International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 04 | April - 2022 Impact Factor: 7.185 ISSN: 2582-3930

Melanoma Cancer Detection using Deep Learning

Mukul Gupta
Electronics & Communication Department
Thakur College of Engineering and Technology
Mumbai, India

Hardika Jain
Electronics & Communication Department
Thakur College of Engineering and Technology
Mumbai, India

Adwait Raut
Electronics & Communication Department
Thakur College of Engineering and Technology
Mumbai, India

Dr. Sangeeta Mishra
Electronics & CommunicationDepartment
Thakur College of Engineering and Technology
Mumbai, India

***Abstract*— this paper addresses the deep learning-based classification model that is used for classifying melanoma cancer. Skin cancer has been a wide concern of medical disease and having an efficient and robust self-learned model for helping detect them in early stages will be very useful. The model in this paper is trained on a very extensive dataset and was validated to have high accuracy for prediction. The model has also used the concept of transfer learning along with other models like VGG16, AlexNet and ResNet.**

Keywords— deep learning, transfer learning, melanoma Cancer ResNet, AlexNet, VGG16.

I. INTRODUCTION

In recent years there is been an increase in various Cancer diseases and a major increase has happened in Skin Cancer. This increase in Cancer diseases is due to unhealthy lifestyles followed by many in today's generation. Melanoma Skin Cancer is the most common skin cancer among all and it's increasing at a rapid pace. There has been a previous successful attempt to classify skin cancer by the image as the data set by using various machine learning classification algorithms like SVM (Support Vector Machine). These algorithms give high accuracy and results, but in recent years due to a vast increase in data set collection and abundance of labeled data, there has been a rise in the use of neural network architecture for classification and prediction problems. Neural nets are considered to be providing a very high accuracy compared to previous known classification algorithms. Just as there is a rise in data the concept of deep learning has started to evolve where there is a very large number of neural network layers which even helps in having better accuracy. But there is a trade-off between better accuracy with computational cost and time, Since the motivation of the new techniques is to reduce the cost and time needed for the given problems we need to find other ways to do it.

There has been a lot of work done in Deep learning for image classification problems. There is a world-renowned architecture that provides the best classification results which were trained using very powerful GPU and were found at a very high computational cost. The neural nets work in a way where they find patterns in given inputs and generalize their outputs. In image classification problems a lot of basic patterns are common among all types of images
e.g Vertical edges, horizontal edges, and if a model is already trained for finding this pattern then we need to use this pre-trained model and make modifications in the last layers as per the need of problem definition. This is called Transfer Learning.

Transfer learning is the concept where we use a pre-trained model which was used for some other problem definition and then make some fine-tuning to this architecture to use it for our problem definition. In deep learning, it's found that early layers try to find basic patterns in the data set e.g. In the image of any type the early layers of any model try to find edges which are a common task for all kinds of images. So using this pertained model parameters weightwe can change or remove the last layers (also called head layers) and run a few epochs to make a model to learn the parameters for head layers and then we can also add some classification layers (softmax layer) as per our problem statement.

In this paper, we have used the concept of transfer learning and we have to use ResNet and fine-tune it as per the problem statement. This helps in reducing the computational cost of training the model and with marginal effect on results and accuracy's

II. METHODOLOGY

AlexNet:-

AlexNet proposed by Krizhevsky et al., where this net possesses many essential characteristics, such as the following:
1. There are many more filters with every layer, which can

enhance features and reduce noise.
2. Each stacked convolution layer is followed by a pooling layer which can reduce the number of features and extract essential features only.
3. The likelihood of gradient vanishing is reduced because of using RELU as activation function instead of tanh, logistic, Arctan, or Sigmoid, which is more biological inspired.
4. From the previous three points, the AlexNet is five times faster than other deep architectures.
5. Some deep architecture required specific hardware; AlexNet can work well with hardware and GPU limitations.

Deep architectures contain more than a hidden layer. These hidden layers help to enhance and extract features in a better way. AlexNet is a big network consisting of several neurons equal to 650,000 and 60 million parameters. The activation function was the first improvement.

