**Assignment-2**

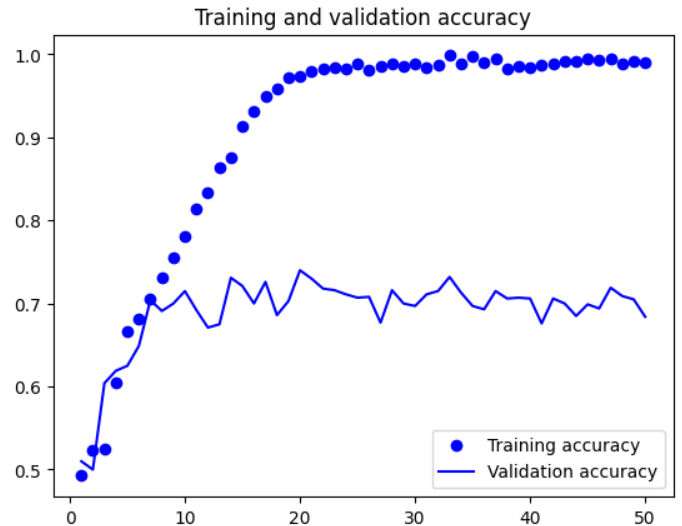
**Summary Report**

**Questions:**

1. Consider the Cats & Dogs example. Start initially with a training sample of 1000, a validation sample of 500, and a test sample of 500 (half the sample size as the sample Jupyter notebook on Canvas). Use any technique to reduce overfitting and improve performance in developing a network that you train from scratch. What performance did you achieve?

The CNN model's training accuracy was 99.32% across 50 epochs. At 70%, the validation accuracy varied, indicating some overfitting. Applying the model to unseen data would be challenging, as evidenced by its test accuracy of 69.9%, which was lower than its training accuracy. The model's architecture consists of convolutional layers sans dropout layers and max-pooling. Various data augmentation methods, such as rotation, zooming, and horizontal flipping, were employed to enhance generalization. Additional improvements, such the use of regularization techniques and a greater variety of data, could enhance the model's performance.

* Training accuracy started at 49.67% and increased to 99.32%.
* Validation accuracy started at 51.0% and reached 70.50%.
* Test accuracy, measured after training, was 69.9%.



A graph of a line graph

Description automatically generated with medium confidence

1. Increase your training sample size. You may pick any amount. Keep the validation and test samples the same as above. Optimize your network (again training from scratch). What performance did you achieve?

The training process made use of data-augmentation techniques like rotation, zooming, and random flipping. The model architecture consisted of a dropout layer to prevent overfitting and many convolutional and max pooling layers. The training accuracy began at 51.68% and rose to 96.11%, whereas the validation accuracy began at 59.60% and reached 84.20%. The test accuracy was assessed at 81.9% following instruction. The early stopping callback was used to halt training when the model's performance on the validation set did not improve. Overall, the excellent accuracy of the CNN's image identification on both the test and validation sets showed how well it performed.

* Training accuracy: Started at 51.68% and increased to 96.11%.
* Validation accuracy: Started at 59.60% and reached 84.20%.
* Test accuracy: 81.9%.

A graph of a graph

Description automatically generated with medium confidence

A graph of a training and validation loss

Description automatically generated

1. Now change your training sample so that you achieve better performance than those from Steps 1 and 2. This sample size may be larger, or smaller than those in the previous steps. The objective is to find the ideal training sample size to get best prediction results.

Data augmentation techniques were employed to enhance the model's performance after it was trained using a convolutional neural network (CNN) architecture. The validation dataset was used to confirm the model's performance during training, whereas the training dataset was used to train the model. The model had a maximum training accuracy of about 95% and a maximum validation accuracy of about 91.40%. The test accuracy of the trained model, when examined on an alternative test dataset, was around 89.3%. In order to avoid overfitting, the model employed early stopping with a patience of 10 and saved the best model based on validation loss. Both the training and test datasets showed high accuracy, indicating that the CNN model did well overall in the image categorization task.

* The training accuracy reached approximately 95%.
* The validation accuracy reached approximately 91.40%.
* The test accuracy reached approximately 89.3%.

A graph of a graph with blue dots

Description automatically generated

A graph of training and validation

Description automatically generated

1. Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.

