International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6, August, 2019

Deep Learning to Detect Skin Cancer using Google Colab

Pratik Kanani, Mamta Padole

Abstract*: *Different mathematical models, Artificial Intelligence approach and Past recorded data set is combined to formulate Machine Learning. Machine Learning uses different learning algorithms for different types of data and has been classified into three types. The advantage of this learning is that it uses Artificial Neural Network and based on the error rates, it adjusts the weights to improve itself in further epochs. But, Machine Learning works well only when the features are defined accurately. Deciding which feature to select needs good domain knowledge which makes Machine Learning developer dependable. The lack of domain knowledge affects the performance. This dependency inspired the invention of Deep Learning. Deep Learning can detect features through self-training models and is able to give better results compared to using Artificial Intelligence or Machine Learning. It uses different functions like ReLU, Gradient Descend and Optimizers, which makes it the best thing available so far. To efficiently apply such optimizers, one should have the knowledge of mathematical computations and convolutions running behind the layers. It also uses different pooling layers to get the features. But these Modern Approaches need high level of computation which requires CPU and GPUs. In case, if, such high computational power, if hardware is not available then one can use Google Colaboratory framework. The Deep Learning Approach is proven to improve the skin cancer detection as demonstrated in this paper. The paper also aims to provide the circumstantial knowledge to the reader of various practices mentioned above.

Keywords* : *Deep Learning, Google Colab, Optimizers, Skin Cancer Detection

I. INTRODUCTION

Machine Learning has three types of algorithms classified as supervised, unsupervised and reinforcement learning. Supervised learning is used when the data and its label is known. On the other hand, unsupervised is used to predict data when class labels are unknown e.g., K- Means clustering. Reinforcement learning is very popular in game theory where agent takes the possible actions and it gets either rewards or penalty, which acts as learning condition for agents.

Revised Manuscript Received on October 30, 2019.

* Correspondence Author

Pratik Kanani*, Department of Computer Science and Engineering, The Maharaja Sayajirao University of Baroda, India

Dr. Mamta Padole, Associate Professor, Department of Computer Science and Engineering, The Maharaja Sayajirao University of Baroda, India.

Machine Learning requires moderate computing which can be done using CPU. Machine Learning is good technique but does not show improvement in its performance with increase in data. This demands a new approach referred as Deep Learning.

Deep Learning is the subset of Machine Learning which in turn the subset of Artificial Intelligence. Deep Learning consists of input, multiple hidden and output layers made up of Neural Networks. It improves the performance with increase in the dataset, i.e. for more dataset it learns far better than Machine Learning. Also, Deep Learning automatically considers all minute features available in the dataset and selects the most important ones and proceeds with learning. Also, it uses different layers like convolution, softmax, pooling layers and optimizers. Each and every layer has different meaning and computations behind the scenes. The order in which different layers are connected is very important. After every epoch the assigned weights should be adjusted again to make the layer more stable. While learning, it calculates the error and updates the weights after each epoch. This way it learns the best and gives more accurate results. Deep Learning requires larger amount of calculations where CPU is not sufficient. So, one has to use GPU too.

Google has made Google Colaboratory which is the online framework where one can write, execute Deep Learning and Machine Learning codes. Here, different versions of python and different runtime environments are available. Also, it can download bigger datasets directly from the servers to google drive at very high speed. You can mount your drive with Google Colab and it can fetch the required file after your authentication. How long the amount of memory and computation provided by Google is available for user is also discussed. It provides very high-speed computations and then saves the Learning model. One can use this hex model for the predictions. In this paper, the Deep Learning terminologies and methods used are explained along with the CPU and GPU concepts. Also, if such hardware resources are not available, then one can use Google Colaboratory framework. Different development and deployment steps for Google Colab are explained here. Skin cancer detection from images is illustrated as an example to corroborate the same.

II. DEEP LEARNING

Deep Learning is the subset of machine learning which is selecting the features. It uses Artificial Neural Networks (ANN) [1] and more computation power.

Machine Learning takes less training time and more testing time whereas the Deep Learning takes more training time and less testing time which makes it more suitable for deployment. Deep learning explores all the feature, parameters and hyperparameters and then applies the filter to decide which features to carry on with.

Such computations are done at each and every layer. Because of large number of computations, Deep Learning takes more computation time. Deep Learning Architecture uses multiple layers like input layer, hidden layers and output layers. To work effectively it has to adjust weights and errors, so that the optimal results can be predicted. Different Deep Learning Terminologies are defined below.

