191023

https://towardsdatascience.com/design-a-federated-learning-system-in-seven-steps-d0be641949c6

1. you need to establish a centralized service to manage the participants. This service will be responsible for coordinating communication between the participants, as well as monitoring the training progress.

From an operational perspective, this service will likely need to:

1. Have authentication and authorization mechanisms built in, along with the support structure to keep it reliable; this includes ensuring that the service is stateless to aid in load balancing, which means that a storage mechanism needs to be chosen to hold intermediate information passing between clients.
2. Be deployed, and maintained to meet the demand of the federated learning system.
3. Be able to administer the training sessions.

* Does authorization or service isolation need to be added to separate different data networks? That is, some groups of participants that can collaborate together, but not between groups.
* Can a client trigger a training session or does it have to be centrally administered?
* How will clients disconnecting and reconnecting affect the training? This problem is compounded by the number of parties participating in the training.
* What statistics should be collected during the training session and how will monitoring be set up so that the system can measure the quality of the model being trained?
* How will the service manage participating clients operating at different speeds?
* How will the system know when to drop a client if they become unreliable so that the model can still continue to be trained?

2. Design a client system

Now it’s time to consider a possible design for the client system. This system needs to be able to

* perform client side training operations,
* and coordinate the model parameters with the central service.
* The client system will also need to fetch new parameters from other clients in the network to update the local model.

Should the client system be a package that’s installable or should it be something like a docker image? How will dependency versioning be managed?

* How will the client authenticate and communicate with the server?
* How will monitoring the training process work?
* How will error recovery be handled during the model training process?

3.The federated learning system needs to know what private data should be used from each client to train the local models for a particular session.

This information needs to come from another user, or the central service.

Therefore, the meta information about available data has to be managed in some form; this is typically done by the central service.

This also requires that clients register this meta information about what datasets are available for other clients. Similarly, metadata for each client will need to be retrieved from the central service about which datasets should be used for a training session.

4. The federated learning system needs to manage model metrics and access so that the appropriate users can use the trained model.

The following questions can be used to guide how the system will manage models:

* Should all participants be able to access a copy of the model or do only some participants get access to it?
* Where will the model be stored (typically this will be by the centralized service)?

FedCV is an FL framework built for computer vision applications. It bridges the gap between research and implementation by providing an easy-to-use unified library with multiple functionalities. The practical applications of FedCV can be found across multiple industries, including healthcare, transportation, or manufacturing.

example image recognition for a smartphone app identifying animals. This is done in the following way:

1. Using the local dataset a model is trained on the smartphone
2. The model is sent to the server
3. The server creates a global model by aggregating all local models
4. The new global model is sent back to all smartphones
5. Each smartphone receives the updated global model

https://www.analyticsvidhya.com/blog/2021/04/federated-learning-for-beginners/

### Let’s Code

Let’s start coding. This example will use TensorFlow compiling a MobileNetV2 model and use the CIFAR-10 dataset. Before we can start coding we need to install all required libraries. You can do this with the following command: `pip install flwr==0.15.0 tensorflow==2.4.1`.

#### Federated Learning Client

We will start by writing the client-specific code. Create a new file and call it `client.py`. Initially import the required libraries `flwr` and `tensorflow`. Afterward, you have to create and compile the model followed by the data loading code. Here is the code:

import flwr as fl

import tensorflow as tf

# Load and compile Keras model

model = tf.keras.applications.MobileNetV2((32, 32, 3), classes=10, weights=None)

model.compile("adam", "sparse\_categorical\_crossentropy", metrics=["accuracy"])# Load CIFAR-10 dataset

(x\_train, y\_train), (x\_test, y\_test) = tf.keras.datasets.cifar10.load\_data()

import tensorflow as tf

# Load and compile Keras model

model = tf.keras.applications.MobileNetV2((32, 32, 3), classes=10, weights=None)

This should look familiar to anyone who has prior experience with TensorFlow or Keras. Next, we build a Flower client called `CifarClient` which is derived from Flower’s convenience class `NumpyClient`. The abstract base class `NumpyClient` defines three methods that clients need to override. These methods allow Flower to trigger training and evaluation of the previously defined Keras model:

# Define Flower client

class CifarClient(fl.client.NumPyClient):

def get\_parameters(self):

return model.get\_weights()

def fit(self, parameters, config):

model.set\_weights(parameters)

model.fit(x\_train, y\_train, epochs=1, batch\_size=32, steps\_per\_epoch=3)

return model.get\_weights(), len(x\_train), {}

def evaluate(self, parameters, config):

model.set\_weights(parameters)

loss, accuracy = model.evaluate(x\_test, y\_test)

return loss, len(x\_test), {"accuracy": accuracy}

Flower’s `NumpyClient.fit` method receives weights from the server, updates the model with those weights, trains the model on the locally held dataset `(x\_train/y\_train)`, and then returns the updated weights (via `model.get\_weights`). Note that you can do a quick “dry run” bypassing `steps\_per\_epoch=3` to `model.fit` – this will only process three batches per epoch instead of the entire dataset. Remove `steps\_per\_epoch=3` to train on the full dataset (this will take longer).

The evaluate method works similarly, but it uses the provided weights to evaluate the model on the locally held dataset `(x\_test/y\_test)`. The last step is to create an instance of CifarClient and run it:

# Start Flower client

fl.client.start\_numpy\_client(server\_address="[::]:8080", client=CifarClient())

#### Federated Learning Server

In a new script called server.py, we add the following two lines to start a Flower server that performs three rounds of  [Federated Averaging](https://arxiv.org/pdf/1602.05629.pdf) that simply takes a weighted model parameter and averages them:

import flwr as fl

fl.server.start\_server(config={"num\_rounds": 3})

That’s it!

Implementing federated learning for devices with YOLOv4-tiny or any other neural network model involves a distributed approach to training machine learning models. In federated learning, you train a global model across multiple decentralized devices while keeping data on those devices. Here's a high-level overview of how you can implement federated learning with YOLOv4-tiny:

1. Set Up the Environment:
   * Ensure that you have a working Python environment with the necessary libraries, including TensorFlow, PyTorch, or any other deep learning framework you intend to use.
   * Install the required federated learning libraries, such as TensorFlow Federated (TFF).
2. Data Preparation:
   * You'll need to have labeled data for YOLOv4-tiny. In a federated learning setting, the data is distributed across multiple devices. Each device should have a local dataset.
   * Ensure that the data is properly formatted for YOLOv4-tiny, and split it into training and validation sets on each device.
3. Model Conversion:
   * Convert your YOLOv4-tiny model into a format compatible with your chosen deep learning framework (e.g., TensorFlow or PyTorch) to ensure compatibility with federated learning libraries.
4. Federated Learning Framework:
   * Choose a federated learning framework like TensorFlow Federated (TFF). TFF is specifically designed for federated learning and has built-in functionalities for federated model training.
5. Model Aggregation:
   * Implement a model aggregation strategy. In federated learning, each device trains a local model on its data, and then the global model is updated through aggregation.
   * Common aggregation strategies include Federated Averaging and Federated SGD.
6. Federated Learning Algorithm:
   * Implement the federated learning algorithm using TFF. You will define the server and client logic in the federated learning setting.
   * TFF provides tools for federated computations, including federated map and federated reduce operations.
7. Communication and Security:
   * Set up secure communication between the central server and the devices. Federated learning often involves secure aggregation to protect the privacy and security of the participating devices.
8. Training Loop:
   * Create a federated training loop that iteratively trains the global model by sending model updates to devices, aggregating their results, and repeating the process for multiple rounds.
9. Evaluation:
   * Implement mechanisms to evaluate the global model's performance on a validation dataset, ensuring that it converges to a satisfactory accuracy.
10. Hyperparameter Tuning:
    * Tune hyperparameters specific to federated learning, such as the number of communication rounds, learning rates, and aggregation strategies.

Deployment:

* Deploy your federated learning system to the devices and the central server.

Testing and Monitoring:

* Continuously monitor the federated learning process for convergence and accuracy.

Handling Device Failures:

* Implement mechanisms to handle devices that go offline or fail during the federated learning process.

211023

python convert.py yolov4-tiny.cfg yolov4-tiny.weights yolov4-tiny.h5

พอ convert data

(x\_train, y\_train), (x\_test, y\_test)

ได้แล้วก็มาต่อกันที่ weight file ค่ะ

convert.py ได้มาแล้ว คราวนี้จะต้องไปลองรันคำสั่งด้านบนดู แต่ไฟล์ต้องไป !.unrar ใน colab

ถ้าได้ไฟล์ output ออกมาคือ yolov4-tiny.h5 ออกมาแล้ว

ก็ต้องไป convert model ต่อ

เหมือนว่าต้องไปเพิ่ม activation function อันใหม่ ชื่อ Mish ที่เพิ่งมีใน v4-tiny

ใน tiny มี layer น้อยกว่าเยอะเลย ต้องแก้ตรงนี้ด้วยหรือเปล่า

แล้วตอนใส่ (x\_train, y\_train) เข้าไปก็ไม่เหมือนด้วยนะ

ตอนที่จะใส่ดาต้าเขข้าไป ให้ใช้ keras.imagedatagenerator() อ่านแล้วใส่เข้าไป

ถ้าใชิของ hunglc007 จะเขียน Dataset() ขึ้นมาเอง คือมี core มาก่อน แล้วก็

1. define config

2. แล้วค่อยไปสร้าง dataset.py

3. เขียนโมเดลYOLOv4-tiny หรือไม่ tiny เป็น python

แล้วหลังจาก core หมด ค่อยเอาไป train.py

คือใส่ annot.path ตั้งแต่ใน config แล้ว ส่วนตัว image ให้ใช้เป็น input.size แล้วก็นับincrement ไป

git clone <https://github.com/hunglc007/tensorflow-yolov4-tflite>

มาเลย

แล้วก็เรียก

# Convert darknet weights to tensorflow

python save\_model.py --weights ./data/yolov4-tiny.weights --output ./checkpoints/yolov4-tiny-416 --input\_size 416 --model yolov4 –tiny

python detect.py --weights ./checkpoints/yolov4-tiny-416 --size 416 --model yolov4 --image ./data/kite.jpg --tiny

แล้วถ้าจะสั่ง train ก็เปลี่ยนเป็น train.py แค่นั้นเหรอ?

