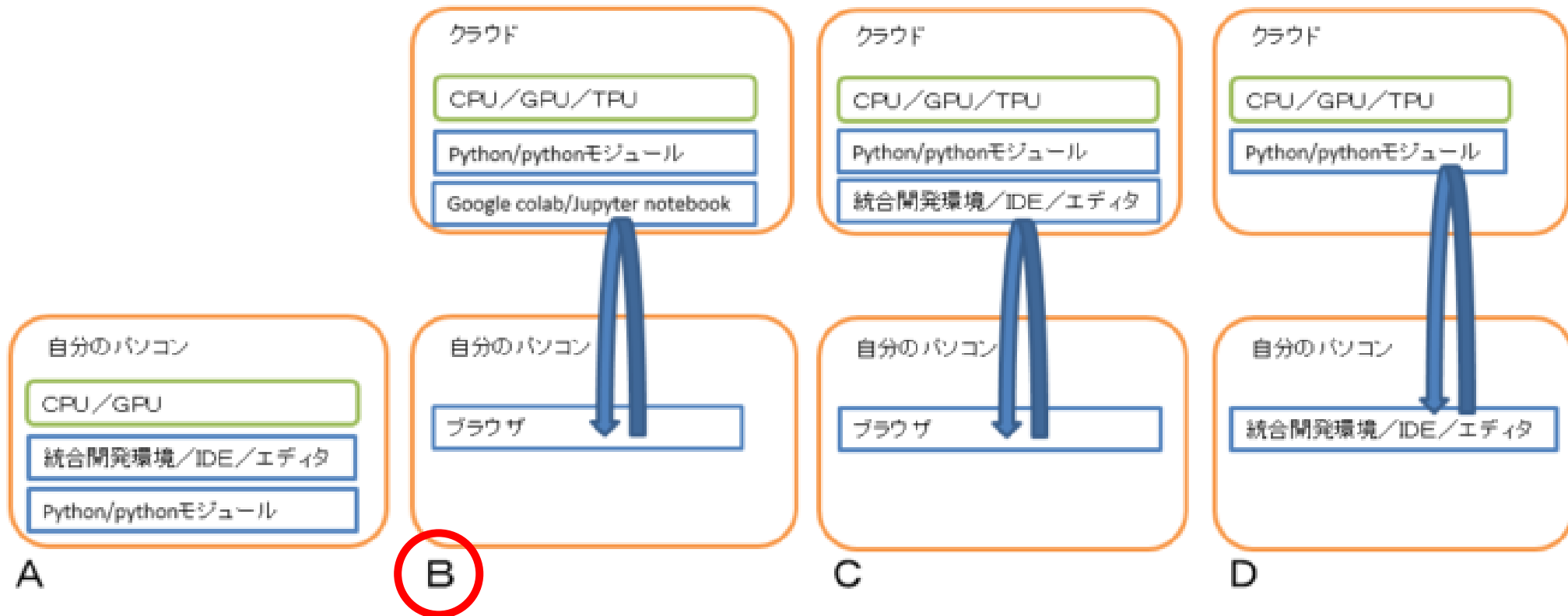


YOLO V5 をGoogle Colabで 学習・推論（検出）する方法

一関高専 未来創造工学科

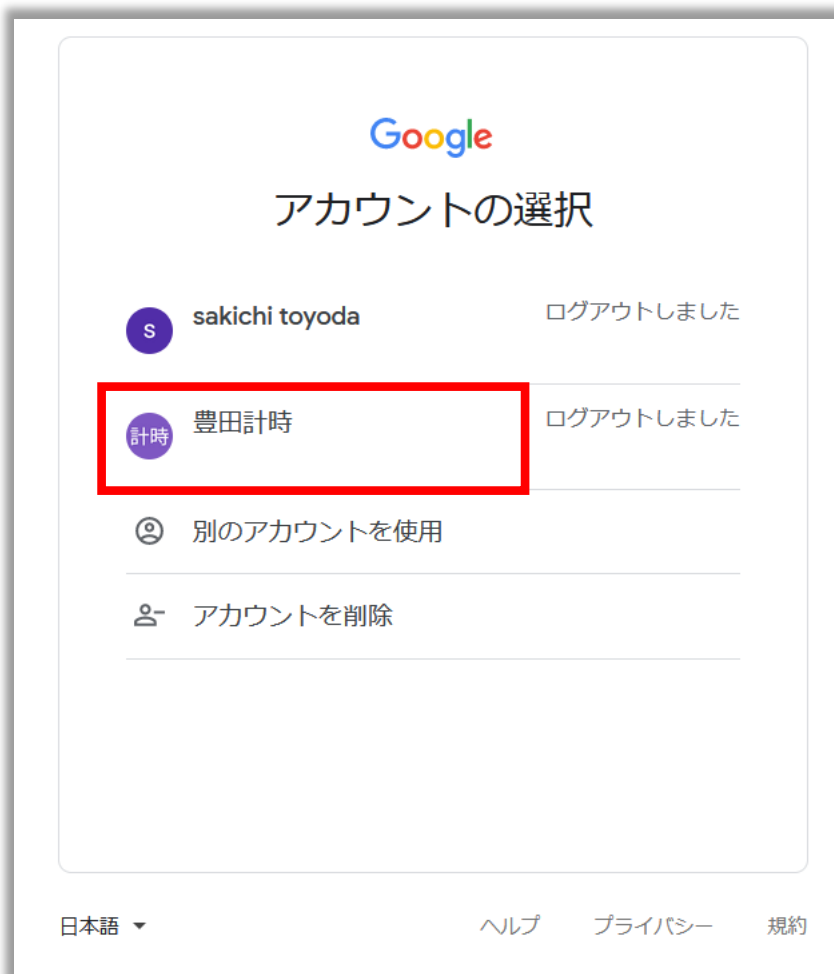
情報・ソフトウェア系 豊田計時

■ 人工知能／機械学習を学習する際の開発環境





今回はこれ


■ アカウントを選択




Google
アカウントの選択

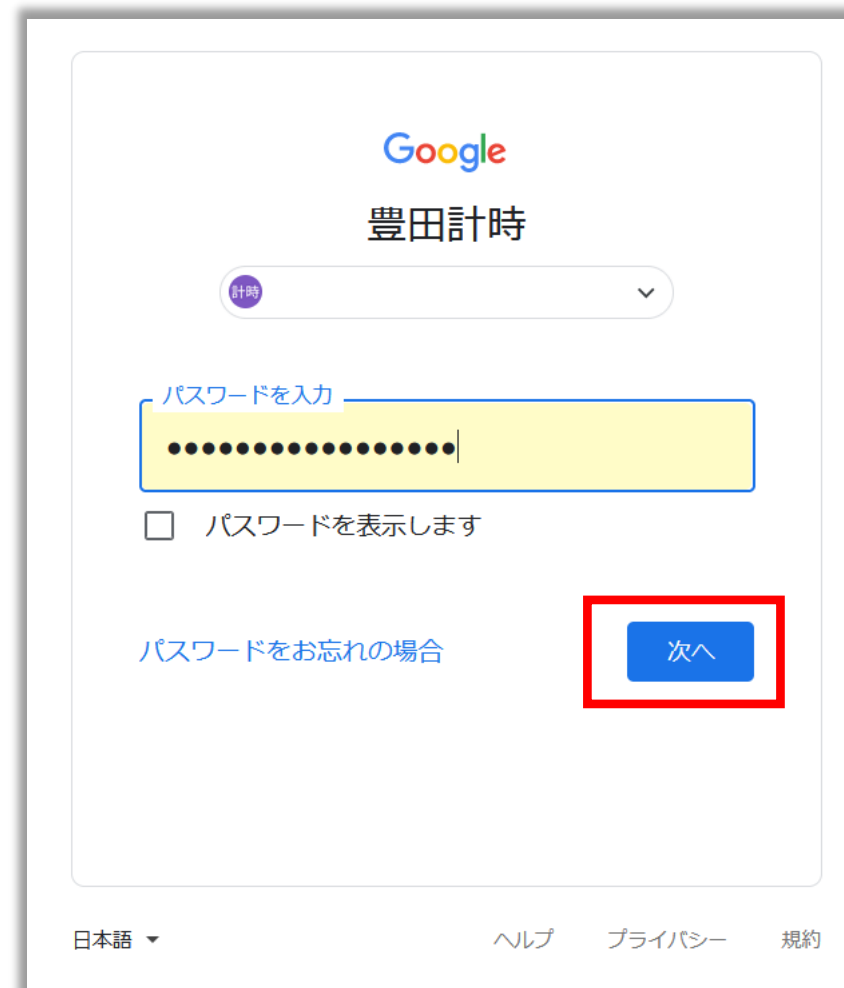
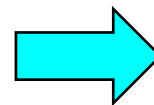