For nonlinearity in the classical neural network, the activation function is limited to arctan, tanh, logistic function, etc. The gradient values using these activation functions will be significant only when the input is around a small range of 0, so these types of activations functions will fall into the problem of gradient vanishing. A new activation function called a rectified linear unit (ReLU) had been used to overcome this problem. If the input is not less than 0, the gradient of RELU is always 1. Also, it accelerates the training process. Features are extracted automatically by the convolutional layers and reduced by the pooling layer. The model will be able to learn from image features by convolution, and these parameters are shared to reduce the complexity of the model. The pooling layers are used to reduce the extracted features. The pooling layers take a group of neighboring pixels from the feature map and generate values for representation. Max pooling is used in AlexNet to reduce the feature map. The classification was done in the fully connected layers. Softmax was used in the fully connected layers as an activation function.

In our case, we have taken 300 images consisting of both melanoma and not melanoma. Using ImageDataGenerator() function the images have been sorted directly into train data and validation data. These images are fed to the model and passed through 5 convolutional layers and 3 fully connected layers and in the end for decision making through a softmax layer. The optimizer used in the case of gradient descent here is SGD(Stochastic Gradient Descent) which decreases its learning rate gradually as we approach the optimum. In our case total, trainable parameters in a network are 62,407,486. Based on experience and to cover all 300 images we have taken 40 steps per epoch and 5 epochs.

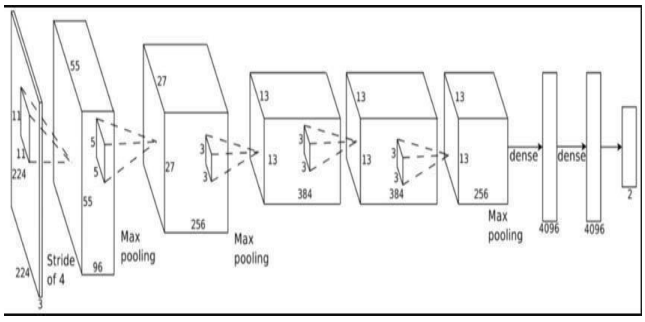


Figure 1: Alex Net Architecture

VGG16:

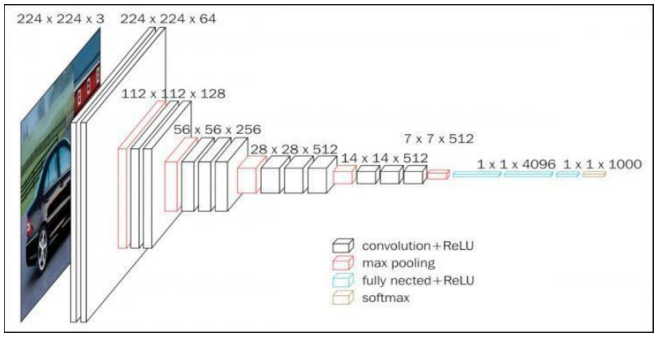
The VGG16 network is a convolutional neural network model that comprises 16 convolutional layers with a 3x3filter-size, five max-pooling layers with a 2x2 window size. A heap of convolutional layers is followed by three fully connected layers. It consists of five convolutional blocks; the first two blocks have two convolutional layers and one max- pooling layer. The remaining three blocks have three convolutional layers and one max-pooling layer. The Input of the first convolutional block is 240x240x3. The output of the first block is 120x120 x 64 that is 64 channels. This feature map is fed into the input of the second block and so on. For the coming blocks, the output is reduced to half the size and the channel size is doubled. The final feature map was produced by the fifth block of size 7x7x512. The fully connected classifier has 3 layers and the first fully connected

(Dense) layer have 4096 outputs, the second fully connected (Dense) layer have 1000 outputs, the third (Dense) layer performs classification and it contains 2 output.

Major drawbacks with VGGNet:

1) It is painfully slow to train.
2) The network architecture weights themselves are quite large (in terms of disk/bandwidth).

