TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA CƠ KHÍ CHẾ TẠO MÁY

BÁO CÁO CUỐI KÌ

TRÍ TUỆ NHÂN TẠO

NHẬN DIỆN CÁC LOẠI PHƯƠNG TIỆN

GVHD : PGS. Nguyễn Trường Thịnh

SVTH : Huỳnh Trọng Trí

MSSV : 20146209

Lớp : Sáng thứ 2

Nhóm : 08

TP.HCM, tháng 5 năm 2023

CHƯƠNG 1. GIỚI THIỆU

1.1 Giới thiệu về nhận diện các loại phương tiện

Nhận diện các loại xe là một công nghệ tiên tiến trong lĩnh vực nhận dạng hình ảnh, cho phép máy tính nhận biết và phân loại các loại xe khác nhau dựa trên hình ảnh hoặc video. Công nghệ này sử dụng các thuật toán máy học và học sâu để phân tích đặc trưng của hình ảnh và tìm ra các đặc điểm phân biệt giữa các loại xe.

Các hệ thống nhận diện xe thường sử dụng camera hoặc cảm biến hình ảnh để thu thập dữ liệu từ môi trường xung quanh. Sau đó, hình ảnh được truyền vào các mô hình máy học hoặc mạng neural để xử lý và phân loại. Quá trình này bao gồm việc trích xuất các đặc trưng quan trọng từ hình ảnh, như hình dạng, kích thước, màu sắc, biểu ngữ và cấu trúc của xe.

1.2 Lý do chọn đề tài

Trong thời đại hiện đại, xe cộ đóng vai trò không thể thiếu trong cuộc sống của chúng ta. Tuy nhiên, việc quản lý và kiểm soát hàng triệu xe trên đường là một thách thức lớn đối với các tổ chức và cơ quan quản lý giao thông. Để giải quyết vấn đề này, việc nhận diện và phân loại các loại xe khác nhau đã trở thành một yếu tố quan trọng trong nghiên cứu và phát triển công nghệ. Việc nhận diện các loại xe không chỉ giúp đảm bảo an toàn và tuân thủ quy tắc giao thông, mà còn mang lại nhiều lợi ích khác. Đầu tiên, việc phân loại và đếm các loại xe trên đường giúp cung cấp thông tin cần thiết cho việc quản lý giao thông, từ việc tối ưu hóa luồng giao thông đến xác định các điểm ùn tắc và thiết kế hệ thống giao thông thông minh.

Thứ hai, nhận diện xe cũng đóng vai trò quan trọng trong lĩnh vực an ninh và an toàn. Với khả năng xác định và theo dõi xe quá tốc độ, xe vi phạm luật lệ giao thông hoặc các phương tiện có liên quan đến các hoạt động bất hợp pháp, hệ thống nhận diện xe có thể giúp cải thiện an ninh đường phố và giúp trong việc điều tra tội phạm.

Cuối cùng, việc nhận diện xe cũng có ứng dụng trong các lĩnh vực kinh doanh và tiếp thị. Việc phân loại và theo dõi loại xe đang di chuyển trên đường có thể được sử dụng để tạo ra quảng cáo và nội dung phù hợp cho từng loại xe và khách hàng tiềm năng. Điều này giúp tối ưu hóa chiến dịch quảng cáo và tăng cường hiệu quả tiếp thị.

1.3. Mục tiêu nghiên cứu

- Xây dựng một hệ thống nhận diện phương tiện đáng tin cậy và chính xác.

- Ứng dụng kết quả nghiên cứu vào thực tế và đóng góp vào việc cải thiện quản lý giao thông, an ninh và các lĩnh vực khác.

1.4 Phương pháp nghiên cứu

- Thu thập Data (ảnh) từ Internet sau đó hiệu chỉnh để ra được nhiều dữ liệu hợn.

- Xây dựng một bộ dữ liệu huấn luyện chứa các hình ảnh của các loại xe khác nhau, kèm theo nhãn xác định loại xe tương ứng.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Thuật toán CNN - Convolutional Neural Network

* Convolutional Neural Network là gì

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay.

CNN được lấy cảm hứng từ cách thức hoạt động của thị giác con người. Nó sử dụng các lớp tích chập để trích xuất đặc trưng từ hình ảnh và các lớp pooling để giảm kích thước của đầu ra. Mạng CNN bao gồm nhiều lớp tích chập, lớp kích hoạt phi tuyến và lớp pooling để tạo ra một biểu diễn có ý nghĩa của dữ liệu đầu vào.

