# Computer Vision Mini Project Report

Jasser Abdelfattah
UH ID: 21033101

## Problem Statement and Dataset Description

Object recognition is a critical computer vision task that involves identifying objects within images. The CIFAR-10 dataset serves as a benchmark for this mini-project, consisting of 60,000 $32 \times 32$ color images distributed equally across ten classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each class contains 6,000 images, with 50,000 allocated for training and 10,000 for testing.

The primary objective of this project is to develop and evaluate an object recognition model capable of accurately classifying images into these ten categories. Such a system could have applications in educational tools for nursery students to learn object categories.

## Approaches Used

### 0.1 Loading and Preprocessing the CIFAR-10 Dataset

The CIFAR-10 dataset is loaded and preprocessed to prepare it for training and evaluation. The dataset is divided into training, validation, and test sets. To enhance the model's generalization ability, data augmentation techniques such as random horizontal flipping and random cropping are applied to the training images. Subsequently, the images are converted into tensors and normalized.

### 0.2 Loading the Pre-trained EfficientNet-b0 Model

A pre-trained EfficientNet-b0 model is loaded and fine-tuned for the CIFAR-10 dataset. The final classification layer is replaced to accommodate the 10 classes in CIFAR-10. The model is then transferred to the appropriate device (GPU or CPU) for further training.

### 0.3 Training the EfficientNet-b0 Model

The model is trained on the training dataset, utilizing mixed precision training for computational efficiency. The training loop consists of forward passes, loss computation, backpropagation, and weight updates. To prevent overfitting, early stopping is implemented, which monitors validation loss and halts training when no improvement is observed.

### 0.4 Evaluating the EfficientNet-b0 Model

After training, the model is evaluated on the test dataset. Performance metrics, including accuracy, classification report, and confusion matrix, are computed and presented. Additionally, the model's ability to generalize to unseen images is demonstrated.

### 0.5 Prediction on Unseen Images

The trained model is used to predict the classes of unseen images. The images undergo preprocessing, and the predictions are displayed alongside the corresponding images.