Figure 2: VGG 16 Architecture



	type	description	r. size		type	description	r. size
1	Conv	64;3x3;p=1,st=1	212	20	Conv	512;3x3;p=1,st=1	20
2	ReLU		210	21	ReLU		18
3	Conv	64;3x3;p=1,st=1	210	22	Conv	512;3x3;p=1,st=1	18
4	ReLU		208	23	ReLU		16
5	Pool	2x2;st=2	208	24	Pool	2x2;st=2	16
6	Conv	128;3x3;p=1,st=1	104	25	Conv	512;3x3;p=1,st=1	8
7	ReLU		102	26	ReLU		6
8	Conv	128;3x3;p=1,st=1	102	27	Conv	512;3x3;p=1,st=1	6
9	ReLU		100	28	ReLU		4
10	Pool	2x2;st=2	100	29	Conv	512;3x3;p=1,st=1	4
11	Conv	256;3x3;p=1,st=1	50	30	ReLU		2
12	ReLU		48	31	Pool		2
13	Conv	256;3x3;p=1,st=1	48	32	FC	(7x7x512)x4096	1
14	ReLU		46	33	ReLU		
15	Conv	256;3x3;p=1,st=1	46	34	Drop	0.5	
16	ReLU		44	35	FC	4096x4096	
17	Pool	2x2;st=2	44	36	ReLU		
18	Conv	512;3x3;p=1,st=1	22	37	Drop	0.5	
19	ReLU		20	38	FC	4096x1000	
				39	σ	(softmax layer)	

Table 1: VGG Net Architecture

In our case, we have taken 300 images consisting of both melanoma and not melanoma. Using ImageDataGenerator() function the images have been sorted directly into train data and validation data. These images are fed to the model and passed through 12 convolutional layers and 3 fully connected layers and in the end for decision making through a softmax layer. The optimizer used in the case of gradient descent here is SGD(Stochastic Gradient Descent) which decreases its learning rate gradually as we approach the optimum. In our case total, trainable parameters in the network are 83,937,090. Based on experience and to cover all 300 images we have taken 40 steps per epoch and 5 epochs.

ResNet:

After the celebrated victory of AlexNet at the ILSVRC2012 classification contest, deep Residual Network was arguably the most groundbreaking work in the computer vision/deep learning community in the last few years. ResNet makes it possible to train up to hundreds or even thousands of layers and still achieves compelling performance. Thanks to this technique they were able to train a network with 152 layers while still having lower complexity than VGGNet. It achieves a top-5 error rate of 3.57% which beats human- level performance on this dataset.

To solve the problem of the vanishing/exploding gradient, this architecture introduced the concept called Residual Network. In this network, we use a technique called ***skip connections***. The skip connection skips training from a few layers and connects directly to the output. The approach behind this network is insteadof layers learning the underlying mapping, we allow the network to fit the residual mapping. So, instead of say H(x), initial map- ping, let the network fit, $F(x) := H(x) - x$ which gives $H(x) := F(x) + x$

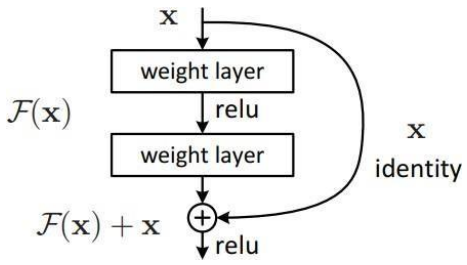


Figure 3: Skip Connection in Residual Network

The advantage of adding this type of skip connection is be-cause if any layer hurt the performance of architecture then it will be skipped by regularization. So, this results in training a very deep neural network without the problems caused by vanishing/exploding gradient. There is a similar approach called "highway networks", these networks also uses skip connection. Similar to LSTM these skip connections also use parametric gates. These gates determine how much information passes through the skip connection. This architecture however has not provided accuracy better than ResNet archi tecture. This network uses a 34-layer plain network architecture inspired by VGG-19 in which then the shortcut connection is added. These shortcut connections then convert the architecture into a residual network.

