Computer Science

Analyzing the Effectiveness of Convolution Neural Networks in Automated Skin Cancer Classification

Question

How do the different Convolution Neural Network architectures ResNet50 and VGG16 compare in classifying various skin lesion images for automated diagnosis of skin cancer?

Word Count: 3984

Contents

List of Figures

1 Introduction

The rapidly evolving nature of the field of computer science has lent itself to advancements that were considered impossible just a few short years ago. A prime example of this are artificial neural networks(ANNs), a computational paradigm that learns the solution to problems through a set of examples, or context (Bishop, 1994). A particularly interesting application of neural networks comes in the form of 'Computer Vision'. Computer vision is a field of study that aims to enable computers to derive information (IBM, nd) or make inferences (Learned-Miller, 2011) about the real world from visual inputs. Accurate, efficient and successful Computer Vision neural networks have many applications, from self-driving cars to medical diagnosis.

Image classification is a foundational problem in Computer Vision, and allows the ability for other computer vision tasks such as localization and detection (Rawat and Wang, 2017). Initially, a dual-staged approach was taken to solve the image classification issue, where "features" of images, which are the distinct attributes or patterns within an image that a neural network can use to differentiate images from one another, were first extracted separately using "feature extractors" which capture the "features" of the image and transform them from visual data into a form that can easily be compared and analyzed. These features were then served as inputs to a trainable classifier (Rawat and Wang, 2017). Due to the dependence on the success of the feature

extraction for these techniques, CNNs are now far more widely adopted for image classification tasks.

CNN models have layers, which are essentially different computational units within the network, and allow them to process information non-linearly, being able to perform feature extraction and image classification much more efficiently. This overcomes many of the pitfalls of the dual-staged approach, and has meant that CNNs are now the most commonly used models for image classification tasks.

CNNs have found applications in various medical contexts, such as with skin cancer classification (Esteva et al., 2017), lymph node detection (Ehteshami Bejnordi et al., 2017), Alzheimer's disease(Awate) and Segmentation in radiology (Yamashita et al., 2018). Since there is past precedent for this technology being used successfully in medical applications, this paper seeks to compare different CNN models, the VGG-16 model and the ResNet-50 model, in skin cancer detection and classification. This EE contributes to a critical understanding of the practicality and effectiveness of these different technologies in a medical context, which aligns with the broader objective of leveraging AI technology to enhance diagnosis and treatment strategies.

2 Background

2.1 Background of Neural Networks

A neural network is essentially a large assembly of interconnected “processing elements”, also known as “neurons” (Agatonovic-Kustrin and Beresford, 2000; Sanderson, 2017). These processing elements perform a small unit of computation upon the output of which they change their state. They can be considered bi-state devices, which are off until the computed value reaches or exceeds a certain threshold, at which point they “activate”, or more simply, turn on. A neural network is made up of “layers”. These layers are structural units of the neural network, and consist of a set of neurons that perform one task. A network most commonly consists of an input layer, "hidden" layers where the inputs are transformed, and the output layer where the input is given a final classification.

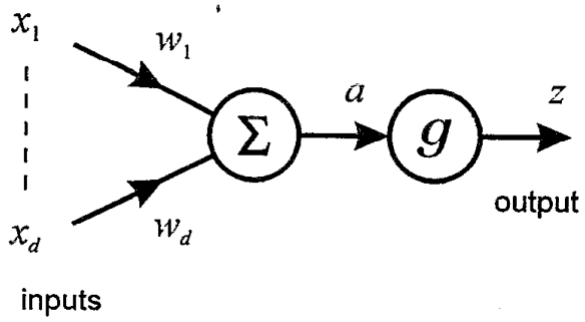

Figure 1: Image of a single neuron computation (Bishop, 1994)

In a network, the most common layer type is the fully connected layer, where each neuron of one layer is connected to all other neurons of the preceding and succeeding layers through synapses(connections) (Karpathy, 2018b), as shown in Fig. 1. The links between different neurons are given "weights" (Sanderson, 2017). The outputs of the neurons interact with these weights multiplicatively (Sanderson, 2017), and are then passed forward to the next neuron. A weighted sum is then computed by the neuron the value was passed on to, summing all the multiplied values of the outputs and weights of different neurons. Next, a non-linear transformation of the weighted sum takes place in the "activation function". Since this weighted sum could output any number, the sum is given as input to an activation function to make sure the values outputted are greater than 0. This can be represented by the formula $f(x) = \phi(\mathbf{w}^T x + b)$, where x is the input to the neuron, $\mathbf{w}$ is the weight, b is the bias and the ϕ is the non-linear activation function (Dodge and Karam, 2016). There are 2 main activation functions that are used; the sigmoid function($\phi(x) = \frac{1}{1+e^{-x}}$) and the ReLU function($f(x) = max(0, x)$). In recent years, the ReLU function has come to be much preferred, due to the fact that it is less computationally expensive, as the activation simply thresholds at 0 (Karpathy, 2018b). This process continues until the neurons on the output layer are activated. Each value in the output layer calculates the probability of the answer being what the neuron with the value represents, and the class with the highest probability is chosen as the output of the model (Sanderson, 2017). The output is then plugged into a loss function, which

calculates the loss based on how close the output was to the given input. This cost function is then minimized through adjustments made to the weights and biases (Sanderson, 2017) through a process called backpropagation. This is how the network "learns".