Lớp tích chập trong CNN làm việc bằng cách trượt một bộ lọc (kernel) qua toàn bộ hình ảnh để tính toán các tích chập và tạo ra các đặc trưng cục bộ. Lớp kích hoạt phi tuyến (thường là hàm ReLU) được áp dụng sau mỗi lớp tích chập để giúp mạng neural học được các đặc trưng phi tuyến tính. Lớp pooling giúp giảm kích thước của đầu ra bằng cách lấy giá trị lớn nhất hoặc trung bình trong một vùng của hình ảnh. Sau các lớp tích chập và pooling, các đặc trưng đã được trích xuất từ hình ảnh sẽ được chuyển đến các lớp kết nối đầy đủ (fully connected layers). Các lớp này có chức năng phân loại và đưa ra dự đoán về đối tượng trong hình ảnh dựa trên các đặc trưng đã học.

CNN đã chứng minh tính hiệu quả cao trong nhiều lĩnh vực như nhận diện đối tượng, nhận dạng khuôn mặt, phân loại hình ảnh và cả nhận diện và phân tích chuỗi thời gian. Với khả năng học tự động các đặc trưng từ dữ liệu và xử lý dữ liệu không gian, CNN đã trở thành công cụ quan trọng trong lĩnh vực trí tuệ nhân tạo và xử lý hình ảnh.

2.2 Các thư viện trong CNN

Có nhiều thư viện thông dụng được sử dụng để triển khai mạng neural tích chập (CNN) và các chức năng liên quan trong việc xử lý hình ảnh. Dưới đây là một số thư viện phổ biến và chức năng của chúng khi làm việc với CNN:

- **TensorFlow**: TensorFlow là một thư viện mã nguồn mở rất mạnh mẽ cho việc xây dựng và triển khai các mạng neural, bao gồm cả CNN. Nó cung cấp một cách tiếp cận linh hoạt để xây dựng, huấn luyện và triển khai mô hình CNN. TensorFlow cung cấp một loạt các lớp, phương pháp và chức năng hỗ trợ cho việc triển khai CNN.
- **PyTorch**: PyTorch là một thư viện mã nguồn mở khác rất phổ biến trong việc xây dựng và huấn luyện mạng neural, bao gồm cả CNN. Nó cung cấp các lớp và công cụ mạnh mẽ để xây dựng và tùy chỉnh các mô hình CNN, cung cấp khả năng tích hợp dễ dàng với các thư viện khác và hỗ trợ tính toán trên GPU hiệu quả.
- **Keras**: Keras là một thư viện mạnh mẽ và dễ sử dụng cho việc xây dựng mô hình mạng neural, bao gồm cả CNN. Nó cung cấp các lớp và phương thức tiện ích để xây dựng, huấn luyện và đánh giá mô hình CNN một cách dễ dàng. Keras cũng hỗ trợ tích hợp với TensorFlow và PyTorch.
- **OpenCV**: OpenCV là một thư viện mã nguồn mở được sử dụng rộng rãi trong xử lý hình ảnh. Nó cung cấp các chức năng và phương pháp để xử lý, biến đổi và trích xuất thông tin từ hình ảnh, góp phần quan trọng trong việc tiền xử lý dữ liệu hình ảnh trước khi đưa vào mạng neural.
- **scikit-learn**: scikit-learn là một thư viện phổ biến trong machine learning. Mặc dù không chuyên về mạng neural, nhưng nó cung cấp một số công cụ hữu ích để huấn luyện và đánh giá mô hình CNN, cùng với các chức năng khác như tiền xử lý dữ liệu và trích xuất đặc trưng.

CHƯƠNG 3. ỨNG DỤNG

3.1. Xây dựng mô hình – Trainning model

Sử dụng Google Colab để xây dựng mô hình trainnin

Bước 1: Khai báo các thư viện cần thiết

```
from os import listdir
from numpy import asarray
from numpy import save
from keras.utils.image_utils import img_to_array
from keras.utils import load_img
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from keras.preprocessing.image import ImageDataGenerator
```