In this section, we aim to better understand the philosophy behind ResNet with Feynman Path-Integral. For a certain ResNet, feature maps at layer are represented as ut. First of all, the update process between ut−1 and ut is considered. In aresidual block, assuming f is the skip connection weight and w is the convolution kernel, ut−1 can be transformed into ut as follows,

$$u(xt) = \text{relu}\{f \cdot u(xt-1) + w(x) * u(xt-1)\}$$

$$u(xt) = X\, xt-1\, \kappa \circ f(\delta(xt - xt-1) + \Omega(xt\ \ xt-1))u(xt-1)$$

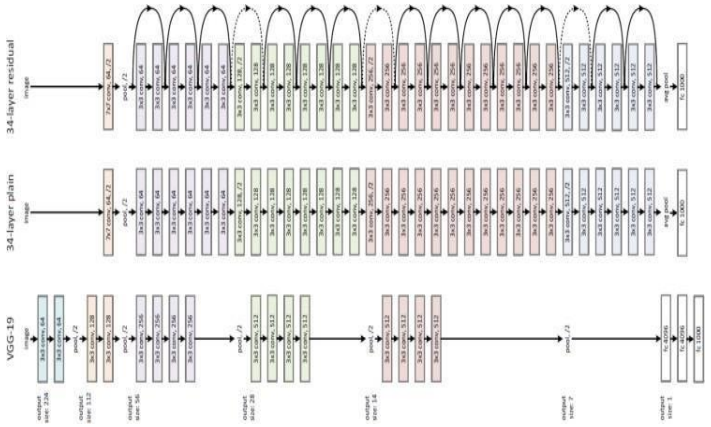
$$\delta(x_t - x_{t-1}) = \frac{1}{M}\sum_{k=0}^{M-1} e^{i\frac{2\pi}{M}k(x_t - x_{t-1})} = \sum_{p_{t-1}} e^{ip_{t-1}(x_t - x_{t-1})}$$

Deep convolutional neural networks have led to a series of breakthroughs for image classification.
Many other visual recognition tasks have also greatly benefited from very deep models. So, over the years there is a trend to go deeper, to solve more complex tasks and to also increase /improve the classification/recognition accuracy.

But, as we go deeper; the training of neural networks becomes difficult and also the accuracy starts saturating and then degrades also. Residual Learning tries to solve both these problems.

In general, in a deep convolutional neural network, several layers are stacked and are trained to the task at hand. The network learns several low/mid/high-level features at the end of its layers. In residual learning, instead of trying to learn some features, we try to learn some residual. Residual can be simply understood as subtraction of features learned from the input of that layer. ResNet does these using shortcut connections (directly connecting the input of nth layer to some (n+x) layer. It has proved that training this form of network is easier than training simple deep convolutional neural networks and also the problem of degrading accuracy is resolved.

Figure 4: ResNet Architecture



Transfer Learning:

In the classic supervised learning scenario of machine learning, if we intend to train a model for some task and domain A, we assume that we are provided with labelled data for the same task and domain. We can now train a model A on this dataset and expect it to perform well on unseen data of the same task and domain. On another occasion, when given data for some other task or domain B, we require again labelled data of the same task or domain that we can use to train a new model B so that we can expect it to perform well on this data. The traditional supervised learning paradigm breaks down when we do not have sufficient labelled data for the task or domain we care about to train a reliable model. If we want to train a model to detect pedestrians on night-time images, we could apply a model that has been trained on a similar domain, e.g. on day- time images. In practice, however, we often experience deterioration or collapse in performance as the model has inherited the bias of its training data and does not know how to generalize to the new domain. If we want to train a model to perform a new task, such as detecting bicyclists, we cannot even reuse an existing model, as the labels between the tasks differ. Transfer learning allows us to deal with these scenarios by leveraging the already existing labelled data of some related task or domain. We try to store this knowledge gained in solving the source task in the source domain and apply it to our problem of interest.