A. Optimizers/Activation Functions

Activation functions are non-linear transformations of inputs and biases to identify whether a given neuron will fire or not. It is a non-linear complex functional mapping of input and output which is often given to the next neuron layer [3]. There are different types of activation functions, out of which the four main types are ReLu, Softmax, Sigmoidal and Tanh function [3].

a) Sigmoid Activation function - This is a non-linear continuously differentiable activation function. It is an S-shaped curve and its value ranges from 0 to 1. This is particularly used in classification of given tuples into 2 classes. It is easy to calculate but not symmetrical about the origin and values are ranging only from 0 to 1.

b) Tan h Activation Function - This is a non-linear continuously differentiable activation function. It is an extended version of Sigmoid function. Its values range from -1 to 1. Tan h function is symmetrical about the origin. It is Symmetrical about the origin and ranges values from -1 to 1, but Gradient of tan h function is steeper which may lead to over fitting problem.

c) ReLu Activation Function - ReLu stands for Rectified Linear Unit. It is non-linear activation function. It is very popular in Deep learning and is the most commonly used. Its values range from 0 to infinity. The main advantage of ReLu is that it only activates few neurons at a time. The neurons which have a negative value do not get activated. Hence, as many neurons have negative values, only some get fired at any time instant. Thus, it makes the network sparse and easy for computation. Another variation of this activation function is leaky ReLu. It is most commonly used activation function as it is very simple to calculate, dead neurons can be identified by Leaky Relu activation function. Sometime it causes the gradient to move towards 0 which may cause a problem.

d) Softmax Activation Function - Sigmoid function is only used to classify into two classes [5]. In Softmax, the output for each class is squeezed between 0 to 1. This kind of signifies the probability of an input belonging to a particular class.

B. Convolution

Convolution neural networks use the concept of convolution. It is a very popular and upcoming technique [7]. In convolution, the input matrix is mixed with the kernel/filter to obtain a feature map. For example, image as an input would be a three-dimensional matrix which holds the value of R-B-G of each pixel [10]. A kernel is decided and this kernel is mixed with the input matrix. Now, this mixture is called a feature map. Convolution reduces the space complexity and is therefore more efficient. Convolution can be defined as mixing of information in a particular way to reduce the space complexity and also retain as much information as possible.

C. Max-Pooling

Pooling can be defined as choosing a value from the window in a particular way to retain the information and reduce the dimensionality [6]. In max-pooling, one selects the maximum value from all available values in the window thus incorporating only the most salient feature of that pixel. Dimensionality of a particular representation is reduced due to pooling. For example, if one has a 3*3 window size over 10*10 feature matrix, then one selects only the maximum value out of the available 9 values to represent that feature window. Average pooling is another technique where instead of just taking the maximum value, the average of all values in the given window is used for representing that feature.

D. Fine Tuning

Fine Tuning is the process of taking weights of a trained neural network for initializing the weights in the new model that is being trained on the same domain to reduce the training time and speed up the process [11-12]. It allows one to incorporate feature extraction done in the previous model without developing them from scratch. There are two major strategies in fine tuning. One is replicating the whole network as initiation network and then performing deep learning on it [12]. Another approach is freezing some pretrained weights and using them for deep learning.

E. Dropout

Dropout is a regularization technique in deep learning to avoid overfitting. Regularization is the process of solving the interdependency between neurons. This inter-dependency causes over fitting in data [13]. In dropout, at each training phase, a fraction of neurons as decided by the dropout rate are ignored randomly. This solves the problem of co-dependency among neurons and reinvents the individual power of each neuron [15].

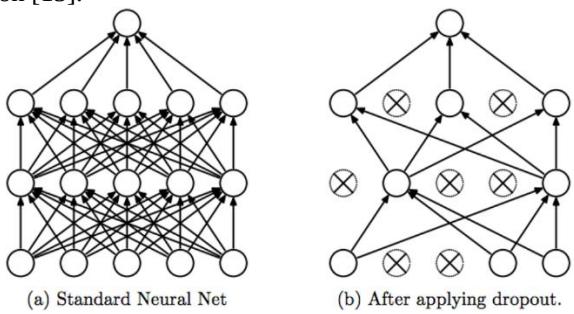

Fig. 1. Dropout in Machine Learning [13]

F. Vanishing Gradient Problem (VGP)

Vanishing gradient problem occurs in Neural networks which use gradient descent approach [2]. A neural network in deep learning consist of thousands and thousands of hidden layers. As there are several hidden layers, it takes time to backpropagate the loss (Error) to the previous hidden layer. Thus, the initial layers take a lot of time to adapt and this causes loss in performance [16]. The neurons in the earlier layers learn very slowly which causes degradation in performance.

Thus, the training phase takes too long and the accuracy of the algorithm is very less. This is solved using ReLu activation function as it improves the accuracy.

G. Exploding Gradient Problem

It is the opposite of VGP. In exploding gradient problem, large error gradients accumulate which causes instability in the network [2]. This is because if large values accumulate and manipulate the weights, these weights keep on changing drastically and the network does not remain stable [17]. In extreme cases, the values of weights become so large which causes overflowing and result in NaN values [18]. Because of unstable network, the network cannot learn efficiently over a long sequence of data. Gradient Clipping is used to solve this problem, where the error values are checked against a threshold value and clipped if necessary.

