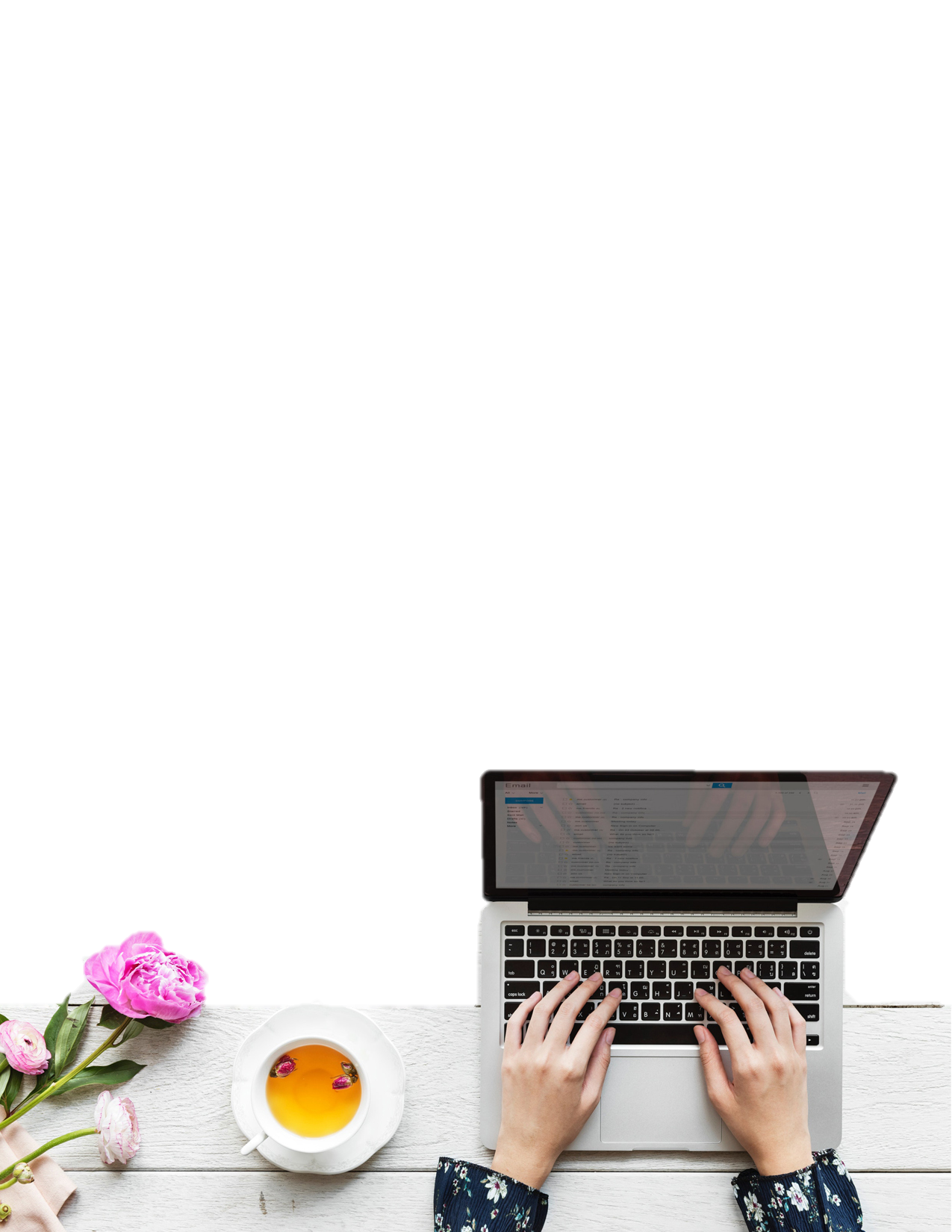
**TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI BÁO CÁO**

**MÔN TRÍ TUỆ NHÂN TẠO**

Chủ đề:

1. Tìm hiểu về công cụ Tensorflow
2. Ứng dụng phát hiện chữ viết tay

Giảng viên hướng dẫn: TS. Nguyễn Đình Hiển

Sinh viên thực hiện:

Dương Tiến – 44.01.104.193

Đoàn Trọng Hải Anh – 44.01.104.049

Lê Thị Kim Hoàng - 43.01.103.018

MỤC LỤC

[I. Lời mở đầu 3](#_Toc56463350)

[1. Lý do chọn đề tài: 3](#_Toc56463351)

[2. Mục tiêu: 3](#_Toc56463352)

[3. Phương pháp thực hiện: 3](#_Toc56463353)

[4. Ứng dụng: 3](#_Toc56463354)

[II. Công cụ Tensorflow 4](#_Toc56463355)

[1. Nền tảng xây dựng Tool TensorFlow và khả năng hoạt động của nó 4](#_Toc56463356)

[a. Nền tảng xây dựng: 4](#_Toc56463357)

[b. Khả năng hoạt động: 4](#_Toc56463358)

[2. Các hỗ trợ từ Tool TensorFlow (hàm, chức năng, cách sử dụng) 5](#_Toc56463359)

[III. Chương trình nhận dạng chữ viết tay 7](#_Toc56463360)

[1. Sử dụng Keras cho bài toán Classification 7](#_Toc56463361)

[2. Sử dụng OpenCV cho bài toán Object Detection 13](#_Toc56463362)

[a. Xử lí ảnh và tìm contours của ảnh đầu vào 13](#_Toc56463363)

[b. Dự đoán các chữ trên ảnh dựa vào model nhận diện chữ viết đã xây dựng 14](#_Toc56463364)

[3. Xây dựng một ứng dụng dựa trên Tool TensorFlow 16](#_Toc56463365)

[IV. Đánh giá chương trình 21](#_Toc56463366)

[V. NGUỒN TÀI LIỆU THAM KHẢO 22](#_Toc56463367)



# Lời mở đầu

## Lý do chọn đề tài:

* Nhằm mục đích hiểu rõ hơn về trí tuệ nhân tạo và cách để ứng dụng nó vào chương trình máy tính:
  + Tìm hiểu về học máy và dùng nó để làm một mô hình dự đoán.
  + Ứng dụng mô hình đó vào chương trình.
* Tại sao lại chọn chữ viết tay?
  + Vì ta đã có sẵn Dataset EMNIST (<https://www.nist.gov/itl/products-and-services/emnist-dataset>)
  + Là một bài toán kinh điển.

