ADHIPARASAKTHI ENGINEERING COLLEGE

MINI PROJECT

ON

Image Classification And Regression On CIFAR10 Dataset

SUBMITTED

BY

NAME	**SWETHA R**
REG NO	**420421104082**
YEAR	**III**
DEGREE	**BE-CSE**

Image Classification And Regression On CIFAR10 Dataset

Problem Statement:

Background:

The CIFAR-10 dataset is a widely used benchmark in the field of computer vision. It consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. The dataset is split into 50,000 training images and 10,000 test images.

Objective:

The objective of this project is twofold:

1. Image Classification: Develop a model capable of accurately classifying images into one of the 10 predefined classes.
2. Image Regression: Implement a regression model to predict a continuous variable (e.g., bounding box coordinates, image attributes, etc.) associated with each image.

Classification:

Steps:

1. Data Preparation
2. Data Preprocessing
3. Model Selection
4. Model Training
5. Model Evaluation
6. Fine-tuning and Optimization
7. Deployment

Program:

```
!pip install keras-tuner
```

Output:

Successfully installed keras-tuner-1.3.4 kt-legacy-1.0.4

Program:

```
import tensorflow as tf
from tensorflow import keras
import keras_tuner as kt
import sklearn
from sklearn.model_selection import train_test_split

import tensorflow as tf

print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
```

Output:

Num GPUs Available: 1

Program:

```
(X_train, y_train), (X_test, y_test) = keras.datasets.cifar10.load_data()
```

Output:

Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz

170498071/170498071 [==============================] - 2s 0us/step

Program:

```
X_train = X_train.astype('float32') / 255.0

X_test = X_test.astype('float32') / 255.0

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

def BasicCNN():

    model = keras.Sequential()

    model.add(keras.layers.Conv2D(filters=32, kernel_size=(3,3), activation="relu", input_shape=(32, 32, 3)))

    model.add(keras.layers.Conv2D(filters=32, kernel_size=(3,3), activation="relu"))
```

```
    model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))

    model.add(keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation="relu", padding="same"))

    model.add(keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation="relu"))

    model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))

    model.add(keras.layers.Flatten())

    model.add(keras.layers.Dense(units=512, activation="relu"))

    model.add(keras.layers.Dense(units=10, activation="softmax"))

    opt = keras.optimizers.Adam(learning_rate=0.01)

    model.compile(optimizer=opt, loss="sparse_categorical_crossentropy", metrics=['accuracy'])

    return model

model = BasicCNN()

model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10)
```

Output:

```
Epoch 1/10

1250/1250 [==============================] - 8s 5ms/step - loss: 2.0044 - accuracy: 0.2714 - val_loss: 1.7237 - val_accuracy: 0.3921

Epoch 2/10

1250/1250 [==============================] - 7s 5ms/step - loss: 1.6246 - accuracy: 0.4178 - val_loss: 1.5953 - val_accuracy: 0.4371

Epoch 3/10

1250/1250 [==============================] - 6s 5ms/step - loss: 1.4343 - accuracy: 0.4851 - val_loss: 1.3435 - val_accuracy: 0.5135

Epoch 4/10

1250/1250 [==============================] - 7s 5ms/step - loss: 1.2986 - accuracy: 0.5371 - val_loss: 1.2377 - val_accuracy: 0.5577
```

Epoch 5/10

1250/1250 [==============================] - 8s 6ms/step - loss: 1.1868 - accuracy: 0.5837 - val_loss: 1.1969 - val_accuracy: 0.5764

Epoch 6/10

1250/1250 [==============================] - 6s 5ms/step - loss: 1.0885 - accuracy: 0.6194 - val_loss: 1.1000 - val_accuracy: 0.6135

Epoch 7/10

1250/1250 [==============================] - 7s 5ms/step - loss: 0.9958 - accuracy: 0.6524 - val_loss: 1.1038 - val_accuracy: 0.6157

Epoch 8/10

1250/1250 [==============================] - 6s 5ms/step - loss: 0.9114 - accuracy: 0.6816 - val_loss: 1.0351 - val_accuracy: 0.6398

Epoch 9/10

1250/1250 [==============================] - 7s 5ms/step - loss: 0.8297 - accuracy: 0.7103 - val_loss: 1.0328 - val_accuracy: 0.6481