 sakichi toyoda ログアウトしました

 豊田計時 ログアウトしました

 別のアカウントを使用

 アカウントを削除

日本語 ▼ ヘルプ プライバシー 規約



Google
豊田計時



パスワードを入力

.....

☐ パスワードを表示します

パスワードをお忘れの場合

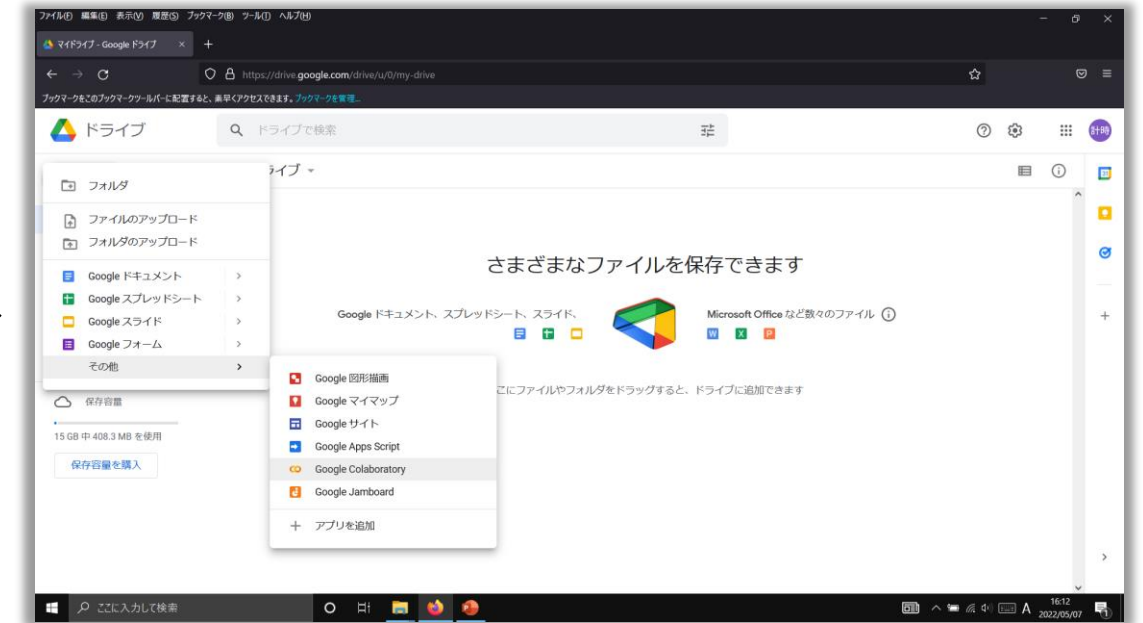
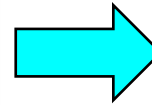
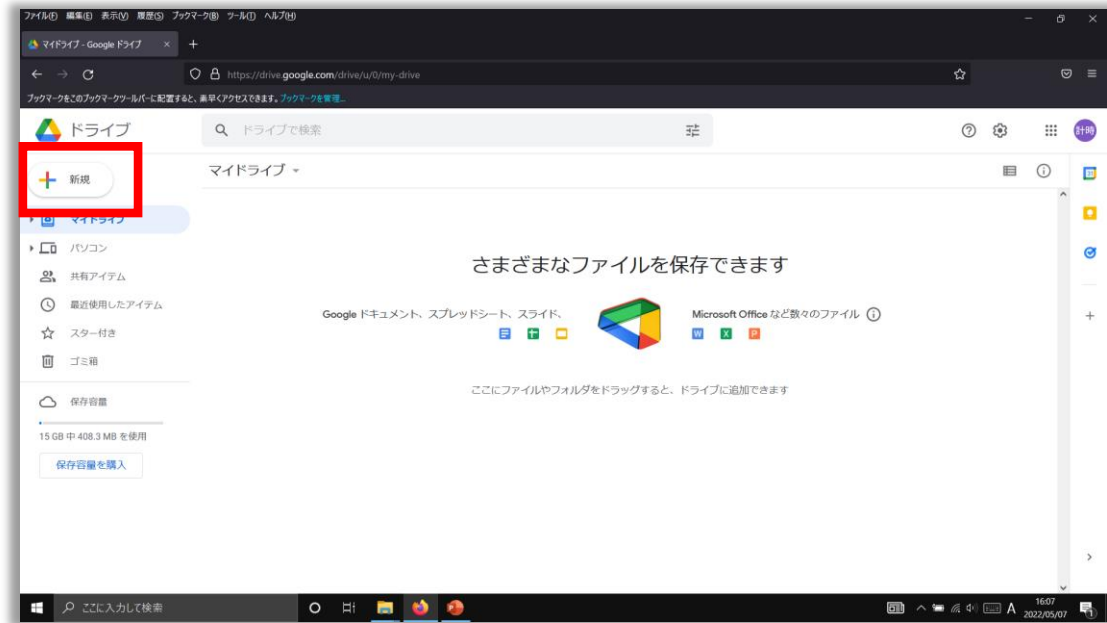
次へ

