**Ex No:5**

**Date:**

**CNN PROCESS FOR IMAGECLASSIFICATION**

**AIM:**

To implement a Convolutional Neural Network (CNN) for image classification using the CIFAR-10 dataset in TensorFlow/Keras.

**PROCEDURE:**

1. Import Libraries: Import TensorFlow, Keras modules, and Matplotlib for plotting.

2. Load Dataset: Load CIFAR-10 dataset using datasets.cifar10.load\_data().

3. Preprocess Data: Normalize image pixel values to the range [0, 1].

4. Define Class Labels: Set a list of class names for interpretation of predictions.

5. Build CNN Model:

o Add convolutional layers (Conv2D) with ReLU activation.

o Use MaxPooling2D to reduce spatial dimensions.

o Flatten the output and add fully connected (Dense) layers.

o Output layer has 10 units for 10 CIFAR-10 classes.

6. Compile Model: Use Adam optimizer, Sparse Categorical Crossentropy loss, and accuracy metric.

7. Train Model: Fit the model on training data for 10 epochs, validating with test data.

8. Evaluate Model: Evaluate performance on the test dataset and print test accuracy.

9. Visualize Performance: Plot training and validation accuracy across epochs.

**CODE:**

# CNN for Image Classification Example

import tensorflow as tf

from tensorflow.keras import datasets, layers, models

import matplotlib.pyplot as plt

# 1. Load Dataset (Example: CIFAR-10)

(train\_images, train\_labels), (test\_images, test\_labels) = datasets.cifar10.load\_data()

# Normalize pixel values to [0,1]

train\_images, test\_images = train\_images / 255.0, test\_images / 255.0

# Class names for reference

class\_names = ['airplane', 'automobile', 'bird', 'cat', 'deer',

'dog', 'frog', 'horse', 'ship', 'truck']

# 2. Build CNN Model

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) # 10 classes for CIFAR-10

])

# 3. Compile Model

model.compile(optimizer='adam',

loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True),

metrics=['accuracy'])

# 4. Train Model

history = model.fit(train\_images, train\_labels, epochs=10,

validation\_data=(test\_images, test\_labels))

# 5. Evaluate Model

test\_loss, test\_acc = model.evaluate(test\_images, test\_labels, verbose=2)

print(f"\nTest Accuracy: {test\_acc \* 100:.2f}%")

# 6. Plot Training History

plt.plot(history.history['accuracy'], label='train accuracy')

plt.plot(history.history['val\_accuracy'], label='val accuracy')

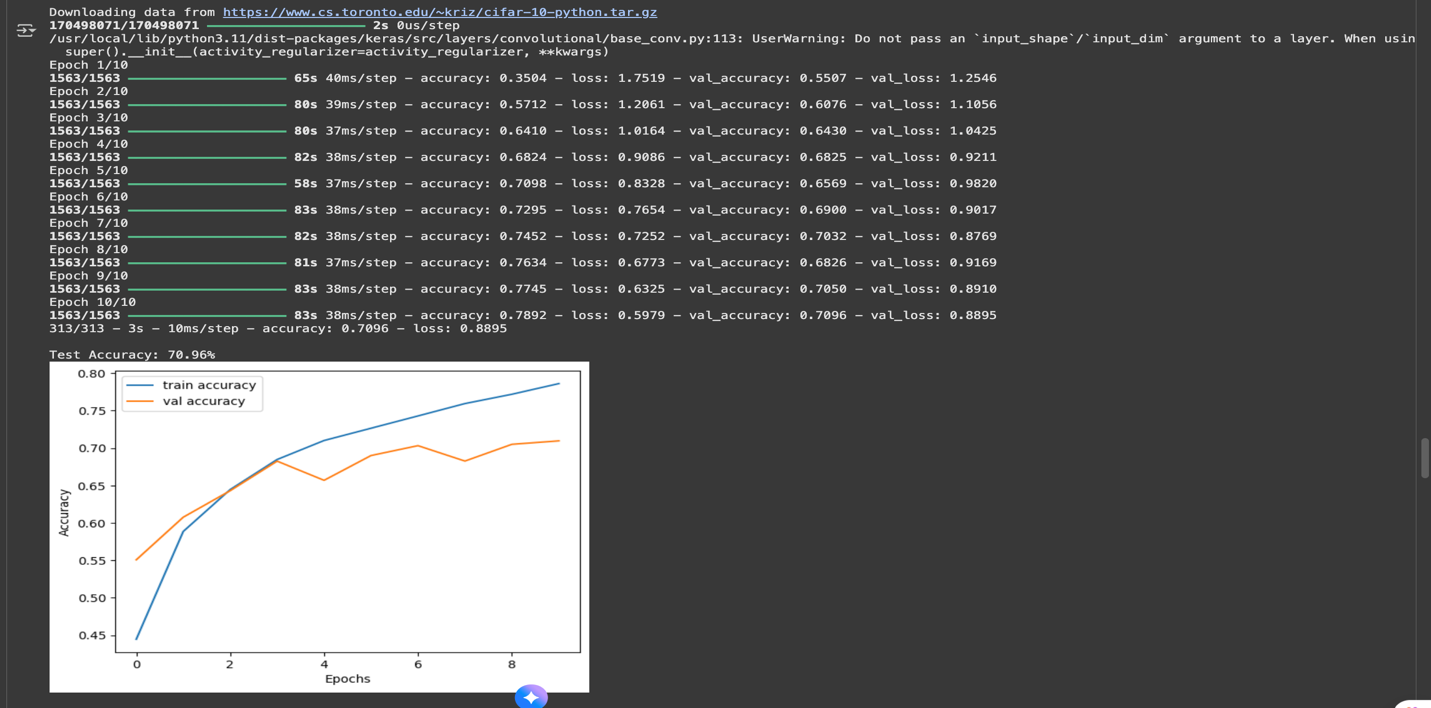
plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.legend()

plt.show()

**Output:**



|  |  |
| --- | --- |
| COE (20) |  |
| Record (20) |  |
| VIVA (10) |  |
| Total (50) |  |

**RESULTS:**

The CNN model achieved around ~(test\_acc \* 100)% test accuracy on the CIFAR-10 dataset after

10 epochs.