Deeper Networks for Image Classification

Introduction

Deep neural networks (DNNs) have, in recent years have dominated the state-of-the-art for image classification, reporting quality results and vastly outperforming shallower, more conventional Convolutional Neural Networks (CNNs). In this paper, I will compare 3 notable models from recent years - ResNet (2015), VGG (2015) and DenseNet (2017) in their effectiveness when trained and tested on the following common image-classification benchmark datasets: MNIST, Fashion-MNIST and CIFAR-10. MNIST is a digit-recognition task wherein models must classify hand-drawn digit samples between 1 and 9; a simple benchmark for models of this scope. Fashion-MNIST is an object classification task that requires models to label slightly more complex objects that have been highly normalised. CIFAR-10 is a more difficult object identification task that contains images with more noise and much more complex targets, some chosen specifically to share common visual features in order to increase task difficulty.

Subsequently, I will analyse their training and testing accuracies for each experiment along with their testing average loss values, in order to determine potential strengths and weaknesses - diagnosing the reasoning for this. Additionally, trends and anomalies synchronous with DNN models' performance will be identified and analysed.

Critical Analysis/ Related Work

Simonyan et al proposed an architecture for large-scale image recognition [9]; a DNN utilising a large number of vary small (3x3) convolutional layers allowing for the depth of the model to be increased to 16 & 19 layers. The model builds on the architecture of previous deep models like AlexNet [2] but is able to increase the depth considerably.

Each of the convolutional layers is implemented with a stride of 1 such that the spatial resolution can be preserved after each convolution. The decision to use many smaller (3x3) convolutions was a result of a comprehensive analysis performed by Simonyan et al, where many smaller convolutions were compared to fewer larger convolutions. It was concluded that using many smaller convolutions was the most effective architecture.

These convolutional layers are followed by 3 fully connected layers, with the first 2 layers having 4096 channels. The final fully connected layer contains 1000 channels since the model is able to perform 1000-way classification and each channel can then represent the probability of a given class. Each of the hidden layers utilise the ReLU activation function in order to introduce non-linearity into the network. As a result of the increased depth to 16 or 19, the model's layers are grouped into blocks, similar to ResNet. This means that when defining the deeper configurations of VGG, each layer does not need to be defined explicitly, and instead, a block can be added containing 3 layers.

Simonyan et al present 6 configurations of the model A-E with different depths. Table 3 contains the performance of each of these configurations with regards to their validation error percentage. The validation error percentage shows an inverse correlation between average loss percentage and depth of the model, with A (the shallowest model) reporting the highest error of 29.6 and E (the deepest) reporting the lowest error of 25.5.

The VGG model has been applied to a wide range of image classification tasks in different industries, for example, automatic image annotation wherein the model classifies the subject of the image and labels it, PCB board error detection [14] where the model is trained in order to recognise features prevalent in PCB boards with errors and object detection from satellite images [11].

The ResNet model proposed by He et al [9] consolidates the idea of the residual network for image classification. The architecture proposed consists of the stacking of Residual 'blocks' of layers in order with the objective of allowing the creation of significantly deeper models without falling into the pitfall of gradient degradation. Instead of placing blind faith into the hypothesis that a stack of layers will directly fit an underlying mapping, they are instead, explicitly fit to do so. Each of these stacks of layers or 'blocks' is defined by the notation $y = F(x,\{Wi\}) + Wsx.$, "where F(x) denotes the output vector of the 'block' and x is a 'short-cut connection'". In order to utilise the benefits of stacking layers thusly, each 'block' must contain more than one single layer, and more than the 3 layers exhibited by He et al in this paper are deemed possible.

He et al tested two models; a plain model and a ResNet, each with two different depths (18 and 34) on the ImageNet 2012 dataset and reported the effect of the increased model depth for each model. Table 2 [9] shows that the plain-18 model recorded a validation average loss percentage of 27.94 and the increased-depth plain-34 model, an average loss of 28.54 - an increased average loss percentage of 0.6. This is compared to the ResNet-18 model recording a percentage average loss of 27.88 but consequently, the deeper ResNet model recorded a reduced average loss of 25.03, a significant decrease in average loss percentage of 3.85, proving the ResNet architecture to reverse the effect of degradation on networks with increased depth on the given models' classification performance. ResNet has been used in a number of different applications, most notably in the medical imaging field where the model is used in colonoscopy scans [7] and in Alzheimer's detection [8] due to its ability to classify complex features in images.

The DenseNet model, proposed by Huang et al in 2018 [6] builds further on the advancements of the previous state-of-the-art by utilising a dense approach to the CNN as a method of negating the 'vanishing gradient' problem. The architecture consists of a number of 'dense blocks' with a unique connectivity pattern, where each layer receives signal from all of its preceding layers. Features are concatenated with those of the current layer and passed through a series of non-linear transformations as shown in Figure 2 of [6]. An advantage of this structure is the strong-gradient flow where error is able to be back-propagated through the network, to earlier layers, more directly, meaning that earlier layers are able to get more reliable supervision.

One problem with this architecture was that deeper layers grew increasingly wide and expensive to compute so this was addressed with the use of 'bottleneck' layers which are relatively cheap as they utilise a convolution of 1x1 to reduce the dimensions of the channels. This results in a much narrower architecture, meaning that increased depth is possible without the number of parameters growing massively and improved performance as a consequence.

