**MỤC LỤC**

[**CHƯƠNG I: GIỚI THIỆU CHUNG VỀ ĐỀ TÀI** 2](#_Toc56988836)

[1.1. Đặt vấn đề 2](#_Toc56988837)

[1.2. Mục tiêu của đề tài 2](#_Toc56988838)

[1.3. Đối tượng chính của đề tài 2](#_Toc56988839)

[1.4. Nội dung thực hiện 2](#_Toc56988840)

[**CHƯƠNG II: CƠ SỞ LÝ THUYẾT** 2](#_Toc56988841)

[2.1. Khái niệm về xử lý ảnh 2](#_Toc56988842)

[2.2. Nhận dạng ảnh 2](#_Toc56988843)

[2.3. Các quá trình xử lý ảnh 3](#_Toc56988844)

[2.4. Phần mềm LabelImg 5](#_Toc56988845)

[2.5. Phần mềm Anaconda 5](#_Toc56988846)

[2.6. TensorFlow 5](#_Toc56988847)

[2.6.1. TensorFlow là gì? 5](#_Toc56988848)

[2.6.2. Lịch sử ra đời 6](#_Toc56988849)

[2.6.3. Tính năng 6](#_Toc56988850)

[2.6.4. Lợi ích 6](#_Toc56988851)

[2.7. Google Colab 7](#_Toc56988852)

[**CHƯƠNG III. QUY TRÌNH THỰC HIỆN** 8](#_Toc56988853)

[3.1. Thu thập dữ liệu 8](#_Toc56988854)

[3.2. Gán nhãn các ảnh 9](#_Toc56988855)

[3.3. Đào tạo mô hình phát hiện đối tượng bằng google colab 10](#_Toc56988856)

[3.4. Chạy model trên anacoda 21](#_Toc56988857)

[**CHƯƠNG IV: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TRONG TƯƠNG LAI** 29](#_Toc56988858)

[4.1. Kết quả đạt được, những vấn đề còn tồn tại 29](#_Toc56988859)

[4.1.1. Kết quả đạt được 29](#_Toc56988860)

[4.1.2. Vấn đề còn tồn tại 29](#_Toc56988861)

[4.2. Hướng hoàn thiện trong tương lại 29](#_Toc56988862)

# **CHƯƠNG I**

# **GIỚI THIỆU CHUNG VỀ ĐỀ TÀI**

* 1. **Đặt vấn đề**
* Hiện nay, thời đại công nghệ bùng nổ, công nghệ thông tin được sử dụng rộng rãi, việc đưa cộng nghệ AI vào đời sống đang được chú ý, việc các phần mềm nhận dạng thông qua hình ảnh hay webcam được chú ý rất nhiều vì nó có thể áp dụng được trong nhiều lĩnh vực như giáo dục, an toàn bảo mật…
* Trên thế giới có rất nhiều loại thú cưng khác nhau nên rất khó để nhận biết các thú cưng đó, thông qua bài này ta có tiền đề để phát triển ứng dụng nhận dạng các thú cưng mà ta không biết thông qua hình dáng cũng như khuôn mặt của loại thú cưng đó.
  1. **Mục tiêu của đề tài**
* Sử dụng các công cụ như TensorFlow, LabelImg, Anaconda nhận dạng thú cưng qua webcam.
  1. **Đối tượng chính của đề tài**
* Các hình ảnh về thú cưng.
  1. **Nội dung thực hiện**
* Tìm hiểu và thu thập hình ảnh của thú cưng cần nhận diện
* Gán nhãn hình ảnh đã thu thập được bằng phần mềm LabelImg
* Đào tạo mô hình phát hiện đối tượng trên GOOGLE COLAB
* Chạy mô hình đã đào tạo trên ANACONDA

# **CHƯƠNG II CƠ SỞ LÝ THUYẾT**

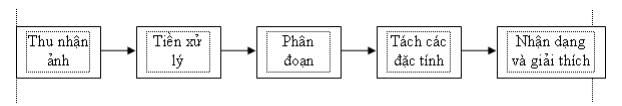
## **2.1. Khái niệm về xử lý ảnh**

* Xử lý ảnh là một phân ngành trong xử lý số tín hiệu với tín hiệu xử lý là ảnh. Đây là một phân ngành khoa học mới rất phát triển trong những năm gần đây. Xử lý ảnh gồm 4 lĩnh vực chính: xử lý nâng cao chất lượng ảnh, nhận dạng ảnh, nén ảnh và truy vấn ảnh. Sự phát triển của xử lý ảnh đem lại rất nhiều lợi ích cho cuộc sống của con người.
* Ngày nay xử lý ảnh đã được áp dụng rất rộng rãi trong đời sống như: photoshop, nén ảnh, nén video, nhận dạng biển số xe, nhận dạng khuôn mặt, nhận dạng chữ viết, xử lý ảnh thiên văn, ảnh y tế,....

**2.2. Nhận dạng ảnh**

* Nhận dạng tự động (automatic recognition), mô tả đối tượng, phân loại và phân nhóm các mẫu là những vấn đề quan trọng trong thị giác máy, được ứng dụng trong nhiều ngành khoa học khác nhau. Tuy nhiên, một câu hỏi đặt ra là: mẫu (pattern) là gì? Watanabe, một trong những người đi đầu trong lĩnh vực này đã định nghĩa: “Ngược lại với hỗn loạn (chaos), mẫu là một thực thể (entity), được xác định một cách ang áng (vaguely defined) và có thể gán cho nó một tên gọi nào đó”. Ví dụ mẫu có thể là ảnh của vân tay, ảnh của một vật nào đó được chụp, một chữ viết, khuôn mặt người hoặc một ký đồ tín hiệu tiếng nói. Khi biết một mẫu nào đó, để nhận dạng hoặc phân loại mẫu đó có thể:
* Hoặc phân loại có mẫu (supervised classification), chẳng hạn phân tích phân biệt (discriminant analyis), trong đó mẫu đầu vào được định danh như một thành phần của một lớp đã xác định.
* Hoặc phân loại không có mẫu (unsupervised classification hay clustering) trong đó các mẫu được gán vào các lớp khác nhau dựa trên một tiêu chuẩn đồng dạng nào đó. Các lớp này cho đến thời điểm phân loại vẫn chưa biết hay chưa được định danh.
* Hệ thống nhận dạng tự động bao gồm ba khâu tương ứng với năm giai đoạn chủ yếu sau đây:
* Thu nhận ảnh
* Tiền xử lý
* Phân đoạn
* Tách các đặc tính
* Nhận dạng và giải thích

## **2.3. Các quá trình xử lý ảnh**



* **Thu nhận ảnh**: Đây là công đoạn đầu tiên mang tính quyết định đối với quá trình XLA. Ảnh đầu vào sẽ được thu nhận qua các thiết bị như camera, sensor, máy scanner,v.v… và sau đó các tín hiệu này sẽ được số hóa. Việc lựa chọn các thiết bị thu nhận ảnh sẽ phụ thuộc vào đặc tính của các đối tượng cần xử lý. Các thông số quan trọng ở bước này là độ phân giải, chất lượng màu, dung lượng bộ nhớ và tốc độ thu nhận ảnh của các thiết bị.
* **Tiền xử lý**: Ở bước này, ảnh sẽ được cải thiện về độ tương phản, khử nhiễu, khử bóng, khử độ lệch,v.v… với mục đích làm cho chất lượng ảnh trở lên tốt hơn nữa, chuẩn bị cho các bước xử lý phức tạp hơn về sau trong quá trình XLA. Quá trình này thường được thực hiện bởi các bộ lọc.
* **Phân đoạn ảnh**: Phân đoạn ảnh là bước then chốt trong XLA. Giai đoạn này phân tích ảnh thành những thành phần có cùng tính chất nào đó dựa theo biên hay các vùng liên thông. Tiêu chuẩn để xác định các vùng liên thông có thể là cùng màu, cùng mức xám v.v… Mục đích của phân đoạn ảnh là để có một miêu tả tổng hợp về nhiều phần tử khác nhau cấu tạo lên ảnh thô. Vì lượng thông tin chứa trong ảnh rất lớn, trong khi đa số các ứng dụng chúng ta chỉ cần trích một vài đặc trƣng nào đó, do vậy cần có một quá trình để giảm lượng thông tin khổng lồ đó. Quá trình này bao gồm phân vùng ảnh và trích chọn đặc tính chủ yếu.
* **Tách các đặc tính**: Kết quả của bước phân đoạn ảnh thường được cho dưới dạng dữ liệu điểm ảnh thô, trong đó hàm chứa biên của một vùng ảnh, hoặc tập hợp tất cả các điểm ảnh thuộc về chính vùng ảnh đó. Trong cả hai trường hợp, sự chuyển đổi dữ liệu thô này thành một dạng thích hợp hơn cho việc xử lý trong máy tính là rất cần thiết. Để chuyển đổi chúng, câu hỏi đầu tiên cần phải trả lời là nên biểu diễn một vùng ảnh dưới dạng biên hay dưới dạng một vùng hoàn chỉnh gồm tất cả những điểm ảnh thuộc về nó. Biểu diễn dạng biên cho một vùng phù hợp với những ứng dụng chỉ quan tâm chủ yếu đến các đặc trưng hình dạng bên ngoài của đối tượng, ví dụ nhƣ các góc cạnh và điểm uốn trên biên chẳng hạn. Biểu diễn dạng vùng lại thích hợp cho những ứng dụng khai thác các tính chất bên trong của đối tượng, ví dụ như vân ảnh hoặc cấu trúc xương của nó. Chúng ta còn phải đưa ra một phương pháp mô tả dữ liệu đã được chuyển đổi đó sao cho những tính chất cần quan tâm đến sẽ được làm nổi bật lên, thuận tiện cho việc xử lý chúng.
* **Nhận dạng và giải thích**: Đây là bước cuối cùng trong quá trình XLA. Nhận dạng ảnh có thể được nhìn nhận một cách đơn giản là việc gán nhãn cho các đối tượng trong ảnh. Giải thích là công đoạn gán nghĩa cho một tập các đối tƣợng đã được nhận biết. Chúng ta cũng có thể thấy rằng, không phải bất kỳ một ứng dụng XLA nào cũng bắt buộc phải tuân theo tất cả các bước xử lý đã nêu ở trên. Một cách tổng quát thì những chức năng xử lý bao gồm cả nhận dạng và giải thích thường chỉ có mặt trong hệ thống phân tích ảnh tự động hoặc bán tự động, được dùng để rút trích ra những thông tin quan trọng từ ảnh.

