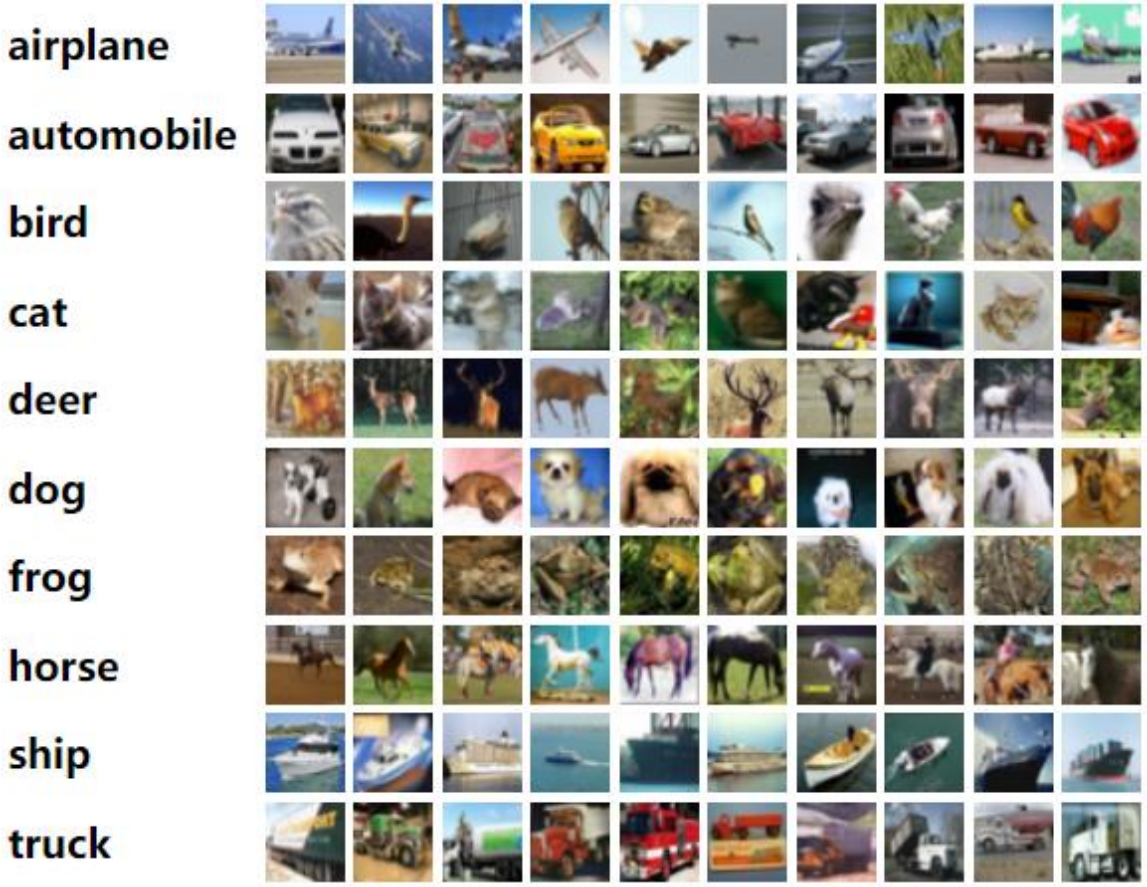
Intro to AI – Final Project

Nitzan Ron ID 215451709, Adam Zelzer ID 328489166

Dataset Information and Feature Description

Dataset

The **CIFAR-10** dataset contains 60,000 32x32 color images in 10 distinct classes:



Data Preparation

The dataset was split into 50,000 training and 10,000 testing images. Images were normalized by scaling pixel values to a range of [0, 1]. Labels were one-hot encoded for compatibility with the neural network.

Algorithms and Learning Techniques

Principal Component Analysis (PCA)

PCA was used to reduce the dimensionality of the flattened image data from 3,072 features down to 40. This allowed faster training of traditional machine learning models while retaining essential information.

Models Tested

Logistic Regression, Perceptron, Decision Tree, K-Nearest Neighbors (KNN), Random Forest, AdaBoost, XGBoost. These models were chosen because of their popularity in machine learning tasks, as well as the relative diversity of model families – linear, tree, ensemble, and boosting.

Convolutional Neural Networks (CNN)

A CNN architecture was designed with convolutional layers followed by max pooling, dropout layers, and fully connected layers. 3 versions were designed.

