Introduction

This report provides a detailed overview of the image classification project using a convolutional neural network (CNN) implemented in the Colab environment. The goal is to classify images of cats and dogs using a dataset and evaluate the model's performance.

Data Acquisition and Preprocessing

* Dataset: The dataset was downloaded from Microsoft using the wget command and extracted using unzip.
* Path Management: Image paths and labels were extracted using os.listdir(). Labels were assigned as 0 for Cats and 1 for Dogs.
* Data Cleaning: Invalid and corrupted images such as Thumbs.db and specific broken image files were removed.
* Visualization: A random selection of 25 cat and dog images was plotted using matplotlib for visual inspection.

Data Splitting

* The data was split into training and test sets using an 80-20 ratio with train\_test\_split().
* Further, an additional split was made to create a validation set from the training data.

| Dataset | Number of Images |
| --- | --- |
| Training | 15998 |
| Validation | 4000 |
| Testing | 5000 |

Data Augmentation

* Applied data augmentation using ImageDataGenerator with operations like:
  + Rescaling
  + Rotation (40 degrees)
  + Shear transformation
  + Zooming (up to 20%)
  + Horizontal flip
  + Filling with nearest neighbor interpolation
* This technique helps in reducing overfitting.

Model Architecture

A CNN was designed using the Keras Sequential API. The architecture consists of:

* Convolutional Layers: Three Conv2D layers with ReLU activation
* Pooling Layers: MaxPooling layers to reduce feature map dimensions
* Flatten Layer: Converts feature maps into a one-dimensional vector
* Dense Layers: Fully connected layers with 512 neurons
* Output Layer: A single neuron with a sigmoid activation for binary classification

Optimizer: Adam  
Loss Function: Binary Crossentropy  
Metrics: Accuracy

Model Training

* The model was trained for 10 epochs using model.fit().
* The training process included real-time data augmentation.

| Epoch | Training Accuracy | Validation Accuracy | Training Loss | Validation Loss |
| --- | --- | --- | --- | --- |
| 1 | 78.81% | 79.72% | 0.4503 | 0.4412 |
| 5 | 80.14% | 81.72% | 0.4246 | 0.4098 |
| 10 | 82.56% | 81.96% | 0.3864 | 0.3928 |

Evaluation and Visualization

* Training and validation accuracy and loss were plotted using matplotlib.
* Results show that the model achieved 82.56% accuracy on the training data and 81.96% accuracy on the validation data, with no significant overfitting.

Conclusion

* The implemented CNN demonstrated strong performance in classifying cats and dogs.
* Future improvements could involve trying additional architectures, increasing epochs, or fine-tuning hyperparameters.

Recommendations

* Experiment with more complex architectures like VGG or ResNet.
* Apply further data augmentation techniques.
* Perform hyperparameter tuning to optimize the model.

References

* Keras Documentation
* Microsoft Dataset Repository
* Colab Environment Guide