## Mục tiêu:

* Hiểu biết hơn về các khái niệm học máy, học sâu, mạng neural,...
* Biết cách xây dựng một mạng neural cơ bản và ứng dụng vào chương trình.

## Phương pháp thực hiện:

* Sử dụng các thư viện có sẵn, đặc biệt là Keras của TensorFlow để tạo và huấn luyện một mô hình dự đoán chữ viết tay bằng Dataset EMNIST.
* Sử dụng thư viện OpenCV cho bài toán nhận dạng chữ viết.

## Ứng dụng:

* Có thể nói ứng dụng cơ bản là biến chữ viết tay thành 🡪 chữ print.



# Công cụ Tensorflow

## Nền tảng xây dựng Tool TensorFlow và khả năng hoạt động của nó

### Nền tảng xây dựng:

* TensorFlow là một nền tảng mã nguồn mở end-to-end dành cho máy học. Nó có một hệ sinh thái toàn diện, linh hoạt gồm các công cụ, thư viện và tài nguyên cộng đồng cho phép các nhà nghiên cứu đẩy mạnh tính năng hiện đại trong ML và các nhà phát triển dễ dàng xây dựng và triển khai các ứng dụng hỗ trợ ML.
  + Xây dựng mô hình dễ dàng: Dễ dàng xây dựng và đào tạo các mô hình ML bằng cách sử dụng các API cấp cao trực quan như Keras với khả năng thực thi nhanh chóng, giúp lặp lại mô hình ngay lập tức và gỡ lỗi dễ dàng.
  + Sản xuất ML mạnh mẽ ở mọi nơi: Dễ dàng xây dựng và đào tạo các mô hình ML bằng cách sử dụng các API cấp cao trực quan như Keras với khả năng thực thi nhanh chóng, giúp lặp lại mô hình ngay lập tức và gỡ lỗi dễ dàng.
  + Thử nghiệm mạnh mẽ để nghiên cứu: Một kiến ​​trúc đơn giản và linh hoạt để đưa các ý tưởng mới từ khái niệm đến mã, đến các mô hình hiện đại và xuất bản nhanh hơn.

(Machine learning (viết tắt là ML): máy có khả năng thích nghi với các điều kiện môi trường xung quanh để rút trích ra các nguyên lý từ tri thức thu nhận được phục vụ cho việc ra quyết định.)

* ***Tensorflow là gì?*** – Với sự bùng nổ của lĩnh vực [Trí Tuệ Nhân Tạo – A.I.](https://topdev.vn/blog/20-tai-lieu-python-hoc-thiet-thuc-nhat-de-tro-thanh-lap-trinh-vien-chuyen-nghiep/) trong thập kỷ vừa qua, machine learning và deep learning rõ ràng cũng phát triển theo cùng. Và ở thời điểm hiện tại, [TensorFlow](https://www.tensorflow.org/) chính là thư viện mã nguồn mở cho machine learning nổi tiếng nhất thế giới, được phát triển bởi các nhà nghiên cứu từ Google. Việc hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning đã giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều.
* ***TensorFlow*** được viết bằng C++ và thao tác interface bằng Python nên phần performance của TensorFlow cực kỳ tốt. Đối tượng sử dụng nó cũng đa dạng không kém: từ các nhà nghiên cứu, nhà khoa học dữ liệu và dĩ nhiên không thể thiếu các lập trình viên.
* TensorFlow 2.0, được ra mắt vào tháng 10 năm 2019, cải tiến framework theo nhiều cách dựa trên phản hồi của người dùng, để dễ dàng và hiệu quả hơn khi làm việc cùng nó (ví dụ: bằng cách sử dụng các Keras API liên quan đơn giản cho việc train model). Train phân tán dễ chạy hơn nhờ vào API mới và sự hỗ trợ cho TensorFlow Lite cho phép triển khai các mô hình trên khá nhiều nền tảng khác nhau. Tuy nhiên, nếu đã viết code trên các phiên bản trước đó của TensorFlow thì bạn phải viết lại, đôi lúc 1 ít, đôi lúc cũng khá đáng kể, để tận dụng tối đa các tính năng mới của TensorFlow 2.0.

### Khả năng hoạt động:

* Các hàm được dựng sẵn trong thư viện cho từng bài toán cho phép TensorFlow xây dựng được nhiều neural network. Nó còn cho phép bạn tính toán song song trên nhiều máy tính khác nhau, thậm chí trên nhiều CPU, GPU trong cùng 1 máy hay tạo ra các dataflow graph – đồ thị luồng dữ liệu để dựng nên các model. Nếu bạn muốn chọn con đường sự nghiệp trong lĩnh vực A.I. này, nắm rõ những điều cơ bản của TensorFlow thực sự rất quan trọng.
* TensorFlow cho phép các lập trình viên tạo ra dataflow graph, cấu trúc mô tả làm thế nào dữ liệu có thể di chuyển qua 1 biểu đồ, hay 1 sê-ri các node đang xử lý. Mỗi node trong đồ thị đại diện 1 operation toán học, và mỗi kết nối hay edge giữa các node là 1 mảng dữ liệu đa chiều, hay còn được gọi là ‘tensor’.
* TensorFlow cung cấp tất cả những điều này cho lập trình viên theo phương thức của ngôn ngữ Python. Vì Python khá dễ học và làm việc, ngoài ra còn cung cấp nhiều cách tiện lợi để ta hiểu được làm thế nào các high-level abstractions có thể kết hợp cùng nhau. Node và tensor trong TensorFlow là các đối tượng Python, và các ứng dụng TensorFlow bản thân chúng cũng là các ứng dụng Python.
* Các operation toán học thực sự thì không được thi hành bằng Python. Các thư viện biến đổi có sẵn thông qua TensorFlow được viết bằng các binary C++ hiệu suất cao. Python chỉ điều hướng lưu lượng giữa các phần và cung cấp các high-level abstraction lập trình để nối chúng lại với nhau.

