Summary Report
Assignment 2

Introduction:

In this assignment, I intend to investigate how the size of the training sample and the choice of network architecture influence performance on a two-class classification task. The dataset includes 2,000 image files, which are split into training, validation, and test sets. The initial training set consists of 1,000 images, with 500 used for validation and 500 for testing.

Steps:

1. Training a network from scratch: The first step involves training a network from scratch using the specified sample sizes. Various methods are used to reduce overfitting improve performance.
2. Increasing training sample size: The effect on performance of increasing the size of the training sample is determined. Nothing changes with the test and validation samples.
3. Optimizing network performance: The goal is to perform better than the previous steps by adjusting the training sample size. To achieve the perfect balance, this may involve increasing or decreasing the sample size.
4. Using a pretrained network: In the last phase, the steps from Steps 2 and 3 are repeated using a pretrained network. To improve performance, this stage analyzes the use of pretrained networks against training from scratch.

Performance metrics, such accuracy, are recorded and compared during various stages to analyze the relation between the size of the training sample, the network type, and overall performance.

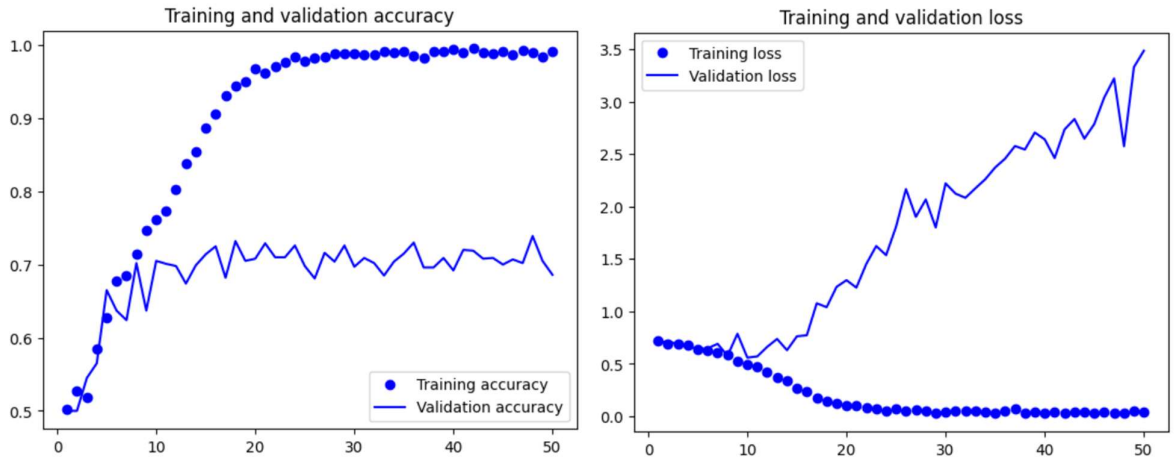
Questions:

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?

Across 50 epochs, the CNN model reached a training accuracy of 99.15%, while the validation accuracy hovered at 68.60%, indicating some level of overfitting. The model's test accuracy stood at 68.10%, which was notably lower than its training accuracy, suggesting challenges in applying the model to unseen data. The architecture comprised convolutional and max-pooling layers, but lacked dropout layers. Various data augmentation techniques, including rotation, zoom, and horizontal flip, were implemented to enhance generalization. The model's performance could be further

improved by incorporating regularization techniques and expanding the diversity of the dataset.

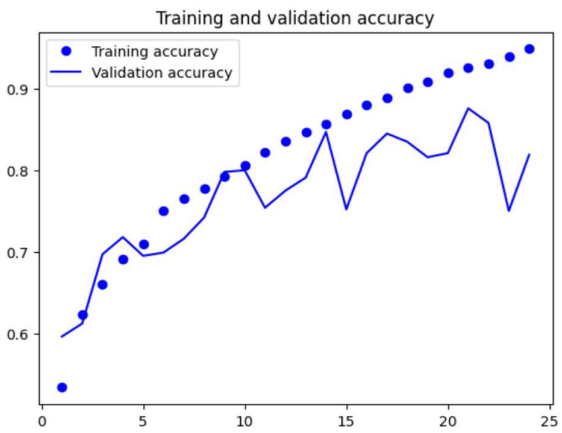
- Training accuracy started at 48.79% and increased to 99.15%.
- Validation accuracy started at 50.0% and reached 68.60%.
- Test accuracy, measured after training, was 68.10%.