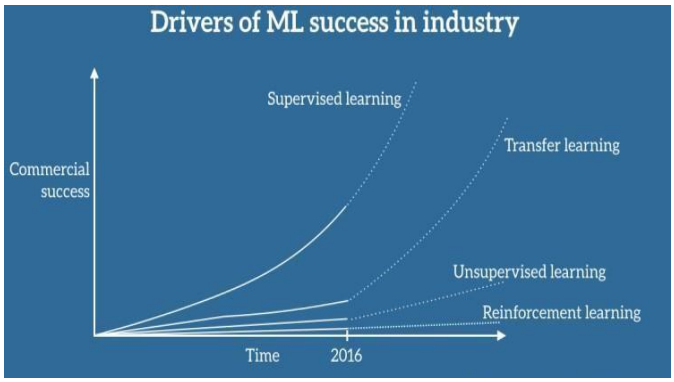


Figure 5: Transfer learning commercial success

Transfer learning involves the concepts of a domain and a task. A domain D consists of a feature space X and a marginal probability distribution P(X) over the feature space, where X=x1,⋯,xn∈X. For document classification with a bag-of-words representation, X is the space of all document representations, xi is the I-th term vector corresponding to some document and X is the sample of documents used for training. Given a domain, D={X, P(X)}, a task T consists of a label space Y and a conditional probability distribution P(Y|X) that is typically and yi is either *True* or *False*. Given a source domain DS, a corresponding source task TS, as well as a target domain DT and a target task TT, the objective of transfer learning now is to enable us to learn the target conditional probability distribution P(YT|XT) in DT with the information gained from DS and TS where DS≠DT or TS≠TT. In most cases, a limited number of labelled target examples, which is exponentially smaller than the number of labelled source examples are assumed to be available.

Given source and target domains DS and DT where D={X, P(X)} and source and target tasks TS and TT where T={Y, P(Y|X) source and target conditions can vary in four ways, which we will illustrate in the following again using our document classification example:

1. XS≠XT

The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages. In the context of natural language processing, this is generallyreferred to as cross-lingual adaptation.

2. P(XS)≠P(XT)

The marginal probability distributions of source and target domain are different, e.g. the documents discuss differenttopics. This scenario is generally known as domain adaptation.

3. YS≠YT

The label spaces between the two tasks are different, e.g. documents need to be assigned different labels in the target task. In practice, this scenario usually occurs with scenario 4, as it is extremely rare for two different tasks to have different label spaces, but the same conditional probability distributions.

4. P(YS|XS)≠P(YT|XT).

The conditional probability distributions of the source and target tasks are different, e.g. source and target documents are unbalanced with regard to their classes. This scenario is quite common in practice and approaches such as over-sampling, under-sampling, or SMOTE [7] are widely used.

III .EXPERIMENTAL RESULTS

Dataset:

Experiments are achieved to evaluate the method on a public challenge dataset of Melanoma Cancer detection released on kaggel.com. The challenge consists of three parts Part 1, part 2 taken from part1 as training images for segmentation with ground truth. Another 100 images from part 2 are held out as a cross- validation dataset for segmentation. From the 300 out as cross-validation data to predict the result. Later the same isimplemented with a total of 17000 images with the same concept i.e training, cross-validation and test images. Later a total of 17000 images with a break up of 10692, 3596 and 3596 images for training, cross-validation and testing are utilized. This complete dataset is utilized for ResNet-50.

Further a new dataset was used sourced form kaggle.com as well. This dataset consists of a total of 2750 images. These are further divided into majorly 3 different classes 1. Seborrheic keratosis, 2. Nevus, 3. Melanoma. This dataset is divided into train, validation and test, with about 2000 images in training, 600 images for validation and 150 images for test purposes.

Performance Metrics:

For performance measurement, the proposed work is evaluated using Accuracy, Sensitivity and Specificity. These metrics are defined in terms of the number of True Positives (TP), True Negatives (TN), False Negatives (FN) and False Positives (FP). True Positive(TP) means Specificity shows that the non-disease patient has a negative result that is the percentage of non-melanoma people who are identified exactly as healthy. Accuracy shows the possibility of the correct determination that is the percentage of patients and healthy people who are diagnosed correctly.

$$TPR = \frac{TP}{Actual\ Positive} = \frac{TP}{TP + FN}$$

$$FNR = \frac{FN}{Actual\ Positive} = \frac{FN}{TP + FN}$$

$$TNR = \frac{TN}{Actual\ Negative} = \frac{TN}{TN + FP}$$

$$FPR = \frac{FP}{Actual\ Negative} = \frac{FP}{TN + FP}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

ROC Curve

A ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate, False Positive Rate

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1)
AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpretingAUC is as the probability that the model ranks a random positive example more highly than a random negative example.
AUC is desirable for the following two reasons: AUC is **scale-invariant**. It measures how well predictions are ranked, rather than their absolute values. AUC is **classification-threshold-invariant**. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