日本語 ▼ ヘルプ プライバシー 規約

実行したいアカウントをクリック

パスワードを入力

■ Google Colaboratoryを選択

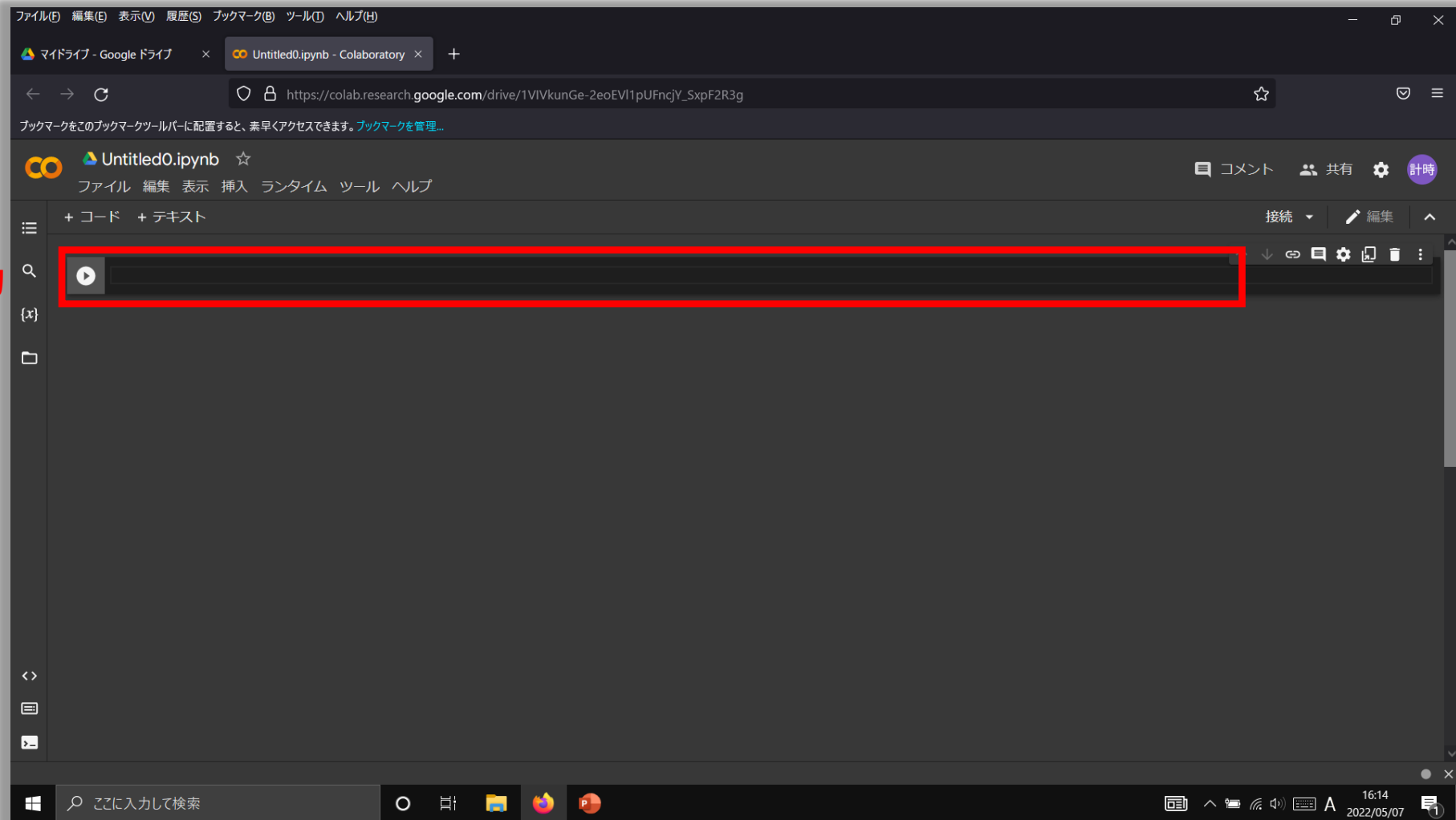


ドライブの新規をクリック

その他 + Google Colaboratoryをクリック

