CNN Skin Lesion/Cancer Image Classification Analysis

Andres Salinas, Natalia Valencia

Florida International University
asali018@fiu.edu, nvale010@fiu.edu

October 30, 2023

Abstract

The goal of our project is to analyze and classify image data of skin lesions using the MNIST HAM10000 dataset. The purpose is to correctly identify skin lesion types thereby helping detect the presence of cancer in patients by the image data of their skin lesions. There are a number of machine learning models that can classify image data and one of the most widely used is convolutional neural networks. To get a better idea of its effectiveness on this dataset, we tested the data using Convolutional Neural Networks. We wanted to compare their accuracy overall and assess their performance on the HAM10000 dataset and potentially other datasets.

I. Introduction

Machine learning has become very popular over the years and is currently the most used discipline in data science and analytics. It has become fundamental in creating and automating analytical model building. Applications of machine learning have become widespread in society. Automated machines and self-driving cars are examples of machine learning in action. Another discipline building on the topic of machine learning is deep learning which introduces more complexity and computation to be performed on data. Today, with such large data sets to work with, performance is key in reading and analyzing the raw data. With that in mind, it is important to perform much computation on this data to learn more about it. With more computation needed, powerful machines are also needed. We dive into this topic in our project.

i. Purpose

Our project aims to examine the Skin Cancer MNIST HAM10000 data set which contains skin lesion images. We wanted to apply skin lesion classification by examining the many images of skin lesions. There are 7 different skin lesion variants in our data. They are 0: actinic keratosis(akiec), 1: basal cell carcinoma(bcc), 2: benign keratosis-like lesions(bkl), 3: dermatofibroma(df), 4: melanocytic nevi(nv), 5: vascular lesions(vasc), and 6: melanoma(mel). Vascular lesions include (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage). We planned to create a model that can be trained to read skin lesion images and correctly classify what kind of skin lesion is present and determine if cancer is present. We will be utilizing neural networks which can take a large amount of data and perform thorough computation to create an efficient model. Specifically, we will be working with Convolutional Neural Networks(CNNs). This type of neural network has been known for image classification so we feel it will be the most applicable choice in this project. We will be detailing the accuracy and performance of four different types of convolutional neural networks, LeNet, AlexNet, ResNet, and MobileNet.

II. Methods

In order to assess our data and develop a model to make classifications on skin lesions/cancer. We will be utilizing these methods/technologies.

- Convolutional Neural Networks
- Support Vector Machines
- Image classification
- Google Colab GPU/TPU
- TensorFlow
- ImageDataGenerator

i. Convolutional Neural Networks

Neural networks are incredible tools for performing machine learning. They help understand and develop models to perform a task based on training samples. The input data is passed through a series of nodes which are housed in different layers and is transformed/altered through a series of complex computations. This process is known as feedforward and the output is ultimately revealed in the final layer which can be used to make predictions. In particular, a convolutional neural network is a type of neural network that involves working with 2-D images and performs the operation of convolution by multiplying our input by a set of weights called a filter. Image classification is one of the most common use cases for convolutional neural networks so there are a number of them to apply to our data set. [Saha, 2018] For this project, we are focusing on testing the efficiency of LeNet, AlexNet, ResNet, and MobileNet on the MNIST Skin Cancer HAM10000 dataset. The goal in mind is to determine how effectively each can classify skin lesion types and detect skin cancer.

ii. LeNet

LeNet is the first proposed CNN. It was created to interpret and perform image classification, specifically working with the MNIST data set. Its architecture is made up of 7 layers that consists of 3 convolutional layers, 2 subsampling layers, and 2 fully connected layers. For its activation functions, it uses tanh and sigmoid functions. [Alake, 2020]

iii. AlexNet

AlexNet builds on the architecture of LeNet. However, it adds two main advantages. It addresses the issue of overfitting by allowing for data augmentation and dropout. It incorporates batch normalization and dropout to reduce overfitting issues and maintain the accuracy of the model. [Wei, 2019] Its architecture is made up of 8 layers that consists of 5 convolutional layers and 3 fully connected layers.