H. Optimizers in Deep Learning

Gradient descent is used in Deep Learning for updating the values. Gradient descent has several issues which can be solved by various optimizing algorithms. Some of them are listed below [20].

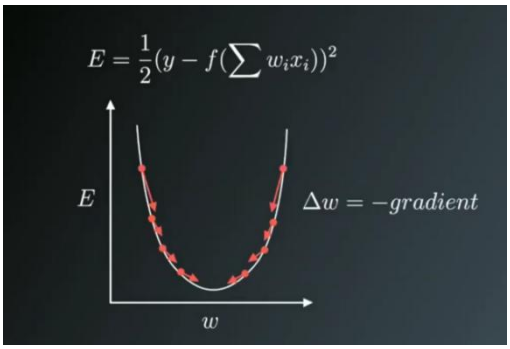

Fig. 2. Gradient Descent [20]

1. Momentum - Generally, gradient descent algorithm suffers from high variance oscillations. Thus, it makes it hard for SGD to reach convergence. This is improved by introducing momentum which accelerates the SGD by navigating in relevant direction and softens the oscillations towards irrelevant directions.
2. Nesterov Accelerated Gradient - In momentum, a fixed parameter is used to update the value of the gradient. This sometimes causes the gradient to miss the optimized point and hence there is a need to variant the value of momentum parameter. This is solved by this method. In this, the new momentum term is manipulated based on the current momentum term.
3. Adagrad - In this method, it manipulates the learning rate η based on the parameters. So, the error for frequent parameters is less and for infrequent parameters is more. It uses different learning rate for different parameters. Adagrad changes the learning rate for every parameter at each time instant based on the past calculated gradient descents. Main disadvantage of this method is that its learning rate is always decaying and decreasing as it accumulates all previous square gradients.
4. AdaDelta - The problem is Adagrad is solved by AdaDelta. In this, instead of taking all the previous square gradients into account, only a fixed number of previous values are taken into consideration which is denoted by w. The running average depends on the current gradient and the previous average. This prevents the vanishing learning rates.
5. Adam - It stands for Adaptive Moment Estimation. AdaDelta only stores the exponentially decaying average of past square gradients. Adam also stores the exponentially decaying average of gradients. It is a mixture of momentum and Adadelta method. M(t) is the first gradient which is also called as the mean and V(t) is the second gradient which is also called as the un centered variance.

I. Ranking based average precision

Ranking based average, also known as weighted rank average is an evaluation measure in machine learning. Ranking is a central part of many information retrieval problems. In this, the predictions are based on their weighted mean and then ranking the predictions in descending order to find the best possible solution.

J. Weighted ranking loss

Weighted ranking loss is the evaluation measure that is used to obtain the weighted rank average and is also a performance measure of the learned rank function. The loss functions are usually the upper bound of such measure-based ranking errors.

K. Coverage error

Coverage error are non-sampling errors and often result in bias in the data and also affects the representatives of the data. This error comprises of under enumeration or over enumeration of the dataset. Coverage error is encountered when there are differences between the target dataset and the sample frame.

III. THE CPU AND THE GPU

No algorithm can be executed without computation. The computation power is given by either the CPU or the GPU. To clear the idea, structure, working and applications of CPU and GPU are given below.

A. Comparing CPU and GPU

1. Definition: A CPU of the computer is a collection of transistors and multiprocessors which performs a variety of calculations to accomplish a given task by manipulating the transistors [21]. The brain of the computer is a CPU. A GPU is a specialized highly customized micro-processing chip which enacts parallelism and performs very specific tasks and optimizes the display graphic [22]. Generally, all computers consist of a CPU but a GPU has to be externally installed to enhance display quality.
2. Cores: A CPU has a very few cores ranging from a single core to octa-core processors. These cores have large cache
3. memory which can perform few software threads. On the other hand, a GPU has multiple cores ranging in hundreds having abundance of cache memory which can perform innumerable threads [23].

4. Degree of Parallelism: A CPU has low degree of parallelism as compared to a Graphical Processing Unit (GPU). GPU performs many tasks simultaneously at a given instant of time as it has thousands of cores running simultaneously [26]. As a result, GPU can render complex 3D graphics which is a necessity for intricate and complex games [27].
5. Flexibility: A CPU has a higher degree of flexibility as compared to GPU [25]. CPU can perform a humungous amount of a variety of different mathematics and calculations. On the other hand, GPU is designed for a very specific application
6. Components: There are three different components of CPU [24] which are ALU, control unit and registers.
   a) ALU - An Arithmetic Logic Unit is an electrical digital circuit which performs various arithmetic and logical operations.
   b) Control Unit - A control unit interacts with IO devices, main memory and different registers. Its main function is to offer its services to the external devices. It allows instructions to be executed in the logic unit. It also accesses instructions in RAM and ROM.
   c) Registers - These are short term memory storage devices which are used to accept, store and transfer instructions, data and other variables that are being used by the CPU.