Bước 2: Liên kết Google Drive

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

Bước 3: Lấy và mô tả dữ liệu

```
[ ]  folder = '/content/drive/MyDrive/Final_AI/anh_xe/'
     photos, labels = list(), list()
     for file in listdir(folder):
       output = 0.0
       if file.startswith('xe_may'):
         output = 1.0  #'xe máy'
       if file.startswith('xe_hoi'):
         output = 2.0  #'xe hơi'
       if file.startswith('xe_dap'):
         output = 3.0  #'xe đạp'
       if file.startswith('xe_tai'):
         output = 4.0  #'xe tải'
       photo = load_img(folder+file, target_size= (128,128))
     #Hiệu chỉnh ảnh
       datagen = ImageDataGenerator(
           rotation_range= 10,
           horizontal_flip=True,
           width_shift_range=0.1,
           height_shift_range=0.1,

       )
       # Thay đổi kích thước ảnh thành (1, height, width, channels)
       photo_np = np.expand_dims(photo, axis=0)
       # Tạo batch dữ liệu biến đổi từ ảnh gốc
       batch = datagen.flow(photo_np, batch_size=1)
       for i in range(4):
           augmented_image = next(batch)[0].astype('uint8')
           photos.append(augmented_image)
           labels.append(output)
```

```
       photo = img_to_array (photo)
       photos.append(photo)
       labels.append(output)
     photos = asarray(photos)
     labels = asarray(labels)
     save('/content/drive/MyDrive/Final_AI/xe_photo.npy', photos)
     save('/content/drive/MyDrive/Final_AI/xe_label.npy', labels)
```

```
[30] #Tải ảnh và nhãn đã lưu lên để xử lí
     photo = np.load('/content/drive/MyDrive/Final_AI/xe_photo.npy')
     label = np.load('/content/drive/MyDrive/Final_AI/xe_label.npy')
```

```
[31] #Chia dữ liệu thành 2 phần train và test để kiểm tra mô hình
     split_index = int(0.2*len(photo))
     test_X,test_Y = photo[:split_index],label[:split_index]
     train_X, train_Y = photo[split_index:],label[split_index:]
```

```
[ ]  train_X = train_X.reshape((2507,128,128,3))
     train_X = train_X.astype('float32')/255
     test_X = test_X.reshape((278,128,128,3))
     test_X = test_X.astype('float32')/255
```

```
[ ]  from keras.utils import to_categorical
```

```
[ ]  train_Y = to_categorical(train_Y,5)
     test_Y = to_categorical(test_Y,5)
```

```
[ ]  # Khai báo các thư viện cần thiết để xây dựng mô hình CNN
     from keras.models import Sequential , Model
     from keras.layers import Dense , Flatten, Dropout, Conv2D, MaxPooling2D ,Normalization,Input
     from keras.optimizers import Adam
```

```
[ ]  # Định nghĩa các biến
     batch_size = 50  # số lượng học
     epochs =  30   #số lần học
     classes = 5    # Số lớp
```

Bước 4: Xây dựn mô hình CNN

```
model=Sequential()    # Khởi tạo đối tượng để xây dựng mô hình CNN
model.add(Conv2D(32,kernel_size=(6,6),activation ='linear',input_shape= (128,128,3),padding= 'same'))
from keras.layers import LeakyReLU
model.add(LeakyReLU(alpha= 0))
model.add(MaxPooling2D((2,2), padding= 'same'))

model.add(Conv2D(32,(6,6), activation= 'linear', padding= 'same'))
model.add(LeakyReLU(alpha= 0.1))
model.add(MaxPooling2D((2,2), padding= 'same'))

model.add(Conv2D(64,(6,6), activation= 'linear', padding= 'same'))
model.add(LeakyReLU(alpha= 0.1))
model.add(MaxPooling2D((2,2), padding= 'same'))

model.add(Conv2D(64,(6,6), activation= 'linear', padding= 'same'))
model.add(LeakyReLU(alpha= 0.1))
model.add(MaxPooling2D((2,2), padding= 'same'))

model.add(Conv2D(128,(6,6), activation= 'linear', padding= 'same'))
model.add(LeakyReLU(alpha= 0.1))
model.add(MaxPooling2D((2,2), padding= 'same'))

from keras.backend import categorical_crossentropy
from keras.losses import categorical_crossentropy
model.add(Flatten())
```

```
model.add(Dense(2048,activation='linear'))
model.add(LeakyReLU(alpha=0.1))
model.add(Dense(1042,activation='linear'))
model.add(LeakyReLU(alpha=0.1))
model.add(Dense(128,activation='linear'))
model.add(LeakyReLU(alpha=0.1))
model.add(Dense(classes,activation='softmax'))
model.summary()
```