ทีนี้ดูตอนใส่ดาต้าเข้าไปใน Dataset

def parse\_annotation(self, annotation):

line = annotation.split()

image\_path = line[0]

if not os.path.exists(image\_path):

raise KeyError("%s does not exist ... " % image\_path)

image = cv2.imread(image\_path)

if self.dataset\_type == "converted\_coco":

bboxes = np.array(

[list(map(int, box.split(","))) for box in line[1:]]

)

elif self.dataset\_type == "yolo":

height, width, \_ = image.shape

bboxes = np.array(

[list(map(float, box.split(","))) for box in line[1:]]

)

bboxes = bboxes \* np.array([width, height, width, height, 1])

bboxes = bboxes.astype(np.int64)

ใน dataset.py

class Dataset(object):

"""implement Dataset here"""

def \_\_init\_\_(self, FLAGS, is\_training: bool, dataset\_type: str = "converted\_coco"):

self.tiny = FLAGS.tiny

self.strides, self.anchors, NUM\_CLASS, XYSCALE = utils.load\_config(FLAGS)

self.dataset\_type = dataset\_type

self.annot\_path = (

cfg.TRAIN.ANNOT\_PATH if is\_training else cfg.TEST.ANNOT\_PATH

)

ก็ต้องไปดูใน config.py ว่าอี self.annotation นี่คืออะไร

# Train options

\_\_C.TRAIN = edict()

\_\_C.TRAIN.ANNOT\_PATH = "./data/dataset/val2017.txt"

ซึ่ง format ก็คือ class\_id bbox[0] bbox[1 bbox[2] bbox[3] นั่นเอง

[list(map(float, box.split(","))) for box in line[1:]]

)

bboxes = bboxes \* np.array([width, height, width, height, 1])

bboxes = bboxes.astype(np.int64)

พอก๊อปไฟล์ convert image\_to\_arrays.py กับ git clone ไปวาง แล้วก็สั่ง python train.py ให้ได้

หลังจากนันก็โหลดใส่ variable model ใน flower

ก็ใกล้เสร็จแล้ว

(x\_train, y\_train), (x\_test, y\_test) = tf.keras.datasets.cifar10.load\_data()

x\_train.shape

(50000, 32, 32, 3)

y\_train.shape

(50000, 1)

แต่ที่เราจะใส่ไปในโมเดลเรามันไม่ใช่แบบนี้อะสิ

เพราะโมเดลเราได้รับการ convert ตาม hunglc007 ซึ่งจะให้ใส่เป็น annot\_path

├── Dataset/

│ ├── image1.jpg

│ ├── image1.txt

│ ├── image2.jpg

│ ├── image2.txt

│ └── ...

I have a directory like this. in each .txt file i have class\_id, x\_center, y\_center, width, height

I would like to make a new .txt file where each line in the file consists of

output\_file.write(f"{image\_path} {x\_min} {y\_min} {x\_max} {y\_max} {int(class\_id)}\n")

ซึ่ง format yolo มา มันเป็น x\_center, y\_center, width, height

class\_id, x\_center, y\_center, width, height = [float(val) for val in txt\_file.read().split()]

# Convert bounding box coordinates to x\_min, y\_min, x\_max, y\_max

x\_min = x\_center - (width / 2)

y\_min = y\_center - (height / 2)

x\_max = x\_center + (width / 2)

y\_max = y\_center + (height / 2)

เราก็ได้ make\_annotation.py มา เพื่อแปลงให้เข้ากับที่ hunglc007 จะใช้

แล้วก็ไปแก้ core/config.py ตามด้านบน คือ classes กับ annot.path

!python save\_model.py ก็ได้มาแบบนี้

2023-10-23 10:37:03.230060: W tensorflow/compiler/tf2tensorrt/utils/py\_utils.cc:38] TF-TRT Warning: Could not find TensorRT

Model: "model"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param # Connected to

==================================================================================================

input\_1 (InputLayer) [(None, 416, 416, 3)] 0 []

zero\_padding2d (ZeroPaddin (None, 417, 417, 3) 0 ['input\_1[0][0]']

g2D)

conv2d (Conv2D) (None, 208, 208, 32) 864 ['zero\_padding2d[0][0]']

batch\_normalization (Batch (None, 208, 208, 32) 128 ['conv2d[0][0]']

Normalization)

tf.nn.leaky\_relu (TFOpLamb (None, 208, 208, 32) 0 ['batch\_normalization[0][0]']

da)

zero\_padding2d\_1 (ZeroPadd (None, 209, 209, 32) 0 ['tf.nn.leaky\_relu[0][0]']

ing2D)

conv2d\_1 (Conv2D) (None, 104, 104, 64) 18432 ['zero\_padding2d\_1[0][0]']

batch\_normalization\_1 (Bat (None, 104, 104, 64) 256 ['conv2d\_1[0][0]']

|

|

|

|

tf.reshape\_7 (TFOpLambda) (None, None, 2) 0 ['tf.compat.v1.boolean\_mask\_1[

0][0]',

'tf.\_\_operators\_\_.getitem\_4[0

][0]',

'tf.\_\_operators\_\_.getitem\_5[0

][0]']

tf.concat\_12 (TFOpLambda) (None, None, 6) 0 ['tf.concat\_11[0][0]',

'tf.reshape\_7[0][0]']

==================================================================================================

Total params: 5882634 (22.44 MB)

Trainable params: 5876426 (22.42 MB)

Non-trainable params: 6208 (24.25 KB)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile\_metrics` will be empty until you train or evaluate the model.

W1023 10:37:06.171170 138618692243456 saving\_utils.py:359] Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile\_metrics` will be empty until you train or evaluate the model.

I1023 10:37:08.477545 138618692243456 signature\_serialization.py:148] Function `\_wrapped\_model` contains input name(s) resource, 2407, 2417 with unsupported characters which will be renamed to model\_conv2d\_20\_biasadd\_readvariableop\_resource, model\_2407, model\_2417 in the SavedModel.

I1023 10:37:09.839528 138618692243456 save.py:274] Found untraced functions such as

\_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op

while saving (showing 5 of 21).

These functions will not be directly callable after loading.

INFO:tensorflow:Assets written to: ../checkpoints/yolov4-tiny-416/assets

I1023 10:37:11.237805 138618692243456 builder\_impl.py:804] Assets written to: ../checkpoints/yolov4-tiny-416/assets

I1023 10:37:11.294840 138618692243456 fingerprinting\_utils.py:48] Writing fingerprint to ../checkpoints/yolov4-tiny-416/fingerprint.pb

3749

import os

len([f for f in os.listdir(myPath)

     if f.endswith('.jpg') and os.path.isfile(os.path.join(myPath, f))])

สิ่งที่ไม่ใช้ แต่เก็บไว้ก่อน เพราะ hunglc007 จะแปลงให้เองใน Dataset.py

# Convert bounding box coordinates to x\_min, y\_min, x\_max, y\_max

#x\_min = x\_center - (width / 2)

#y\_min = y\_center - (height / 2)

#x\_max = x\_center + (width / 2)

#y\_max = y\_center + (height / 2)

เทรนโมเดลแล้วโว้ย

เริ่มจากใส่ดาต้าเข้าไปก่อน

model.load\_weights(FLAGS.weights)

File "/usr/local/lib/python3.10/dist-packages/keras/src/utils/traceback\_utils.py", line 70, in error\_handler

raise e.with\_traceback(filtered\_tb) from None

File "/usr/local/lib/python3.10/dist-packages/tensorflow/python/ops/resource\_variable\_ops.py", line 760, in \_restore\_from\_tensors

raise ValueError(

ValueError: Received incompatible tensor with shape (128,) when attempting to restore variable with shape (64,) and name batch\_normalization\_2/beta:0.

WARNING:tensorflow:Detecting that an object or model or tf.train.Checkpoint is being deleted with unrestored values. See the following logs for the specific values in question. To silence these warnings, use `status.expect\_partial()`.

See [https://www.tensorflow.org/api\_docs/python/tf/train/Checkpoint#restorefor](https://www.tensorflow.org/api_docs/python/tf/train/Checkpoint" \l "restorefor) details about the status object returned by the restore function.

W1023 21:53:56.688521 132012375605248 checkpoint.py:208] Detecting that an object or model or tf.train.Checkpoint is being deleted with unrestored values.

See the following logs for the specific values in question. To silence these warnings, use `status.expect\_partial()`.

See [https://www.tensorflow.org/api\_docs/python/tf/train/Checkpoint#restorefor](https://www.tensorflow.org/api_docs/python/tf/train/Checkpoint" \l "restorefor) details about the status object returned by the restore function.