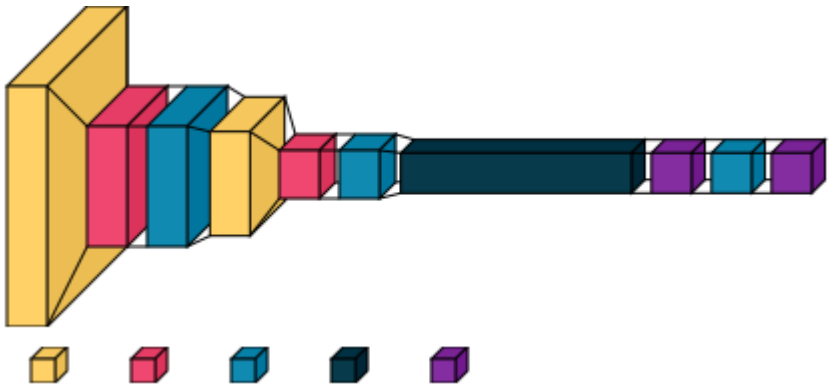
CNNs were chosen because they are specifically designed for image data, making them ideal for tasks like image classification. Unlike traditional models, CNNs automatically learn spatial hierarchies and local patterns, which are crucial for distinguishing objects in images. Their ability to handle high-dimensional data efficiently, by reducing the number of parameters through shared weights and feature maps, makes them well-suited for large datasets like CIFAR-10. Additionally, advanced techniques like data augmentation and dropout further enhance their performance, making CNNs the optimal choice for this project.

Version 1: Basic CNN Architecture

Architecture Overview: This version consists of two convolutional layers followed by max pooling and dropout, ending with a fully connected layer. It serves as the baseline architecture.

- **Conv2D Layer 1**: 32 filters of size 3x3, ReLU activation.

- **MaxPooling2D**: Pool size of 2x2 to reduce spatial dimensions.
- **Dropout**: 25% dropout to prevent overfitting after the first convolution.
- **Conv2D Layer 2**: Another set of 32 filters of size 3x3, ReLU activation.
- **MaxPooling2D**: 2x2 pooling again to further reduce dimensions.
- **Dropout**: 25% dropout after the second convolution.
- **Flatten Layer**: The 2D feature maps are flattened into a 1D vector.
- **Dense Layer**: 128 neurons, ReLU activation.
- **Dropout**: **50**% dropout after the dense layer.
- **Output Layer**: A softmax layer with 10 neurons corresponding to the 10 CIFAR-10 classes.

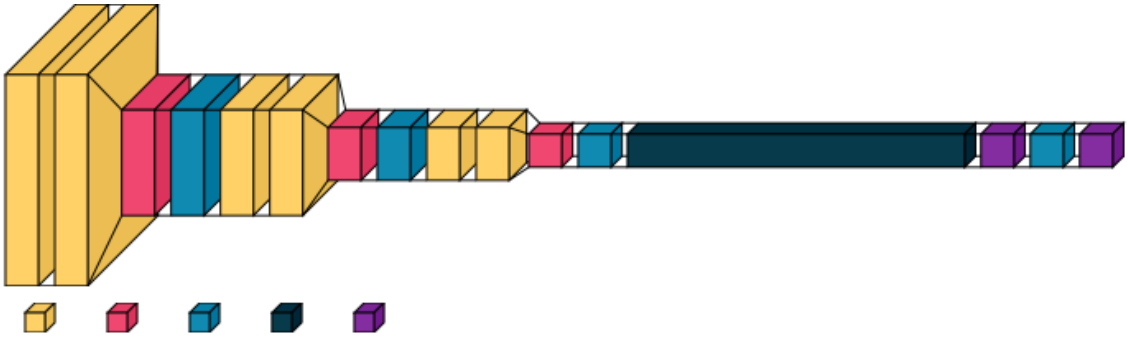


Version 2: Expanded CNN

Architecture Overview: This version builds upon the first architecture by introducing more convolution layers, as well as a more gradual dropout to decrease overfitting. It also increases the number of filters in each convolutional layer.

- **Conv2D Layer 1+2**: 32 filters of size 3x3, ReLU activation.
- **MaxPooling2D**: Pool size of 2x2.
- **Dropout**: 20% dropout after the first pooling layer.
- **Conv2D Layer 3+4**: 64 filters of size 3x3, ReLU activation.
- **MaxPooling2D**: 2x2 pooling.
- **Dropout**: 30% dropout.