As is stated in the introduction, neural networks process information non-linearly. This non-linearity is the result of activation functions. Taking an example of the ReLU activation function, it applies a non-linear transformation to the output of the weighted sum, outputting 0 if the weighted sum is negative, and the weighted sum itself if it is positive. Without the activation function for the neurons, neural networks would just be multi-layered linear transformations, no matter how complex the network. The non-linearity of neural networks give them many advantages, such as the ability to have complex and flexible decision boundaries, unlike traditional logistic regression, which is only effective when classes can be separated linearly (Ali, 2018). For neural networks, the use of excessive hidden layers has been observed to result in a problem called "overfitting". This problem is dependent upon two things: variance and bias. Bias refers to how much the model ignores the training data, and variance refers to how dependent our model is on the data (Cleveland, 2018). If the model is highly dependent on the test data and has low bias, meaning is heavily influenced by test data, it has a very likely chance of being overfit. Since we cannot trust the data to always be the most optimal source of learning, we account for this by using bias, which makes our model more resistant to learning, essentially requiring a higher threshold

for "activation". Opposingly, high bias and low variance means the model is considered underfit. A model that is too critical of the training data fails to learn the underlying relationships between input and output (Cleveland, 2018). As such, a good balance is always necessary when training. This has been a brief introduction to how neural networks compute non-linearly. All of this processing is applicable to the "fully-connected" layers of CNNs.

2.2 Convolution neural networks

Convolution neural networks are a type of neural network that specialize in processing data that comes in the form of multiple arrays, such as images. CNNs are feedforward networks, meaning the information flow within them is only in one direction; from the input layer to the output layer. They are inspired by the human visual cortex (Yamashita et al., 2018). CNNs mainly consist of 3 types of layers: the convolution layer, the pooling layer and the fully connected layer (Yamashita et al., 2018). The convolution and pooling layers are used for feature extraction, while the fully connected layers are functionally the same as those described above (Karpathy, 2018a). Below, we will be discussing more about the inner workings of the feature extraction layers of the CNN.

2.2.1 Tensors and convolution layers

Central to CNN models is the "convolution" layer of the network. It carries much of the computational load within the network. The network takes images as inputs and computes them in the form of tensors, which are multi-dimensional arrays, or matrices, that store the pixel values of any given image.

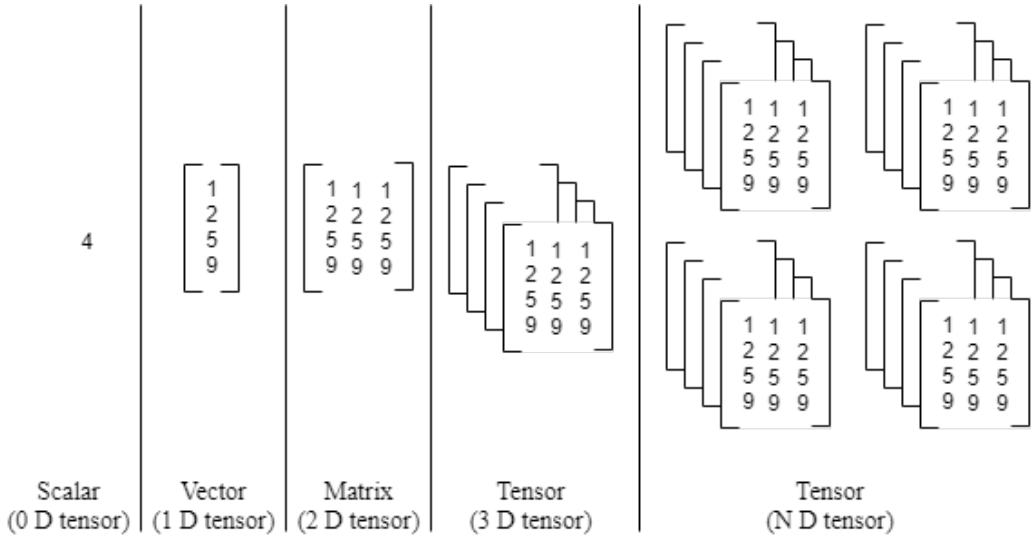

Figure 2: Visual representation of different dimension tensors (Aslam, 2022)

Color images, which are what mainly constitute the data being used in this paper, are represented as 3 dimensional tensors, as shown in Fig. 2, while a batch, or a number of different color images being fed to a network is represented as 4 dimensional tensors. Each matrix in the stack of three matrices for a color image contains the pixel values for one of the three primary colors - red, green and blue. These matrices are also called "channels"

and will be referred to as such henceforth.

The convolution layer of the CNN performs "feature extraction", in which, rather than just analyzing the tensor of the given image straight away, it looks for patterns and structures in the tensor instead. This is less computationally expensive than analyzing the tensors of the image straight away, as the complexity of an image scales exponentially with size (Big-O of $O(n^2)$). It achieves this through a number of linear and non-linear computations, i.e. convolution operation and the activation functions (Yamashita et al., 2018) discussed before.

The convolution layer is not a fully connected layer, as with high volume of input such as the ones we get with images, each neuron being connected to each output from the previous layer is impractical. Rather, the neurons in the convolution layers have a hyperparameter(a parameter that can be adjusted) called the *receptive field* (Karpathy, 2018a). This receptive field defines how much of the input volume, or the output of the previous layer each respective neuron is connected to. This is called a **local region** of the input volume (Karpathy, 2018a). The neurons in a convolution layer are grouped into "feature maps"(Rawat and Wang, 2017). All neurons within a feature map share the same local region of input, and also share the same learnable parameters (parameters such as weights and biases that define the *kernels*, which we will see more of now).

These feature maps each contain a number of learnable filters (small matrices),

also known as *kernels*, which "slide over" the input given to the convolution layer. The height and width of each kernel is smaller than those of the input volume, but the depth of the kernel will always be the same as the input volume. This means that the kernel can also be considered a stack of three matrices, where each matrix corresponds to one of the channels of the input volume.

