# Project Title:

Image Classification using Convolutional Neural Networks (CNN) in TensorFlow

# Objective:

To build and train a deep learning model using TensorFlow to classify images from the CIFAR-10 dataset into one of the 10 predefined classes. The goal is to demonstrate understanding of CNN architecture, training methodology, and evaluation techniques in computer vision tasks.

# Tools and Libraries Used:

- Python  
- TensorFlow  
- NumPy  
- Matplotlib  
- CIFAR-10 dataset

# Dataset Information:

CIFAR-10 is a labeled subset of the 80 million tiny images dataset. It consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images.

# Code and Implementation:

import tensorflow as tf  
from tensorflow.keras import layers, models  
import matplotlib.pyplot as plt  
import numpy as np  
  
(x\_train, y\_train), (x\_test, y\_test) = tf.keras.datasets.cifar10.load\_data()  
class\_names = ['Airplane', 'Car', 'Bird', 'Cat', 'Deer',  
 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']  
x\_train, x\_test = x\_train / 255.0, x\_test / 255.0  
  
model = models.Sequential([  
 layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)),  
 layers.MaxPooling2D((2, 2)),  
 layers.Conv2D(64, (3, 3), activation='relu'),  
 layers.MaxPooling2D((2, 2)),  
 layers.Conv2D(64, (3, 3), activation='relu'),  
 layers.Flatten(),  
 layers.Dense(64, activation='relu'),  
 layers.Dense(10)  
])  
  
model.compile(optimizer='adam',  
 loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True),  
 metrics=['accuracy'])  
  
history = model.fit(x\_train, y\_train, epochs=10, validation\_data=(x\_test, y\_test))  
test\_loss, test\_acc = model.evaluate(x\_test, y\_test, verbose=2)  
print('\nTest accuracy:', test\_acc)

# Model Architecture:

1. Conv2D -> ReLU -> MaxPooling  
2. Conv2D -> ReLU -> MaxPooling  
3. Conv2D -> ReLU  
4. Flatten -> Dense (ReLU) -> Dense (Output Layer)  
5. Loss: Sparse Categorical Crossentropy  
6. Optimizer: Adam

# Results:

The model achieved high training and validation accuracy. The visualization shows performance trends and example predictions.

# Conclusion:

This task successfully demonstrates the implementation of a CNN for image classification. It highlights preprocessing, model training, evaluation, and result interpretation using TensorFlow.