ADVANCED MACHINE LEARNING

Assignment – 3

Rashed Syed

Synopsis

We are using the "Dog-vs-Cats" dataset from Kaggle to build a novel convolutional neural network that is suited for computer vision tasks. Although the small size of the dataset presents a problem, convolutional neural networks are excellent at identifying spatial patterns in images, which makes them perfect for tasks such as segmentation, classification, and object detection.

We think our convnet model can produce good results despite the small dataset. Even with little information, convolutional neural networks are remarkably good at extracting significant features from images and using that knowledge in novel contexts. We first train the model using the available data, and then we use transfer learning techniques to improve its performance even more. Lastly, we'll assess the accuracy of the model with particular metrics.

In conclusion, our objective is to create a convolutional neural network that minimizes the quantity of training data needed while efficiently classifying images from the "Dog-vs-Cats" dataset.

The issue

Finding out if a picture shows a dog or a cat is the aim of the Cats-vs-Dogs dataset binary classification experiment.

Dataset

There are 25,000 images in the Cats-vs-Dogs dataset, with 12,500 images in each category, equally divided between dogs and cats. Three subsets will make up our new dataset: 500 samples per class for the test set, 500 samples per class for the validation set, and 1000 samples per class for the training set. Every sample has been extracted and downloaded.

We are extending the architecture of our neural network to address the increased complexity of our problem. We are adding a second stage to the current Conv2D + MaxPooling2D configuration. By making this change, we can increase the network's capacity and ensure that feature map sizes remain appropriate as we get closer to the Flatten layer.

The feature maps initially start out at 150x150 and gradually get smaller as we move through the network layers, reaching 7x7 just before the Flatten layer. While it might seem a little arbitrary at first, the initial input size works well for our task.

Preprocessing:

To create RGB pixel grids, retrieve the image files and extract the JPEG content. Transform these grids into tensors with floating points. Scale the pixel values appropriately to make sure that their initial range of 0 to 255 falls inside the [0, 1] range, which is the preferred input format for neural networks.

Data Augmentation:

Using data augmentation methods will help us increase the model's accuracy. Through the use of random variations, data augmentation creates new data from preexisting training samples, allowing us to obtain trustworthy results even with small datasets. During training, this method exposes the model to various versions of the images, which improves the model's ability to generalize. We plan to randomly apply different transformations, like flipping, rotating, and zooming, to the training set images in order to achieve this goal. This procedure produces variations on the original images, which increases the variability of the dataset and reinforces the resilience of our model.

Pre-trained model:

This dataset contains many animal classifications, including different dog and cat breeds. VGG16 is an example of a convolutional neural network architecture that is appropriate for ImageNet tasks.

Using a pre-trained network is advantageous when working with a large and diverse original dataset because it is a flexible model whose attributes can be tailored to different computer vision tasks. One of deep learning's main advantages over other machine learning techniques is its capacity to transfer learned features across different tasks. Examining a large-scale pre-trained convolutional neural network with the ImageNet dataset serves as an excellent illustration of this idea. which includes 1.4 million annotated images and 1,000 unique classes.

