Convolutional Neural Networks to Classify Skin Tumors

PHYSICS 3G03

Neel Gokhale

400126042

gokhalen@mcmaster.ca

April 30, 2022

Abstract

This report aims to evaluate different Convolutional Neural Networks (CNNs hereafter) models to classify benign and malignant tumors in skin moles. The dataset utilized involves 3600 images of skin moles afflicted by tumors. A self-made CNN model was built and compared to two industry-leading CNNs: ResNet50 and VGG19. For each model, the metrics of interest in comparison were training / validation accuracies, confusion matrix counts and test data prediction accuracy scores.

1 Theory

CNNs are a special category of neural networks, typically reserved for input types that involve image data. The goal of an effective CNN is to reduce the complexity of image inputs into processable matrices without losing key 'features' in these images.

By applying special matrices called kernels or filters to images, one can 'extract' specific features within each image like distinctive lines, spirals, color boundaries etc. Next, by 'pooling' or summarizing chunks of pixels of the original image into a compressed version, these features are amplified. These pooled layers are then passed through deep layers and finally classified using softmax activation. Figure 1 illustrates a flow diagram of how a CNN works, from the input layer all the way to the output classification layer.

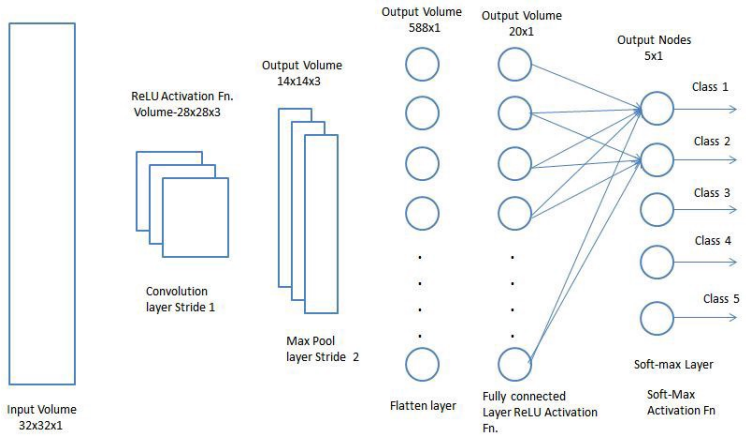

Figure 1: CNN flow diagram (sourced from (Saha 2018))

2 CNNs Used

As mentioned, this report evaluates three CNNs, two of which are developed by teams of researchers and are now widely used in the industry. In this section, we briefly review each network and their differentiating characteristics. It is important to note that in either case, we do not use their pre-trained instances (trained on the ImageNet database), but are only interested in using their model architecture. We train both VGG19 and ResNet50 with the same training datasets.

2.1 VGG19

The VGG (Visual Geometry Group) CNN model comes in a few variants (VGG16, VGG19, etc.) each referring to the number of layers involved in the network. VGG19 contains 19 layers: 16 convolution layers, 3 fully connected layer, 5 max-pooling layers and 1 softmax layer.

The VGG networks were a ground-breaking improvement for image-recognition and segmentation applications. They yielded high benchmarks in predicting large datasets of a diverse range of images. The VGGs however suffered from a critical issue which was apparent once additional layers were added to the existing models. During the backpropogation of the gradient through each layer, the updates to the weights have a decremental influence. By the time the algorithm reaches the top-most layers, the gradient has almost no effect and this considerably lengthens the training process. This is referred to as the 'vanishing gradient' problem, and is what restrains most applications of VGG models to a limited amount of layers.

2.2 ResNet50

Deep Residual Network, popularly coined ResNet is a super-deep CNN that solved the vanishing gradient problem, making it possible to build models with up to thousands of layers. ResNet models (ResNet50, ResNet90, etc.) also follow the naming-convention of VGG models, in that their accompanying numbers indicate the number of layers utilized.

In its initial training stage, the ResNet creates several convolutional layers but does not use them. ResNet then re-trains in the second stage, this time employing all of the 'residual' layers. The second training stage identifies additional features within a permissable amount of training time, given that the most basic features have already been identified in the previous, shallow CNN model. As a result, ResNet models can employ several sets of layers without running into a diminishing gradient.

3 Program Architecture

In this section, we review the dataset utilized and the structure of the self-made CNN model.

3.1 Datasets Used

The dataset utilized for this project was a repository of images of skin moles and retrieved from (Fancioni 2019). These images were ultimately collected and archived by the International Skin Imaging Collaboration (ISIC). These moles were afflicted with benign or malignant tumors.