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.gamma

W1023 21:53:56.688831 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.gamma

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.beta

W1023 21:53:56.688927 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the re

int could not be found in the restored object: (root).layer\_with\_weights-11.beta

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.moving\_mean

W1023 21:53:56.689005 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.moving\_mean

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.moving\_variance

W1023 21:53:56.689080 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-11.moving\_variance

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-12.kernel

W1023 21:53:56.689153 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-12.kernel

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-29.gamma

W1023 21:53:56.689225 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-29.gamma

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-29.beta

W1023 21:53:56.689296 132012375605248 checkpoint.py:217] Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-29.beta

WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).layer\_with\_weights-29.moving\_mean

241023

!python save\_model.py --weights ../yolov4-tiny.weights --output ../checkpoints/yolov4-tiny-416 --input\_size 416 --model yolov4 --tiny

2023-10-23 21:46:04.584915: E tensorflow/compiler/xla/stream\_executor/cuda/cuda\_dnn.cc:9342] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered

2023-10-23 21:46:04.584976: E tensorflow/compiler/xla/stream\_executor/cuda/cuda\_fft.cc:609] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT when one has already been registered

2023-10-23 21:46:04.585009: E tensorflow/compiler/xla/stream\_executor/cuda/cuda\_blas.cc:1518] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered

2023-10-23 21:46:04.595153: I tensorflow/core/platform/cpu\_feature\_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.

To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.

2023-10-23 21:46:06.161348: W tensorflow/compiler/tf2tensorrt/utils/py\_utils.cc:38] TF-TRT Warning: Could not find TensorRT

Model: "model"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param # Connected to

==================================================================================================

input\_1 (InputLayer) [(None, 416, 416, 3)] 0 []

zero\_padding2d (ZeroPaddin (None, 417, 417, 3) 0 ['input\_1[0][0]']

g2D)

conv2d (Conv2D) (None, 208, 208, 32) 864 ['zero\_padding2d[0][0]']

batch\_normalization (Batch (None, 208, 208, 32) 128 ['conv2d[0][0]']

Normalization)

tf.nn.leaky\_relu (TFOpLamb (None, 208, 208, 32) 0 ['batch\_normalization[0][0]']

da)

zero\_padding2d\_1 (ZeroPadd (None, 209, 209, 32) 0 ['tf.nn.leaky\_relu[0][0]']

ing2D)

conv2d\_1 (Conv2D) (None, 104, 104, 64) 18432 ['zero\_padding2d\_1[0][0]']

batch\_normalization\_1 (Bat (None, 104, 104, 64) 256 ['conv2d\_1[0][0]']

chNormalization)

tf.nn.leaky\_relu\_1 (TFOpLa (None, 104, 104, 64) 0 ['batch\_normalization\_1[0][0]'

mbda) ]

conv2d\_2 (Conv2D) (None, 104, 104, 64) 36864 ['tf.nn.leaky\_relu\_1[0][0]']

batch\_normalization\_2 (Bat (None, 104, 104, 64) 256 ['conv2d\_2[0][0]']

chNormalization)

tf.nn.leaky\_relu\_2 (TFOpLa (None, 104, 104, 64) 0 ['batch\_normalization\_2[0][0]'

mbda) ]

tf.split (TFOpLambda) [(None, 104, 104, 32), 0 ['tf.nn.leaky\_relu\_2[0][0]']

(None, 104, 104, 32)]

conv2d\_3 (Conv2D) (None, 104, 104, 32) 9216 ['tf.split[0][1]']

batch\_normalization\_3 (Bat (None, 104, 104, 32) 128 ['conv2d\_3[0][0]']

chNormalization)

tf.nn.leaky\_relu\_3 (TFOpLa (None, 104, 104, 32) 0 ['batch\_normalization\_3[0][0]'

mbda) ]

conv2d\_4 (Conv2D) (None, 104, 104, 32) 9216 ['tf.nn.leaky\_relu\_3[0][0]']

batch\_normalization\_4 (Bat (None, 104, 104, 32) 128 ['conv2d\_4[0][0]']

chNormalization)

tf.nn.leaky\_relu\_4 (TFOpLa (None, 104, 104, 32) 0 ['batch\_normalization\_4[0][0]'

mbda) ]

tf.concat (TFOpLambda) (None, 104, 104, 64) 0 ['tf.nn.leaky\_relu\_4[0][0]',

'tf.nn.leaky\_relu\_3[0][0]']

conv2d\_5 (Conv2D) (None, 104, 104, 64) 4096 ['tf.concat[0][0]']

batch\_normalization\_5 (Bat (None, 104, 104, 64) 256 ['conv2d\_5[0][0]']

chNormalization)

tf.nn.leaky\_relu\_5 (TFOpLa (None, 104, 104, 64) 0 ['batch\_normalization\_5[0][0]'

mbda) ]

tf.concat\_1 (TFOpLambda) (None, 104, 104, 128) 0 ['tf.nn.leaky\_relu\_2[0][0]',

'tf.nn.leaky\_relu\_5[0][0]']

max\_pooling2d (MaxPooling2 (None, 52, 52, 128) 0 ['tf.concat\_1[0][0]']

D)

conv2d\_6 (Conv2D) (None, 52, 52, 128) 147456 ['max\_pooling2d[0][0]']

batch\_normalization\_6 (Bat (None, 52, 52, 128) 512 ['conv2d\_6[0][0]']

chNormalization)

tf.nn.leaky\_relu\_6 (TFOpLa (None, 52, 52, 128) 0 ['batch\_normalization\_6[0][0]'

mbda) ]

tf.split\_1 (TFOpLambda) [(None, 52, 52, 64), 0 ['tf.nn.leaky\_relu\_6[0][0]']

(None, 52, 52, 64)]

conv2d\_7 (Conv2D) (None, 52, 52, 64) 36864 ['tf.split\_1[0][1]']

batch\_normalization\_7 (Bat (None, 52, 52, 64) 256 ['conv2d\_7[0][0]']

chNormalization)

tf.nn.leaky\_relu\_7 (TFOpLa (None, 52, 52, 64) 0 ['batch\_normalization\_7[0][0]'

mbda) ]

conv2d\_8 (Conv2D) (None, 52, 52, 64) 36864 ['tf.nn.leaky\_relu\_7[0][0]']

batch\_normalization\_8 (Bat (None, 52, 52, 64) 256 ['conv2d\_8[0][0]']

chNormalization)

tf.nn.leaky\_relu\_8 (TFOpLa (None, 52, 52, 64) 0 ['batch\_normalization\_8[0][0]'

mbda) ]

tf.concat\_2 (TFOpLambda) (None, 52, 52, 128) 0 ['tf.nn.leaky\_relu\_8[0][0]',

'tf.nn.leaky\_relu\_7[0][0]']

conv2d\_9 (Conv2D) (None, 52, 52, 128) 16384 ['tf.concat\_2[0][0]']

batch\_normalization\_9 (Bat (None, 52, 52, 128) 512 ['conv2d\_9[0][0]']

chNormalization)

tf.nn.leaky\_relu\_9 (TFOpLa (None, 52, 52, 128) 0 ['batch\_normalization\_9[0][0]'

mbda) ]

tf.concat\_3 (TFOpLambda) (None, 52, 52, 256) 0 ['tf.nn.leaky\_relu\_6[0][0]',

'tf.nn.leaky\_relu\_9[0][0]']

max\_pooling2d\_1 (MaxPoolin (None, 26, 26, 256) 0 ['tf.concat\_3[0][0]']

g2D)

conv2d\_10 (Conv2D) (None, 26, 26, 256) 589824 ['max\_pooling2d\_1[0][0]']

batch\_normalization\_10 (Ba (None, 26, 26, 256) 1024 ['conv2d\_10[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_10 (TFOpL (None, 26, 26, 256) 0 ['batch\_normalization\_10[0][0]

ambda) ']

tf.split\_2 (TFOpLambda) [(None, 26, 26, 128), 0 ['tf.nn.leaky\_relu\_10[0][0]']

(None, 26, 26, 128)]

conv2d\_11 (Conv2D) (None, 26, 26, 128) 147456 ['tf.split\_2[0][1]']

batch\_normalization\_11 (Ba (None, 26, 26, 128) 512 ['conv2d\_11[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_11 (TFOpL (None, 26, 26, 128) 0 ['batch\_normalization\_11[0][0]

ambda) ']

conv2d\_12 (Conv2D) (None, 26, 26, 128) 147456 ['tf.nn.leaky\_relu\_11[0][0]']

batch\_normalization\_12 (Ba (None, 26, 26, 128) 512 ['conv2d\_12[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_12 (TFOpL (None, 26, 26, 128) 0 ['batch\_normalization\_12[0][0]

ambda) ']

tf.concat\_4 (TFOpLambda) (None, 26, 26, 256) 0 ['tf.nn.leaky\_relu\_12[0][0]',

'tf.nn.leaky\_relu\_11[0][0]']

conv2d\_13 (Conv2D) (None, 26, 26, 256) 65536 ['tf.concat\_4[0][0]']

batch\_normalization\_13 (Ba (None, 26, 26, 256) 1024 ['conv2d\_13[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_13 (TFOpL (None, 26, 26, 256) 0 ['batch\_normalization\_13[0][0]

ambda) ']

tf.concat\_5 (TFOpLambda) (None, 26, 26, 512) 0 ['tf.nn.leaky\_relu\_10[0][0]',

'tf.nn.leaky\_relu\_13[0][0]']