The DenseNet model became the new state-of-the-art for the CIFAR-10 dataset (with data augmentation), with the DenseNet250 configuration achieving a test error of 3.6%, beating the previous state-of-the-art by 0.8%. Additionally, the DenseNet model beat the state-of-the-art for the dataset without data augmentation, with the DenseNet100 and DenseNet250 accomplishing test errors of 5.9% and 5.2% respectively. The increased performance without data augmentation is a result of the model's ability to maintain simple features, deeper into the model, resulting in improved performance in scenarios where training data isn't sufficient. Furthermore, the DenseNet100 model contained significantly fewer parameters; 0.8 million with the closest model for performance being the ResNet1001 which contained 10.2 million.

The DenseNet model has been applied to a number of different scenarios such as Automatic speech recognition, with the addition of BLSTM architecture [1] as well as electricity theft identification [5], where it has performed effectively.

Method / Model Description

The model implementation for these experiments was sourced/based on code found in open-source repositories with the exception of VGG. The VGG model was sourced from the Applications module of the Keras API for TensorFlow. The DenseNet121 model was sourced from [17] and the ResNet110 implementation was sourced from GitHub [16]. The models are trained in line with the training parameters outlined in their original papers in order to express the best performance achievable.

However, the batch size used in these experiments is set to 126 as a result of GPU memory limitations which may affect performances. The number of epochs has also been standardised to 100 in order to maintain consistency and reduce training time for some models.

The method of the experiments is to test ResNet, VGG and DenseNet on three of the most popular benchmark datasets – MNIST, Fashion-MNIST and CIFAR-10; identifying their training and validation accuracy as well as their average loss after each epoch, as shown in Figure 1. The MNIST and Fashion-MNIST will test the models' ability to complete standardised character and object recognition tasks. The CIFAR-10 dataset is a harder task, classifying less simple objects from images with no normalisation and some additional image clutter. Data augmentation is often used for such tasks to improve the quality of training and so the models will be tested with and without data-augmentation [7].

Experiments

Datasets

MNIST

The MNIST dataset is a large handwritten-digit dataset that is often utilised for training and evaluating image classification tasks. Each sample image is greyscale and of the dimensions 28x28. The MNIST dataset has been used as a benchmark test for image processing systems. The images are classified into 10 classes, representing digits of 1-10.

The dataset has been used as a benchmark test since its images are of low-resolution. This means that it can be used to train a model quickly and thus is an effective tool for testing different algorithms. Additionally, each sample is centred and normalised in order to improve consistency of results. Consequently, of this normalisation, the MNIST dataset will be used as a benchmark assessment for the experiments.

Each of these experiments will be run 3 times for each dataset and model in order to gain a more robust understanding of model performance in each experiment and reduce the potential impact of anomalies on results. Therefore, an average will be taken of the final training and test accuracies and average loss after the 3 iterations of experimentation.

Fashion-MNIST

Fashion-MNIST is a dataset consisting of greyscale images of garments from Zalando's articles. The dataset contains 60,000 training examples and a further 10,000 testing examples. Each image is formatted in greyscale with the dimensions 28x28 and contains a garment of clothing. The images are classified into 10 classes such as: Bag, Sneaker, Shirt, Sandal etc. The dataset is similar to the original MNIST dataset although the task is somewhat harder as a result of the objects consisting of more complex structure than single digits.

CIFAR-10

CIFAR-10 is a widely used dataset for the testing of Computer Vision models [4]. It contains 60,000 images with the dimensions 32x32 labelled within 10 different classes. Classes represent objects such as ships and trucks or animals like dogs and cats. This means that it can be used to train models for object detection.

Similarly, to the MNIST dataset, since the images are of low-dimensions, CIFAR-10 is an effective benchmark to train and test algorithms on, since the low-dimensionality of the samples allows for training and testing to take place more quickly than if they were of larger dimensions.

CIFAR-10 is the most difficult dataset to train DNNs on as the dataset contains clutter in the images and the target objects are much more complex in shape, with some classes such as those containing animals, sharing many very similar features such as legs and tails, making them harder to distinguish from each other. The images have undergone less normalisation. As a result, some the experiments will be undergone with (denoted by CIFAR-10+) and without data augmentation in order to measure the effects of such normalisation.

Testing Results

MNIST Experiments

After training on the MNIST dataset, the average accuracy and average loss for all models is very similar (+- 0.05 average testing accuracy) since the use of DNN models for this task would usually be excessive. In Table 1, the lowest average testing accuracy was produced by the ResNet110 (0.989) and the highest by DenseNet121 (0.994) . In addition, the training runs were very similar too - with the lowest accuracy after the first epoch being 0.696 for VGG19 as shown in Figure 4. DenseNet121 and ResNet110 achieved training accuracies of around 0.99 as shown in the experiment examples of Figures 2 and 3.

However, the testing average loss values have slightly more variation. ResNet recorded the highest average loss value for this experiment (0.730), substantially more than VGG19 (0.006) and DenseNet (0.034). This may be a consequence of the deep architecture of the ResNet model not allowing for average loss to be back-propagated as clearly to the shallower layers, not allowing them to learn quite as effectively as VGG19, which is significantly shallower, and DenseNet which utilises a different connection architecture for this reason.