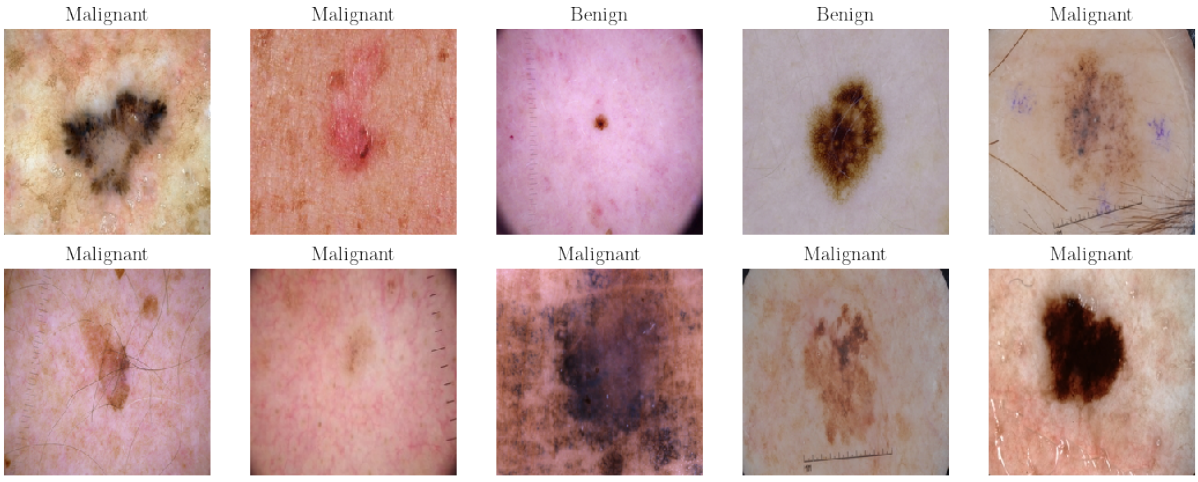

Figure 2: Examples from the dataset

Figure 2 contains a few examples from this dataset. Note that these are labeled 'Benign' and 'Malignant'.

These images are 224×224 pixels in dimension and when vectorized, contain 3 RGB values ranging from $0 - 255$.

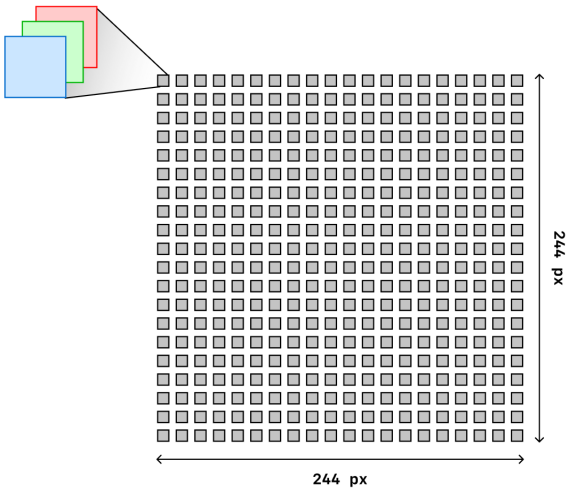

Figure 3: Data structure of an image

The data was stored in a `archive` directory with two separate folders for `training` and `testing`. In each grouping, the images were further grouped into `benign` and `malignant` subfolders. As a result, the first step during the model-building process involved collapsing and segmenting each of these folders into shuffled training and testing datasets.

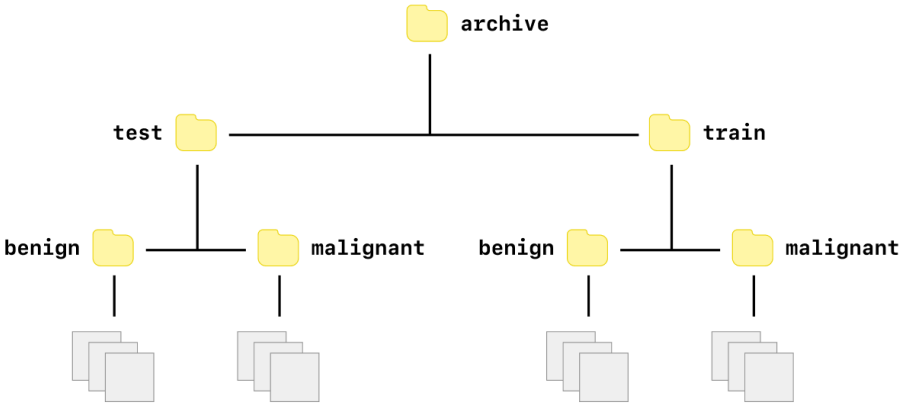

Figure 4: Folder structure of data repository

Label arrays `Y_train` and `Y_test` were also created based on the numbers of each image

class and were one-hot encoded for better applicability.

