Advanced Machine Learning

Assignment – 2

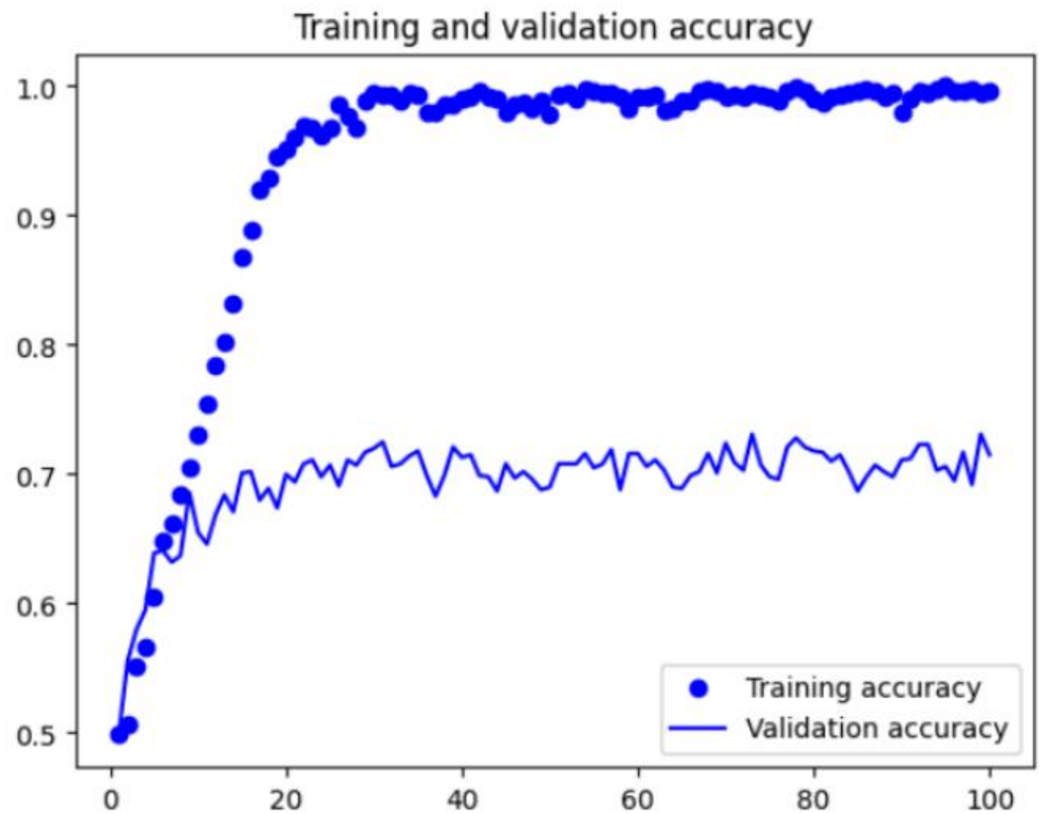
Convolution

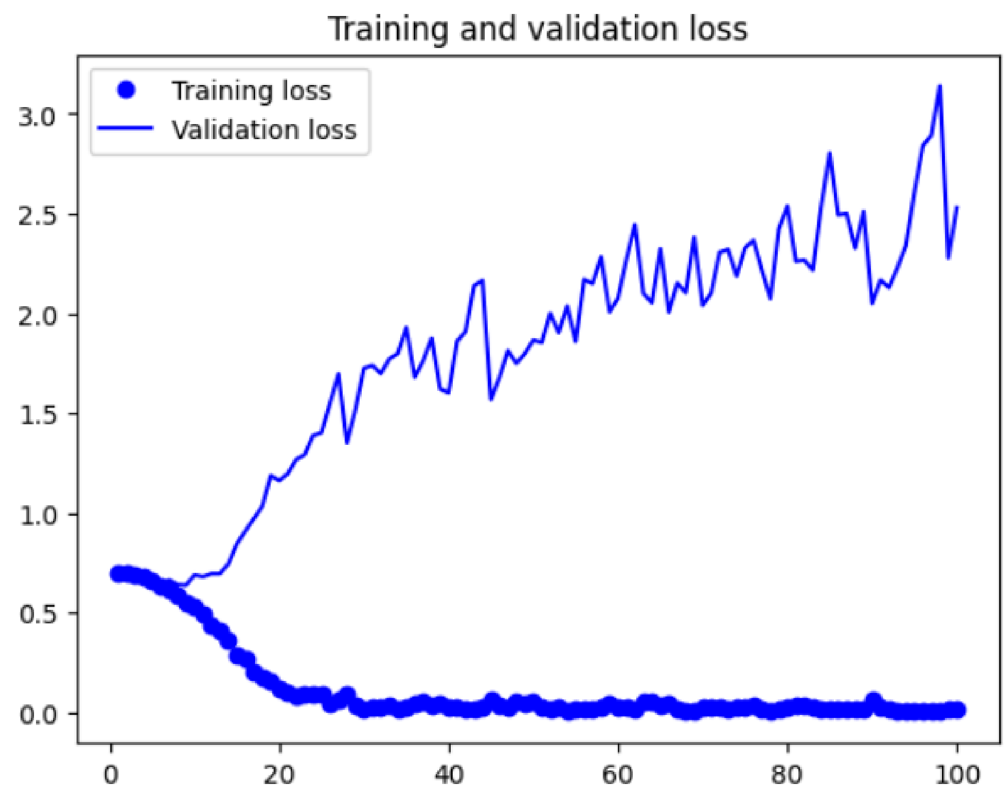
Convolutional Neural Networks: This document discusses creating a convolutional neural network (CNN), a specialized computer program designed to identify whether an image contains a cat or a dog. The images used to train this program are sourced from Kaggle. Although there are numerous images available, we utilize only 2000 of them for training purposes.

Question1: Consider the Cats & Dogs example. Start initially with a training sample of 1000, a validation sample of 500, and a test sample of 500 (like in the text). Use any technique to reduce overfitting and improve performance in developing a network that you train from scratch. What performance did you achieve?

