Advance machine learning Assignment

Convolution Report

Harshith Kumar Yadav

**Introduction :-**

Working on google colab , using Kaggle a subset of the well-known "Dogs-vs-Cats" dataset presents me with a challenging opportunity to create a highly effective model with limited data. Convolutional neural networks, or convnets, are well-known for their outstanding capacity to learn and recognize spatial patterns in images, making them the preferred method in computer vision for image recognition, object detection, and segmentation. Despite the limited information provided, I am certain that I will be able to achieve outstanding results by leveraging convnets' ability to extract and identify key properties from photos.

I'd like to train my model on a small dataset, fine-tune it with cutting-edge transfer learning techniques, and then evaluate its performance against relevant assessment criteria. I am dedicated to creating an accurate and efficient convolutional neural network that can accurately classify photographs from the "Dogs-vs-Cats" dataset with minimal input. I am delighted to demonstrate my model's capabilities and driven to push the limits of what is achievable in computer vision with minimum input. By focusing on innovation and efficiency, I am confident that my convolutional neural network will contribute significantly to the field of computer vision.

**Pre-trained model :-**

Since the original dataset is broad and diverse, a pretrained network can be utilized as a generic model, having properties that apply to a wide range of computer vision applications. One of the most significant advantages of deep learning over other machine learning algorithms is its ability to transfer learned properties across tasks. Consider a massive convolutional neural network trained on the ImageNet dataset, which has 1.4 million annotated images and 1,000 different classifications. This collection includes numerous animal classifications, including cat and dog breeds. This network's design, known as VGG16, is a simple and widely used convnet architecture for ImageNet.

We propose using data augmentation methodologies to increase the accuracy of our model. We can achieve decent results even with limited datasets by creating new data from the provided training samples using random tweaks. As a result, the model will never see the same image twice when training, which aids in generalization. For our specific purpose, we want to randomly modify the photographs in the training set, such as flipping, rotating, and zooming. By doing so, we can develop versions of the present photographs, increasing the variety of the dataset and improving our model's resilience.

**Data Augmentation:**

The Cats-vs-Dogs dataset is a case of binary classification in which you must determine whether a given image belongs to the dog or cat category.   
• Open the image files.   
• Convert the JPEG material into RBG pixel grids.   
• Convert them to tensors in floating point.   
• Rescale the pixel values (between 0 and 255) to the [0, 1] range (as you know, neural networks like small input values).

The Cats-vs-Dogs dataset has 25,000 pictures of dogs and cats (12,500 from each class) and is 543MB in size. (compressed). After downloading and uncompressing it, we will create a new dataset with three subsets: a training set with 1000 samples of each class, a validation set with 500 samples of each class, and finally a test set with 500 samples of each class. We need to increase the capacity of our neural network because to the larger picture size and more complex nature of the task we're working on. To do this, we will incorporate a new stage into our existing Conv2D + MaxPooling2D design. This will boost network capacity while also reducing the size of the feature maps, ensuring that they do not get too large when we reach the Flatten layer. Our input photographs are initially 150x150 in size, and as we progress through the network layers, the feature maps gradually diminish in size until they reach 7x7 immediately before the Flatten layer. This input size selection is pretty arbitrary, but it is appropriate for the given situation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model no** | **Train**  **Size** | **Validation and Test sample size** | **Data Augmentation** | **Test Accuracy%** | **Validation  Accuracy%** |
| Model 1 | 1000 | 500,500 | NO | 76.8 | 70.6 |
| Model 1a | 1000 | 500,500 | YES | 67.1 | 64.2 |
| Model 2 | 1500 | 500,500 | NO | 83 | 71.9 |
| Model 2a | 1500 | 500,500 | YES | 70.37 | 70.3 |
| Model 2b | 1500 | 500,500 | YES | 81.7 | 73.2 |
| Model 2c | 1500 | 500,500 | NO | 72.7 | 73.8 |

***Table for Pre-Trained Models***

|  |  |  |
| --- | --- | --- |
| **Data Augmentation** | **Train**  **Accuracy %** | **Validation  Accuracy%** |
| NO | 99.6 | 97 |
| YES | 95.8 | 97.2 |

Summary:

The model settings and sample sizes for the train, test, and validation sets are shown in the tables above. We provide results with and without data augmentation for the model built from scratch, as well as models trained with a larger train size or with different train and validation sizes. We compare the accuracy, validation accuracy, and data augmentation of the pre-trained model.

The results show that models trained with data augmentation did not consistently outperform those trained without it. Increasing the size of the training set or adjusting the size of the validation set improves the model's accuracy. When we examine the pre-trained model with and without data augmentation, we see that data augmentation did not improve the model's or validation accuracy. Overall, pre-trained models outperform models built from scratch, especially when limited training data is available.