TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM

KHOA CƠ KHÍ CHẾ TẠO MÁY

BÁO CÁO MÔN HỌC THỊ GIÁC MÁY

ĐỀ TÀI: NHẬN DIỆN PHƯƠNG TIỆN GIAO THÔNG VÀ PHÂN CHIA LỐI ĐI HỢP LÝ

GVHD:	**TS. Nguyễn Văn Thái**	
SVTH:	**Dương Thành Đạt**	**20146179**
	Nguyễn Quốc Tiến	**20146538**
	Chíu Sáng Hùng	**20146493**
Nhóm:	**MAVI332529_22_2_04**	
Ngành:	**CNKT CƠ ĐIỆN TỬ**	

Tp. Hồ Chí Minh, tháng 6 năm 2022

Nhận xét của giảng viên

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

TP. Hồ Chí Minh, ngày tháng năm 2023

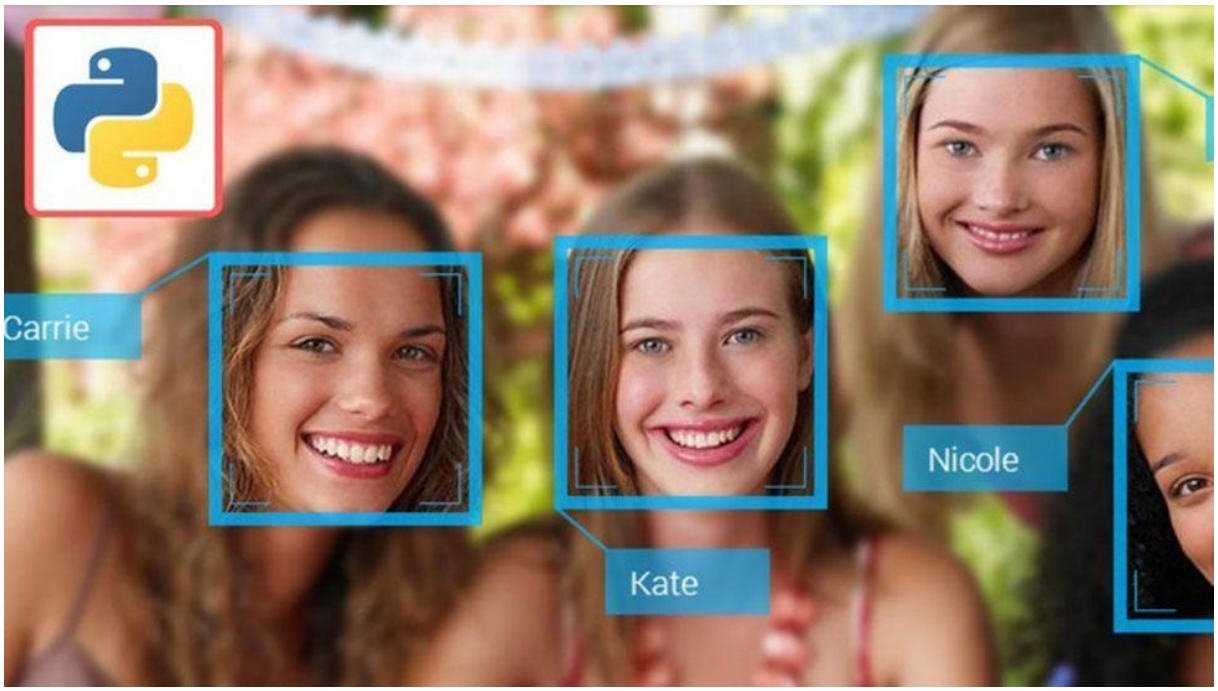
CHƯƠNG 1: TỔNG QUAN

1.1. Giới thiệu

Từ xưa đến nay con người chúng ta luôn luôn quan sát và thu thập thông tin phần lớn thông qua hình ảnh. Vì thế công nghệ xử lí ảnh ra đời có vai trò vô cùng quan trọng trong nền văn minh nhân loại. Nó xử lí được chính xác và có thể phát hiện được những cấu trúc nhỏ mà con người không thể quan sát được. Đã có nhiều phát minh có giá trị to lớn trong hầu hết lĩnh vực: y tế, quân sự, khoa học, … Điển hình như một số ứng dụng nhận dạng khuôn mặt, nhận dạng các vật thể, xử lí nhiễu, nhận dạng hành động, nhận dạng các loại bệnh của người cũng như của thực vật, …

