SKIN LESION CLASSIFICATION

CHAUDHARY Mudit

SID: 1155085166

BYUN Jiyeon

SID: 1155086596

Supervisor: Prof. K.S. Leung

TA Supervisor: Mr. FU Xinyu

Abstract

The tasks of Image recognition and classification have been greatly improved because of the emergence of deep learning methods. We can use these deep learning methods in medical image analysis so that the doctors can take better and expeditious decisions. In this paper we develop ways to classify Melanoma and Seborrheic Keratosis using Deep Residual Networks. We have been getting accuracy of 78.67% for Melanoma classification.

Keywords

Skin Lesion, Machine Learning, Deep Learning, Convolutional Neural Networks, Deep Residual Network, Medical Image Classification

1. Introduction

Melanoma, which is derived from cells called melanocytes, is the most malignant form of skin tumor. In the United States, the estimated new cases of melanoma were more than 76,000, and estimated deaths from melanoma were over 10,000 by 2016 [1]. Even though it is very fatal as it spreads out exponentially, melanoma can be treated under control if it is detected early.

However, the diagnosis of melanoma might not be so accurate: the specially trained dermatologists examine the removed tissue from the patients under microscope and take other factors such as family history, immune system or physical characteristics into consideration [2]. Statistically, it is reported that clinical examination results in accuracy of 67% and dermoscopic examinations diagnoses melanoma with accuracy of 80% [3].

What is more severe problem than undiagnosed case of melanoma is misdiagnosed case of melanoma to other benign skin lesion. For this reason, computer-based assistance on diagnosis would be a big help for both the patients and physicians in skin lesion classification.

In this project, the first task is to classify melanoma from other two benign skin tumors called nevus and seborrheic keratosis. The second task is to classify seborrheic keratosis, which is non-melanocytic benign skin tumor, from the other two types of skin tumor.

In this paper, skin lesion is classified by implementing deep neural network. The model uses deep residual network (ResNet) architecture. Two different methods are used to train the model: (i) training ResNet from the scratch; (ii) fine-tuning ResNet. The paper will cover project details such as proposed solution, data pre-processing and data augmentation, experiment and results, and discussion on how the model can be implemented further.

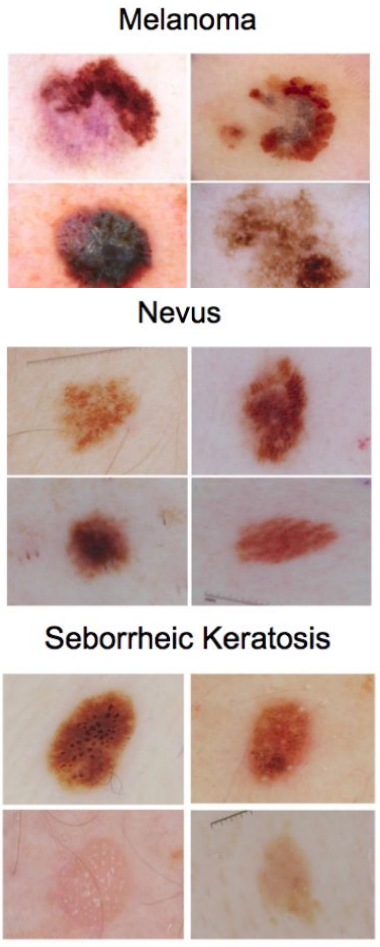

Figure 1. Images for Melanoma, Nevus and Seborrheic Keratosis

2. Dataset and Challenges

All the dataset used in this project were provided by International Skin Imaging Collaboration (ISIC). ISIC puts effort for melanoma diagnosis internationally and contains the largest collection of dermoscopic skin lesion images that is publicly accessible. All the images in ISIC archive are assessed and labeled by melanoma experts.

For the training image data of the project, among 2000 images, 374 images were labeled melanoma, 254 images were labeled seborrheic keratosis and 1372 images were nevus. The images were provided in JPEG format along with CSV file, which contains labels. For validation data, 150 images were provided, and 600 images were provided for test data.