3.2 Model Structure

The self-made CNN model contained 9 layers and was structured in the following format and sequence:

1st Convolution layer - 2D layer that takes in a $224 \times 224 \times 3$ input shape, has 64 filters and applies `ReLu` activation.

1st Pooling layer - 2D max-pooling layer that reduces the dimensionality to $112 \times 112 \times 3$.

1st Dropout layer - a layer that masks 25% of neurons and leaves others applicable to following layers.

2nd Convolution layer - 2D layer that takes in a $112 \times 112 \times 3$ input shape, has 64 filters and applies `ReLu` activation.

2nd Pooling layer - 2D max-pooling layer that reduces the dimensionality to $56 \times 56 \times 3$.

2nd Dropout layer - a layer that masks 25% of neurons and leaves others applicable to following layers.

Flattening layer - flattens the input shape to a single dimensioned arrays

Dense layer - a 128 neuron dense layer that applies `ReLu` activation.

Output Dense layer - a 2 neuron output layer that applies the `softmax` activation function.

4 Results

Both `tensorflow` and `sklearn` libraries were used for the machine learning processes, while `numpy, pandas, PIL` and `pickle` were used for graphing, storing and handling image data. In each case, each network was trained under 30 epochs with a batch-size of 64 and the Adam optimizer was employed with a learning rate of $\lambda = 1 \times 10^{-5}$. The training data was split into training and validation sets ($80 - 20\%$ split). During the training process, the metrics tracked were the training and validation accuracy and confusion matrix counts. Finally, the trained model was used to predict the testing sample data and an accuracy score was generated.

4.1 Self-made CNN

The accuracy chart of the self-made CNN over 30 epochs is as follows. We observe that although the training accuracy hits above the 90% range, the validation accuracy stabilizes at around 80%.

Figure 5: Self-made CNN accuracy over epochs

Looking at the confusion matrix counts, the validation counts follow a much more erratic ascent to fully true positive and true negative counts, while the training counts are much more smoother in transition. We expect some degree of over-fitting based on these results.

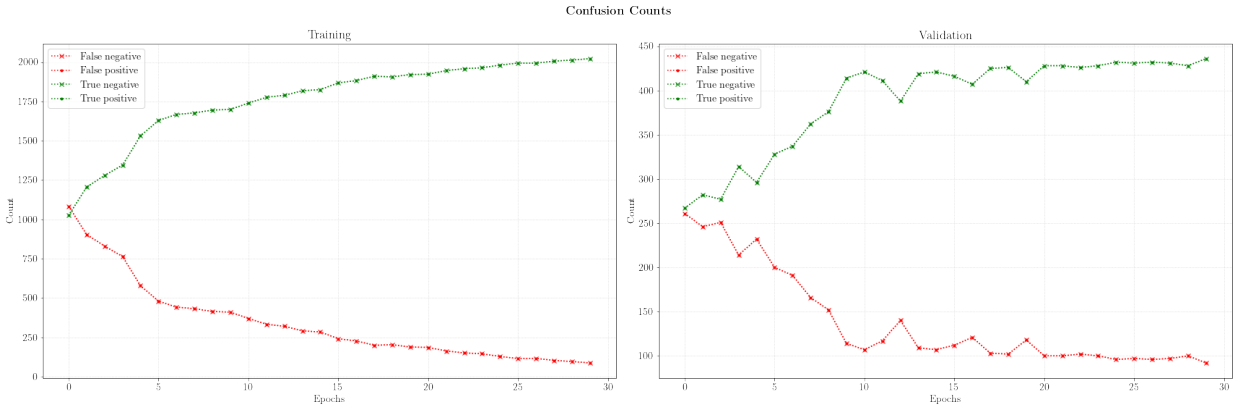

Figure 6: Self-made CNN confusion matrix counts over epochs

4.2 ResNet50

Below we observe the same graphics, this time tracking the metrics for the ResNet50 implementation.

