**Batch: HO DL 1 Experiment Number: 4**

**Roll Number: 16010421073 Name: Keyur Patel**

**Title of the Experiment: Transfer Learning with CNN**

**Program:**

import tensorflow as tf

from tensorflow.keras.datasets import cifar10

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

from tensorflow.keras.utils import to\_categorical

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report

import matplotlib.pyplot as plt

import numpy as np

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

num\_classes = 10

class\_labels = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

class\_images = {label: [] for label in class\_labels}

for i in range(num\_classes):

    class\_images[class\_labels[i]] = train\_images[np.where(train\_labels == i)[0]]

plt.figure(figsize=(12, 8))

for i in range(num\_classes):

    plt.subplot(2, 5, i + 1)

    random\_image\_index = np.random.randint(0, len(class\_images[class\_labels[i]]))

    plt.imshow(class\_images[class\_labels[i]][random\_image\_index])

    plt.title(class\_labels[i])

    plt.axis('off')

plt.show()

model = Sequential([

    Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)),

    MaxPooling2D((2, 2)),

    Conv2D(64, (3, 3), activation='relu'),

    MaxPooling2D((2, 2)),

    Flatten(),

    Dense(128, activation='relu'),

    Dropout(0.5),

    Dense(num\_classes, activation='softmax')

])

model.summary()

tf.keras.utils.plot\_model(model, to\_file='model.png', show\_shapes=True, show\_layer\_names=True)

train\_images = train\_images / 255.0  # Normalize pixel values to [0, 1]

train\_labels = to\_categorical(train\_labels, num\_classes)

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

history = model.fit(train\_images, train\_labels, epochs=10, batch\_size=64, validation\_split=0.2)

train\_accuracy = history.history['accuracy'][-1]

val\_accuracy = history.history['val\_accuracy'][-1]

print(f"Training Accuracy: {train\_accuracy}, Validation Accuracy: {val\_accuracy}")

model\_regularized = Sequential([

    Conv2D(32, (3, 3), activation='relu', input\_shape=(32, 32, 3)),

    MaxPooling2D((2, 2)),

    Conv2D(64, (3, 3), activation='relu'),

    MaxPooling2D((2, 2)),

    Flatten(),

    Dense(128, activation='relu', kernel\_regularizer=tf.keras.regularizers.l2(0.001)),

    Dropout(0.5),

    Dense(num\_classes, activation='softmax')

])

model\_regularized.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

history\_regularized = model\_regularized.fit(train\_images, train\_labels, epochs=10, batch\_size=64, validation\_split=0.2)

plt.plot(history\_regularized.history['loss'], label='Training Loss')

plt.plot(history\_regularized.history['val\_loss'], label='Validation Loss')

plt.xlabel('Epochs')

plt.ylabel('Loss')

plt.title('Loss Over Time')

plt.legend()

plt.show()

plt.plot(history\_regularized.history['accuracy'], label='Training Accuracy')

plt.plot(history\_regularized.history['val\_accuracy'], label='Validation Accuracy')

plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.title('Accuracy Over Time')

plt.legend()

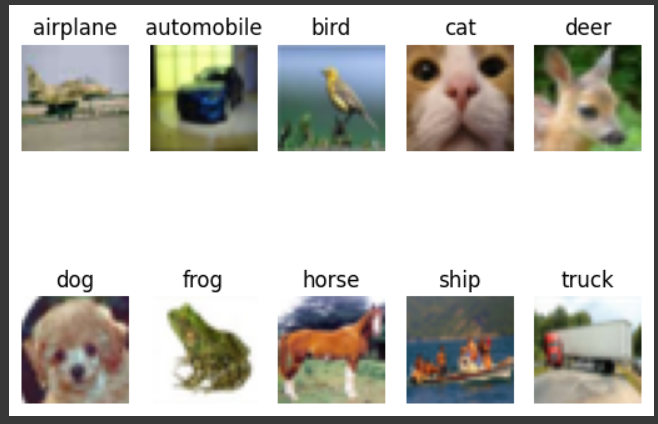
plt.show()