Answer:

We used 1000 images to train the computer program and then validated it with an additional 500 images. After that, we tested the program with another set of 500 images to evaluate its performance. To prevent the program from overfitting to the training images, we employed a technique called dropout. Before training, we converted the image files into a format that the computer could process, adjusted the colors, and resized the images. During testing, the program achieved an accuracy of 99.5%, and during the training validation, it was approximately 71.5% accurate.

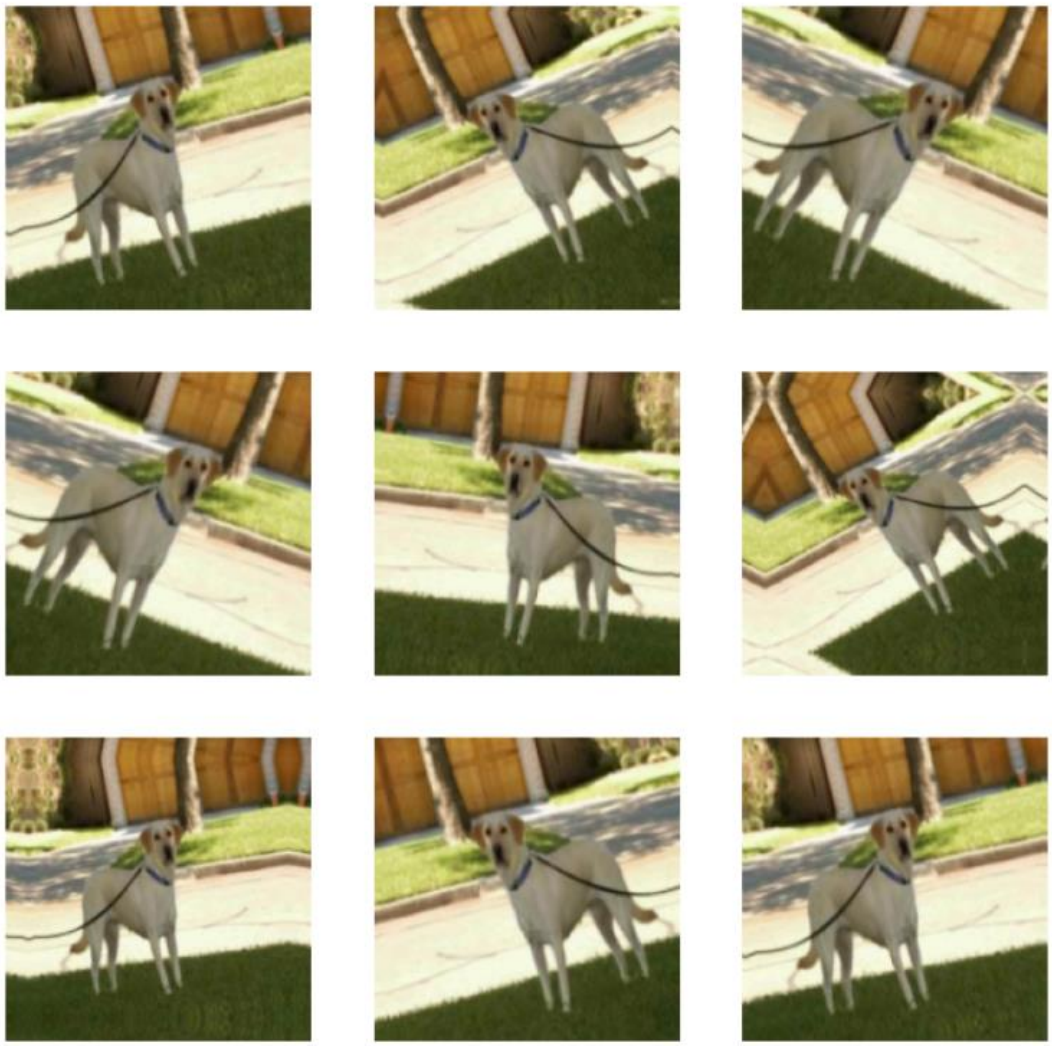

Training and validation accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100
Training accuracy
Validation accuracy

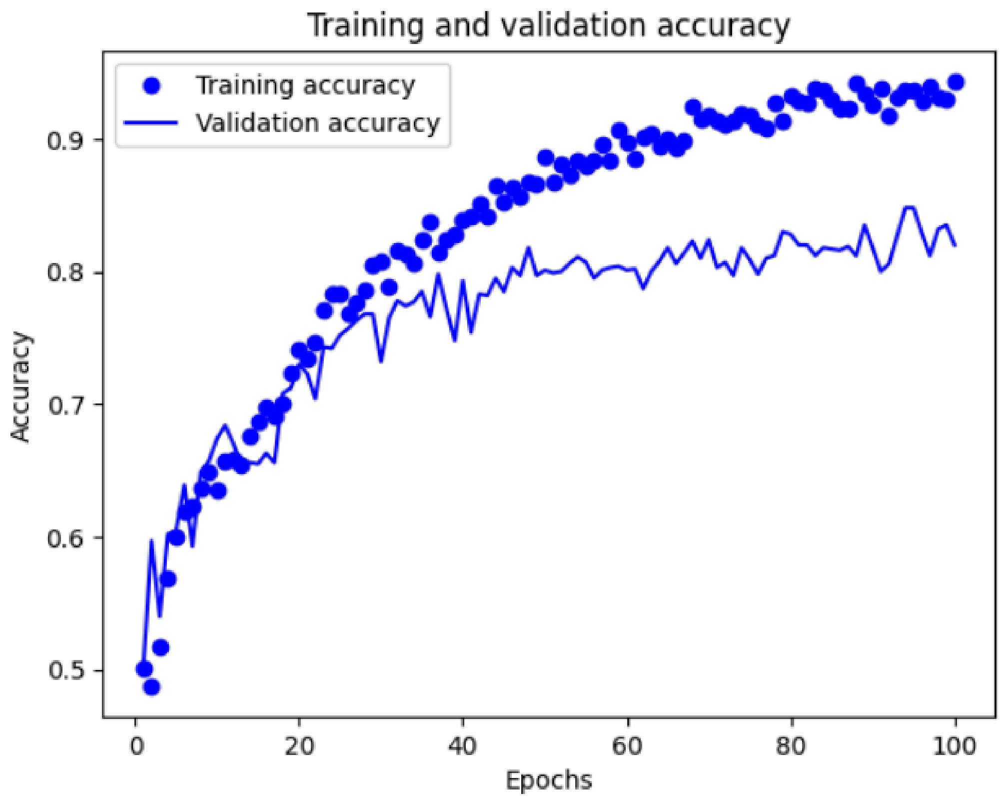


<u>Question2:</u> Increase your training sample size. You may pick any amount. Keep the validation and test samples the same as above. Optimize your network (again training from scratch). What performance did you achieve?