Hình 1. 1: nhận dạng vật thể

Hình 1. 2: Nhận dạng biển số xe



Hình 1. 3: Nhận diện khuôn mặt

1.2. Vấn đề đặt ra

Nhận biết sớm các vấn đề liên quan đến an toàn giao thông và các tai nạn giao thông thường xuyên xãy ra do đi không đúng phần đường và làn đường dẫn đến đến các tai nạn thương tâm. Phần lớn để giúp cho CSGT nhận biết được phương tiện giao thông và giám sát các phương tiện đó có đi đúng làn đường hay không? Và giúp cho người dân có thể nhận biết được làn đường đúng để đi để góp phần làm giảm thiểu tai tan giao thông xãy ra và nâng cao ý thức trách nhiệm về văn hoa giao thông. Từ vấn đề trên, chúng em quyết định ứng dụng xử lý ảnh và trí tuệ nhân tạo dùng để có thể nhận dạng được các phương tiện giao thông và phân chia lối đi hợp lý.

1.3. Mục tiêu đặt ra

- Sử dụng thành thạo được ngôn ngữ lập trình Python.
- Áp dụng được các lý thuyết từ môn học.
- Thiết kế được giao diện cho khách hàng dễ sử dụng.
- Áp dụng trí tuệ nhân tạo và xử lý ảnh vào việc nhận dạng phương tiện giao thông.

1.4. Phương pháp nghiên cứu

- Vận dụng các kiến thức đã được học ở môn Xử Lý Ảnh kết hợp với Trí Tuệ Nhân Tạo (AI)
- Tham khảo các dự án có sẵn của các bài viết trong và ngoài nước để tìm ra phương pháp tối ưu cho dự án của nhóm.

1.5. Phạm vi đề tài

- Ứng dụng rộng rãi được trong các lĩnh vực giao thông .
- Giúp phát hiện nhanh chóng và chính xác các phương tiện giao thông và phân làn hợp lý.
- Từ đó có thể giúp giảm thiểu tai nạn giao thông và nâng cao ý thức trách nhiệm về văn hóa giao thông.

CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT

2.1. Lý thuyết xử lý ảnh

Xử lý ảnh (XLA) là đối tượng nghiên cứu của lĩnh vực thị giác máy, là quá trình biến đổi từ một ảnh ban đầu sang một ảnh mới với các đặc tính và tuân theo ý muốn của người sử dụng. Xử lý ảnh có thể gồm quá trình phân tích, phân lớp các đối tượng, làm tăng chất lượng, phân đoạn và tách cạnh, gán nhãn cho vùng hay quá trình biên dịch các thông tin hình ảnh của ảnh. Cũng như xử lý dữ liệu bằng đồ hoạ, xử lý ảnh số là một lĩnh vực của tin học ứng dụng. Xử lý dữ liệu bằng đồ họa đề cập đến những ảnh nhân tạo, các ảnh này được xem xét như là một cấu trúc dữ liệu và được tạo bởi các chương trình. Xử lý ảnh số bao gồm các phương pháp và kỹ thuật biến đổi, để truyền tải hoặc mã hoá các ảnh tự nhiên. Mục đích của xử lý ảnh gồm:

- Biến đổi ảnh làm tăng chất lượng ảnh.
- Tự động nhận dạng ảnh, đoán nhận ảnh, đánh giá các nội dung của ảnh.

Nhận biết và đánh giá các nội dung của ảnh là sự phân tích một hình ảnh thành những phần có ý nghĩa để phân biệt đối tượng này với đối tượng khác, dựa vào đó ta có thể mô tả cấu trúc của hình ảnh ban đầu. Có thể liệt kê một số phương pháp nhận dạng cơ bản như nhận dạng ảnh của các đối tượng trên ảnh, tách cạnh, phân đoạn hình ảnh, … Kỹ thuật này được dùng nhiều trong y học (xử lý tế bào, nhiễm sắc thể), nhận dạng chữ trong văn bản Module tiết diện.