max\_pooling2d\_2 (MaxPoolin (None, 13, 13, 512) 0 ['tf.concat\_5[0][0]']

g2D)

conv2d\_14 (Conv2D) (None, 13, 13, 512) 2359296 ['max\_pooling2d\_2[0][0]']

batch\_normalization\_14 (Ba (None, 13, 13, 512) 2048 ['conv2d\_14[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_14 (TFOpL (None, 13, 13, 512) 0 ['batch\_normalization\_14[0][0]

ambda) ']

conv2d\_15 (Conv2D) (None, 13, 13, 256) 131072 ['tf.nn.leaky\_relu\_14[0][0]']

batch\_normalization\_15 (Ba (None, 13, 13, 256) 1024 ['conv2d\_15[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_15 (TFOpL (None, 13, 13, 256) 0 ['batch\_normalization\_15[0][0]

ambda) ']

conv2d\_18 (Conv2D) (None, 13, 13, 128) 32768 ['tf.nn.leaky\_relu\_15[0][0]']

batch\_normalization\_17 (Ba (None, 13, 13, 128) 512 ['conv2d\_18[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_17 (TFOpL (None, 13, 13, 128) 0 ['batch\_normalization\_17[0][0]

ambda) ']

tf.image.resize (TFOpLambd (None, 26, 26, 128) 0 ['tf.nn.leaky\_relu\_17[0][0]']

a)

tf.concat\_6 (TFOpLambda) (None, 26, 26, 384) 0 ['tf.image.resize[0][0]',

'tf.nn.leaky\_relu\_13[0][0]']

conv2d\_19 (Conv2D) (None, 26, 26, 256) 884736 ['tf.concat\_6[0][0]']

conv2d\_16 (Conv2D) (None, 13, 13, 512) 1179648 ['tf.nn.leaky\_relu\_15[0][0]']

batch\_normalization\_18 (Ba (None, 26, 26, 256) 1024 ['conv2d\_19[0][0]']

tchNormalization)

batch\_normalization\_16 (Ba (None, 13, 13, 512) 2048 ['conv2d\_16[0][0]']

tchNormalization)

tf.nn.leaky\_relu\_18 (TFOpL (None, 26, 26, 256) 0 ['batch\_normalization\_18[0][0]

ambda) ']

tf.nn.leaky\_relu\_16 (TFOpL (None, 13, 13, 512) 0 ['batch\_normalization\_16[0][0]

ambda) ']

conv2d\_20 (Conv2D) (None, 26, 26, 21) 5397 ['tf.nn.leaky\_relu\_18[0][0]']

conv2d\_17 (Conv2D) (None, 13, 13, 21) 10773 ['tf.nn.leaky\_relu\_16[0][0]']

tf.compat.v1.shape (TFOpLa (4,) 0 ['conv2d\_20[0][0]']

mbda)

tf.compat.v1.shape\_1 (TFOp (4,) 0 ['conv2d\_17[0][0]']

Lambda)

tf.\_\_operators\_\_.getitem ( () 0 ['tf.compat.v1.shape[0][0]']

SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_1 () 0 ['tf.compat.v1.shape\_1[0][0]']

(SlicingOpLambda)

tf.reshape (TFOpLambda) (None, 26, 26, 3, 7) 0 ['conv2d\_20[0][0]',

'tf.\_\_operators\_\_.getitem[0][

0]']

tf.reshape\_3 (TFOpLambda) (None, 13, 13, 3, 7) 0 ['conv2d\_17[0][0]',

'tf.\_\_operators\_\_.getitem\_1[0

][0]']

tf.split\_3 (TFOpLambda) [(None, 26, 26, 3, 2), 0 ['tf.reshape[0][0]']

(None, 26, 26, 3, 2),

(None, 26, 26, 3, 1),

(None, 26, 26, 3, 2)]

tf.split\_4 (TFOpLambda) [(None, 13, 13, 3, 2), 0 ['tf.reshape\_3[0][0]']

(None, 13, 13, 3, 2),

(None, 13, 13, 3, 1),

(None, 13, 13, 3, 2)]

tf.math.sigmoid (TFOpLambd (None, 26, 26, 3, 2) 0 ['tf.split\_3[0][0]']

a)

tf.math.sigmoid\_3 (TFOpLam (None, 13, 13, 3, 2) 0 ['tf.split\_4[0][0]']

bda)

tf.math.multiply (TFOpLamb (None, 26, 26, 3, 2) 0 ['tf.math.sigmoid[0][0]']

da)

tf.tile (TFOpLambda) (None, 26, 26, 3, 2) 0 ['tf.\_\_operators\_\_.getitem[0][

0]']

tf.math.multiply\_4 (TFOpLa (None, 13, 13, 3, 2) 0 ['tf.math.sigmoid\_3[0][0]']

mbda)

tf.tile\_1 (TFOpLambda) (None, 13, 13, 3, 2) 0 ['tf.\_\_operators\_\_.getitem\_1[0

][0]']

tf.math.subtract (TFOpLamb (None, 26, 26, 3, 2) 0 ['tf.math.multiply[0][0]']

da)

tf.cast (TFOpLambda) (None, 26, 26, 3, 2) 0 ['tf.tile[0][0]']

tf.math.subtract\_1 (TFOpLa (None, 13, 13, 3, 2) 0 ['tf.math.multiply\_4[0][0]']

mbda)

tf.cast\_1 (TFOpLambda) (None, 13, 13, 3, 2) 0 ['tf.tile\_1[0][0]']

tf.math.sigmoid\_1 (TFOpLam (None, 26, 26, 3, 1) 0 ['tf.split\_3[0][2]']

bda)

tf.math.sigmoid\_2 (TFOpLam (None, 26, 26, 3, 2) 0 ['tf.split\_3[0][3]']

bda)

tf.math.sigmoid\_4 (TFOpLam (None, 13, 13, 3, 1) 0 ['tf.split\_4[0][2]']

bda)

tf.math.sigmoid\_5 (TFOpLam (None, 13, 13, 3, 2) 0 ['tf.split\_4[0][3]']

bda)

tf.\_\_operators\_\_.add (TFOp (None, 26, 26, 3, 2) 0 ['tf.math.subtract[0][0]',

Lambda) 'tf.cast[0][0]']

tf.math.exp (TFOpLambda) (None, 26, 26, 3, 2) 0 ['tf.split\_3[0][1]']

tf.\_\_operators\_\_.add\_1 (TF (None, 13, 13, 3, 2) 0 ['tf.math.subtract\_1[0][0]',

OpLambda) 'tf.cast\_1[0][0]']

tf.math.exp\_1 (TFOpLambda) (None, 13, 13, 3, 2) 0 ['tf.split\_4[0][1]']

tf.math.multiply\_3 (TFOpLa (None, 26, 26, 3, 2) 0 ['tf.math.sigmoid\_1[0][0]',

mbda) 'tf.math.sigmoid\_2[0][0]']

tf.math.multiply\_7 (TFOpLa (None, 13, 13, 3, 2) 0 ['tf.math.sigmoid\_4[0][0]',

mbda) 'tf.math.sigmoid\_5[0][0]']

tf.math.multiply\_1 (TFOpLa (None, 26, 26, 3, 2) 0 ['tf.\_\_operators\_\_.add[0][0]']

mbda)

tf.math.multiply\_2 (TFOpLa (None, 26, 26, 3, 2) 0 ['tf.math.exp[0][0]']

mbda)

tf.math.multiply\_5 (TFOpLa (None, 13, 13, 3, 2) 0 ['tf.\_\_operators\_\_.add\_1[0][0]

mbda) ']

tf.math.multiply\_6 (TFOpLa (None, 13, 13, 3, 2) 0 ['tf.math.exp\_1[0][0]']

mbda)

tf.reshape\_1 (TFOpLambda) (None, None, 2) 0 ['tf.math.multiply\_3[0][0]',

'tf.\_\_operators\_\_.getitem[0][

0]']

tf.reshape\_4 (TFOpLambda) (None, None, 2) 0 ['tf.math.multiply\_7[0][0]',

'tf.\_\_operators\_\_.getitem\_1[0

][0]']

tf.concat\_7 (TFOpLambda) (None, 26, 26, 3, 4) 0 ['tf.math.multiply\_1[0][0]',

'tf.math.multiply\_2[0][0]']

tf.concat\_8 (TFOpLambda) (None, 13, 13, 3, 4) 0 ['tf.math.multiply\_5[0][0]',

'tf.math.multiply\_6[0][0]']

tf.concat\_10 (TFOpLambda) (None, None, 2) 0 ['tf.reshape\_1[0][0]',

'tf.reshape\_4[0][0]']

tf.reshape\_2 (TFOpLambda) (None, None, 4) 0 ['tf.concat\_7[0][0]',

'tf.\_\_operators\_\_.getitem[0][

0]']

tf.reshape\_5 (TFOpLambda) (None, None, 4) 0 ['tf.concat\_8[0][0]',

'tf.\_\_operators\_\_.getitem\_1[0

][0]']

tf.math.reduce\_max (TFOpLa (None, None) 0 ['tf.concat\_10[0][0]']

mbda)

tf.concat\_9 (TFOpLambda) (None, None, 4) 0 ['tf.reshape\_2[0][0]',

'tf.reshape\_5[0][0]']

tf.math.greater\_equal (TFO (None, None) 0 ['tf.math.reduce\_max[0][0]']

pLambda)

tf.compat.v1.boolean\_mask (None, 4) 0 ['tf.concat\_9[0][0]',

(SlicingOpLambda) 'tf.math.greater\_equal[0][0]'