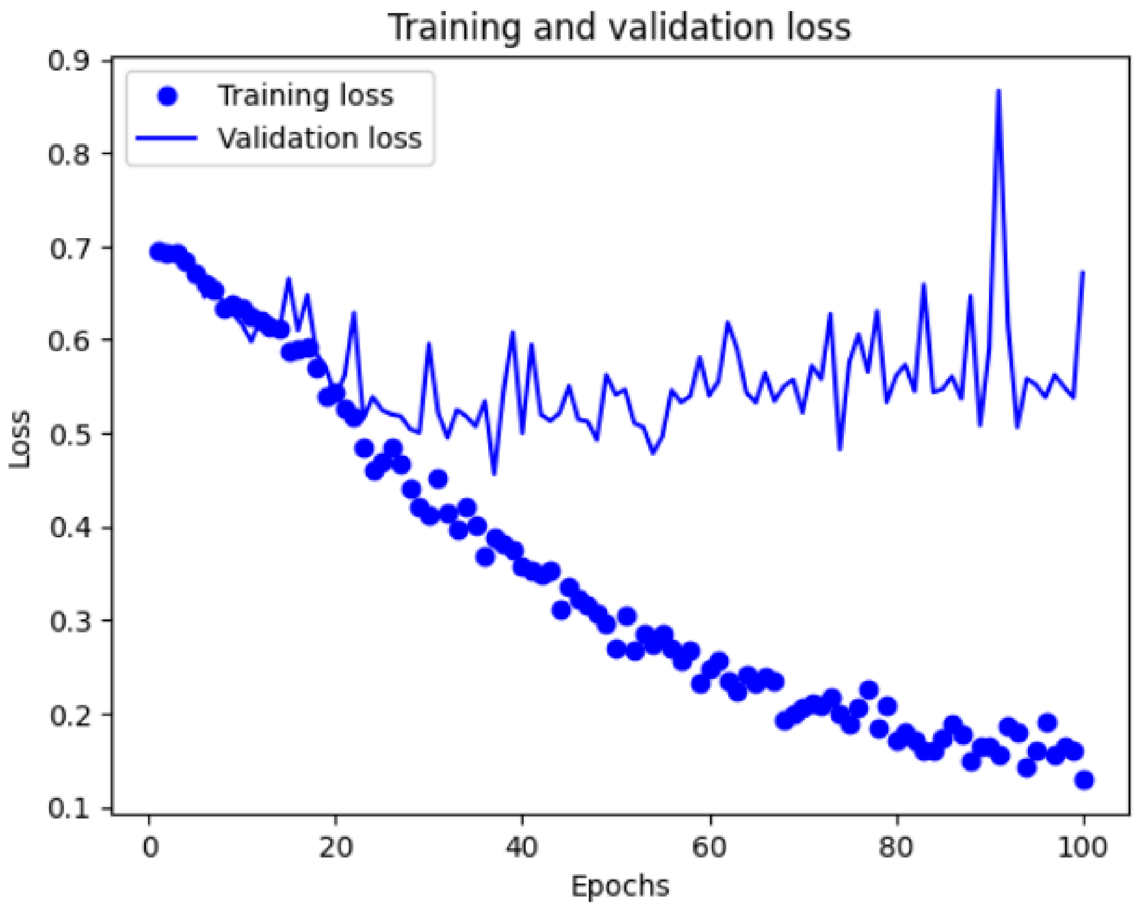
<u>Answer:</u>

We increased the number of training images to 1500, while still using 500 images for validation during training and another 500 for testing. To enhance the program's performance, we applied techniques like flipping, rotating, and zooming in on the training images. These augmentations helped the program learn more effectively. As a result, the program improved its accuracy, achieving 94.35% correctness during training and 82% accuracy during validation.