There were a few challenges faced with the provided dataset. First challenge is that the provided dataset was very small. Small dataset can be problematic as it can incur overfitting, higher risk on measurement error and missing values. Second challenge is that the dataset is unbalanced. As given dataset is rather small, this biased dataset can become worse problem as the model tends to misclassify the underrepresented category.

3. Project Details

3.1 Proposed Solutions

As the task is an image classification problem, we plan to use Convolutional Networks because it fits our problem the best. In this section, we describe the selected Convolutional Networks architecture and discuss why we selected this architecture and its implementation aspects.

3.1.1 Deep Residual Network (ResNet)

We took two architectures into consideration VGG-16 and ResNet. ResNet showed much better results in ILSVRC & COCO 2015 Image Recognition competitions [4], so we plan to implement ResNet-50 and ResNet-101. We will then compare the performance of aforementioned ResNet(s). ResNet is a well-documented and highly used ConvNet architecture for image recognition tasks. We will be implementing the ResNet first proposed in the paper- Deep Residual Learning for Image Recognition [4]. The reason why ResNet perform better is that they have ability to deal with vanishing gradient problem. According to the **Universal Approximation Theorem** in the mathematical theory of artificial neural networks, a feed-forward network with a single layer can approximate any continuous function on compact subsets of Euclidean space. However, using a single layer is prone to overfitting of the data. Therefore, we try to stack more layers so as to prevent overfitting to better generalize the function. Increasing the network depth solves that problem but gives rise to a new problem: vanishing gradient. Vanishing gradient problem is observed because, as the gradient is back-propagated to earlier layers, repeated multiplications from many layers makes the gradient infinitesimally small. Consequently, the performance saturates in deeper networks and, in some cases, degrades. To solve this problem, Residual Networks (ResNet) introduces 'identity shortcut connections'. These shortcut connection can skip one or more layers thus providing a skip-connection for activation function from a shallower layer to a deeper layer. A residual block as represented in the paper is shown in Figure 2.

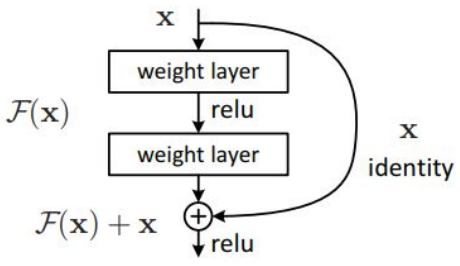

Figure 2. Residual Block

The identity shortcuts in the ResNet has to take into account the dimensionality changes between the output of shallow layer and the input of the deeper layer. For dealing with this complication, ResNet introduces two different identity mappings.

If the input and output of the layers connected by the skip-connection are of the same dimensions, the following identity mapping is used:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$$

If the input and output of the layers connected by the skip-connection are of different dimensions, the skip-connection performs a linear projection W_s to match the dimensions. The following identity mapping is used:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$$

3.1.2 Design considerations

For ease of implementation purposes we turn our two independent binary classification problems to two independent categorical classification problems with two categories each.

The original ResNet needs to be modified for our needs. As we are performing a categorical classification with just two classes, we modify the last layer to a fully connected layer with two nodes rather than a fully-connected layer with 1000 nodes as proposed in "Deep Residual Learning for Image Recognition" [4]. The last two nodes output the probabilities of the image being of category 1 or category 2, where category 1 & 2 are just categorical counterparts of binary classification.

The modified the ResNet-34 is shown in Figure 3. The remaining ResNet-50 and ResNet-101 will have the same modification in the last classifying fully connected layer.

