Github Link : https://github.com/nayoo1119/CECS_456_Final_Project

1. Introduction

Image classification is one of the most common applications of deep learning, where Convolutional Neural Networks (CNNs) have proven to be highly effective. This project focuses on classifying images into eight categories using a custom-designed CNN. The Natural Images dataset, which contains diverse image classes such as airplanes, cars, and animals, serves as the foundation for training and evaluating the model. The main objective of this project is to develop a CNN that achieves high accuracy while demonstrating the ability to generalize effectively across the dataset.

2. Dataset and Related Work

Dataset: The dataset used is the Natural Images Dataset sourced from Kaggle. It contains 8 categories:

- Airplane / Car / Cat / Dog / Flower / Fruit / Motorbike / Person

Key Details:

- Images of 13,798 total and split into 80% training and 20% validation data.
- Resizing all images to 128x128 pixels.
- Normalizing pixel values to the range [0, 1].

Related Work:

CNNs like AlexNet, VGG, and ResNet have set benchmarks in image classification tasks. These models demonstrate the power of convolutional layers to learn hierarchical spatial features. Inspired by these architectures, this project uses a simplified CNN to effectively classify the Natural Images dataset.

3. Methodology

The CNN was designed to classify images into 8 categories. The architecture includes:

1. Input Layer: Accepts RGB images of size (128, 128, 3).
2. Convolutional Layers: Three convolutional layers with 64, 128, and 128 filters, kernel sizes of 7x7, 3x3, and 3x3, respectively, and ReLU activation.
3. Pooling Layers: MaxPooling layers with a pooling size of (2, 2) to reduce spatial dimensions.
4. Flatten Layer:Converts the feature maps into a vector for input into dense layers.
5. Fully Connected Layers:A dense layer with 128 units and ReLU activation and An output layer with 8 units and softmax activation for multi-class classification.

The model was compiled with:

- Loss Function: Categorical Crossentropy (suitable for multi-class classification).
- Optimizer: Adam (adaptive learning rate optimization).

- Metrics: Accuracy to track performance.

4. Experimental Setup

Environment: The project was implemented in Google Colab, utilizing GPU acceleration for faster training. TensorFlow/Keras libraries were used to implement the model.

Training Details

- Batch Size: 32 / Epochs: 10 / Validation Split: 20%.

Data Generators

Training and validation datasets were prepared using ImageDataGenerator, which rescaled images and ensured the correct class distributions.

5. Measurement

The model's performance was evaluated using the following metrics:

- Accuracy: Measures the proportion of correctly classified images.
- Loss: Quantifies the prediction error.

During training, both metrics were tracked for the training and validation datasets.

6. Results Analysis, Intuitions, and Comparison

Results

- Validation Accuracy: 99.29% / Validation Loss: 0.0218.

Analysis

- The high validation accuracy (99.29%) demonstrates the effectiveness of the CNN in learning features across diverse classes. The low validation loss indicates minimal prediction error.

Intuitions : Effective architecture with sufficient convolutional layers and Adequate training data across all 8 classes.

Comparison

- This model performs comparably to state-of-the-art architectures for similar tasks, demonstrating that even a simple CNN can achieve excellent results when applied to well-prepared datasets.

7. Conclusion

This project successfully classified images into 8 categories using a CNN. The model achieved a validation accuracy of 99.29% and demonstrated excellent generalization capabilities. This success highlights the potential of CNNs for image classification tasks and the importance of dataset preparation and architectural design.

8. Contributions

This project was completed as an individual effort by Nay Thuya Oo.