iv. ResNet

The question arises whether adding more layers or complexity to CNN models improves performance and accuracy, and the answer is "not always". ResNet tackles this question by proposing a model that incorporates the residual block. The residual block is a technique that is commonly referred to as the 'Switch Connection', it allows for input data from previous layers to skip proceeding layers. This works by using identity mapping to add the outputs from previous layers to the outputs of stacked layers. This process reduces degradation that happens when using multiple layers. [Shorten, 2019]

v. MobileNet

MobileNet is a lightweight architecture. MobileNet uses depth-wise separable convolutions, which is a form of factorized convolutions which factorize a standard convolution into a depthwise convolution and a 1×1 convolution called a pointwise convolution, to build light weight deep neural networks. MobileNet employs the use of two global hyper-parameters to efficiently trade off between latency and accuracy. These hyper-parameters allow one to specify the right model size based on the constraints of the problem.[Howard, 2017]

vi. Pre-processing

In order to properly set up and utilize the CNN architecture, the input data needed to be resized or transformed from 28 x 28 pixels to 32 x 32 pixels. Since we were reading in the RGB data, we utilized a depth of 3 for each tensor. This allows for our input data to factor in the 3 channels of RGB. The only difference in input shape was when pre-processing the data for AlexNet and MobileNet which required resizing from 32 x 32 x 3 pixels to 227 x 227 x 3 pixels and 224 x 224 x 3 pixels respectively. In addition, the data needed to be normalized using min - max normalization to allow the model to interpret the data more efficiently. To train our models, we split our data sets into training and test data sets. We tried out 70-30 split and 80-20 split to compare results. The data sets we tested for our analysis include the standalone 28x28 RGB data set, 5000 images from the Part 1 folder of the data set, and augmented images of these 5000.

III. Results

We compared and analyzed the output of each CNN model on skin lesion/skin cancer classification using the HAM10000 data.

i. Performance

Overall, the performance of our model training showed that batch size and epoch number were key factors in creating an effective model. Once the physical image files in the data set were read, the training performance of each model was heavily dictated by the processing and graphical power of a powerful computer. We used Google Colab GPU to train and test our model. There were about 10,000 images to read for the data set. For our training purpose, we read in 5000 images from the Part 1 folder. Training each model utilizing the original images provided yielded poor results regardless of using different settings for batch size and epoch number. Below are the results of the model training.

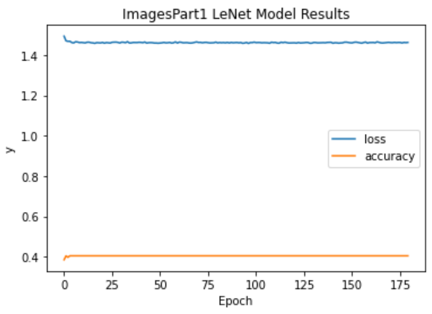


Figure 1: *LeNet Model P1 Results*

As you can see, the performance of the LeNet model under batch size of 45 and epoch number 180 shows a slight increase in accuracy followed by a flat constant accuracy of 0.405. Loss also did not improve much and faced a similar result.

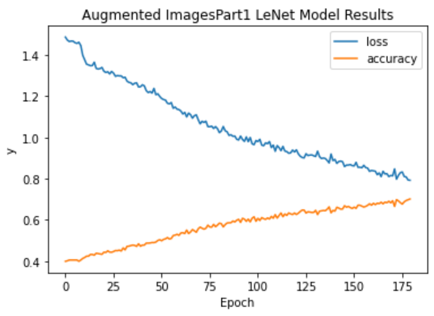


Figure 2: *LeNet Model Augmented P1 Results*

For the purpose of improving the performance of our models, the original images were augmented by shifting their position and rotating them. With this procedure done, the result of the LeNet model training improved as shown above. The accuracy steadily increased to a maximum of about 70 percent over 180 epochs. The model would have steadily increased given a large number of epochs. Testing the validation data averaged 66 percent accuracy.

Building on LeNet, AlexNet included the use of more convolution layers and required the images to be resized to 227 x 277 x 3 shape.

When running the original images through AlexNet, the accuracy of the model climbs and then rests at 40 percent. The original images

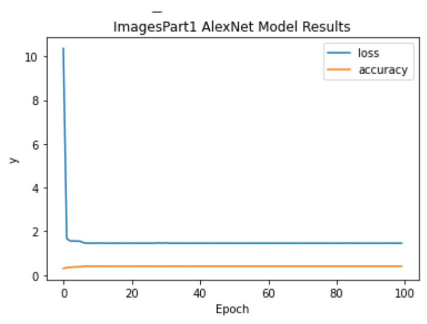


Figure 3: *AlexNet Model P1 Results*

do not provide much variance and therefore the model suffers due to this. It is clear that the original images did not provide enough variance or input to properly train the model.