2.2. Các thư viện thường gặp

2.2.1. OpenCV – Xử lý hình ảnh

- OpenCV là một gói mô-đun hình ảnh lý tưởng cho phép bạn đọc và ghi, thay đổi dữ liệu nhiều hình ảnh cùng một lúc.
- Tạo ra thị giác máy tính cho phép bạn xây dựng lại, gián đoạn và thông hiểu môi trường 3D từ môi trường 2D tương ứng của nó.
- OpenCV được xử dùng nhiều trong nhận diện vật thể và hình ảnh được thiết lập trước, chẳng hạn như khuôn mặt, động vật, cây cối, các vật thể di chuyển, etc.

- Bạn cũng có thể lưu và chụp bất kỳ khoảnh khắc nào của video và cũng có thể phân tích các thuộc tính khác nhau của nó như chuyển động, nền, etc.
- OpenCV tương thích với nhiều hệ điều hành như Windows, OS-X, Open BSD và nhiều hệ điều hành khác.

2.2.2. Pillow – Xử lý hình ảnh

- Khi sử dụng Pillow, người dùng có thể mở và lưu hình ảnh, mà còn có thể xử lý đặc điểm của hình ảnh, chẳng hạn như màu sắc, độ mờ, độ sáng, tối, ...
- Pillow hỗ trợ xử lý nhiều tệp hình ảnh khác nhau như PDF, WebP, PCX, PNG, JPEG, GIF, PSD, WebP, PCX, GIF, IM, EPS, ICO, BMP, và còn nhiều hơn thế.
- Với Pillow, bạn có thể dễ dàng tạo ra những ảnh thu nhỏ (thumbnails) cho hình ảnh, những ảnh thu nhỏ này mang hầu hết đặc điểm của hình ảnh và gần như không khác gì ảnh gốc ngoại trừ chúng được thu nhỏ lại.
- Pillow hỗ trợ một bộ sưu tập các bộ lọc hình ảnh như – FIND_EDGES, DETAIL, SMOOTH, BLUR, CONTOUR, SHARPEN, SMOOTH_MORE, etc.

2.2.3. Tkinter

Tkinter là thư viện GUI tiêu chuẩn cho Python. Tkinter trong Python cung cấp một cách nhanh chóng và dễ dàng để tạo các ứng dụng GUI. Tkinter cung cấp giao diện hướng đối tượng cho bộ công cụ Tk GUI.

2.2.4. Flask – Micro web framework

- Flask là một Micro web framework được viết bằng Python, không yêu cầu tool hay thư viện cụ thể nào. “Micro” không có nghĩa là thiếu chức năng mà “Micro” theo triết lý thiết kế là cung cấp một lõi chức năng “súc tích” nhất cho ứng dụng web, nhưng người dùng có thể mở rộng bất cứ lúc nào.
- Flask luôn hỗ trợ các thành phần tiện ích mở rộng cho ứng dụng như tích hợp cơ sở dữ liệu, xác thực biểu mẫu, xử lý upload, các công nghệ xác thực, template, email, RESTful, … chỉ khác nhau là khi nào bạn muốn thì bạn mới đưa vào.

- Người dùng có thể tập trung xây dựng web application ngay từ đầu trong một khoảng thời gian rất ngắn và có thể phát triển quy mô của ứng dụng tùy theo yêu cầu.

2.2.5. Pytorch – sử dụng trong các bài toán về Deep Learning

- Pytorch là framework được phát triển bởi Facebook. Đây là một ông lớn về công nghệ đầu tư rất nhiều nguồn lực cho việc phát triển Trí tuệ nhân tạo. Pytorch được phát triển với giấy phép mã nguồn mở do đó nó tạo được cho mình một cộng đồng rất lớn. Một cộng đồng lớn đồng nghĩa với nhiều tài nguyên để học và các vấn đề của bạn có thể đã có ai đó giải quyết và chia sẻ với cộng đồng.
- Pytorch cùng với Tensorflow và Keras là một trong những framework phổ biến được sử dụng trong các bài toán về Deep Learning hiện nay. Đặc biệt, trong các lĩnh vực nghiên cứu, hầu như các tác giả đều sử dụng pytorch để triển khai bài toán của mình.
- Pytorch cho thấy lợi thế của nó trong lĩnh vực nghiên cứu bởi việc rất dễ dàng để bạn debug và visuallize, ngoài ra nó theo cơ chế Dynamic Graphs cho phép giảm thời gian huấn luyện mô hình.

