**ADVANCED MACHINE LEARNING**

**Assignment – 2**

**Uday Kiran Gonuguntla**

**Summary**

We're in the process of developing a novel convolutional neural network tailored for computer vision tasks, leveraging the "Dog-vs-Cats" dataset from Kaggle. The dataset's limited size poses a challenge, but convolutional neural networks excel at recognizing spatial patterns in images, making them ideal for tasks like object detection, classification, and segmentation.

Convolutional layers lie at the heart of a CNN. These layers apply convolution operations on the input data. These convolutional operations work by sweeping a small filter, also called a kernel, across the input data, and computing the dot product between this filter and the input corresponding parts. With such operation local patterns and special features of the input data are caught which in turn helps the network to learn hierarchical representations of the data.

CNNs can additionally be used for self-learning the necessary features relevant to raw data eliminating the routine manual feature extraction. The ability of CNNs to carry out tasks based on complicated and high-dimensional data that are usually represented in the form of images is their advantage over the handcrafted features that may not be useful or effective.

In summary, our goal is to develop a convolutional neural network that can effectively classify images in the "Dog-vs-Cats" dataset while minimizing the amount of training data required.

**Agenda**

The objective of the Cats-vs-Dogs dataset binary classification task is to determine whether an image depicts a dog or a cat using different convolution and reducing overfitting techniques.

**Techniques to reduce Overfitting:**

**Preprocessing:**

Pre-processing implies all steps or procedures that are applied to raw data for inclusion into an analysis or to train a machine learning model. It handles various responsibilities - cleaning, formatting, scaling, and dividing the data into sections that will be used for specific purposes to enhance the accuracy of varying models.

This includes:

Data Cleaning

Data transformation

feature scaling

**Data Augmentation:**

Our objective is to employ data augmentation techniques to improve the accuracy of our model. Data augmentation involves generating new data from existing training samples by introducing random variations, enabling us to achieve reliable results even with limited datasets. By exposing the model to different versions of the images during training, this approach enhances its ability to generalize effectively.

To fulfil this aim, we intend to randomly apply various transformations such as flipping, rotating, and zooming to the images in the training set. Through creation of new sets of data which differ from the one which is used for training, data augmentation helps in improving the performance and robustness of machine learning models by exposing them to large variety of data patterns thus, in turn, increasing their capability to generalize to examples they haven’t seen before.

**Pre-trained model:**

Within this dataset, there are numerous animal classifications, encompassing various breeds of both dogs and cats. An illustrative example of a convolutional neural network architecture suited for ImageNet tasks is VGG16.

When dealing with a sizable and varied original dataset, leveraging a pretrained network proves beneficial as it serves as a versatile model whose features can be adapted to various computer vision tasks. The capability of deep learning to transfer learned features across disparate tasks stands as a significant advantage over alternative machine learning approaches.

Pre-training involves supplementing the model with abundant amounts of labelled instruction data, which helps it to accumulate patterns and features that are critical to the task which is being performed. The model, after training, can be adapted to new tasks or else preserve the current ones, therefore, objectively saving time and computational resources for developers.

Feature extraction and fine-tuning represent the primary approaches for utilizing a pretrained network. In this case, our focus will be on feature extraction to enhance the results. Initially, we'll extract features without data augmentation, followed by the incorporation of augmented data.

**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?**

For the Cats & Dogs Dataset, we utilized a training sample comprising 1000 instances, with validation sample size of 500 and test sample size of 500 each. Acknowledging the potential for overfitting with this training sample size, we implemented a dropout strategy of 0.5 to mitigate this issue.

**Hyper tuning parameters:**

We transformed the data using the flattening technique and set the batch size to 255. Through this process, we determined that the validation accuracy stood at 70.92%, while the test accuracy reached 70.1%.

A graph of a training and validation accuracy

Description automatically generated

A graph of a graph

Description automatically generated

**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?**

We have increased the training sample size from 1000 to 1500. The obtained results indicate a validation accuracy of 79.22% and a test accuracy of 78.6% as we saw in the code.

The results indicate a notable improvement compared to the previous outcomes (Question 1). The augmentation of our training sample by 500 instances (from 1000 to 1500) has significantly enhanced the model's performance, as evidenced by the notable increases in both training and validation accuracy, each by more than 10%. Additionally, the incorporation of data augmentation alongside the convolution layer has further contributed to enhancing feature extraction and ultimately achieving superior 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.**

While augmenting the training data is a well-established strategy for enhancing model performance, determining the optimal sample size can pose challenges.