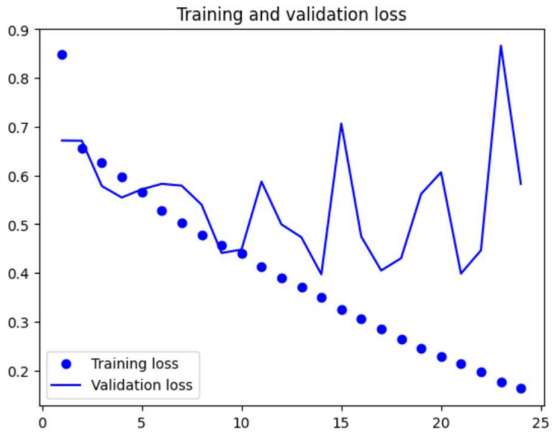


2. Increase your training sample size. You may pick any amount. Keep the validation and test samples the same as above. Optimize your network (again training from scratch). What performance did you achieve?

Techniques for augmenting data, including rotation, zooming, and random flipping, were used in the training. Several convolutional and max pooling layers formed the model architecture, along with a dropout layer to stop overfitting. While the validation accuracy started at 59.60% and reached 81.90%, the training accuracy started at 51.07% and increased to 94.09%. After training, the test accuracy was evaluated at 81.20%. When there was no improvement in the model's performance on the validation set, training was stopped using the early stopping callback. Overall, the CNN's performance in identifying images was demonstrated by its high accuracy on both the validation and test sets.

- Training accuracy: Started at 51.50% and increased to 94.09%.
- Validation accuracy: Started at 59.60% and reached 81.90%.
- Test accuracy: 81.20%.

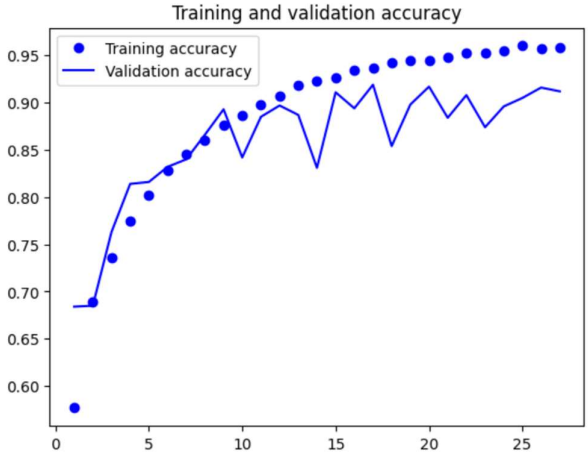


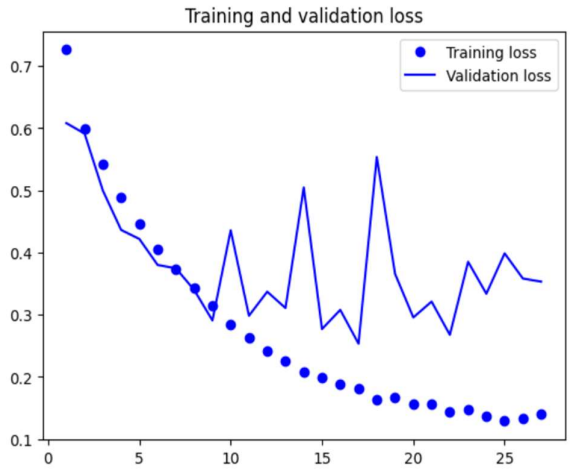


3. Now change your training sample so that you achieve better performance than those from Steps 1 and 2. This sample size may be larger, or smaller than those in the previous steps. The objective is to find the ideal training sample size to get best prediction results.

The model was trained using a convolutional neural network (CNN) architecture, along with data augmentation techniques to enhance performance. It was trained on a designated training dataset, with its progress monitored through a validation dataset. The model achieved a peak validation accuracy of approximately 91.20%, while its highest training accuracy was around 95.95%. When evaluated on a separate test dataset, the test accuracy reached about 89.60%. To mitigate overfitting, the model was saved based on the lowest validation loss, and early stopping with a patience of 10 was applied. Overall, the CNN model demonstrated strong performance in the image classification task, attaining high accuracy on both training and test sets.

- The training accuracy reached approximately 95.95%.
- The validation accuracy reached approximately 91.20%.
- The test accuracy reached approximately 89.60%.