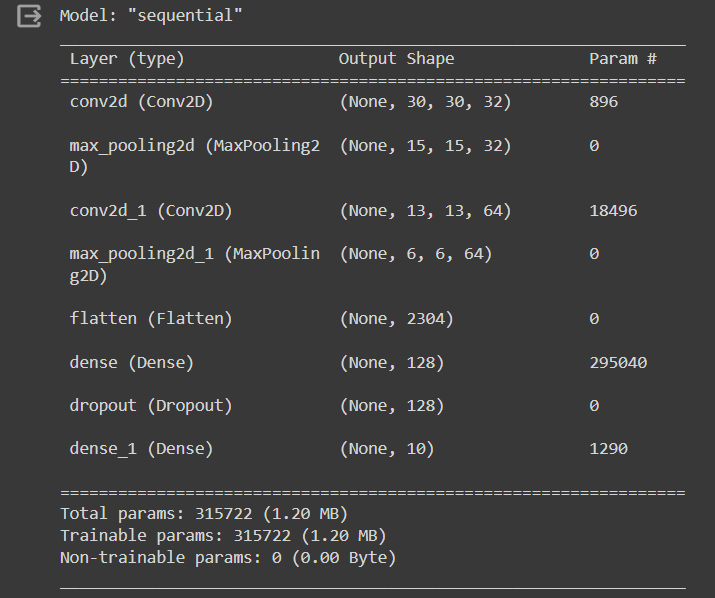
predictions = model\_regularized.predict(train\_images)

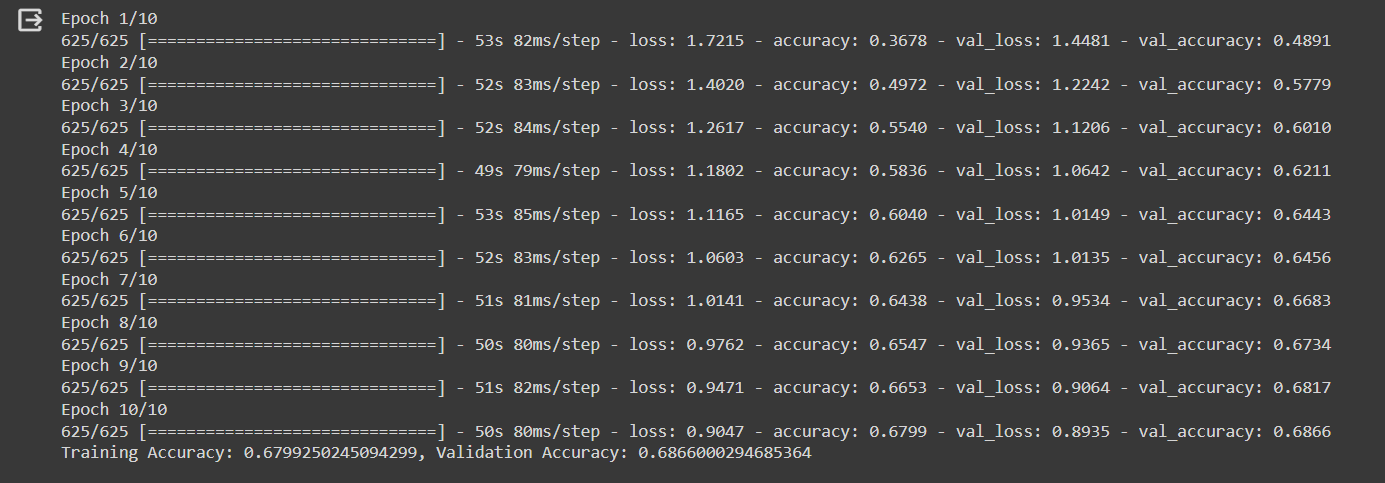
predicted\_classes = np.argmax(predictions, axis=1)

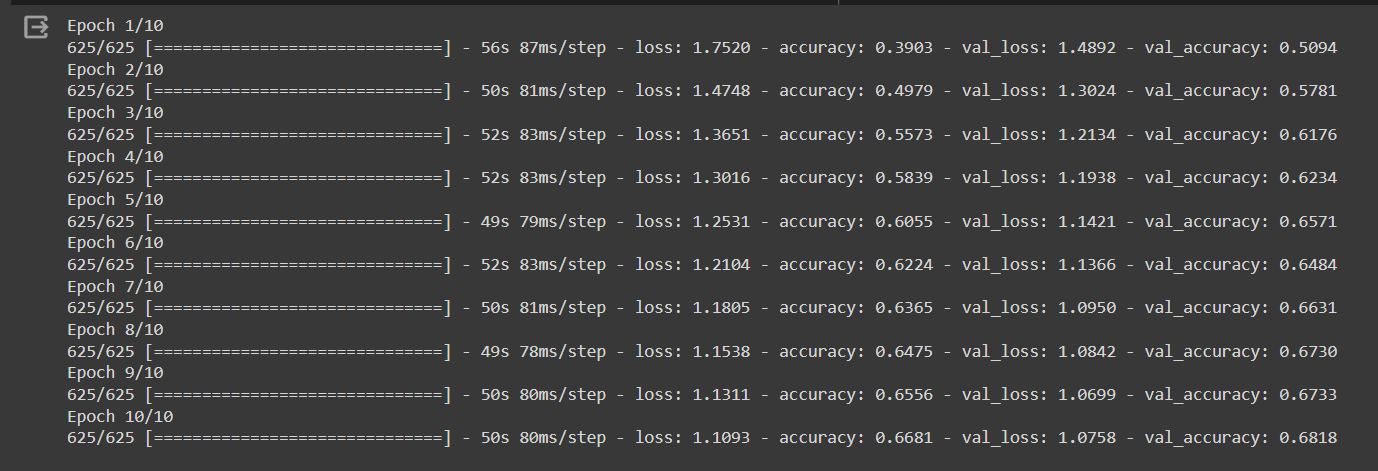
print(classification\_report(np.argmax(train\_labels, axis=1), predicted\_classes))