In this case, employing data augmentation methods and augmenting the dataset with an additional 500 samples resulted in a marked improvement in model performance, increasing from 79.36% to 78.8%.

Despite the augmented data and larger sample size within the specified convolutional architecture, the model appears to exhibit limitations in acquiring new information, showcasing a clear instance of this phenomenon.

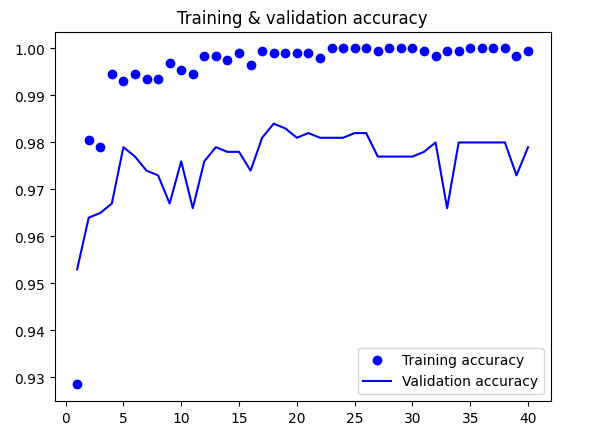
This discovery prompts consideration of alternative approaches to further enhance the model's performance.

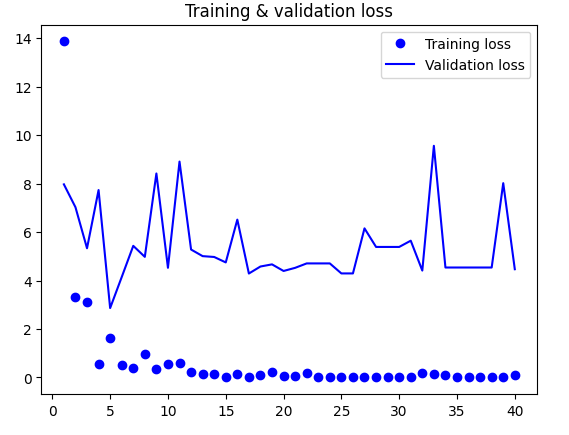
**Q4: Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.**

Using a pre-trained model without augmentation, the validation accuracy reached 87.6%, with a corresponding test accuracy of 87.9%. While the test accuracy shows promise compared to the initial training of a smaller model, there's a notable concern regarding overfitting.

Visual representations of the plots highlight this overfitting phenomenon, despite the application of dropout regularization at a relatively high dropout rate.

Although the dropout plots indicate early signs of overfitting, suggesting potential challenges in generalizing to unseen data, the model demonstrates strong performance on the validation data, which was used for fine-tuning hyperparameters.





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

**Model with Pre-Training and Data Augmentation:**

The selection of data for model evaluation requires careful consideration, as the complexity of each dataset can vary significantly. Achieving favourable results on one dataset may not necessarily translate to success on others.

To illustrate this point, consider the accuracy of the pre-trained model, which achieved 98.3% without data augmentation and 97.4% with data augmentation.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Training Samples** | **Validation Accuracy** | **Test Accuracy** |
| Model 1 | 1000 | 70.92 | 70.1 |
| Model 2 | 1500 | 79.22 | 78.6 |
| Model 3 | 2000 | 79.36 | 78.8 |

**Using a pretrained Model with and without and with data augmentation:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Training samples** | **Validation Accuracy** | **Test Accuracy** |
| Pretrained Model | Pretrained Model without data augmentation | 87.6 | 87.9 |
| Pretrained Model | Pretrained Model with data augmentation | 88.3 | 87.4 |

**Conclusion:**

The study examines the impact of data augmentation techniques, validation set size, and training data size on the performance of pre-trained and scratch-built models. The key findings are outlined as follows:

Increasing the training set size or decreasing the size of the validation set leads to improved accuracy, regardless of whether the model is pre-trained or built from scratch.

Data augmentation did not notably enhance accuracy for either model type.

Overall, pre-trained models outperform scratch-built models, especially when data is limited, owing to their ability to leverage prior task knowledge.