Despite this, this experiment is not an effective one for comparing results of models of this depth and scope. Since all models converged within 15 epochs (as shown in an example experiments for all models in Figure 2, 3 and 4) as they were quickly able to fit the simpler data, and achieved minimum testing accuracies of 0.98, their results are all within such close proximity that it is difficult to derive much intuition from them.

Fashion-MNIST Experiments

Training on the Fashion-MNIST dataset is a more difficult task; evidenced by a broader range in experimental results of the 3 models as shown in Table 2. DenseNet121 again, achieved the highest average testing accuracy (0.993) compared to VGG19 and ResNet110 which achieved testing Accuracies of 0.910 and 0.882 respectively. However, the training accuracies for all models in this experiment remained high - VGG19 accomplished an average training accuracy of 1.000, whereas, ResNet110 and DenseNet121 accomplished training accuracies of 0.999. Despite this, the testing accuracies for VGG19 and ResNet110 were significantly lower than that of DenseNet121. The disparity between training and average testing accuracy suggests that VGG19 and ResNet121 overfit the training data in this experiment, as shown in Figures 7 and 8. Overfitting is characterised by a significantly reduced accuracy for unseen data as the model learns a function that corresponds too closely to the specific details of the training data, such that it isn't sufficiently general to represent the more general trends. The ResNet has been found to be likely to overfit in some contexts, particularly with a small number of training samples as a result of its increased depth [14]. Although the quantity of training samples here is sufficient, these samples are highly standardised and as a result, the features aren't sufficiently varied resulting in the ResNet overfitting.

DenseNet121 does not succumb to overfitting for this experiment however; the difference between final testing and average training accuracy is only 0.006. This is also shown in the original DenseNet paper by Huang et al [6]. This is likely as a result of the model's connectivity pattern being such that the model retains a large number of less complex features, deeper into the model and as such doesn't overfit to the complex and more specific features of the training set. As a result, the model does not overfit the training data (as can be seen in Figure 10, and as a consequence, achieves a smaller testing average loss.

CIFAR-10

When the experiment was carried out without data augmentation, the models significantly overfit the training data as shown in Table 3. ResNet110 overfit the training data by the most significant degree with a 0.344 difference between the training and testing accuracies. This disparity between training and testing accuracy can be seen throughout an entire training run as shown in Figure 11. In addition, the ResNet model scored the highest average loss value of 3.280. The VGG19 model also overfit the data substantially, although less so than ResNet110 which can be seen in Figure 12. VGG19 recorded a difference of 0.259 between training and average testing accuracy and so is significantly better than

ResNet110 in this experiment. DenseNet121 however, overfits the least with a training to average testing accuracy difference of 0.141 as seen in Figure 13.

Additionally, the class Testing Accuracies of Class 1 (Automobile) and class 9 (Truck) both have accuracies of 0.77 and 0.81 respectively, for a run of DenseNet121 on CIFAR-10, as shown in Table 4. This is because the two classes have very similar features such as wheels or windows etc. This is also evident in Figure 3.2 of Krizhevsky's book [3], that expresses an increased confusion between these two classes. This shows that without data augmentation, classes with similar visual features can be miss-labelled by DNNs as a result of overfitting specific features to specific classes.

CIFAR-10+ /Improvement

In order to maximise the performance of the deep image classification models on the CIFAR-10 dataset, data augmentation is often used. Mirroring/shifting are "widely used" as stated by Huang et al in [7] in order to promote the identification of more robust features by the models rather than placement or composition of features common to the images in this dataset specifically.

As shown in Table 6, ResNet110 achieved the lowest test accuracy (0.757) in addition to the highest testing average loss (0.1533). However, the improvement over the model's performance without data augmentation (0.655) (Table 3), the model achieved an improvement of 0.102. VGG19 also achieved an improved average testing accuracy (0.827) as a result of data augmentation, over the previous experiment (0.741), an improvement of 0.086. DenseNet, additionally, reported an improved average testing accuracy (0.877) over its previous accuracy of (0.845), an increase of 0.032.

Consequently, the overfitting of all models seen in the CIFAR-10 experiment without augmentation has improved dramatically in all models in the CIFAR-10+ experiment. The difference between testing and average training accuracy improved to 0.153 for VGG19 as is seen in Figure 14, a decrease in overall difference of 0.106. ResNet also reduced its overfitting – the difference between testing and average training accuracy improved to 0.129 (Figure 15), a decrease of 0.215 over the previous experiment. Lastly, DenseNet improved the average testing accuracy to 0.877 as Figure 16 shows, improving by 0.032 over the previous experiment.

Furthermore, the addition of data augmentation improved the class accuracies of classes 1 and 9 that previously had lower accuracies. The per-class accuracies of a given CIFAR-10+ experiment (0.77) and (0.81) as shown in Table 4, to 0.84 and 0.85 respectively as shown in Table 5. This is because the data augmentation promoted the learning of features that were more robust and as a result was able alleviate some of the confusion between classes.