■ Google Colaboratoryを実行

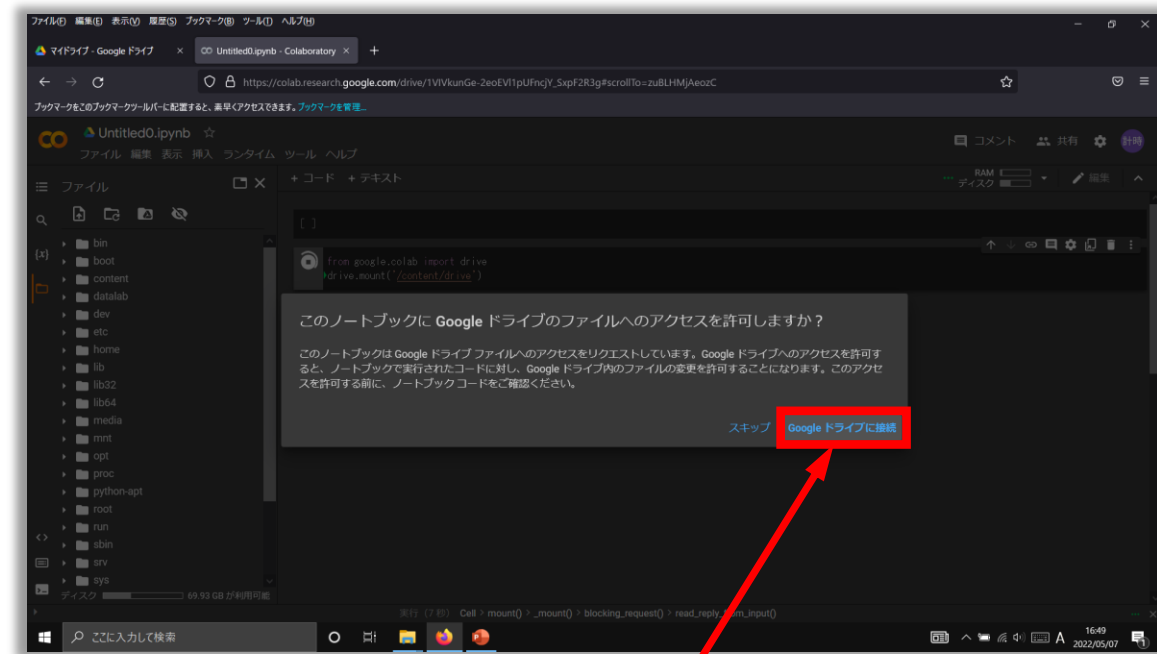
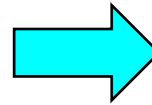
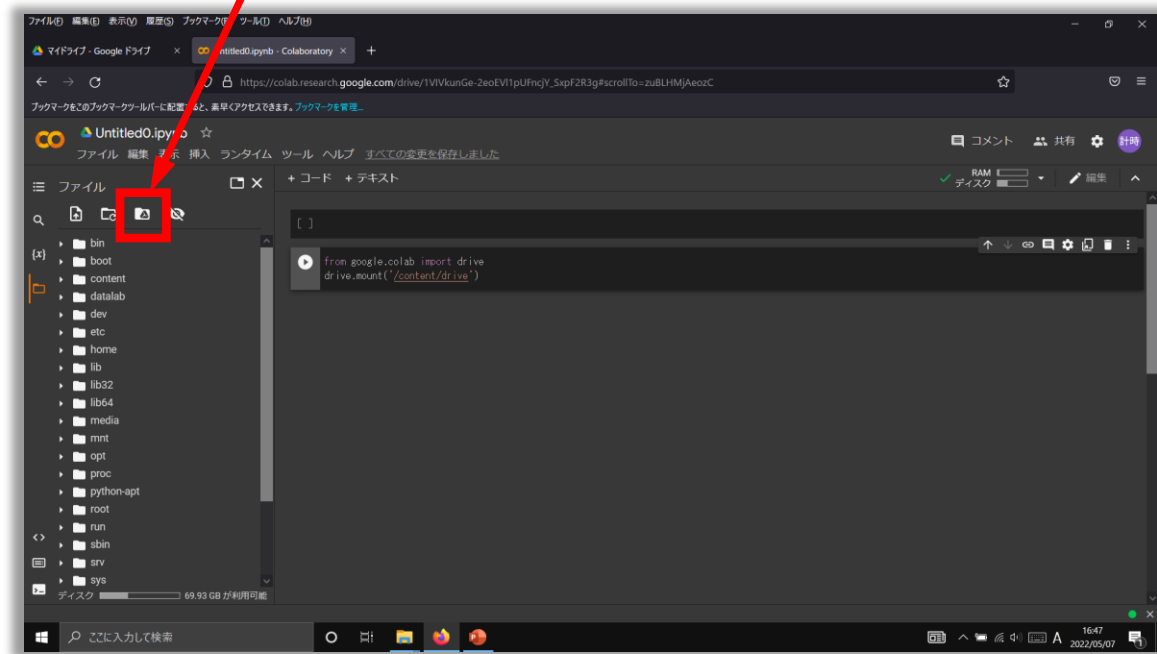
コードを入力