A GPU on the other hand consists of numerous cores and transistors. It is a significant element of a graphic card. A graphic card is an external device which is used to enhance video quality and speed up operations. It is either a video adapter or a display card [24]. A graphic card consists of DVI/HDMI ports, CrossFire Link, 3 pin fan connector, Power connectors, interface and a GPU.

7. Functions: A CPU has numerous functions [24] such as
   - Fetch instructions
   - Decode instructions
   - Execute instructions
   - Write data to main memory
   - Implement control instructions meant for timing and memory storage purposes
   - Perform mathematical and logical calculations
   - Give command to various components of the computer

A GPU has following functions [24]

- Perform complex and intricate 3-dimensional calculations
- Render animations, videos, images for computer display unit
- Optimized for display functions and operations
- Optimized for floating point operations

8. Speed: A CPU has higher clock speed as compared to a GPU. Hence, individual operations are run at a faster rate in CPU [23]. But due to high degree of parallelism, overall operations are run at a much faster rate in GPU. Hence, the overall time needed to perform a particular task which consists of multiple operations is less in GPU.
9. Memory Requirement and Consumption: A CPU generally needs more memory as it needs to store many values because of less degree of parallelism [28]. Thus, the memory consumption is also high in CPU. This is improved in GPU as it has many cores and the result can be integrated and stored in less memory blocks.
10. A CPU prefers low latency as it does not execute many instructions simultaneously [25]. But a GPU prefers high latency as the instructions executed in parallel may be dependent on each other and thus, require some delay to resolve the dependency.
11. Installation: A CPU is always preinstalled in the computer. It is the heart of any computer. But the installation of a graphic card is a bit complex. You need to first locate the PCI-e slot on the motherboard and use the x-16 slots to connect the graphic card. And to enable the functioning of the graphic card, you need to download AMD and NVIDIA drivers. There are various toolkits available as well to help with the process.

IV. GOOGLE COLAB

Google Colaboratory widely known as Google Colab is an open source service provided by Google to any person having a Gmail account. Google Colab [33] provides GPU for research to the people who do not have enough resources or cannot afford one. The Google Colab service provides 12.72 GB of RAM and 358.27 GB of hard disk space in one runtime. Every runtime lasts for 12 hours after which the runtime is reset and the user has to establish a connection again. This is to ensure that people do not use the GPU service for crypto currency mining and other illegal purposes. Different run cases are shown below.

Once the user opens a Google Colab file they need to select a runtime type. There are 3 available options for the same,

1. None (that will use the computer's CPU the user is using)
2. GPU
3. TPU (especially for tensor processing)

The selection box is found in Runtime -> Change runtime type and looks like the image given below.

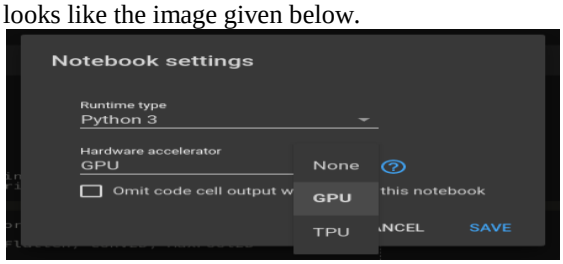

Fig. 3. Google Colab Notebook Setting

Next one needs to connect the file to a runtime. This connect button is available at the top right corner of the page that looks something like in the image given below.

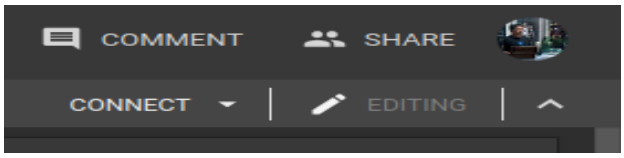

Fig. 4. File Connect

To execute a cell one needs to press Ctrl + Enter or click on the play button on the extreme left of every cell as given below.

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
```

Fig. 5. A Cell with run button

Google Colab is essentially an online Jupyter Notebook and has all functionalities of one. Users can connect Google Colab to their Google Drives using the code snippet given below.

