

This is the final layer of CNN in which softmax function which is the type of squashing function applied and it result into the range 0 to 1. These final values can be interpreted as a probability. In this way softmax layer used for image classification in CNN model. Mathematics representation of Softmax layer is as follow

Suppose fully connected layer result into

$$z = \begin{bmatrix} y1 \\ y2 \\ y3 \\ y4 \end{bmatrix}$$

Then result will be obtained by following formula of softmax.

$$Zs = \begin{bmatrix} e^{y1}/(e^{y1} + e^{y2} + e^{y3} + e^{y4}) \\ e^{y2}/(e^{y1} + e^{y2} + e^{y3} + e^{y4}) \\ e^{y3}/(e^{y1} + e^{y2} + e^{y3} + e^{y4}) \\ e^{y4}/(e^{y1} + e^{y2} + e^{y3} + e^{y4}) \end{bmatrix}$$

III. CNN MODELS

A. LeNet

The LeNet architecture shown in figure 10 was proposed by Yann LeCun.[1]. LeNet will work well on black and white images. LeNet work as follow.

Step1: **[32x32]** INPUT
Step2: **[28x28]** CONV1: 6 5x5 filters at stride 1, tanh
Step3: **[14x14]** AVG POOL1: 2x2 filters at stride 2, tanh
Step4: **[10x10]** CONV2: 16 5x5 filters at stride 1, tanh
Step5: **[5x5]** AVG POOL2: 2x2 filters at stride 2, tanh
Step6: **[1x1]** CONV3: 120 5x5 filters at stride 1, tanh
Step7: **[84]** FC1: tanh
Step8: **[10]** FC2: softmax

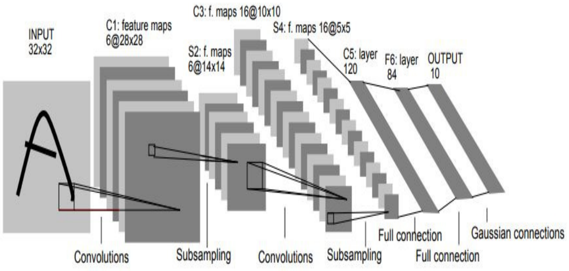


Fig. 10. LeNet CNN Model.

B. AlexNet, ZFNet and VGGNET

Architecture of AlexNet designed by Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever. AlexNet Architecture follows the same structure of LeNet but it is deeper. In major changes AlexNet using 227x227 image size and11x11 filter size. Pooling techniques is also changed in AlexNet. It used Max Pooling instead of Average Pooling. AlexNet achieved better result than LeNet by replacing activation function tanh with ReLU. AlexNet work as follow

Step1: [227x227x3] INPUT
Step2: [55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
Step3: [27x27x96] MAX POOL1: 3x3 filters at stride 2
Step4: [27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
Step5: [13x13x256] MAX POOL2: 3x3 filters at stride 2
Step6: [13x13x256] NORM2: Normalization layer
Step7: [13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
Step8: [13x13x384] CONV4: c384 3x3 filters at stride 1, pad 1
Step9: [13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
Step10: [6x6x256] MAX POOL3: 3x3 filters at stride 2
Step11: [4096] FC6: 4096 neurons
Step12: [4096] FC7: 4096 neurons
Step13: [1000] FC8: 1000 neurons
Step14: Softmax(class scores)
Overall of AlexNet architecture look like below.

In ZFNet two more Convolution Layers means total seven convolution layers are used and reduce initial filter size from 11x11 to 7x7.

To optimize the result of AlexNet and ZFNet, In VGGNet total 13 Convolution Layer are used and reduce initial filter size from 11x11 to 3x3.

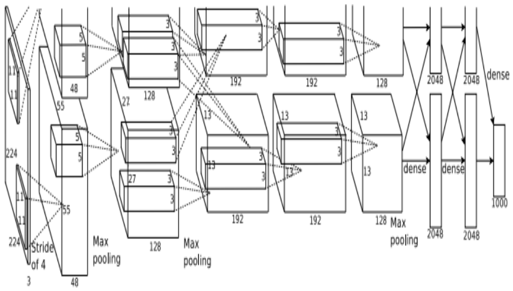


Fig. 11. AlexNet CNN Model.