Further Evaluation

Table 7 details the number of parameters, FLOPs and Depth of each model, utilising figures in Table 8 in the paper by Li et al [12] and Table 1 from the paper by Veit et al [3]. The table shows that ResNet110 has many fewer parameters (1.7 million) than VGG16 (134.2 million) whilst maintaining a significantly increased depth. As a result, the performance is significantly better (0.5x10^9) than VGG19 (1.55x10^10) meaning that ResNet110 is a much more efficient model as well as gaining the inherent benefits of its depth. This means that the ResNet110 model performs better as well as requiring significantly less computation. Performance is incredibly important when comparing DNN models since training time is a significant drawback to many models even when trained on a GPU.

DenseNet121, however, boasts the most depth out of all models considered and utilising less than half of the number of parameters of ResNet110. However, it does require more computation than the ResNet110, although this is a trade-off for better accuracy on many tasks.

Conclusion

All DNNs achieved high accuracy percentages for the MNIST dataset experiment; the task did not provide enough challenge to significantly test the capability of the DNNs and so their performance had low variance – close to 100% accuracy on seen and unseen data. However, the DenseNet did achieve the highest accuracy, followed by VGG19, then ResNet110. This ranking was consistent through all experiments where the DenseNet outperformed the VGG and ResNet, largely as a result of its reduced

propensity to overfit the training data as a result of its architecture retaining less complex features, deeper into the model. Additionally, its functionality allows for the error to be back propagated to early layers much more effectively and so it is able to maximise the efficiency of these layers. The DenseNet model also achieves these improved results with significantly fewer parameters than ResNet and VGG, making it much more efficient computationally as well as achieving better results. However, the tasks used for this paper do not play to the strengths of the ResNet as it is built with the intention of the increased depth allowing it to learn more complex visual features since the image classification tasks exhibited here do not contain such intricate subjects.

Despite this, VGG19 outperformed the ResNet on all experiments despite being significantly shallower. This is because, despite the reduced depth, the model contains 79 times more parameters than the ResNet. As a result of this, despite the marginally better performance, the training time and expense is significantly larger, so a trade-off is created between the two models with regards to speed and efficiency or improved accuracy on the tasks tested in this paper.

In conclusion, the general performance of all DNNs in this paper was very good; each of the models reported low average loss values and high training and testing accuracies with regards to the particular tasks utilised in this paper as each of the models have represented the state-of-the-art since their publication.

References

[1] A. Dutta and A. Zisserman, "The VIA Annotation Software for Images, Audio and Video", *27th ACM International Conference on Multimedia*, 2019. [Accessed 6 May 2020].

[2]A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet classification with deep convolutional neural networks", *Communications of the ACM*, vol. 60, no. 6, pp. 84-90, 2017. Available: 10.1145/3065386.

[3]A. Veit and S. Belongie, "Convolutional Networks with Adaptive Inference Graphs", *Computer Vision – ECCV 2018: 15th European Conference*, 2018. [Accessed 4 May 2020].

[4] A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*. 2009.

[5] B. Li, K. Xu, X. Cui, Y. Wang, X. Ai and Y. Wong, "Multi-scale DenseNet-Based Electricity Theft Detection", *CIC 2018: Intelligent Computing Theories and Application*, 2018. [Accessed 5 May 2020].

[6] G. Huang, Z. Liu, L. van der Maaten and K. Weinberger, "Densely Connected Convolutional Networks", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [Accessed 4 May 2020].

[7] J. Ker, L. Wang, J. Rao and T. Lim, "Deep Learning Applications in Medical Image Analysis", *IEEE Access (Volume: 6)*, 2017. [Accessed 5 May 2020].

[8] J. Näppi, T. Hironaka and H. Yoshida, "Detection of colorectal masses in CT colonography: application of deep residual networks for differentiating masses from normal colon anatomy", *SPIE 10575, Medical Imaging 2018: Computer-Aided Diagnosis*, 2018. [Accessed 5 May 2020].

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [Accessed 1 May 2020].

[10] K. Simonyan and A. Zisserman, "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION", *International Conference on Learning Representations*, 2015. [Accessed 1 May 2020].

[11] M. Napiorkowska, D. Petit and P. Marti, "Three Applications of Deep Learning Algorithms for Object Detection in Satellite Imagery", *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 2020. [Accessed 2 May 2020].

[12] M. Strake, P. Berh, T. Lohrenz and T. Fingscheidt, "Densenet Blstm for Acoustic Modeling in Robust ASR", *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018. [Accessed 2 May 2020].

[13] W. Li, X. Zhu and S. Gong, "Harmonious Attention Network for Person Re-Identification", *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. [Accessed 3 May 2020].

[14] Y. Li and J. Guo, "A VGG-16 Based Faster RCNN Model for PCB Error Inspection in Industrial AOI Applications", *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 2018. [Accessed 3 May 2020].

[15] Z. Wang, W. Yan and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline", *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017. [Accessed 7 May 2020].