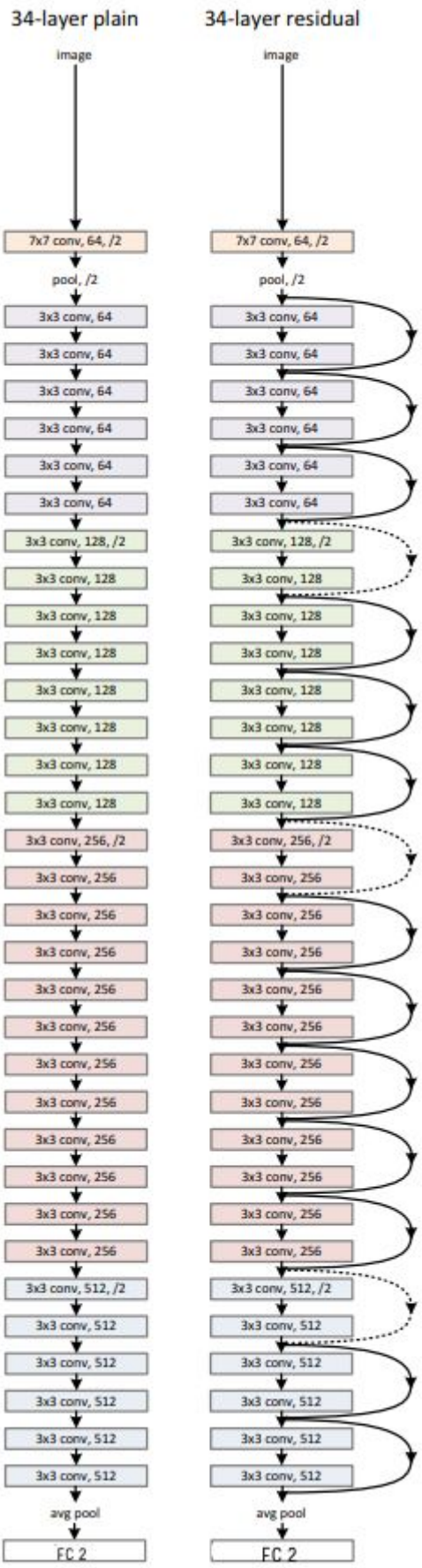

Figure 3. Modified ResNet-34

3.2 Technologies Used

All the neural network implementation is done using Keras high-level API with Tensorflow backend [5][6]. We use Keras because of its modularity and user-friendliness. For data-augmentation we use *imgaug* python library by Alexander Jung. The auxiliary functions like plotting graph are coded in python.

3.3 Data Pre-Processing

We pre-process the data before feeding it into the network for performance and ease-of-use purposes.

As we are using Tensorflow as our deep learning library, we convert the image data into native Tensorflow format i.e. TFRecords.

The images are also converted to 224 x 224 px size. We don't normalize the pixel values [0,1] as pre-trained ResNet on ImageNet dataset uses pixel values [0-255]. For Fine-Tuning we also subtract the mean image from Imagenet dataset i.e. [103.939, 116.779, 123.68].

TFRecords is Tensorflow's own binary storage format. TFRecords take much less space on the disk as compared to raw image data. This eases data sharing. The comparison for the size of datasets is shown below:

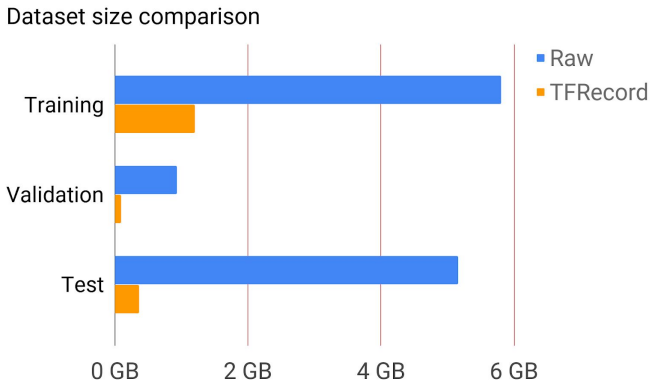

Figure 4. Comparison of Dataset sizes Raw vs TFRecords

Binary files take less time to copy and can be read much more efficiently from the disk. Moreover, it offers us to use **tf.data** API using which we can highly optimize our input pipeline. We can use **tf.data.Dataset.map** transformation. Because input elements are independent of one another, the pre-processing can be parallelized across multiple CPU

cores. We parallelize the data input pre-processing into 8 parallel calls which highly optimizes our input pipeline, hence decreasing the training time.

