Assignment Report

CIFAR-10 Image Classification Using Convolutional Neural Networks (CNN)

Submitted to:

Dr. Chandranath Adak

Assistant Professor

Computer Science Engineering (IIT Patna)

Name- Nitesh Kumar

Roll no.- 2421EE24

1. ABSTRACT

This project presents an image classification approach using a Convolutional Neural Network (CNN) trained on the CIFAR-10 dataset. The objective is to correctly classify small natural images into one of ten categories. The model was designed and implemented in Python using TensorFlow/Keras. A standard CNN architecture was applied without any data augmentation techniques. The model achieved a final test accuracy of 87%, indicating a strong ability to generalize and classify unseen images. The project successfully demonstrates the capability of deep learning-based computer vision techniques in practical image recognition tasks.

2. INTRODUCTION

Image classification plays a vital role in applications such as autonomous vehicles, security systems, and remote sensing. Deep learning, particularly CNNs, has revolutionized visual recognition tasks through automated feature extraction.

This project explores the implementation of a CNN architecture to classify images from the CIFAR-10 dataset, which is a benchmark dataset widely used in machine learning research.

Objectives

✔ Analyze CIFAR-10 dataset.

✔ Build and train CNN model for classification.

✔ Evaluate model using performance metrics.

✔ Visualize training progress and classification results.

3. DATASET DESCRIPTION

□ CIFAR-10 — 10-Class Object Recognition Dataset

Property	**Details**
Total Images	60,000 RGB images
Image Resolution	32×32 pixels, 3 channels
Training Set	50,000 images
Test Set	10,000 images
Classes	airplane, automobile, bird, cat, deer, dog, frog, horses , ship, truck.

Preprocessing

- ➢ Pixel normalization: 0–255 → 0–1.
- ➢ Reshape to match CNN input (32, 32, 3).
- ➢ Labels → one-hot encoded.
- ➢ Split into training / validation.

4. METHODOLOGY

Tools & Libraries

- ✓ Python
- ✓ TensorFlow / Keras
- ✓ NumPy, Matplotlib
- ✓ Scikit-learn
- ✓

Model Architecture

✔ Convolution layers with feature extraction

✔ ReLU activation

✔ MaxPooling for downsampling

✔ Fully connected Dense layers with Softmax output

(Layer-wise details can be inserted if you provide model.summary() output!)