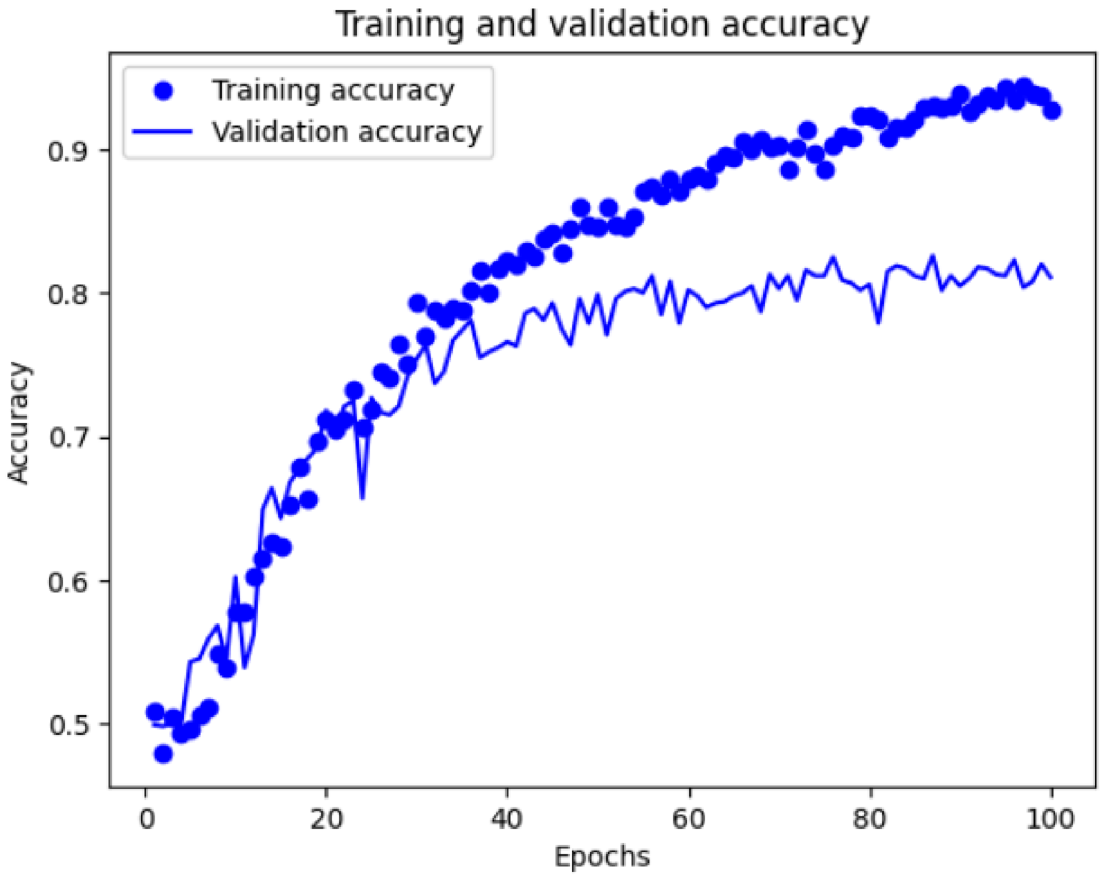
Training and validation accuracy
Training accuracy
Validation accuracy
Accuracy
0.5
0.6
0.7
0.8
0.9
0
20
40
60
80
100
Epochs

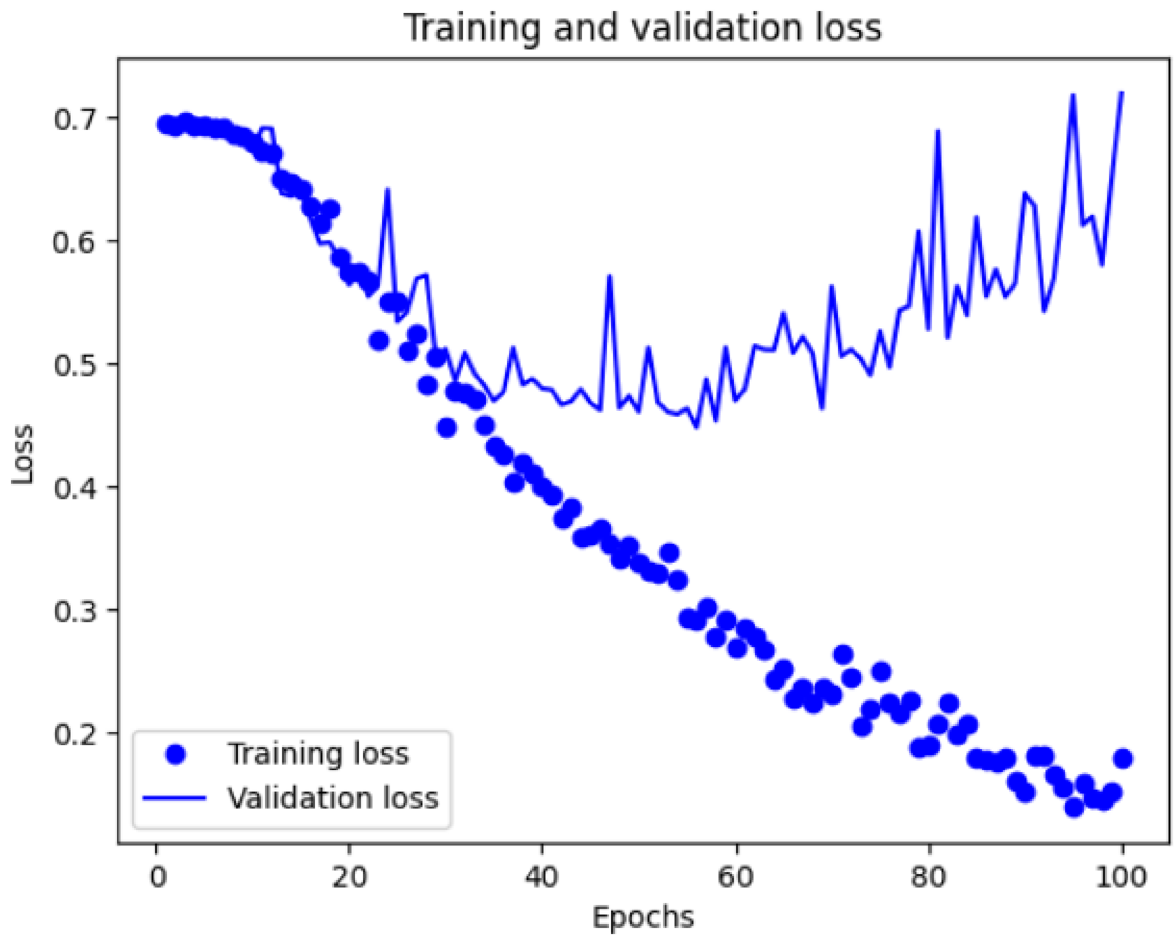


<u>Question3:</u> Now change your training sample so that you achieve better performance than those from Steps 1 and 2. This sample size may be larger, or smaller than those in the previous steps. The objective is to find the ideal training sample size to get the best prediction results.

<u>Answer:</u>

We increased the number of training images to 2000 to further improve the program's performance. We continued using techniques such as flipping, rotating, and zooming on these images during training. With the larger dataset and these enhancements, the program became better at interpreting images. As a result, its accuracy during training reached approximately 94.45%, and during validation, it achieved about 82.6% accuracy.