```
from google.colab import drive
drive.mount('/content/drive')
import os
os.chdir("drive/My Drive/app")

Go to this URL in a browser: https://accounts.google.com/o/oa

Enter your authorization code:
······
```

Fig. 6. Google Colab with Google Drive Authentication

The above given code snippet is used to connect the Google Colab file to the Google drive account. This gives one the flexibility in using any type of code one wants to use and any size of dataset one wishes to have. The above code snippet mounts an image of Google Drive onto Google Colab. When the code snippet is executed, the user is prompted with a URL and a text box. The user has to click on the URL and give permissions to Google Colab to have access to our Google Drive and it's files, once the user is granted permission a one time token is provided that needs to be entered into the textbox below the URL. If the token is identified and authenticated, Google Drive gets successfully mounted onto Google Colab. Once Google Drive is mounted onto Google Colab one can either use it as a Jupyter Notebook and write the working code into the cells or else have python files in the Google Drive and execute those files from the Google Colab file just like one executes them from the Linux terminal but with a '!' mark before the statement as given below.

```
!python3 main.py
```

Fig. 7. Executing main.py file

Google Colab also allows one to directly download datasets into the Google Drive from Kaggle. Kaggle generates an API key and gives a .json file with the key inside it. One needs to execute the following code snippet after that.

```
!pip install -q kaggle
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!ls ~/.kaggle
!chmod 600 /root/.kaggle/kaggle.json  # set permission
```

Fig. 8. Google Colab and Kaggle Dataset linking for download

This will establish a connection between Kaggle and Google Colab and to download the dataset to execute the following line:

!kaggle competitions download -c 'name_of_competition' -p "target_colab_dir"

V. SKIN CANCER DETECTION USING DEEP LEARNING AND GOOGLE COLAB

Here in this paper, a deep learning model to identify 7 different types of skin cancer using the HAM10000 ("Human Against Machine with 10000 training images") dataset [35-36] has been proposed. This dataset consists of 10,015 dermatoscopic images that are released in the form of a training dataset for academic and machine learning purposes and are available publicly through the ISIC archive [35]. This benchmark dataset can be used for machine learning and for comparisons with human experts. It has been noticed that human experts can identify the skin cancer with an accuracy of approximately 75% [34]. The proposed model tends to achieve a higher accuracy than the one proposed earlier by [34].

The seven different types of skin cancer that will be detected by the proposed model are:

1. Melanocytic nevi
2. Melanoma
3. Benign keratosis like lesions
4. Basal cell carcinoma
5. Actinic keratoses
6. Vascular lesions
7. Dermatofibroma

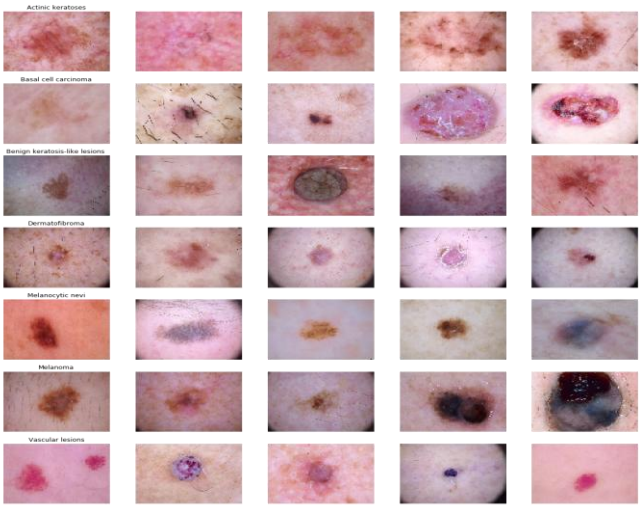

Fig. 9. Skin Cancer dataset images

A. Preprocessing:

In the preprocessing stage one, first resize the images from their original size of (600 x 450) to (100 x 75) otherwise the Keras library would not be able to handle such large scale images, even if it does, a lot of RAM would be required. Hence, it is best to downsize the images. All the images used are RGB images hence have 3 channels each for one each color. Accordingly, the absolute dimensions of the image after downsizing are (100 x 75 x 3). One must also augment the dataset. In order to avoid overfitting of the model, one artificially expands the HAM 10000 dataset. This process makes the existing dataset even larger. There are different approaches that alter the training data in ways and also change the array representations keeping the label of the dataset same known as data augmentation techniques.

A few popular augmentations that are used by people are horizontal flips, grayscales, random crops, vertical flips, rotations, color jitters, translations, and so on. By applying just a few of these transformations to the training dataset, one can easily increase the number of training examples by two or three times and create a very robust model without any overfitting problems and with greater accuracy. The augmentations applied are rotate, zoom, height shift and width shift.

B. Model

Keras Sequential API has been used in the code, where one has to add only one layer at a time, starting from the input layer. The first layer is a convolutional (Conv2D) layer which is like a set of learnable variables and filters. 32, 64, 128 and 256 filters for the conv2D layers have been chosen, and 1024 filters in the last fully connected layer. Every filter transforms a part of the image (that is defined by the size of the kernel) using the kernel filter. The kernel filter matrix is applied on the entire image. The filters can be understood to be a transformation of the image. The CNN can isolate features that are useful every-where from these transformed images and create feature maps. The second most important layer in CNN is the pooling layer. MaxPool2D is used in this case. This layer acts as a down sampling filter. It looks at the 2 neighboring pixels and picks the maximum value of the two. Pooling is used to reduce computational cost needed, and also reduce overfitting to a certain extent. One has to choose the pooling size carefully as higher the pooling dimension, greater is the down sampling.

With the combination of convolutional and pooling layers, convolutional neural networks are able to combine the locally learned features and is also able to learn more global features of the training images. Dropout is a regularization method, where a proportion of nodes in the layer are randomly ignored (setting their weights to zero) for each training sample. This way a random proportion of the network is dropped out thereby forcing the network to learn features in a certain distributed way. This technique reduces overfitting and also improves generalization greatly.

'ReLU' is the rectifier having the activation function max(0,x). The rectifier activation function is used make the network nonlinear. The 'Flatten' layer used in the model is used to convert the final feature mappings into a single 1D vector. The flattening step is needed so that one can make use of fully connected layers after some convolutional/maxpool layers. This layer also combines all the locally learned features of the previously defined convolutional layers. At the end one uses the features in two fullyconnected 'Dense' layers with 1024 filters which is nothing but the classifier needed. In this last layer '(Dense(7, activation="softmax"))' the neural network outputs distribution of probability of each class.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 75, 100, 32)       896
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 75, 100, 32)       9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 37, 50, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 37, 50, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 37, 50, 64)        18496
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 37, 50, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 18, 25, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 18, 25, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 18, 25, 128)       73856
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 18, 25, 128)       147584
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 9, 12, 128)        0
_________________________________________________________________
dropout_3 (Dropout)          (None, 9, 12, 128)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 9, 12, 256)        295168
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 9, 12, 256)        590080
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 4, 6, 256)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 4, 6, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6144)              0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              6292480
_________________________________________________________________
dropout_5 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 7175
=================================================================
Total params: 7,471,911
Trainable params: 7,471,911
Non-trainable params: 0
```