Các thông số Input, Ouput cụ thể được thể hiện trong bảng thông số mô hình dưới dây:

```
Model: "sequential_14"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_78 (Conv2D)          (None, 128, 128, 32)      3488

 leaky_re_lu_120 (LeakyReLU)  (None, 128, 128, 32)     0

 max_pooling2d_78 (MaxPoolin  (None, 64, 64, 32)       0
 g2D)

 conv2d_79 (Conv2D)          (None, 64, 64, 32)        36896

 leaky_re_lu_121 (LeakyReLU)  (None, 64, 64, 32)       0

 max_pooling2d_79 (MaxPoolin  (None, 32, 32, 32)       0
 g2D)

 conv2d_80 (Conv2D)          (None, 32, 32, 64)        73792

 leaky_re_lu_122 (LeakyReLU)  (None, 32, 32, 64)       0

 max_pooling2d_80 (MaxPoolin  (None, 16, 16, 64)       0
 g2D)
```

```
conv2d_81 (Conv2D)          (None, 16, 16, 64)        147520

leaky_re_lu_123 (LeakyReLU)  (None, 16, 16, 64)       0

max_pooling2d_81 (MaxPoolin  (None, 8, 8, 64)         0
g2D)

conv2d_82 (Conv2D)          (None, 8, 8, 128)         295040

leaky_re_lu_124 (LeakyReLU)  (None, 8, 8, 128)        0

max_pooling2d_82 (MaxPoolin  (None, 4, 4, 128)        0
g2D)

flatten_14 (Flatten)        (None, 2048)              0

dense_56 (Dense)            (None, 2048)              4196352

leaky_re_lu_125 (LeakyReLU)  (None, 2048)             0

dense_57 (Dense)            (None, 1042)              2135058

leaky_re_lu_126 (LeakyReLU)  (None, 1042)             0

dense_58 (Dense)            (None, 128)               133504

leaky_re_lu_127 (LeakyReLU)  (None, 128)              0

dense_59 (Dense)            (None, 5)                 645

=================================================================
Total params: 7,022,295
Trainable params: 7,022,295
Non-trainable params: 0
_________________________________________________________________
```

Bước 5: Huấn luyện mô hình

```
[ ] from keras.losses import categorical_crossentropy
    model.compile(optimizer= Adam(), loss= categorical_crossentropy, metrics= ['accuracy'])
    train = model.fit(train_X, train_Y, epochs= epochs, batch_size= batch_size, verbose= 1)
```

Mô hình được huấn luyện với số lượng học 50, lần học 30, ta được kết quả như sau:

```
Epoch 1/30
51/51 [==============================] - 3s 42ms/step - loss: 1.4413 - accuracy: 0.2756
Epoch 2/30
51/51 [==============================] - 2s 37ms/step - loss: 1.2819 - accuracy: 0.3618
Epoch 3/30
51/51 [==============================] - 2s 38ms/step - loss: 1.0237 - accuracy: 0.5030
Epoch 4/30
51/51 [==============================] - 2s 40ms/step - loss: 0.8569 - accuracy: 0.5716
Epoch 5/30
51/51 [==============================] - 2s 44ms/step - loss: 0.7545 - accuracy: 0.6498
Epoch 6/30
51/51 [==============================] - 2s 39ms/step - loss: 0.5554 - accuracy: 0.7551
Epoch 7/30
51/51 [==============================] - 2s 37ms/step - loss: 0.4842 - accuracy: 0.8133
Epoch 8/30
51/51 [==============================] - 2s 37ms/step - loss: 0.3429 - accuracy: 0.8656
Epoch 9/30
51/51 [==============================] - 2s 37ms/step - loss: 0.2162 - accuracy: 0.9158
Epoch 10/30
51/51 [==============================] - 2s 37ms/step - loss: 0.1554 - accuracy: 0.9422
Epoch 11/30
51/51 [==============================] - 2s 38ms/step - loss: 0.0924 - accuracy: 0.9669
Epoch 12/30
51/51 [==============================] - 2s 40ms/step - loss: 0.1226 - accuracy: 0.9569
Epoch 13/30
51/51 [==============================] - 2s 38ms/step - loss: 0.1151 - accuracy: 0.9637
Epoch 14/30
51/51 [==============================] - 2s 37ms/step - loss: 0.1235 - accuracy: 0.9605
Epoch 15/30
51/51 [==============================] - 2s 37ms/step - loss: 0.0278 - accuracy: 0.9912
```

```
Epoch 15/30
51/51 [==============================] - 2s 37ms/step - loss: 0.0278 - accuracy: 0.9912
Epoch 16/30
51/51 [==============================] - 2s 37ms/step - loss: 0.0583 - accuracy: 0.9821
Epoch 17/30
51/51 [==============================] - 2s 37ms/step - loss: 0.1269 - accuracy: 0.9605
Epoch 18/30
51/51 [==============================] - 2s 38ms/step - loss: 0.0476 - accuracy: 0.9848
Epoch 19/30
51/51 [==============================] - 2s 39ms/step - loss: 0.0028 - accuracy: 0.9996
Epoch 20/30
51/51 [==============================] - 2s 38ms/step - loss: 2.2726e-04 - accuracy: 1.0000
Epoch 21/30
51/51 [==============================] - 2s 37ms/step - loss: 1.0155e-04 - accuracy: 1.0000
Epoch 22/30
51/51 [==============================] - 2s 37ms/step - loss: 5.2387e-05 - accuracy: 1.0000
Epoch 23/30
51/51 [==============================] - 2s 37ms/step - loss: 4.0640e-05 - accuracy: 1.0000
Epoch 24/30
51/51 [==============================] - 2s 38ms/step - loss: 3.3877e-05 - accuracy: 1.0000
Epoch 25/30
51/51 [==============================] - 2s 39ms/step - loss: 2.8823e-05 - accuracy: 1.0000
Epoch 26/30
51/51 [==============================] - 2s 40ms/step - loss: 2.4668e-05 - accuracy: 1.0000
Epoch 27/30
51/51 [==============================] - 2s 38ms/step - loss: 2.1748e-05 - accuracy: 1.0000
Epoch 28/30
51/51 [==============================] - 2s 37ms/step - loss: 1.9261e-05 - accuracy: 1.0000
Epoch 29/30
51/51 [==============================] - 2s 37ms/step - loss: 1.7351e-05 - accuracy: 1.0000
Epoch 30/30
51/51 [==============================] - 2s 37ms/step - loss: 1.5693e-05 - accuracy: 1.0000
```