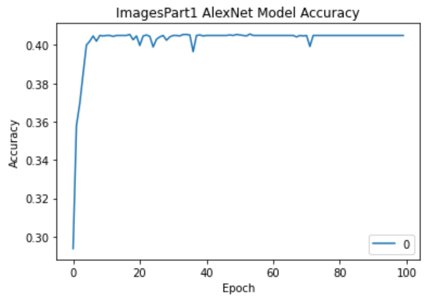


Figure 4: *AlexNet Model P1 Accuracy*

Taking a closer look here, we can see that the model started at 30 percent accuracy. From there, it climbed and finally peaked at 40 percent. Just as LeNet, the images needed to be augmented to create a stronger model.

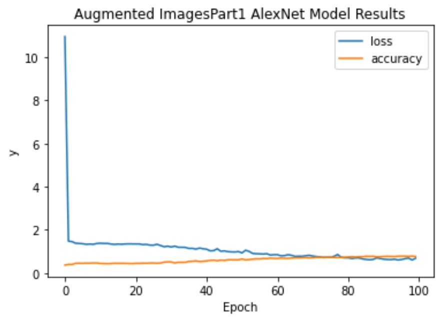


Figure 5: *AlexNet Model Augmented P1 Results*

Once the images were augmented, the model was able to account for more variance and create a stronger model. Loss went down and the accuracy slowly went up.

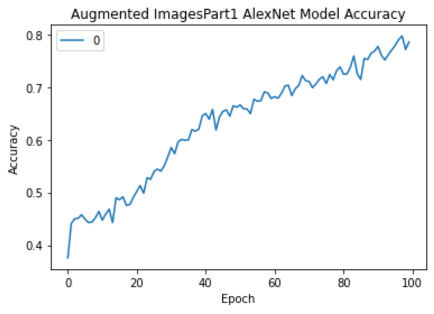


Figure 6: *AlexNet Model Augmented P1 Accuracy*

Since the difference is so high during model training, the accuracy is hard to read in the previous graph. This graph showcases the accuracy performance over the 100 epochs. We can see that it neared 80 percent accuracy at 79.72 percent accuracy. The model shows greater performance with more varied input. The validation test data accuracy reached 71.9 percent accuracy on average as well. This was very promising to see.

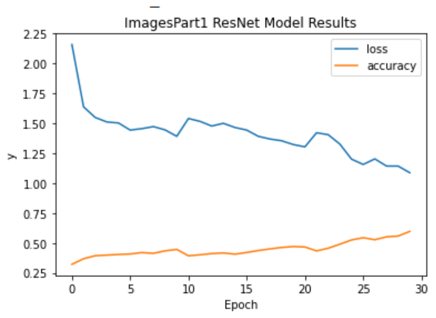


Figure 7: *ResNet Model P1 Results*

ResNet shows much greater performance than the previous models. As mentioned before, it allows for much deeper layers without compromising accuracy and performance. Despite the lack of augmentation and using the original images, the model is able to reach close to 60 percent accuracy. This is accomplished in 30 epochs in comparison to 100 epochs in AlexNet.

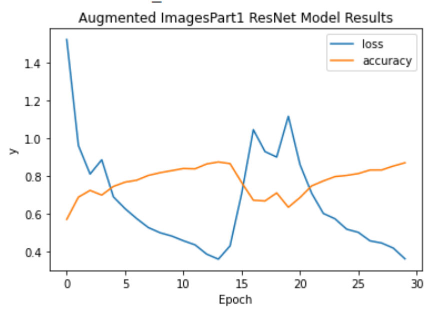


Figure 8: *ResNet Model Augmented P1 Results*

When we use augmented images, the performance of the ResNet model vastly improved. The accuracy soared to 87.4 percent accuracy with an average 77.2 percent accuracy. However, despite this high training accuracy, the same cannot be said about the validation test data accuracy. On average, it was much lower. This prompted us to play around with the hyper-parameters of batch size and epoch number. Validation test accuracy averaged around 45 percent which was quite abysmal.