2.2.6. Matplotlib – Vẽ đồ thị 2D

- Matplotlib có thể tạo ra những đồ thị chất lượng và xuất ra một cách dễ dàng và thuận tiện, hoàn toàn đáp ứng nhu cầu của mọi ngành học. Các đồ thị được tạo ra bằng Matplotlib có sẵn bản sao cứng trên các nền tảng tương tác khác nhau.
- Bạn có thể dùng Matplotlib với nhiều bộ công cụ như Python Scripts, IPython Shells, Jupyter Notebook, và nhiều công cụ khác.
- Một số thư viện của bên thứ ba có thể được tích hợp với các ứng dụng Matplotlib. Chẳng hạn như seaborn, ggplot, và các bộ công cụ chiếu xạ, mapping khác như basemap.
- Ngoài ra, bạn còn có thể theo dõi bất kỳ lỗi nào phát sinh trong quá trình coding, các bản vá mới, đồng thời còn có thể đóng góp các tính năng mới tại GitHub. Đó

là một trang chính thức để nêu ra các vấn đề liên quan đến Matplotlib và cùng giải quyết chúng.

2.2.7. Numpy – Xử lý mảng đa chiều, ma trận

- Numpy là một mô-đun mở rộng mã nguồn mở cho Python, cung cấp các chức năng biên dịch nhanh cho các thao tác toán học và số, thậm chí là với những ma trận và mảng có lượng dữ liệu khổng lồ. Bên cạnh đó các mô-đun cung cấp một thư viện lớn các chức năng toán học cấp cao để hoạt động trên các ma trận và mảng một cách dễ dàng và thuận tiện.
- Numpy cung cấp những masked arrays đồng thời với mảng gốc. Nó cũng đi kèm với các chức năng như thao tác với hình dạng logic, biến đổi Fourier rời rạc, đại số tuyến tính tổng quát, và nhiều hơn nữa.
- Gói mô-đun này cung cấp các công cụ hữu ích để tích hợp với các ngôn ngữ lập trình khác. Chẳng hạn như C, C++, và ngôn ngữ lập trình Fortran.
- Numpy cung cấp các chức năng tương đương với MATLAB. Cả hai đều cho phép người dùng thao tác nhanh hơn.

2.2.8. Module OS

- Module os trong Python cung cấp các chức năng được sử dụng để tương tác với hệ điều hành và cũng có được thông tin liên quan về nó. OS đi theo các Module tiện ích tiêu chuẩn của Python. Module này cung cấp một cách linh động sử dụng chức năng phụ thuộc vào hệ điều hành.
- Module os trong python cho phép chúng ta làm việc với các tập tin và thư mục.

2.2.9. Module Subprocess

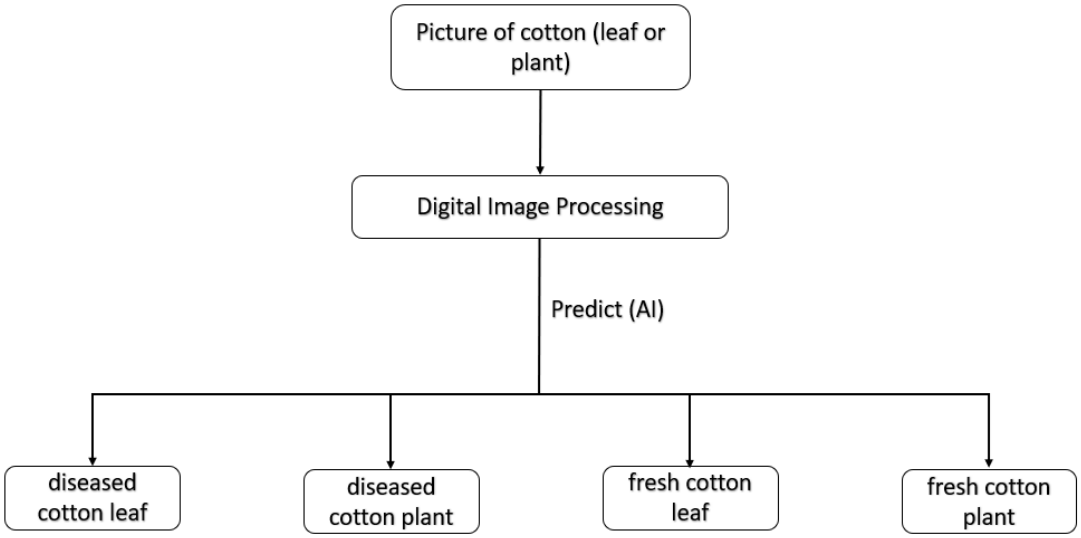
Subprocess trong Python là một mô-đun giúp chạy lệnh hoặc các ứng dụng khác từ chương trình Python và thu về kết quả thực thi. Subprocess còn được gọi là mô-đun quy trình con và được bao gồm trong thư viện chuẩn Python.