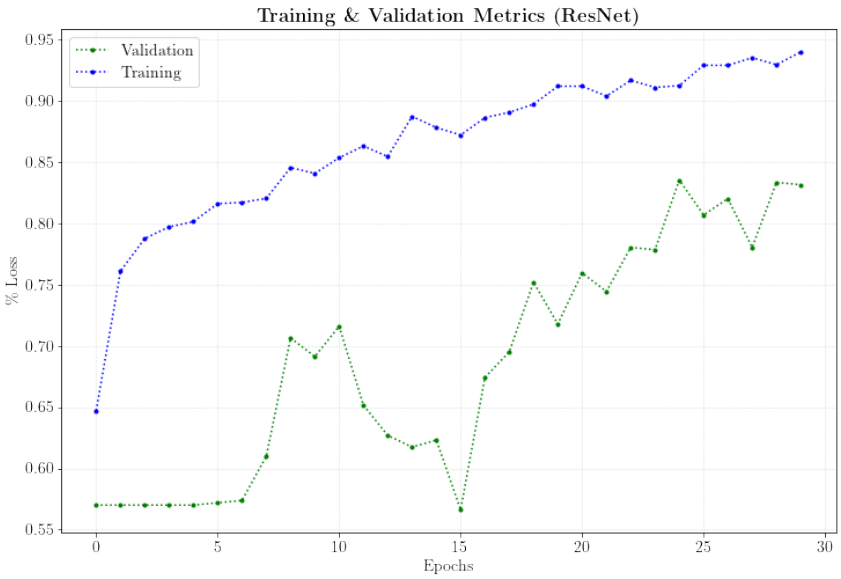

Figure 7: ResNet50 accuracy over epochs

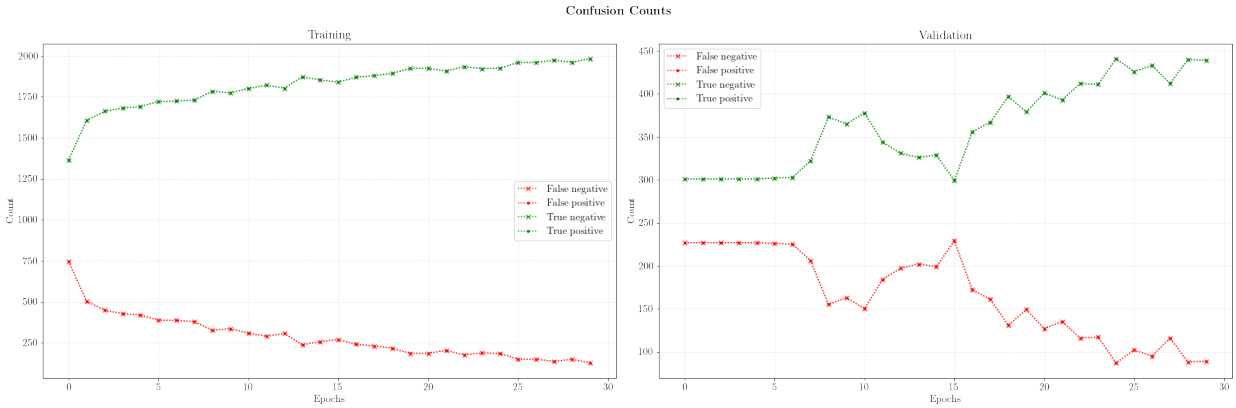

Figure 8: ResNet50 confusion matrix counts over epochs

The ResNet50 model has a much harder time classifying the dataset even though it does achieve a higher validation accuracy over 30 epochs.

4.3 VGG19

Below we observe the same graphics, this time tracking the metrics for the VGG19 implementation.

