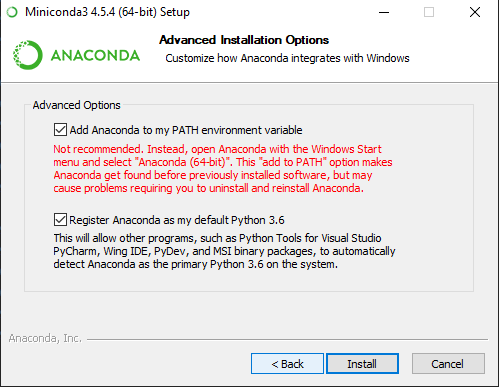
**YOLO PROJECT**

# Sau khi clone source về.

Cài đặt file "Miniconda3-4.5.4-Windows-x86\_64.exe".

Lưu ý trong quá trình cài đặt, nhớ chọn 2 option này để tạo biến môi trường trong windown.



# Sau khi cài đặt xong.

Mở Command Prompt or Windown PowerShell lên chạy các lệnh sau:

- conda create -n tensorflow pip python=3.6.5

- conda activate tensorflow

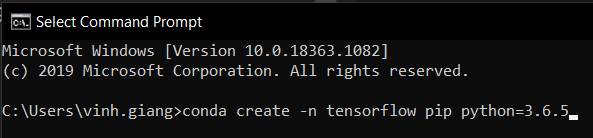
- pip install tensorflow==1.14

- pip install opencv-python

- pip install matplotlib

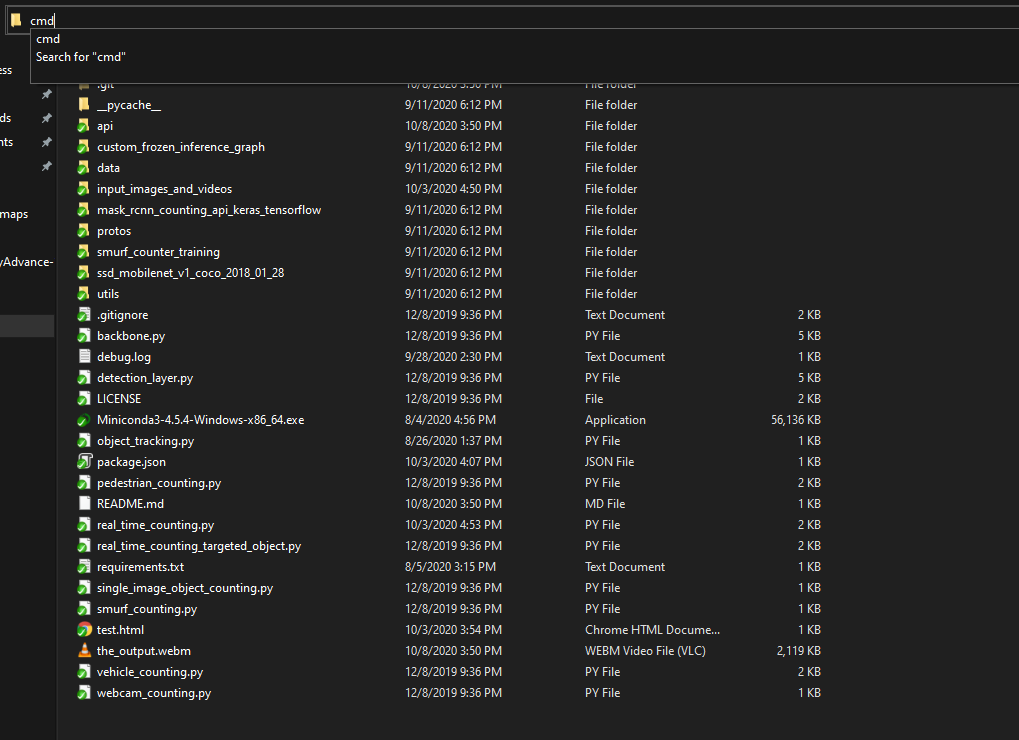
- pip install scipy

EXP:



Tắt hết Command Prompt or Windown PowerShell

# Sau khi install package thì vào trực tiếp project.



Gõ cmd vào đường dẫn folder và open Command Prompt or Windown PowerShell

Tiếp tục chạy lệnh sau để hệ thống hoạt động.