C. GoogleNet with Inception Module

In each layer of CNN, the techniques of pooling as well as some filter size e.g. 2x2, 3x3, 5x5, 11x11, etc. are selected and it will apply on whole image. It is very complex to select techniques which provide optimize result. All of these are actually beneficial to the modeling power of the CNN network. To come from the dilemma of pooling techniques and filter size, inception module introduce in GoogleNet, in which the composition of pooling techniques and filter size taken to extract features from image instead of selecting particular pooling techniques or filter size. Finally the results obtained from all these composition are merged and it's become an input for the next layer of CNN.

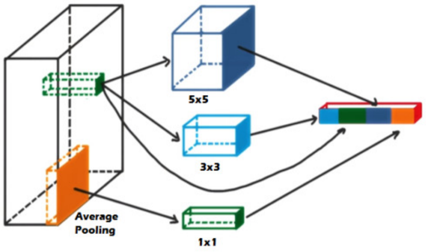


Fig. 12. Inception Model.

D. Microsoft ResNet with Residual Block

To help the neural network to learn the identity function, residual block introduce in Microsoft ResNet CNN model. Residual block contains two of path. First path is the regular neural network path and it is consider as main branch of residual block. Second path is the direct path from input to the output which is known as shortcut path. Finally, the result of both path merged and the ReLU activation function applied on it.

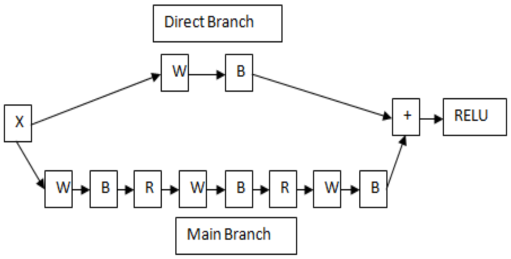


Fig. 13. *Residual Block.*

E. Comparison of CNN Models

Below table shows the comparative study of all CNN models discussed.

Table 2: Comparative Study of different CNN Models.

ConvNet	Year	Developed By	No. of Convolution Layer	Filter Size	Input Image Pixel	Top 5 Error Rate*	Parameter
LeNet	1998	Yann LeCun	5	5x5	32X32 (Grey scale)	26%	60 thousands
AlexNet	2012	Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever	5 +3FC	11x11	227×227x3	15.3%	60 millions
ZFNet	2013	Matthew Zeiler And Ron Fergus	7 +3FC	7x7	224X224x3	14.8%	60 millions
GoogLeNet	2014	Google	22	1x1	227x227x3	6.67%	4 millions
VGGNet	2014	Simonyan, Zisserman	13 +3FC	3x3	224X224x3	7.3%	138 millions
MS ResNet	2015	Kaiming He	152	3x3 1x1	256×256x3 64x64x3	3.57%	25.5 millions (ResNet 50)

*** ImageNet Large Scale Visual Recognition Challenge (ILSVRC)**

Tools and Techniques to Implement CNN: There are so many deep learning libraries available which provides in-built features to implement CNN model. Some of the well known libraries are listed below in the table.

Table 3: List of Deep Learning Libraries

Deep Learning Libraries	
TenserFlow	Theano
Keras	TFLearn
Caffe	Torch
Microsoft Cognitive	Caffe2
PyTorch	PaddlePaddle
Apache MXnet	DLib
DeepLearning4J	Chainer
Lasagne	Neon

Details study of python based libraries and framework done by Thaker Nimit & Shukla Abhilash in their paper titled "Python as Multi Paradigm Programming Language." [4]

According to survey done by Dan Clark, KDnuggets in 2018, Tenserflow is the most used library for the implementation of AI. This survey is done based on contributors and stars get by these libraries on github.

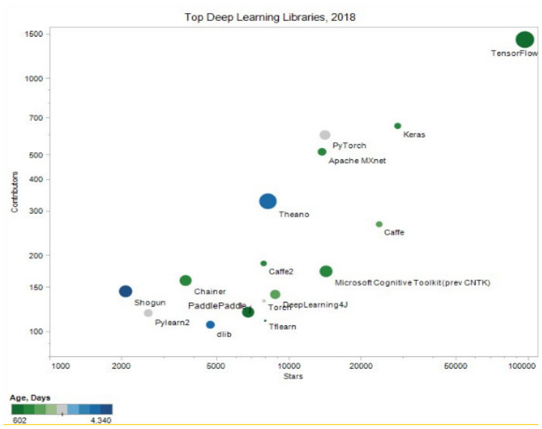


Fig. 14. Shows Popularity of Deep Learning Libraries.