CHƯƠNG 3: NỘI DUNG THỰC HIỆN

3.1. Yêu cầu đề tài

Nhận biết bệnh của lá bằng việc ứng dụng xử lý ảnh và AI trực tiếp từ hình ảnh của cây lá cần nhận dạng đưa vào.

3.2. Thiết kế thuật toán



Hình 3. 1: Thuật toán

Ban đầu ta sẽ đưa hình ảnh của cây hoặc lá bông (INPUT) sau đó sẽ xử lý ảnh và nhận dạng bằng AI mà ta đã training. Kết luận sẽ cho ta thấy được và phân tích hình ảnh hiện các phương tiện giao thông và phân làn hợp lý cho từng plan (OUTPUT).

3.3. Viết chương trình thực hiện

3.3.1. Chương trình Training bằng Pytorch

```python
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
train=train_datagen.flow_from_directory('Dataset_MV/Train',
                                        target_size=(64,64),
                                        batch_size=32,
                                        class_mode ='categorical')
test=train_datagen.flow_from_directory('Dataset_MV/Test/',
                                        target_size=(64,64),
                                        batch_size=32,
                                        class_mode ='categorical')
```

Python

```
Found 2127 images belonging to 3 classes.
Found 31 images belonging to 3 classes.
```

```python
import tensorflow as tf
from tensorflow import keras
from keras.models import Sequential
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.layers import Flatten, Dense, Dropout, Activation

model = Sequential()
model.add(Conv2D(32,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same',input_shape=(64,64,3)))
model.add(Conv2D(32,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same'))
model.add(MaxPooling2D((2,2)))
model.add(Dropout(0.2))
model.add(Conv2D(64,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same'))
model.add(Conv2D(64,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same'))
model.add(MaxPooling2D((2,2)))
model.add(Dropout(0.2))
model.add(Conv2D(128,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same'))
model.add(Conv2D(128,(3,3),activation = 'relu',kernel_initializer='he_uniform',padding='same'))
model.add(MaxPooling2D((2,2)))
model.add(Dropout(0.2))
model.add(Flatten())
model.add(Dense(256,activation='relu',kernel_initializer = 'he_uniform'))
model.add(Dropout(0.2))
model.add(Dense(128,activation='relu',kernel_initializer = 'he_uniform'))
model.add(Dropout(0.2))
model.add(Dense(3,activation='softmax'))

model.summary()
```

Python

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 32)        896

 conv2d_1 (Conv2D)           (None, 64, 64, 32)        9248

 max_pooling2d (MaxPooling2D  (None, 32, 32, 32)       0
 )

 dropout (Dropout)           (None, 32, 32, 32)        0

 conv2d_2 (Conv2D)           (None, 32, 32, 64)        18496

 conv2d_3 (Conv2D)           (None, 32, 32, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 16, 16, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 16, 16, 64)        0

 conv2d_4 (Conv2D)           (None, 16, 16, 128)       73856

 conv2d_5 (Conv2D)           (None, 16, 16, 128)       147584
...
Total params: 2,417,699
Trainable params: 2,417,699
Non-trainable params: 0
_________________________________________________________________
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# Thiết lập thông số
from keras.optimizers import Adam
model.compile( loss = 'categorical_crossentropy', optimizer=Adam(), metrics = ['accuracy'])
```

[12] Python

+ Code + Markdown

```python
# Training
history = model.fit_generator( train, validation_data=test, epochs=80, steps_per_epoch=len(train),
                    validation_steps=len(test))
```

[13] Python

+ Code + Markdown | Run All | Clear All Outputs | Outline ··· Select Kernel