* **Pretrained Model 1: VGG16 Pretrained Convnet Network**

Using the VGG16 convolutional base and transfer learning, the model was trained. The convolutional basis was adjusted for the new dataset following pretraining on the ImageNet dataset. The model architecture includes a classifier and a data augmentation stage. Early halting was done in order to prevent overfitting. A batch size of 32 was used, and the training procedure lasted for 11 epochs. The test accuracy was roughly 91.50%, the validation accuracy was roughly 91.50%, and the training accuracy was roughly 95.41%. The model's great generalization to unknown data is demonstrated by its ability to balance training and validation performance. This demonstrates the efficacy of transfer learning for picture categorization problems.

* Training accuracy: 96.95%
* Validation accuracy: 90.60%
* Test accuracy: 90.6%

A graph with blue lines

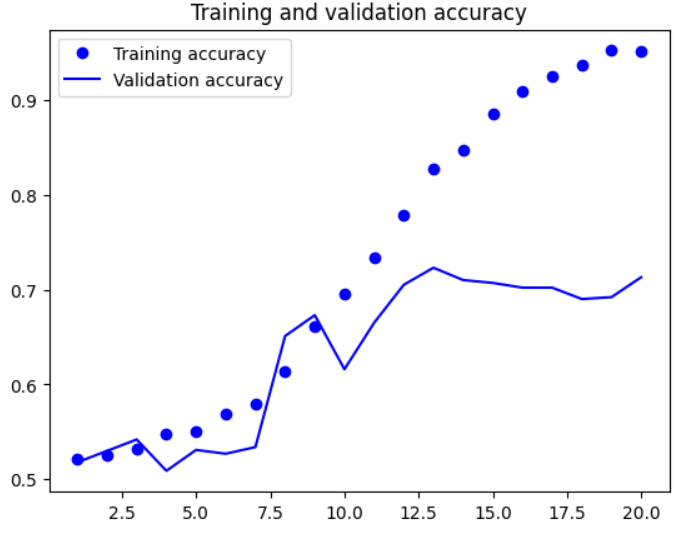
Description automatically generatedA graph with blue lines and dots

Description automatically generated

* **Pretrained Model 2: ResNet50V2 Convolutional Base**

This little piece of code classified images of cats and dogs using a convolutional neural network (CNN) integrated into TensorFlow's Keras API. Training, validation, and test sets are created from the 5000, 1000, and 1000 picture datasets. Following a number of convolutional layers with max pooling and ReLU activation, the model architecture is composed of fully linked layers. The last layer employs a sigmoid activation function for binary classification. The model is trained using the Adam optimizer and the binary crossentropy loss function. Given that the training accuracy was 95.04% and the validation accuracy was 71.30%, overfitting may have occurred.

* Training accuracy: 95.04%
* Validation accuracy: 71.30%
* Test accuracy: 57.20%



A graph with blue dots

Description automatically generated

* **Pretrained Model 3: MobileNetV2 Convolutional Base**

The final layers of the model are based on the MobileNetV2 convolutional base and are optimized for binary classification. Zooms, rotations, and random flips were used to enhance the dataset. The model was trained for 50 epochs with early stopping to prevent overfitting. The training accuracy rose across the remaining epochs, from 56.95% to 96.00% at the conclusion. Additionally, the validation accuracy increased from 77.10% to 98.50%. The model performed exceptionally well on the test set, with an accuracy of 98.60%. Transfer learning and data augmentation have been used to achieve high classification accuracy.

* Training accuracy: 94.02%.
* Validation accuracy: 96.50%.
* Test accuracy: 97%.

A graph of a training and validation accuracy

Description automatically generated

A graph of training and validation

Description automatically generated

**Accuracy Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Type** | **Training Accuracy** | **Validation Accuracy** | **Test Accuracy** |
| Initial CNN Model | 99.17% | 68.40% | 69.9% |
| CNN Model with Increased Training | 96.11% | 84.20% | 81.9% |
| Optimized CNN Model | 95% | 91.40% | 89.3% |
| Pretrained Model 1 | 96.95% | 90.60% | 90.6% |
| Pretrained Model 2 | 95.04% | 71.30% | 57.2% |
| Pretrained Model 3 | 94.02% | 96.50% | 97.0% |

**Conclusion:**

To sum up, this assignment investigated the effects of network design selection and training sample size on a two-class picture classification job. Overfitting was noted even though training CNN models from scratch yielded extremely high training accuracy. Network optimization and a larger training sample size improved test and validation accuracy. The efficiency of transfer learning in image classification was demonstrated by the noticeably better accuracies obtained when pretrained models (ResNet50V2, MobileNetV2, and VGG16) were used.