University Institute of Engineering

Department of Electronics & Communication Engineering

Experiment No. :- 5

Student Name: Priyanshu Mathur   **UID:** 20BEC1073
Branch: Electronics and Communication Engineering   **Section/Group:** A
Semester: 7th   **Date of Performance:** 12/09/2023
Subject Name: Artificial Intelligence & Machine Learning   **Subject Code:** 20ECA-445

1. **Aim of the practical:** Write a program for understanding and implementation of unsupervised learning in machine learning.

2. **Tool Used:** Google Colab

3. Theory:

Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision. Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format. Unsupervised learning is helpful for finding useful insights from the data. Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.

4. Steps for experiment/practical:

Step 1: - Open Google Collab

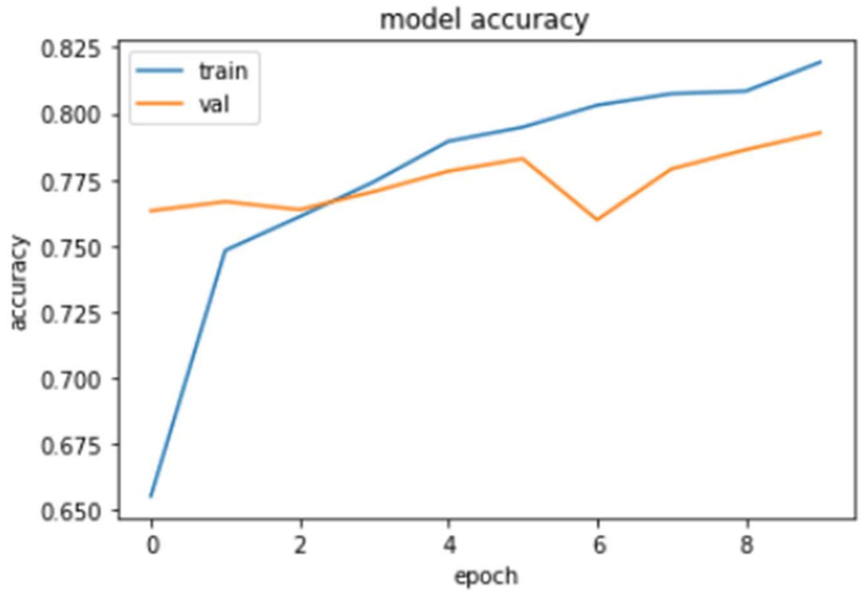
Step 2: - Create a new notebook

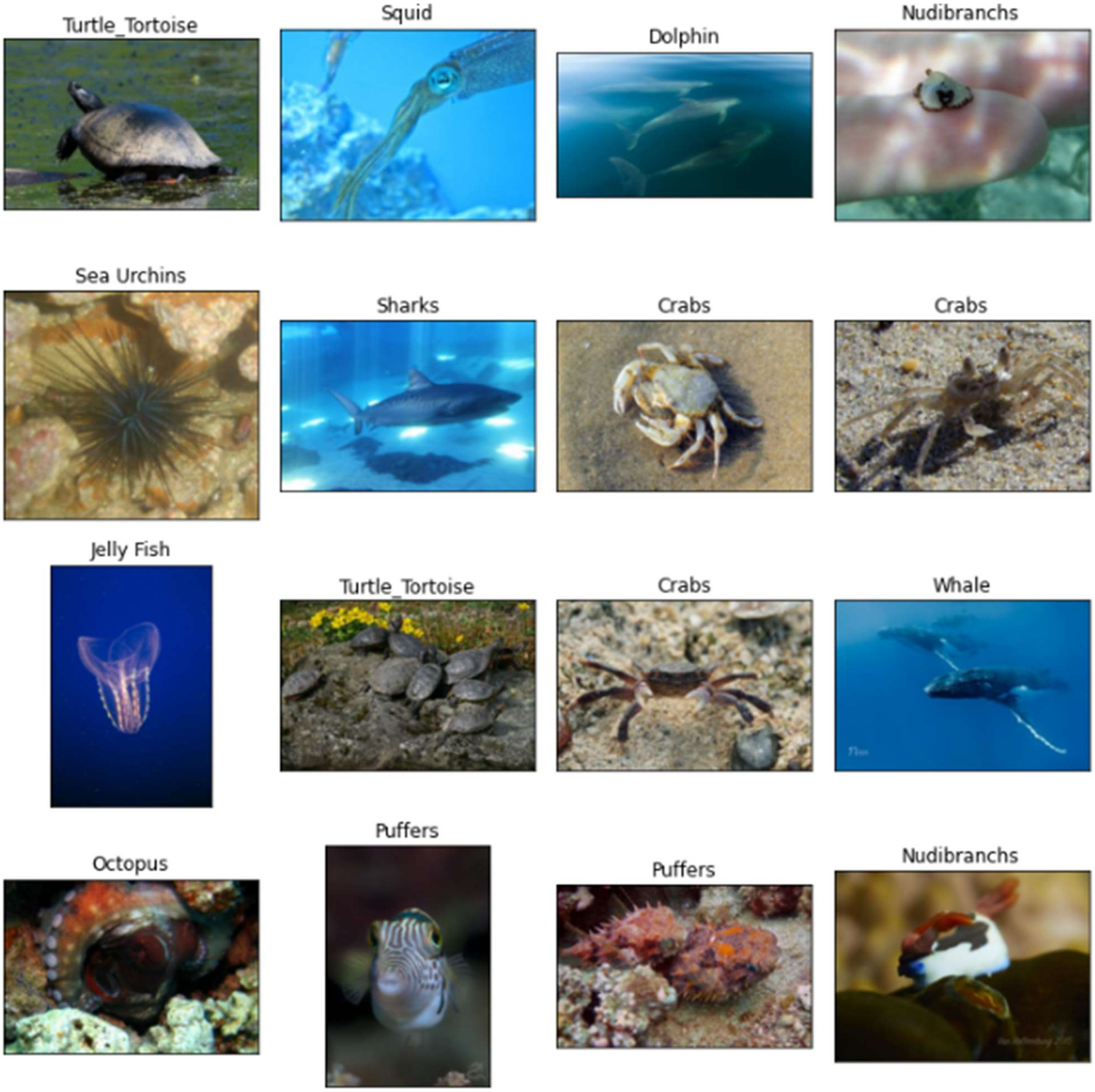
Step 3: - Write the code given below and run it.

