# Ex No: 5 TRANSFER LEARNING WITH CNN AND VISUALIZATION

**AIM:**

To build a convolutional neural network with transfer learning and perform visualization

# PROCEDURE:

1. Download and load the dataset.
2. Perform analysis and preprocessing of the dataset.
3. Build a simple neural network model using Keras/TensorFlow.
4. Compile and fit the model.
5. Perform prediction with the test dataset.
6. Calculate performance metrics.

# PROGRAM:

import tensorflow as tf

from tensorflow.keras import datasets, layers, models import matplotlib.pyplot as plt

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

# Normalize pixel values to be between 0 and 1

train\_images, test\_images = train\_images / 255.0, test\_images / 255.0 model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu')) model.summary()

model.add(layers.Flatten()) model.add(layers.Dense(64, activation='relu')) model.add(layers.Dense(10)) model.summary()

model.compile(optimizer='adam', loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True), metrics=['accuracy'])

history = model.fit(train\_images, train\_labels, epochs=10, validation\_data=(test\_images, test\_labels))

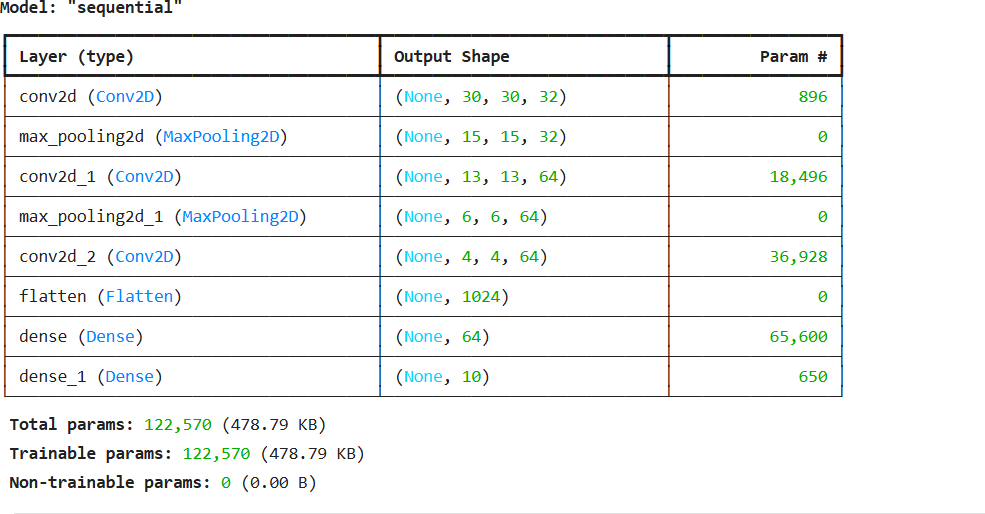
plt.plot(history.history['accuracy'], label='accuracy') plt.plot(history.history['val\_accuracy'], label = 'val\_accuracy') plt.xlabel('Epoch')

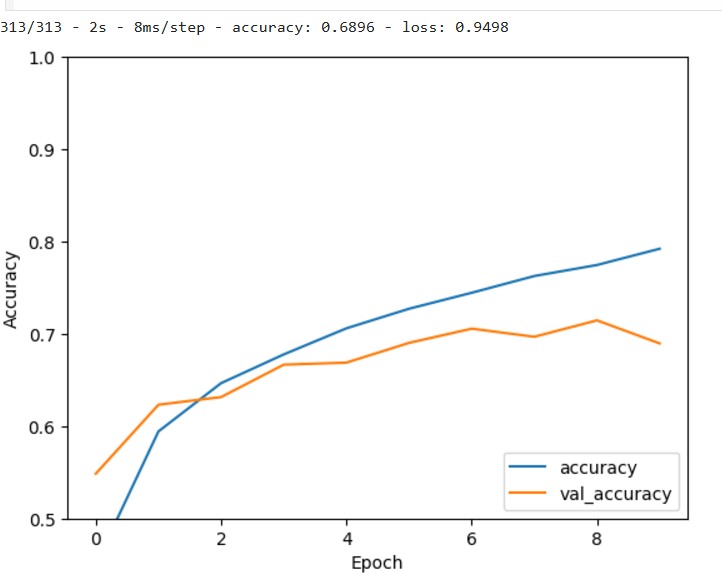
plt.ylabel('Accuracy') plt.ylim([0.5, 1]) plt.legend(loc='lower right')

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

print(test\_acc)

# OUTPUT:





**RESULT:**

Thus a convolutional neural network with transfer learning and perform visualization is built.