]

tf.compat.v1.shape\_2 (TFOp (3,) 0 ['tf.concat\_10[0][0]']

Lambda)

tf.compat.v1.shape\_3 (TFOp (2,) 0 ['tf.compat.v1.boolean\_mask[0]

Lambda) [0]']

tf.\_\_operators\_\_.getitem\_2 () 0 ['tf.compat.v1.shape\_2[0][0]']

(SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_3 () 0 ['tf.compat.v1.shape\_3[0][0]']

(SlicingOpLambda)

tf.reshape\_6 (TFOpLambda) (None, None, 4) 0 ['tf.compat.v1.boolean\_mask[0]

[0]',

'tf.\_\_operators\_\_.getitem\_2[0

][0]',

'tf.\_\_operators\_\_.getitem\_3[0

][0]']

tf.split\_5 (TFOpLambda) [(None, None, 2), 0 ['tf.reshape\_6[0][0]']

(None, None, 2)]

tf.\_\_operators\_\_.getitem\_7 (None, None, 2) 0 ['tf.split\_5[0][1]']

(SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_6 (None, None, 2) 0 ['tf.split\_5[0][0]']

(SlicingOpLambda)

tf.math.truediv (TFOpLambd (None, None, 2) 0 ['tf.\_\_operators\_\_.getitem\_7[0

a) ][0]']

tf.math.truediv\_2 (TFOpLam (None, None, 2) 0 ['tf.\_\_operators\_\_.getitem\_7[0

bda) ][0]']

tf.math.subtract\_2 (TFOpLa (None, None, 2) 0 ['tf.\_\_operators\_\_.getitem\_6[0

mbda) ][0]',

'tf.math.truediv[0][0]']

tf.\_\_operators\_\_.add\_2 (TF (None, None, 2) 0 ['tf.\_\_operators\_\_.getitem\_6[0

OpLambda) ][0]',

'tf.math.truediv\_2[0][0]']

tf.compat.v1.boolean\_mask\_ (None, 2) 0 ['tf.concat\_10[0][0]',

1 (SlicingOpLambda) 'tf.math.greater\_equal[0][0]'

]

tf.math.truediv\_1 (TFOpLam (None, None, 2) 0 ['tf.math.subtract\_2[0][0]']

bda)

tf.math.truediv\_3 (TFOpLam (None, None, 2) 0 ['tf.\_\_operators\_\_.add\_2[0][0]

bda) ']

tf.compat.v1.shape\_4 (TFOp (3,) 0 ['tf.concat\_10[0][0]']

Lambda)

tf.compat.v1.shape\_5 (TFOp (2,) 0 ['tf.compat.v1.boolean\_mask\_1[

Lambda) 0][0]']

tf.\_\_operators\_\_.getitem\_8 (None, None, 1) 0 ['tf.math.truediv\_1[0][0]']

(SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_9 (None, None, 1) 0 ['tf.math.truediv\_1[0][0]']

(SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_1 (None, None, 1) 0 ['tf.math.truediv\_3[0][0]']

0 (SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_1 (None, None, 1) 0 ['tf.math.truediv\_3[0][0]']

1 (SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_4 () 0 ['tf.compat.v1.shape\_4[0][0]']

(SlicingOpLambda)

tf.\_\_operators\_\_.getitem\_5 () 0 ['tf.compat.v1.shape\_5[0][0]']

(SlicingOpLambda)

tf.concat\_11 (TFOpLambda) (None, None, 4) 0 ['tf.\_\_operators\_\_.getitem\_8[0

][0]',

'tf.\_\_operators\_\_.getitem\_9[0

][0]',

'tf.\_\_operators\_\_.getitem\_10[

0][0]',

'tf.\_\_operators\_\_.getitem\_11[

0][0]']

tf.reshape\_7 (TFOpLambda) (None, None, 2) 0 ['tf.compat.v1.boolean\_mask\_1[

0][0]',

'tf.\_\_operators\_\_.getitem\_4[0

][0]',

'tf.\_\_operators\_\_.getitem\_5[0

][0]']

tf.concat\_12 (TFOpLambda) (None, None, 6) 0 ['tf.concat\_11[0][0]',

'tf.reshape\_7[0][0]']

==================================================================================================

Total params: 5882634 (22.44 MB)

Trainable params: 5876426 (22.42 MB)

Non-trainable params: 6208 (24.25 KB)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile\_metrics` will be empty until you train or evaluate the model.

W1023 21:46:09.971431 133823588032512 saving\_utils.py:359] Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile\_metrics` will be empty until you train or evaluate the model.

I1023 21:46:14.635306 133823588032512 signature\_serialization.py:156] Function `\_wrapped\_model` contains input name(s) resource, 2407, 2417 with unsupported characters which will be renamed to model\_conv2d\_20\_biasadd\_readvariableop\_resource, model\_2407, model\_2417 in the SavedModel.

I1023 21:46:16.758316 133823588032512 save.py:289] Found untraced functions such as \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op, \_jit\_compiled\_convolution\_op while saving (showing 5 of 21).

These functions will not be directly callable after loading.

INFO:tensorflow:Assets written to: ../checkpoints/yolov4-tiny-416/assets

I1023 21:46:18.777787 133823588032512 builder\_impl.py:801] Assets written to: ../checkpoints/yolov4-tiny-416/assets

I1023 21:46:18.864661 133823588032512 fingerprinting\_utils.py:47] Writing fingerprint to ../checkpoints/yolov4-tiny-416/fingerprint.pb

/usr/local/lib/python3.10/dist-packages/keras/src/initializers/\_\_init\_\_.py:144: UserWarning: The `keras.initializers.serialize()` API should only be used for objects of type `keras.initializers.Initializer`. Found an instance of type <class 'tensorflow.python.ops.init\_ops\_v2.RandomNormal'>, which may lead to improper serialization.

warnings.warn(

/usr/local/lib/python3.10/dist-packages/keras/src/initializers/\_\_init\_\_.py:144: UserWarning: The `keras.initializers.serialize()` API should only be used for objects of type `keras.initializers.Initializer`.

Found an instance of type <class 'tensorflow.python.ops.init\_ops\_v2.Constant'>, which may lead to improper serialization.

warnings.warn(

The error message you're encountering, "ValueError: Received incompatible tensor with shape (128,) when attempting to restore variable with shape (64,) and name batch\_normalization\_2/beta:0,"

indicates a shape mismatch between the variable stored in your checkpoint or model weights and the variable expected by the model during restoration.

This type of error typically occurs when you're trying to load weights from a model checkpoint or saved model into a model that has different layer shapes or architecture. In this specific case, it appears to be related to the batch normalization layer.

batch\_normalization\_2 (Bat (None, 208, 208, 64) 256 ['conv2d\_2[0][0]']

If there's a mismatch between the model architecture and the loaded weights, you may encounter errors like the one you mentioned earlier. It's important to ensure that the

model and weights are aligned in terms of

architecture,

layer names, and

variable shapes.

Also tensorflow/keras version

If you're working with a pre-trained model or trying to fine-tune a model, make sure you're using the correct pre-trained weights for that specific model architecture. When creating a custom model, ensure that the model architecture is consistent with the weights you intend to load.

So it seems that hunglc007 save\_model.py is doing a tf.save\_model model which means

A SavedModel contains a complete TensorFlow program, including trained parameters (i.e, [tf.Variable](https://www.tensorflow.org/api_docs/python/tf/Variable)s) and computation. It does not require the original model building code to run, which makes it useful for sharing or deploying with [TFLite](https://tensorflow.org/lite), [TensorFlow.js](https://js.tensorflow.org/), [TensorFlow Serving](https://www.tensorflow.org/tfx/serving/tutorials/Serving_REST_simple), or [TensorFlow Hub](https://tensorflow.org/hub).

พอไปเอา yolov4-tiny-custom\_last.weights ที่ draw bbox but doesn’t draw label มาแล้ว ก็ลองมา save\_model และ python detect.py ดู ก็ได้ error มาแบบนี้ คิดว่าคงเป็นที่ annotation เหมือนเดิม

lite/core/utils.py", line 153, in draw\_bbox

cv2.rectangle(image, c1, c2, bbox\_color, bbox\_thick)

cv2.error: OpenCV(4.8.0) :-1: error: (-5:Bad argument) in function 'rectangle'

> Overload resolution failed:

> - Can't parse 'pt1'. Sequence item with index 0 has a wrong type

> - Can't parse 'pt1'. Sequence item with index 0 has a wrong type

> - Can't parse 'rec'. Expected sequence length 4, got 2

> - Can't parse 'rec'. Expected sequence length 4, got 2

coor = out\_boxes[0][i]

coor[0] = int(coor[0] \* image\_h)

coor[2] = int(coor[2] \* image\_h)

coor[1] = int(coor[1] \* image\_w)

coor[3] = int(coor[3] \* image\_w)

fontScale = 0.5

score = out\_scores[0][i]

class\_ind = int(out\_classes[0][i])

bbox\_color = colors[class\_ind]

bbox\_thick = int(0.6 \* (image\_h + image\_w) / 600)

c1, c2 = (coor[1], coor[0]), (coor[3], coor[2])

cv2.rectangle(image, c1, c2, bbox\_color, bbox\_thick)

if show\_label:

bbox\_mess = '%s: %.2f' % (classes[class\_ind], score)

t\_size = cv2.getTextSize(bbox\_mess, 0, fontScale, thickness=bbox\_thick // 2)[0]