## Các hỗ trợ từ Tool TensorFlow (hàm, chức năng, cách sử dụng)

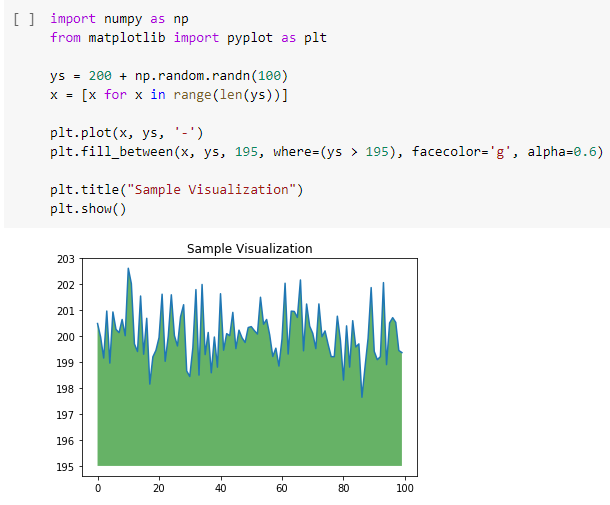
* Colab:

**Colaboratory là gì?**

Colaboratory (gọi tắt là "Colab") cho phép bạn viết và thực thi Python trong trình duyệt với các lợi ích sau:

* Không yêu cầu cấu hình
* Sử dụng miễn phí GPU (Graphics Processing Unit – bộ xử lý đồ họa)
* Chia sẻ dễ dàng

Với Colab, bạn có thể khai thác toàn bộ sức mạnh của các thư viện Python phổ biến để phân tích và trực quan hóa dữ liệu. Ô chứa mã ở bên dưới sử dụng **numpy** để tạo một số dữ liệu ngẫu nhiên và sử dụng **matplotlib** để trực quan hóa dữ liệu đó. Để chỉnh sửa mã này, bạn chỉ cần nhấp vào ô đó và bắt đầu chỉnh sửa.



Với Colab, chỉ cần sử dụng [một vài dòng mã](https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb) là bạn có thể nhập tập dữ liệu hình ảnh, huấn luyện một trình phân loại hình ảnh dựa trên tập dữ liệu đó và đánh giá mô hình này. Sổ tay trên Colab thực thi mã trên các máy chủ đám mây của Google. Nhờ đó, bạn có thể tận dụng sức mạnh của phần cứng Google, bao gồm cả [GPU và TPU](https://colab.research.google.com/notebooks/welcome.ipynb#using-accelerated-hardware), cho dù máy tính của bạn sử dụng sức mạnh phần cứng nào. Bạn chỉ cần một trình duyệt.

Colab được sử dụng rộng rãi trong cộng đồng máy học với các ứng dụng như:

* Bắt đầu sử dụng TensorFlow
* Phát triển và huấn luyện mạng nơron
* Thử nghiệm có sử dụng TPU
* Phổ biến nghiên cứu về AI (trí tuệ nhân tạo)
* Tạo hướng dẫn

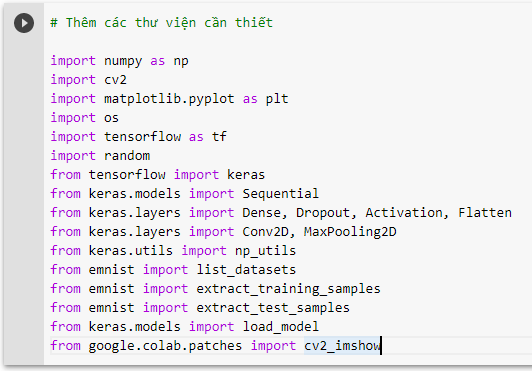
# Chương trình nhận dạng chữ viết tay

## Sử dụng Keras cho bài toán Classification

Bước đầu chúng ta sẽ tải các dữ liệu cài đặt tập ảnh viết tay



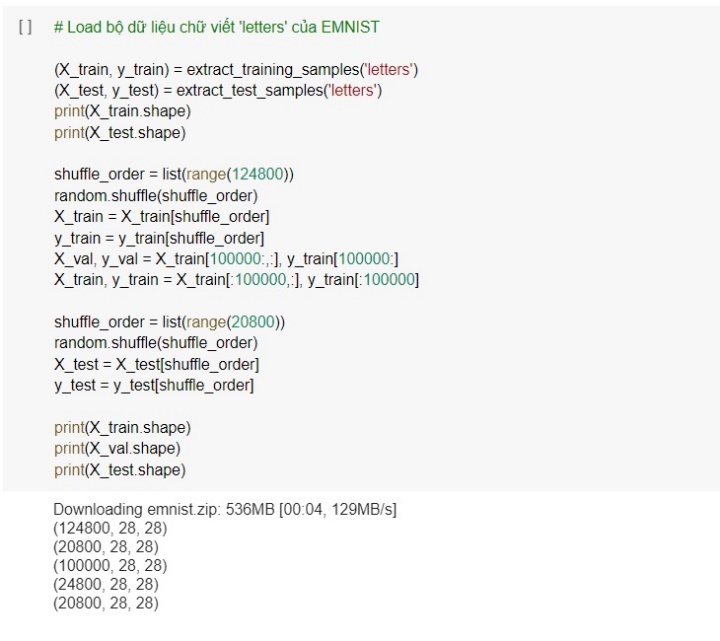
* + 1. Set up: Ở ô đầu tiên, chúng ta sẽ thêm các thư viện cần thiết



Trong đó:

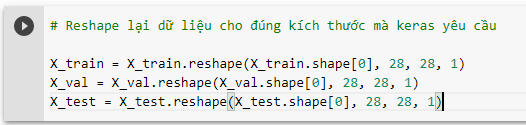
* **keras**: là thư viện chủ yếu được sử dụng để tạo mạng neural và huấn luyện model
* **emnist**: bộ dữ liệu chữ viết tay EMNIST
* **cv2**: OpenCV, được sử dụng để xử lí ảnh
  + 1. Load dữ liệu EMNIST và kiểm tra dữ liệu theo yêu cầu

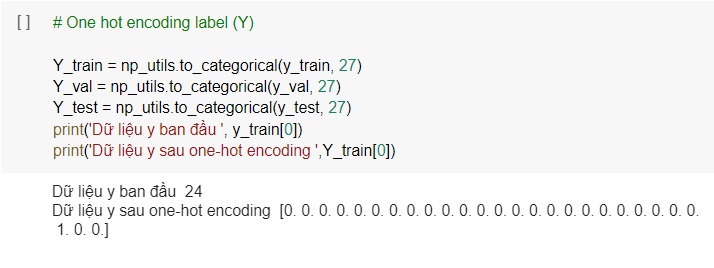
Load dữ liệu từ EMNIST dataset, chọn bộ dữ liệu 'letters' bao gồm 124800 training set và 20800 test set, mỗi ảnh là ảnh xám có kích thước 28x28 pixel. Sau đó chia bộ traning set thành hai: 100000 cho training set và 24800 dữ liệu cho validation set. Training set được dùng để huấn luyện và validation set được dùng để đánh giá độ chính xác của model trong quá trình huấn luyện.