IV. IMPLEMENTATION OF CNN

Below code demonstrate to build a CNN model and to train it on training dataset of breast cancer. Code is written in Python which use Tenserflow libraries to build a CNN model. Code is implemented in Google Coble IDE.

Python Code

```
# install tenserflow !pip install tensorflow-gpu==2.0.0-rc0
# Import Require Libraries
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Flatten, Dense, Dro
pout, BatchNormalization
from tensorflow.keras.layers import Conv1D, MaxPool1
D
from tensorflow.keras.optimizers import Adam
import pandas as p
import numpy as n
import seaborn as sb
import matplotlib.pyplot as pt
from sklearn import datasets, metrics
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
#Load Dataset
cancer = datasets.load_breast_cancer()
X = p.DataFrame(data = cancer.data, columns=cancer.f
eature_names)
y = cancer.target
#Conver into type which can provide to CNN model
sc = StandardScaler()
X_training = sc.fit_transform(X_training)
X_testing = sc.transform(X_testing)
#Reshaping into 3 Dimensions
X_training=X_training.reshape(455,30,1)
X_testing=X_testing.reshape(114, 30, 1)
#Prepare Model
iteration = 50
CN = Sequential()
CN.add(Conv1D(filters=32, kernel_size=2, activation='re
lu',input_shape=(30,1)))
CN.add(BatchNormalization())
Cmodel.add(Dropout(0.2))
CN.add(Conv1D(filters=64, kernel_size=2, activation='re
lu'))
CN.add(BatchNormalization())
CN.add(Dropout(0.5))
CN.add(Flatten())
CN.add(Dense(64, activation='relu'))
CN.add(Dropout(0.5))
CN.add(Dense(1,activation='sigmoid'))
#Compile Model and Fit the data into model
```

```
CN.compile(optimizer=Adam(lr=0.00005), loss = 'binary
_crossentropy', metrics=['accuracy'])
h = CN.fit(X_training, y_training, epochs=iteration, valid
ation_data=(X_testing, y_testing), verbose=1)
#Visualize the Model Accuracy and Model Loss
def plot_LC(h,e):
# Plot accuracy
er = range(1, e+1)
  pt.plot(er, h.history['accuracy'])
pt.plot(er, h.history['val_accuracy'])
pt.title('Accuracy of Model')
pt.ylabel('Accuracy')
pt.xlabel('Epoch')
pt.legend(['Train', 'Val'], loc='upper left')
pt.show()
# Plot loss
pt.plot(er,h.history['loss'])
pt.plot(er,h.history['val_loss'])
pt.title('Loss of Model')
pt.ylabel('Loss')
pt.xlabel('Epoch')
pt.legend(['Train', 'Val'], loc='upper left')
pt.show()
h.history
plot_LC(h,e)
```

V. RESULT

Below figures shows the Model Accuracy and Model Loss achieved at every epoch.

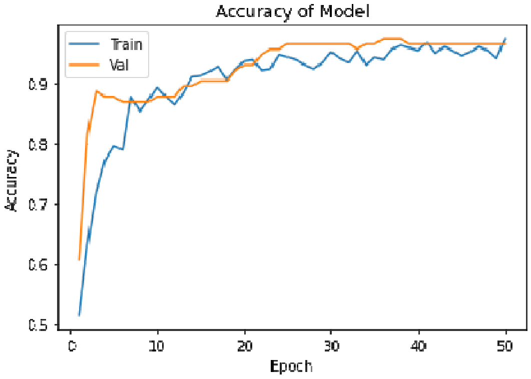


Fig. 15. Model Accuracy.

