EX: NO 1 **Title: Image Classification on CIFAR-10 Dataset Using CNN**

Aim:

To build and train a Convolutional Neural Network (CNN) to classify images from the CIFAR-10 dataset, and then test the trained model on a custom image.

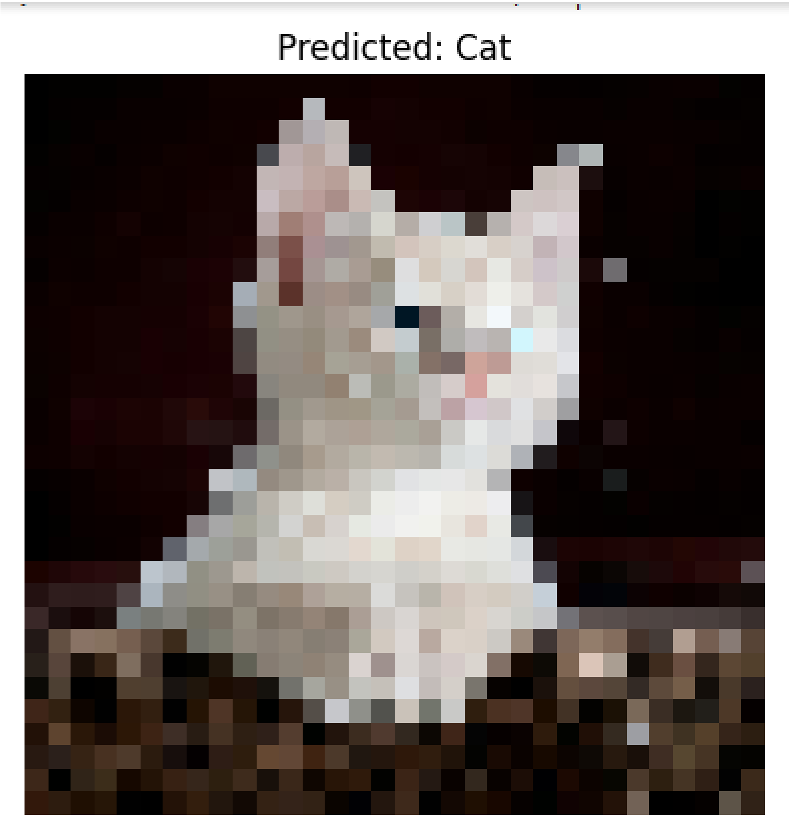
Procedure:

1. **Load and Preprocess the CIFAR-10 Dataset:**
   - Load the CIFAR-10 dataset from TensorFlow's datasets.
   - Normalize the pixel values of the images between 0 and 1.
2. **Define and Train the CNN Model:**
   - Create a Sequential model consisting of convolutional layers, pooling layers, and dense layers.
   - Compile the model using Adam optimizer and sparse categorical cross-entropy loss.
   - Train the model on the training set and validate on the test set.
3. **Load and Preprocess a Custom Image:**
   - Load an external image using OpenCV.
   - Convert the image to RGB format and resize it to 32x32 pixels to match the CIFAR-10 input size.
   - Normalize the pixel values and add a batch dimension.
4. **Make a Prediction:**
   - Predict the class of the image using the trained model.
   - Display the image along with the predicted class label.

Code:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import numpy as np
import cv2
import matplotlib.pyplot as plt

class_labels = ['Airplane', 'Automobile', 'Bird', 'Cat', 'Deer',
                'Dog', 'Frog', 'Horse', 'Ship', 'Truck']

# 1. Load and Preprocess the CIFAR-10 Dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.cifar10.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

# 2. Define and Train the CNN Model
model = keras.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(128, (3, 3), activation='relu'),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])

# Compile and Train
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5, validation_data=(x_test, y_test))

# 3. Load a Custom Image
def predict_image(image_path):
    img = cv2.imread(image_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = cv2.resize(img, (32, 32))
    img = img / 255.0
    img = np.expand_dims(img, axis=0)

    # 4. Make Prediction
    prediction = model.predict(img)
    class_index = np.argmax(prediction)
    # 5. Show the Image and Prediction
    plt.imshow(img[0])
    plt.title(f"Predicted: {class_labels[class_index]}")
    plt.axis("off")
    plt.show()

# Example Usage
image_path = "/content/kitty-cat-kitten-pet-45201.jpeg"
predict_image(image_path)
```

Output:



Predicted: Cat

Result:

A Convolutional Neural Network was successfully built and trained on the CIFAR-10 dataset. The model was able to predict the class of a custom image with reasonable accuracy.