Fig. 10. Deep Learning Model view on Google Colab

Thus, a total of approximately 7.5 million trainable variables/parameters are present in the model. A variable learning rate has been used, which is randomly initialized and is reduced in value whenever the model hits a plateau state while training.

C. Results

The model is trained for 100 epochs and the model achieves 77.98% test accuracy and 77.31% validation accuracy and approximately 82% training accuracy. Which is better as compared to the results depicted in [34] which has an accuracy of 77.03%.

```
loss, accuracy = model.evaluate(x_test, y_test, verbose=1)
loss_v, accuracy_v = model.evaluate(x_validate, y_validate, verbose=1)
print("Validation: accuracy = %f  ;  loss_v = %f" % (accuracy_v, loss_v))
print("Test: accuracy = %f  ;  loss = %f" % (accuracy, loss))
model.save("model.h5")

2003/2003 [==============================] - 1s 394us/step
802/802 [==============================] - 0s 347us/step
Validation: accuracy = 0.773067  ;  loss_v = 0.618428
Test: accuracy = 0.779830  ;  loss = 0.611239
```

Fig. 11. Model Accuracy

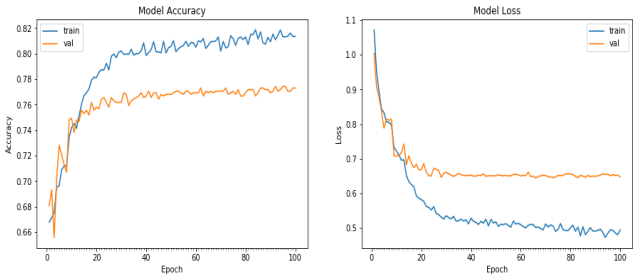

Fig. 12. Accuracies and Loses

Given below is the confusion matrix that shows the true labels vs. the predicted labels.

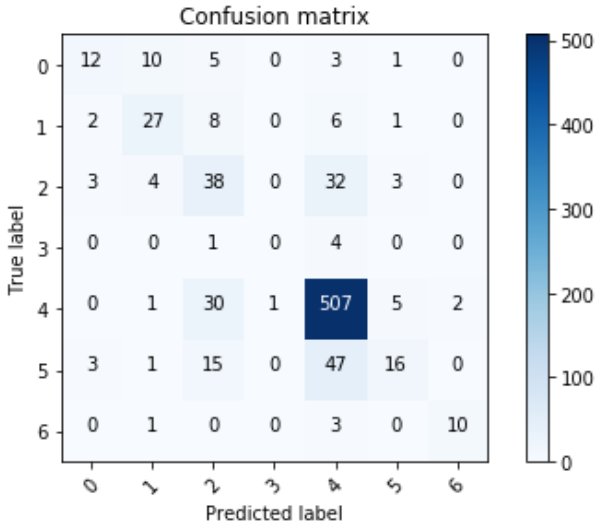

Fig. 13. Confusion Matrix

The bar graph given below depicts the fraction of classes predicted correctly.