- test.html && activate tensorflow && python real\_time\_counting.py

# Thông tin chi tiết.

File video sau khi recorded: the\_output.webm

Hệ thống sử dụng:

- Phiên bản Python 3.6

- Phiên bản Tensorflow 1.14

- Phiên bản Miniconda3-4.5.4

Hệ thống kết nối trực tiếp với camera qua đường truyền thông qua giao thức RTSP để nhận diện và đếm đối tượng.

Vì là đường truyền thông qua giao thức RTSP nên sẽ bị chậm và đôi khi video bị lướt qua. Vì vừa thu video vừa kiểm tra đối tượng.

Khi kiểm tra đối tượng từ 1 video hoàn chỉnh. Sẽ không có tình trạng này xảy ra.

# Các khái niệm.

***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.

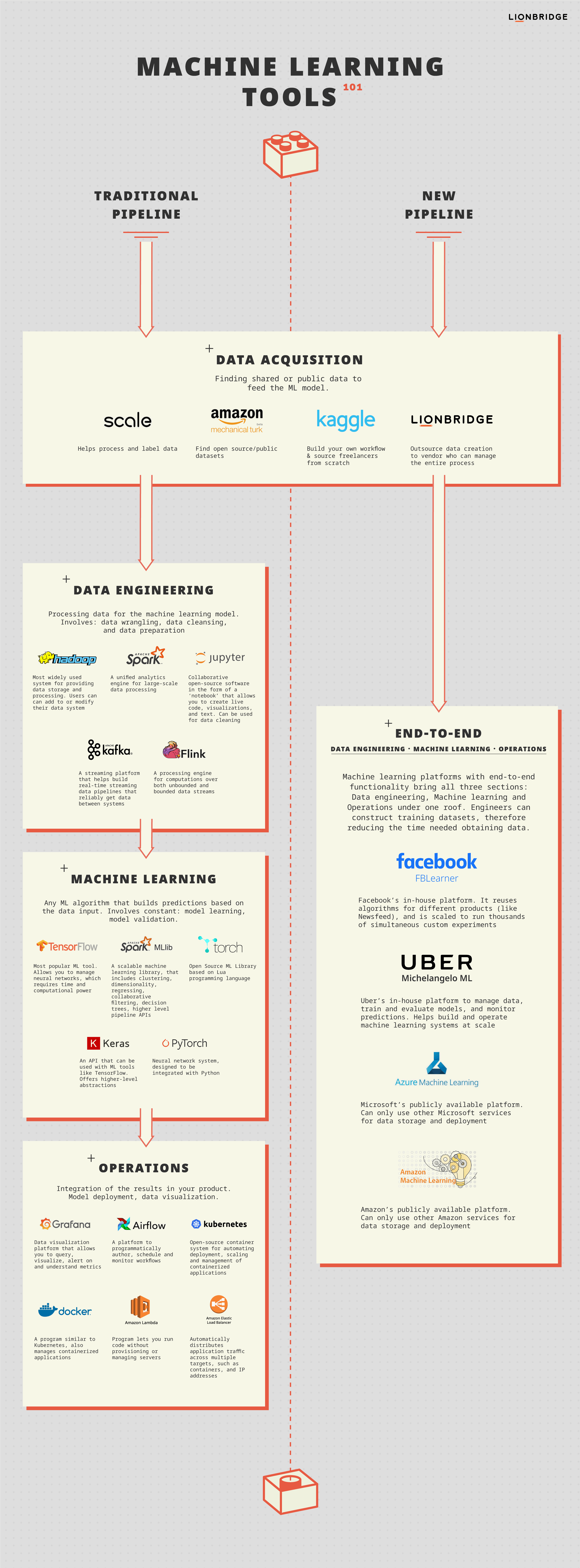
Các hàm được dựng sẵn trong thư viện cho từng bài toán cho phép TensorFlow xây dựng được nhiều neural network. Nó còn cho phép bạn tính toán song song trên nhiều máy tính khác nhau, thậm chí trên nhiều CPU, GPU trong cùng 1 máy hay tạo ra các dataflow graph – đồ thị luồng dữ liệu để dựng nên các model. 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.