```
C:\Users\sangh\AppData\Local\Temp\ipykernel_17660\1306036311.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future ver
  history = model.fit_generator( train, validation_data=test, epochs=80, steps_per_epoch=len(train),
C:\Users\sangh\AppData\Roaming\Python\Python311\site-packages\PIL\Image.py:992: UserWarning: Palette images with Transparency expressed in bytes should
  warnings.warn(
Epoch 1/80
67/67 [==============================] - 24s 350ms/step - loss: 1.1862 - accuracy: 0.4711 - val_loss: 1.2183 - val_accuracy: 0.5484
Epoch 2/80
67/67 [==============================] - 18s 267ms/step - loss: 0.7972 - accuracy: 0.6629 - val_loss: 1.2227 - val_accuracy: 0.5161
Epoch 3/80
67/67 [==============================] - 19s 284ms/step - loss: 0.5959 - accuracy: 0.7781 - val_loss: 0.8625 - val_accuracy: 0.6129
Epoch 4/80
67/67 [==============================] - 22s 328ms/step - loss: 0.4841 - accuracy: 0.8152 - val_loss: 0.8025 - val_accuracy: 0.6452
Epoch 5/80
67/67 [==============================] - 22s 329ms/step - loss: 0.4470 - accuracy: 0.8430 - val_loss: 0.7709 - val_accuracy: 0.6452
Epoch 6/80
67/67 [==============================] - 20s 290ms/step - loss: 0.4002 - accuracy: 0.8514 - val_loss: 0.7935 - val_accuracy: 0.5806
Epoch 7/80
67/67 [==============================] - 24s 364ms/step - loss: 0.4004 - accuracy: 0.8463 - val_loss: 0.5303 - val_accuracy: 0.6452
Epoch 8/80
67/67 [==============================] - 25s 369ms/step - loss: 0.3597 - accuracy: 0.8669 - val_loss: 0.5030 - val_accuracy: 0.7742
Epoch 9/80
67/67 [==============================] - 19s 285ms/step - loss: 0.3297 - accuracy: 0.8778 - val_loss: 0.5010 - val_accuracy: 0.6774
Epoch 10/80
67/67 [==============================] - 17s 258ms/step - loss: 0.2986 - accuracy: 0.8839 - val_loss: 0.2719 - val_accuracy: 0.8387
Epoch 11/80
67/67 [==============================] - 17s 258ms/step - loss: 0.2723 - accuracy: 0.8970 - val_loss: 0.2814 - val_accuracy: 0.9355
Epoch 12/80
67/67 [==============================] - 19s 279ms/step - loss: 0.2638 - accuracy: 0.9069 - val_loss: 0.3102 - val_accuracy: 0.8065
Epoch 13/80
...
Epoch 79/80
67/67 [==============================] - 19s 286ms/step - loss: 0.0453 - accuracy: 0.9864 - val_loss: 7.3073e-04 - val_accuracy: 1.0000
Epoch 80/80
67/67 [==============================] - 18s 273ms/step - loss: 0.0304 - accuracy: 0.9920 - val_loss: 0.0012 - val_accuracy: 1.0000
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
model.save('model.h5')
```

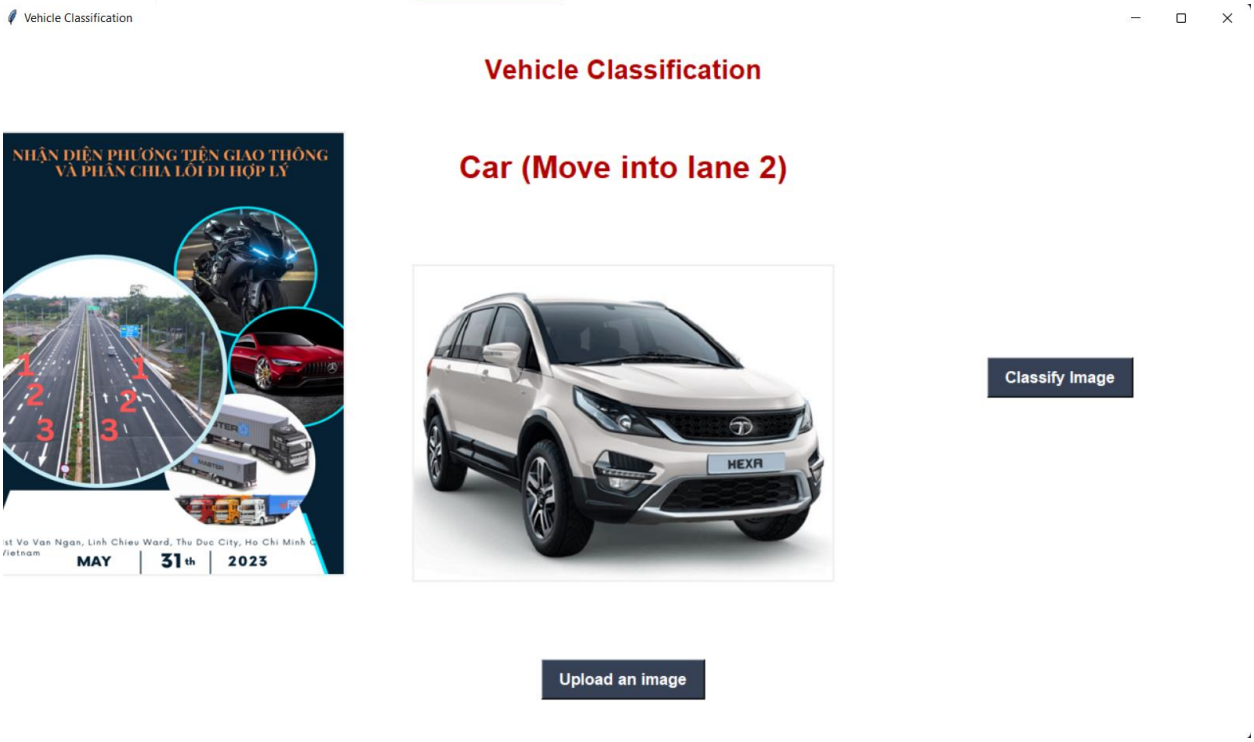
[14] Python

3.3.2. Chương trình nhận dạng và giao diện trên Visual Studio Code