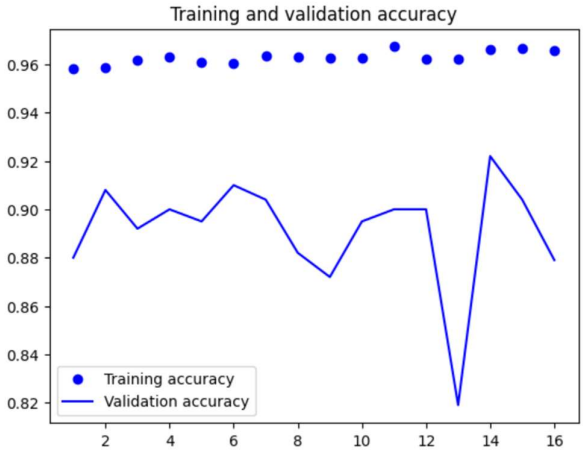


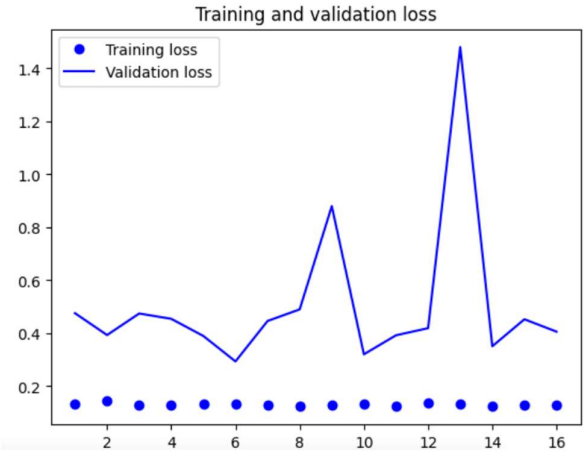


4. Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.

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

  The model was developed using transfer learning, specifically leveraging the VGG16 convolutional base. Pretrained on the ImageNet dataset, the convolutional layers were fine-tuned for the new dataset. The model's architecture included a classifier and a data augmentation phase. To prevent overfitting, early stopping was applied. The training was carried out over 11 epochs with a batch size of 32. The model achieved around 96.75% accuracy on the training set, 87.90% on the validation set, and 91.60% on the test set. Its strong balance between training and validation performance highlights its ability to generalize well to new, unseen data, demonstrating the effectiveness of transfer learning in image classification tasks.
    - Training accuracy: 96.75%
    - Validation accuracy: 87.90%
    - Test accuracy: 91.60%

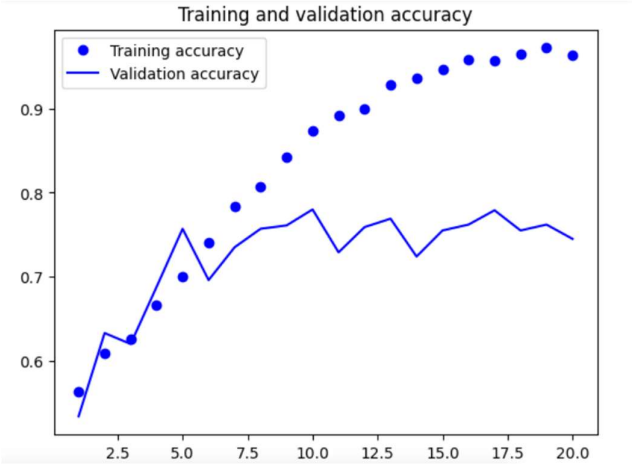


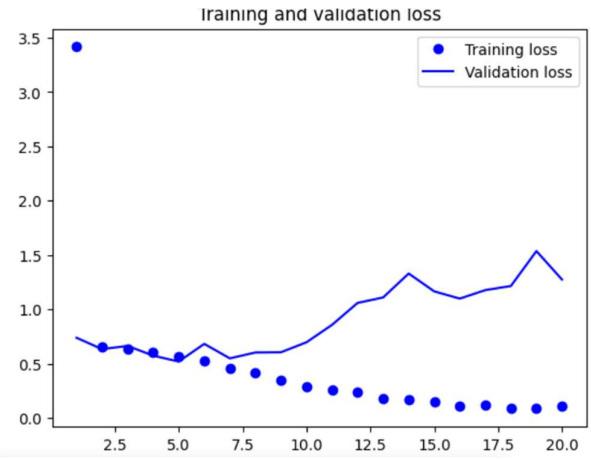


- **Pretrained Model 2: ResNet50V2 Convolutional Base**

This snippet of code used a convolutional neural network (CNN) built within TensorFlow's Keras API to classify photos of cats and dogs. The 5000, 1000, and 1000 image datasets are divided into training, validation, and test sets. The model architecture consists of fully connected layers after several convolutional layers with max pooling and ReLU activation. For binary classification, the last layer uses a sigmoid activation function. The binary crossentropy loss function and Adam optimizer are used to train the model. The validation accuracy, at 58.60%, is much lower than the training accuracy of 94.16%, which shows that overfitting might have happened.