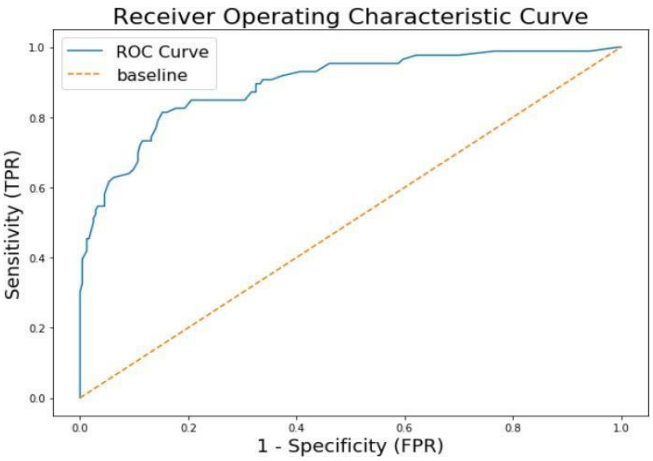
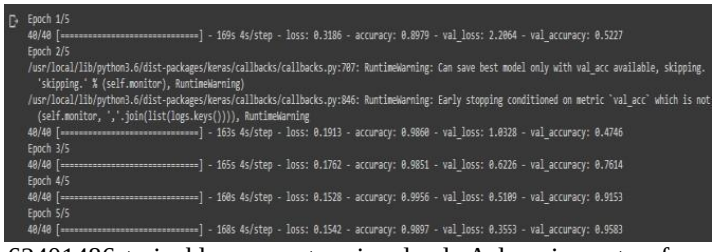


Figure 6: ROC Curve

AlexNet:

For the implementation of Alex-Net, we have used input dimensions of 175x175x3. This is applied to the first layer with akernel size of 11x11 and stride size of 4x4 and max-pooling of the “same” category. For the second layer, a kernel size of 5x5 and stride size of 1x1 with 256 filters is used. The third layer uses 384 filters and kernel of 3x3 and stride of 1x1. The fourth layer is equivalent to the third layer. The fifth layer is identical to the second layer. The rest of the layers are fully connected layers. All the layers except to last layer use “RELU” activation. An important feature of the AlexNet is th Unit) Nonlinearity. Tanh or sigmoid activation functions used to be the usual way to train a neural network model. AlexNet showed that using ReLU nonlinearity, deep CNNs could be trained much faster than using the saturating activation functions like tanh or sigmoid. The figure below from the paper shows that using ReLUs(solid curve), AlexNet could achieve a 25% training error rate six times faster than an equivalent network using tanh(dotted curve). This was tested on the CIFAR-10 dataset.

The last layer uses “SIGMOID” activation. There is a total of



62401486 trainable parameters involved. A learning rate of 0.001 is used in the optimizer. This network is implemented using Stochastic Gradient Descent (SGD)optimizer. After trying the network on this dataset we have learned to set for 40 steps per epoch and in a total of 5 epochs. For this setting, we have achieved final training accuracy of 98.97% and a cross- validation accuracy of 95.83%.

VGG16:

For the implementation of VGG-16, we have used input dimensions 175x175x3. This is applied to the first layer with a kernel size of 3x3 and 64 filters. For the second layer, similar parameters are set. The third layer is of 128 filters and the same kernel size. Then we consistently increase the filter sizes with every two layers to 256 to 512 and so on. Lastly, the last 3 layers are fully connected. All the layers except the last layer use the "RELU" activation function. The last layer is "SIGMOID" activation. A total of 83937090 trainable parameters were involved. A learning rate of 0.001 is used in the optimizer. This network is implemented using Stochastic Gradient Descent (SGD) optimizer. With this configuration we were able to achieve an accuracy of 94.3% and cross-validation accuracy of 93.00%.