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.

## **Lịch sử ra đời TensorFlow**

Vài năm trước, khi phải xử lý lượng dữ liệu khổng lồ, deep learning bắt đầu cho thấy hiệu năng vượt trội so với tất cả các thuật toán machine learning khác. Google sớm nhận ra tiềm năng này và nghĩ rằng họ nên sử dụng deep neural network để cải thiện các dịch vụ của mình, trong đó có:  
  
– Gmail.  
– Hình ảnh  
– Google search engine

Thế là họ dựng 1 framework có tên là TensorFlow để các nhà nghiên cứu cũng như lập trình viên có thể làm việc cùng nhau trên model A.I. 1 khi đã được phát triển và scale hoàn chỉnh, rất nhiều người đã có thể sử dụng được nó.

Ra mắt lần đầu vào cuối năm 2015, phiên bản TensorFlow ổn định cuối cùng cũng xuất hiện vào năm 2017. Là mã nguồn mở dưới sự cho phép của [Apache Open Source](https://www.apache.org/), giờ đây bạn có thể sử dụng, điều chỉnh và tái đóng góp phiên bản được điều chỉnh đó, đổi lại không cần phải trả bất cứ đồng nào cho Google. 

### **Kiến trúc của TensorFlow**

Kiến trúc TensorFlow hoạt động được chia thành 3 phần:  
  
– Tiền xử lý dữ liệu  
– Dựng model  
– Train và ước tính model

### **Cách TensorFlow hoạt động**

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.

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. 

#### **Lợi ích từ TensorFlow**

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/)

Lưu ý: Trong 1 số công việc training, vài chi tiết về việc triển khai của TensorFlow làm cho nó khó có thể quyết định được hoàn toàn kết quả training model . Đôi khi 1 model được train trên 1 hệ thống này sẽ có thay đổi 1 chút so với 1 model được train trên hệ thống khác, ngay cả khi chúng được cung cấp dữ liệu như nhau. Các nguyên nhân cho điều này cũng xê xích hay 1 số hành vi khi không được xác định khi sử dụng GPU. Điều này nói rằng, các vấn đề đó có thể giải quyết được, và đôi ngũ của TensorFlow cũng đang xem xét việc kiểm soát nhiều hơn để ảnh hưởng đến tính quyết định trong quy trình làm việc.

Giới thiệu các Component của TensorFlow

### **Tensor**

Tên của TensorFlow được đưa ra trực tiếp là nhờ vào framework cốt lõi của nó: Tensor. Trong TensorFlow, tất cả các tính toán đều liên quan tới các tensor. 1 tensor là 1 **vector** hay **ma trận** của n-chiều không gian đại diện cho tất cả loại dữ liệu. Tất cả giá trị trong 1 tensor chứa đựng loại dữ liệu giống hệt nhau với 1 **shape** đã biết (hoặc đã biết 1 phần). Shape của dữ liệu chính là chiều của ma trận hay mảng.  
  
1 tensor có thể được bắt nguồn từ dữ liệu input hay kết quả của 1 tính toán. Trong TensorFlow, tất cả các hoạt động được tiến hành bên trong 1 **graph** – biểu đồ. Biểu đồ là 1 tập hợp tính toán được diễn ra liên tiếp. Mỗi operation được gọi là 1 **op node**  (operation node) và được kết nối với nhau.  
  
Biểu đồ phát thảo các op và kết nối giữa các node. Tuy nhiên, nó không hiển thị các giá trị. Phần edge của các node chính là tensor, 1 cách để nhập operation với dữ liệu.