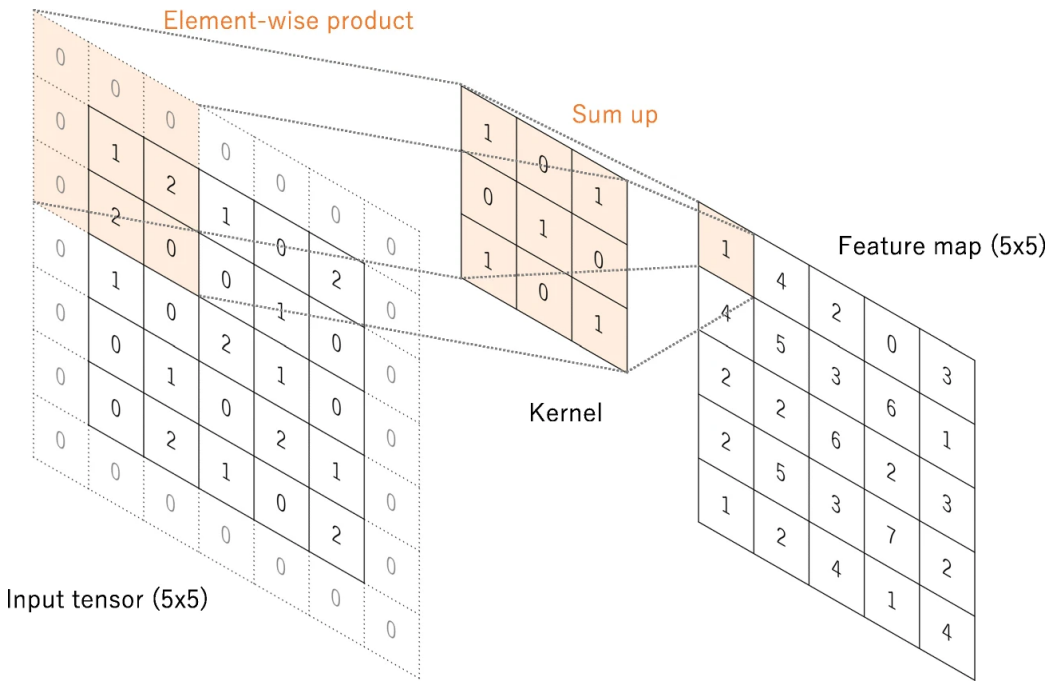

Figure 3: Kernels at work (Yamashita et al., 2018)

Fig. 3 shows the process of kernels sliding over the input tensor, with a padding of 0 around the borders. Zero-padding is a hyperparameter, and is typically used when working with kernels, as without padding, the spatial dimensions of the output is constantly reduced as convolution processes take

place, which can be detrimental to the output given by a network when the network has a large stack of convolution layers (Yamashita et al., 2018). The content of these kernels can be considered the *weights* of the neurons within the feature map. The computation is similar to those described above for a general neural network, meaning each element of the kernel interacts multiplicatively with the value at a given location of the tensor, and the sum of these values are then plugged in a non-linear function(either sigmoid or ReLU), and then stored in the neuron at the corresponding position on the feature map (Rawat and Wang, 2017). The *stride* of a kernel refers to the step size it moves across the input data once each time. A stride of 3x3, for example, means that the kernel will step across the data towards the right 3 steps each time, and if there are no more elements on the right, it will start again on the extreme left of the row directly below. The spatial size of the output of a convolution layer is given by the formula $(\frac{W-F+2P}{S}+1)$, where W is the spatial size of the input, F is the receptive field size, P is the padding used and S is the stride applied (Karpathy, 2018a). Then, the output of these feature maps are given as inputs to pooling layers.

2.2.2 Pooling and fully connected layers

The feature map, which is the output of a convolution layer, becomes the input to the other layers of the CNN. This often results in high computational complexity, and can also result in overfitting. As such, it is common to find

pooling layers between convolution layers. A pooling layer's function is to downsample, or to reduce the spatial size of the output. How pooling layers control overfitting is specially important.

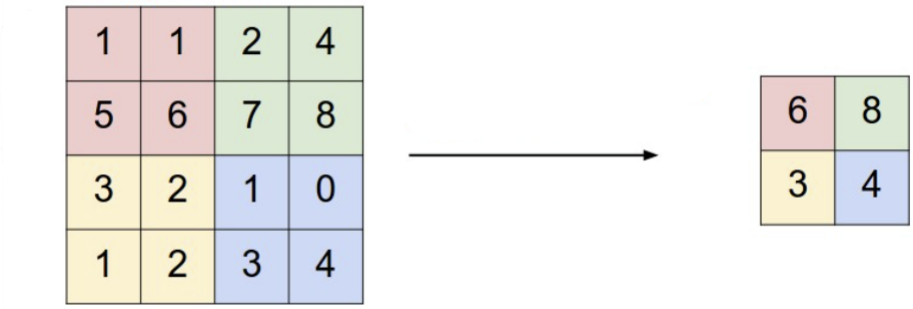

Figure 4: How a max pooling layer works (Karpathy, 2018a)

Max pooling is the most common application of pooling layers. It works by selecting the largest value as a representative value from a group of pixels, as can be seen in Fig. 4. So, from any given set of features, the pooling layer ensures that only the most important component is selected, and the noise is filtered. This makes the model better at generalizing for testing data that it hasn't seen during training, as it has not been trained on the noise that may come with the inputs.

The fully connected layers are exactly the same as the ones described for general neural networks. They have weighted synapses that connect to every neuron in the preceding layer.

3 Methodology

Empirical experimentation is the main source of data for this paper. For this experimental analysis of which model of CNNs are more appropriate for image classification tasks in the medical field, 2 different CNN models have been chosen and a large public dataset is being used.