- **Conv2D Layer 5+6**: Additional convolutional layers with 128 filters of size 3x3.
- **Dropout**: 40% dropout.
- **Flatten Layer**: Feature maps are flattened.
- **Dense Layer**: 256 neurons, ReLU activation
- **Dropout**: 50% dropout.
- **Output Layer**: Softmax layer for 10 classes.

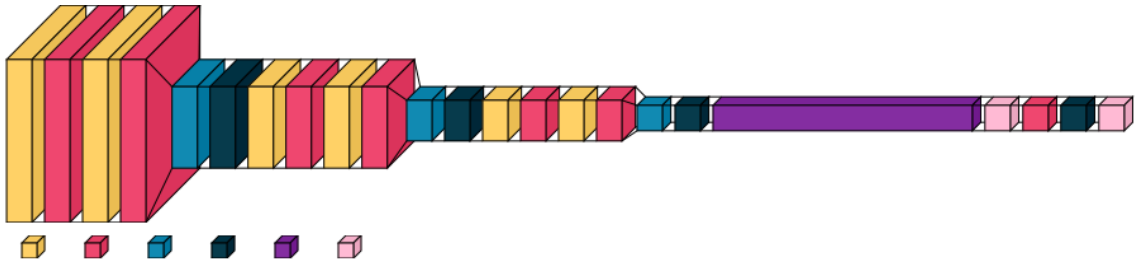


Version 3: Deep CNN with Data Augmentation

Architecture Overview: The third version of the CNN introduced more convolutional layers and made extensive use of data augmentation to improve generalization. It also included more aggressive regularization.

- **Conv2D Layer 1**: 32 filters of size 3x3, ReLU activation.
- **BatchNormalization**: Applied after the first convolution.
- **MaxPooling2D**: 2x2 pooling.
- **Dropout**: 20% dropout.
- **Conv2D Layer 2**: 64 filters of size 3x3, ReLU activation.
- **BatchNormalization**: Applied after the second convolution.
- **MaxPooling2D**: 2x2 pooling.
- **Dropout**: 30% dropout.
- **Conv2D Layer 3**: 128 filters of size 3x3, ReLU activation.

- **BatchNormalization**: Applied after the third convolution.
- **MaxPooling2D**: 2x2 pooling.
- **Dropout**: 40% dropout before the fully connected layers.
- **Dense Layer**: 128 neurons, ReLU activation.
- **Dropout**: 50% dropout.
- **Output Layer**: Softmax layer for 10 classes.
- **Data Augmentation**: Various augmentation techniques, such as rotation and horizontal flipping, were applied to increase the variety in the training data. This helped the model generalize better to unseen data.



Performance Evaluation and Risk Function

Performance Metrics

Accuracy was the primary evaluation metric for all models. Cross-Validation was applied for model selection, and confusion matrices were generated to assess classification performance. One-vs-Rest ROC curves were plotted to evaluate multi-class classification performance.

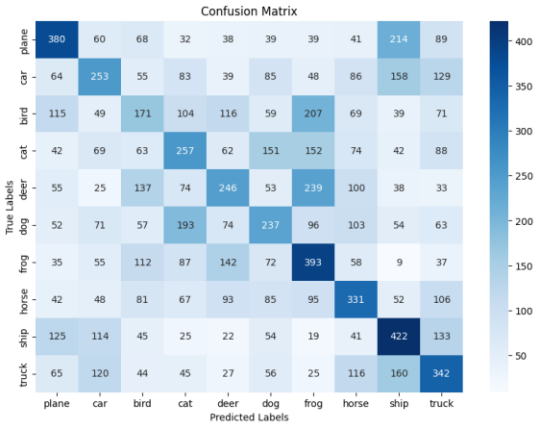
CNN Performance

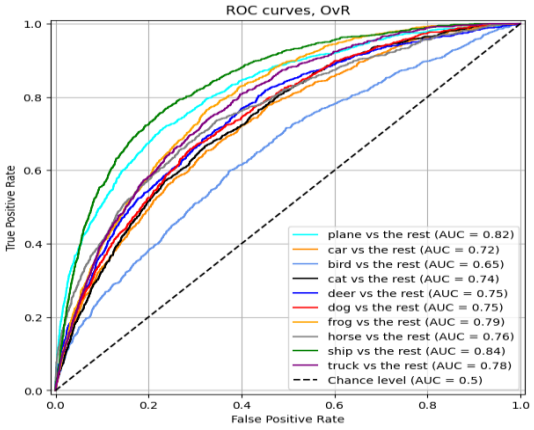
The loss function used was Categorical Cross-Entropy. After training for multiple epochs, the CNN achieved 85% accuracy, a significant improvement over traditional models.