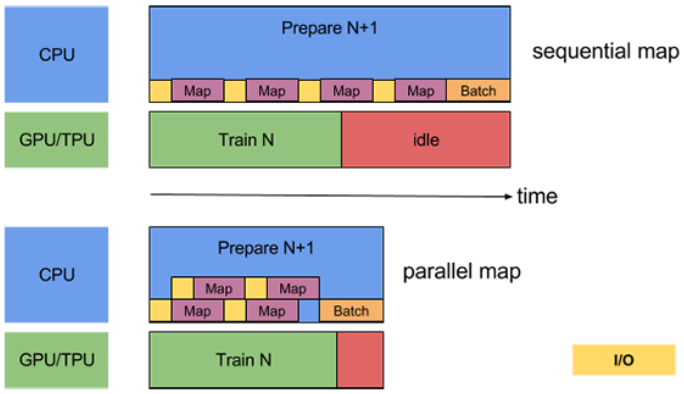

Figure 5. Parallelizing input data pre-processing

3.4 Data Augmentation

The provided image dataset is relatively small to image datasets provided for CIFAR-10 and Imagenet. So, we apply data augmentation techniques to increase the performance and accuracy of our neural network model.

We apply the data augmentation techniques provided in the paper- Data Augmentation for Skin Lesion Analysis [7].

We make some minor changes to the data augmentation techniques in the aforementioned paper. We apply the following data augmentation sequentially using the ***imgaug*** Python image augmentation library by Alexander Jung.

Table 1.Slight modifications in the techniques mentioned in [7]

Step	Name	Description
1	Image Crop	Cropping the image 0-20% from all sides.
2	Affine	Rotate the image by upto 90 degrees, shear by upto 20 degrees and scale the area by x = 0.8 and y = 1.2. The pixels are filled symmetrically at the edges.
3	Flip	Randomly flip image vertically.
4	Saturation, Brightness, Contrast and Hue	Change brightness and contrast by a factor which is a random value between [0.7 , 1.3]. Add to hue and saturation with a random value between [-0.1 , 0.1].

Some examples of the data augmentation applied on images are below:

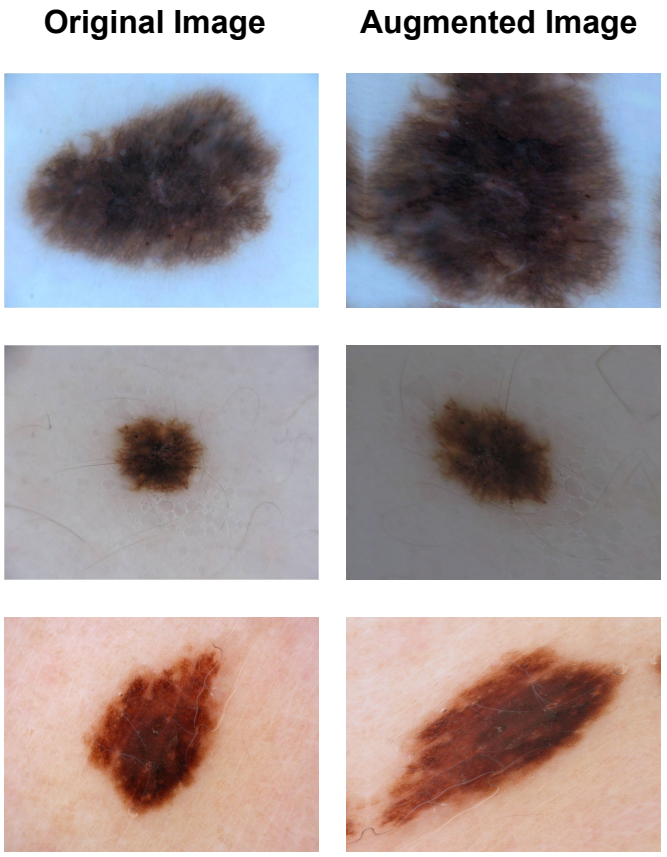

Figure 6. Before and after data augmentation

All the data-augmentation is done in the process to create TFRecord which makes it really easy to manipulate the images before adding them as features to .tfrecords file.