## **2.4. Phần mềm LabelImg**

* Là công cụ chú thích hình ảnh đồ họa, được viết bằng python và sử dụng Qt cho giao diện đồ họa. Về cơ bản, đây là vẽ các hộp xung quanh đối tượng của bạn. Chương trình nhãn sẽ tạo một tệp xml có chứa thông tin đối tượng trong hình ảnh tức là. Đường dẫn đối tượng, tên đối tượng, tọa độ đối tượng (Xmin, Ymin, Xmax và Ymax) trong hình ảnh. Bạn có thể lưu tệp xml của mình ở định dạng Pascal VOC được sử dụng bởi định dạng Imagenet hoặc YoLo.

## **2.5. Phần mềm Anaconda**

* Anaconda là nền tảng mã nguồn mở về Khoa học dữ liệu trên Python thông dụng nhất hiện nay. Anaconda với hơn 11 triệu người dùng, Anaconda là cách nhanh nhất và dễ nhất để học Khoa học dữ liệu với Python hoặc R trên Windows, Linux và Mac OS X
* Anaconda là một Distribution miễn phí và mã nguồn mở của Python và R.
* Anaconda giúp đơn giản hóa việc cài đặt, quản lý và triển khai packages (numpy, scipy, tensorflow, …).
* Anaconda phục vụ cho nhiều mục địch, đặc biệt trong Data Science (Khoa học dữ liệu), Machine learnig (Máy học), Big Data (Dữ liệu lớn), Image Processing (Xử lý ảnh), …
* Lợi ích của Anaconda:
* Dễ dàng tải 1500+ packages về Python/R cho data science
* Quản lý thư viện, môi trường và dependency giữa các thư viện dễ dàng
* Dễ dàng phát triển mô hình machine learning và deep learning với scikit-learn, tensorflow, keras
* Xử lý dữ liệu tốc độ cao với numpy, pandas
* Hiện thị kết quả với Matplotlib, Bokeh

## **2.6. TensorFlow**

### 2.6.1. TensorFlow là gì?

* TensorFlow là một thư viện phần mềm [mã nguồn mở](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m_ngu%E1%BB%93n_m%E1%BB%9F) dành cho [máy học](https://vi.wikipedia.org/wiki/M%C3%A1y_h%E1%BB%8Dc) trong nhiều loại hình tác vụ nhận thức và hiểu ngôn ngữ. Nó hiện đang được sử dụng cho cả nghiên cứu lẫn sản xuất bởi 50 đội khác nhau trong hàng tá sản phẩm thương mại của Google, như [nhận dạng giọng nói](https://vi.wikipedia.org/wiki/Nh%E1%BA%ADn_d%E1%BA%A1ng_ti%E1%BA%BFng_n%C3%B3i), [Gmail](https://vi.wikipedia.org/wiki/Gmail), [Google Photos](https://vi.wikipedia.org/wiki/Google_Photos), và tìm kiếm, nhiều trong số đó đã từng sử dụng chương trình tiền nhiệm DistBelief của nó. TensorFlow nguyên thủy được phát triển bởi đội Google Brain cho mục đích nghiên cứu và sản xuất của [Google](https://vi.wikipedia.org/wiki/Google) và sau đó được phát hành theo [giấy phép mã nguồn mở Apache 2.0](https://vi.wikipedia.org/wiki/Gi%E1%BA%A5y_ph%C3%A9p_Apache) vào ngày 9/11/2015.

### 2.6.2. Lịch sử ra đời

* Bắt đầu từ năm 2011, [Google Brain](https://vi.wikipedia.org/w/index.php?title=Google_Brain&action=edit&redlink=1) xây dựng DistBelief như là hệ thống [máy học](https://vi.wikipedia.org/wiki/H%E1%BB%8Dc_m%C3%A1y) thế hệ đầu tiên, [độc quyền](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m_s%E1%BB%9F_h%E1%BB%AFu_%C4%91%E1%BB%99c_quy%E1%BB%81n) của mình. Hơn 50 đội tại Google và các công ty [Alphabet](https://vi.wikipedia.org/wiki/Alphabet_Inc.) khác đã triển khai các [mạng nơ-ron](https://vi.wikipedia.org/wiki/M%E1%BA%A1ng_n%C6%A1-ron_nh%C3%A2n_t%E1%BA%A1o) [học sâu](https://vi.wikipedia.org/wiki/H%E1%BB%8Dc_s%C3%A2u) của DistBeliistBelief trong các sản phẩm thương mại của Google, bao gồm [Google Search](https://vi.wikipedia.org/wiki/Google_Search),[Google Voice Search](https://vi.wikipedia.org/w/index.php?title=Google_Voice_Search&action=edit&redlink=1), quảng cáo, [Google Photos](https://vi.wikipedia.org/wiki/Google_Photos), [Google Maps](https://vi.wikipedia.org/wiki/Google_Maps), [Google Street View](https://vi.wikipedia.org/wiki/Google_Street_View), [Google Translate](https://vi.wikipedia.org/wiki/Google_D%E1%BB%8Bch), và [YouTube](https://vi.wikipedia.org/wiki/YouTube). Google đã chỉ định các nhà khoa học máy tính, như tiến sĩ [Geoffrey Hinton](https://vi.wikipedia.org/w/index.php?title=Geoffrey_Hinton&action=edit&redlink=1) và tiến sĩ [Jeff Dean](https://vi.wikipedia.org/w/index.php?title=Jeff_Dean_(computer_scientist)&action=edit&redlink=1), để đơn giản hóa và [cải tiến mã nguồn](https://vi.wikipedia.org/wiki/C%E1%BA%A3i_ti%E1%BA%BFn_m%C3%A3_ngu%E1%BB%93n) codebase của DistBelief để trở thành một thư viện lớp ứng dụng nhanh hơn, mạnh mẽ hơn, mà sau này trở thành TensorFlow. Trong năm 2009, đội ngũ được lãnh đạo bởi Hinton đã có thể giảm số lỗi trong các mạng nơ-ron sử dụng DistBelief một lượng đáng kể; đột phá này đã được thực hiện bởi các đột phá khoa học của Hinton trong [truyền ngược](https://vi.wikipedia.org/w/index.php?title=Backpropagation&action=edit&redlink=1) tổng quát hóa. Đột phá đáng chú ý nhất của Hinton trực tiếp dẫn đến việc giảm lỗi trong phần mềm nhận dạng giọng nói của Google ít nhất là 25%.

**2.6.3. Tính năng**

* TensorFlow cung cấp một [API Python](https://vi.wikipedia.org/w/index.php?title=API_Python&action=edit&redlink=1), cũng như một ít tài liệu [API C/C++](https://vi.wikipedia.org/w/index.php?title=API_C/C%2B%2B&action=edit&redlink=1).

**2.6.4. Lợi ích**

* Lợi ích dễ thấy nhưng quan trọng nhất mà TensorFlow cung cấp cho việc lập trình machine learning chính là abstraction. Thay vì phải đối phó với những tình huống rườm rà từ việc thực hiện triển khai các thuật toán, hay tìm ra cách hợp lý để chuyển output của 1 chức năng sang input của 1 chức năng khác, giờ đây bạn có thể tập trung vào phần logic tổng thể của 1 ứng dụng hơn. TensorFlow sẽ chăm sóc phần còn lại thay cho bạn
* Ngoài ra TensorFlow còn ung cấp các tiện ích bổ sung cho các lập trình viên cần debug cũng như giúp bạn tự suy xét các ứng dụng TensorFlow. Chế độ [eager execution](https://www.tensorflow.org/programmers_guide/eager) cho phép bạn đánh giá và sửa đổi từng operation của biểu đồ 1 cách riêng biệt và minh bạch, thay vì phải dựng toàn bộ biểu đồ dưới dạng 1 đối tượng độc lập vốn khá mơ hồ hay phải đánh giá chung tổng thể. Cuối cùng, 1 tính năng khá độc đáo của TensorFlow là [TensorBoard](https://www.tensorflow.org/programmers_guide/summaries_and_tensorboard). TensorBoard cho phép bạn quan sát 1 cách trực quan những gì TensorFlow đang làm
* TensorFlow còn có nhiều cải tiến từ sự hậu thuẫn từ các ekíp thương mại hạng A tại Google. Google không những tiếp lửa cho tiến độ nhanh chóng cho sự phát triển đằng sau dự án, mà còn tạo ra nhiều phục vụ độc đáo xung quanh TensorFlow để nó dễ dàng deploy và sử dụng: như silicon TPU mình đã nói ở trên để tăng tốc hiệu suất đám mây Google, 1 online hub cho việc chia sẻ các model được tạo với framework, sự hiện diện của [in-browser](https://js.tensorflow.org/) và [gần gũi với mobile](https://www.tensorflow.org/mobile/tflite/) của framework, và [nhiều hơn thế nữa…](https://siliconangle.com/2018/04/02/new-google-investments-tensorflow-now-ais-leading-development-framework/)

**2.7. Google Colab**

* Colaboratory hay còn gọi là Google Colab, là một sản phẩm từ Google Research, nó cho phép chạy các dòng code python thông qua trình duyệt, đặc biệt phù hợp với Data analysis, machine learning và giáo dục. Colab không cần yêu cầu cài đặt hay cấu hình máy tính, mọi thứ có thể chạy thông qua trình duyệt, bạn có thể sử dụng tài nguyên máy tính từ CPU tốc độ cao và cả GPUs và cả TPUs đều được cung cấp cho bạn.
* Colab cung cấp nhiều loại GPU, thường là Nvidia K80s, T4s, P4s and P100s, tuy nhiên người dùng không thể chọn loại GPU trong Colab, GPU trong Colab thay đổi theo thời gian. Vì là dịch vụ miễn phí, nên Colab sẽ có những thứ tự ưu tiên trong việc sử dụng tài nguyên hệ thống, cũng như giới hạn thời gian sử dụng, thời gian sử dụng tối đa lên tới 12 giờ.