3.1 Model Selection

The two CNNs being used for this experimentation are the Visual Geometry Group's VGG16 model and the Residual Network's ResNet50 architectures. These two models were chosen as they are both very popular for image classification tasks, but have very different architectures. The ResNet50 has a depth of 50, and is a far deeper CNN than VGG16, which only has a depth of 16. Investigation into which model can more successfully and accurately diagnose different types of skin cancer can prove to be fruitful for further implementations of Computer Aided Diagnosis(CAD) systems. Given the task's specific, diagnostic nature, the public dataset being used has a lot of clear visual patterns. This can be a cause of *overfitting* in the deeper ResNet50, while the VGG16's shallower architecture could be better at generalizing with this data.

3.2 Dataset

The HAM10000 dataset ("Human Against Machine with 10000 training images") is being used for this experimentation. It is a public dataset consisting of 11721 images of skin lesions from 8 different classes, along with appropriate metadata for each image. The classes included in the dataset are Actinic keratoses and intraepithelial carcinoma / Bowen's disease *(akiec)*, basal cell carcinoma *(bcc)*, benign keratosis-like lesions *(solar lentigines / seborrheic keratoses and lichen-planus like keratoses, bkl)*, dermatofibroma *(df)*, melanoma *(mel)*, melanocytic nevi *(nevus)*, vascular lesions *(angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, vasc) and squamous cell carcinoma* (SCC. (Tschandl et al., 2018).

3.3 Variables

The dependent variables for this experimentation will be the accuracy of the two models in correctly diagnosing different types of cancer, and the F1 Score and Confusion Matrix that they achieve, while the independent variables will be the different parameters of each network. All hyperparameters, such as the number of epochs trained, the batch size, the learning rates and the training, validation and testing datasets will be the controlled variables, and will be the same for both models.

3.3.1 Accuracy

The accuracy of the two different CNN models in correctly diagnosing skin cancer given an input was compared after implementation and training, using the a subsection of the dataset specifically separated for testing the models. The accuracy of the model quantifies the correctness of the classification of the network in comparison to the actual class of the input. It can be defined as the proportion of correct prediction made by the model with respect to the number of predictions made.

3.3.2 F1 Score

The F1 Score is the mean of both the precision and recall metrics that can be calculated for the multi-class(or multi-label) classifications made by a model. It is a single metric that identifies both the false positives(represented by the precision) and false negatives(represented by the recall) of the model, and is a more comprehensive measure of the accuracy of multi-label classifications made by the model. It is bound by 0 and 1, with 0 being the minimum precision and recall that can be achieved by a model, and 1 being the maximum.

3.3.3 Confusion Matrix

The confusion matrix is a table that shows the correct and incorrect predictions for each class against every other class. This matrix can be evaluated to see which model performs better at identifying images that belong to which classes, and where the model fails most often, from which insight about

different training methods and ideas for further development can be gleaned.

3.4 Experimentation procedure

The dataset was first divided into three sections for training, validation and testing. 70% of the dataset was used for training, and all the images were divided into folders of their respective classes as defined in the metadata for the dataset. 15% of the dataset was used for validation, and the other 15% was used for testing the two different models. The code used to achieve this is listed in appendix C.1 of this paper. The procedure was carried out on the platform "Google Colab", utilizing the Tensor T4 GPU runtime. The pre-trained implementation of these models came from the Keras library offered through Tensorflow, and initialize with the same learning rate of 0.001 by default. The weights of these pre-trained models were trained on the ImageNet database, which is a very large database of images that belong to 1000 classes. However, as the dataset for this experimentation only extends up to 8 classes, the final output layer and the input layer of both the models were discarded and new fully connected layers were created for each, with a softmax activation function for classification in the output layer. These models were then fine tuned on the training dataset. This means that after every "batch" of input images, which was set to a size of 32 images for this experimentation, the network performed a loss calculation and backpropagation to adjust weights and biases accordingly. The code for this initialization and fine-tuning can be found in appendix C.2.

4 Results of the experimentation

4.1 Analyzing accuracy

The table in Appendix "A" showcases the correctness of each model as a binary selection between "True" and "False" based on the classification it makes for each image from the testing dataset. The classes range from 0-7, demonstrating the 8 classes that the images come from. We can see that the ResNet50 model achieves an accuracy percentage of 74.83% in comparison to the 72.7% accuracy achieved by the VGG16 model. It must be stated though, that the accuracy both models achieved on the training data was superior to these metrics, at 82% and 79% respectively. This shows that there may have been some overfitting for both models when in the training process. The superior accuracy of the ResNet model elucidates that with all other hyperparameters remaining constant, its deeper architecture allows it to adapt to the specific task better. This could be due to the fact that the Resnet50 is more parameter-efficient than the VGG16 despite having significantly more layers. An abstracted overview of the number of parameters for each model is given in appendices B.1 and B.2 respectively. It achieves this through the "skip connections" it utilizes in its residual layers, which allows neurons to learn the residual output it must give, meaning that if the output a neuron is expected to give is originally $H(X)$, a neuron in the residual layer instead learns to give the output $F(x) = H(x) - x$, where x is the neuron's input. This means that if the value of the input to this certain neuron is close to the

input value desired for the neuron it is connected to, then $F(x) = x$, making the model far more efficient and suited for deep learning.