Bước 6: Lưu mô hình đã huấn luyện

```
[ ]  model.save('/content/drive/MyDrive/Final_AI/train.h5')
```

CHƯƠNG 4. KẾT LUẬN

❖ **Sử dụng PyCharm để làm giao diện và test ảnh**

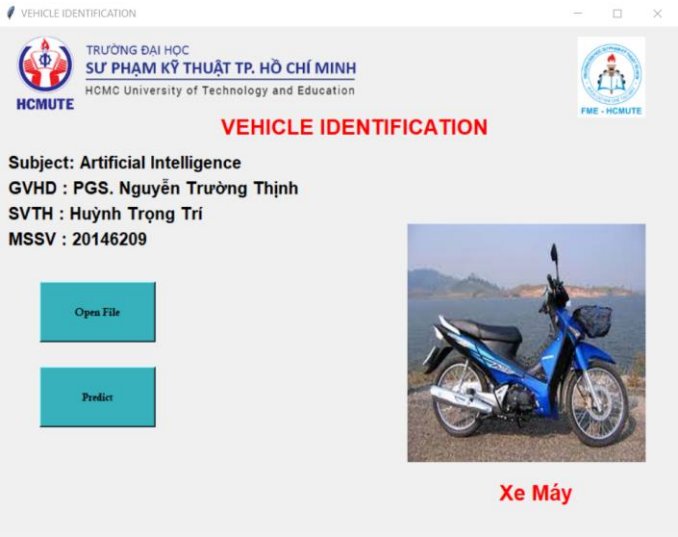
4.1 Tạo giao diện và chạy chương trình

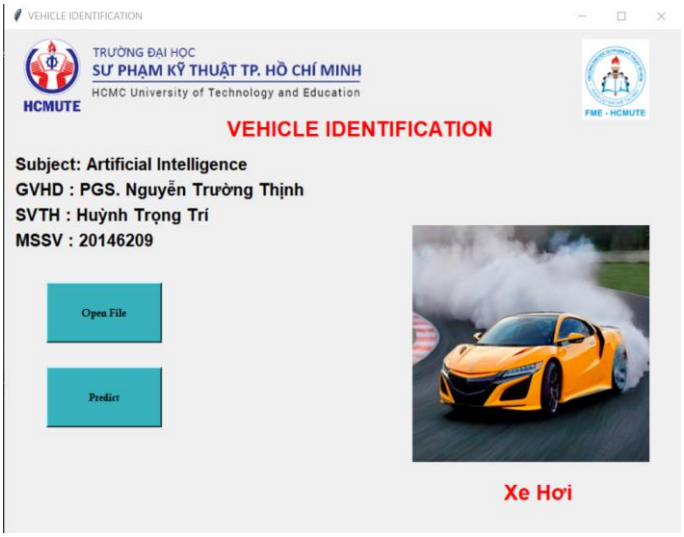
```
from tkinter import filedialog
from tkinter import *
from PIL import ImageTk, Image
from keras.models import load_model
import numpy as np
from keras.utils.image_utils import load_img, img_to_array
model = load_model('train.h5')
def open():
    myLabel18.configure(text="")
    global filename
    filename = filedialog.askopenfilename (initialdir="", title ="select A file", filetypes =(("jpg files", "*.jpg"),("all file","*.*")))
    global image
    image= Image.open(filename)
    image = image.resize((280,280))
    global my_image
    my_image = ImageTk.PhotoImage(image)
    global L1
    L1 = Label(root, image=my_image)
    L1.place(x=480, y=230)

# Tạo cửa sổ giao diện chính
root = Tk()
root.title("VEHICLE IDENTIFICATION")
root.geometry("800x600")

lgtruong = ImageTk.PhotoImage(Image.open("logotruong.png"))
lblgtruong = Label(image=lgtruong)
lblgtruong.place(x=20, y=5)
```

```
lgkhoa = ImageTk.PhotoImage(Image.open("logokhoa.png"))
lblgkhoa = Label(image=lgkhoa)
lblgkhoa.place(x=680, y=10)

def recognition():
    img = load_img(filename, target_size=(128, 128))
    img = img_to_array(img)
    img = img.reshape(1, 128, 128, 3)
    img = img.astype('float32')
    img = img / 255
    Custom = np.argmax(model.predict(img), axis=-1)
    if (Custom == 1):
        myLabel18.configure(text=" Xe Máy")
    if (Custom == 2):
        myLabel18.configure(text=" Xe Hơi")
    if (Custom == 3):
        myLabel18.configure(text=" Xe Đạp")