Training and validation accuracy
Training accuracy
Validation accuracy
Accuracy
Epochs
0.5
0.6
0.7
0.8
0.9
0
20
40
60
80
100



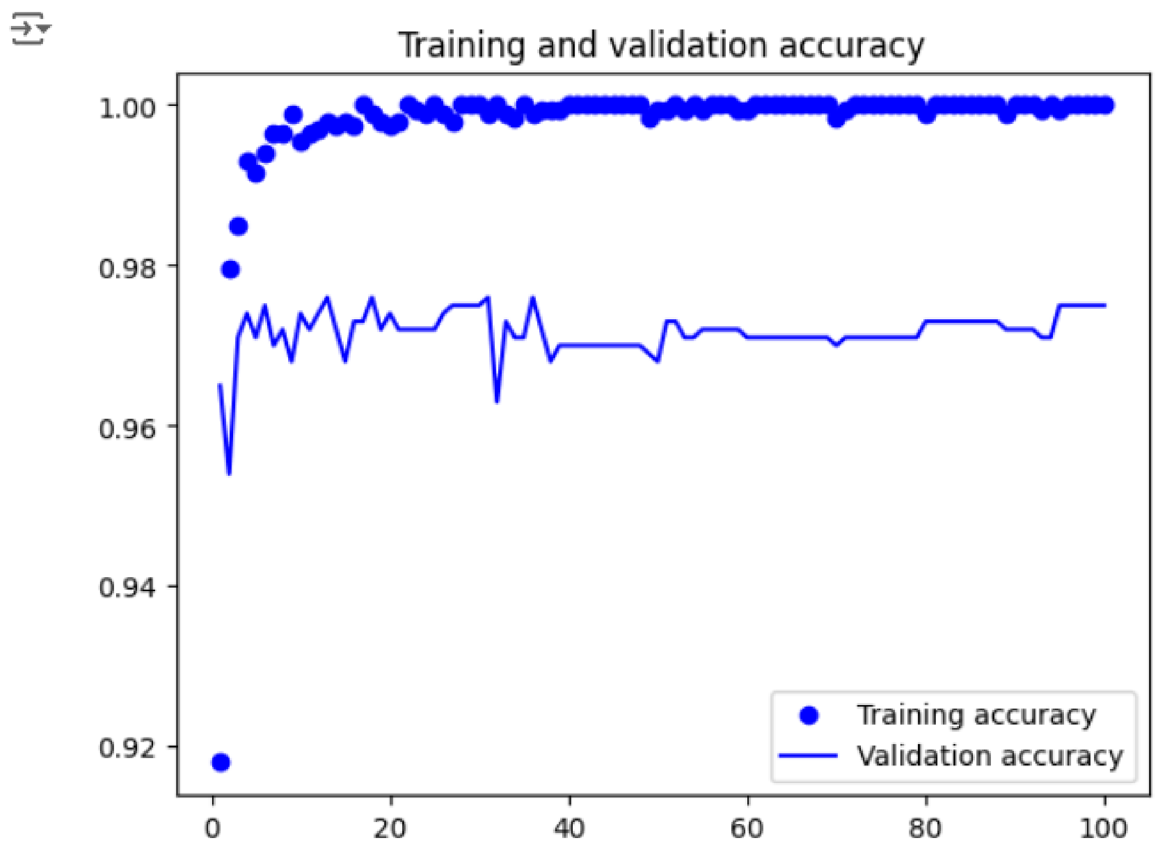
<u>Question4:</u> Repeat Steps 1-3, but now using a pretrained network.

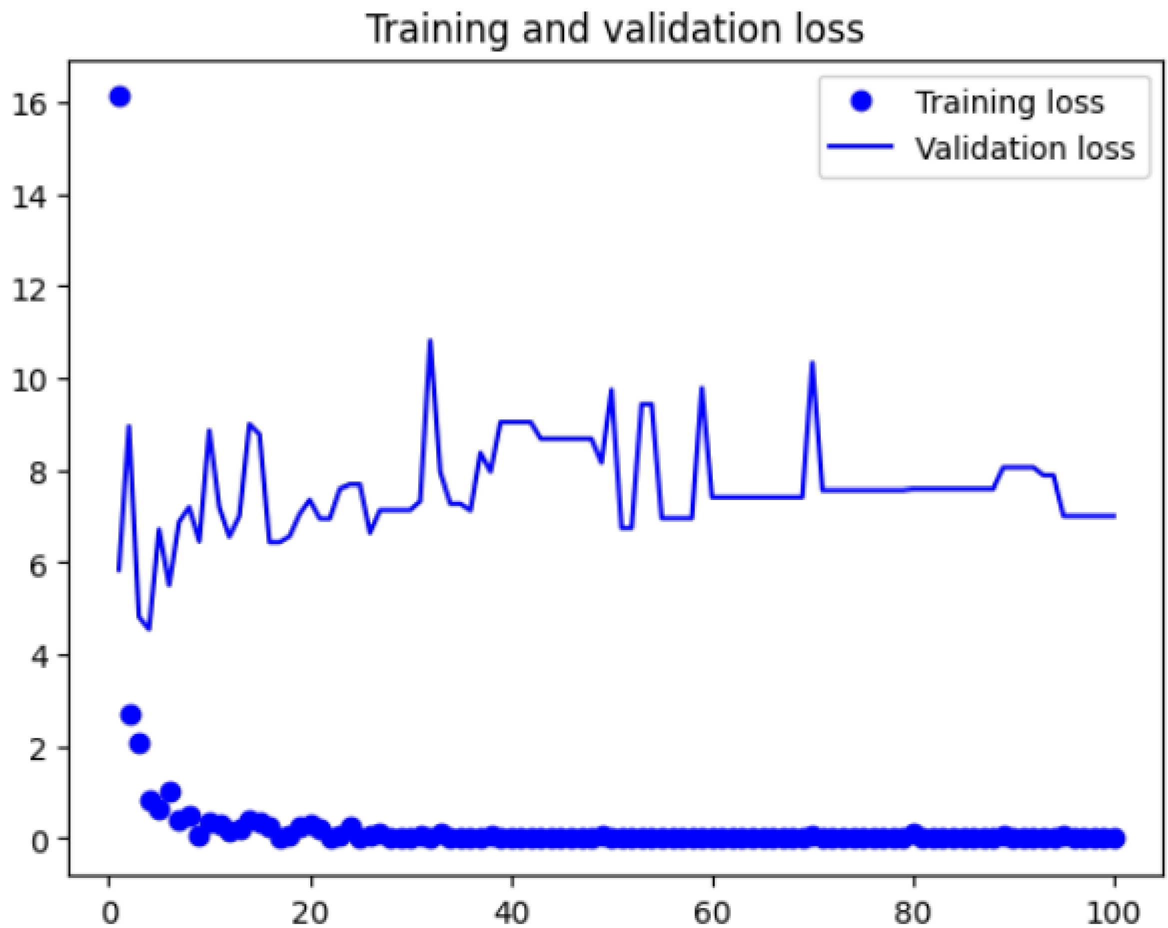
The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use all optimization techniques to get the best performance.