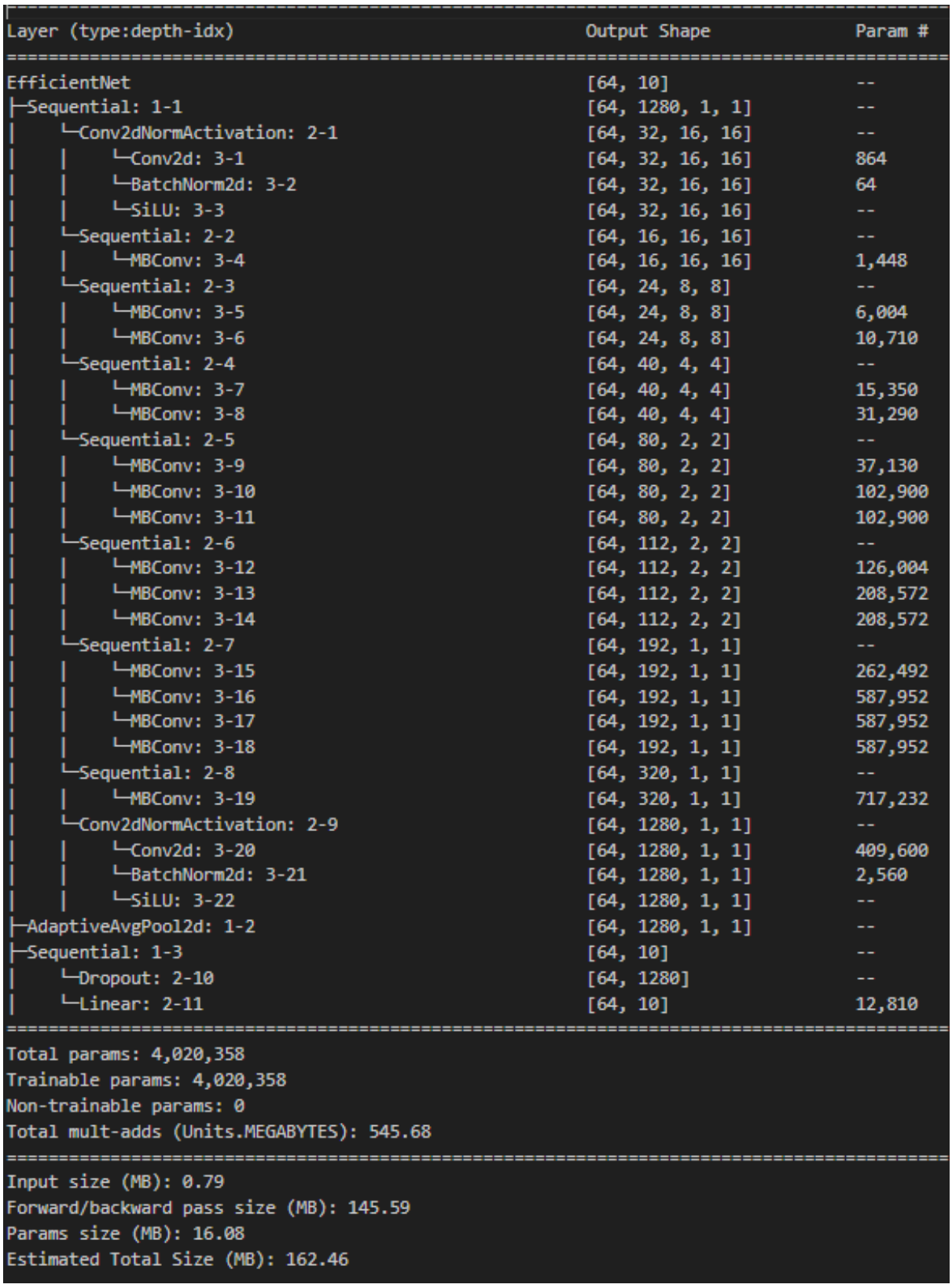
## Discussion of Results

The training and validation loss graph in Figure 2 demonstrates a consistent decrease, indicating that the model's loss is stable throughout the training process. Similarly, the training and validation accuracy graph in Figure 3 shows an upward trend, which is a positive sign, reflecting the stability and effectiveness of the model's learning.

The overall accuracy of the model on the test dataset (10,000 images) is 86 percent, as depicted in Figure 5. The classification report in Figure 4 highlights an average precision of 86 percent, recall of 86 percent, and F1-score of 86 percent, further emphasizing the balanced performance of the model across all classes.

The confusion matrix reveals closely related and comparable values between the classes, suggesting that the model has learned to distinguish them effectively. Moreover, the model's predictions on unseen images show a high success rate, correctly identifying 9 out of 10 images. The only misclassification observed was a dog incorrectly identified as a deer.