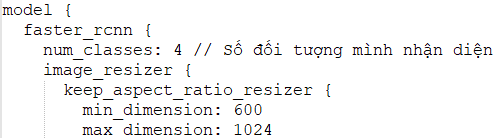
# **CHƯƠNG III**

# **QUY TRÌNH THỰC HIỆN**

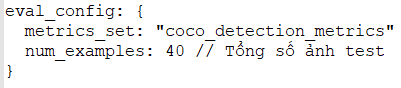
## **3.1. Thu thập dữ liệu**

* Tạo một thư mục images có hai thư mục test và train. Thư mục test có tầm 30% ảnh của thư mục train.
* Tạo các file faster\_rcnn\_inception\_v2\_pets.config, generate\_tfrecord.py, labelmap.pbtxt, train.py (<https://github.com/DataExplained/How-to-train-Custom-Object-Detection-Model-Using-Google-Colab>)
* Trong file faster\_rcnn\_inception\_v2\_pets.config thay đổi:

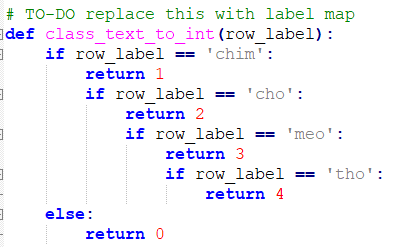
+ num\_classes



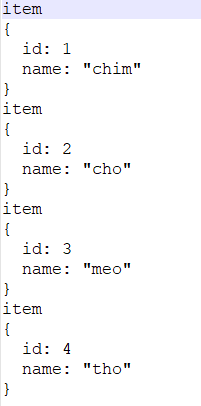
+ num\_examples



* Trong file generate\_tfrecord.py thay thế các nhãn đối tượng



* Trong file labelmap.pbtxt là các đối tượng



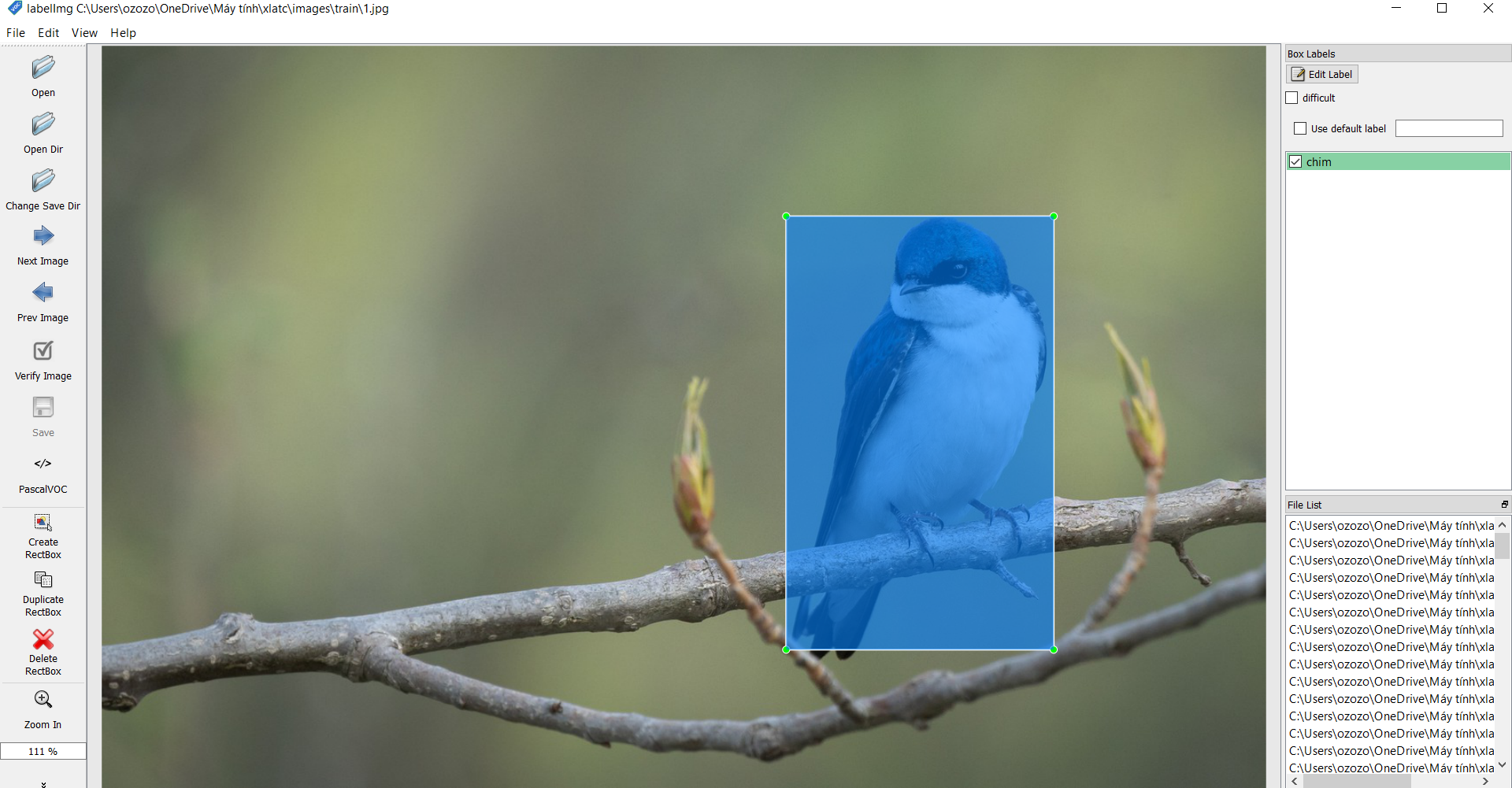
## **3.2. Gán nhãn các ảnh**

* Tôi sẽ giới thiệu LabelImg là công cụ chú thích hình ảnh đồ họa, được viết bằng python và sử dụng Qt cho giao diện đồ họa. Về cơ bản, đây là vẽ các hộp xung quanh đối tượng của bạn. Chương trình nhãn sẽ tạo một tệp xml có chứa thông tin đối tượng trong hình ảnh tức là. Đường dẫn đối tượng, tên đối tượng, tọa độ đối tượng (Xmin, Ymin, Xmax và Ymax) trong hình ảnh. Bạn có thể lưu tệp xml của mình ở định dạng Pascal VOC được sử dụng bởi định dạng Imagenet hoặc YoLo.
* Cài đặt LabelImg (<https://github.com/maihaan/XLA_K62>). Gán nhãn:

+ Mở phần mềm vào View-> Auto Save Mode

+ Vào Change Save Dir chọn chỗ lưu ảnh đã gán nhãn và Open Dir chọn chỗ ảnh bạn đã thu thập.

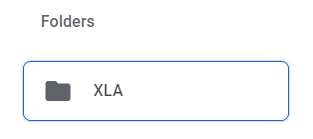
+ Vẽ các hộp xung quanh đối tượng của bạn, ghi tên đối tượng và nhấn Save.



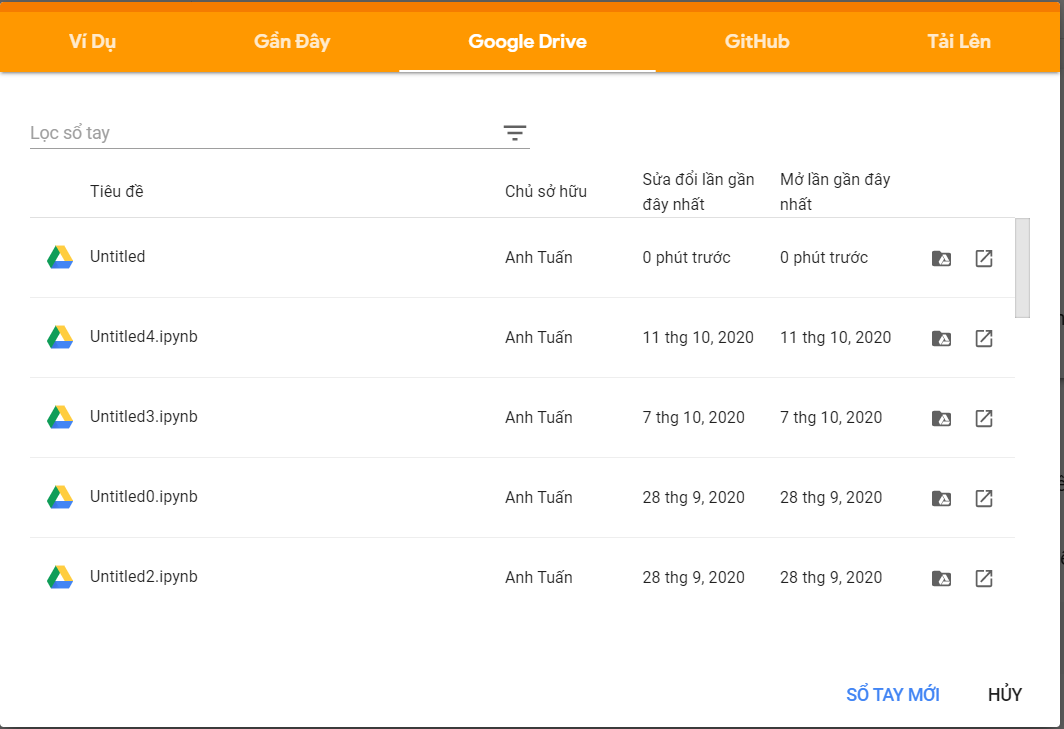
* Gán nhãn xong nén thư mục chứa thư mục images, labelmap.pbtxt, train.py, generate\_tfrecord.py, faster\_rcnn\_inception\_v2\_pets.config. Và tải lên google drive cùng với file Data Explained ModelTrainingOnColab\_final\_faster\_rcnn.ipynb.

## **3.3. Đào tạo mô hình phát hiện đối tượng bằng google colab**

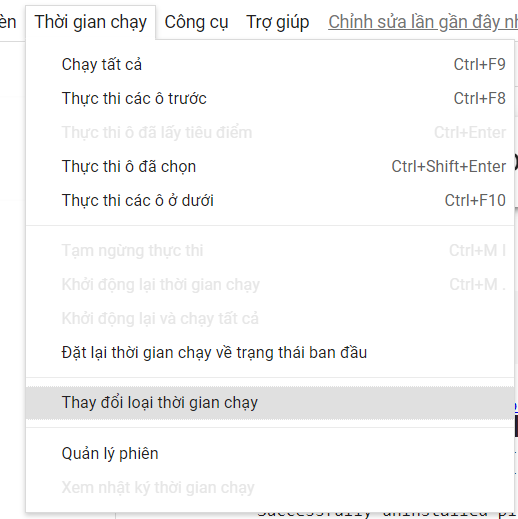
* **Bước 1**: Tạo một thư mục trong google drive của bạn nơi bạn có thể lưu tất cả các tệp cần thiết cho việc đào tạo mô hình.