4.2 The F1 Scores

Model	**F1 Score**
Resnet50	0.7157902175
VGG16	0.7119235467

The ResNet50 achieved a higher F1 Score than the VGG16 model, as shown in the table above. A sample-weighted F1 score was measured for both the models to account for the heavy imbalance in the classes of the dataset. The macro averaged or micro averaged F1 scores are usually the most common F1 metrics that are calculated, but they are only applicable for models that have been trained and tested on datasets with balanced classes. For datasets with imbalanced classes, the sample-weighted F1 Score is preferred, as it is the *weighted* average of each class's individual F1 Score, and the weight for the class is determined by the number of samples that the class has in the dataset. The higher F1 Score of the ResNet50 model, although by a small margin, suggests that its ability to balance the precision(ability to only give the correct classification) and recall(ability to find all instances of a given classification, even though some classifications made in the process might be wrong). This is especially important in applications such as medicine, in

which both false positives and false negatives are essential to minimize. The code used to calculate the F1 scores of the models is in appendix C.4.

4.3 Analyzing the Confusion Matrix for both models

Table 1: Confusion Matrix for ResNet50

	actinic keratosis	basal cell carcinoma	dermatofibroma	melanoma	nevus	pigmented benign keratosis	squamous cell carcinoma	vascular lesion
actinic keratosis	3	1	0	1	5	5	0	0
basal cell carcinoma	0	22	1	6	28	5	0	0
dermatofibroma	0	0	5	1	9	1	0	0
melanoma	0	0	0	53	71	6	0	0
nevus	0	4	0	22	735	12	0	1
pigmented benign keratosis	0	2	1	13	73	44	0	1
squamous cell carcinoma	0	4	0	1	13	4	1	0
vascular lesion	0	1	1	1	1	0	0	14

From the two tables given above, we can distinguish the strengths and weaknesses of each model for this image classification task. Based on the confusion matrices, the ResNet50 shows signs of overfitting, as the number of images it has incorrectly classified as "nevus" is significantly more than the VGG16 model. However, the ResNet50 has 877 correct predictions while the VGG16 has only 852 correct predictions. Both models can be seen to be struggling significantly with classifying squamous cell carcinoma and dermatofibria. This is most likely due to the lack of samples that these classes have in the testing and training dataset. The VGG model significantly outperforms ResNet when classifying samples belonging to basal cell carcinoma, as the ResNet50 tends to classify them as nevus with far greater frequency. Even though the volume of correct predictions for the ResNet50 is greater, the distribution of these correct predictions are not

diverse, being heavily concentrated on correctly predicting samples from nevus, which seems to be both a strength for it, and also possibly a concern due to overfitting concerns. Evidently, it can be claimed that VGG16's distribution of correct predictions makes it more likely to achieve a greater balance between precision and recall (the F1 score) if the dataset was not class-imbalanced.

5 Limitations for the experiment

The first limitation that must be addressed for this experimentation is the class imbalance present in the dataset that was used. A dataset with more balanced classes could be used for further research into this topic. Alternatively, measures to artificially increase the samples in underrepresented classes could be implemented. Another limitation is the training duration. Due to limitations in connectivity to the Google Colab runtime, the models were only trained for 30 epochs each. A more accurate reflection of the performance of these models can be achieved with more comprehensive training. For more fine-tuned models, both the VGG16 and the ResNet50 could be manually implemented without the use of pre-trained variants.

6 Conclusion

This paper has explored the general workings of CNNs and how they have achieved "computer vision", as well as the efficiency of two different CNN architectures, ResNet50 and VGG16 in correctly identifying skin cancer, with all hyperparameters kept constant.

It was found through empirical testing that the ResNet50 architecture was generally more accurate in the classification task. It can be theorized that this is due to the architecture leveraging "skip connections" to be more parameter-efficient than VGG16, while also being a deeper CNN. The performance gains that ResNet50 demonstrates over the VGG model could also be attributed to it being a more recent implementation of CNNs, having been designed to avoid the pitfalls of the VGG models themselves, as well as problems that plague other deep CNNs such as the vanishing gradient problem.

However, an analysis of the confusion table for both models led to a possible finding of overfitting in ResNet50, which did not seem to be as prominent in VGG16. This could again be due to the deeper architecture of ResNet50, which tends to result in an over-dependence and specialization on the training data when compared to shallower networks. The distribution of correct predictions made by VGG16 was more diverse, and could be proof of the fact that given a balanced dataset, it would outperform the ResNet50.

Hopefully, this exploration has aided further development and exploration

of reliable Computer Aided Diagnosis systems. I am excited to see future technological breakthroughs in this field, hopefully powered by computer vision.