Code References:

[16] Keras-team (2019) CIFAR-10 ResNet [Source code], DOI: https://github.com/keras-team/keras/tree/master/docs

[17] Asrst (2019) DenseNet (tf-keras) on CIFAR-10 [Source code], DOI: https://www.kaggle.com/asrsaiteja/densenet-tf-keras-on-cifar-10

Appendix

Tables

Model	Training Accuracy	Testing Accuracy	Testing Loss
VGG19	1.000	0.992	0.006
ResNet110	1.000	0.989	0.730
DenseNet121	0.999	0.994	0.034

Table 1, Average MNIST Experimental Results

Model	Training Accuracy	Testing Accuracy	Testing Loss
VGG19	1.000	0.910	1.034
ResNet110	0.999	0.882	0.069
DenseNet121	0.999	0.993	0.039

Table 2, Average Fashion-MNIST Experiment Results

Model	Training Accuracy	Testing Accuracy	Testing Loss
VGG19	1.000	0.741	2.704
ResNet110	0.999	0.655	3.280
DenseNet121	0.986	0.845	1.013

Table 3, CIFAR-10 Average Experimental Results

Class	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
Accuracy	0.82	0.77	0.82	0.89	0.88	0.87	0.91	0.87	0.84	0.81

Table 4, Example Per-Class Testing Accuracies for DenseNet121 on CIFAR-10

Class	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
Accuracy	0.82	0.84	0.85	0.9	0.87	0.89	0.92	0.87	0.85	0.85

Table 5, Example Class Testing Accuracies for DenseNet121 on CIFAR-10+

Model	Training Accuracy	Testing Accuracy	Testing Loss
VGG19	0.980	0.827	0.834
ResNet110	0.886	0.757	1.533
DenseNet121	0.954	0.877	0.532

Table 6, CIFAR-10+ Average Experimental Results

Model	No. of Parameters (Million)	FLOPs	Depth
VGG19	134.2	1.55x10^10	19
ResNet110	1.7	0.5x10^9	110
DenseNet121	0.8	3x10^9	121

Table 7, Model Size and Complexity

Figures

```
Epoch 1/100
469/469 [==============================] - 131s 279ms/step - loss: 1.7772 - accuracy: 0.6768 - val_loss: 1.4867 - val_accuracy: 0.7567
Epoch 2/100
469/469 [==============================] - 102s 218ms/step - loss: 1.3460 - accuracy: 0.8052 - val_loss: 1.3393 - val_accuracy: 0.8123
Epoch 3/100
469/469 [==============================] - 102s 218ms/step - loss: 1.2560 - accuracy: 0.8412 - val_loss: 1.2740 - val_accuracy: 0.8372
Epoch 4/100
469/469 [==============================] - 103s 219ms/step - loss: 1.2051 - accuracy: 0.8605 - val_loss: 1.2331 - val_accuracy: 0.8540
Epoch 5/100
469/469 [==============================] - 103s 220ms/step - loss: 1.1698 - accuracy: 0.8730 - val_loss: 1.2389 - val_accuracy: 0.8489
Epoch 6/100
469/469 [==============================] - 103s 220ms/step - loss: 1.1367 - accuracy: 0.8844 - val_loss: 1.2170 - val_accuracy: 0.8573
Epoch 7/100
469/469 [==============================] - 103s 220ms/step - loss: 1.1135 - accuracy: 0.8926 - val_loss: 1.1939 - val_accuracy: 0.8686
Epoch 8/100
469/469 [==============================] - 103s 220ms/step - loss: 1.0927 - accuracy: 0.9001 - val_loss: 1.1849 - val_accuracy: 0.8697
Epoch 9/100
469/469 [==============================] - 103s 220ms/step - loss: 1.0737 - accuracy: 0.9062 - val_loss: 1.1794 - val_accuracy: 0.8742
Epoch 10/100
469/469 [==============================] - 103s 220ms/step - loss: 1.0540 - accuracy: 0.9131 - val_loss: 1.2337 - val_accuracy: 0.8588
Epoch 11/100
469/469 [==============================] - 103s 219ms/step - loss: 1.0411 - accuracy: 0.9168 - val_loss: 1.1498 - val_accuracy: 0.8811
Epoch 12/100
469/469 [==============================] - 103s 219ms/step - loss: 1.0251 - accuracy: 0.9227 - val_loss: 1.1788 - val_accuracy: 0.8750
Epoch 13/100
469/469 [==============================] - 103s 220ms/step - loss: 1.0081 - accuracy: 0.9289 - val_loss: 1.1745 - val_accuracy: 0.8755
Epoch 14/100
469/469 [==============================] - 103s 220ms/step - loss: 0.9921 - accuracy: 0.9338 - val_loss: 1.1687 - val_accuracy: 0.8770
Epoch 15/100
469/469 [==============================] - 103s 220ms/step - loss: 0.9816 - accuracy: 0.9373 - val_loss: 1.1692 - val_accuracy: 0.8744
Epoch 16/100
469/469 [==============================] - 103s 220ms/step - loss: 0.9655 - accuracy: 0.9426 - val_loss: 1.2055 - val_accuracy: 0.8751
Epoch 17/100
469/469 [==============================] - 103s 220ms/step - loss: 0.9547 - accuracy: 0.9459 - val_loss: 1.1968 - val_accuracy: 0.8752
Epoch 18/100
469/469 [==============================] - 103s 220ms/step - loss: 0.9416 - accuracy: 0.9506 - val_loss: 1.2183 - val_accuracy: 0.8733
Epoch 19/100
469/469 [==============================] - 104s 222ms/step - loss: 0.9314 - accuracy: 0.9541 - val_loss: 1.2265 - val_accuracy: 0.8736
Epoch 20/100
469/469 [==============================] - 104s 222ms/step - loss: 0.9204 - accuracy: 0.9579 - val_loss: 1.2437 - val_accuracy: 0.8717
```