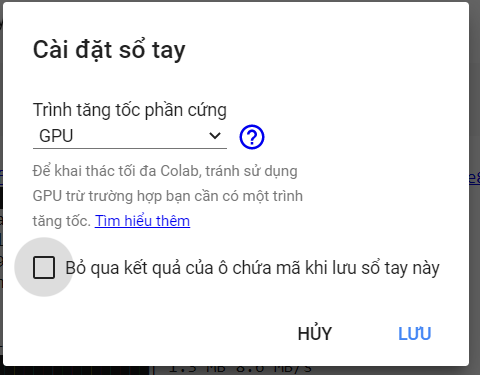


* **Bước 2**: Truy cập Google Colab, đăng nhập bằng cùng một tài khoản Google được sử dụng cho google-drive và tạo một sổ ghi chép mới.



* **Bước 3**: Trong máy tính xách tay, hãy chuyển đến Runtime> Change Runtime Type và đảm bảo chọn GPU làm bộ tăng tốc phần cứng.

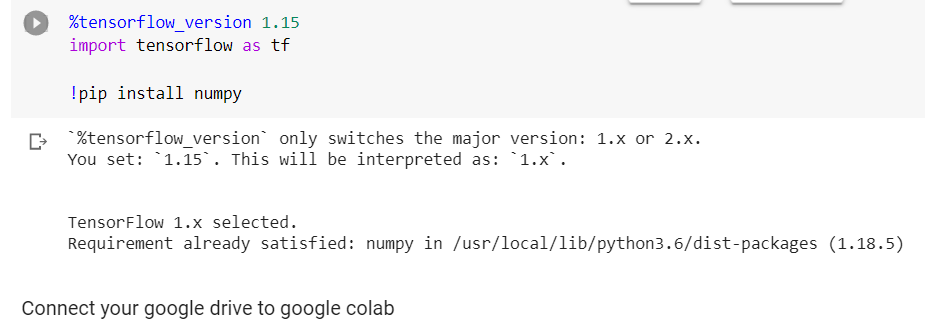




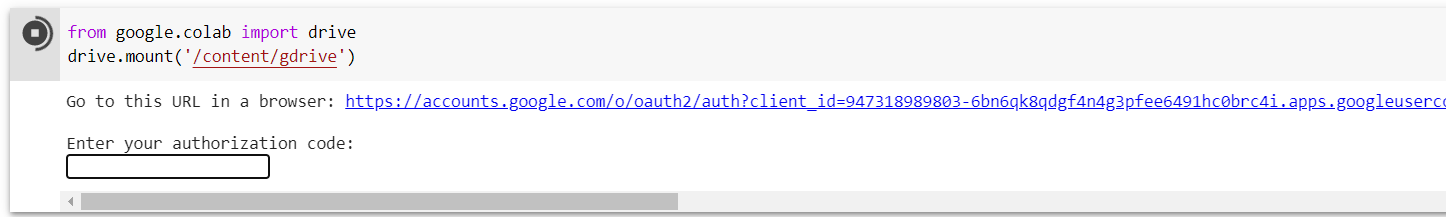
* **Bước 4**: Cài đặt Numpy. Chạy mã bên dưới

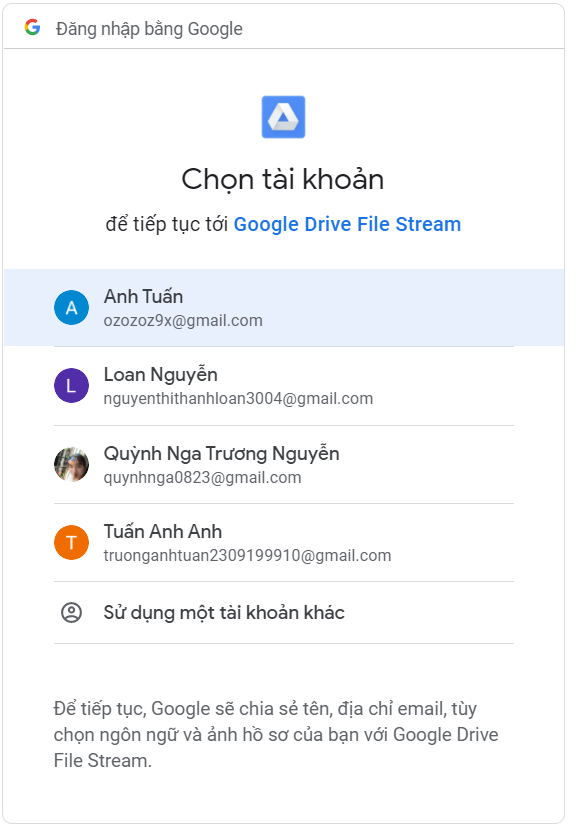


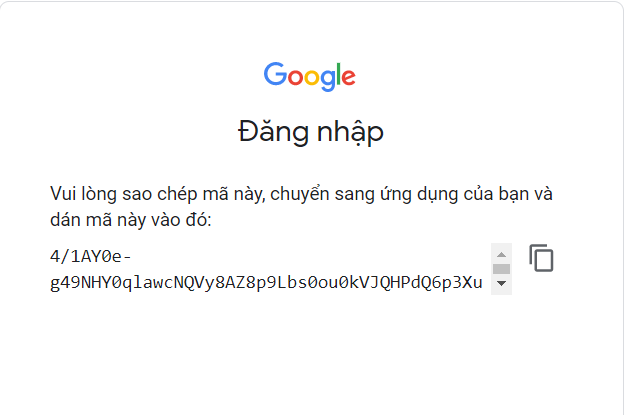
* **Bước 5**: Cài đặt Tensortflow. Chạy mã bên dưới

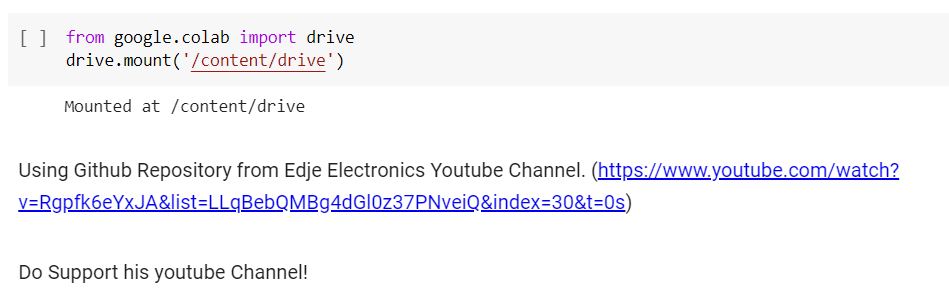


* **Bước 6**: Nhấp vào đường link.



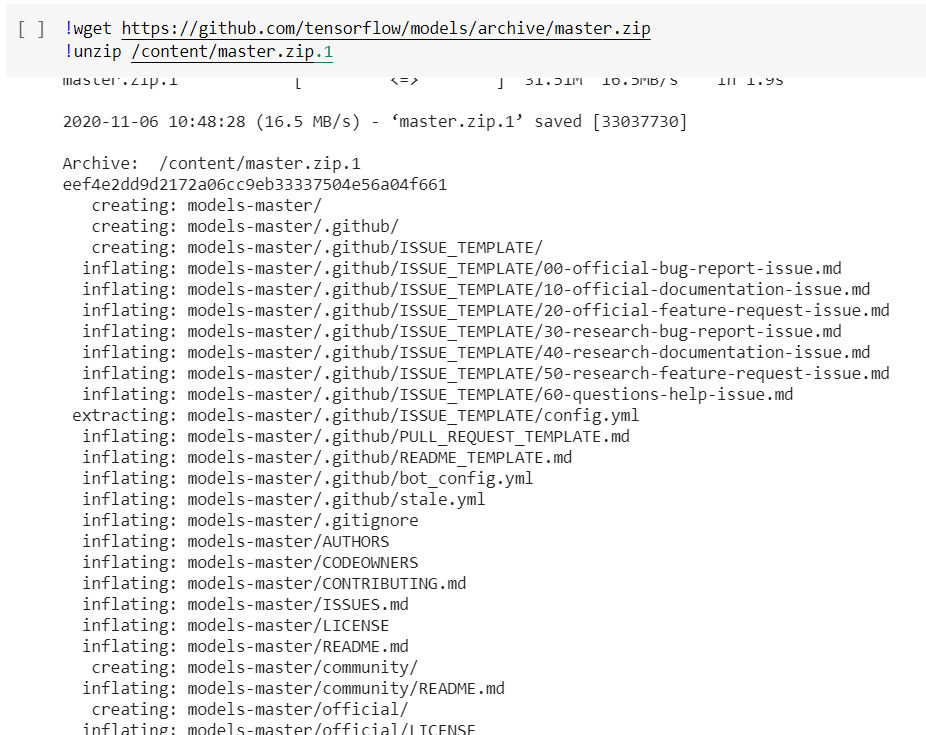
* **Bước 7**: Sau đó đăng nhập bằng tài khoản google drive mà bạn dùng
* **Bước 8**: Sao chép mã trên trang đó và dán vào hộp văn bản của phiên Colab mà bạn đang chạy, sau đó nhấn phím ENTER.