3.5 Methods Of Training

We train the modified ResNet using two different methods: (i) Training from scratch; ii) Fine-Tuning ResNet with pre-trained weights on ImageNet Dataset.

i) Training from scratch

We train the ResNet-50 and ResNet-101 from scratch without using any pre-trained weights. This training method is time consuming as we are training all the layers of the ResNet. This also poses a challenge because our dataset is not large enough, hence its performance should be relatively poor.

ii) Fine-Tuning ResNet

Keras provides a ResNet-50 model with pre-trained weights on ImageNet dataset. We fine-tune this model by freezing some of the residual blocks. We only fine-tune the higher-level portion of the network because the lower levels contain more generic features while the higher-level portion learn more specific features about the image dataset. It will reduce our training time and should give better results.

4. Experiment and Results

This section discusses the experiments, how the experiments were performed and the results we got.

4.1 Balancing the Dataset

The provided dataset is unbalanced with 374 “melanoma”, 254 “seborrheic keratosis” and 1372 “nevus”. We balance the data slightly by adding more positives using data augmentation techniques.

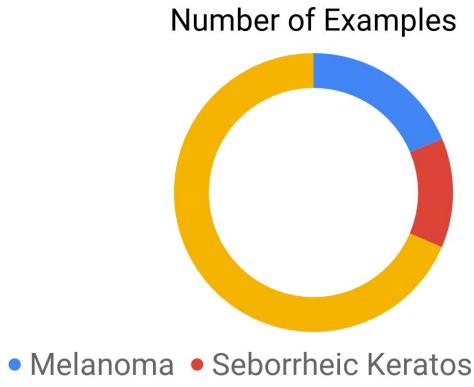

Figure 7. Comparison between examples in different categories

We create 2 datasets i) Melanoma vs. Others ii) Seborrheic Keratosis vs Others. We first apply data augmentation on the whole dataset thus doubling the examples and then we add positive examples for the corresponding classifications.

After applying data augmentation techniques and adding more positives, our final datasets are as follows.

Table 2. Number of images in each categories

Melanoma vs. Others		**Seb. Ker. vs Others**	
Melanoma	1122	Melanoma	748
Seb. Ker.	508	Seb. Ker.	762
Nevus	2744	Nevus	2744

The dataset is still highly unbalanced and adding more positives using data augmentation will reduce the performance of our network. So, we implemented a **Weighted Categorical Cross-Entropy** loss function function. The problem is a medical problem so we want the predictions to have high sensitivity (Recall). Therefore, we increase the cost of misclassifying Melanoma or Seborrheic Keratosis as others.

4.2 Parameters

For the optimizing function, we use Stochastic Gradient Descent optimizer. For our Method 2- Fine Tuning of ResNet, we use a learning rate of 0.0001 so that we don’t distort the pre-trained weights quickly. We also implement a **Learning Rate Scheduler**, which decays the learning rate by a decay factor of 0.1 after 20 epochs and then with a decay factor of 0.1 after every 10 epochs.