Chúng ta nên kiểm tra lại dữ liệu và thay đổi kích thước phù hợp cho mô-đun đã chọn.

Dữ liệu input cho mô hình convolutional neural network là 1 tensor 4 chiều (N, W, H, D), trong bài này là ảnh xám nên W = H = 28, D = 1, N là số lượng ảnh cho mỗi lần training. Do dữ liệu ảnh ở trên có kích thước là (N, 28, 28) tức là (N, W, H) nên rần reshape lại thành kích thước N 28 28 \* 1 để giống kích thước mà keras yêu cầu.





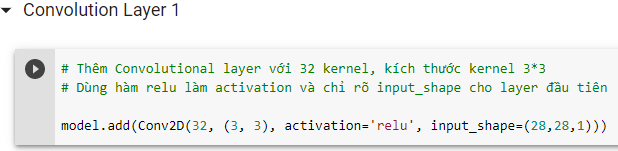
* + 1. Xây dựng một mạng CNN với 5 layers

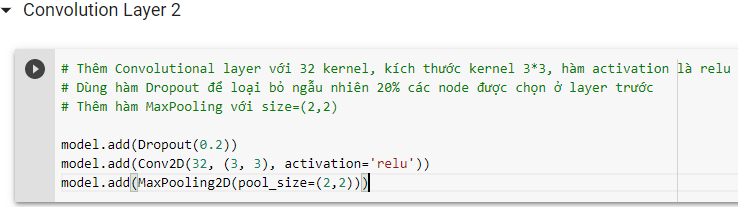
Trong mô hình này, chúng ta sử dụng ở đây gồm 3 lớp Convolution Layer và 2 lớp Fully Conneted Layer nối tiếp nhau

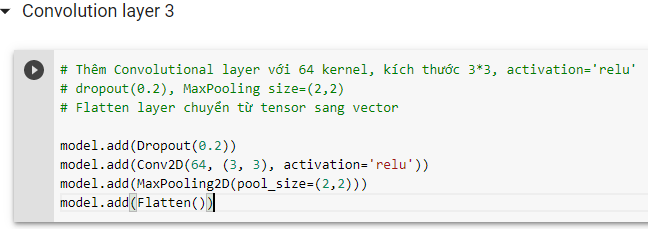
* Bước 1: Định nghĩa model

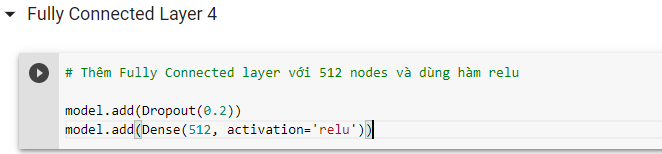


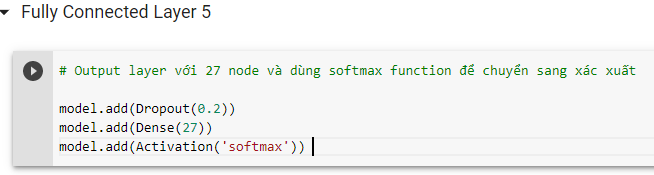
* Bước 2: Tạo các layer

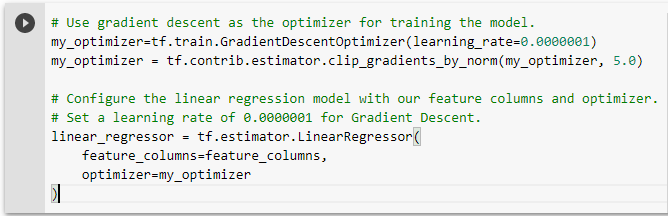










* Bước 3: Biên dịch model 
* Bước 4: Đào tạo mẫu

X\_train, Y\_train là dữ liệu và label cho training set. Tương tự X\_val, Y\_val là dữ liệu cho validation set.

batch\_size: dùng bao nhiêu dữ liệu cho mỗi lần tính và cập nhật hệ số.

một epoch được tính là khi đưa hết tất cả dữ liệu vào mạng neural network 1 lần.

verbose = 1 sẽ hiển thị một thanh tiến trình hoạt động có dạng [=====>.......] để có thể xem tiến trình đào tạo cho mỗi epoch.

* loss: giá trị của hàm mất mát cho dữ liệu đào tạo
* accuracy: độ chính xác cho dữ liệu đào tạo
* val\_loss: giá trị của hàm mất mát cho dữ liệu kiểm tra
* val\_accuracy: độ chính xác cho dữ liệu kiểm tra.