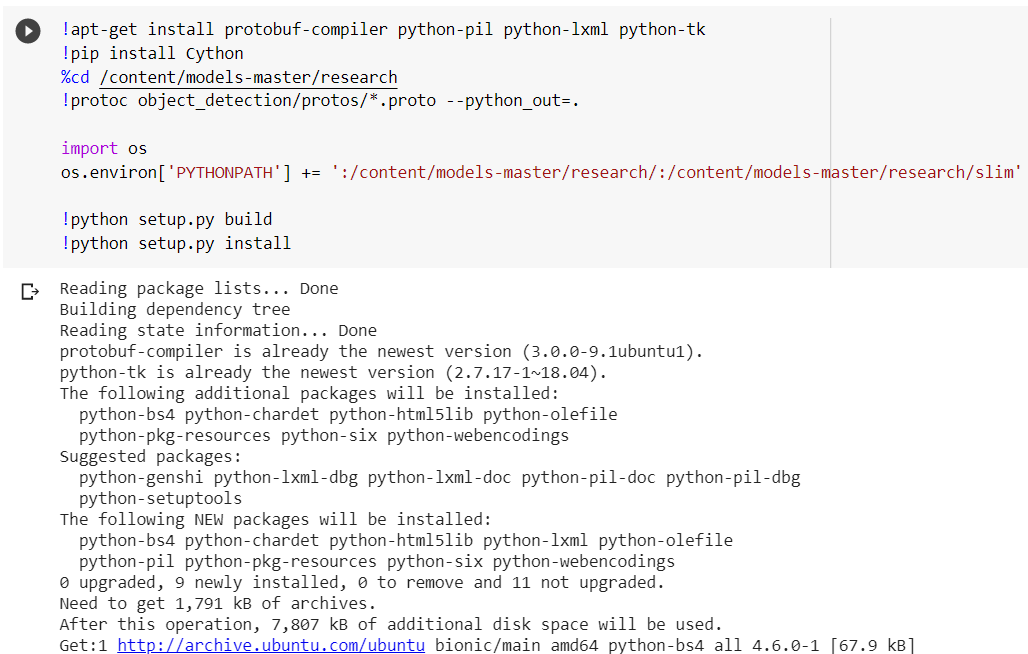


* **Bước 9**: Chạy tiếp các mã bên dưới

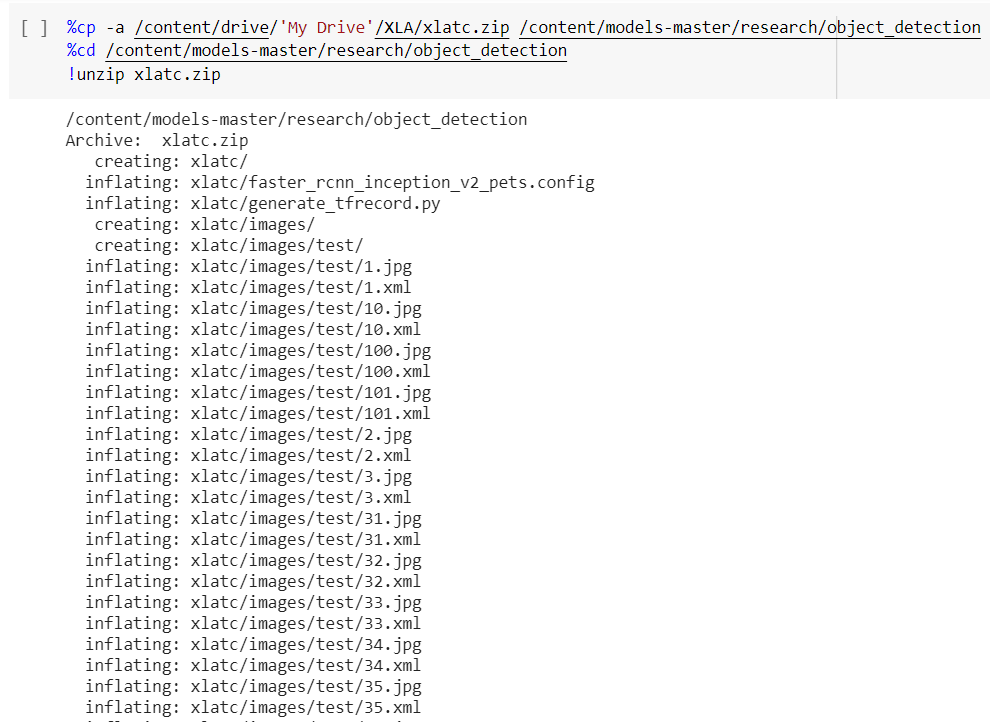


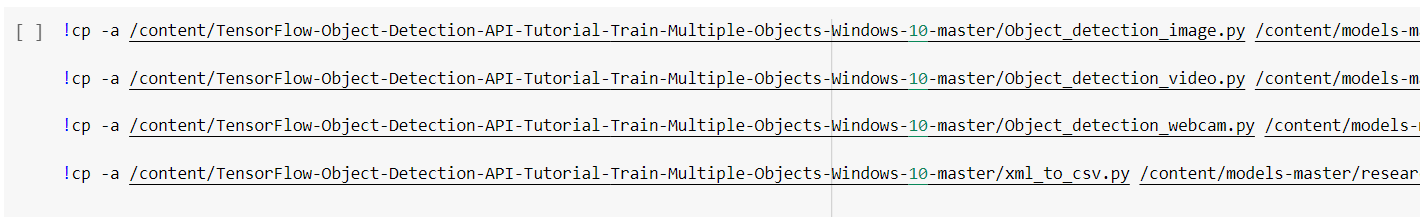




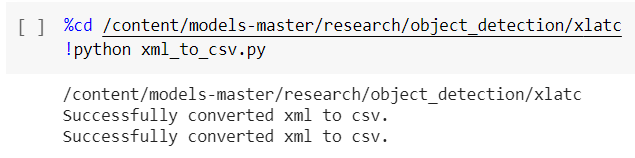


* **Bước 10**: Sửa đến đúng đường dẫn đến thư mục để các file tải lên Google Drive.

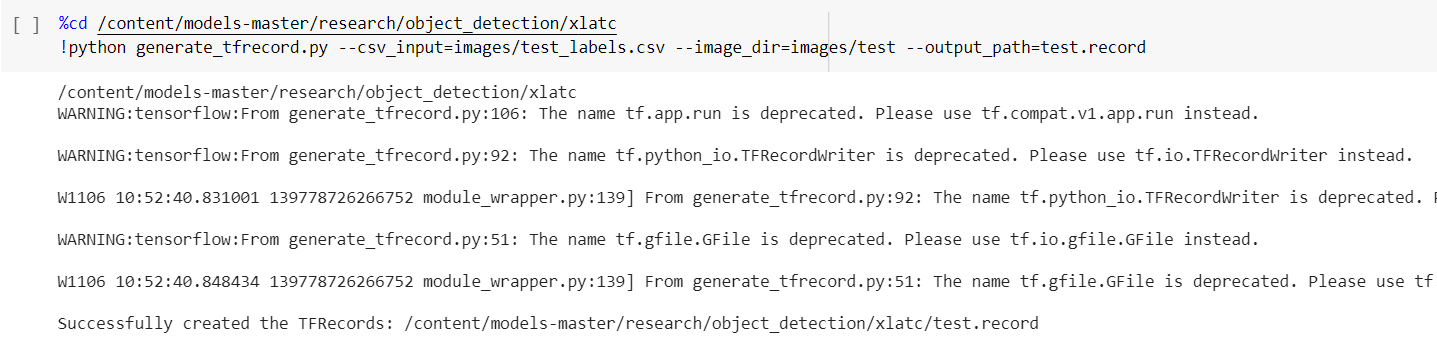




* **Bước 11** : Chuyển images từ xml sang csv



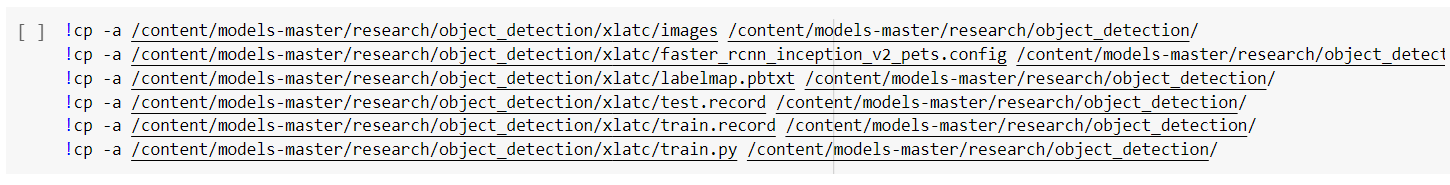
* **Bước 12**: Tạo dữ liệu kiểm tra



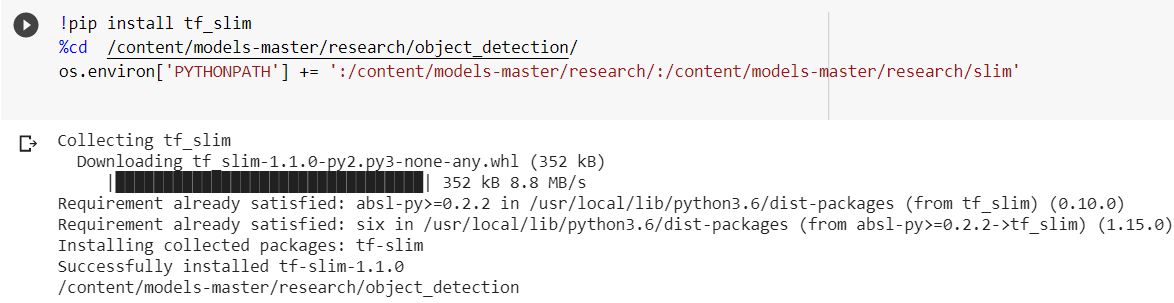
* **Bước 13:** Tạo dữ liệu đào tạo

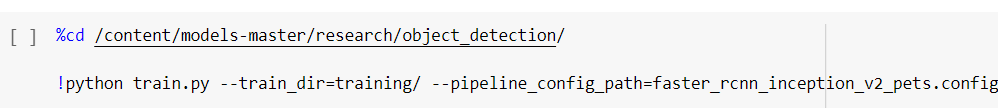


* **Bước 14**: Sửa đúng đường dẫn đến thư mục

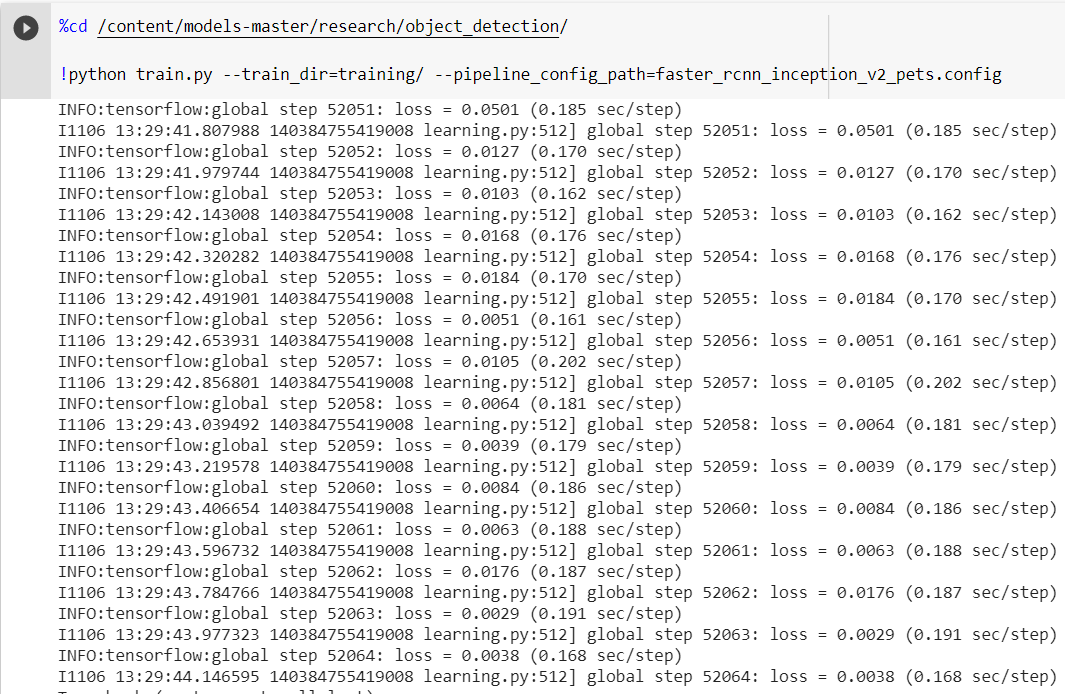


* **Bước 15**: Bắt đầu đào tạo và chạy các mã bên dưới để đào tạo mô hình.