### **Graph**

TensorFlow sử dụng framework dạng biểu đồ. Biểu đồ tập hợp và mô tả tất cả các chuỗi tính toán được thực hiện trong quá trình training. Biểu đồ cũng mang rất nhiều lợi thế:  
  
– Nó được làm ra để chạy trên nhiều CPU hay GPU, ngay cả các hệ điều hành trên thiết bị điện thoại.  
– Tính di động của biểu đồ cho phép bảo toàn các tính toán để bạn sử dụng ngay hay sau đó. Biểu đồ có thể được lưu lại để thực thi trong tương lai.  
– Tất cả tính toán trong biểu đồ được thực hiện bằng cách kết nối các tensor lại với nhau. 1 **tensor** có 1 **node** và 1 **edge**. Node mang operation toán học và sản xuất các output ở đầu cuối. Các edge giải thích mối quan hệ input/output giữa các node.

## **Danh sách các thuật toán nổi bật được hỗ trợ bởi TensorFlow**

Tính tới thời điểm phiên bản ra mắt TensorFlow 1.10, nó đã sở hữu built-in API cho:

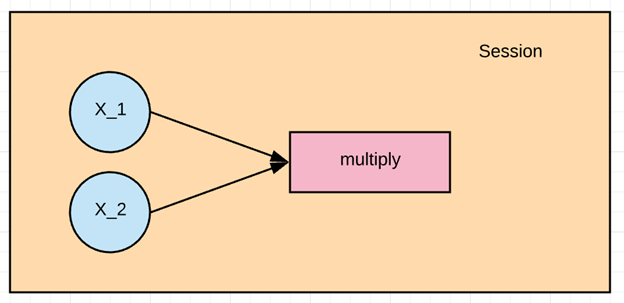
– **Linear regression:** tf.estimator.LinearRegressor  
– **Classification:** tf.estimator.LinearClassifier  
– **Deep learning classification:** tf.estimator.DNNClassifier  
– **Deep learning wipe and deep:** tf.estimator.DNNLinearCombinedClassifier  
– **Booster tree regression:** tf.estimator.BoostedTreesRegressor  
– **Boosted tree classification:** tf.estimator.BoostedTreesClassifie

Ví dụ TensorFlow đơn giản

import numpy as np

import tensorflow as tf

Trong 2 line code đầu, ta đã import tensorflow là tf. Với Python, nó chỉ là 1 bài tập thông dụng khi dùng tên viết tắt cho 1 thư viện. Lợi thế là để tránh gõ đầy đủ tên của thư viện khi ta cần sử dụng nó. Ví dụ: chúng ta có thể import tensorflow là tf, và gọi tf khi chúng ta muốn dùng 1 chức năng TensorFlow.  
  
Ta hãy luyện tập quy trình làm việc cơ bản của TensorFlow với 1 ví dụ cơ bản sau đây:Tạo 1 đồ thị tính toán nhân 2 số lại với nhau.  
  
Chúng ta sẽ nhân X\_1 và X\_2 lại với nhau. TensorFlow sẽ tạo 1 node để kết nối với operation. Trong ví dụ này, nó được gọi là multiply. Khi đồ thị được xác định, engine tính toán TensorFlow sẽ nhân X\_1 và X\_2 lại cùng nhau.

  
Cuối cùng, chúng ta chạy 1 session TensorFlow mà sẽ chạy đồ thị tính toán với giá trị của X\_1 và X\_2 và print phần kết quả của phép nhân.  
  
Hãy xác định các input node của X\_1 và X\_2. Khi ta tạo 1 node trong TensorFlow, chúng ta phải chọn nó là loại node gì. Các node X1 và X2 sẽ là node placeholder – node giữ chỗ. Placeholder chỉ định 1 giá trị mới mỗi khi ta làm 1 phép tính. Ta sẽ tạo chúng như là 1 node TF chấm placeholder.  
  
**Bước 1: Xác định giá trị**

X\_1 = tf.placeholder(tf.float32, name = "X\_1")

X\_2 = tf.placeholder(tf.float32, name = "X\_2")

**Bước 2: Xác định phần tính toán**

multiply = tf.multiply(X\_1, X\_2, name = "multiply")

Giờ ta có thể xác định node sẽ thực hiện operation phép nhân. Trong TensorFlow, chúng ta có thể làm điều đó bằng cách tạo 1 node tf.multiply.  
  
