**Name: Vishnu Mohan**

**Roll No:51**

**Batch: MCA-B**

**Date: 24-11-2022**

**DATA SCIENCE LAB**

**Experiment No.: 12**

**Aim**

Implementation of CNN using keras network

**Procedure**

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()

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

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

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

plt.figure(figsize=(10,10))

for i in range(25):

plt.subplot(5,5,i+1)

plt.xticks([])

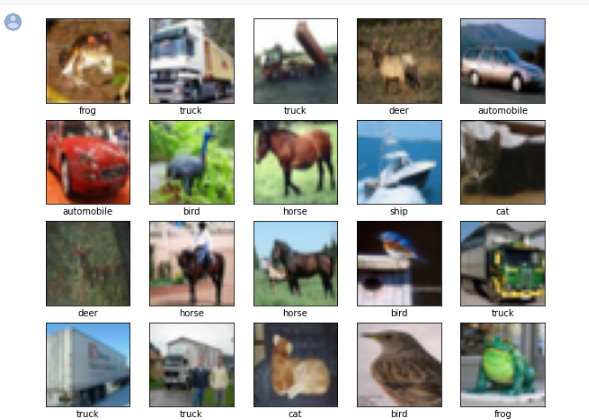
plt.yticks([])

plt.grid(False)

plt.imshow(train\_images[i])

plt.xlabel(class\_names[train\_labels[i][0]])

plt.show()



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()

Output

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 30, 30, 32) 896

max\_pooling2d (MaxPooling2D (None, 15, 15, 32) 0

)

conv2d\_1 (Conv2D) (None, 13, 13, 64) 18496

max\_pooling2d\_1 (MaxPooling (None, 6, 6, 64) 0

2D)

conv2d\_2 (Conv2D) (None, 4, 4, 64) 36928

flatten (Flatten) (None, 1024) 0

dense (Dense) (None, 64) 65600

dense\_1 (Dense) (None, 10) 650

=================================================================

Total params: 122,570

Trainable params: 122,570

Non-trainable params: 0

model.compile(optimizer='adam',

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

              metrics=['accuracy'])

history = model.fit(train\_images, train\_labels, epochs=5,

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

Output

Epoch 1/5

1563/1563 [==============================] - 16s 5ms/step - loss: 1.5253 - accuracy: 0.4442 - val\_loss: 1.2627 - val\_accuracy: 0.5531

Epoch 2/5

1563/1563 [==============================] - 8s 5ms/step - loss: 1.1625 - accuracy: 0.5867 - val\_loss: 1.1056 - val\_accuracy: 0.6121

Epoch 3/5

1563/1563 [==============================] - 8s 5ms/step - loss: 1.0065 - accuracy: 0.6467 - val\_loss: 0.9735 - val\_accuracy: 0.6567

Epoch 4/5

1563/1563 [==============================] - 7s 5ms/step - loss: 0.9101 - accuracy: 0.6816 - val\_loss: 0.9356 - val\_accuracy: 0.6720

Epoch 5/5

1563/1563 [==============================] - 7s 5ms/step - loss: 0.8382 - accuracy: 0.7062 - val\_loss: 0.9111 - val\_accuracy: 0.6862

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

print(test\_acc)

Output

0.6862000226974487

**Result**

The program was executed and the result was successfully obtained. Thus CO4 was obtained.