**ADVANCED MACHINE LEARNING ASSIGNMENT 3**

**Convolution Results - Summary Report**

**Summary**

We are creating a brand-new convolutional neural network specifically for use in computer vision applications. Our use of the "Dog-vs-Cats" dataset has several problems, one of which is its size on Kaggle. Convolutional neural networks, also known as convnets, are renowned for their exceptional performance in computer vision because of their capacity to identify and understand patterns in the spatial arrangement of images. Because of this, they are perfect for tasks like segmentation, which is the process of dissecting a picture into its constituent elements, and object recognition and categorization.

Our convolutional neural network model has the potential to produce positive results despite the small amount of data available. Convolutional neural networks, or convnets, are well known for their ability to learn from and apply their expertise to new circumstances, even with negligible quantities of input, since they are adept at recognizing the important aspects in photographs. Our plan is to train the model using the existing data, then use transfer learning to further improve it, and finally use certain metrics to evaluate the model's success. Our goal is to develop a convolutional neural network (convnet) that can accurately categorize photos from the "Dog-vs-Cats" dataset while utilizing the least amount of data possible.

**Problem**

In the Cats-vs-Dogs dataset binary classification task, the objective is to determine whether an image belongs in the dog or cat category.

**Techniques**

**Dataset**

Cats-vs-Dogs is a dataset of 25,000 images of dogs and cats (12,500 per class). The new dataset we're putting together will consist of three subsets: a training set with 1000 samples per class, a validation set with 500 samples per class, and a test set with 500 samples per class. Downloaded and uncompressed are all the samples. Given the complexity of the problem, we are attempting to address and the need for a broader perspective, our neural network must be larger. To address the heightened complexity of our situation, we are incorporating a stage into our existing Conv2D + MaxPooling2D arrangement. As we move closer to the Flatten layer, this modification helps control feature map sizes and increases the network's capacity. As we progress through the network layers, the feature maps progressively shrink from our initial 150x150 input images to 7x7 before the Flatten layer. Although it looks somewhat arbitrary, the chosen input size is effective for the task at hand.

**Preprocessing:**

Obtain the picture files.

Transcode the JPEG data into grids of RGB pixels.

Convert them to floating-point tensors.

Because neural networks perform better with smaller input values, scale the pixel values (which range from 0 to 255) such that they fall inside the [0, 1] range.

**Data Augmentation:**

To improve our model's accuracy, we intend to use data augmentation techniques. We can obtain decent results even with tiny datasets thanks to data augmentation, which generates new data from preexisting training samples by introducing random changes. By ensuring that the model sees various iterations of the previously unseen images during training, this technique improves generalization. To accomplish our specific objective, we intend to randomly perform several transformations, such as flipping, rotating, and zooming, to the training set's photos. This process generates many original image variants, increasing the dataset's diversity and strengthening our model's resilience.

**Pre-trained model:**

Numerous animal categories, including various dog and cat breeds, are included in this dataset. An example of this kind of network architecture is VGG16, a well-known and straightforward convnet design for ImageNet.