Ta sẽ nhập node X\_1 và X\_2 tới node nhân. Nó sẽ nói với TensorFlow để liên kết những node đó trong đồ thị tính toán, nên ta đang yêu cầu nó để pull các giá trị từ x và y và nhân phần kết quả. Hãy cho node nhân cái tên multiply. Nó là toàn bộ định nghĩa cho đồ thị tính toán đơn giản này.  
  
**Bước 3: Thực thi operation**  
  
Để thực thi các operation trong đồ thị, ta phải tạo 1 session. Trong TensorFlow, nó được thực hiện bằng tf.Session(). Giờ ta có 1 session ta có thể hỏi session để chạy operation trên đồ thị tính toán của ta bằng cách gọi session. Để chạy phần tính toán, chúng ta sẽ dùng run.  
  
Khi operation bổ sung chạy, nó sẽ thấy rằng nó cần để lấy các giá trị của node X\_1 và X\_2, nên chúng ta cũng cần cung cấp các trị cho X\_1 và X\_2. Ta có thể dùng điều đó bằng cách cung cấp 1 tham số được gọi là feed\_dict. Chúng ta chuyển giá trị 1,2,3 cho X\_1 và 4,5,6 cho X\_2.  
  
Chúng ta print phần kết quả với print(result). Chúng ta sẽ thấy 4,10 và 18 cho 1×4, 2×5 và 3,6.

## **Các option tải dữ liệu vào TensorFlow**

Bước đầu tiên trước khi train 1 thuật toán machine learning là load dữ liệu. Có 2 cách thông dụng để load dữ liệu:  
  
1. Load dữ liệu vào bộ nhớ: đây là phương pháp đơn giản nhất. Bạn load tất cả dữ liệu vào bộ nhớ như 1 mảng đơn. Bạn cũng có thể viết code bằng Python. Những dòng code này không liên quan gì tới TensorFlow.

2. Pipeline dữ liệu TensorFlow. TensorFlow sở hữu built-in API và nó sẽ giúp bạn load dữ liệu, thực thi các operation và feed thuật toán machine learning 1 cách dễ dàng. Phương pháp này hoạt động tốt đặc biệt khi bạn có 1 dataset lớn. Ví dụ: các hình ảnh thu được được biết khá là khổng lồ và không thể fit vào bộ nhớ. Pipeline dữ liệu sẽ tự quản lý phần bộ nhớ.

### **Giải pháp sẽ là gì?**

### **Load dữ liệu vào bộ nhớ**

Nếu dataset của bạn không quá lớn, chẳng hạn như dưới 10 GB, bạn có thể dùng phương pháp đầu tiên. Dữ liệu có thể dễ dàng fit vào bộ nhớ. Bạn cũng có thể dùng 1 thư viện nổi tiếng có tên là Pandas để import các tệp CSV. 1  
  
**Load dữ liệu với TensorFlow pipeline**  
  
Phương pháp thứ 2 sẽ hoạt động tốt nhất nếu bạn có 1 dataset lớn. Ví dụ: nếu bạn có 1 dataset nặng 50 GB và máy tính của bạn chỉ có 16GB dung lượng thì rõ ràng là máy sẽ crash thôi.

Trong tình huống này, bạn cần dựng 1 TensorFlow pipeline. Đường ống sẽ load dữ liệu trong batch, hay chunk nhỏ. Mỗi batch sẽ được push tới pipeline và sẵn sàng cho việc training. Dựng 1 pipeline là 1 giải pháp tuyệt vời vì nó còn cho bạn sử dụng phép tính toán song song. Nghĩa là TensorFlow sẽ train model qua nhiều CPU. Thúc đẩy sự tính toán và cho phép training mạng lưới thần kinh mạnh mẽ hơn.  
  
**Tóm tắt:**  
-Nếu bạn có 1 set dữ liệu nhỏ, bạn có thể load dữ liệu trong bộ nhớ với thư viện Pandas.  
-Nếu bạn có 1 set dữ liệu lớn và muốn sử dụng nhiều CPU, thì bạn sẽ thoải mái hơn khi làm việc cùng TensorFlow pipeline.