■ Google Colaboratoryを実行

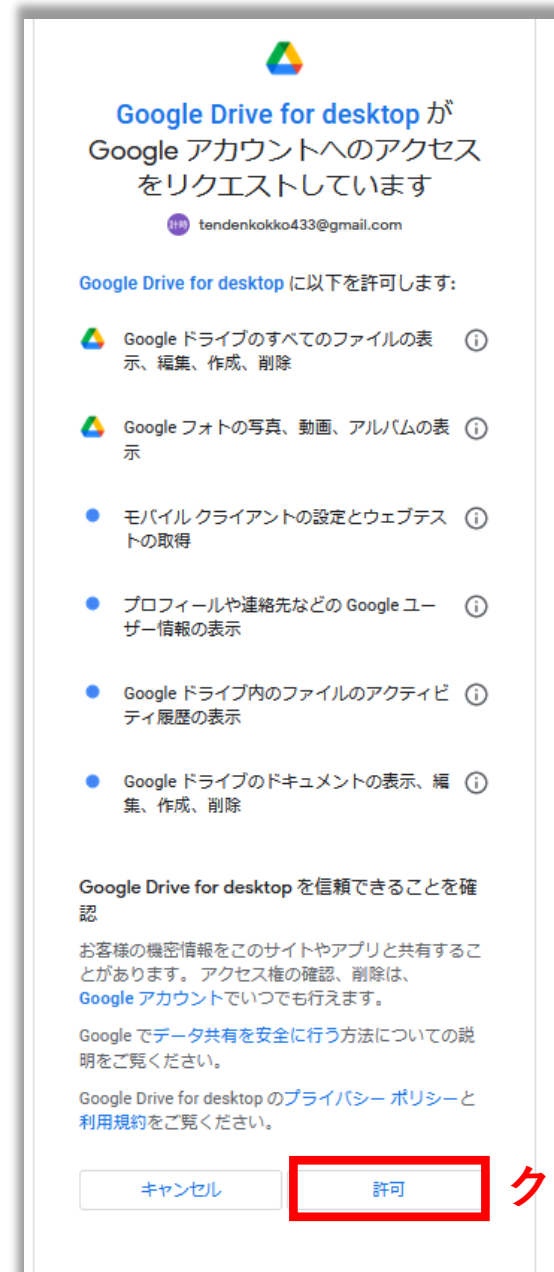
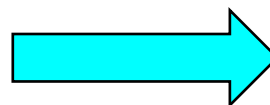


ドライブをマウ
ントをクリック



Googleドライブに接続をクリック

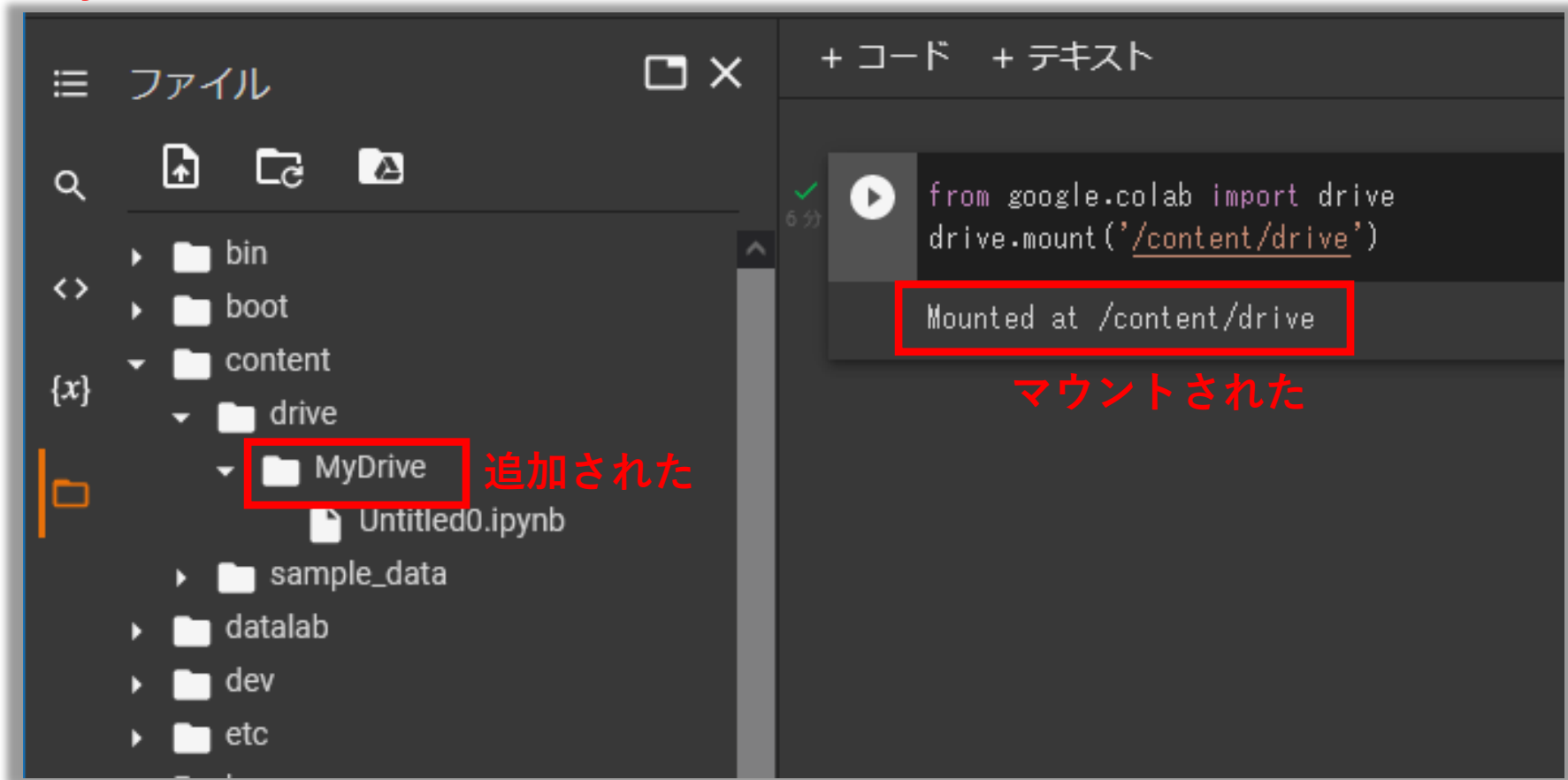
■ Google Colaboratoryを実行



クリック

■個人のGoogleドライブを使用する設定

MyDriveが追加される



■使用GPUを決める

コード

```
!nvidia-smi
```

応答

```
0 秒
!nvidia-smi
Sun May 8 08:35:51 2022
+-----+
| NVIDIA-SMI 460.32.03      Driver Version: 460.32.03      CUDA Version: 11.2     |
+-----+-----+
| GPU   Name                Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|=====+=====+
| 0   Tesla T4               Off        | 00000000:00:04.0 Off  |             0        |
| N/A   40C    P8             9W / 70W | 0MiB / 15109MiB |      0%      Default |
+-----+-----+
|
| Processes:
|  GPU   GI    CI          PID    Type   Process name                  GPU Memory
|   ID   ID     ID              |    |                  |   Usage
|=====+=====+
| No running processes found
+-----+-----+
```