<u>Answer:</u>

<u>Pre-Trained Without Augmentation:</u>

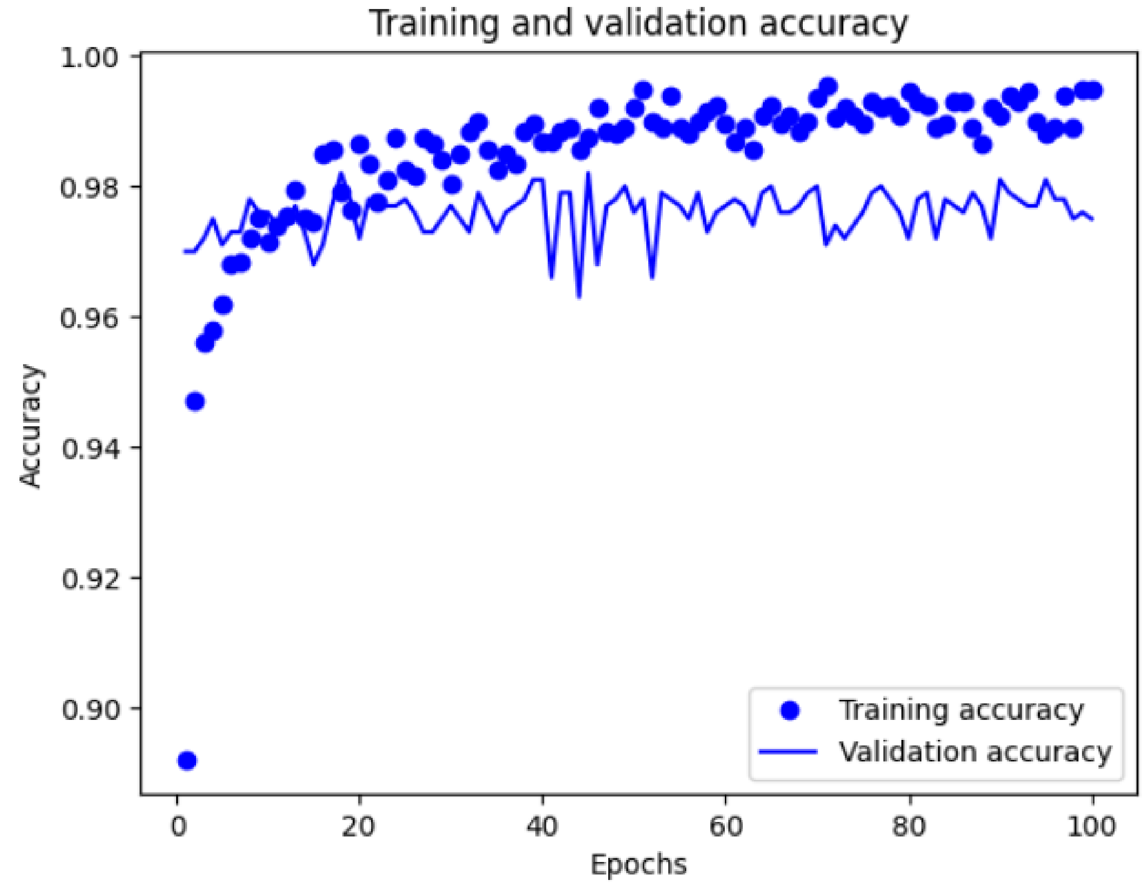
We tested a pre-trained model for this task, which had already been trained on a large dataset of images. This time, we did not use augmentation techniques like flipping or rotating. Even without these methods, the pre-trained model performed exceptionally well. It achieved nearly 100% accuracy during training, which might suggest that it is overfitting to the training data and may not perform as well on new data. During validation, the model achieved an accuracy of about 97.3%, indicating that while it excels on the training images, it might not generalize as effectively to unseen images.



