# Training and Evaluation of a Convolutional Neural Network (CNN) on CIFAR-10 Dataset

Semih Özdemir

# Purpose of the project

- The goal of this project is to develop a Convolutional Neural Network (CNN) and evaluate its performance on the CIFAR-10 dataset.
- This presentation will cover data preprocessing, model architecture, training process, evaluation results and challenges faced.

# Dataset

- The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 different classes. Each class has 6,000 images.
- **Training Set:** 50,000 images are used for training.
- **Test Set:** 10,000 images are used for testing.
- **Classes:** The classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

# Data Preprocessing

- The images are normalized to have pixel values between 0 and 1 by dividing by 255.0. This helps in faster convergence during training.
- **One-Hot Encoding:** The class labels are converted into one-hot encoded vectors, transforming categorical labels into binary vectors. This is essential for multi-class classification tasks.

# Model Architecture

- **Layers:** The CNN model consists of several layers:
- Input Layer: 32x32x3 dimensions for the color images.
- Convolutional Layers: Three Conv2D layers with 32 and 64 filters.
- MaxPooling Layers: Two MaxPooling2D layers to reduce spatial dimensions.
- Flatten Layer: Converts 2D matrices into a vector.
- Dense Layers: Two dense layers with 64 and 10 units respectively.
- Functions: Each convolutional layer uses the ReLU activation function. The final dense layer uses the softmax activation function for classification.

# Model Compilation

- **Loss Function:** The model uses categorical cross-entropy as the loss function, which is suitable for multi-class classification problems.

  **Optimizer:** The Adam optimizer is used for training the model due to its efficiency and adaptive learning rate properties.

  **Metrics:** Accuracy is used as the primary metric to evaluate model performance during training and testing.

# Training Process

- **Parameters:** The model is trained for 10 epochs with a batch size of 32. These parameters were chosen to balance training time and performance.
  **Datasets:** The training set consists of 50,000 images, and the validation set consists of 10,000 images. Data augmentation techniques could be applied to enhance the training process.

# Custom Callback

- **MetricsCallback:** A custom callback class called MetricsCallback is defined to calculate and display precision, recall, and f1-score at the end of each epoch.  
  **Purpose:** This callback helps in monitoring the model's performance using additional metrics beyond accuracy, providing a more comprehensive evaluation.

# Model Training Results

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| airplane | 0.84 | 0.58 | 0.69 | 1000 |
| automobile | 0.84 | 0.8 | 0.82 | 1000 |
| bird | 0.59 | 0.57 | 0.58 | 1000 |
| cat | 0.54 | 0.44 | 0.48 | 1000 |
| deer | 0.56 | 0.71 | 0.63 | 1000 |
| dog | 0.63 | 0.6 | 0.62 | 1000 |
| frog | 0.56 | 0.9 | 0.69 | 1000 |
| horse | 0.77 | 0.69 | 0.73 | 1000 |
| ship | 0.8 | 0.82 | 0.81 | 1000 |
| truck | 0.86 | 0.72 | 0.78 | 1000 |
| | | | | |
| Accuracy | | | | 0.68 |
| Macro Avg | 0.7 | 0.68 | 0.68 | 10000 |
| Weighted Avg | 0.7 | 0.68 | 0.68 | 10000 |

# Challenges Faced

- **Challenges:** One of the most difficult aspects was determining an appropriate layer configuration and deciding which metrics would be meaningful.
- **Solutions**: I researched and learned through experimentation to solve my challenges.