- **Tesla P100** : 16GB
- **Tesla T4** : 15GB
- **Tesla K80** : 12GB

高速なGPUを狙い、
リセマラ（リセット
マラソン）
をやり過ぎると利用
制限がかかります

■YOLOv5リポジトリのクローン作成

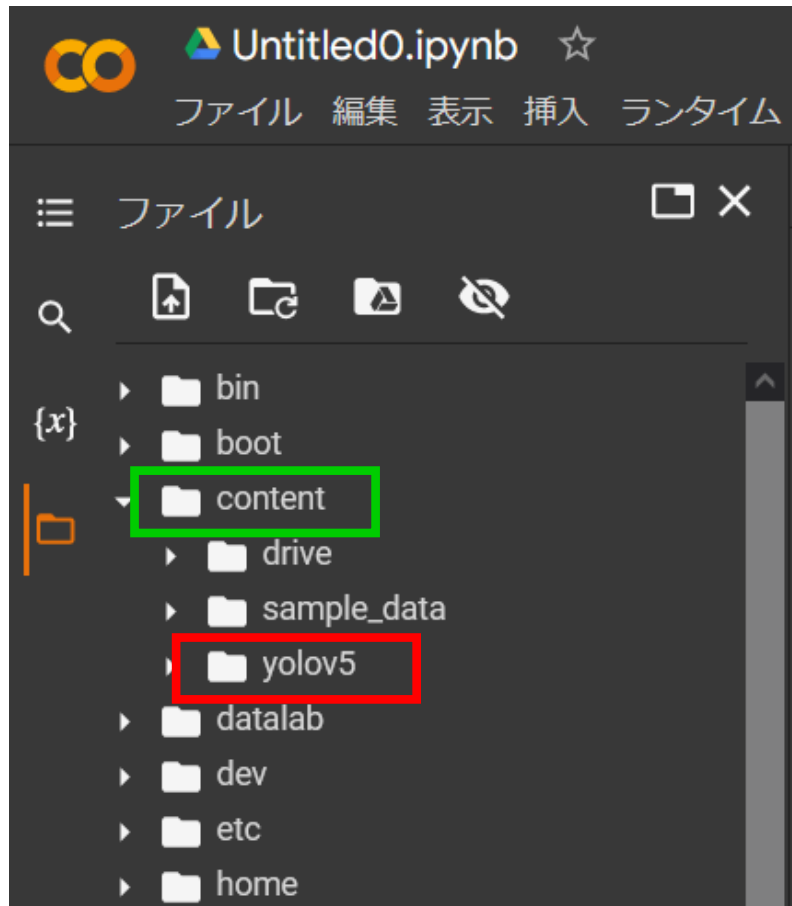
コード

```
!git clone https://github.com/ultralytics/yolov5
```

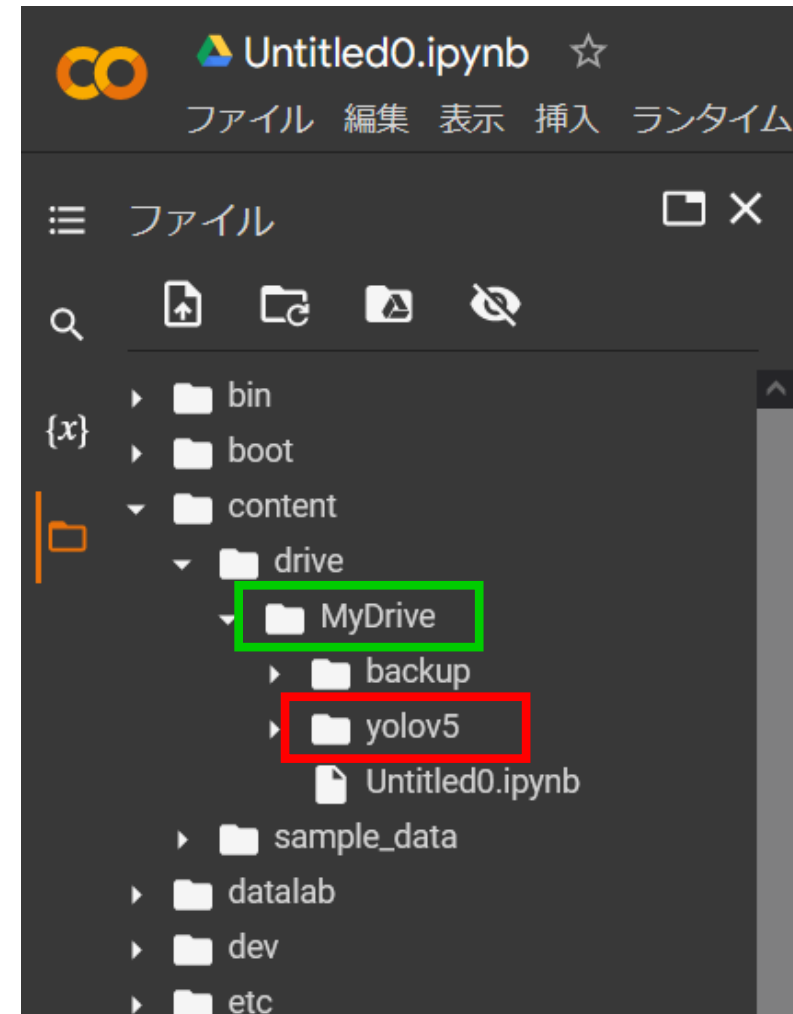
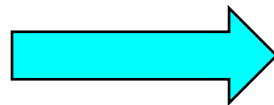
応答

```
Cloning into 'yolov5'...
remote: Enumerating objects: 7276, done.
remote: Counting objects: 100% (398/398), done.
remote: Compressing objects: 100% (252/252), done.
remote: Total 7276 (delta 251), reused 263 (delta 146), pack-reused 6878
Receiving objects: 100% (7276/7276), 9.20 MiB | 6.50 MiB/s, done.
Resolving deltas: 100% (4979/4979), done.
```