Fine-Tuning Parameters for ResNet

For Fine-Tuning the Resnet we take 2 approaches.

i) We freeze all the layers except for the last conv5_x block as shown in Fig.5

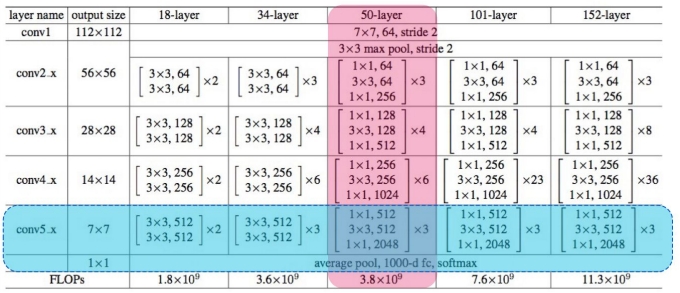

Figure 8. Fine-Tuning conv5_x block

ii) We freeze all the layers except for the Conv5_x and conv4_x as shown in Fig.6

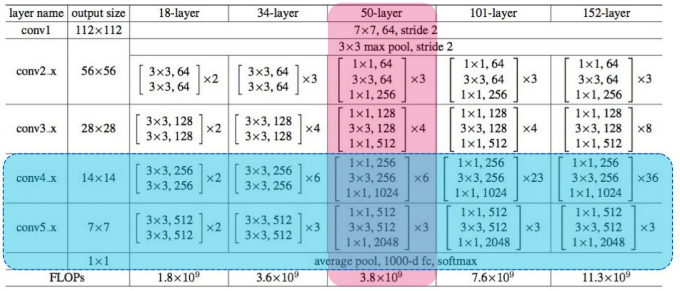

Figure 9. Fine-Tuning conv5_x and conv4_x block

4.3 Results

The evaluation of the model is based on the Test dataset with 600 images.

The model is evaluated on the following metrics:

1. Precision: It is the fraction of relevant instances among the instances retrieved.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

2. Recall (Sensitivity): It is the fraction of true positives that were correctly identified.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

3. F-measure: It is the weighted average of precision and recall.

4. Area Under the Receiver Operating Characteristic Curve (AUC ROC Score): ROC is

a 2D graph (True Positive Rate vs False Positive Rate). The value of ROC is from 0 to 1. The further away the value from 0.5, better the model.