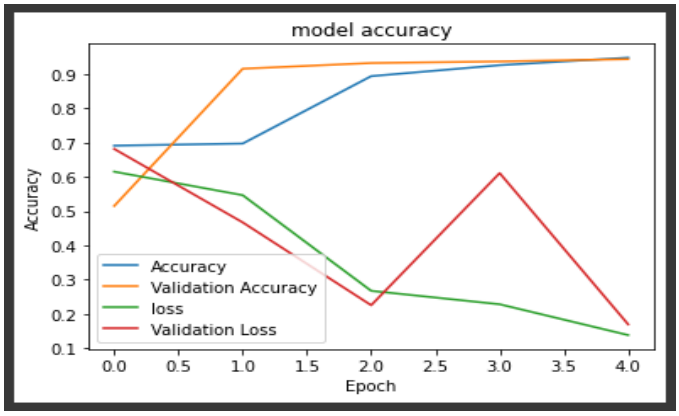
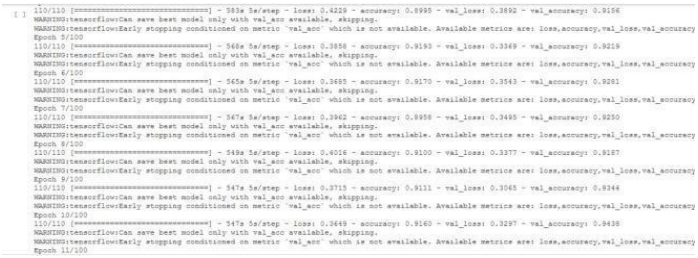


Figure 8: VGG 16 Output

ResNet-50:

For the implementation of ResNet, we have used input dimensions of 175x175x3. For this architecture, the concept of transfer learning is used. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Here a pre-trained model is import- ed and based upon the features learned the parameters are already. Initially, we had trained the model using no hardware accelerators and as google, colab has runtime restrictions we could not complete the model training but up until the 42 epoch, we had attainedan accuracy of 94%.

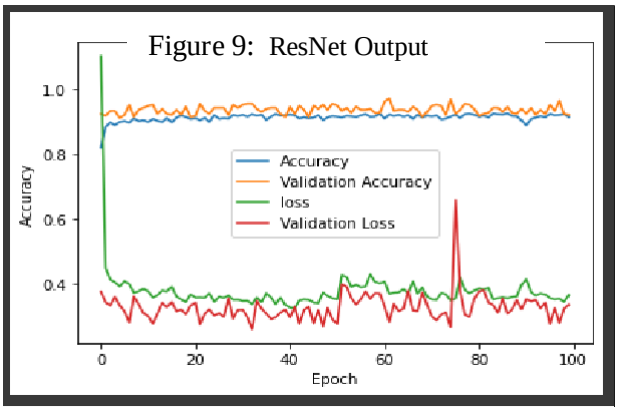


Later to counter the exponential amount of time required we opted to use GPU as a hardware accelerator. Now when using GPU our model was able to train faster but it had its tradeoffs.

1) **TPU:** The tensor Processing Unit is highly optimized for large batches and CNNs and has the highest training throughput.
2) **GPU:** Graphics Processing Unit shows better flexibility and programmability for irregular computations, such as small batches and non MatMul computations.
3) **CPU:** Central Processing Unit achieves the highest FLOPS utilization for RNNs and supports the largest model because of its large memory capacity.

On this GPU our model was able to train faster but suffered at accuracy and after 100 epochs we were able to get an accuracy of 92%. Here point to be noted is that the model went into saturation at 10 epoch itself and from then later the accuracy was in the range of 89-92%.

Figure 9: ResNet Output



ResNet-50 for updated dataset:

Now the same model was implemented on the new dataset by making some minor changes and tweaks to the previous code. For the new dataset the input shape of the images was 224x224x3 and For this architecture, the concept of transfer learning is used. Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Here a pre-trained model is imported and based upon the features learned the parameters are already. After importing the model and its pre-trained features, additional 2 dense layers were added with "relu" activation function and connected via a drop out layer with a rate of 0.2. Finally for output another dense layer was used with "softmax" as an activation function. Later on this model was compiled with losses as "categorical_crossentropy" and "Adam " optimizer, keeping learning rate as low as "0.001". This model was trained with 5 epochs and the length of training as a parameter for steps per epoch. Finally the training accuracy found was up to 91% and the testing accuracy was found to be 68%.