# Figures



```
Layer (type:depth-idx)                                  Output Shape              Param #
==========================================================================================
EfficientNet                                            [64, 10]                  --
├─Sequential: 1-1                                       [64, 1280, 1, 1]          --
│    └─Conv2dNormActivation: 2-1                        [64, 32, 16, 16]          --
│    │    └─Conv2d: 3-1                                 [64, 32, 16, 16]          864
│    │    └─BatchNorm2d: 3-2                            [64, 32, 16, 16]          64
│    │    └─SiLU: 3-3                                   [64, 32, 16, 16]          --
│    └─Sequential: 2-2                                  [64, 16, 16, 16]          --
│    │    └─MBConv: 3-4                                 [64, 16, 16, 16]          1,448
│    └─Sequential: 2-3                                  [64, 24, 8, 8]            --
│    │    └─MBConv: 3-5                                 [64, 24, 8, 8]            6,004
│    │    └─MBConv: 3-6                                 [64, 24, 8, 8]            10,710
│    └─Sequential: 2-4                                  [64, 40, 4, 4]            --
│    │    └─MBConv: 3-7                                 [64, 40, 4, 4]            15,350
│    │    └─MBConv: 3-8                                 [64, 40, 4, 4]            31,290
│    └─Sequential: 2-5                                  [64, 80, 2, 2]            --
│    │    └─MBConv: 3-9                                 [64, 80, 2, 2]            37,130
│    │    └─MBConv: 3-10                                [64, 80, 2, 2]            102,900
│    │    └─MBConv: 3-11                                [64, 80, 2, 2]            102,900
│    └─Sequential: 2-6                                  [64, 112, 2, 2]           --
│    │    └─MBConv: 3-12                                [64, 112, 2, 2]           126,004
│    │    └─MBConv: 3-13                                [64, 112, 2, 2]           208,572
│    │    └─MBConv: 3-14                                [64, 112, 2, 2]           208,572
│    └─Sequential: 2-7                                  [64, 192, 1, 1]           --
│    │    └─MBConv: 3-15                                [64, 192, 1, 1]           262,492
│    │    └─MBConv: 3-16                                [64, 192, 1, 1]           587,952
│    │    └─MBConv: 3-17                                [64, 192, 1, 1]           587,952
│    │    └─MBConv: 3-18                                [64, 192, 1, 1]           587,952
│    └─Sequential: 2-8                                  [64, 320, 1, 1]           --
│    │    └─MBConv: 3-19                                [64, 320, 1, 1]           717,232
│    └─Conv2dNormActivation: 2-9                        [64, 1280, 1, 1]          --
│    │    └─Conv2d: 3-20                                [64, 1280, 1, 1]          409,600
│    │    └─BatchNorm2d: 3-21                           [64, 1280, 1, 1]          2,560
│    │    └─SiLU: 3-22                                  [64, 1280, 1, 1]          --
├─AdaptiveAvgPool2d: 1-2                                [64, 1280, 1, 1]          --
├─Sequential: 1-3                                       [64, 10]                  --
│    └─Dropout: 2-10                                    [64, 1280]                --
│    └─Linear: 2-11                                     [64, 10]                  12,810
==========================================================================================
Total params: 4,020,358
Trainable params: 4,020,358
Non-trainable params: 0
Total mult-adds (Units.MEGABYTES): 545.68
==========================================================================================
Input size (MB): 0.79
Forward/backward pass size (MB): 145.59
Params size (MB): 16.08
Estimated Total Size (MB): 162.46
```

Figure 1: EfficientNet-b0 model architecture.

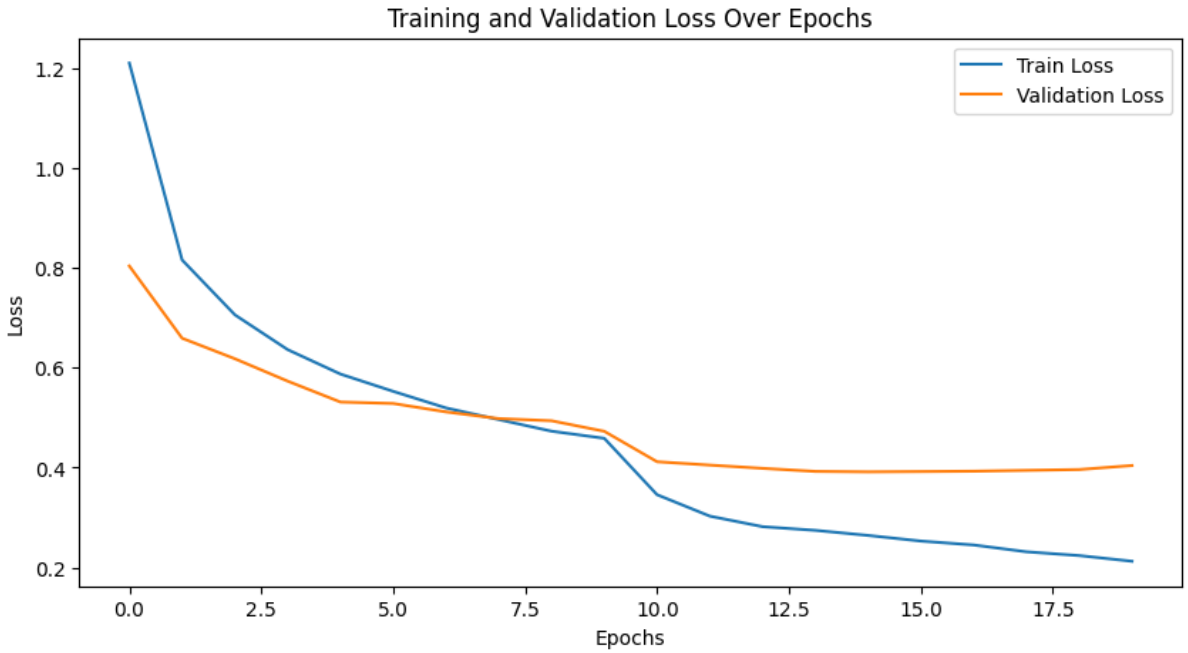



Figure 2: Training and Validation Loss Over Epochs.