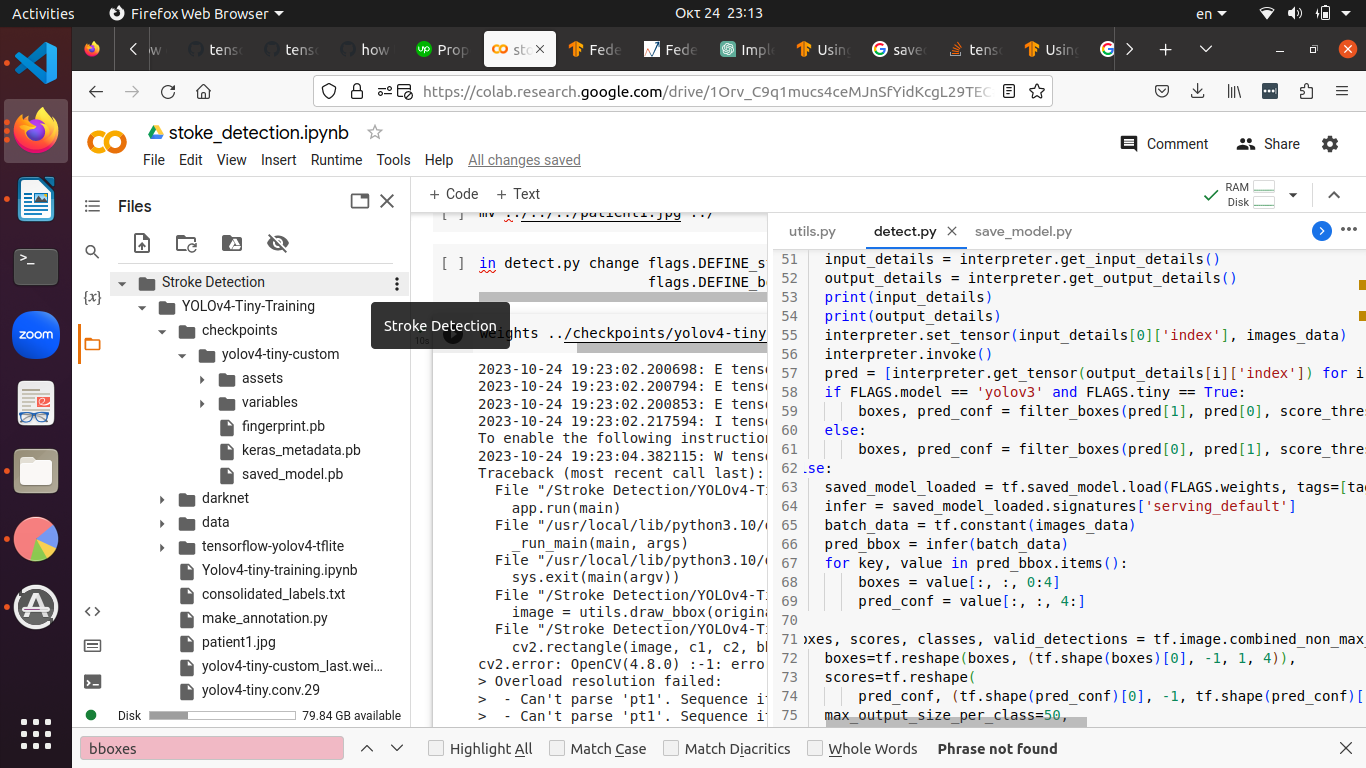
c3 = (c1[0] + t\_size[0], c1[1] - t\_size[1] - 3)

cv2.rectangle(image, c1, (np.float32(c3[0]), np.float32(c3[1])), bbox\_color, -1) #filled

cv2.putText(image, bbox\_mess, (c1[0], np.float32(c1[1] - 2)), cv2.FONT\_HERSHEY\_SIMPLEX,

fontScale, (0, 0, 0), bbox\_thick // 2, lineType=cv2.LINE\_AA)

return image



saved\_model\_loaded = tf.saved\_model.load(FLAGS.weights, tags=[tag\_constants.SERVING])

infer = saved\_model\_loaded.signatures['serving\_default']

batch\_data = tf.constant(images\_data)

pred\_bbox = infer(batch\_data)

for key, value in pred\_bbox.items():

boxes = value[:, :, 0:4]

pred\_conf = value[:, :, 4:]

boxes, scores, classes, valid\_detections = tf.image.combined\_non\_max\_suppression(

boxes=tf.reshape(boxes, (tf.shape(boxes)[0], -1, 1, 4)),

scores=tf.reshape(

pred\_conf, (tf.shape(pred\_conf)[0], -1, tf.shape(pred\_conf)[-1])),

max\_output\_size\_per\_class=50,

max\_total\_size=50,

iou\_threshold=FLAGS.iou,

score\_threshold=FLAGS.score

)

pred\_bbox = [boxes.numpy(), scores.numpy(), classes.numpy(), valid\_detections.numpy()]

image = utils.draw\_bbox(original\_image, pred\_bbox)

# image = utils.draw\_bbox(image\_data\*255, pred\_bbox)

image = Image.fromarray(image.astype(np.uint8))

image.show()

image = cv2.cvtColor(np.array(image), cv2.COLOR\_BGR2RGB)

cv2.imwrite(FLAGS.output, image)

!python detect.py --weights ../checkpoints/yolov4-tiny-custom --size 416 --model yolov4 --image ../patient1.jpg --tiny

$ saved\_model\_cli show --dir \

/tmp/saved\_model\_dir --tag\_set serve --signature\_def serving\_default

The given SavedModel SignatureDef contains the following input(s):

  inputs['x'] tensor\_info:

      dtype: DT\_FLOAT

      shape: (-1, 1)

      name: x:0

The given SavedModel SignatureDef contains the following output(s):

  outputs['y'] tensor\_info:

      dtype: DT\_FLOAT

      shape: (-1, 1)

      name: y:0

Method name is: tensorflow/serving/predict

261023

โอยอีห่า ให้กูแก้โค้ด คือตีกรอบไม่ได้ ต้องเอาโค้ดที่เขาส่งมาให้ไป patch ใส่ใน darknet.py แล้วถึงจะตีตาราง แต่ก็ยังพิมพ์ label ไม่ได้อยู่ดี แต่อย่างน้อยดีเทคได้แล้ว

sed 's/jpg/txt/g' input.txt > output.txt

ตอนนี้จะลองรัน detector map เพื่อ evaluate จะต้องใส่ label.txt ไปให้มันก่อน

sed 's/jpg/txt/g' ../data/test.txt > ../data/test\_labels.txt

edit darknet.py

เขียนการสร้างโฟลเด้อสำหรับ validation ใหม่ เพื่อให้ ./darknet detector map ทำงานได้

เพื่อจะไปเปรียบเทียบว่า หลัง federated แล้วดีขึ้นจริงหรือเปล่า

งานนี้แม่งทำมาอาทิตย์นึงแล้วนะ ได้ยี่สิบยูโร ไม่อยากจะเชื่อ แต่ที่ได้คือการเซ็ทอัพแอคเค้าและรู้ว่ามันทำงานยังไง ก็คุ้มอยู่ โค่บอกให้สมัครแอคเค้าใหม่เรื่อยๆ เราบอกว่า ดีกว่าอีกเว็บนึงที่แม่งบอกว่ากูไม่แมทช์แล้วไม่ให้สมัครเลย แสรด นี่ถ้าโทเค่นไม่พอก็ให้ไปซื้อเอา มีขาย เออ ดีกว่าแมะ