```python
import tkinter as tk
from tkinter import *
from PIL import ImageTk
from PIL import Image,ImageOps
from tkinter import filedialog
import cv2 as cv
from keras.models import load_model
from keras.utils import img_to_array
import numpy as np

np.set_printoptions(suppress=True)

# Load the model
model = load_model(r'model.h5', compile=False)

#Giao diện bìa
def run():
    global anhbia
    anhbia.destroy()
#Tạo cửa sổ giao diện
anhbia = tk.Tk() #Đặt cửa sổ giao diện tên anhbia
anhbia.geometry("720x720")
anhbia.title("While Blood Cell Detection")
# Mở hình ảnh bìa
anh=Image.open("BIA0.png")
# Chỉnh kích thước ảnh
resizeimage=anh.resize((720, 720))
a = ImageTk.PhotoImage(resizeimage)
img=tk.Label(image=a)
img.grid(column=0,row=0)
#Nút nhấn START
Btn=tk.Button(anhbia,text="START",font=("Constantia",20,'bold'),bg= 'green',fg='black',command= run )
Btn.place(x=200,y=620)
anhbia.mainloop() #Lặp lại câu lệnh của anhbia để hiện cửa sổ liên tục = giữ cửa sổ hiển thị

#Set up cái đối tượng trong giao diện chính (trang tiếp theo)
class GUI:
    def __init__(self):
        # Initialise GUI
        self.top = tk.Tk()
        self.top.geometry('1280x720')
        self.top.title('Vehicle Classification')
        self.top.configure(background='#FFFFFF')
        self.label = Label(self.top, background='#FFFFFF', font=('arial', 15, 'bold'))
        self.sign_image = Label(self.top)

        self.upload = Button(self.top, text="Upload an image", command=self.upload_image, padx=12, pady=5)
        self.upload.configure(background='#364156', foreground='white', font=('arial', 13, 'bold'))
        self.upload.pack(side=BOTTOM, pady=40)

        self.sign_image.pack(side=BOTTOM, expand=True)
        self.label.pack(side=BOTTOM, expand=True)
        self.heading = Label(self.top, text="Vehicle Classification", pady=20, font=('arial', 20, 'bold'))
        self.heading.configure(background='#FFFFFF', foreground='#B20000')
        self.heading.pack()


        self.top.mainloop()

    def classify(self, file_path):
        image = Image.open(file_path)
        image = image.resize((64, 64))
        image = img_to_array(image)
        test_image = np.expand_dims(image, axis=0)
        result = (model.predict(test_image) > 0.5).astype("int32")
        x = 0
        c = 0
        i = 0
        while i < 3:
```