Program Code and Simulation Output:-
Code:-

```
#Environment check
import os
import pandas as pd
import numpy as np
import tensorflow as tf, GlobalAveragePooling2D
import seaborn as sns
import matplotlib.pyplot as plt


# Metrics
from sklearn.metrics import classification_report, confusion_matrix
import itertools
data = "../input/sea-animals-image-dataste"
image_dir = Path(data)

# Get filepaths and labels
filepaths = list(image_dir.glob(r'**/*.JPG')) + list(image_dir.glob(r'**/*.jpg')) +
list(image_dir.glob(r'**/*.png')) + list(image_dir.glob(r'**/*.PNG'))

labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], filepaths))

filepaths = pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='Label')

# Concatenate filepaths and labels
image_df = pd.concat([filepaths, labels], axis=1)
import PIL
from pathlib import Path
from PIL import UnidentifiedImageError

path = Path("../input/sea-animals-image-dataste").rglob("*.jpg")
for img_p in path:
    try:
        img = PIL.Image.open(img_p)
    except PIL.UnidentifiedImageError:
            print(img_p)
image_df
# Display 16 picture of the dataset with their labels
random_index = np.random.randint(0, len(image_df), 16)
fig, axes = plt.subplots(nrows=4, ncols=4, figsize=(10, 10),
                        subplot_kw={'xticks': [], 'yticks': []})

for i, ax in enumerate(axes.flat):
    ax.imshow(plt.imread(image_df.Filepath[random_index[i]]))
    ax.set_title(image_df.Label[random_index[i]])
plt.tight_layout()
plt.show()
train_datagen = ImageDataGenerator(rescale=1./255,rotation_range = 40, width_shift_range = 0.2,
height_shift_range = 0.2,
                                  shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode =
'nearest',
    validation_split=0.2) # set validation split
train_images = train_datagen.flow_from_directory(
    data,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training') # set as training data

validation_images = train_datagen.flow_from_directory(
    data , # same directory as training data
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation') # set as validation data
# Load the pretained model
mobile_model = Sequential()

pretrained_model = tf.keras.applications.MobileNetV2(
                    input_shape=(224, 224, 3),
   include_top=False,
   weights='imagenet',
   pooling='avg'
)

pretrained_model.trainable = False

mobile_model.add(pretrained_model)
mobile_model.add(Flatten())
mobile_model.add(Dense(512, activation='relu'))
mobile_model.add(Dropout(0.2))
mobile_model.add(Dense(19, activation='softmax'))
mobile_model.summary()
mobile_model.compile(loss = 'categorical_crossentropy', optimizer = tf.keras.optimizers.Adam(),
metrics = ['accuracy'])

history = mobile_model.fit(train_images,
   steps_per_epoch=len(train_images),
   validation_data=validation_images,
   validation_steps=len(validation_images),
   epochs=10)
print(history.history.keys())
#Accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
# loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

Plot :





Result and Discussion: - In this experiment we have a dataset of Aquatic life . In the dataset we have image of aqatic animals .We are achive detection of aquatic life detection by image of such animal

Learning outcomes (What I have learnt):

- Learnt about unsupervised learning.
- Learn about different types of unsupervised learning algorithm.
- Learnt about K-Mean clustering algorithm.