■ YOLOv5のインストール先



デフォルトだとcontent配下になる
(いずれ消えてしまう)



MyDriveの配下に移動する
(これで消えなくなる)

■必要なパッケージをインストール

コード

```
!pip install -r /content/drive/MyDrive/yolov5/requirements.txt
```

応答

```
Requirement already satisfied: matplotlib>=3.2.2 in  
/usr/local/lib/python3.7/dist-packages (from -r yolov5/requirements.txt  
(line 4)) (3.2.2)
```

(途中省略)

```
Installing collected packages: PyYAML, thop
```

```
Found existing installation: PyYAML 3.13
```

```
Uninstalling PyYAML-3.13:
```

```
Successfully uninstalled PyYAML-3.13
```

```
Successfully installed PyYAML-5.4.1 thop-0.0.31.post2005241907
```

■ 各datasetの意味

データ	説明
train dataset	分類器の パラメータを更新するための学習用データ 。（ニューラルネットワークだと重みを更新）
validation dataset	手動で設定するパラメータの良し悪しを確かめるための検証用データ 。学習は行わない。（ニューラルネットワークだと各層のニューロン数、隠れ層の数、バッチサイズ、学習係数など。ニューラルネットワークの重みは自動更新されるのでハイパーパラメータには含まれない）
test dataset	パラメータの学習後に 汎化性能を確かめるためのテスト用のデータ 。 学習には用いず 、理想的には 最後の一度だけ 使用する。


<https://algorithm.joho.info/programming/python/keras-train-validation-test-dataset/>

■YOLOv5を使用しオブジェクト検出する

コード **detect.py**に至るpath, taxi_1に至るpathを指定する

```
!python /content/drive/MyDrive/yolov5/detect.py --source  
/content/drive/MyDrive/taxi_1.jpg --weights yolov5s.pt --conf 0.25
```

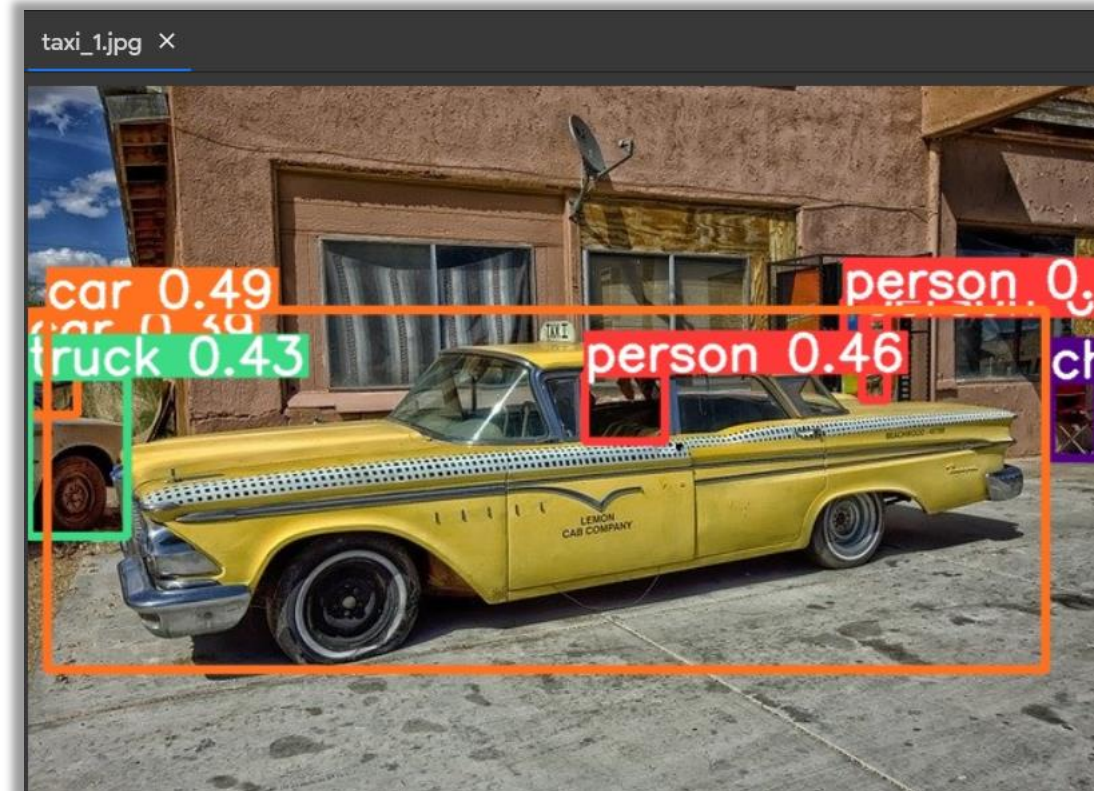
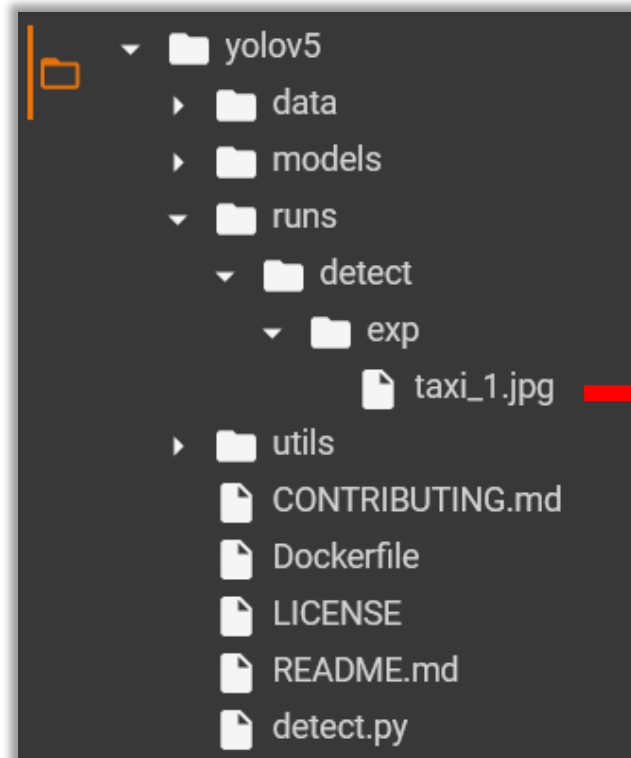
応答

```
requirements: /content/requirements.txt not found, check failed.  
YOLOv5  v5.0-223-gb83e1a4 torch 1.9.0+cu102 CUDA:0 (Tesla K80,  
11441.1875MB)  
Fusing layers...  
    (途中省略)  
Results saved to runs/detect/exp 検出結果はここに保存される  
Done. (1.303s)
```