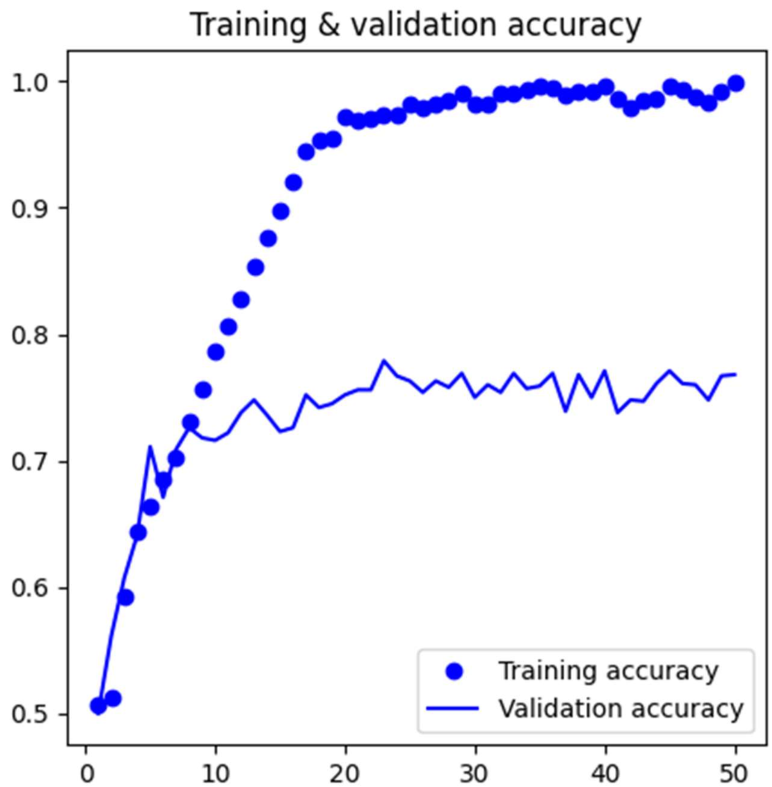
The two main methods for using a pre-trained network are feature extraction and fine-tuning. In this instance, we'll concentrate on feature extraction to improve the outcomes. First, features will be extracted without any augmented data, and then the augmented data will be added.

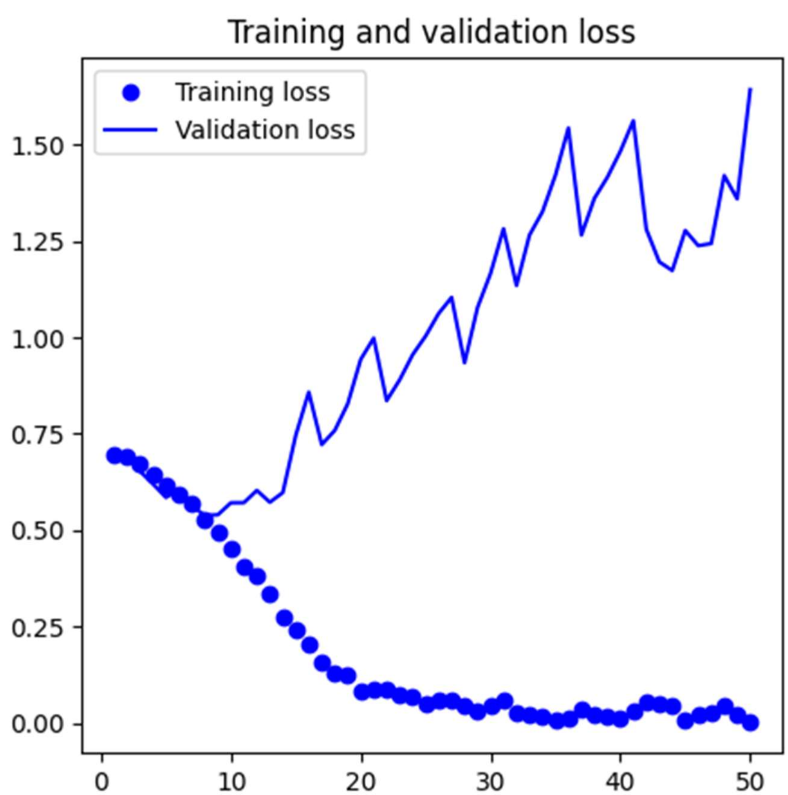
Q1: Consider the Cats & Dogs example. Start initially with a training sample of 1000, a validation sample of 500, and a test sample of 500 (half the sample size as the sample Jupiter notebook on Canvas). Use any technique to reduce overfitting and improve performance in developing a network that you train from scratch. What performance did you achieve?