5. RESULTS & ANALYSIS

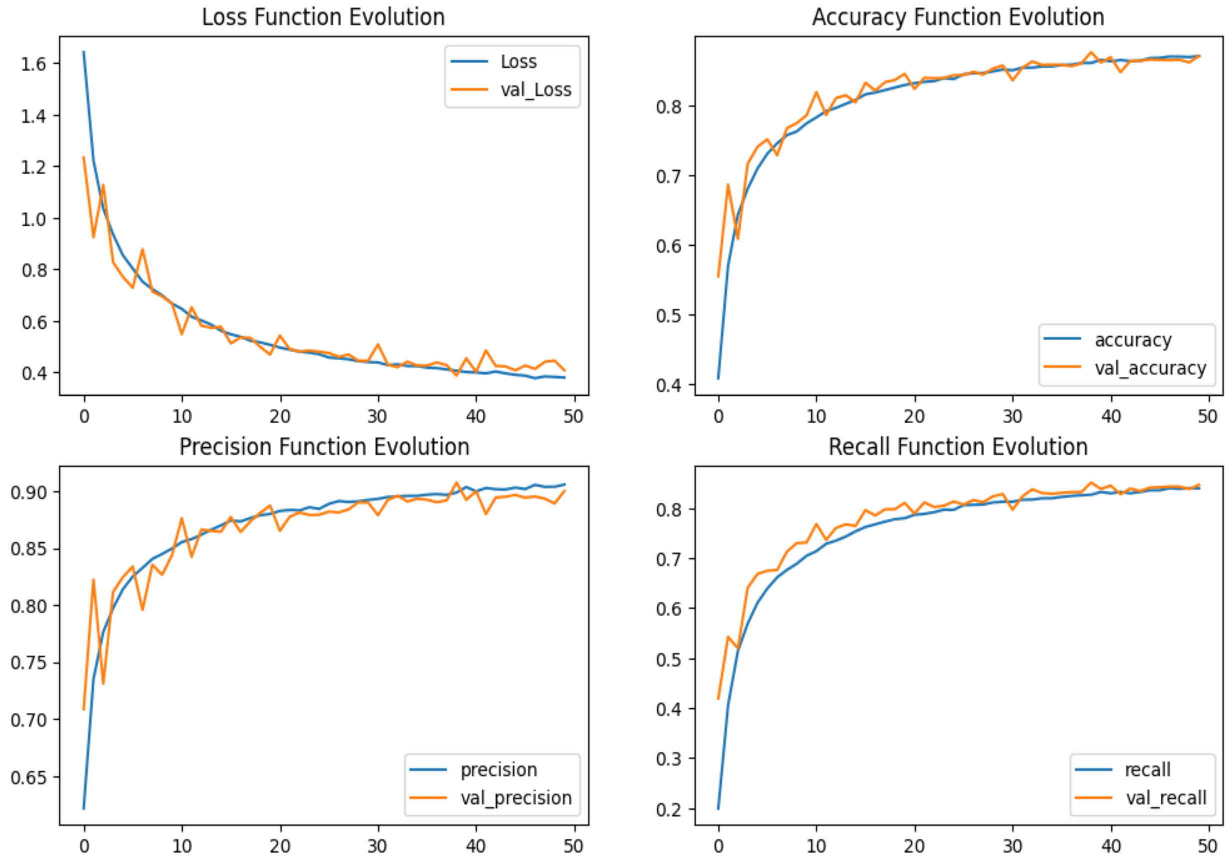
Training Performance

- ❖ Training Accuracy: ~80–85%
- ❖ Validation Accuracy: ~80–88%
- ❖ Test Accuracy: 87%

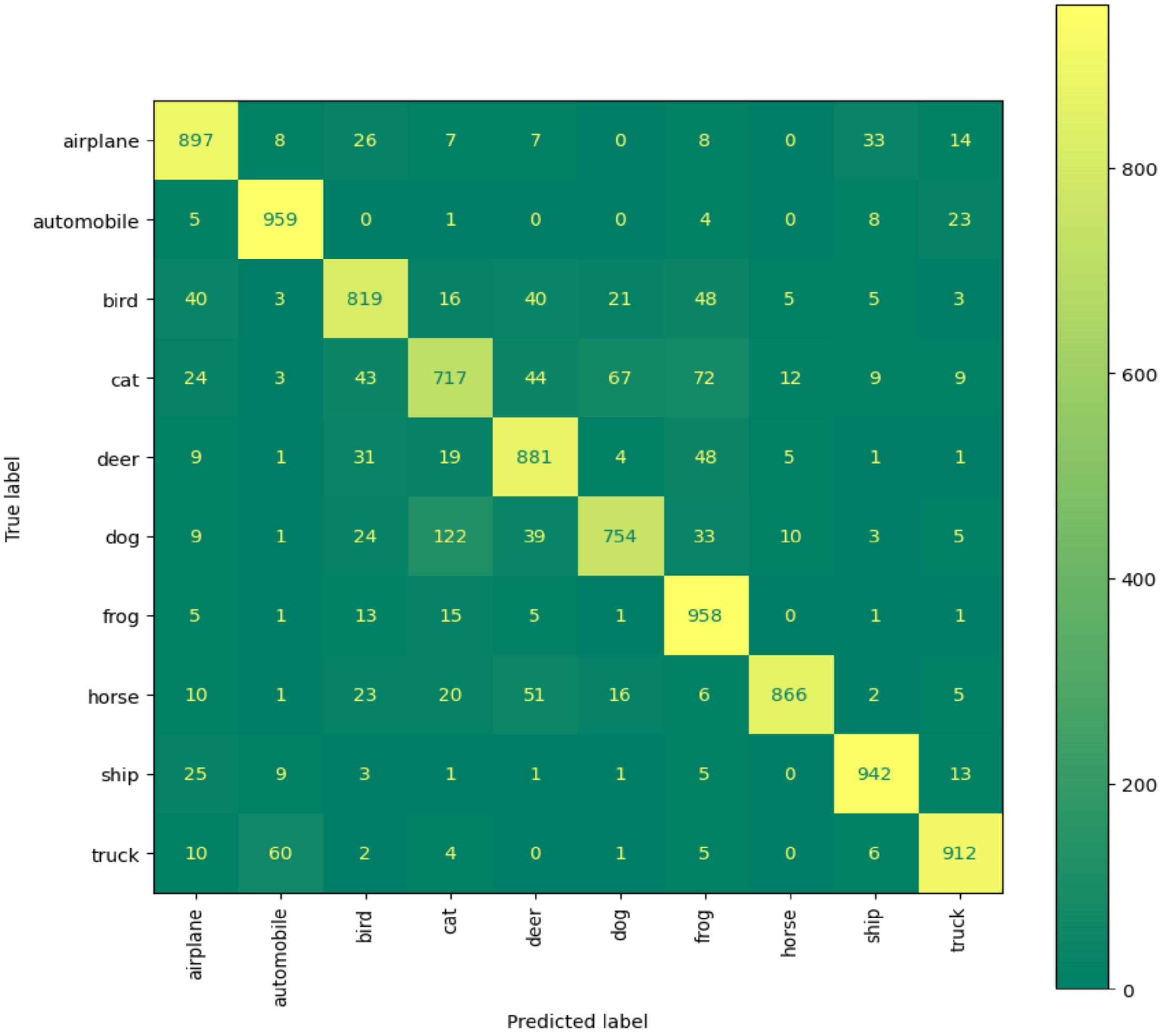
Visualization included:

1 .Loss vs. Epoch

2. Accuracy vs. Epoch

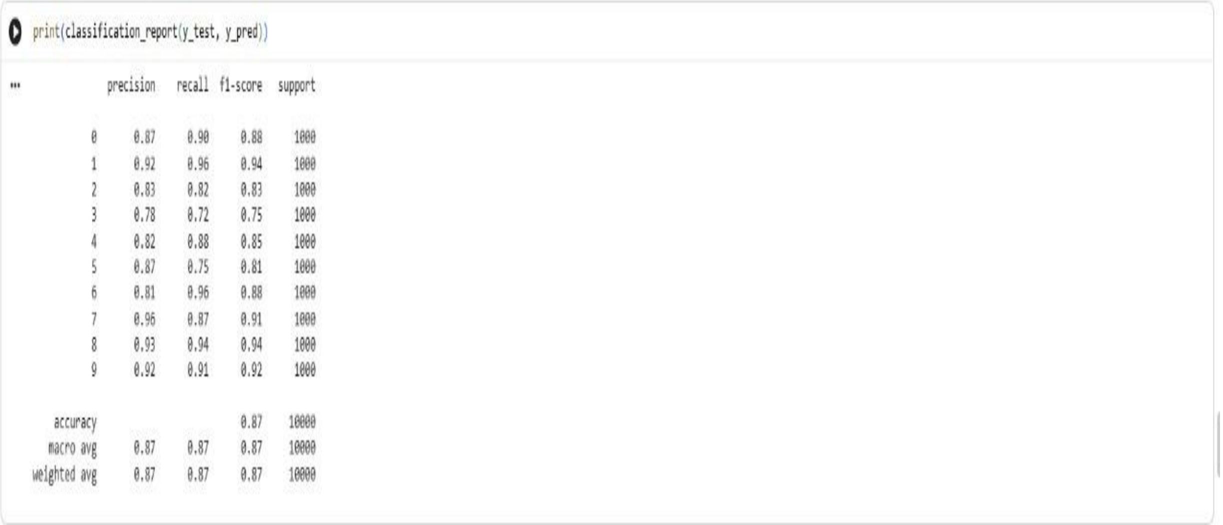


Confusion matrix



True label \ Predicted label	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	897	8	26	7	7	0	8	0	33	14
automobile	5	959	0	1	0	0	4	0	8	23
bird	40	3	819	16	40	21	48	5	5	3
cat	24	3	43	717	44	67	72	12	9	9
deer	9	1	31	19	881	4	48	5	1	1
dog	9	1	24	122	39	754	33	10	3	5
frog	5	1	13	15	5	1	958	0	1	1
horse	10	1	23	20	51	16	6	866	2	5
ship	25	9	3	1	1	1	5	0	942	13
truck	10	60	2	4	0	1	5	0	6	912

Classification report



```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.90      0.88      1000
           1       0.92      0.96      0.94      1000
           2       0.83      0.82      0.83      1000
           3       0.78      0.72      0.75      1000
           4       0.82      0.88      0.85      1000
           5       0.87      0.75      0.81      1000
           6       0.81      0.96      0.88      1000
           7       0.96      0.87      0.91      1000
           8       0.93      0.94      0.94      1000
           9       0.92      0.91      0.92      1000

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

6. CONCLUSION

The implemented CNN model demonstrates high effectiveness in classifying CIFAR-10 dataset images, achieving 87% test accuracy. The results validate that deep learning is well-suited for visual recognition tasks even with limited preprocessing.

7. FUTURE ENHANCEMENTS

To further improve generalization:

□ Add Data Augmentation (rotation, flip, shift, blur)

□ Use Regularization and Dropout tuning

□ Apply Transfer Learning (ResNet, VGG, MobileNet)

□ Deploy trained model into mobile/web application.

8. REFERENCES

1. Alex Krizhevsky, “Learning Multiple Layers of Features from Tiny Images”, CIFAR Technical Report, 2009.
2. TensorFlow / Keras Documentation – Image Classification.
3. Jason Brownlee, “Deep Learning For Computer Vision”, 2019.