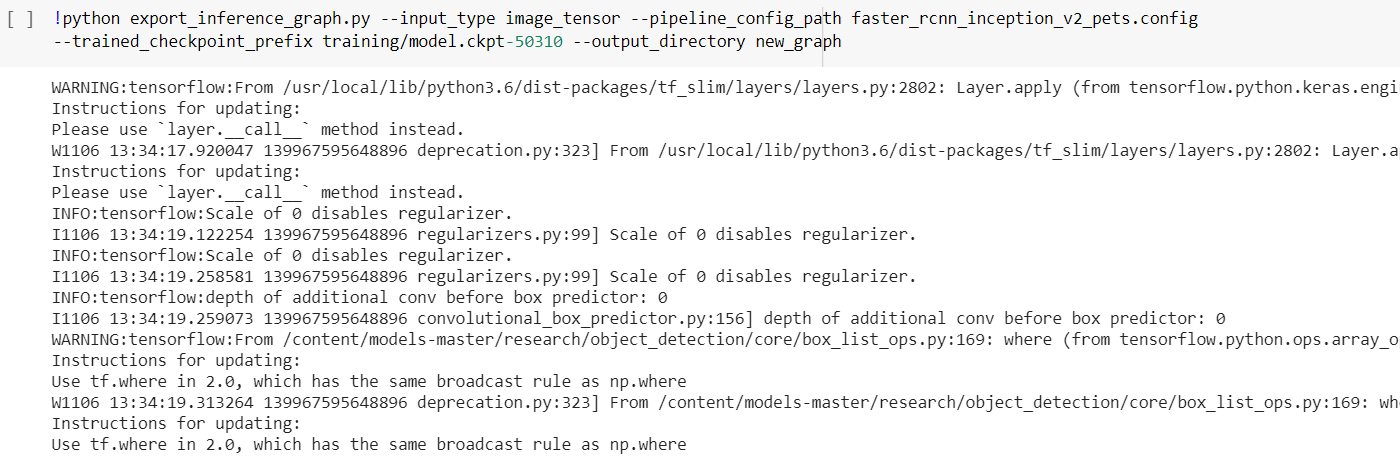




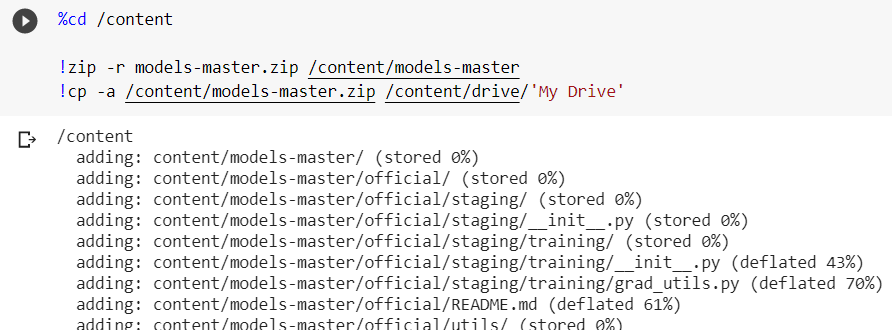
* **Bước 16**: Đào tạo cho đến khi global step hơn 50000 và loss nhỏ hơn 0.05 (Lúc thấy điểm kiểm tra và bạn thấy thích hợp thì dừng lại)



* **Bước 17**: Vào models-master -> research -> object\_detection -> training lấy điểm kiểm tra cao nhất.
* **Bước 18**: Lấy điểm kiểm tra đó thay vào đuôi ckpt

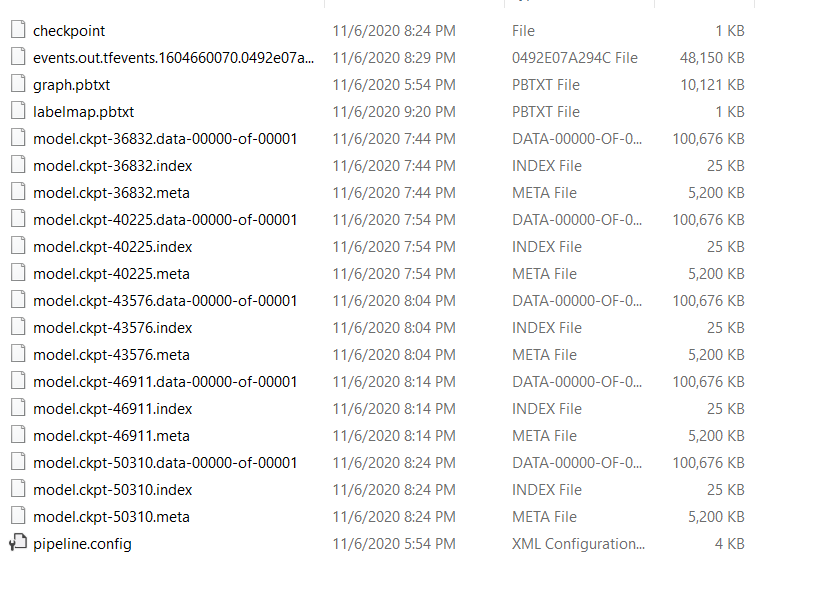


* **Bước 19**: Chạy mã bên dưới và thay đổi đường dẫn để nén lại model và tải lên google drive.

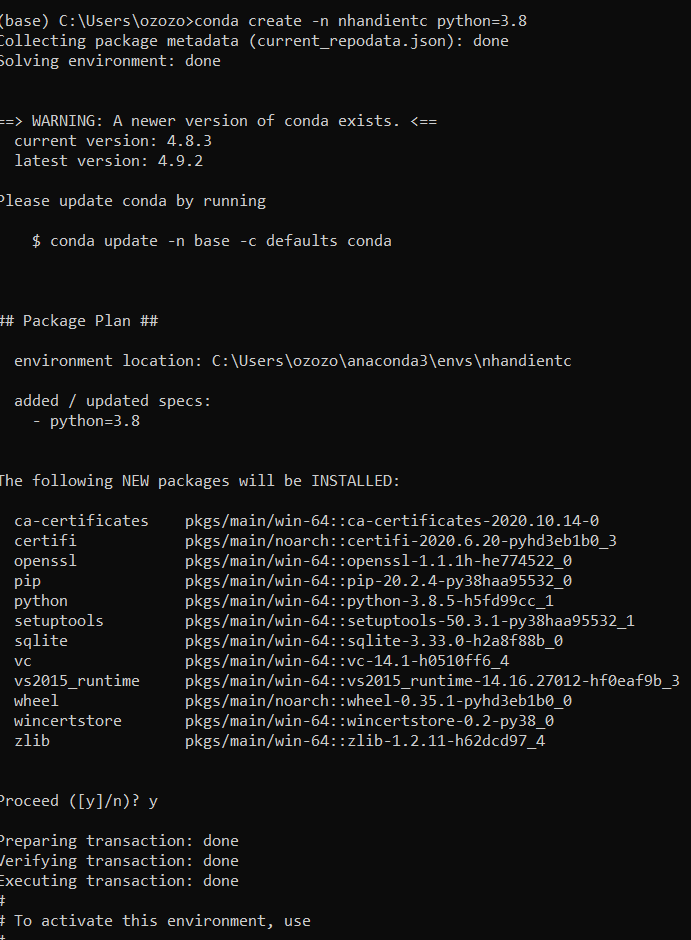


## **3.4. Chạy model trên anacoda**

* **Bước 1**: Tải model về máy. Vào thư mục object\_detection copy file labelmap.pbtxt dán vào thư mục training



* **Bước 2**: Mở anaconda
* **Bước 3**: Gõ câu lệnh *conda create –n <tên mình đặt> python=3.8*



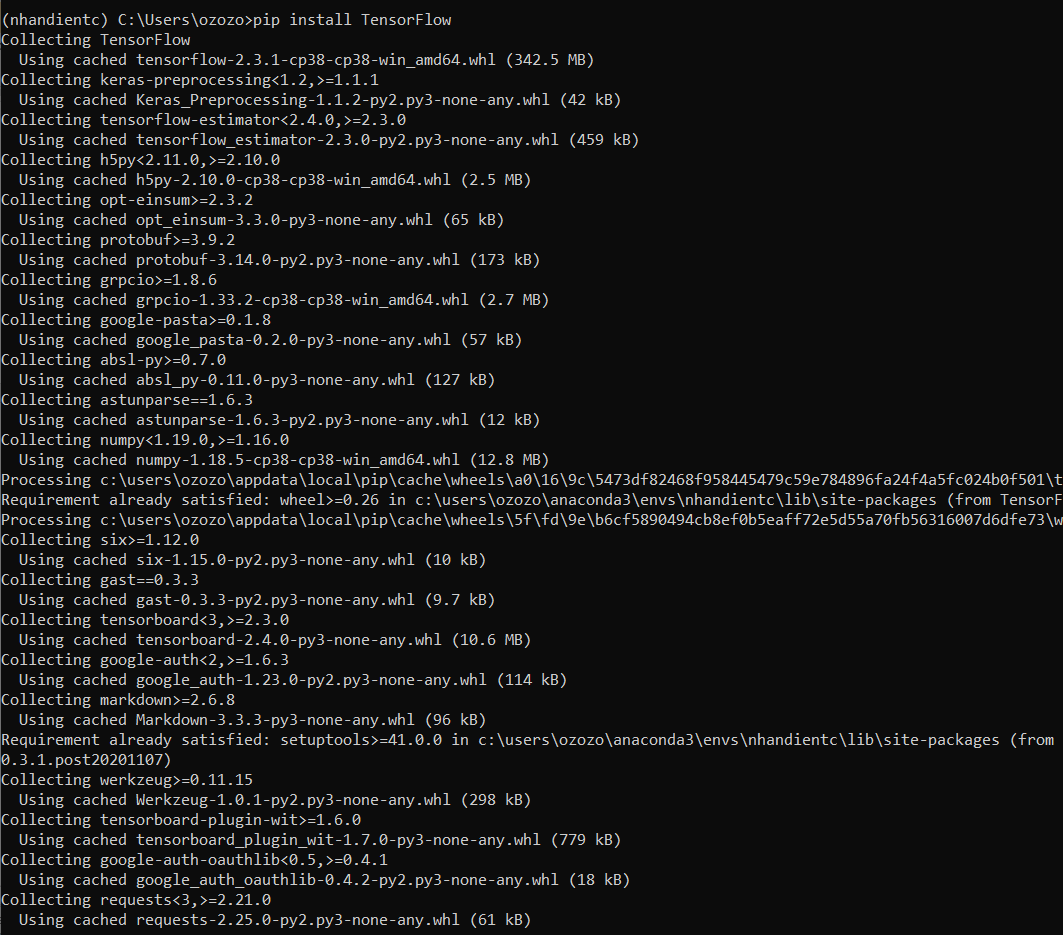
* **Bước 4**: Gõ câu lệnh *conda activate <tên mình đặt>*



