**CONVOLUTIONAL NEURAL NETWORKS**

**Siddhartha Chandrabanda**

**811290812**

The aim of this research is to develop a convolutional neural network that can accurately and efficiently recognize photos of dogs and cats by recognizing their distinctive characteristics.  
The Kaggle dataset used in this experiment contains 12,500 test and 25,000 training images, including dogs and cats in equal numbers.

**Problem to be defined:**

The Cats-vs-Dogs dataset aims to determine if a picture belongs to the dog or cat class.

**Methods:**

**Data:**

The Cats-vs-Dogs dataset, which has a compressed size of 543MB, has 25,000 images that are split evenly between cats and dogs. After downloading and unzipping the dataset, I created a new dataset with three subsets:

* Training dataset with 1000 samples from every class
  + Validation dataset with 500 samples
  + Test dataset with 500 samples

We need to increase the size of our neural network since the problem we are working on is more complicated and requires a larger picture. This will be accomplished by adding a step to our current Conv2D + MaxPooling2D design. This not only increases the capacity of the network but also ensures that   
  
When we get to the Flatten layer, the feature maps aren't very large. Our input images are initially 150x150 pixels in size. Prior to the Flatten layer, the feature maps get smaller as we move up the network's tiers, reaching 7x7. The input size selection appears rather arbitrary, but it is appropriate for the task at hand.

**Data Preprocessing:**

• Examine the image files.

• Create RBG pixel grids from the JPEG data.  
• Use grids to construct floating point tensors.  
Since neural networks function best with small input values, the pixel values, which vary from 0 to 255, should be rescaled to the [0, 1] interval.

I took batch size as 255 and applied the data flattening technique to convert the data transformation. With the help of 10 epochs, we got to know the validation accuracy as 65% and test accuracy as 71%.

A graph with blue dots

Description automatically generated A graph with blue dots

Description automatically generated

From the above result we can conclude that the test accuracy with no data augmentation is about 66.4% when the Training accuracy is about 48.4%.

**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?**

**Data Augmentation:**

The model's accuracy can be increased by using this method. One method that enables reliable results to be obtained even from limited datasets is data augmentation. To create fresh data, the provided training samples must be subjected to random modifications. By ensuring that the model sees a wide variety of images during training, this technique increases the model's ability to generalize successfully.

The training sample of 1500 and the validation test of 500 form the basis of all the results below.

showing the trained augmented pictures

Test Accuracy: 51.6% are evident from the validation accuracy of 51%, which may be attributed to the following factors:

The following are the reasons why the model's performance has improved:   
In order to boost test and validation accuracy by nearly 10%, we added 500 (1000–1500) training samples. We also added data augmentation to the convolution layer, which helped us enhance the featured extractions and enhance performance.

A collage of a dog

Description automatically generated

**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**

* Since using more and more data would help to improve the model's performance, we are unable to estimate the right sample size.
* To do this, test sets of 500 samples and 2000 training samples with confirmation. I've discovered that the test accuracy is greater with 1500
* Training accuracy increases when there are 1000 training samples.
* Increasing the training sample to 2000 while keeping the validation and test sets at 500 samples each

**Results:**

|  |  |  |  |
| --- | --- | --- | --- |
| Training samples | Validation accuracy | Test accuracy | Data augmentation |
| 1000 | 65% | 67% | NO |
| 1500 | 50% | 58% | YES |
| 2000 | 77% | 73.6% | YES |

**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.**

The main applications of trained networks are in fine-tuning and feature extraction. If the original dataset of a pretrained network is large and varied, it may be used as a generic model with its features applied to many computer vision applications. Applying learnt characteristics to a range of tasks is one of the main advantages of deep learning over previous machine learning methods. With 1.4 million annotated photos and 1,000 distinct classifications, the ImageNet dataset serves as an illustration of how to analyze a sizable trained convolutional neural network. The collection includes a wide variety of animal species, including several dog and cat varieties. VGG16 is a popular and straightforward convolutional neural network design for ImageNet.

In this case, feature extraction will be used to improve the outcomes, initially without data augmentation and then with it.

Our train accuracy is 99.4% and our validation accuracy is 96.2%. Even if we use dropout at a rather high rate, the plots show that we are overfitting practically instantly.

A graph of training and validation

Description automatically generated A graph of training and validation

Description automatically generated

Pre-Trained model with Data Augmentation:

* validation accuracy is 94.8%
* train accuracy is 99.8%

Fine-tuning a pretrained model

• validation accuracy is 50%

• train accuracy is 50.98%

**TABLE FOR PRE-TRAINED MODEL**

|  |  |  |
| --- | --- | --- |
| Data Augmentation | Train Accuracy (%) | Validation Accuracy (%) |
| NO | 50.98% | 50% |
| YES | 99.8% | 94.8% |

**CONCLUSION**

Using a small training set of 1000 samples, we were able to achieve a 48.4% training accuracy.  
The reduction of overfitting is the aim of data augmentation.   
  
Strategies to stop overfitting:

* It's not always possible to increase the training sample. Data augmentation is one technique to maximize the restricted amount of training data.
* The amount of overfitting that happens when the model's size is reduced depends on the number of learnable parameters in the model, or the number of layers and units in layers.
* restricting a network's complexity and preventing or reducing overfitting may be achieved by regularizing the distribution of weight values by restricting them to only take extremely tiny values.
* Zeroing off a few of the layer's output properties during training is a smart method to reduce overfitting. A "dropout rate" is the percentage of traits that are null.

The sample sizes and model parameters for the train, test, and validation sets are shown in the tables above. Both the results for the original model without data augmentation and the results with augmentation for the trained models with varying train and validation sizes or with an increase in train size are shown. We contrast the data augmentation, accuracy, and validation accuracy for the pre-trained model.

The model's accuracy is increased by either using a larger training set or a different size validation set. The accuracy and validation accuracy of the pre-trained model with and without data augmentation were not enhanced by data augmentation. Overall, pre-trained models outperform models created from scratch when there is a shortage of training data.