Figure 1, Example Training Output (Screenshot)

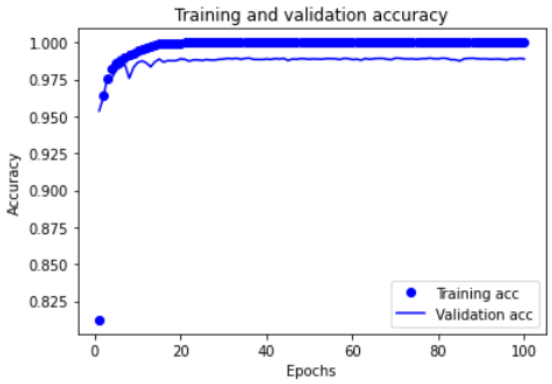

Figure 2, ResNet110 MNIST Training & Validation Accuracy

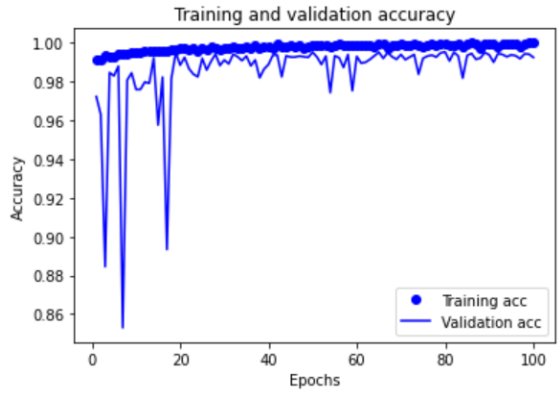

Figure 3, DenseNet121 MNIST Training & Validation Accuracy

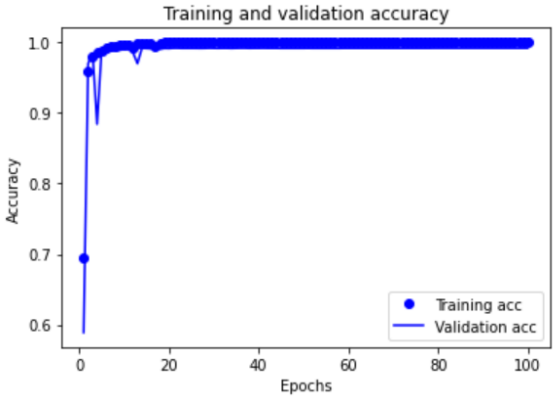

Figure 4, VGG19 MNIST Training & Validation Accuracy

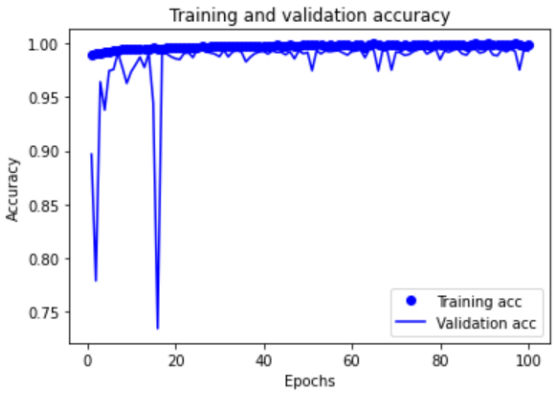

Figure 5, DenseNet121 Fashion-MNIST Training & Test Accuracy

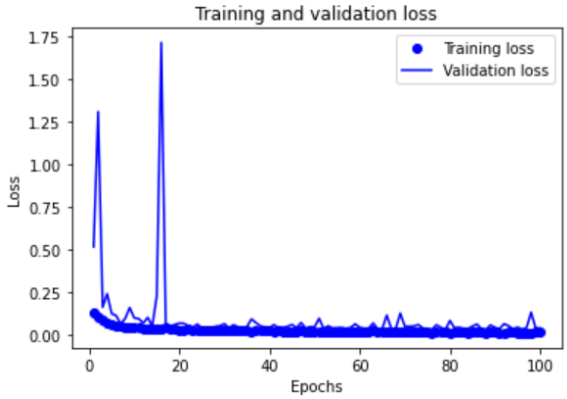

Figure 6, DenseNet121 Fashion-MNIST Training & Validation Average loss