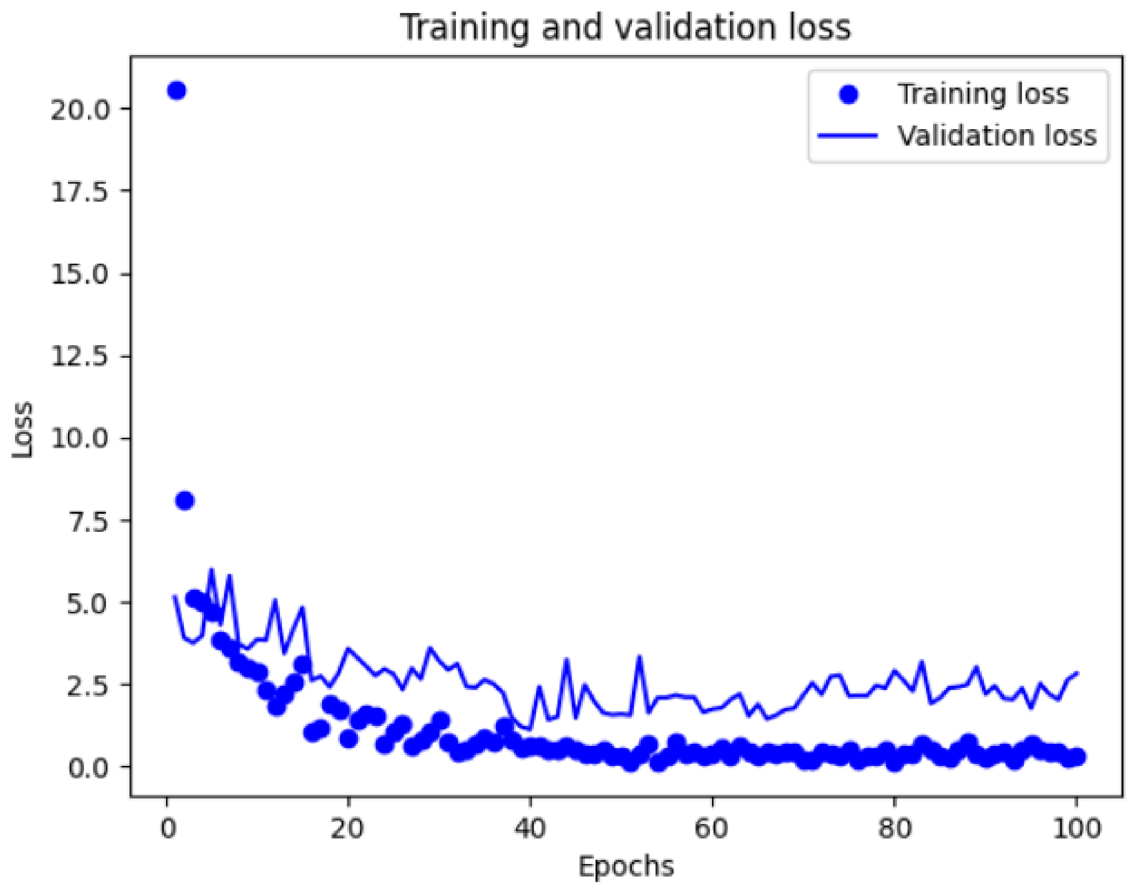


Pre-Trained with Augmentation:

Without applying any data enhancement techniques, the pre-trained model performed well, achieving a validation accuracy of 98.1%. The author then applied a technique called fine-tuning, which involves making slight adjustments to the pre-trained model to better suit the specific task. After fine-tuning and incorporating data augmentation techniques, the model's performance improved further. It reached approximately 99.2% accuracy during training and about 98.1% accuracy during validation.

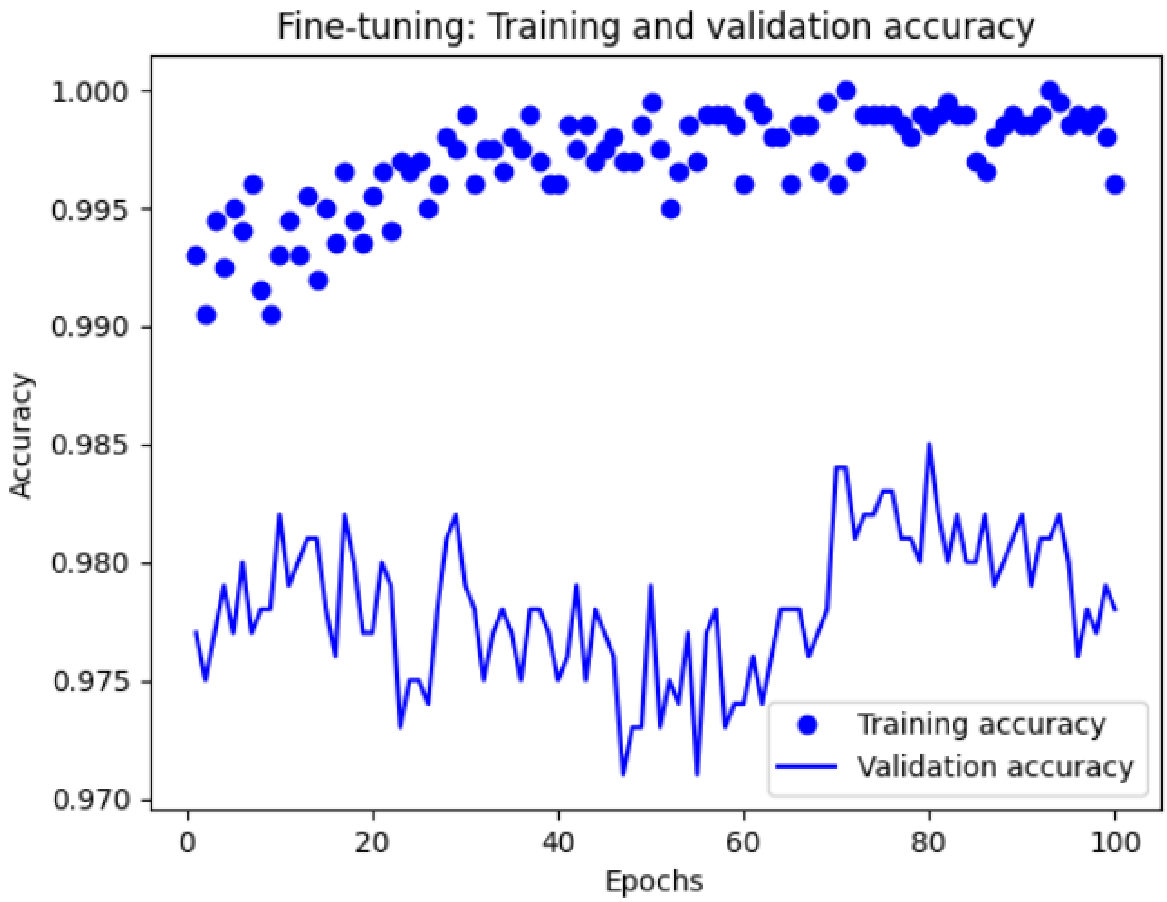


Training and validation accuracy
Accuracy
1.00
0.98
0.96
0.94
0.92
0.90
0
20
40
60
80
100
Epochs
Training accuracy
Validation accuracy