* **Bước 5**: Gõ câu lệnh *conda install pip*



* **Bước 6**: Gõ câu lệnh *pip install Tensorflow*



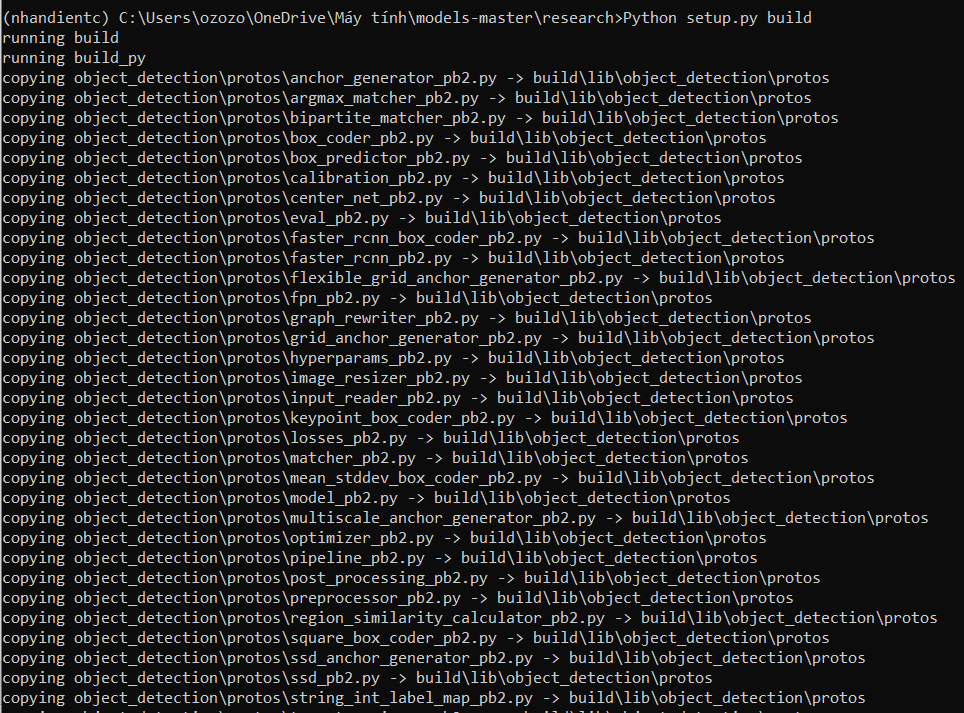
* **Bước 7**: Chuyển đến thư mục research bằng câu lệnh cd



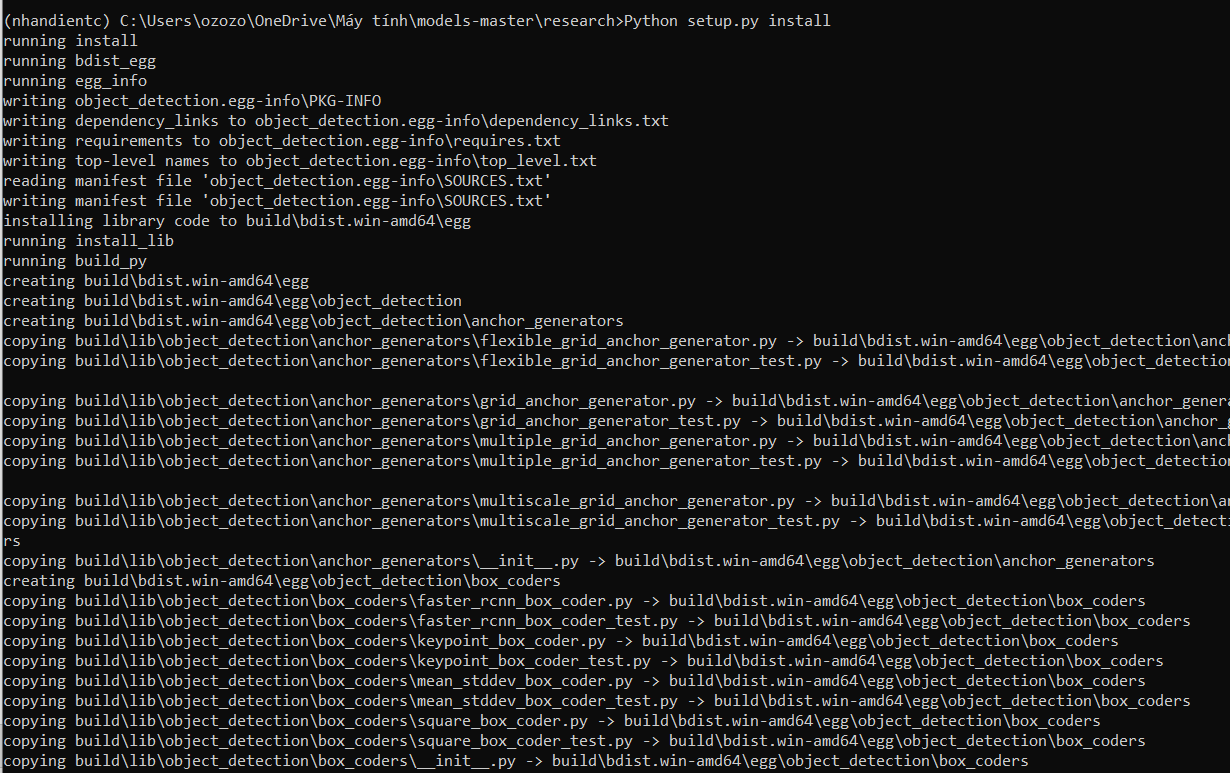
* **Bước 8**: Gõ câu lệnh *protoc object\_detection/protos/\*.proto --python\_out=.*



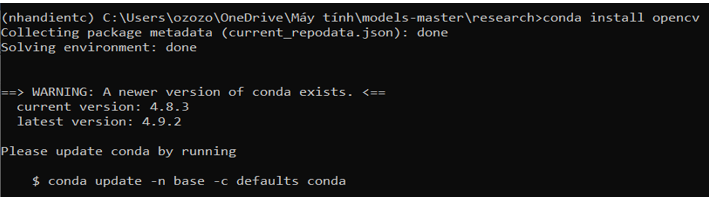
* **Bước 9**: Gõ câu lệnh *python setup.py build*