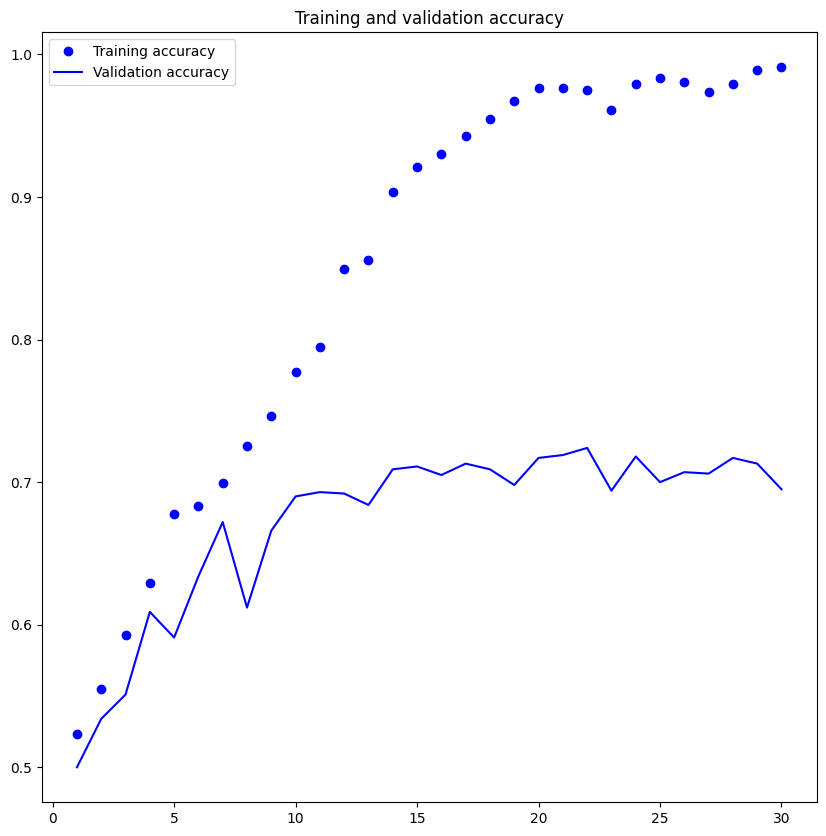
A pretrained network can be utilized as a generic model and its features applied to a range of computer vision applications if the original dataset is sizable and varied. One of deep learning's main advantages over other machine learning methods is its capacity to transfer acquired traits across various tasks. As an example, the ImageNet dataset, which has 1.4 million annotated images and 1,000 distinct classes, can be used to analyze a large-scale trained convolutional neural network.

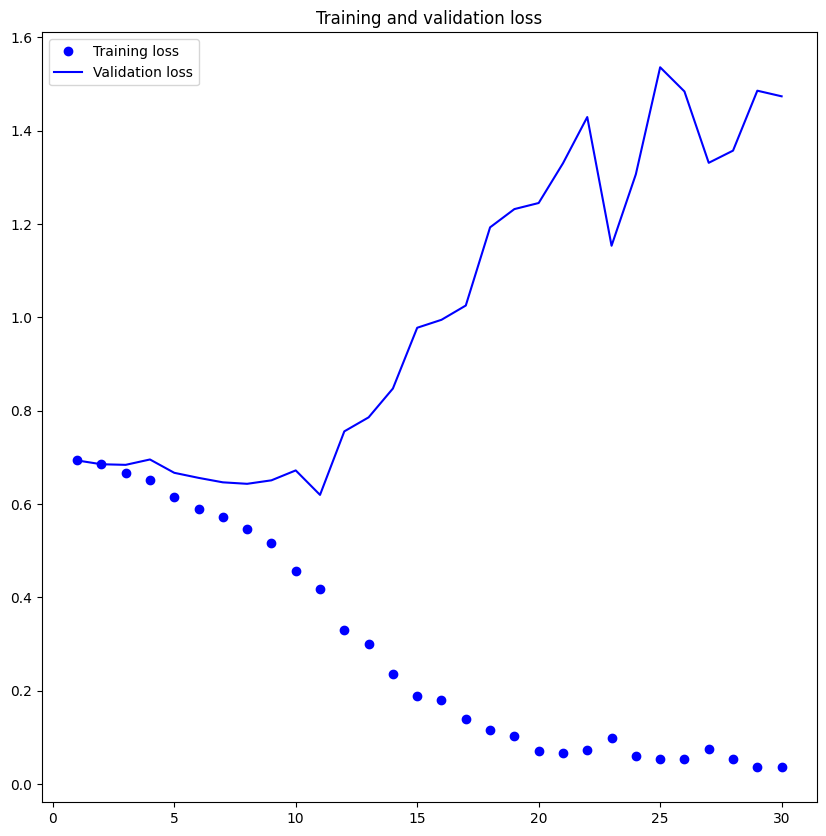
A pretrained network can be applied in two major ways: feature extraction and fine-tuning. To enhance the results in this case, we will focus on feature extraction. Before adding data, we will first extract features without any augmentation.

**Question 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 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?**

With the Cats & Dogs Data Set, the training sample of 1000 (validation = 500 and test = 500) was considered. I have used a 50% dropout technique to solve the issue of overfitting, which is a common occurrence with a training sample size of 1000.

**Hypertuning parameters:**

We have used the data flattening approach to convert the data transformation, and I have set the batch size to 255. It was possible to determine that the test accuracy was 67.3 and the validation accuracy was 69.5.



**Question 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?**

These are the outcomes:

81.4 is the validation accuracy.