5. Accuracy: It is the fraction of correctly classified instances among all instances

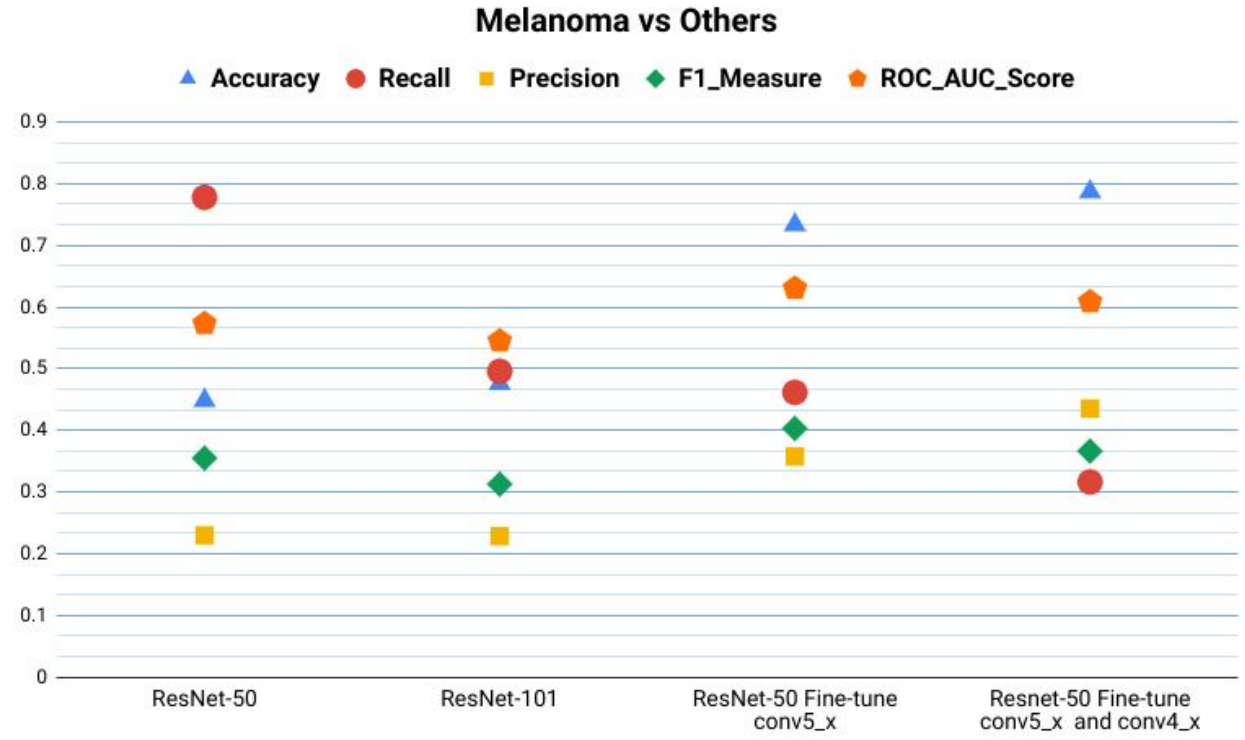

Fig. Plot of metrics for Melanoma vs Others

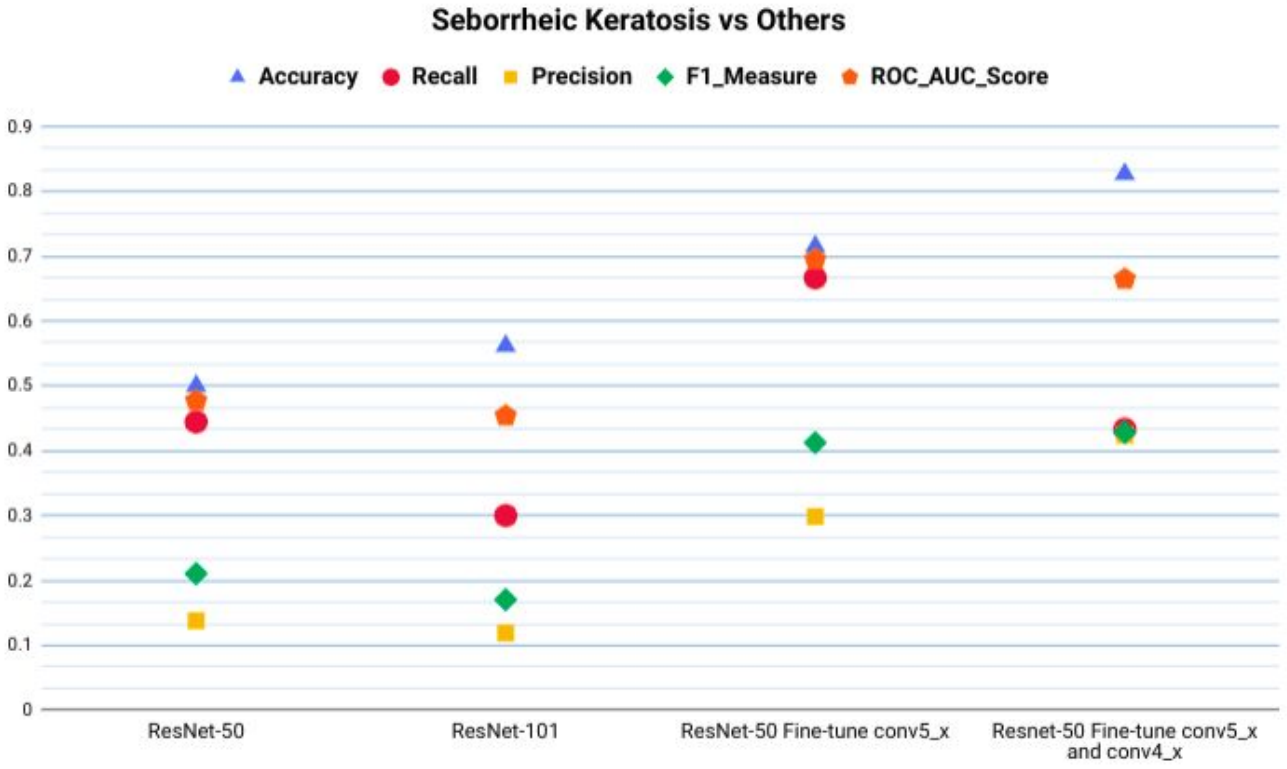

Fig. Plot of metrics for Seborrheic Keratosis vs Others

Model	Accuracy	Recall	Precision	F Measure	ROC_AUC Score
ResNet-50	0.4483	0.77	0.2297	0.3547	0.5731
ResNet-101	0.476	0.4957	0.2283	0.3126	0.5449
ResNet-50 Fine-tune conv5_x	0.7333	0.4615	0.3576	0.4029	0.6303
Resnet-50 Fine-tune conv5_x and conv4_x	0.7867	0.3162	0.4352	0.3663	0.6084

Table. Metrics for Melanoma vs Others

Model	Accuracy	Recall	Precision	F Measure	ROC_AUC Score
ResNet-50	0.5	0.4444	0.1379	0.2105	0.4771
ResNet-101	0.5617	0.3	0.1189	0.1703	0.4539
ResNet-50 Fine-tune conv5_x	0.715	0.6666	0.2985	0.4123	0.695
Resnet-50 Fine-tune conv5_x and conv4_x	0.8267	0.4333	0.4239	0.4285	0.6647

Table. Metrics for Seborrheic Keratosis vs Others

4.4 Conclusion

We can notice that the Method 2 of Fine-Tuning the pre-trained ResNet-50 has a better AUC-ROC, F-measure and accuracy. Therefore, in comparison Method 2 works better than Method 1 of training from scratch. This result was expected because of the concerns raised earlier for Method 1. Also while using Method 1, the graph didn't converge well because of smaller dataset and also lack of training. Training using Method 1 takes lots of time which makes it impractical for us to use it.

We also observe the comparable performance inside Method 2. Fine-Tuning by training only the conv5_x block gave us better results than training conv5_x and conv4_x. This is due to the aforementioned reasons that the deeper layers tend to learn about more specific features of the dataset while the shallower layers learn more generic features.