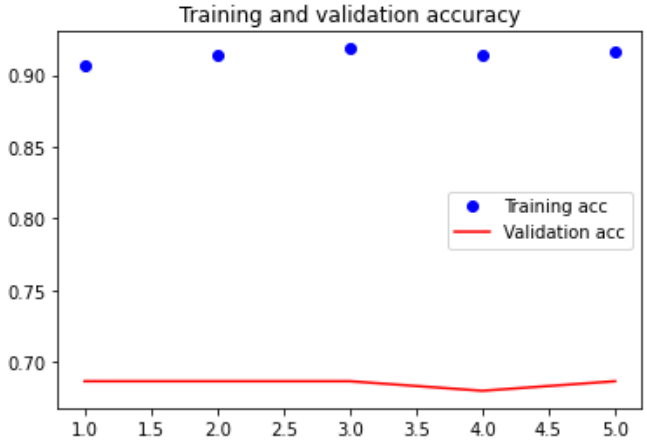


Figure 10: ResNet Output on revised dataset

IV .CONCLUSION

In this paper, a Deep Neural Net and CNN based segmentation and classification to achieve the challenges of automated melanoma classification in dermoscopy images, which consists of three steps: Segmentation, Feature Extraction and Classification. Here we had used AlexNet, VGG16 and ResNet-50 architectures and can compare the results obtained. AlexNet being more versatile and best suited for classification purposes gave the best result compared to VGG16.VGG16 even though after having more layers and a deep network could not deliver outstanding results because this network is best suited for image segmentation purposes. Also, the training time taken by each network affect its performance, AlexNet was the one that could be trained much faster and VGG16 owing to its size was much harder to train. Overall AlexNet proved to be a better architecture and choice when it comes to melanoma cancer detection. But with the introduction of ResNet-50 the whole game was changed and with an even deeper network and different parameter learning methodology we were able to achieve very high accuracy on a very large dataset.

V .FUTURE SCOPES

AlexNet even though has proved to be better than VGG16 but it still is very expensive to compute on a large dataset. With the use of complex Neural Net algorithms like ResNet-50, we have achieved high accuracy through deeper networks without trading off training time. So the next step could be with better pre-processing and use of Google's vision API on the google cloud platform we can achieve higher accuracy and also compare the accuracy and losses and overall feasibility of the whole model and according to the set benchmarks can improve our existing models. Also, this fully developed model can be deployed on a cloud platform thus making it available for service 24 by 7 and eliminating any hardware dependency.

VI .REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

[2] Khalid M. Hosny, Mohamed A. Kassem, Mohamed M. Fouad. "Classification of Skin Lesions into Seven Classes Using Transfer Learning with AlexNet" , Journal of Digital Imaging, 2020

[3] theintelligenceofinformation.wordpress.com
[4] iq.opengenus.org
[5] Željko Vujović. "A case study of the application of WEKA software to solve the problem of liver inflamation" , Research Square Platform LLC, 2021

[6] developers.google.com
[7] Sachin Mungmode, R.R. Sedamkar, Niranjan Kulkarni. "A Modified High Frequency Adaptive Security Approach using Steganography for Region Selection based on Threshold Value" , Procedia Computer Science, 2016

[8] Ju Zhang, Lundun Yu, Decheng Chen, Weidong Pan, Chao Shi, Yan Niu, Xinwei Yao, Xiaobin Xu, Yun Cheng. "Dense GAN and Multi-layer Attention based Lesion Segmentation Method for COVID-19 CT Images" , Biomedical Signal Processing and Control, 2021

[9] Sarah Sheikh, Uvais Qidwai. "Chapter 35 Smartphone-Based Diabetic Retinopathy Severity Classification Using Convolution Neural Networks" , Springer Science and Business Media LLC, 2021

[10] Saurav Vinod, Manoj V. Thomas. "A Comparative Analysis on Deep Learning Techniques for Skin Cancer Detection and Skin Lesion Segmentation" , 2021 International Conference on Communication, Control and Information Sciences (ICCISc), 2021

[11] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[12] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[13] K. Elissa, “Title of paper if known,” unpublished.

[14] R. Nicole, “Title of paper with the only first word capitalized,” J. Name Stand. Abbrev., in press.

[15] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-opticalmedia and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[16] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.