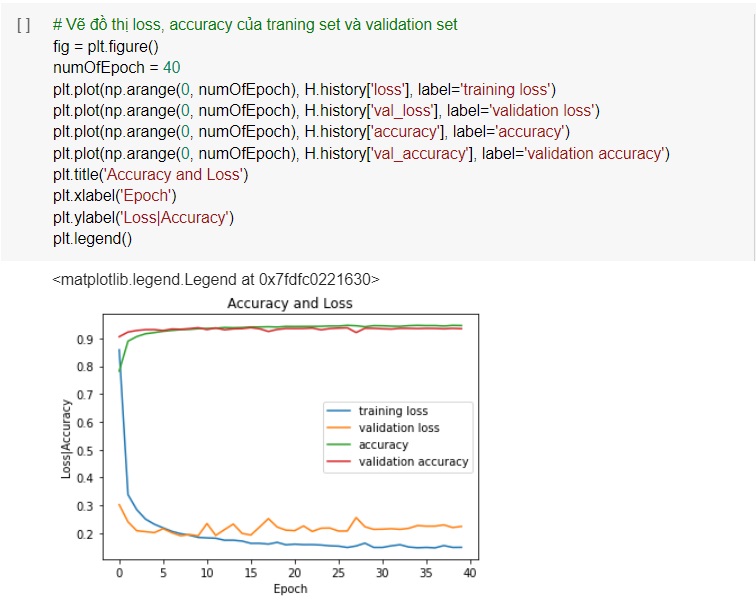
Table

Description automatically generated

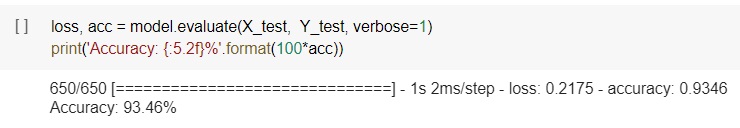
* Bước 5: Lưu model đã huấn luyện



* Bước 6: Vẽ đồ thị loss, accuracy của training set và validation set

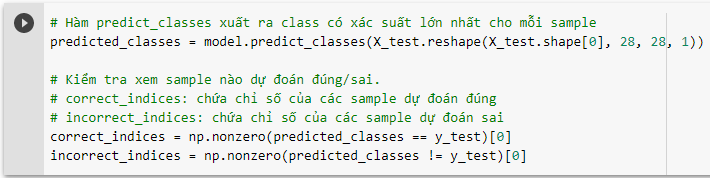


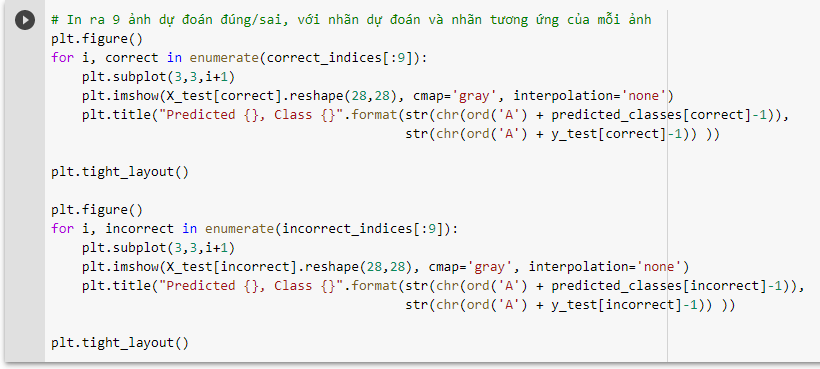
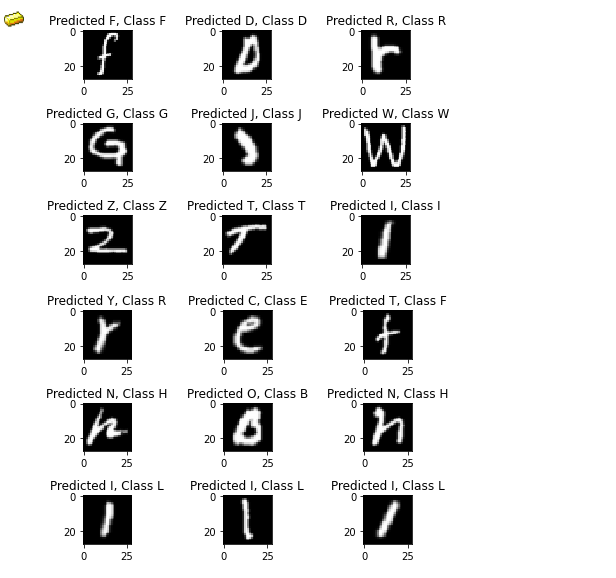
* Bước 7: Đánh giá độ chính xác của model với dữ liệu test set



* Bước 8: Kiểm tra đầu ra

Kiểm tra đầu ra để đảm bảo mọi thứ đều ổn. Ở đây sẽ xem xét về một số ví dụ dự đoán đúng và một số ví dụ dự đoán sai.





## Sử dụng OpenCV cho bài toán Object Detection

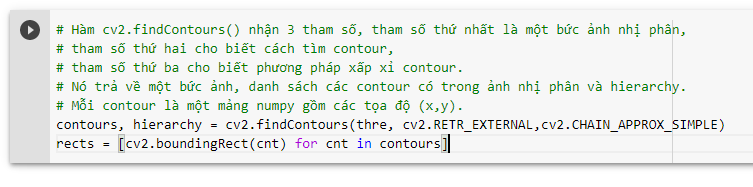
### Xử lí ảnh và tìm contours của ảnh đầu vào

Đến bước này ta sẽ dùng opencv, đầu tiên ta sẽ xử lý ảnh và tìm contours (contour là một đường cong nối liền các điểm có cùng cường độ hoặc màu sắc) của từng chữ viết trên ảnh. Các bước để nhận dạng các chữ viết có trong ảnh:

* Chuyển đổi từ hệ màu BGR sang màu xám
* Giảm nhiễu bằng Gaussian
* Dùng threshold chuyển về ảnh binary
* Tìm contour và vẽ bouding box







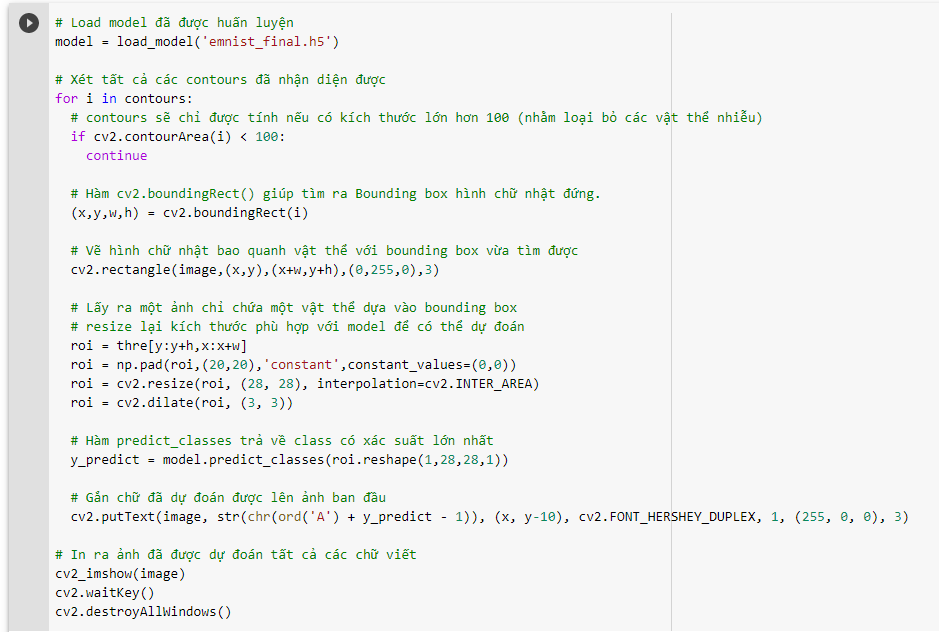
### Dự đoán các chữ trên ảnh dựa vào model nhận diện chữ viết đã xây dựng