```
            if result[0][i] >= x:
                x = result[0][i]
                c = i
            i += 1
        if x <= 0 and c >= 2:
            c = 3
        if c == 0:
            prediction = 'Bike (Move into lane 1)'
        elif c == 1:
            prediction = 'Car (Move into lane 2)'
        elif c == 2:
            prediction = 'Container (Move into lane 3)'

        self.label.configure(foreground='#B20000', text=prediction, font=('arial', 24, 'bold'))

    def show_classify_button(self, file_path):
        classify_b = Button(self.top, text="Classify Image", command=lambda: self.classify(file_path), padx=12, pady=5)
        classify_b.configure(background='#364156', foreground='white', font=('arial', 13, 'bold'))
        classify_b.place(relx=0.79, rely=0.46)

    def upload_image(self):
        try:
            file_path = filedialog.askopenfilename()
            uploaded = Image.open(file_path)
            uploaded.thumbnail(((self.top.winfo_width() / 2.25), (self.top.winfo_height() / 2.25)))
            im = ImageTk.PhotoImage(uploaded)
            self.sign_image.configure(image=im)
            self.sign_image.image = im
            self.label.configure(text='')
            self.show_classify_button(file_path)
        except:
            pass

mainObj = GUI()
```

3.3.3. Kết quả giao diện



CHƯƠNG 4: NHẬN XÉT VÀ ĐÁNH GIÁ KẾT QUẢ

4.1. Kết quả

- Xây dựng được một chương trình có thể nhận dạng được bệnh của lá bằng việc ứng dụng Xử Lý Ảnh và AI.
- Khả năng ứng dụng vào thực tiễn cao.
- Giao diện dễ sử dụng với người dùng.
- Hiểu rõ được các thuật toán của chương trình.
- Chương trình hoạt động mượt mà, ổn định.

4.2. Hạn chế

Độ thông minh của AI tùy thuộc vào dữ liệu dataset mà chúng ta có được nên đôi khi còn sai sót và chưa được chính xác.

4.3. Hướng phát triển

Từ đề tài này chúng ta có thể phát triển thêm được bằng cách nhận dạng them biển số xe và thông tin của tài xế lái xe để dễ dàng kiểm soát trong việc điều tra và xử lý vi phạm (nếu có).

4.4. Kết luận

Việc phát hiện được các phương tiện giao thông và phân chia lối đi hợp lý cho từng phương tiện góp phần không nhỏ trong việc giảm thiểu tai nạn giao thông nguyên nhân do đi sai làn đường. Giúp cho CSGT dễ dàng quản lý những tình trạng sai phạm giao thông và giúp cho người dân có hiểu biết về phần làn đường của từng phương tiện nhất là những người ở địa phương khác đi lại khu vực mới đến, hoặc những khu vực có mật độ giao thông phức tạp. Giảm thiểu được con số đáng kể về tai nạn giao thông và hơn hết là nâng cao ý thức người dân về an toàn giao thông và văn hóa giao thông.

- TÀI LIỆU THAM KHẢO

[1] Rafael C. Gonzalez, Richard E. Woods - Digital Image Processing (2008, Prentice Hall).

[2] Learning Image Processing with OpenCV - Gloria Bueno GarcíaOscar, Deniz Suarez, … (2015).

[3] Flask Framework, Link: https://csc.edu.vn/lap-trinh-va-csdl/tin-tuc/kien-thuc-lap-trinh/7-ly-do-ban-nen-chon-Flask-Framework-4130.

[4] Pytorch, Link: https://viblo.asia/p/huong-dan-tat-tan-tat-ve-pytorch-de-lam-cac-bai-toan-ve-ai-YWOZrNkNZQ0.

[5] Thư viện Numpy, Link: https://codelearn.io/sharing/tim-hieu-thu-vien-numpy-trong-python.

[6] Module OS, Link: https://blog.luyencode.net/module-os-trong-python/

[7] Module Subprocess, Link: https://laptrinhcanban.com/python/nhap-mon-lap-trinh-python/dong-goi-chuong-trinh-python/goi-lenh-hoac-ung-dung-ben-ngoai-tu-python/

[8] Matplotlib, Link: https://viblo.asia/p/gioi-thieu-ve-matplotlib-mot-thu-vien-rat-huu-ich-cua-python-dung-de-ve-do-thi-yMnKMN6gZ7P