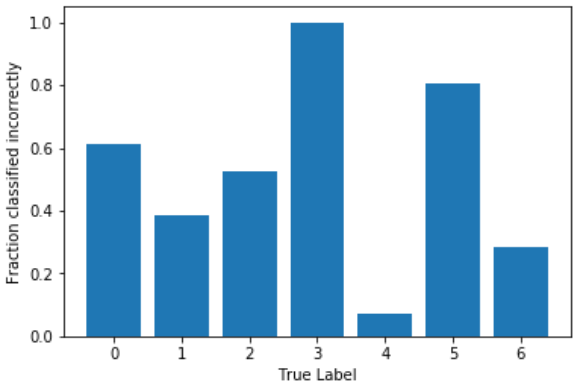

Fig. 14. Fraction of classes and their predictions

It can be seen that class 3 has 100% inaccuracy which is due to faultiness in the dataset and very less number of samples from this category are present in the dataset.

VI. TEST CASES

Different test cases are conducted and the results are explained below. When figure 15 left half is given as input to the proposed model, it shows the output as in figure 15 right half, where different probabilities are predicted for the given image, where the maximum value is at class 5, which is "Actinic keratoses".

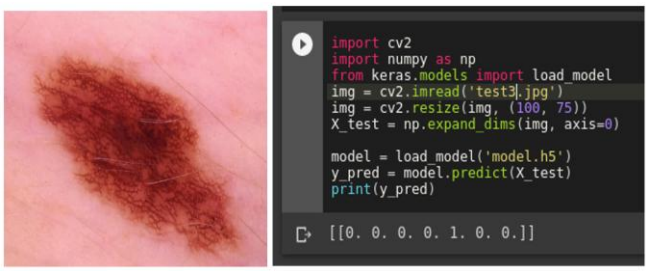

Fig. 15. Skin Cancer image and its deployment code with result prediction

VII. CONCLUSION

The Deep Learning overcomes the performance of Machine Learning as it automatically defines and selects the features. The availability of more and more training data makes it more accurate in testing. One has to be careful while deciding number of epochs and batch size for training data. The CPUs and GPUs are used for computations, where CPUs are able to do more general and complex tasks and GPU does simple and light weight tasks. GPU has a greater number of cores than the CPUs. Nonetheless if CPU and GPUs are not available then too one can employ the Google Colab framework to execute Deep Learning algorithms. Google Colab not only executes the Machine Learning calculations but also it provides dataset download facility directly from the server, it also authenticates user and able to mount the content directly from the Google drive. The usage and coding details of Google Colab are presented in this paper. Finally, the skin cancer type detection is executed using Deep Learning Approach and is improved in terms of accuracies.

REFERENCES

1. Artificial neural network. (2019, April, 5) [Online]. Available: https://en.wikipedia.org/wiki/Artificial_neural_network
2. WILDML Artificial Intelligence, Deep Learning and NLP. (2019, April, 7) [Online]. Available: http://www.wildml.com/deep-learning-glossary/
3. Fundamentals of Deep Learning – Activation Functions and When to Use Them? (2019, April, 10) [Online]. Available: https://www.analyticsvidhya.com/blog/2017/10/fundamentals-deep-learning-activation-functions-when-to-use-them/
4. The Softmax Function, Neural Net Outputs as Probabilities, and Ensemble Classifiers. (2019, April, 13) [Online]. Available: https://towardsdatascience.com/the-softmax-function-neural-net-outputs-as-probabilities-and-ensemble-classifiers-9bd94d75932
5. Activation functions and it's types-Which is better? (2019, April, 15) [Online]. Available: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f
6. Pooling: Overview. (2019, April, 18) [Online]. Available: http://deeplearning.stanford.edu/tutorial/supervised/Pooling/
7. Understanding Convolution in Deep Learning. (2019, April, 20) [Online]. Available: https://timdettmers.com/2015/03/26/convolution-deep-learning/
8. Convolutional neural network. (2019, April, 22) [Online]. Available: https://en.wikipedia.org/wiki/Convolutional_neural_network.
9. Applied Deep Learning - Part 4: Convolutional Neural Networks. (2019, April, 25) [Online]. Available: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2
10. Intuitively Understanding Convolutions for Deep Learning. (2019, April, 27) [Online]. Available: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1.