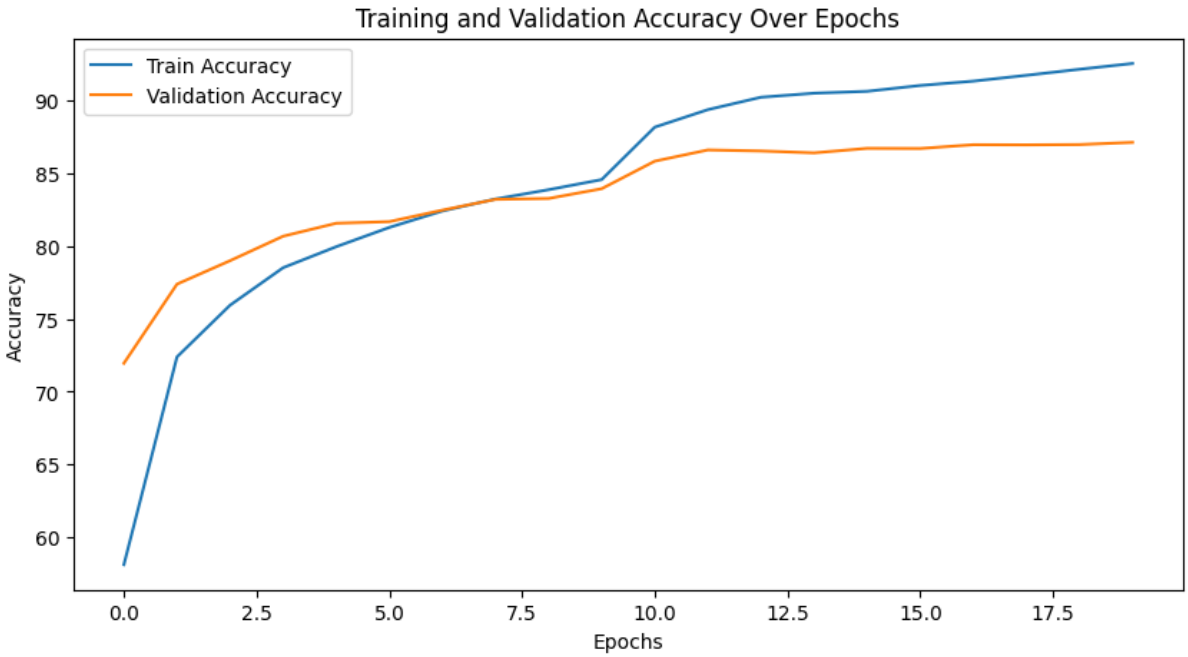



Figure 3: Training and Validation Accuracy Over Epochs.

```
Classification Report:
              precision    recall  f1-score   support

       plane       0.87      0.89      0.88      1000
  automobile       0.93      0.93      0.93      1000
        bird       0.84      0.83      0.84      1000
         cat       0.75      0.72      0.74      1000
        deer       0.85      0.87      0.86      1000
         dog       0.80      0.78      0.79      1000
        frog       0.89      0.93      0.91      1000
       horse       0.89      0.90      0.90      1000
        ship       0.93      0.93      0.93      1000
       truck       0.91      0.90      0.91      1000

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000
```

Figure 4: Classification Evaluations.

```
Accuracy of plane : 89 %
Accuracy of automobile : 92 %
Accuracy of  bird : 82 %
Accuracy of   cat : 73 %
Accuracy of  deer : 86 %
Accuracy of   dog : 78 %
Accuracy of  frog : 92 %
Accuracy of horse : 89 %
Accuracy of  ship : 92 %
Accuracy of truck : 89 %
Accuracy of the network on the 10000 test images: 86 %
```

Figure 5: Model's Accuracy on Test set (10,000 Images).