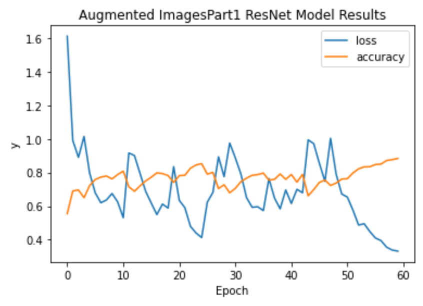


Figure 9: *ResNet Model Augmented P1 Results(Updated)*

Figure 9 now shows us an improved ResNet model. By tweaking the batch size to 100 and increasing our epoch number to 60. We were able to see much more promising results. The model showed a steady increase in overall training accuracy and decrease in loss. When evaluating the model to create predictions using the test data, we saw an average of more than 70 percent in identifying the correct skin lesion type. We see that further training is needed but the performance of ResNet is clear.

For MobileNet on the other hand, we first augmented the images due to the imbalance present in the original data set. We generated about 35,000 more images to compensate for the imbalance. We split the data 90 / 10 stratified. The train set contained 42,296 images, while the validation set contained 4,700 images. Since MobileNet is a lightweight architecture, we applied transfer learning to fit our model as it reduces the training time considerably, since we're training over 40,000 images.Before fitting the model, the images were resized to 224 x 224 x 3 and encoded the batch of images into a numpy tensor.The training data was trained for 10 epochs. Each epoch trained 91 batches. We achieved an average accuracy of 90 percent, with our best accuracy being at 97.07 percent. After evaluating the model on the validation set, our accuracy converged at 88 percent.

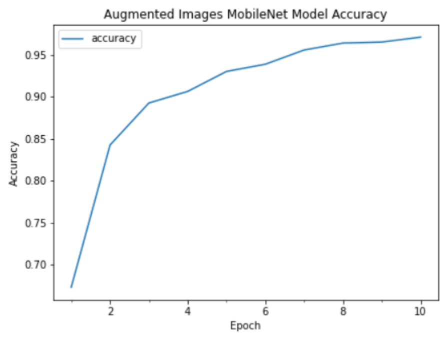


Figure 10: *MobileNet Model Augmented Accuracy*

We can see from Figure 10 that the performance of the MobileNet model starts at around 68 percent accuracy and increases over time to an impressive 97.07 percent.

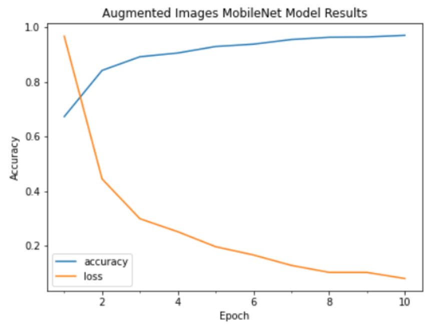
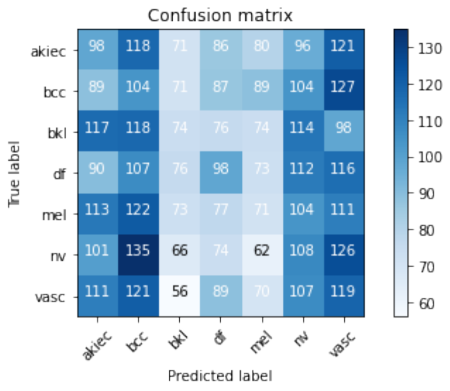


Figure 11: *MobileNet Model Augmented Results*

Figure 11 demonstrates the efficacy of the model when we increase the size of the training data to account for class imbalance. With just 10 epochs we can reach an accuracy of 97 percent and an 88 percent validation accuracy.



	akiec	bcc	bkl	df	mel	nv	vasc
akiec	98	118	71	86	80	96	121
bcc	89	104	71	87	89	104	127
bkl	117	118	74	76	74	114	98
df	90	107	76	98	73	112	116
mel	113	122	73	77	71	104	111
nv	101	135	66	74	62	108	126
vasc	111	121	56	89	70	107	119

Figure 12: *MobileNet Model Augmented Confusion Matrix*

The Confusion Matrix above shows the classes correctly and incorrectly predicted after using the validation data to evaluate the model.As we can see from the image, akiec was correctly predicted 98 times, bcc was correclty predicted 104 times, bkl was correctly predicted 74 times,df was correctly predicted 98 times, mel was correctly predicted 71 times, nv was correctly predicted 108 times, and vasc was correctly predicted 119 times. bkl was the only class not highly classified as vasc. A lot of images were also grouped into bcc incorrectly.The highest concentration of images incorrectly predicted was 135 nv into bcc.The highest concentration of images correctly predicted was 119 vasc.