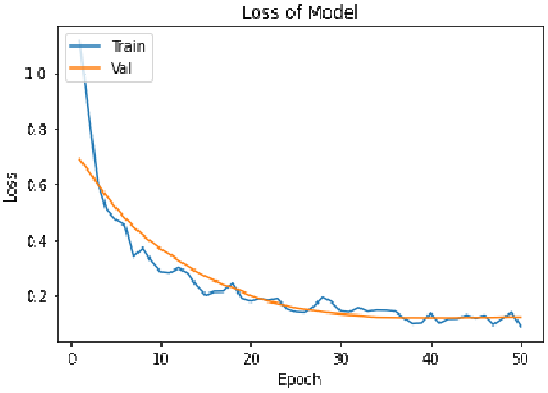


Fig. 16. Model Loss.

VI. FUTURE SCOPE

The future work will be to compare various other model of CNN and find best suited model to detect cancer.

ACKNOWLEDGMENT

The authors are grateful to the authorities Charotar University of Science and Technology (CHARUSAT), Gujarat, India.

Conflict of Interest. No.

REFERENCES

[1]. Lecun, Yann & Bottou, Leon & Bengio, Y. & Haffner, Patrick. (1998). Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE. 86. 2278 - 2324. 10.1109/5.726791.

[2]. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems,2012.

[3]. Josiah Yoder. Determining Optimum Drop-out Rate for Neural Networks. The Bridge, The Magazine of IEEE-Eta Kappa Nu (IEEE-HKN), Vol 115, Issue 2, June 2018.

[4]. Thaker Nimit & Shukla Abhilash. (2020). Python as Multi Paradigm Programming Language. International Journal of Computer Applications. 177. 38-42. 10.5120/ijca2020919775.

[5]. Lei, Xinyu & Pan, Hongguang & Huang, Xiangdong. (2019). A dilated CNN model for image classification. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2927169.

[6]. Demir, Fatih & Abdullah, Daban & Sengur, Abdulkadir. (2020). A New Deep CNN model for Environmental Sound Classification. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.2984903.

[7]. Chengyu Wang, Daiyun Huang, Jionglong Su, Limin Yu and Fei Ma (2020). Generalization and Visual Comprehension of CNN Models on Chromosome Images. Journal of Physics: Conference Series. 1487. 012027. 10.1088/1742-6596/1487/1/012027.

[8]. Perumal, Muthu & Raja, S.Selvakumar. (2019). Convolutional Neural Network based Image Classification and Detection of Abnormalities in MRI Brain Images. 0548-0553. 10.1109/ICCSP.2019.8697915.

[9]. Ševo, Igor & Avramovic, Aleksej. (2016). Convolutional Neural Network Based Automatic Object Detection on Aerial Images. IEEE Geoscience and Remote Sensing Letters. 13. 1-5. 10.1109/LGRS.2016.2542358.

[10].Bharali, Parismita & Bhuyan, Chandrika & Boruah, Abhijit. (2019). Plant Disease Detection by Leaf Image Classification Using Convolutional Neural Network. 10.1007/978-981-15-1384-8_16.

[11]. Manikonda, Santhosh & Gaonkar, Dattatraya. (2019). An islanding detection method based on image classification with convolution neural network. The Journal of Engineering. 10.1049/joe.2019.0025.

[12].Niu, L. & Song, Y. & Su, J. & Zhang, H. (2018). A DEEP LEARNING STUDY OF EXTRACTING

NAVIGATION AREA FROM CAD BLUEPRINTS. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. IV-4. 155-162. 10.5194/isprs-annals-IV-4-155-2018.

[13]. Ganatra Nilay & Patel Atul. (2018). A Comprehensive Study of Deep Learning Architectures, Applications and Tools. International Journal of Computer Sciences and Engineering. 6. 701-705. 10.26438/ijcse/v6i12.701705.

[14]. Patel Sanskruti & Patel Atul. (2018). Deep Leaning Architectures and its Applications A Survey. International Journal of Computer Sciences and Engineering. 6. 1177-1183. 10.26438/ijcse/v6i6.11771183.

[15]. Priya, N. and Radhika, C. (2019). A Survey on Predicting Autism Spectrum Disorder using Machine Learning techniques. *International Journal on Emerging Technologies, 10*(3): 10(3): 422–427.

How to cite this article: Shukla, A. and Patel, A. (2020). An Approach of Convolution Neural Network to Detect Cancer from Medical Radiology Images. *International Journal on Emerging Technologies*, *11*(3): 897–903.