ถ้าเกิดไปทำจริง ควรจะได้ตังเท่าไหร่

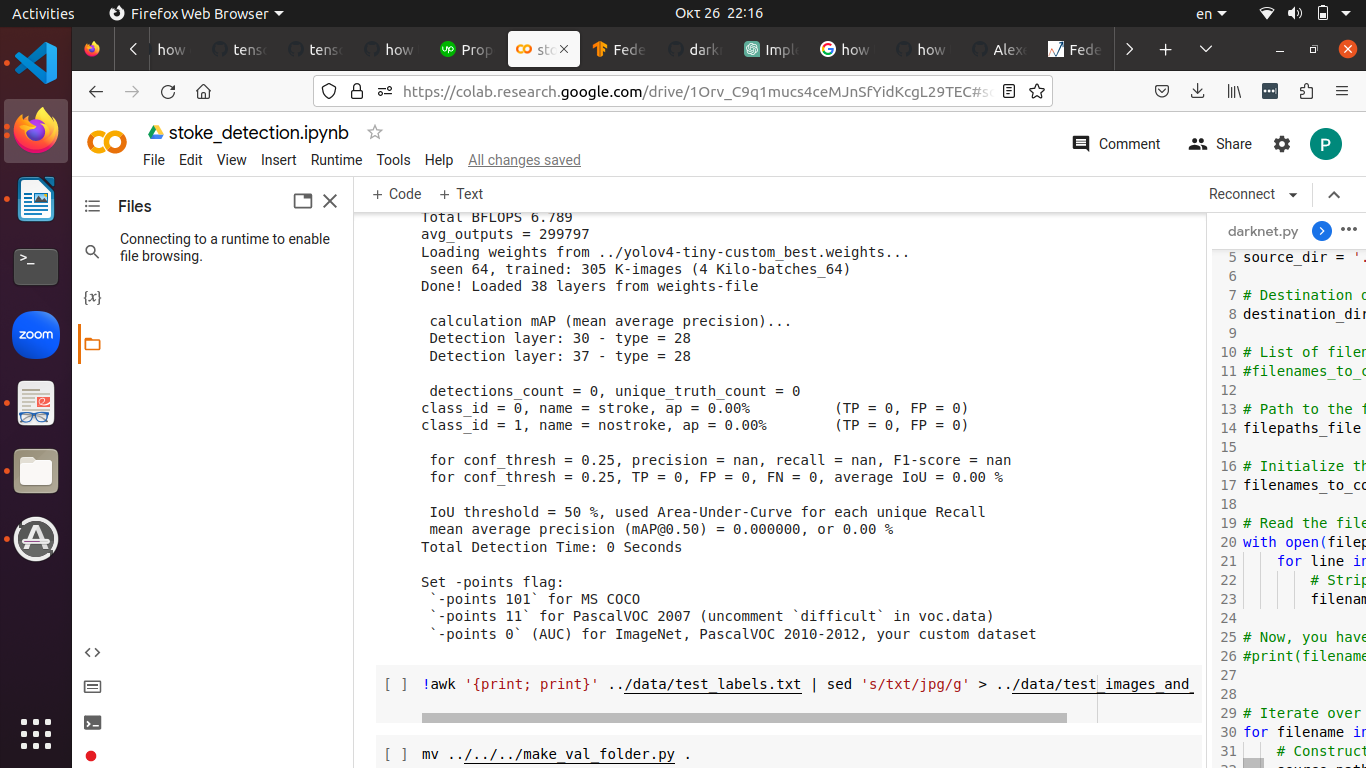
ฝึกให้กะปริมาณงานให้เป็น

นอกนั้นก็คือ ทำความเข้าใจเรื่องการใส่ไฟล์ annot.path เข้าไปใน keras.yolov4-tiny.model แหละ เป็น SavedModel คือมี weights รวมอยู่ในนั้นเลย เพราะต้องใส่เข้าไปตอน convert model ก็คือตอนเรียก save\_model.py

แล้วก็ลองไปเรียก evaluate.py ดูแล้ว ก็คือใช้ไฟล์จาก ../checkpoints/yolov4-tiny-custom ที่เราก็เป็นคนแจ้งไปเหมือนกันตอน save\_model.py

ใน checkpoint ก็มีไฟล์ตามที่อยากได้แล้ว คือ variables (data, .index) fingerprint.pb ละก็ saved\_model.pb

evaluate ได้แล้ว แต่ average precision (ap) ยังเป็น 0 อยู่



261023

obj.data that works for ./darknet detector map

!./darknet detector map ../data/obj.data cfg/yolov4-tiny-custom.cfg ../yolov4-tiny-custom\_best.weights -points 0

classes = 2

train = data/train.txt

valid = ../data/Dataset/ <--เอาโฟลเด้อที่มีไฟล์นั้นอยู่จริงๆ ไม่ใช่แค่ชื่อไฟล์

names = ../data/obj.names

backup = /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data

test.txt มีแบบนี้

content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_0379.jpg /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_0229.jpg /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_2358.jpg /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_2935.jpg /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_3487.jpg /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data/Dataset/img\_0682.jpg

แต่จะเอาแค่ img\_0682.jpg

ให้ทำแบบนี้ print เฉพาะคอลัมน์สุดท้าย

!awk -F/ '{print $NF}' test.txt > test\_filenames.txt

double each line and replace .jpg with .txt only in 1 of the 2 lines

# Define the input and output file paths

input\_file = 'test\_filenames.txt'

output\_file = 'test\_images\_and\_labels.txt'

# Open the input and output files

with open(input\_file, 'r') as input\_f, open(output\_file, 'w') as output\_f:

for line in input\_f:

# Write the original line to the output file

output\_f.write(line)

# Write the modified line (replace .jpg with .txt) to the output file

output\_f.write(line.replace(".jpg", ".txt"))

ได้ test\_images\_and\_labels.txt มา เป็นแบบนี้

img\_3692.jpg

img\_3692.txt

img\_2794.jpg

img\_2794.txt

img\_0012.jpg

img\_0012.txt

img\_0967.jpg

img\_0967.txt

แล้วค่อยเอา test\_images\_and\_labels.txt ไปใส่ใน make\_val\_folder.py

# Path to the file containing file paths

filepaths\_file = '../data/test\_images\_and\_labels.txt'

# Initialize the list to store filenames

filenames\_to\_copy = []

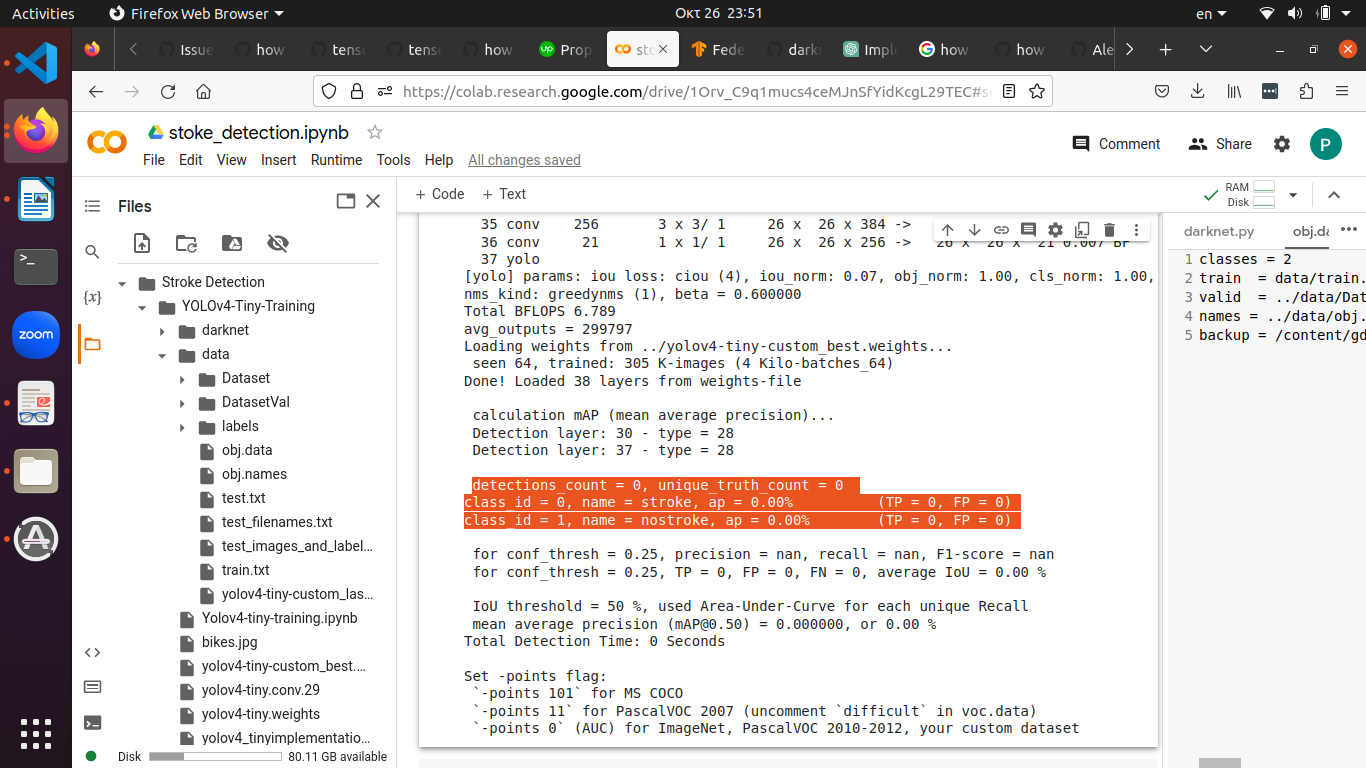
# Read the file and extract filenames

with open(filepaths\_file, 'r') as file:

for line in file:

# Strip any leading/trailing whitespace and add to the list

filenames\_to\_copy.append(line.strip())



ทำตั้งนาน แต่ ap ยังได้ 0% อยู่ดี เพราะ detections\_count = 0 อะไรๆก็=0 ไปด้วย

อาจจะต้องแก้ใน cfg

#Testing

batch=1

subdivision =1

obj.data

classes = 2

train = data/train.txt

valid = ../data/DatasetVal

names = ../data/obj.names

backup = /content/gdrive/MyDrive/Yolov4-Tiny-Training/darknet/data

input\_info = signature\_keys.inputs

output\_info = signature\_keys.outputs

print("Input Tensors:")

for name, tensor\_info in input\_info.items():

print(f"Name: {name}, Shape: {tensor\_info.shape}, Dtype: {tensor\_info.dtype}")

print("Output Tensors:")

for name, tensor\_info in output\_info.items():

print(f"Name: {name}, Shape: {tensor\_info.shape}, Dtype: {tensor\_info.dtype}")