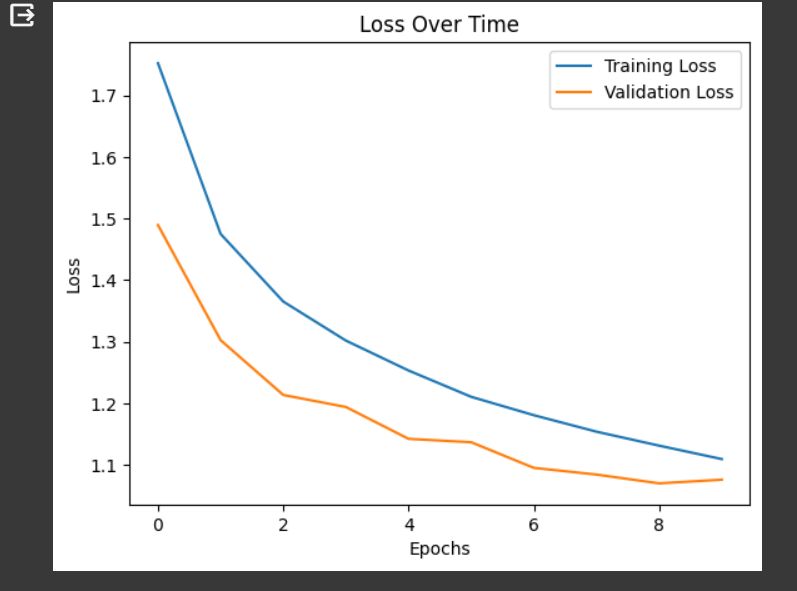
**Output:**

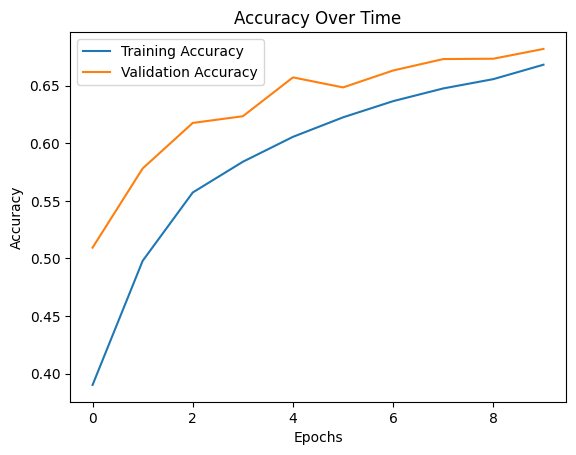
****

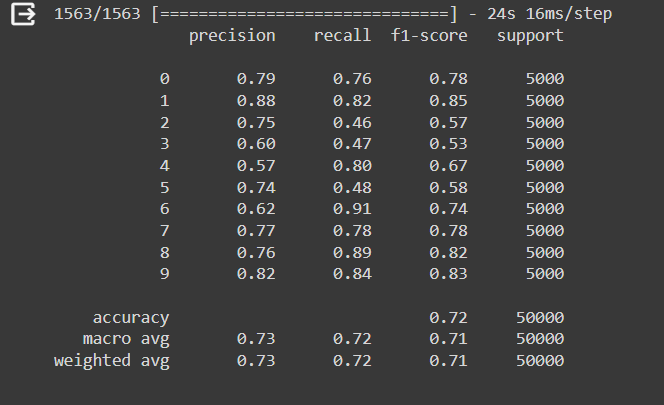
****

****

****

****



****

**CO: CO3: Assimilate fundamentals of Convolutional Neural Network.**

**Conclusion:**

In this experiment we learnt about transfer learning with CNN and performed the same. The output of the same has been displayed.