|  |
| --- |
| **LAB FINAL** |
| **ARTIFICIAL**  **INTELLIGENCE** |
|  |

**NAME:**

**Muhammad Ali Zain**

**REG NO:**

**SP21-BSE-065**

**SUBMITTED TO:**

**DR Ahmad Khan**

**DATED:**

**18-January 2024**

**Image Classification using Convolutional Neural Networks (CNN)**

**CODE:**

import keras  
import tensorflow as tf  
from keras import layers, models  
from keras.preprocessing.image import ImageDataGenerator  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import classification\_report, confusion\_matrix  
  
  
(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 # Normalize pixel values to [0, 1]  
  
datagen = ImageDataGenerator(  
 rotation\_range=15,  
 width\_shift\_range=0.1,  
 height\_shift\_range=0.1,  
 horizontal\_flip=True,  
 zoom\_range=0.1  
)  
  
datagen.fit(x\_train)  
  
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(128, (3, 3), activation='relu'))  
model.add(layers.Flatten())  
model.add(layers.Dense(128, activation='relu'))  
model.add(layers.Dense(10, activation='softmax'))  
  
model.compile(optimizer='adam',  
 loss='sparse\_categorical\_crossentropy',  
 metrics=['accuracy'])  
  
x\_train, x\_val, y\_train, y\_val = train\_test\_split(x\_train, y\_train, test\_size=0.2, random\_state=42)  
  
history = model.fit(datagen.flow(x\_train, y\_train, batch\_size=64),  
 epochs=5,  
 validation\_data=(x\_val, y\_val))  
  
test\_loss, test\_acc = model.evaluate(x\_test, y\_test)  
print(f'Test Accuracy: {test\_acc}')  
y\_pred = model.predict(x\_test)  
y\_pred\_classes = tf.argmax(y\_pred, axis=1)  
print("Classification Report:\n", classification\_report(y\_test, y\_pred\_classes))  
print("Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred\_classes))  
  
model.save('image\_classification\_model')  
  
loaded\_model = keras.models.load\_model('image\_classification\_model')

**OUTPUT:**

****

A screenshot of a computer

Description automatically generated