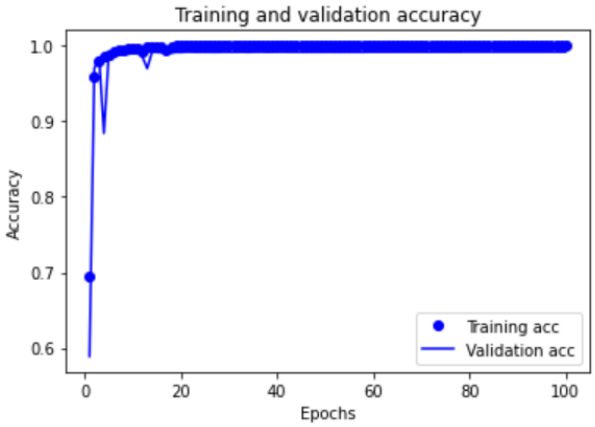

Figure 7, Example VGG19 MNIST Training & Validation Accuracy

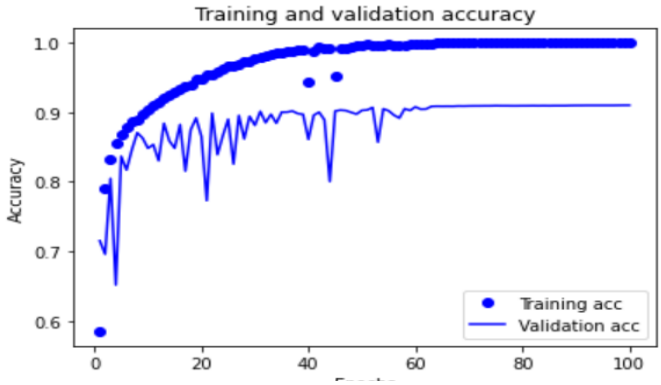

Figure 8, Example VGG19 Fashion-MNIST Training & Validation Accuracy

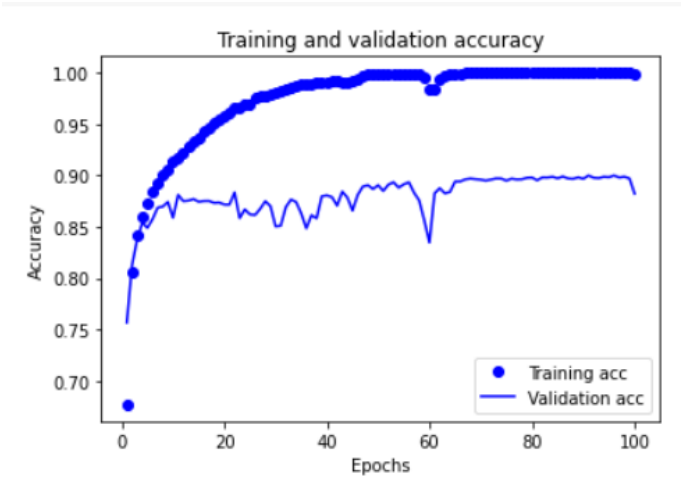

Figure 9, Example ResNet110 Fashion-MNIST Training & Validation Accuracy

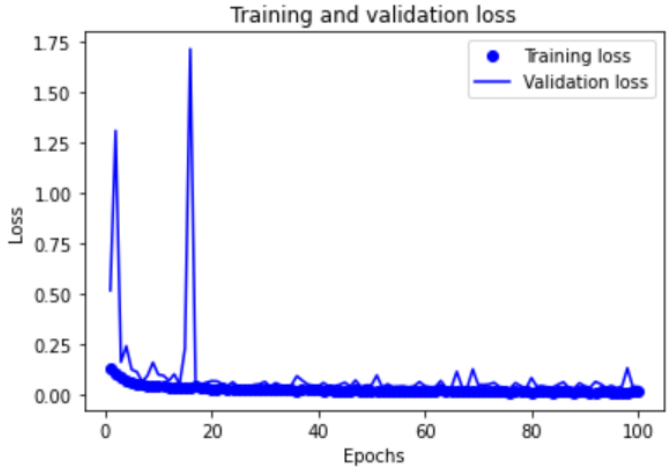

Figure 10, Example DenseNet121 Fashion MNIST Training & Validation Loss

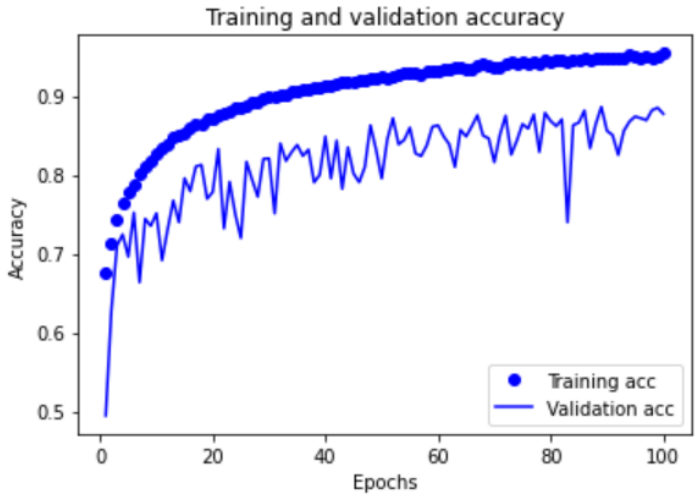

Figure 11, Example DenseNet121 CIFAR-10 Training and Validation Accuracy