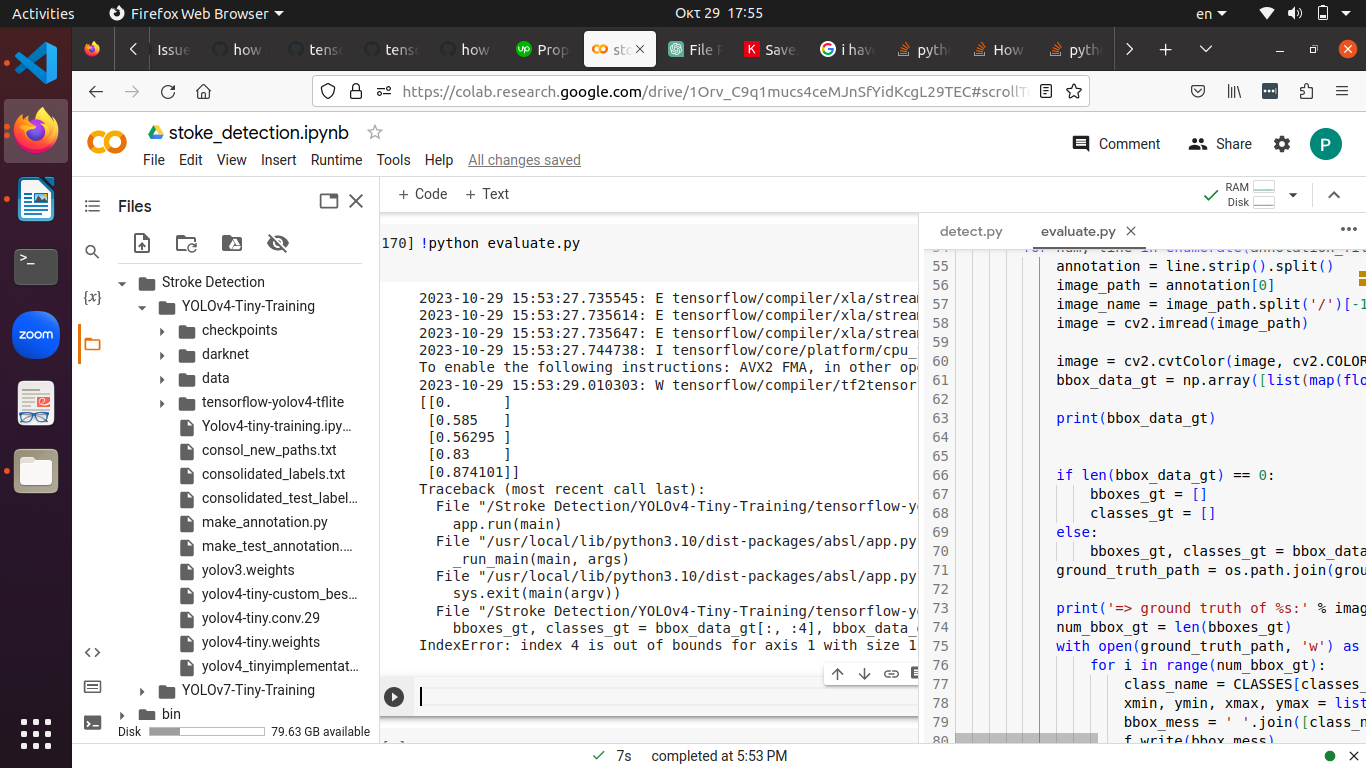
use this to check if evaluate.py can cv2.imread(image\_path) correctly

if image is not None:

print(image)

else:

print("Failed to load the image.")



[[0. ]

[0.585 ]

[0.56295 ]

[0.83 ]

[0.874101]]

Traceback (most recent call last):

File "/Stroke Detection/YOLOv4-Tiny-Training/tensorflow-yolov4-tflite/evaluate.py", line 145, in <module>

app.run(main)

File "/usr/local/lib/python3.10/dist-packages/absl/app.py", line 308, in run

\_run\_main(main, args)

File "/usr/local/lib/python3.10/dist-packages/absl/app.py", line 254, in \_run\_main

sys.exit(main(argv))

File "/Stroke Detection/YOLOv4-Tiny-Training/tensorflow-yolov4-tflite/evaluate.py", line 70, in main

bboxes\_gt, classes\_gt = bbox\_data\_gt[:, :4], bbox\_data\_gt[:, 4]

IndexError: index 4 is out of bounds for axis 1 with size 1

bbox\_data\_gt = np.array([list(map(float, box.split(','))) for box in annotation[1:]])

[[0. ]

[0.585 ]

[0.56295 ]

[0.83 ]

[0.874101]]

bboxes\_gt, classes\_gt = bbox\_data\_gt[:, :4], bbox\_data\_gt[:, 4]

IndexError: index 4 is out of bounds for axis 1 with size 1

น่าจะสลับ box กับ class\_id อันนึงมี 4 อันนึง 1

edit detect.py

edit evaluate.py

กว่าจะมีถึงจุดนนี้ได้ ต้อง

upload make\_annotation.py

ได้มาแล้ว เอาไปใส่ใน core/config.py

annot.path = (‘../consolidated\_labels.txt’)

\_\_C.TEST.annot.path ด้วย

แต่ต้องไปทำ ../data/DatasetVal ก่อน

โอย ซึ่งจะทำอันนั้น ก็ต้อง make\_val\_folder.py

ทำ test\_images\_and\_labels.txt

เพื่อที่จะมา save\_model.py

เพื่อเอา model ที่ว่านั้นไป detect และ evaluate

นี่ยังไม่ได้ไปดูเรื่อง

model.loaded\_model(weights) เลยนะ

tf.models.load\_model(weights) มันเป็น tf แต่ไม่ได้เป็น keras

ใช้ model.summary() ไม่ได้

ต้องเป็น signature key อะไรไม่รู้

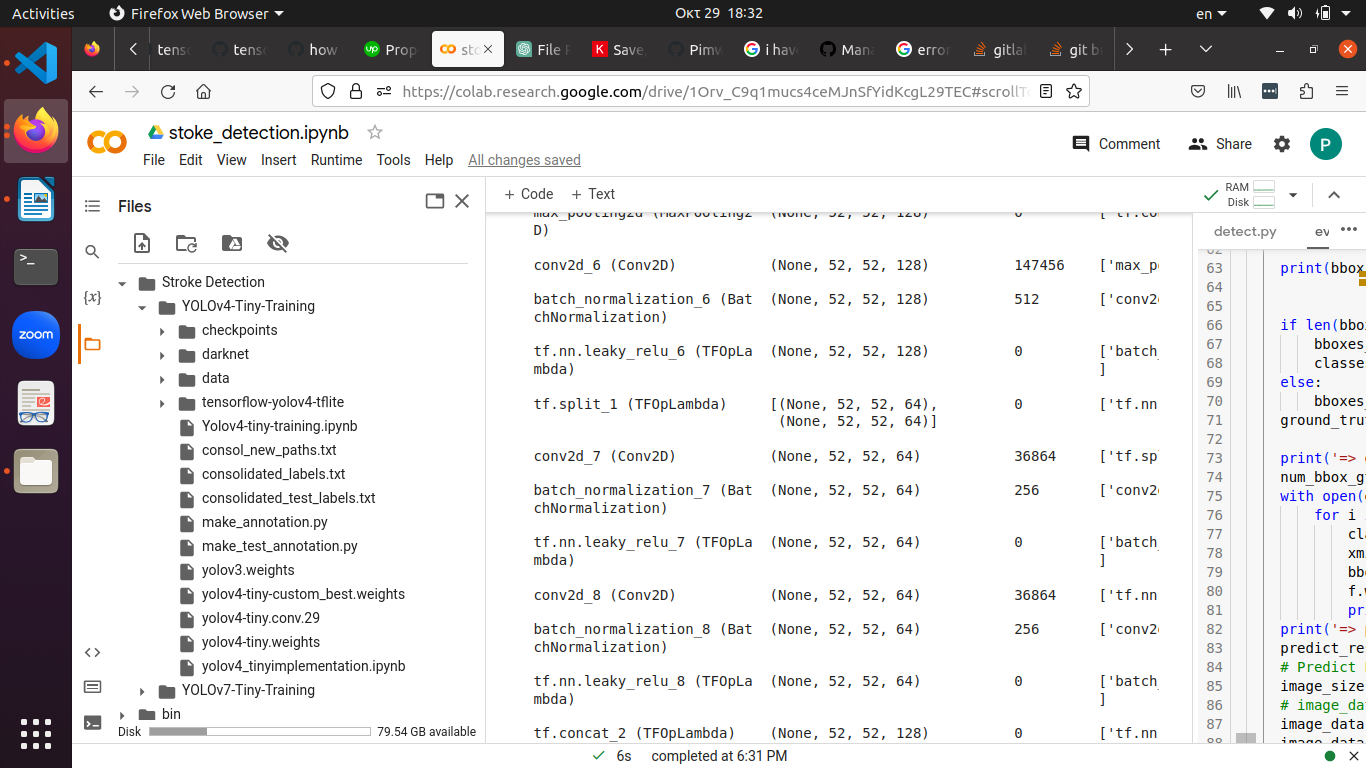
แต่เราทำให้เป็น keras model ได้ไหม ต้องทำตอนเซฟหรือตอนโหลด

tf.keras.models.save\_model() เหรอออ

ตอนนี้มีเรื่อง git submodule มา ก็คือ ทำไปตั้งเยอะแล้ว อยากเซฟทั้งหลายใส่ git ไว้ คราวหน้าเปิดมาจะได้ไม่ต้องทำอีกรอบ แปลว่า ต้อง add submodule แต่ตอนเนี้ย git add . ไปแล้ว

ต้อง remove cache ใช่ไหม

ประสบการณ์สอนไว้ว่า อย่าทำ git ตอนเหนื่อยหรือเมา ซึ่งคือตอนนี้

 รอบนี้ใช้ darknet.py alexeyAB ถึงได้เวิร์คมาไกลขนาดนีเ้