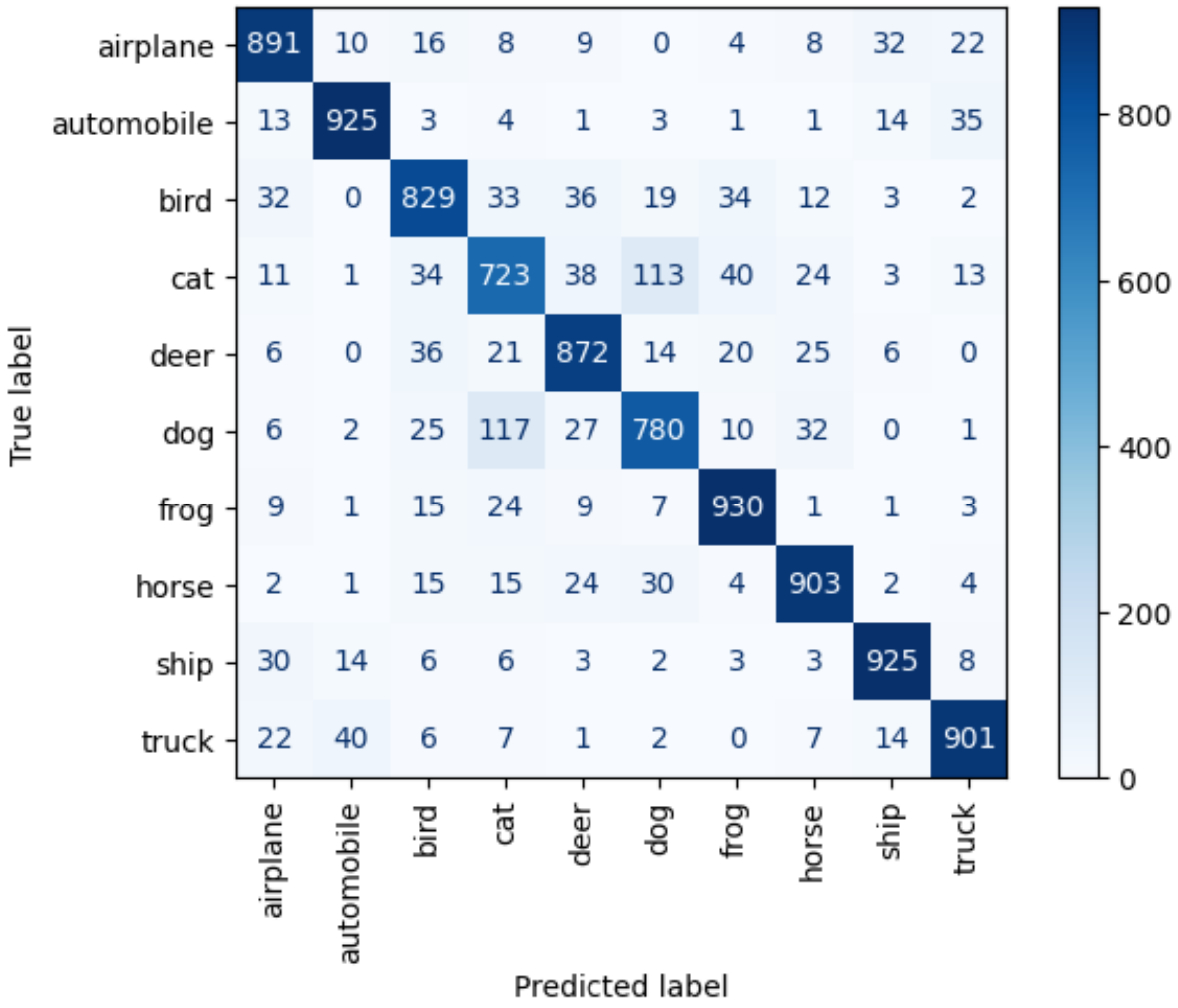



Figure 6: Confusion matrix.

| Image Label | Predicted Label |
|---|---|
| automobile | automobile |
| bird | bird |
| cat | cat |
| deer | deer |
| dog | deer |
| frog | frog |
| horse | horse |
| plane | plane |
| ship | ship |
| truck | truck |

Figure 7: Table of Prediction on Unseen Images.