11. What is meant by fine-tuning of neural network? (2019, April, 28) [Online]. Available: https://stats.stackexchange.com/questions/331369/what-is-meant-by-fine-tuning-of-neural-network
12. Fine tuning. (2019, May, 3) [Online]. Available: http://wiki.fast.ai/index.php/Fine_tuning
13. Dropout in (Deep) Machine learning. (2019, May, 5) [Online]. Available: https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5
14. A Gentle Introduction to Dropout for Regularizing Deep Neural Networks. (2019, May, 7) [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/
15. Understanding Dropout with the Simplified Math behind it. (2019, May, 10) [Online]. Available: https://towardsdatascience.com/simplified-math-behind-dropout-in-deep-learning-6d50f3f47275
16. The Vanishing Gradient Problem. (2019, May, 15) [Online]. Available: https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484
17. A Gentle Introduction to Exploding Gradients in Neural Networks. (2019, May, 18) [Online]. Available: https://machinelearningmastery.com/exploding-gradients-in-neural-networks/
18. The curious case of the vanishing & exploding gradient. (2019, May, 20) [Online]. Available: https://medium.com/learn-love-ai/the-curious-case-of-the-vanishing-exploding-gradient-bf58ec6822eb
19. Exploding Gradient Problem. (2019, May, 22) [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/exploding-gradient-problem
20. Types of Optimization Algorithms used in Neural Networks and Ways to Optimize Gradient Descent. (2019, May, 22) [Online]. Available: https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f
21. What's the Difference Between a CPU and a GPU? (2019, May, 24) [Online]. Available: https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/
22. MTE Explains: The Difference Between a CPU and a GPU. (2019, May, 27) [Online]. Available: https://www.maketecheasier.com/difference-between-cpu-and-gpu/
23. What's the difference between a CPU and a GPU? When I switch on my computer, it shows GPU information. What does it mean? (2019, May, 28) [Online]. Available: https://www.quora.com/Whats-the-difference-between-a-CPU-and-a-GPU-When-I-switch-on-my-computer-it-shows-GPU-information-What-does-it-mean
24. Major Difference Between CPU and GPU. (2019, May, 29) [Online]. Available: https://www.techtapo.com/trendy-tech/cpu-and-gpu/
25. Difference between CPU and GPU. (2019, June, 2) [Online]. Available: https://techdifferences.com/difference-between-cpu-and-gpu.html
26. What's the difference between a CPU and a GPU? (2019, June, 5) [Online]. Available: https://www.techopedia.com/whats-the-difference-between-a-cpu-and-a-gpu/7/32875
27. Difference Between CPU and GPU. (2019, June, 10) [Online]. Available: http://www.differencebetween.net/technology/difference-between-cpu-and-gpu/
28. Understanding the differences between CPU, GPU, and APU. (2019, June, 12) [Online]. Available: https://www.windowscentral.com/difference-between-cpu-gpu-and-apu
29. CPU vs GPU in Machine Learning. (2019, June, 17) [Online]. Available: https://www.datascience.com/blog/cpu-gpu-machine-learning
30. A Comparison of Modern GPU and CPU Architectures: And the Common Convergence of Both by Jonathan Palacios and Josh Triska, March 15, 2011. https://cours.do.am/ParadigmeAgent/final.pdf
31. M. Rahmad et. Al., "Comparison of CPU and GPU implementation of computing", 2011 IEEE International Conference on Control System, Computing and Engineering, Nov. 2011, Penang, Malaysia.
32. IEEE 754 Standard Compliance: CPU vs GPU or, a War between Intel and NVIDIA. (2019, July, 5) [Online]. Available: https://software.intel.com/en-us/forums/watercooler-catchall/topic/681450
33. Welcome to Colaboratory! (2019, July, 15) [Online]. Available: https://colab.research.google.com/notebooks/welcome.ipynb#scrollTo=-Rh3-Vt9Nev9
34. Step wise Approach: CNN Model. (2019, July, 20) [Online]. Available: https://www.kaggle.com/sid321axn/step-wise-approach-cnn-model-77-0344-accuracy
35. ISIC 2018: Skin Lesion Analysis Towards Melanoma Detection. (2019, July, 25) [Online]. Available: https://challenge2018.isic-archive.com/
36. Skin Cancer MNIST: HAM10000. (2019, July, 27) [Online]. Available: https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000

AUTHORS PROFILE

Pratik Kanani. Pratik received his Bachelors and Master's Degree from University of Mumbai. He has more than 25 research contributions in International and National Journals as well as Conferences. He is an active researcher in the field of Network Security, IoT, Machine Learning and Fog Computing. Currently he is pursuing his Doctoral studies from The MS university of Baroda.

Dr. Mamta Padole is working as an Associate Professor in the Department of Computer Science and Engineering at The Maharaja Sayajirao University of Baroda. She has more than 20 years of academic and industrial experience. Her research papers are indexed in Web of Science, SCOPUS, DBLP, ACM and IEEE. Her areas of interest are Distributed Computing, Cloud Computing, Fog Computing, IoT, Big Data Analytics and Bioinformatics.