**CONVOLUTION REPORT**

**Pallepati Pallavi**

**Chaitanya Vishnu   
(GROUP 1)**

**Introduction:**

The goal of this project is to build a convolutional neural network (CNN) that can precisely and reliably identify between images of dogs and cats based on their unique characteristics. A dataset of 25,000 training images and 12,500 test images, evenly split between cats and dogs, are used in the experiment from Kaggle.

**Problem statement:**

The Cats-vs-Dogs dataset is designed to classify images into either the dog or cat category.

**Methods used:**

Data:

There are 25,000 photos in the Cats-vs-Dogs dataset, which has a compressed size of 543 MB and is divided equally between cats and dogs. I extracted the dataset and then downloaded it to create a new dataset with three subsets:

Dataset for training: 1000 samples total from each class   
Dataset for validation: holding 500 samples

Test dataset:

500 samples are also included.   
We have to expand the size of our neural network due to the intricacy of the current problem and the requirement for a more comprehensive image. We add a step to our current Conv2D + MaxPooling2D architecture in order to accomplish this. This augmentation makes sure that the feature maps don't get unduly big as we get closer to the Flatten layer, in addition to increasing the network's capacity. First, our input images have 150 x 150 dimensions.

The feature maps get smaller as we move through the network's layers, reaching a 7x7 size right before the Flatten layer. The input size selection may appear random, but it works well for the purpose at hand.

**Data Preprocessing:**

- Examine the picture files.   
- Convert the JPEG data to arrays of RGB pixels.   
Transform the pixel arrays into tensors with floating points.   
Neural networks function best when their input values are smaller, so in order to maximize neural network performance, the pixel values, which range from 0 to 255, must be normalized to fall within the [0, 1] range.   
To convert the data transformation, I used data flattening and selected a batch size of 255. We discovered that after 30 epochs of operation, the test accuracy was 73.7% and the validation accuracy was 70.5%.

A screenshot of a graph

Description automatically generated

**Question 2:**

**Data Augmentation:**

Utilizing this strategy can enhance the model's accuracy. Data augmentation is a technique that enables obtaining reliable results even with limited datasets. It involves applying random modifications to the existing training samples to generate additional data. This approach ensures that the model is exposed to a diverse range of images during training, thereby enhancing its ability to generalize effectively.

A training sample of 1500 images and a validation set of 500 images serve as the foundation for all ensuing results.   
The enhanced images that were trained

A collage of a dog

Description automatically generated

The test accuracy exceeds the results in question 1 by 80.9%, and the validation accuracy is 80.3% higher.   
Potential Causes of the Model Performance Improvement:   
The following factors are responsible for the improvement in the model's performance:   
• Data augmentation combined with the convolutional layer led to better feature extraction and better performance. • The addition of 500 training samples (from 1000 to 1500) resulted in nearly a 10% increase in test and validation accuracy.

**Question 3:**

There is no way to know for sure what the optimal sample size is because it is generally accepted that using larger datasets tends to enhance model performance.   
  
• We used test sets with 2000 training samples with validation and 500 samples for training in our experiment. In contrast to training samples of 1000 and 2000, we found that test accuracy is higher when 1500 images are used.   
• We observed an increase in training accuracy with 1000 training samples.

**RESULT :**

|  |  |  |  |
| --- | --- | --- | --- |
| Training samples | Validation Accuracy | Test Accuracy | Data Augmentation |
| 1000 | 70.5 | 73.7 | NO |
| 1500 | 80.3 | 80.9 | YES |
| 2000 | 98.1 | 97.9 | YES |

**Question 4:**

**Pre-trained Model:**

The main uses of trained networks are in feature extraction and fine-tuning. A pretrained network can function as a flexible model whose acquired features can be used for a range of computer vision tasks after it has been trained on a sizable and diverse dataset. Deep learning differs from other machine learning techniques in that it can transfer learned characteristics across different tasks, which is one of its main advantages.

ImageNet is a sample dataset for pretrained convolutional neural network analysis. It comprises 1.4 million annotated images in 1,000 classes, including many animal categories like different dog and cat breeds. VGG16 is a popular architecture in this context; it is a simple convolutional network design specifically made for the ImageNet dataset.

In this case, we will use feature extraction to enhance the outcomes, first without using data augmentation methods and then adding them later.

A graph of training and validation loss

Description automatically generated

Validation accuracy for the pre-trained model without data augmentation is 97.8%, and train accuracy is 99.8%.   
Pre-trained model utilizing augmented data:   
Test accuracy is 97.9%; train accuracy is 98.5%; validation accuracy is 98.1%.   
Perfecting a previously trained model   
• 98.1% validation accuracy   
• The train's 99.75% accuracy   
• 98.0% test accuracy.

**Conclusion:**

With a brief training set consisting of 1000 Samples, we were able to achieve a 98% training accuracy.

**Methods for Mitigating Overfitting:**

* Increasing the size of the training dataset might not always be possible. Augmenting data is one way to make the most of the limited amount of training data available.
* The number of learnable parameters in the model, including the number of layers and units within those layers, affects how much overfitting occurs.
* Restricting the weights to extremely small values can help prevent or lessen overfitting, which will reduce the network's complexity by regularizing the weight distribution.
* One useful technique to lessen overfitting is to introduce dropout during training. A portion of the layer's output features are randomly set to zero in a process known as dropout; the percentage of features that are zeroed out is indicated by the dropout rate.

The model configurations and sample sizes for the training, testing, and validation datasets are shown in the previously mentioned tables. For models trained with different sizes of training and validation sets, or with an increase in the training size, we present results both with and without data augmentation for the original model. We also compare validation accuracy, overall accuracy, and the effect of data augmentation for the pretrained model.   
The model's accuracy increases with both increasing the size of the training set and modifying the size of the validation set. On the other hand, we found no improvement in validation accuracy or accuracy when comparing the pretrained model with and without data augmentation. Notably, pretrained models typically perform better than models built using limited training data from scratch.