Results and Quality of Classification

Traditional Models

Training time was quick, but performance was poor. The best performing model was **XGBoost**, with a 50% CV accuracy, but on the test set the performance was quite poor, with only 30% accuracy.

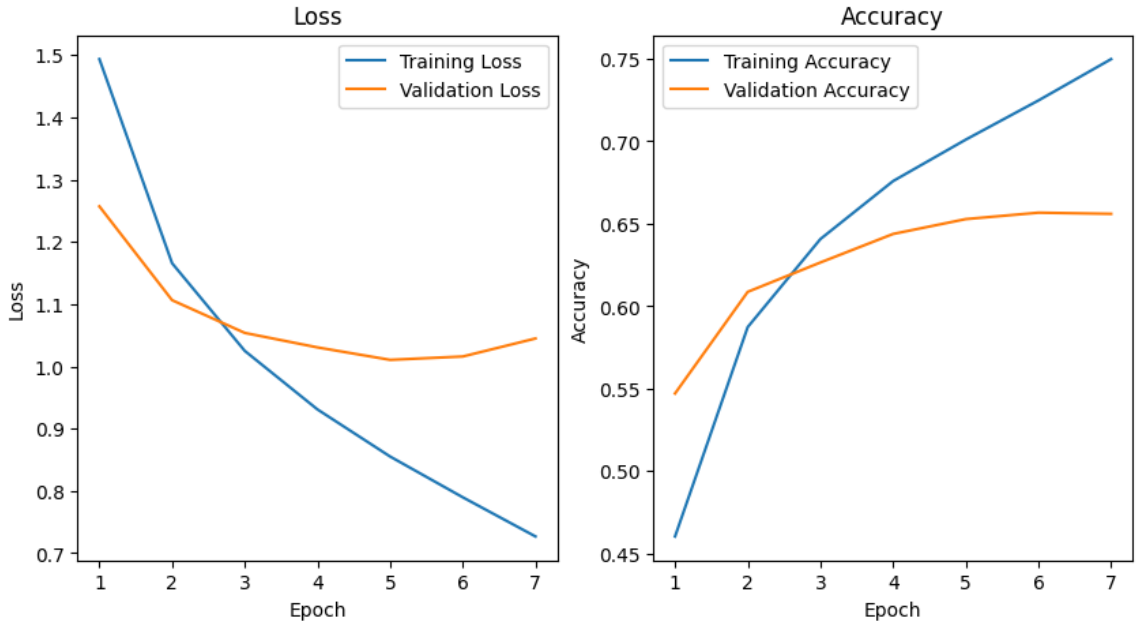


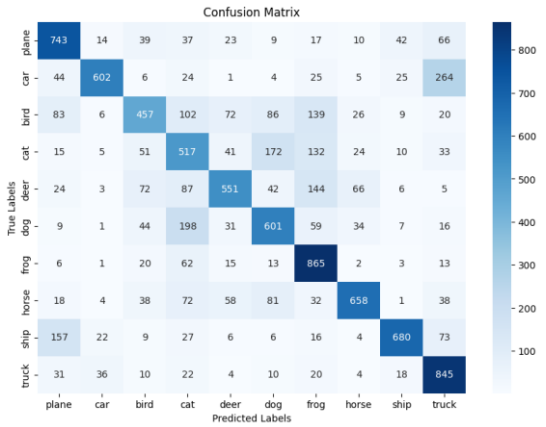


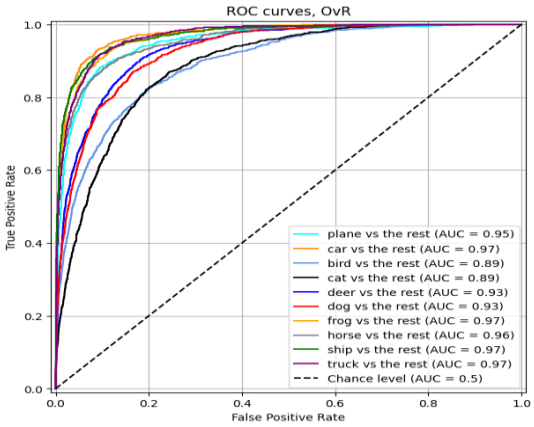
CNN Models

Version 1: Basic CNN Architecture

This basic model achieved around **65% accuracy**. It was prone to overfitting, as indicated by the divergence between training and validation accuracy after a few epochs.