We used a training sample of 1000 instances for the Cats & Dogs Dataset, along with 500 test sets and validation sets. Given the possibility of overfitting with this training sample size, I took measures to address it by implementing a 50% dropout strategy.

Hypertuning parameters:

We set the batch size to 255 and used the flattening technique to transform the data. We discovered via this procedure that the validation accuracy was 76.8%, whereas the test accuracy reached 68.8.%


Training & validation accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0
10
20
30
40
50
Training accuracy
Validation accuracy


Training and validation loss
Training loss
Validation loss
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40
50

Q2: 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 results show that the test accuracy was 76.6% and the validation accuracy was 76.5%.

The findings show a significant improvement over the earlier results (Question 1). The model's performance has improved dramatically with the addition of 500 examples to our training sample (from 1000 to 1500), as demonstrated by the appreciable increases in both training and validation accuracy, each by more than 10%. Furthermore, by using data augmentation in conjunction with the convolution layer, feature extraction has been improved, leading to improved performance.

Q3: Now change your training sample so that you achieve better performance than those from Steps1 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.

Although adding more data to the training set is a tried-and-true method of improving model performance, figuring out the right sample size can be difficult.

Here, adding 500 more samples to the dataset and using data augmentation techniques led to a significant increase in model performance, which went from 76.5% to 74.45%

The model seems to have limits when it comes to learning new information, even with the enhanced data and bigger sample size in the designated convolutional architecture—a glaring example of this phenomenon.

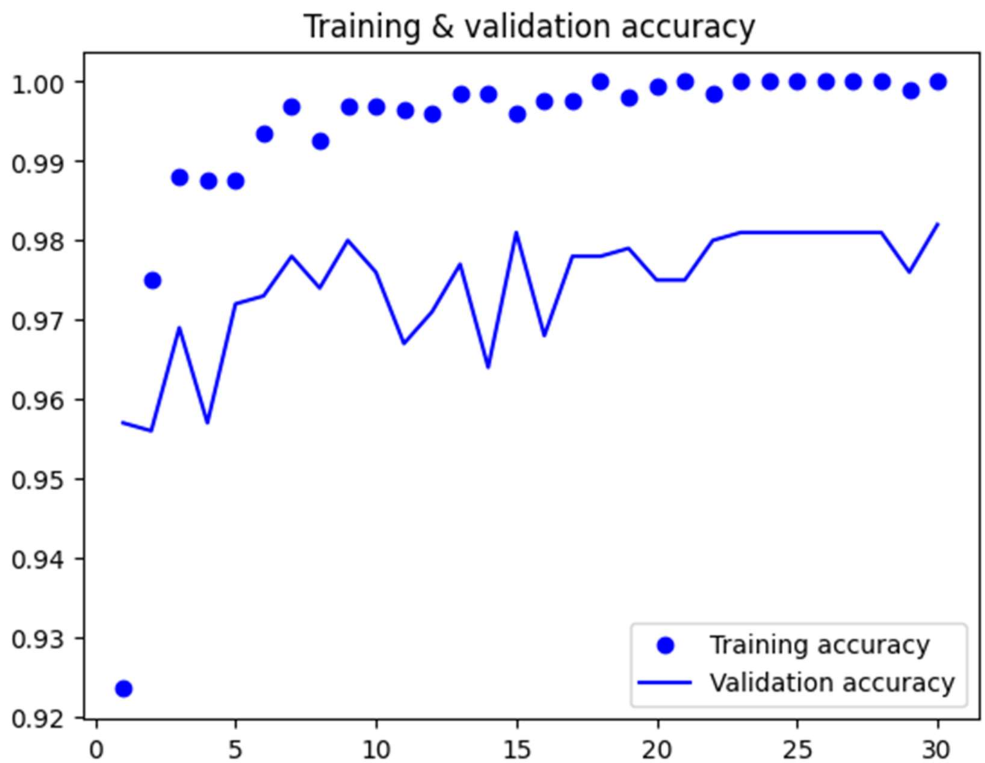
This finding encourages research into different strategies to improve the model's performance even more.