Table 1: *Overall Model Performance*

CNN	Data	Batch	Epoch	Acc
LeNet	28x28	45	180	0.7137
LeNet	P1	45	180	0.4049
LeNet	P1Aug	45	180	0.5701
AlexNet	28x28	99	200	0.8725
AlexNet	P1	99	100	0.4017
AlexNet	P1Aug	99	100	0.6304
ResNet	28x28	128	20	0.8939
ResNet	P1	128	20	0.4475
ResNet	P1Aug	128	20	0.7719
ResNet	P1Aug	100	60	0.7663
MobileNet	Aug	91	10	0.9036

The overall results of model performance testing is listed above and showcases the different data sets and pre-processing used to determine what works the best. The CNN refers to the type of Convolutional Network being tested, the data, the batch size, epoch number, and average accuracy of the model.

IV. Discussion

i. Important Points

With all our models trained and tested, we determined that our use case of skin lesion image classification can in fact be performed using Convolutional Neural Networks. Our strongest model which yielded the best results was MobileNet and ResNet. These two CNN model incorporate multiple layers however they do not lose their performance despite this. To better gauge our data sets, we performed train-test data split in 70-30 and 80-20. We noticed that the performance of each model was better when only using 70 percent of our data for training. This is an odd result but it could be related to the batch size and epoch number settings. In addition, when we were using the 28X28 RGB csv files as input, the models performed better. This likely shows that processing less data avoids any errors and the pixel values are already given. When using the physical images provided, we first needed to convert each image to 28x28 RGB pixel values and then begin pre-processing. Since this procedure is dependent on the power of the machine we are using and it is a manual effort on our end, there is a chance for error during the conversion. Perhaps the conversion of the images to pixel values was not without error which could have led to a drop in model performance. This is a key part of our analysis and is likely the reason for the numbers we saw.

ii. Future Work

Following this experiment, we would like to explore a better way to balance the data as there is a strong bias towards label 4: melanocytic nevi. SMOTE analysis was an option we tried but it seemed to create inefficient models despite balancing the data. Image augmentation was key in this project so it would be fruitful to explore other avenues that incorporate more robust data augmentation. Finally, aside from testing our validation data, it would be very interesting to explore how our models would perform when receiving different skin lesion data sets or real world images. I believe only then can we assess and improve our models.

References

[Alake, 2020] Alake, Richmond. (2020, June 25). Understanding and Implementing LeNet-5 CNN Architecture (Deep Learning). https://towardsdatascience.com/understanding-and-implementing-lenet-5-cnn-architecture-deep-learning-a2d531ebc342

[Blickerton, 2018] Blickerton, Charlie. (2018, Aug. 1) A beginner's guide to decision tree classification. https://towardsdatascience.com/a-beginners-guide-to-decision-tree-classification-6d3209353ea.

[Culfaz, 2018] Culfaz, Ferhat. (2018, Nov. 6) Transfer Learning using Mobilenet and Keras. https://towardsdatascience.com/transfer-learning-using-mobilenet-and-keras-c75daf7ff299

[Howard, 2017] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[Mach, 2019] Mach, Joey. (2019, Oct. 10) Teaching Machines to Detect Skin Cancer. https://medium.com/analytics-vidhya/teaching-machines-to-detect-skin-cancer-bd165566f0fe

[Mader, 2018] Mader, K. (2018). Skin Cancer MNIST: HAM10000. https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000

[Ray, 2017] Ray, Sunil. (2017, Sept 3). Understanding Support Vector Machine(SVM) algorithm from examples (along with code). https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[Saha, 2018] Saha, Sumit. (2018, Dec. 15). A Comprehensive Guide to Convolutional Neural Networks. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[Shorten, 2019] Shorten, Connor. (2019, Jan. 24). Introduction to ResNets. https://towardsdatascience.com/introduction-to-resnets-c0a830a288a4

[Wei, 2019] Wei, Jerry. (2019, July 2). AlexNet: The Architecture that Challenged CNNs. https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951