Figure 9: VGG19 accuracy over epochs

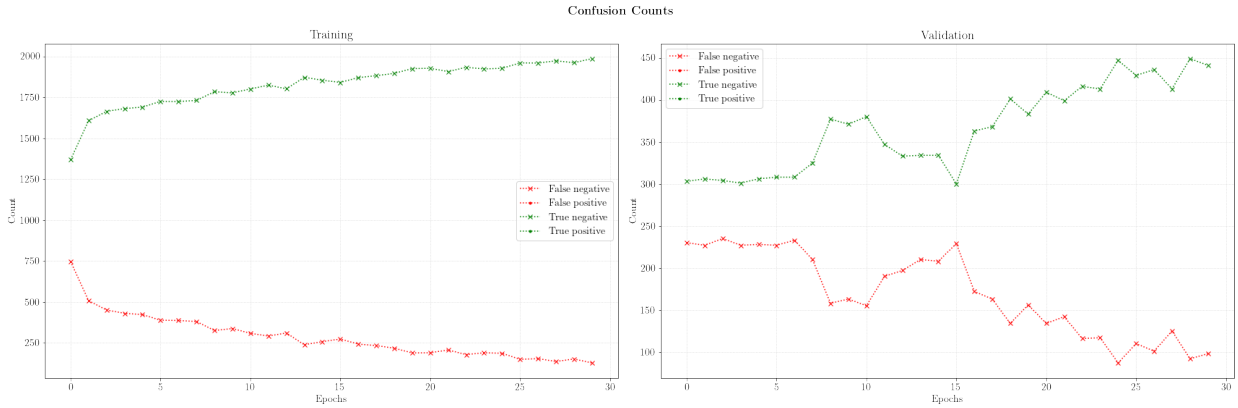

Figure 10: VGG19 confusion matrix counts over epochs

As with the ResNet model, the VGG19 model also has a much harder time classifying the dataset and it results in the same validation accuracy over 30 epochs as the self-made CNN model.

5 Final Comparison

The following barchart compares the accuracy scores of each implementation after running each model through a prediction process over all the testing sample data. Note that this data was not previously used during the training process, so it yields a valuable overall score

of each model.

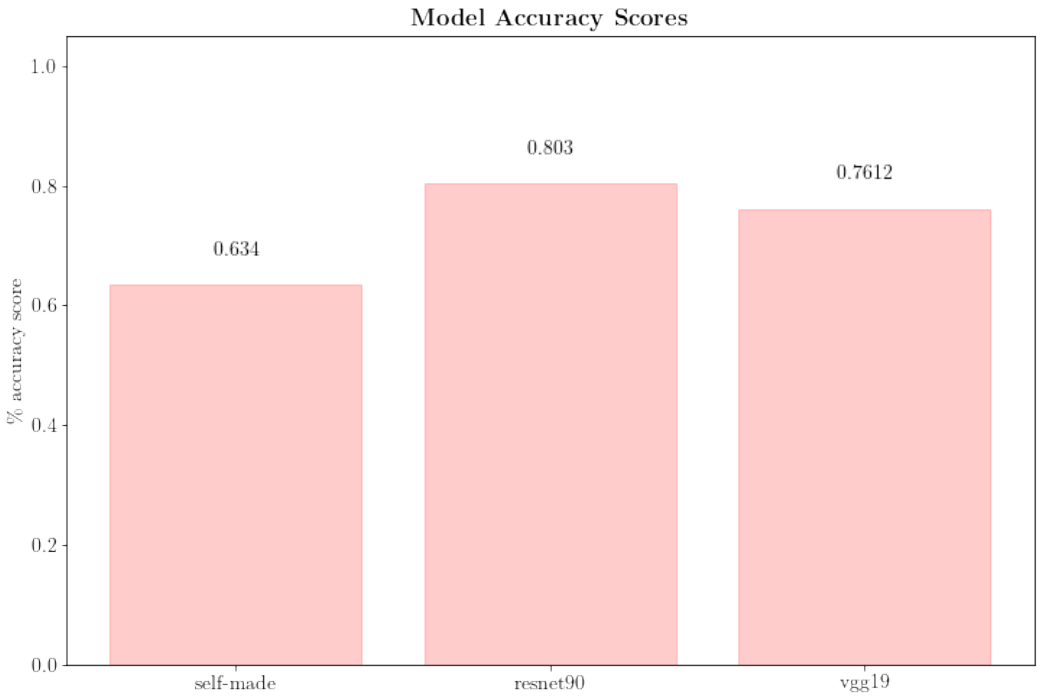

Figure 11: Model accuracy scores

As expected, due to overfitting, the self-made model does not perform well in a blind-test scenario, with new data. Because the ResNet and VGG models employ much more complex feature-recognition layers, they gather more information over the same amount of epochs and are less likely to overfit. Looking forward, a few ways to improve the performance of the self-made model is to employ cross-validation to increase the variation within the same dataset. Another key implementation that could benefit all of the above mentioned models is feature clustering, whereby an image is modified to highlight certain features (such as dark spots and streaks) and other more prevalent features (in this case, hair on the surrounding skin) are masked so that the input data is more focused on the subject at hand - the skin moles. Further testing is required to confirm this hypothesis.

References

Fancioni, C. (2019). *Skin Cancer - Malignant vs. Benign*. Retrieved: 2022-04-29. URL: https://www.kaggle.com/datasets/fanconic/skin-cancer-malignant-vs-benign?resource=download.

Saha, S. (2018). *A Comprehensive Guide to Convolutional Neural Networks*. Retrieved: 2022-04-29. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.