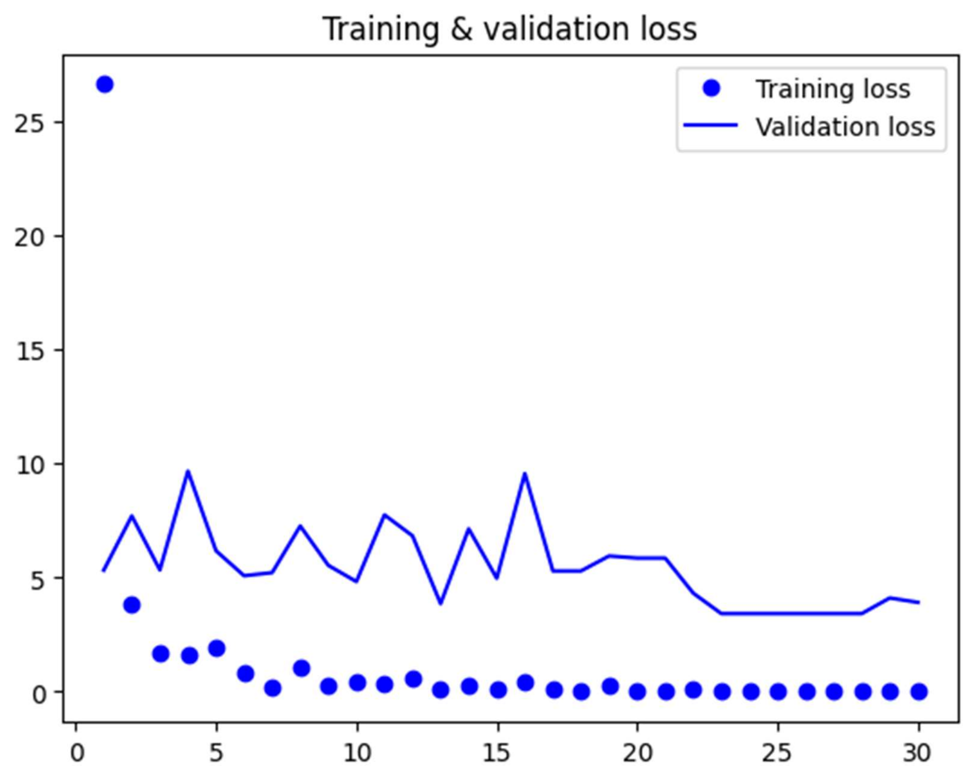
Q4: Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.

The validation accuracy was 98.2% with a corresponding test accuracy of 97.7% when using a pre-trained model without augmentation. Compared to the first training of a smaller model, the test accuracy is promising, but overfitting is a significant cause for concern.

Even though dropout regularization was applied at a relatively high dropout rate, the overfitting phenomenon is highlighted visually in the plots.

The model shows strong performance on the validation data, which was used to fine-tune hyperparameters, despite the dropout plots indicating early signs of overfitting, suggesting potential challenges in generalizing to unseen data.


Training & validation accuracy
1.00
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
0
5
10
15
20
25
30
Training accuracy
Validation accuracy


Training & validation loss
25
20
15
10
5
0
0
5
10
15
20
25
30
Training loss
Validation loss

In conclusion, the study looks at how training data size, validation set size, and data augmentation methods affect the performance of scratch-built and pre-trained models. The following summarizes the main findings:

Whether the model is pre-trained or constructed from scratch, increased accuracy is achieved by either decreasing the size of the validation set or increasing the size of the training set.

For both model types, data augmentation did not significantly improve accuracy.

Because pre-trained models can take advantage of prior task knowledge, they generally perform better than scratch-built models, especially when data is scarce.