Deep Learning Image Classification with CNN

Diego Rosa Paz
Sylvia Pérez Montero

January 24, 2025

Dataset Overview

1. Dataset Used:

- **CIFAR-10:** A dataset consisting of 60,000 color images (32x32 pixels) across 10 classes.
    - **Classes:** Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, Truck.
- The dataset is split into two sets:
    - **Training set:** 50,000 images.
    - **Testing set:** 10,000 images.

2. Shape of the Data:

- **Training Data:**
    - **Features** (X_train): (50,000, 32, 32, 3).
    - **Labels** (y_train): (50,000, 1).
- **Testing Data:**
    - **Features** (X_test): (10,000, 32, 32, 3).
    - **Labels** (y_test): (10,000, 1).

Model Architecture

Base Model

DenseNet121, a Convolutional Neural Network (CNN), was used as the backbone model for this project. Also known as Dense Convolutional Network, it is a pre-trained deep learning model designed for image classification.

- It connects each layer to every other layer in a feed-forward manner.
- Its architecture promotes feature reuse, reduces vanishing gradients, and minimizes the number of parameters required compared to other architectures like ResNet.

- Because it is pre-trained on ImageNet (1.2 million+ images x 1,000 classes), the model has a strong feature extractor.

Some modifications were applied for the CIFAR-10 dataset:

- Global average pooling was applied to reduce feature maps into a single vector (pre-trained base).
- A fully connected, or dense, layer with 10 output neurons was added for multi-class classification.
- The softmax activation function was applied to generate outputs class probabilities.

Training Configuration

- **Loss function:** categorical cross-entropy
- **Optimizer:** Adam was chosen for its adaptive learning rate and convergence speed.
- **Metrics:**
    - Accuracy - percentage of correctly classified samples
    - Precision - ratio of correctly predicted positive instances.
    - Recall - ratio of actual positives correctly predicted.

Data Augmentation

- To improve generalization, horizontal flipping and width/height shifts were applied.
    - The horizontal flipping randomly flips images along the horizontal axis.
    - The width / height shifts randomly shifts the image by up to 10% of its dimensions.

Key Steps in the Analysis

1. **Data Visualization**
    a. Random Image Grid
        i. Displayed a 10 x 10 grid of randomly selecting training images with their corresponding class label.
        ii. Verified the diversity and correctness of the dataset.
    b. Class Distribution
        i. Feature Scaling was performed to improve the numerical stability during training.
        ii. One hot encoding was used to encode the labels compatibly with the model.
2. **Training Configuration**
    a. 100 epochs were set with early stopping enabled, to monitor validation loss and prevent overfitting.

Results

1. Training Performance
    a. Accuracy:
        i. Training ~95%
        ii. Validation: ~85%
    b. Early stopping occurred after 20 epochs. This demonstrates good generalization while avoiding overfitting.
2. Evaluation on Test Set (results presented are averaged across all classes)
    a. Test accuracy: 84.7%
    b. Precision: 85.2%
    c. Recall: 83.4%
3. Visualization of Predictions
    a. Certain classes with overlapping features (for example, cat vs dog) showed higher confusion. This contributed to reduced precision and recall.

Analysis

Positives

- The application of DenseNet121 allowed the model to generalize well to CIFAR-10
- The balance within the dataset contributed to consistent precision and recall across all classes.
- Flipping and shifting appear to have improved robustness

Negatives

- The dataset's resolution (32 x 32) limited the model's ability to capture fine details.
- Feature overlaps across certain classes and the limited resolution resulted in occasional misclassifications.
- The early stopping feature may have had an impact on the model's ability to optimize further.

Further research is recommended to evaluate model optimization. Some of the tuning that could be experimented with are:

- Resizing images / increasing the input resolution
- Fine-tuning the pre-trained layers of DenseNet121
- More augmentation techniques