The limitations of our network is that the accuracy isn't good enough to be used in real-life applications. One of the reasons for this limitation is the small dataset. Due to lack of time, we couldn't get more data from ISIC archive. Getting more data would have increased the performance of our model. Moreover, we are currently using the pre-trained weights from Imagenet but it will be better if we use the weights from Dermnet, which is a skin related dataset. In future, we can try getting more data, applying weights using Dermnet and also perform some more tweaks to our network including but not limited to fine-tuning of the hyper-parameters.

Acknowledgements

We would like to thank Prof. K.S. Leung for his guidance and support during the research. We would also like to gratefully acknowledge the constant support from our supervisor Mr. FU Xinyu, without whom we wouldn't get the right direction. We would also like to thank the CSE Department of The Chinese University of Hong Kong for providing us access to GPU clusters to train our neural networks.

References

[1] Siegel, R., Miller, K., & Jemal, A. (2016). Cancer statistics, 2016. CA: *A Cancer Journal for Clinicians*, 66(1), 7-30.
[2] Odle, T. (2017). Melanoma. 460-463.
[3] Argenziano, G. P., & Soyer, H. (2001). Dermoscopy of pigmented skin lesions - a valuable tool for early diagnosis of melanoma. Lancet Oncology, 2(7), 443-449.
[4] He, K., Zhang, X., Ren, S., & Sun J. (2015). "Deep Residual Learning for Image Recognition" *Cornell University Library, 1.* doi:arXiv:1512.03385v1
[5] Keras: The Python Deep Learning library. Retrieved from https://keras.io/
[6] Module: tf, *Tensorflow.* Retrieved from https://www.tensorflow.org/versions/r1.10/api_docs/python/tf
[7] Perez, F., Vasconcelos, C., Avila, S. & Valle, E. (2018). "Data Augmentation for Skin Lesion Analysis" *Cornell University Library, 1.* doi:arXiv:1809.01442v1