Appendices

A Accuracy table

Table 2: Model Predictions and Accuracy

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
0	6	4	False	False
0	5	4	False	False
0	5	4	False	False
0	1	0	False	True
0	4	4	False	False
0	0	0	True	True
0	0	3	True	False
0	5	4	False	False
0	0	5	True	False
0	1	1	False	False
0	1	5	False	False
0	0	5	True	False
0	5	5	False	False
0	1	0	False	True
0	0	5	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
1	4	4	False	False
1	4	4	False	False
1	5	1	False	True
1	1	4	True	False
1	4	4	False	False
1	3	3	False	False
1	1	1	True	True
1	1	3	True	False
1	4	3	False	False
1	3	1	False	True
1	1	1	True	True
1	1	5	True	False
1	3	3	False	False
1	1	1	True	True
1	1	4	True	False
1	4	4	False	False
1	1	1	True	True
1	1	1	True	True
1	1	4	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
1	5	5	False	False
1	1	1	True	True
1	1	1	True	True
1	1	4	True	False
1	1	4	True	False
1	4	4	False	False
1	1	4	True	False
1	1	1	True	True
1	5	5	False	False
1	1	1	True	True
1	1	1	True	True
1	1	4	True	False
1	1	1	True	True
1	1	1	True	True
1	1	1	True	True
1	1	5	True	False
1	5	4	False	False
1	5	4	False	False
1	1	1	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
1	1	1	True	True
1	1	4	True	False
1	1	1	True	True
1	4	4	False	False
1	4	4	False	False
1	4	5	False	False
1	1	1	True	True
1	1	4	True	False
1	5	1	False	True
1	4	4	False	False
1	1	1	True	True
1	1	1	True	True
1	4	4	False	False
1	1	2	True	False
1	5	4	False	False
1	4	4	False	False
1	1	4	True	False
1	1	4	True	False
1	1	4	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
1	0	4	False	False
1	4	3	False	False
1	4	4	False	False
1	4	4	False	False
1	3	3	False	False
2	1	4	False	False
2	4	4	False	False
2	4	4	False	False
2	2	2	True	True
2	5	5	False	False
2	1	4	False	False
2	4	4	False	False
2	4	2	False	True
2	5	3	False	False
2	1	2	False	True
2	5	4	False	False
2	5	4	False	False
2	1	4	False	False
2	4	2	False	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
2	2	2	True	True
2	4	4	False	False
3	3	3	True	True
3	3	4	True	False
3	5	3	False	True
3	4	4	False	False
3	3	3	True	True
3	3	4	True	False
3	3	3	True	True
3	3	3	True	True
3	4	4	False	False
3	4	4	False	False
3	5	5	False	False
3	5	3	False	True
3	4	4	False	False
3	4	4	False	False
3	3	4	True	False
3	4	4	False	False
3	4	4	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	4	True	False
3	4	4	False	False
3	4	3	False	True
3	4	4	False	False
3	4	4	False	False
3	5	3	False	True
3	3	4	True	False
3	4	4	False	False
3	3	3	True	True
3	1	3	False	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	4	4	False	False
3	3	3	True	True
3	5	4	False	False
3	3	3	True	True
3	4	3	False	True
3	3	3	True	True
3	4	4	False	False
3	1	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	3	True	True
3	5	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	3	True	True
3	3	3	True	True
3	3	4	True	False
3	4	4	False	False
3	3	4	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	4	4	False	False
3	3	3	True	True
3	4	4	False	False
3	5	4	False	False
3	0	3	False	True
3	3	3	True	True
3	4	4	False	False
3	3	4	True	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	3	True	True
3	4	3	False	True
3	4	4	False	False
3	3	4	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	5	True	False
3	5	3	False	True
3	4	3	False	True
3	5	3	False	True
3	3	3	True	True
3	4	4	False	False
3	4	4	False	False
3	5	5	False	False
3	3	3	True	True
3	4	3	False	True
3	4	3	False	True
3	4	4	False	False
3	3	3	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	5	5	False	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	3	True	True
3	3	3	True	True
3	3	3	True	True
3	5	4	False	False
3	3	4	True	False
3	4	4	False	False
3	4	4	False	False
3	4	4	False	False
3	5	3	False	True
3	3	3	True	True
3	3	3	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
3	5	4	False	False
3	4	4	False	False
3	4	4	False	False
3	3	3	True	True
3	3	3	True	True
3	3	5	True	False
3	3	5	True	False
3	3	3	True	True
3	4	4	False	False
3	5	3	False	True
3	4	4	False	False
3	4	4	False	False
3	3	4	True	False
3	3	3	True	True
3	4	3	False	True
3	5	4	False	False
3	3	3	True	True
3	3	3	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	1	True	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	1	1	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	**VGG Prediction**	**ResNet Prediction**	**VGG Correct**	**ResNet Correct**
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	1	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	6	4	False	True
4	4	4	True	True
4	1	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	1	True	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	5	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	1	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	1	4	False	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	3	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	5	True	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	5	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	1	1	False	False
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	5	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	3	3	False	False
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	3	4	False	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	5	4	False	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	7	7	False	False
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	3	4	False	True
4	7	4	False	True
4	4	3	True	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	3	5	False	False
4	3	4	False	True
4	5	4	False	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	5	4	False	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	7	4	False	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	3	True	False
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	3	3	False	False
4	5	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	4	False	True
4	5	5	False	False
4	4	4	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	5	5	False	False
4	4	4	True	True
4	4	4	True	True
4	5	5	False	False
4	3	4	False	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
4	4	4	True	True
5	5	5	True	True
5	5	4	True	False
5	4	5	False	True
5	5	5	True	True
5	1	1	False	False
5	4	4	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	5	3	True	False
5	0	4	False	False
5	4	7	False	False
5	4	4	False	False
5	3	4	False	False
5	5	4	True	False
5	4	4	False	False
5	4	4	False	False
5	4	4	False	False
5	1	2	False	False
5	3	3	False	False
5	4	5	False	True
5	3	3	False	False
5	5	5	True	True
5	5	4	True	False
5	5	5	True	True
5	5	5	True	True
5	4	4	False	False
5	3	3	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	4	4	False	False
5	4	4	False	False
5	5	3	True	False
5	5	5	True	True
5	4	4	False	False
5	4	4	False	False
5	4	5	False	True
5	4	4	False	False
5	5	5	True	True
5	5	4	True	False
5	4	4	False	False
5	4	4	False	False
5	5	5	True	True
5	4	4	False	False
5	5	5	True	True
5	5	5	True	True
5	5	4	True	False
5	4	4	False	False
5	3	4	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	4	4	False	False
5	3	5	False	True
5	5	5	True	True
5	5	5	True	True
5	5	1	True	False
5	4	4	False	False
5	4	4	False	False
5	5	5	True	True
5	4	4	False	False
5	5	5	True	True
5	5	5	True	True
5	1	5	False	True
5	1	4	False	False
5	5	5	True	True
5	1	4	False	False
5	3	4	False	False
5	3	3	False	False
5	4	4	False	False
5	5	4	True	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	5	5	True	True
5	1	4	False	False
5	1	5	False	True
5	4	4	False	False
5	5	4	True	False
5	4	4	False	False
5	4	4	False	False
5	4	4	False	False
5	5	5	True	True
5	4	4	False	False
5	4	4	False	False
5	0	4	False	False
5	5	5	True	True
5	4	4	False	False
5	4	4	False	False
5	4	5	False	True
5	4	4	False	False
5	4	4	False	False
5	5	5	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	5	4	True	False
5	3	4	False	False
5	5	4	True	False
5	4	4	False	False
5	5	5	True	True
5	4	4	False	False
5	4	4	False	False
5	4	4	False	False
5	5	4	True	False
5	3	3	False	False
5	3	3	False	False
5	4	4	False	False
5	3	3	False	False
5	5	5	True	True
5	5	4	True	False
5	5	5	True	True
5	3	3	False	False
5	5	5	True	True
5	3	4	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	4	4	False	False
5	3	5	False	True
5	3	4	False	False
5	4	4	False	False
5	4	4	False	False
5	5	5	True	True
5	4	4	False	False
5	3	4	False	False
5	5	5	True	True
5	4	4	False	False
5	7	3	False	False
5	4	5	False	True
5	3	5	False	True
5	3	4	False	False
5	5	4	True	False
5	3	5	False	True
5	4	4	False	False
5	3	5	False	True
5	5	5	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
5	5	5	True	True
5	5	5	True	True
5	3	4	False	False
5	4	4	False	False
5	5	4	True	False
5	4	4	False	False
5	5	5	True	True
5	3	4	False	False
5	5	5	True	True
5	5	5	True	True
5	3	3	False	False
5	1	4	False	False
5	5	4	True	False
5	3	3	False	False
6	4	4	False	False
6	7	1	False	False
6	4	4	False	False
6	6	4	True	False
6	5	5	False	False