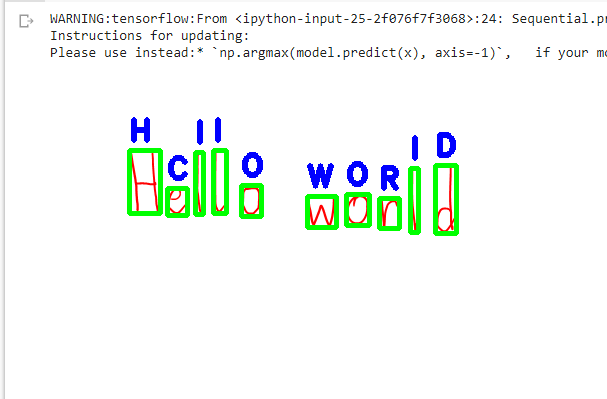
Ta load model đã được huấn luyện vào chương trình sửa dụng hàm **keras.models.load\_model()**. Tiếp đó, xét các contour được tìm thấy:

* Nếu contour bé hơn một kích thước đề ra (ở đây ta để 100) thì loại để khử nhiễu
* Ngược lại ta sử dụng contour đó để vẽ bounding box cho từng đối tượng trong hình.

Sau khi có bounding box, ta dựa vào đó để lấy vị trí và kích thước của vật thể rồi resize lại cho phù hợp với đầu vào của model.

Biến **y\_predict** chính là kết quả dự đoán của model về vật thể được xét. Ta chỉ cần đặt kết quả đó lên trên bounding box và in ra màn hình.





## Xây dựng một ứng dụng dựa trên Tool TensorFlow

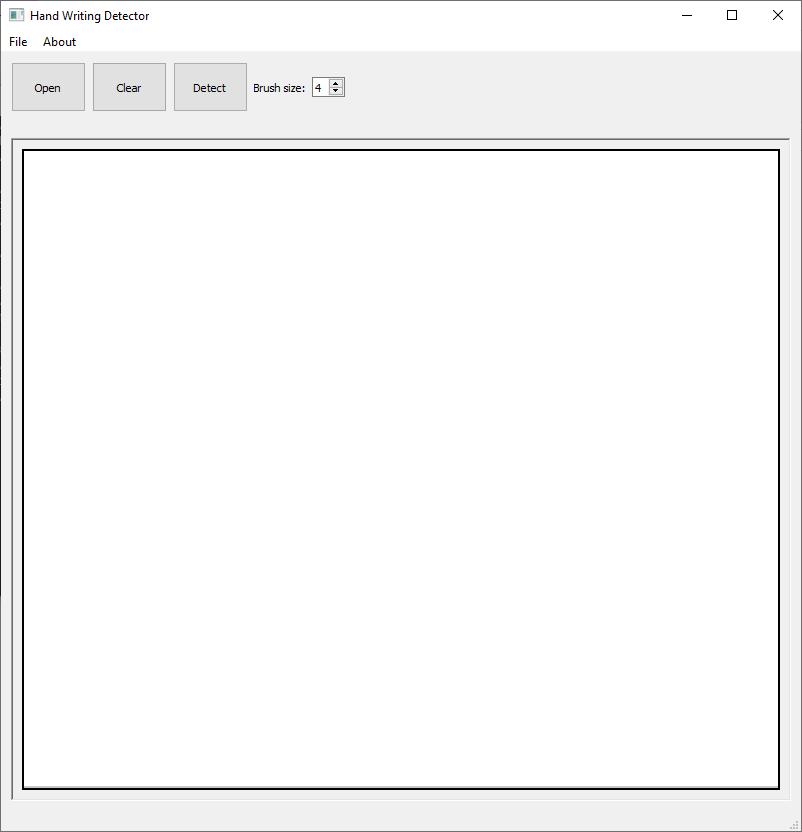
Ứng dụng nhận dạng chữ viết tay với các tính năng

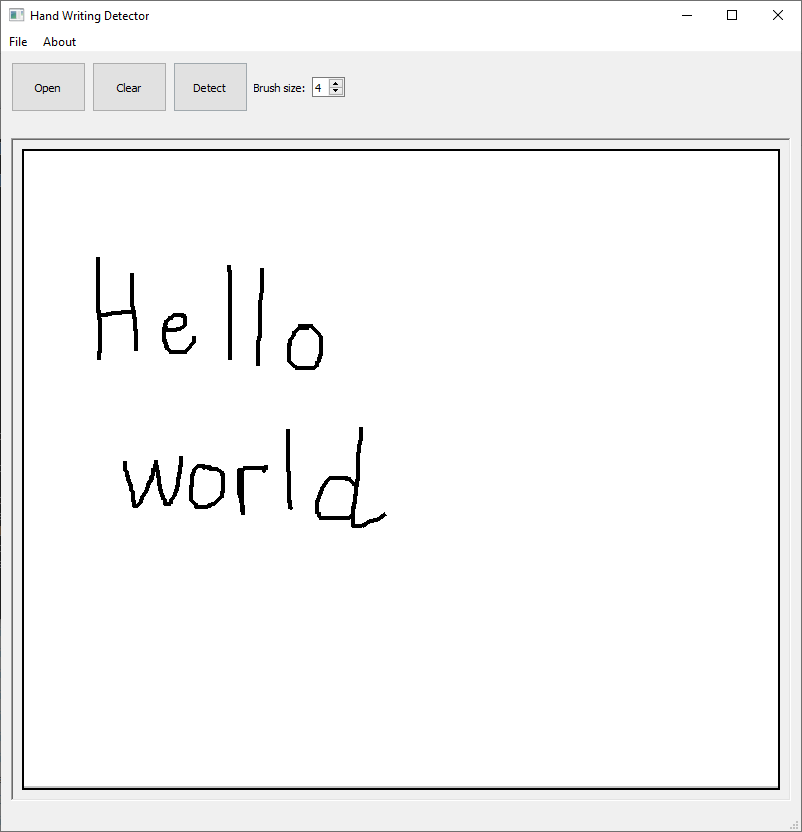
 Vẽ trực tiếp ký tự vào vùng làm việc của ứng dụng