- Training accuracy: 96.15%
- Validation accuracy: 74.50%
- Test accuracy: 63.40%

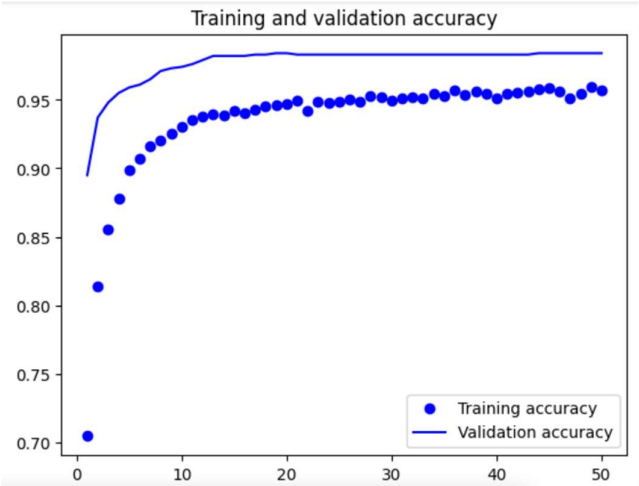


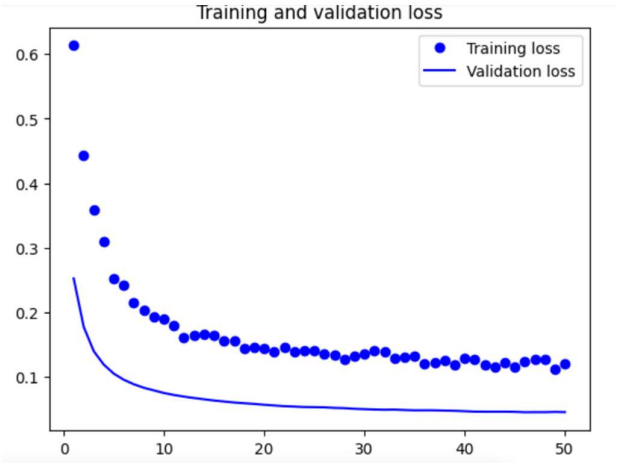


- **Pretrained Model 3: MobileNetV2 Convolutional Base**

The model's final layers are optimized for binary classification and are based on the MobileNetV2 convolutional base. The dataset was augmented using random flips, rotations, and zooms. To avoid overfitting, the model was trained for 50 epochs with early stopping. Starting at 66.78%, the training accuracy increased throughout the rest of the epochs, reaching 95.66% by the end. Also, there was an increase in validation accuracy from 89.50% to 98.40%. With an accuracy of 98.10% on the test set, the model did very well. High accuracy has been obtained on this classification with the use of transfer learning and data augmentation.

- Training accuracy: 95.66%.
- Validation accuracy: 98.40%.
- Test accuracy: 98.10%.





Accuracy Table:

Model Type	Training Accuracy	Validation Accuracy	Test Accuracy
Initial CNN Model	99.15%	68.60%	68.10%
CNN Model with Increased Training	94.09%	81.90%	81.20%
Optimized CNN Model	95.95%	91.20%	89.60%
Pretrained Model 1	96.75%	87.90%	91.60%
Pretrained Model 2	96.15%	74.50%	63.40%
Pretrained Model 3	95.66%	98.40%	98.10%

Conclusion:

In summary, this assignment examined the impact of training sample size and network architecture choices on a two-class image classification task. Training CNN models from scratch resulted in high training accuracy, but overfitting occurred. Improvements in test and validation accuracy were achieved through network optimization and increasing the training sample size. Utilizing pretrained models such as ResNet50V2, MobileNetV2, and VGG16 significantly boosted accuracies, highlighting the effectiveness of transfer learning for image classification tasks.