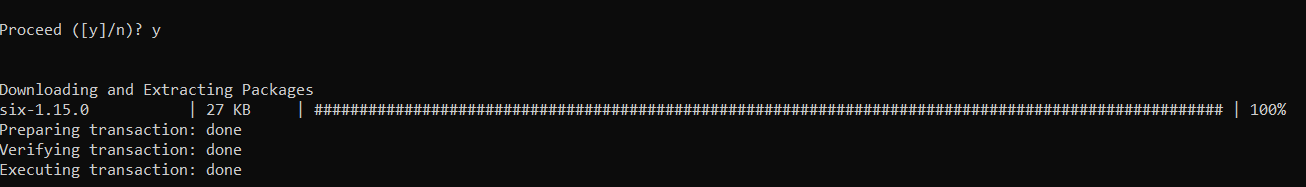


* **Bước 10**: Gõ câu lệnh *python setup.py install*

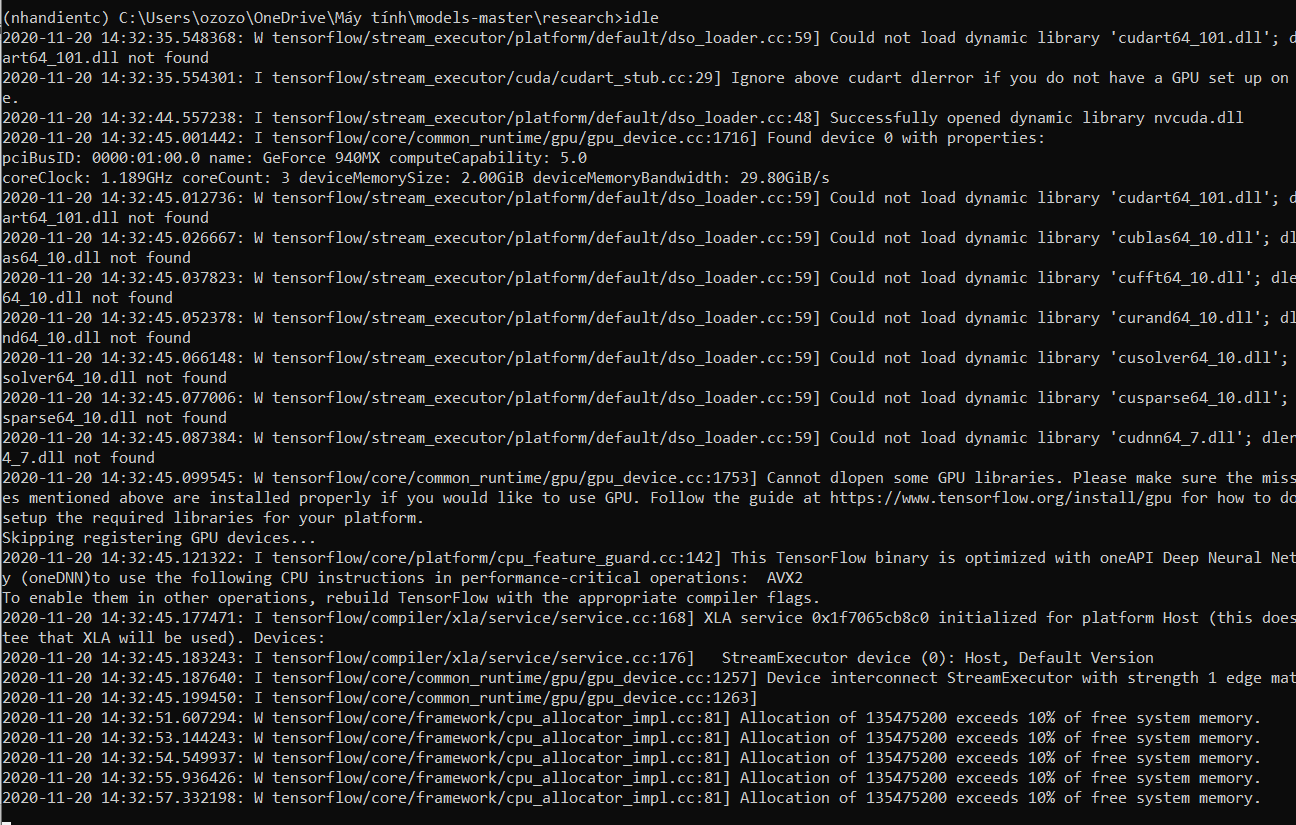


* **Bước 11**: Gõ câu lệnh *conda install opencv* để tải opencv

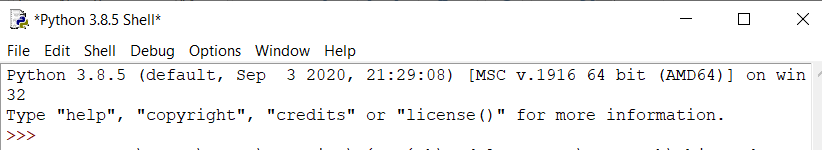




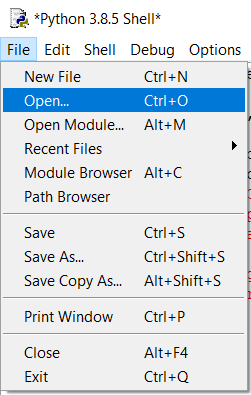
* **Bước 12**: Gõ câu lệnh *idle*



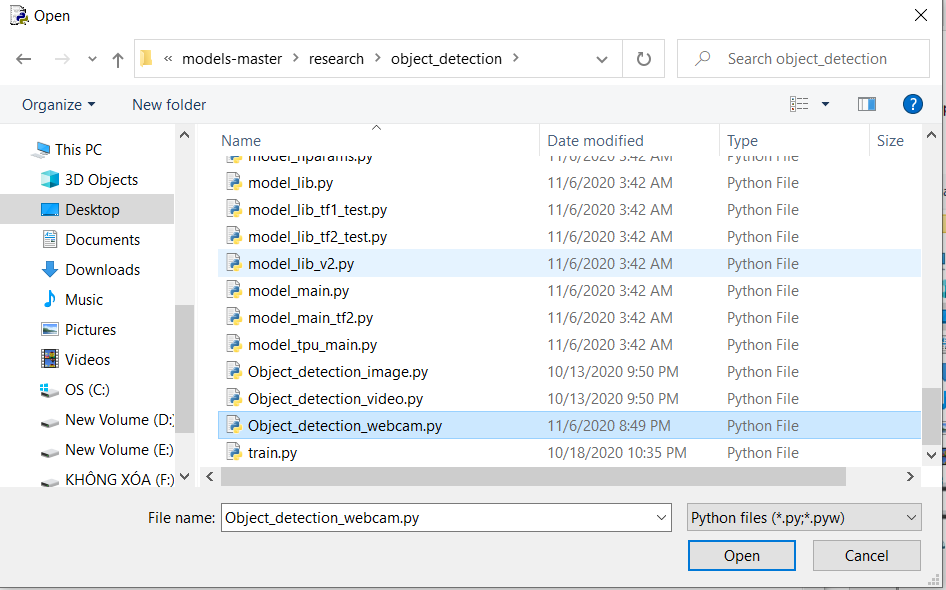
* **Bước 13**: Sau khi gõ câu lệnh idle sẽ xuất hiện hình bên dưới



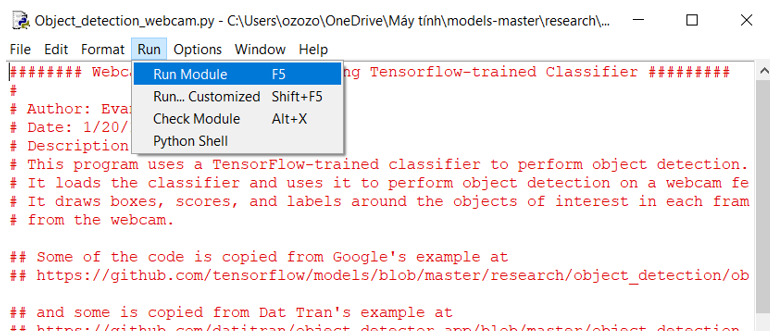
* **Bước 14**: Nhấn File -> Open.



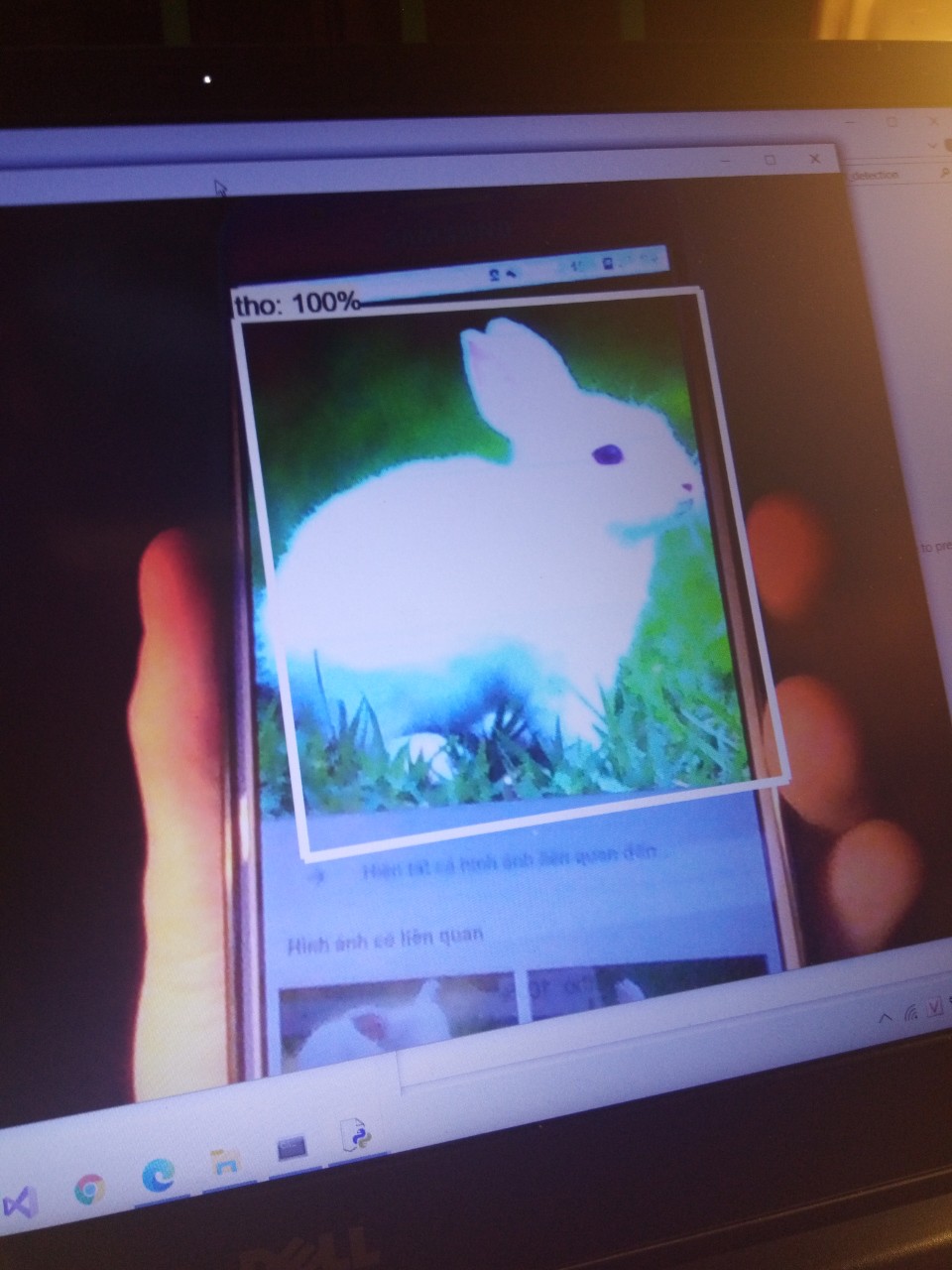
* **Bước 15**: Chọn model-master 🡪 chọn research🡪 chọn Object\_detection 🡪 chọn Object\_detection\_webcam.



* **Bước 16**: Chọn Run -> Run Module.



* **Bước 17**: Hiển thị ra màn hình webcam như dưới



# **CHƯƠNG IV**

# **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TRONG TƯƠNG LAI**

## **4.1. Kết quả đạt được, những vấn đề còn tồn tại**

### 4.1.1. Kết quả đạt được

* Về cơ bản, đã đáp ứng được phần nào yêu cầu đã đề ra trước đó là nhận diện thú cưng qua webcam.
* Kết quả đạt được mặc dù còn nhiều chỗ chưa đúng ý muốn nhưng cơ bản đã nhận diện được các thú cưng đã đào tạo.
* Demo chạy được không gặp lỗi
* Tỉ lệ nhận diện có độ chính xác cao

### 4.1.2. Vấn đề còn tồn tại

* Chưa thể nhận diện được tất cả các thú cưng mà chỉ nhận diện được các thú cưng đã đào tạo

## **4.2. Hướng hoàn thiện trong tương lại**

* Sẽ cố gắng có thể nhận diện được tất cả các thú cưng.