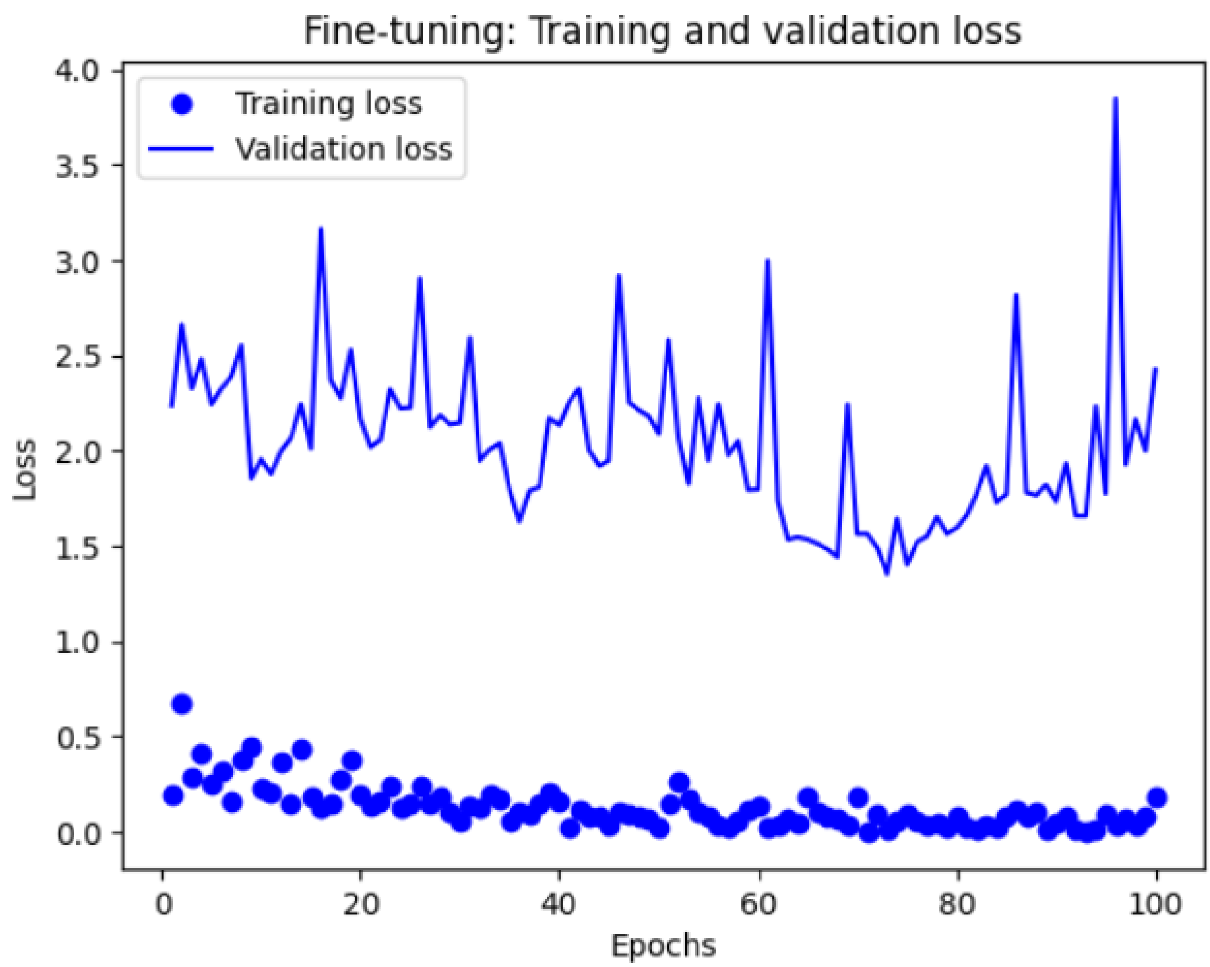


Fine-Tuning With Augmentation:

After evaluating the pre-trained model with and without data enhancement techniques, I advanced to fine-tuning. This involved adjusting the pre-trained model to better fit my specific task by allowing its layers to learn from the newly augmented data, which included techniques like flipping and rotating. The fine-tuning process significantly improved the model's performance. It achieved approximately 99.6% accuracy during training and around 98.2% accuracy during validation. These results highlight the effectiveness of combining a pre-trained model with data augmentation and fine-tuning. The fine-tuned model outperformed the pre-trained model without enhancements, demonstrating the importance of diverse data and customizing the model for the task at hand.



Fine-tuning: Training and validation accuracy
Accuracy
Epochs
1.000
0.995
0.990
0.985
0.980
0.975
0.970
0
20
40
60
80
100
Training accuracy
Validation accuracy



Results:

Sample Size	Data Augmentation	Train Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)
1000	NO	99.5	71.5	66.8
1500	YES	94.35	82	78.4
2000	YES	94.45	82.6	79.5
Pre-Trained	NO	100	97.3	97.5
Pre-Trained	YES	99.2	98.1	97.2
Fine Tuned	YES	99.6	98.2	97.2

Effect on Accuracy:

- **Training Accuracy**: With the increased training sample size of 1500 images, the program achieved 94.35% accuracy during training. This suggests that having more training images helped the model learn the data better, leading to higher accuracy on the training set.
- **Validation Accuracy**: The validation accuracy was 82%. Although this is lower than the training accuracy, it still shows a significant improvement compared to previous metrics. This indicates that the increased training sample size contributed to a better generalization of the model, but there might still be room for improvement or overfitting considerations.
- **Training Sample Size Impact**: However, the observed improvement in training accuracy suggests that increasing the number of training images can positively affect the model's learning ability and performance.
- **Training Accuracy**: This is printed out during each epoch of training in the loop. For example, in the train function
- Here, train_acc is the accuracy on the training set after each epoch.
- **Validation Accuracy**: This is computed at the end of each epoch during validation in the evaluate function:
- **Test Accuracy**: This is computed after training is complete, in the section where you evaluate the model on the test set:

Conclusion:

The model's performance is influenced by the quality of the data it trains on. When we doubled the number of training images from 1000 to 2000, the model's ability to recognize patterns improved, with its accuracy increasing from 76.4% to 80.3% during testing. Additionally, using a pre-trained model combined with data enhancement techniques can yield even better results. Overall, the author believes that increasing the dataset and applying data augmentation techniques can significantly enhance prediction accuracy and improve the model's understanding.