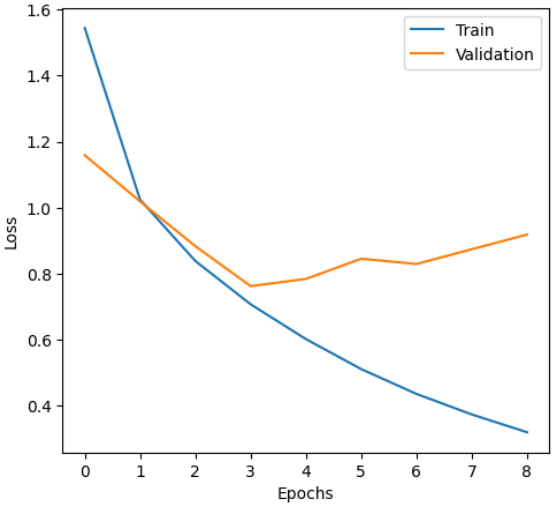


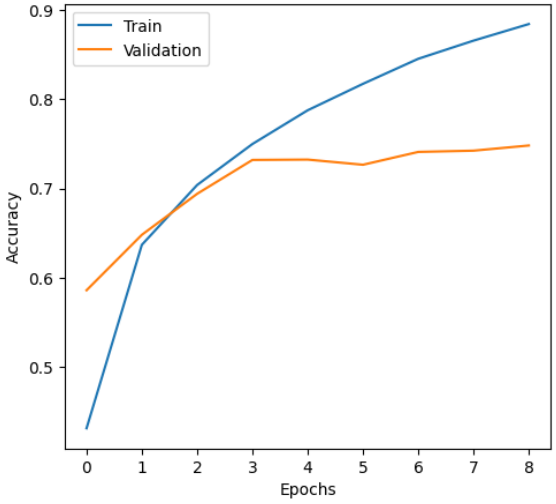


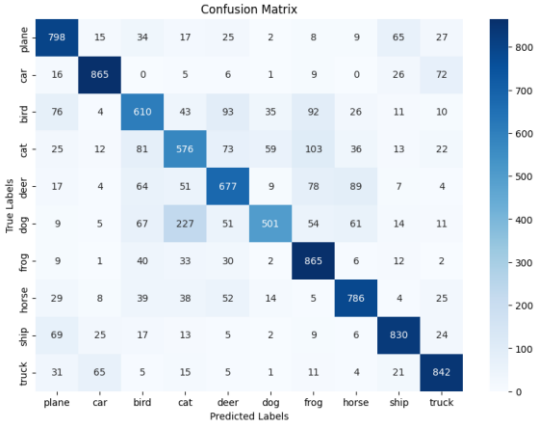


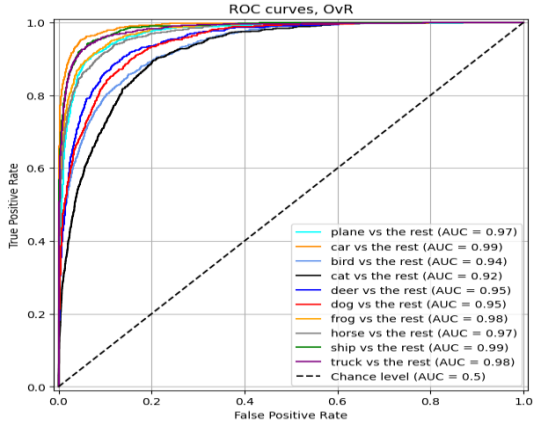
Version 2: Expanded CNN

With gradual dropout and additional convolutional layers, this model improved accuracy to around **75%**. Overfitting was better controlled, but further tuning was needed.