Mở các hình ảnh từ một file ảnh

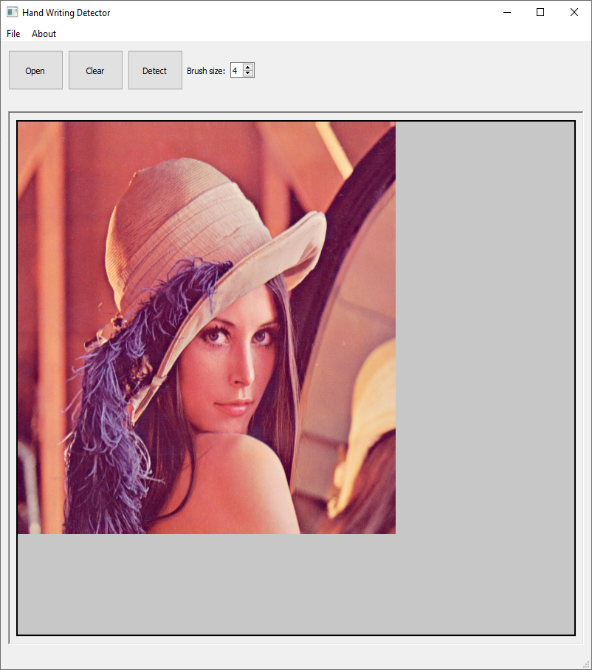
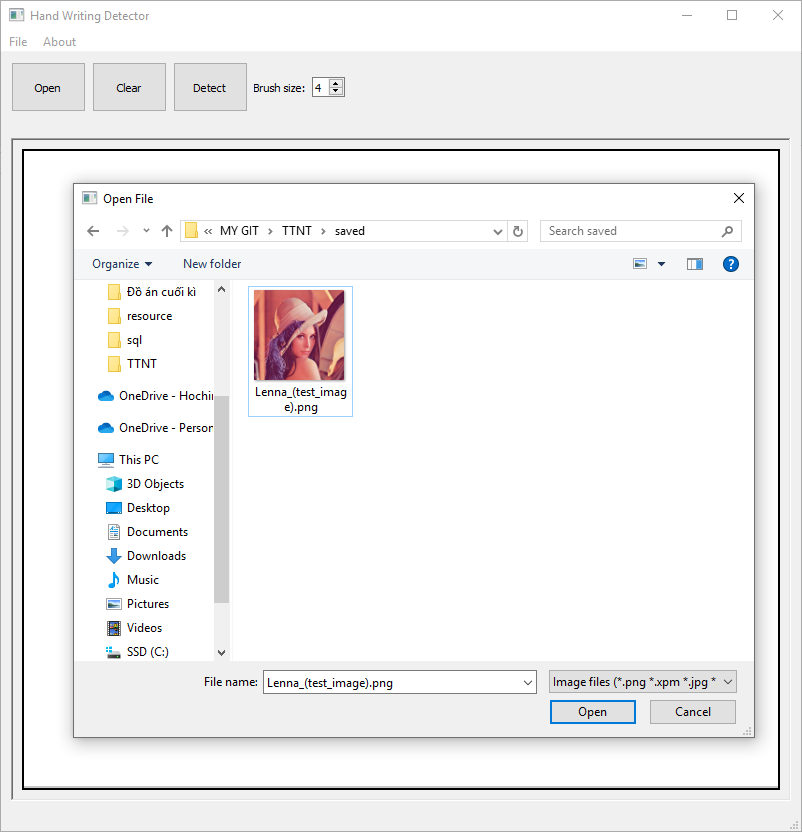
 Xóa các thao tác trước đó trên vùng làm việc của ứng dụng

 Thực thi nhận diện chữ viết tay

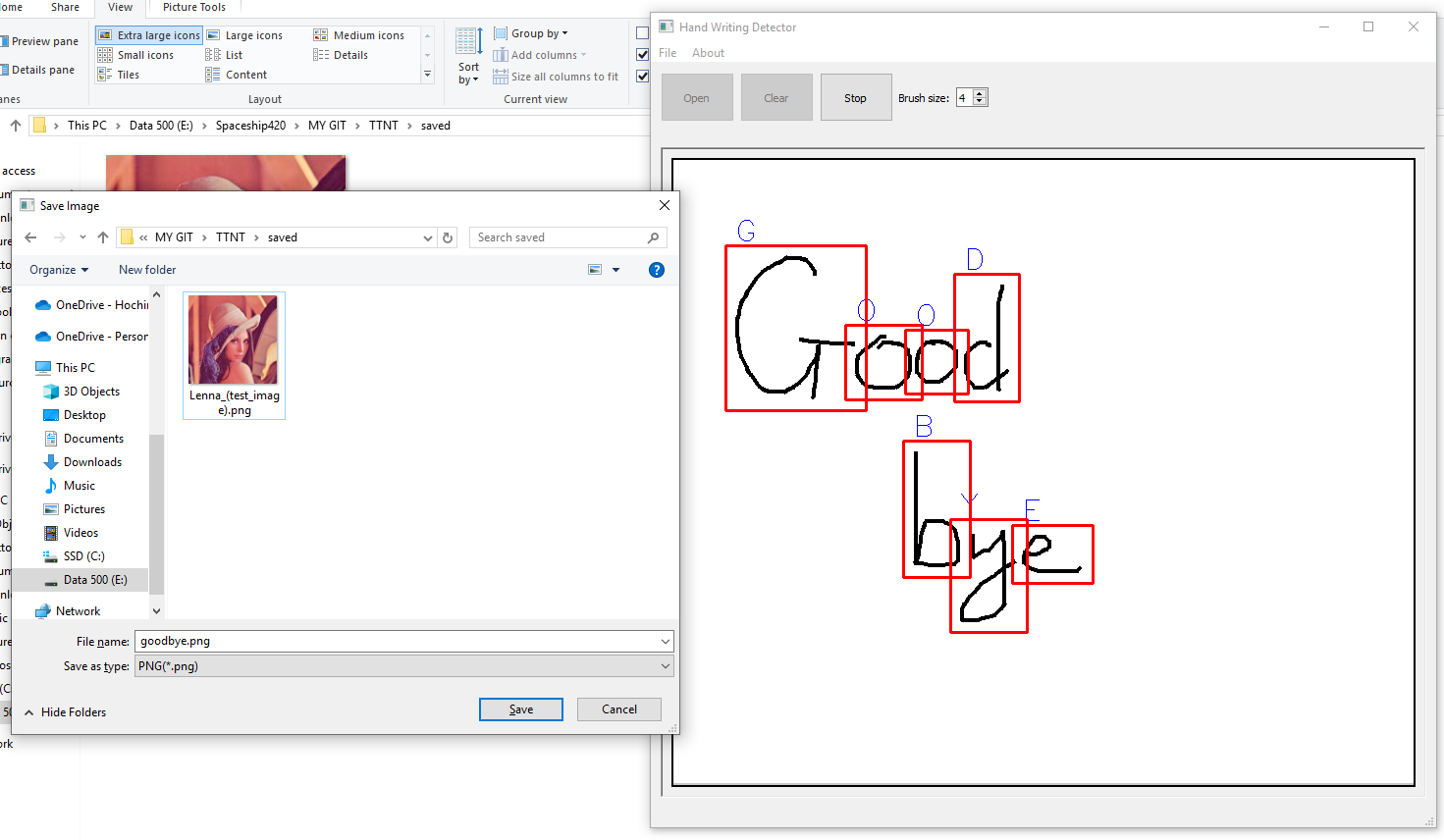
1. Giao diện màn hình chính
2. Thao tác vẽ với chuột trên vùng làm việc