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
6	1	1	False	False
6	6	4	True	False
6	1	1	False	False
6	3	4	False	False
6	3	4	False	False
6	5	5	False	False
6	1	4	False	False
6	1	1	False	False
6	3	4	False	False
6	4	4	False	False
6	4	3	False	False
6	2	4	False	False
6	5	4	False	False
6	4	4	False	False
6	5	5	False	False
6	5	4	False	False
6	1	5	False	False
6	4	6	False	True
7	7	7	True	True

Continued on next page

Table 2 – *Continued from previous page*

True Label	VGG Prediction	ResNet Prediction	VGG Correct	ResNet Correct
7	7	7	True	True
7	7	7	True	True
7	7	7	True	True
7	4	2	False	False
7	7	7	True	True
7	7	7	True	True
7	7	7	True	True
7	7	7	True	True
7	7	7	True	True
7	7	4	True	False
7	7	7	True	True
7	1	7	False	True
7	7	7	True	True
7	1	1	False	False
7	1	3	False	False
7	7	7	True	True
7	7	7	True	True

B Model Summaries and direct link

Link to Drive folder with both models can be found **here**.

B.1 ResNet50 summary

```
Total params: 23604104 (90.04 MB)
Trainable params: 23550984 (89.84 MB)
Non-trainable params: 53120 (207.50 KB)
```

B.2 VGG16 summary

```
Total params: 14718792 (56.15 MB)
Trainable params: 14718792 (56.15 MB)
Non-trainable params: 0 (0.00 Byte)
```

C Code used

C.1 Code used to preprocess data

```
import os
import shutil
import pandas as pd
from sklearn.model_selection import train_test_split

# Define paths
```

```
metadata_path = "C:/Users/vivaa/Downloads/
    ↪ ham10000_metadata_2024-02-11.csv"
images_directory = r"C:\Users\vivaa\Downloads\ISIC-
    ↪ images"
dataset_directory = r"C:\Users\vivaa\Desktop\IB\Computer
    ↪  Science HL\Extended essay\Dataset"

# Read the metadata file
metadata = pd.read_csv(metadata_path)


train_val, test = train_test_split(metadata, test_size
    ↪ =0.10, stratify=metadata['diagnosis'],
    ↪ random_state=42)
train, val = train_test_split(train_val, test_size
    ↪ =0.1667, stratify=train_val['diagnosis'],
    ↪ random_state=42)


# Organize the dataset
def organize_dataset(dataset_split, dataset_split_name):
    for _, row in dataset_split.iterrows():
        src_file = os.path.join(images_directory, row['
            ↪ isic_id'] + '.jpg')
```

```
23
24          dest_dir = os.path.join(dataset_directory,
                ↪ dataset_split_name, row['diagnosis'])
25          if not os.path.exists(dest_dir):
26              os.makedirs(dest_dir)
27
28          dest_file = os.path.join(dest_dir, row['isic_id'
                ↪ ] + '.jpg')
29
30          shutil.copy(src_file, dest_file)
31
32
33  # Apply the organization function to each split
34  organize_dataset(train, 'train')
35  organize_dataset(val, 'val')
36  organize_dataset(test, 'test')
37
38  print("Dataset organization complete.")
```