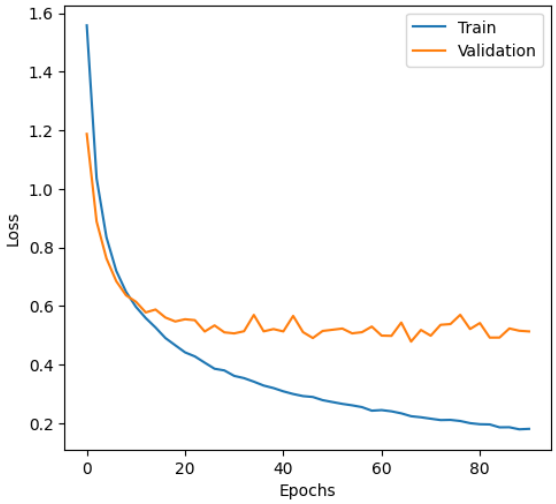


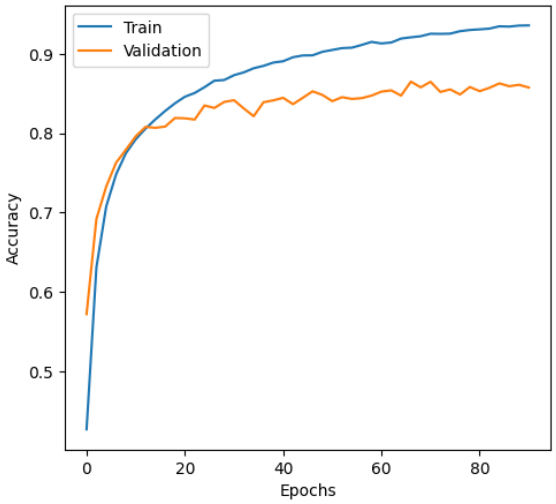


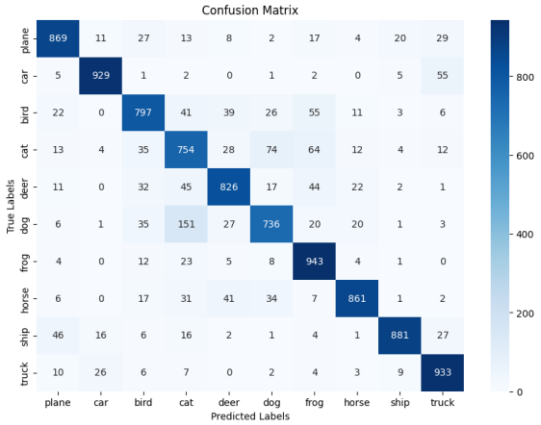


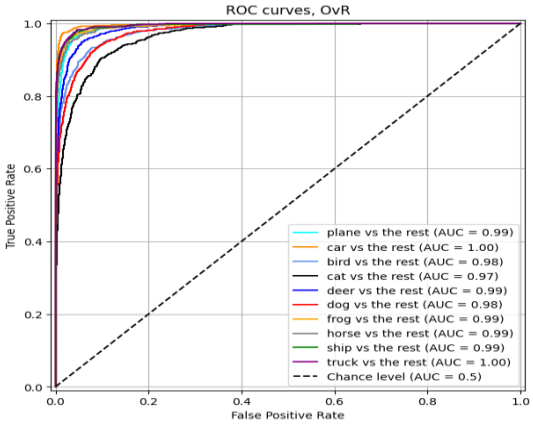
Version 3: Deep CNN with Data Augmentation

This deep CNN architecture, combined with batch normalization, data augmentation and aggressive dropout, achieved the best accuracy of **85%**. The data augmentation significantly reduced overfitting, and the deeper architecture allowed the model to learn more complex features.









Training

Traditional Models

The dataset was divided into training and testing sets. Then 5-fold cross-validation was used to determine the best performing models. We then used a randomized search with 5-fold CV to determine the best hyperparameters for each model. In addition, l_2 and l_1 regularizations were used.

CNN Models

For the CNN, 10% of the training set was used as the validation set. Hyperparameters, such as learning rate (0.001), optimizer (Adam) and filter size were set manually. In terms of regularization, gradual dropout was applied to reduce overfitting. Data Augmentation helped reduce overfitting by exposing the model to a variety of images. Early stopping was also employed to halt training when the validation loss plateaued.

Final Remarks

As expected, the CNN model proved to be the most effective for the CIFAR-10 dataset, achieving an accuracy of 85%. Techniques like dropout, data augmentation, and early stopping were key in improving generalization and preventing overfitting.