Epoch 10/10

1250/1250 [==============================] - 6s 5ms/step - loss: 0.7507 - accuracy: 0.7401 - val_loss: 0.9984 - val_accuracy: 0.6603

<keras.callbacks.History at 0x7f07000a2a00>

Program:

model.evaluate(X_test, y_test)

Output:

313/313 [==============================] - 12s 39ms/step - loss: 2.3732 - accuracy: 0.6480

[2.373150110244751, 0.6480000019073486]

Regression:

Steps:

1. Data Preparation
2. Data Preprocessing
3. Model Definition
4. Loss Function
5. Training
6. Model Evaluation
7. Fine-tuning and Optimization
8. Deployment

Program:

```
import tensorflow as tf
import sklearn
import keras_tuner as kt
from tensorflow import keras
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt

(X_train, y_train), (X_test, y_test) = keras.datasets.cifar10.load_data()
X_train = X_train.astype('float32') / 255.0
X_test = X_test.astype('float32') / 255.0
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2,
random_state=42)
```

Output:

Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz

170498071/170498071 [==============================] - 2s 0us/step

Program:

```
def BaseClassification():

  model = keras.Sequential()

  model.add(keras.layers.Flatten(input_shape=(32,32,3)))
```

```
  model.add(keras.layers.Dense(units=10, activation="softmax"))

  opt = keras.optimizers.Adam(learning_rate=0.01)

  model.compile(optimizer=opt, loss="sparse_categorical_crossentropy", metrics=['accuracy'])

  return model

model = BaseClassification()

model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10)
```

Output:

Epoch 1/10

1250/1250 [==============================] - 6s 4ms/step - loss: 1.9631 - accuracy: 0.2980 - val_loss: 1.8718 - val_accuracy: 0.3392

Epoch 2/10

1250/1250 [==============================] - 3s 3ms/step - loss: 1.8577 - accuracy: 0.3438 - val_loss: 1.9037 - val_accuracy: 0.3357

Epoch 3/10

1250/1250 [==============================] - 3s 3ms/step - loss: 1.8230 - accuracy: 0.3615 - val_loss: 1.8111 - val_accuracy: 0.3642

Epoch 4/10

1250/1250 [==============================] - 4s 3ms/step - loss: 1.8029 - accuracy: 0.3749 - val_loss: 1.7976 - val_accuracy: 0.3687

Epoch 5/10

1250/1250 [==============================] - 3s 3ms/step - loss: 1.7893 - accuracy: 0.3767 - val_loss: 1.7996 - val_accuracy: 0.3709

Epoch 6/10

1250/1250 [==============================] - 4s 3ms/step - loss: 1.7810 - accuracy: 0.3794 - val_loss: 1.7825 - val_accuracy: 0.3824

Epoch 7/10

1250/1250 [==============================] - 4s 3ms/step - loss: 1.7712 - accuracy: 0.3836 - val_loss: 1.8278 - val_accuracy: 0.3514

Epoch 8/10

1250/1250 [==============================] - 3s 3ms/step - loss: 1.7624 - accuracy: 0.3870 - val_loss: 1.7511 - val_accuracy: 0.3924

Epoch 9/10

1250/1250 [==============================] - 3s 3ms/step - loss: 1.7553 - accuracy: 0.3905 - val_loss: 1.7967 - val_accuracy: 0.3693

Epoch 10/10

1250/1250 [==============================] - 4s 3ms/step - loss: 1.7523 - accuracy: 0.3922 - val_loss: 1.8263 - val_accuracy: 0.3684

Program:

model.evaluate(X_test, y_test)

Output:

313/313 [==============================] - 1s 2ms/step - loss: 1.7593 - accuracy: 0.3857

[1.7593483924865723, 0.385699987411499]

Conclusion:

In this project, we explored the tasks of image classification and regression on the CIFAR-10 dataset, which serves as a benchmark dataset in the field of computer vision. We approached the problem from two perspectives: classifying images into predefined categories and predicting continuous variables associated with each image. In conclusion, this project contributes to the understanding of image classification and regression tasks on the CIFAR-10 dataset, paving the way for future research and applications in computer vision.