### Khái niệm cơ bản trong Tensorflow

Khi thực hành với Tensorflow, sẽ có rất nhiều khái niệm phức tạp. Tuy nhiên chỉ ở mức cơ bản, chúng ta sẽ đi đến khái niệm quan trọng nhất trong Tensorflow là **Tensor**[[Tensor]](https://kipalog.com/posts/Bat-dau-voi-Machine-Learning-thong-qua-Tensorflow--Phan-I-2#tensor).

#### Node

Vì Tensorflow mô tả lại dòng chảy của dữ liệu thông qua graph nên mỗi một điểm giao cắt trong graph thì được gọi là Node. Tại sao điều này quan trọng thì là vì các Node chính là điểm đại diện cho việc thay đổi của dữ liệu nên việc lưu trữ lại tham chiếu của các Node này là rất quan trọng.

#### Tensor

Như trong bài viết trước mình có đề cập, để giải được các bài toán Machine Learning, cần phải làm cho máy tính có thể hiểu được dữ liệu của tập nguồn và dữ liệu của tập đích. Tensorflow cung cấp một loại dữ liệu mới được gọi là Tensor[[Tenensorflow]](https://www.tensorflow.org/api_docs/python/tf/Tensor). Trong thế giới của Tensorflow, mọi kiểu dữ liệu đều được quy về một mối được gọi là Tensor hay trong Tensorflow, tất cả các loại dữ liệu đều là Tensor. Vậy nên có thể hiểu được phần nào cái tên Tensorflow là một thư viện mô tả, điều chỉnh dòng chảy của các Tensor.

Tensor là một kiểu dữ liệu dạng mảng có nhiều chiều được mô tả dạng Tensor = [[[1,1,1],[178,62,74]],[[45,2,2],[19,0,17]],[[7,5,2],[0,11,4]],[[8,13,5],[1,6,7]]]. Mảng nhiều chiều này được đính kèm thêm một vài thuộc tính tham chiếu khác. Các thuộc tính của Tensor được mô tả trong tài liệu bao gồm:

* **device**: Tên của thiết bị mà Tensor hiện tại sẽ được xuất bản. Có thể None.
* **graph**: Đồ thị chứa Tensor hiện tại.
* **name**: Tên của Tensor hiện tại.
* **shape**: Trả về TensorShape mô tả lại Shape của Tensor hiện tại.
* **op**:  được sử dụng để xuất bản Tensor hiện tại.
* **dtype**: Kiểu của các  trong Tensor hiện tại.

##### Rank

Rank là bậc hay độ sâu của một Tensor. Ví dụ như Tensor = [1] sẽ có rank là 1, Tensor = [[[1,1,1],[178,62,74]]] sẽ có rank bằng 3, Tensor = [[1,1,1],[178,62,74]] sẽ có rank bằng 2. Cách nhanh nhất để xác định rank của một Tensor là đếm số lần mở ngoặc vuông cho đến giá trị khác ngoặc vuông đầu tiên. Việc phân rank này khá quan trọng vì nó đồng thời cũng giúp phân loại dữ liệu của Tensor. Khi ở cách rank đặc biệt cụ thể, Tensor có những tên gọi riêng như sau:

* **Scalar**: Khi Tensor có rank bằng 0, Tensor đại diện cho một số hoặc một chuỗi cụ thể. Ví dụ: scalar=123.
* **Vector**: Vector là một Tensor rank 1. Trong python thì Vector là một list hay mảng một chiều[[Pyton]](https://docs.python.org/2/tutorial/datastructures.html#more-on-lists) chứa các số. Ví dụ: list=[123,345].
* **Matrix**: Đây là một Tensor rank 2 hay mảng hai chiều theo khái niệm của Python. Ví dụ: matrix=[[1,2],[2,1]].
* **N-Tensor**: Khi rank của Tensor tăng lên lớn hơn 2, chúng được gọi chung là N-Tensor.
* [Object detection](https://www.fritz.ai/object-detection/) is a branch of [computer vision](https://heartbeat.fritz.ai/the-5-computer-vision-techniques-that-will-change-how-you-see-the-world-1ee19334354b), in which visually observable objects that are in images of videos can be detected, localized, and recognized by computers. An image is a single frame that captures a single-static instance of a naturally occurring event . On the other hand, a video contains many instances of static images displayed in one second, inducing the effect of viewing a naturally occurring event.
* Technically, a single static image in a video is called a **video frame**. In most videos, the number of frames in one second of the video ranges between 20 to 32, and this value is called the **frames-per-second**(fps).
* Detecting objects in images and videos accurately has been highly successful in the second decade of the 21st century due to the rise of machine learning and deep learning algorithms. [Specialized algorithms](https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e) have been developed that can detect, locate, and recognize objects in images and videos, some of which include RCNNs, SSD, RetinaNet, [YOLO](https://heartbeat.fritz.ai/gentle-guide-on-how-yolo-object-localization-works-with-keras-part-2-65fe59ac12d),and others.
* Using these algorithms to detect and recognize objects in videos requires an understanding of applied mathematics and solid technical knowledge of the algorithms as well as thousands of lines of code. This is a highly technical and time-consuming process, and for those who desire to implement object detection can find the process very inconvenient.
* In this article, we we’ll be using a Python library called [ImageAI](https://github.com/OlafenwaMoses/ImageAI/) that has made it possible for anyone with basic knowledge of Python to build applications and systems that can detect objects in videos using only a few lines of programming code. ImageAIsupports [YOLOv3](https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b),which is the object detection algorithm we’ll use in this article.
* To get started, you will install a number of Python libraries and ImageAI**.**If you have any of the dependencies mentioned below already installed on your computer, you can jump straight to the installation of ImageAI**.**Also ensure the Python version you have installed on your computer is [**Python**](https://www.python.org/downloads/release/python-360/)
* Note: If you have a computer system with an NVIDIA GPUand you installed the GPU version of TensorFlow, this detection process should be done in less than a minute. Otherwise, it may take a few minutes. The detection will progress for each frame of the video detected, and the detected video that’s saved will be automatically updated for each frame detected.
* Once the detection is done, we’ll find the detected video in the folder that contains our Python file. When we open and play the video, it will contain the original video, but in this case, with boxes that locate various objects, the names of those objects, and the probability of the recognition as a percentage.

## Giới thiệu về Opencv

### 1. Opencv là gì

OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. OpenCV đươc viết bằng C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần

### 2. Ứng dụng

Thư viện OpenCV bao gồm một số tính năng nổi bật như:

* Bộ công cụ hỗ trợ 2D và 3D
* Nhận diện khuôn mặt
* Nhận diện cử chỉ
* Nhận dạng chuyển động, đối tượng, hành vi,
* Tương tác giữa con người và máy tính
* Điều khiển Robot
* Hỗ trợ thực tế tăng cường

Opencv có rất nhiều ứng dụng:

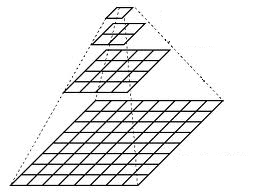
* Nhận dạng ảnh
* Thực tế ảo
* Giám sát tự động
* Tìm kiểm, phục hồi, xử lý ảnh/video
* Nhận dạng khuôn mặt, cử chỉ.
* Nhận dạng chữ viết, con số, ký tự.
* Các ứng dụng khác

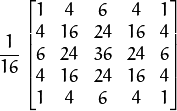
## Pyramid

Pyramid, hoặc pyramid representation, là một dạng biểu diễn tín hiệu đa dạng được dùng trong computer vision, xử lý hình ảnh hoặc các xử lý tín hiệu. Trong đó, một tín hiệu hoặc một hình ảnh được lặp lại với các phương pháp smoothing và subsampling. Tên gọi pyramid biểu thị cho việc biểu diễn trong không gian và các phương pháp phân tích. Một image pyramid thực chất là một tập hợp các hình ảnh mà tất cả chúng đều phát sinh từ một hình ảnh ban đầu duy nhất, hình ảnh này được liên tục được lấy mẫu xuống cho tới khi đạt đến một điểm dừng mong muốn. Có hai loại image pỷamid ảnh phổ biến:

* Gaussian pyramid: Sử dụng để downsample hình ảnh tức là để giảm độ phân giải của hình ảnh
* Laplacian pyramid: Được sử dụng để tái tạo lại một hình ảnh được lấy mẫu từ một hình ảnh thấp hơn trong pyramid (có độ phân giải thấp hơn)

### Gaussian pyramid

Gaussian pyramid là một tập các layer trong đó layer ở vị trí càng cao thì càng có kích thước nhỏMỗi layer được đánh số từ dưới lên trên, do đó, layer(i + 1) (được biểu thị là G(i + 1) nhỏ hơn layer i (G(i))

* Gắn lại G(i) thành một Gaussian kernel: 
* Loại bỏ tất cả các hàng và cột số chẵn

Bạn có thể dễ dàng nhận thấy rằng hình ảnh kết quả sẽ chỉ chính xác bằng một phần tư ảnh trước đó. Lặp đi lặp lại quá trình nhiều lần với hình ảnh đầu vào G(0) (hình ảnh ban đầu) ta sẽ có một tập layer image pyramid.

Quá trình trên đã làm giảm độ chính xác của ảnh, vậy để tăng kích thước thì sao? Đầu tiên, ta tăng ma trận hình ảnh lên gấp đôi bản gốc trong mỗi chiều, với các hàng và cột mới đầy những số không (0). Sau đó, thực hiện các bước lặp với cùng một kernel ở trên.

### Laplacian pyramid

Laplacian pyramid được tạo ra Gaussian pyramid. Nếu bạn mở một ảnh Laplacian pyramid thì nó là một ảnh chỉ có các đường viền bao ngoài các vật thể, đã xóa hết các phần khác. Hầu hết các phần tử trong ma trận là zeros. Laplacian pyramid thường được sử dụng trong nén hình ảnh. Một level (mức độ) Laplacian pyrmaid được hình thành bởi sự khác biệt giữa level đó trong Gaussian pyramid và level trên nó (layer đã được mở rộng) trong Gaussian pyramid.

# **Matplotlib**

Để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu của bạn và Matplotlib là một trong những giải pháp như vậy cho người dùng Python. Nó là một thư viện vẽ đồ thị rất mạnh mẽ hữu ích cho những người làm việc với Python và NumPy. Module được sử dụng nhiều nhất của Matplotib là Pyplot cung cấp giao diện như MATLAB nhưng thay vào đó, nó sử dụng Python và nó là nguồn mở.

Để cài đặt Matplotlib nếu bạn có Anaconda chỉ cần gõ conda install matplotlib hoặc sử dụng tools pip pip install matplotlib

## Khái niệm chung

Một Matplotlib figure có thể được phân loại thành nhiều phần như dưới đây:

* **Figure:** Như một cái cửa sổ chứa tất cả những gì bạn sẽ vẽ trên đó.
* **Axes:** Thành phần chính của một figure là các axes (những khung nhỏ hơn để vẽ hình lên đó). Một figure có thể chứa một hoặc nhiều axes. Nói cách khác, figure chỉ là khung chứa, chính các axes mới thật sự là nơi các hình vẽ được vẽ lên.
* **Axis:** Chúng là dòng số giống như các đối tượng và đảm nhiệm việc tạo các giới hạn biểu đồ.
* **Artist:** Mọi thứ mà bạn có thể nhìn thấy trên figure là một artist như Text objects, Line2D objects, collection objects. Hầu hết các Artists được gắn với Axes.

Thông tin tham khảo từ:

<https://stackabuse.com/object-detection-with-imageai-in-python/>

<https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/install.html#install-cuda-toolkit>

<https://heartbeat.fritz.ai/detecting-objects-in-videos-and-camera-feeds-using-keras-opencv-and-imageai-c869fe1ebcdb>

<https://towardsdatascience.com/real-time-mobile-video-object-detection-using-tensorflow-a75fa0c5859d>

<https://github.com/ahmetozlu/tensorflow_object_counting_api>