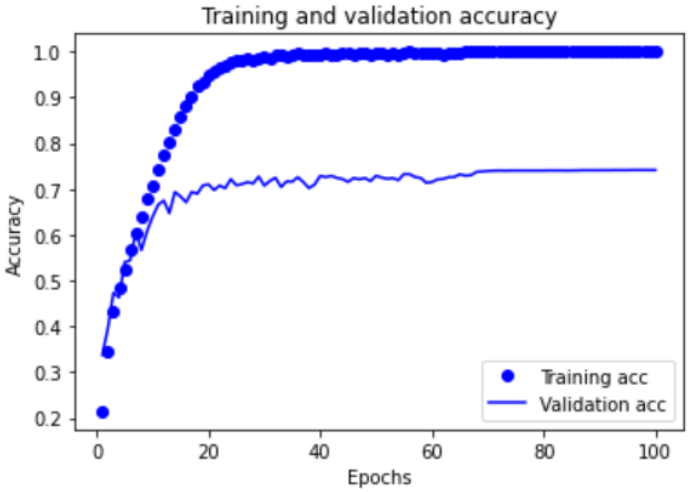

Figure 12, Example VGG19 Training & Validation Accuracy for CIFAR-10

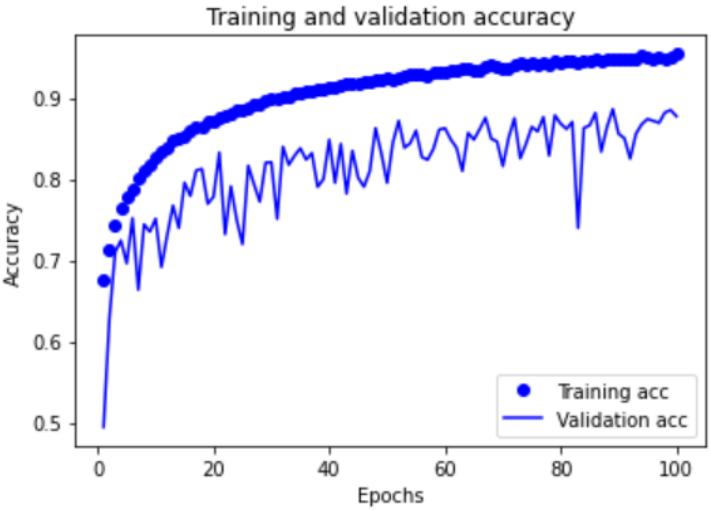

Figure 13, Example DenseNet Training & Validation Accuracy for CIFAR-10

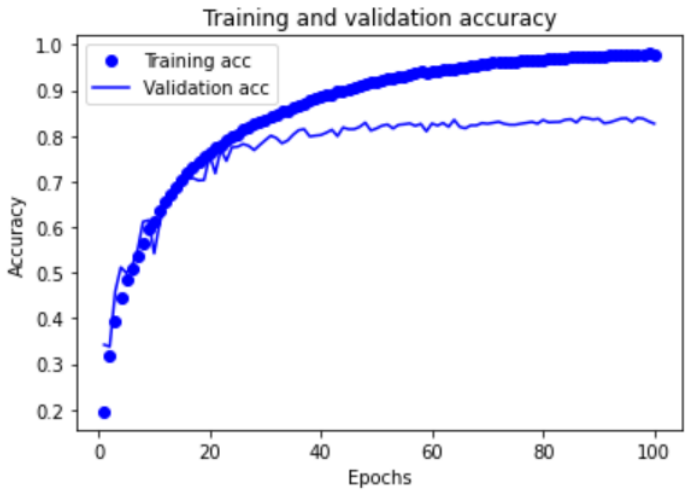

Figure 14, Example VGG19 Training & Validation Accuracy for CIFAR-10+

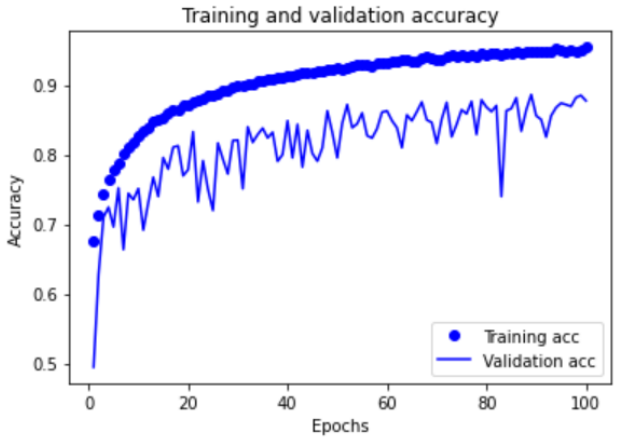

Figure 15, Example Resnet110 Training & Validation Accuracy for CIFAR-10+

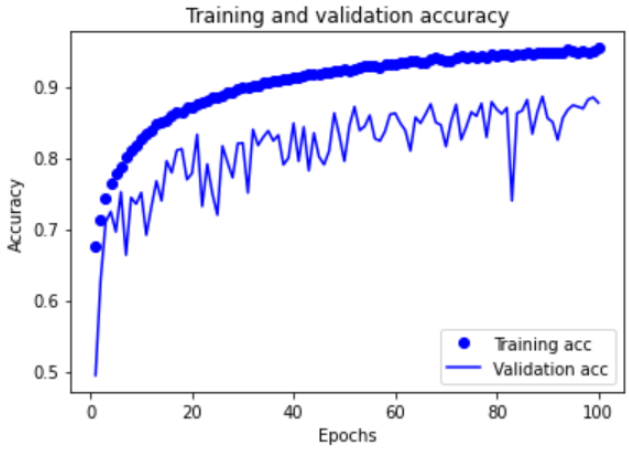

Figure 16, Example DenseNet121 Training & Validation Accuracy for CIFAR-10+