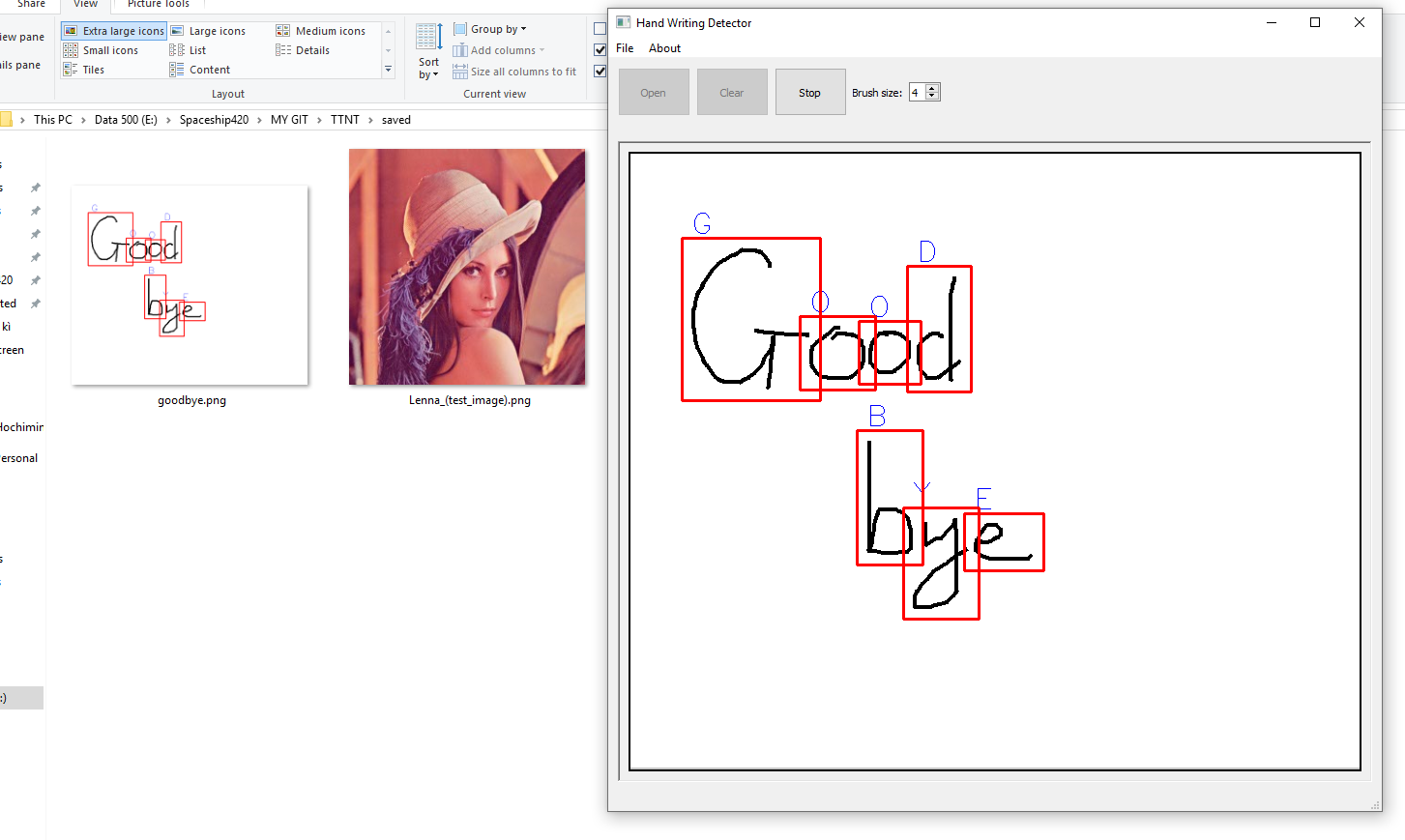


1. Mở tệp từ file .PNG

Kết quả sau khi Open ảnh

1. Lưu kết quả nhận dạng chữ viết tay thành dạng ảnh có file .PNG



Kết quả sau khi thực hiện save một file

# Đánh giá chương trình

Nhìn chung, chương trình có thể nhận dạng với độ chính xác tương đối cao. Tuy nhiên vẫn còn nhiều hạn chế đáng:

* + Nếu ảnh quá nhiễu thì sẽ không nhận dạng được.
  + Hiện chỉ nhận dạng được chữ in thường và chữ in hoa
  + Nếu chữ viết liền nhau thì đó được coi là một đối tượng, gây khó cho việc nhận dạng.

# NGUỒN TÀI LIỆU THAM KHẢO

1. **TensorFlow\_Thư viện code deep learning**

<https://ongxuanhong.wordpress.com/2017/01/06/tensorflow-thu-vien-code-deep-learning/?fbclid=IwAR1051ArBZdWH3-rExTcyXGgL9X97IpBccpWBr2uuCLtb-p_2XVduyE3zYg>

1. **TensorFlow\_Introduce**

<https://www.tensorflow.org/learn?fbclid=IwAR3PjEUZb9HtUBJUAAvP4J8qrGCyAJZTy5kAWDP3HmRPD3VCZ5vMjgnxTKM>

1. **Wikipedia\_TensorFlow**

<https://vi.wikipedia.org/wiki/TensorFlow?fbclid=IwAR3TX6icUgRxzTTtmmoSvW_s4HrcCmckvzMikEWsQ3NlM0tclkVtLBmI5lU>