C.2 Code used to train the two models

```
1  import numpy as np
2  import tensorflow as tf
3  from keras.preprocessing.image import ImageDataGenerator
4  from keras.applications import ResNet50, VGG16
```

```
from keras.layers import Dense, GlobalAveragePooling2D
from keras.models import Model
from keras.callbacks import EarlyStopping,
    ↪ ModelCheckpoint

IMG_WIDTH, IMG_HEIGHT = 224, 224
BATCH_SIZE = 32

train_dir = '/content/Dataset/train'
val_dir = '/content/Dataset/val'
test_dir = '/content/Dataset/test'

train_datagen = ImageDataGenerator(rescale=1./255,
    ↪ rotation_range=40, width_shift_range=0.2,
height_shift_range=0.2, shear_range=0.2, zoom_range=0.2,
horizontal_flip=True, fill_mode='nearest')
val_test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    ↪ train_dir, target_size=(IMG_WIDTH, IMG_HEIGHT),
batch_size=BATCH_SIZE, class_mode='categorical')
val_generator = val_test_datagen.flow_from_directory(
    ↪ val_dir, target_size=(IMG_WIDTH, IMG_HEIGHT),
```

```
batch_size=BATCH_SIZE, class_mode='categorical', shuffle
    ↪ =False)
test_generator = val_test_datagen.flow_from_directory(
    ↪ test_dir, target_size=(IMG_WIDTH, IMG_HEIGHT),
batch_size=BATCH_SIZE, class_mode='categorical', shuffle
    ↪ =False)

def create_model(base_model, num_classes):

    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    predictions = Dense(num_classes, activation='softmax
        ↪ ')(x)
    model = Model(inputs=base_model.input, outputs=
        ↪ predictions)
    return model

num_classes = len(train_generator.class_indices)

resnet_base = ResNet50(weights='imagenet', include_top=
    ↪ False, input_shape=(IMG_WIDTH, IMG_HEIGHT, 3))
resnet_model = create_model(resnet_base, num_classes)

```

```
vgg_base = VGG16(weights='imagenet', include_top=False,
    ↪ input_shape=(IMG_WIDTH, IMG_HEIGHT, 3))
vgg_model = create_model(vgg_base, num_classes)

resnet_model.compile(optimizer='adam', loss='
    ↪ categorical_crossentropy', metrics=['accuracy'])
vgg_model.compile(optimizer='adam', loss='
    ↪ categorical_crossentropy', metrics=['accuracy'])


early_stopping = EarlyStopping(monitor='val_loss',
    ↪ patience=10)
resnet_checkpoint = ModelCheckpoint('resnet_model.keras'
    ↪ , monitor='val_loss', save_best_only=True)
vgg_checkpoint = ModelCheckpoint('vgg_model.keras',
    ↪ monitor='val_loss', save_best_only=True)

history_resnet = resnet_model.fit(train_generator,
    ↪ epochs=25, validation_data=val_generator,
callbacks=[early_stopping, resnet_checkpoint])

history_vgg = vgg_model.fit(train_generator, epochs=25,
    ↪ validation_data=val_generator,
callbacks=[early_stopping, vgg_checkpoint])
```

C.3 Code used to calculate the accuracy of the two models

C.4 Code used to calculate the F1 Score and the Confusion matrix

```
import numpy as np
from sklearn.metrics import f1_score, confusion_matrix
from keras.models import load_model
from keras.preprocessing.image import ImageDataGenerator
import pandas as pd


model1 = load_model('/content/drive/MyDrive/Models/
    ↪ resnet_model.keras')
model2 = load_model('/content/drive/MyDrive/Models/
    ↪ vgg_model.keras')


test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_directory(
    '/content/Dataset/test',
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
```

```
18      shuffle=False)
19
20
21  true_labels = test_generator.classes
22
23  test_steps_per_epoch = np.math.ceil(test_generator.
        ↪ samples / test_generator.batch_size)
24
25  predictions1 = model1.predict(test_generator, steps=
        ↪ test_steps_per_epoch)
26  predictions1_binary = np.argmax(predictions1, axis=1)
27
28  predictions2 = model2.predict(test_generator, steps=
        ↪ test_steps_per_epoch)
29  predictions2_binary = np.argmax(predictions2, axis=1)
30
31  f1_score1 = f1_score(true_labels, predictions1_binary,
        ↪ average='weighted')
32  f1_score2 = f1_score(true_labels, predictions2_binary,
        ↪ average='weighted')
33
34  data = {
35      'Model': ['Model 1', 'Model 2'],
36      'F1 Score': [f1_score1, f1_score2]
```

```
}

comparison_df = pd.DataFrame(data)

excel_path = '/content/drive/MyDrive/model_comparison
    ↪ (2).xlsx'
comparison_df.to_excel(excel_path, index=False)

print("F1 scores comparison saved to Excel.")
```

References

Agatonovic-Kustrin, S. and Beresford, R. (2000). Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, 22(5):717–727.

Ali, A. (2018). Logistic regression with practical implementation. *The Art of Data Science*.

Aslam, N. (2022). Data representation in neural networks- tensor. *The Art of Data Science*.

Bishop, C. M. (1994). Neural networks and their applications. *Review of Scientific Instruments*, 65(6):1803–1832.

Cleveland, J. (2018). Overfitting vs underfitting: Difference explained. *The Art of Data Science*.

Dodge, S. and Karam, L. (2016). Understanding how image quality affects deep neural networks.

Ehteshami Bejnordi, B., Veta, M., Johannes van Diest, P., van Ginneken, B., Karssemeijer, N., Litjens, G., van der Laak, J. A., Hermsen, M., Manson, Q. F., Balkenhol, M., et al. (2017). Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *JAMA*, 318(22):2199–2210.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M.,

and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118.

IBM (n.d.). Computer vision.

Karpathy, A. (2018a). Convolutional neural networks for visual recognition. *The Art of Data Science*.

Karpathy, A. (2018b). Neural networks. *The Art of Data Science*.

Learned-Miller, E. G. (2011). *Introduction to computer vision*. University of Massachusetts, Amherst.

Rawat, W. and Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29(9):2352–2449.

Sanderson, G. (2017). Neural networks playlist.

Tschandl, P., Rosendahl, C., and Kittler, H. (2018). The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions.

Yamashita, R., Nishio, M., Do, R. K., and Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629.