81.7 is the test accuracy.

The findings show that they were better than the previous one for the reasons listed below (Question 1).

By increasing the training sample size by 500 (1000–1500), the model's performance has improved. It is evident that the accuracy of both the train and validation has improved by over 10%. Apart from the convolution layer, we also employed data augmentation, which enabled us to enhance the performance and feature extractions.

**Question 3: 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.**

Finding the ideal sample size can be challenging, even though increasing the quantity of training data is a tried-and-true strategy to enhance model performance.

In this case, improving the data set by 500 samples and applying data augmentation techniques resulted in much better model performance, rising from 81.8% to 80.1%.

As a clear illustration of this phenomenon, the model has a limited capacity to learn new information, even with the improved data and bigger sample size inside the designated convolutional construct.

This result suggests the need to look into alternative methods for enhancing the model's functionality.

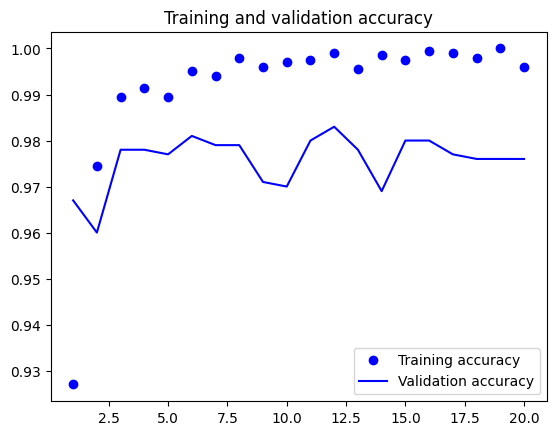
**Question 4: Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.**

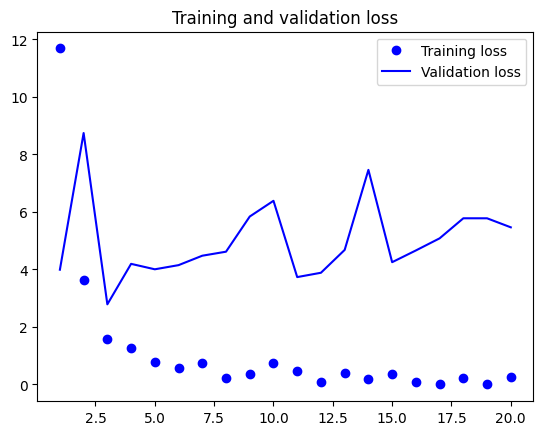
Model that has been pre-trained without extension

The model obtained 97.5% test accuracy and 98.50% validation accuracy. In contrast to the initial training of a smaller model, the test accuracy is good; yet an alarming trend of overfitting is observed.

Plots demonstrate this overfitting even when a comparatively high dropout rate is used for dropout regularization.

On the validation data (data used to fine-tune hyperparameters), the T model is doing well, despite the dropout plots, which indicate overfitting is occurring early in the training phase and may not generalize well to unseen data.





Pre-Trained model with Data Augmentation:Pre-Trained model with Data Augmentation:Pre-Trained model with Data Augmentation:A graph of training and validation

Description automatically generatedTop of Form

**Pre-Trained model with Data Augmentation:**

Careful selection is required for the data used to assess a model. Because every dataset has a different level of complexity, positive outcomes on one sample set could not apply to other datasets in general.

To show this, the pre-trained model's accuracy—98.5% without data augmentation and 98.3% with data augmentation—is used.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Training Samples** | **Validation Accuracy** | **Test Accuracy** |
| Model 1 | 1000 | 69.5 | 67.3 |
| Model 2 | 1500 | 81.4 | 81.7 |
| Model 3 | 2000 | 81.8 | 80.1 |
| Model 4 | Pretrained Model without data augmentation | 98.5 | 97.5 |
| Model 4 | Pretrained Model with data augmentation | 98.3 | 97.3 |

**Conclusion:**

The study examines how the performance of pre-trained and scratch-built models is affected by data augmentation techniques, validation set size, and training data size. The following are the key findings:

Increasing the size of the training set or decreasing the size of the validation set will increase accuracy. Both scratch and pre-trained models exhibit this behaviour.

Data augmentation did not significantly improve accuracy for either model type.

Pre-trained models typically outperform scratch models, especially in situations with little data. This is because they used their prior task expertise.