    if (Custom == 4):
        myLabel18.configure(text=" Xe Tải")

B1 = Button(root, text="Open File", bg='#37b1bc', cursor="hand2", font=("Calisto MT", 9, "bold"), height=4, width=18,command=open)
B1.place(x=50, y=300)
B2 = Button(root, text="Predict", bg='#37b1bc', cursor="hand2", font=("Calisto MT", 9, "bold"), height=4, width=18,command=recognition)
B2.place(x=50, y=400)
```

```
myLabel4 = Label(root, text="VEHICLE IDENTIFICATION", font="Arial 19 bold", fg="red")
myLabel4.place(x = 260, y = 100)
myLabel3 = Label(root, text="SVTH : Huỳnh Trọng Trí", font="Arial 15 bold")
myLabel3.place(x = 10, y = 220,anchor = W)
myLabel2 = Label(root, text="MSSV : 20146209", font="Arial 15 bold")
myLabel2.place(x = 10, y = 250,anchor = W)
myLabel5 = Label(root, text="Subject: Artificial Intelligence", font="Arial 15 bold")
myLabel6 = Label(root, text="GVHD : PGS. Nguyễn Trường Thịnh", font="Arial 15 bold")
myLabel5.place(x = 10, y = 160,anchor = W)
myLabel6.place(x = 10, y = 190,anchor = W)

myLabel18 = Label(root,text= "", font="Arial 19 bold", fg="red")
myLabel18.place(x = 580, y = 530)
root.mainloop()
```

❖ Kết quả sau khi test





4.2 Kết luận

Sử dụng trí tuệ nhân tạo (AI) để nhận diện phương tiện đã mang lại những tiến bộ đáng kể. AI giúp tự động hóa việc nhận diện và phân loại phương tiện với độ chính xác và hiệu suất cao. Ứng dụng của AI trong lĩnh vực này rất đa dạng và góp phần cải thiện an toàn và hiệu suất trong giao thông và vận tải. Mặc dù còn một số thách thức, nhưng tiềm năng của AI trong nhận diện phương tiện là rất hứa hẹn và đang tiếp tục được nghiên cứu và phát triển.

Tài Liệu Tham Khảo

Link drive:

https://drive.google.com/drive/u/0/folders/1PInhTZWOZFacTKQkr19ybjtECj62nwUl

QR GitHub