<https://laboratory.kazuuu.net/i-used-yolo-v5-with-google-colaboratory/>

■ オブジェクト検出した結果を確認する

yolov5/runs/detect/expに結果が格納されるtaxi_1.jpg



taxi_1.jpg

■ apex-0.1をインストールする

trainを高速化するための処理

コード

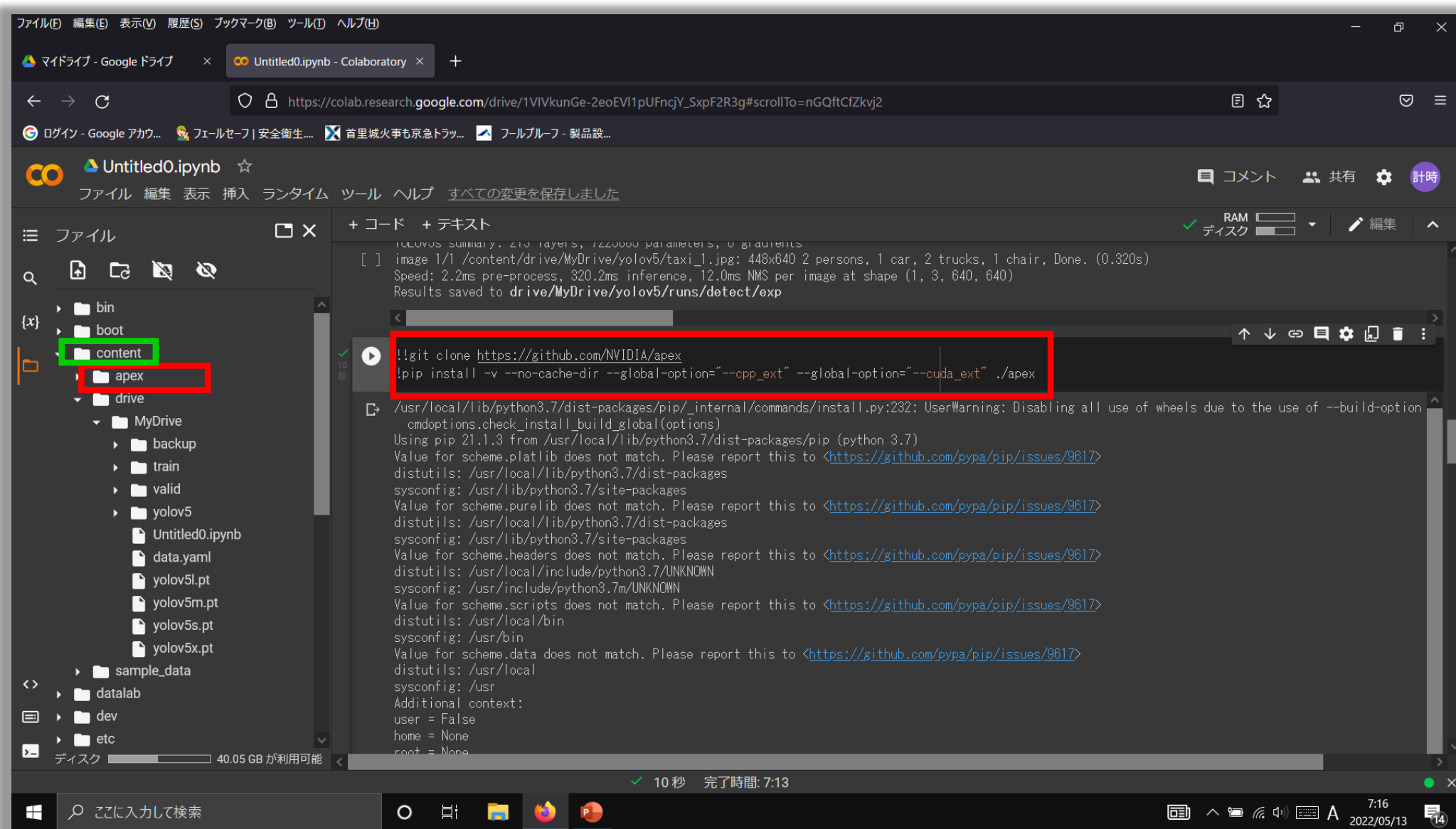
```
!git clone https://github.com/NVIDIA/apex  
!pip install -v --no-cache-dir --global-option="--cpp_ext" --global-  
option="--cuda_ext" ./apex
```

応答

```
Cloning into 'apex'...  
remote: Enumerating objects: 9064, done.  
remote: Counting objects: 100% (135/135), done.  
remote: Compressing objects: 100% (112/112), done.  
  (省略) とても長い (約20分)  
Successfully installed apex-0.1  
Removed build tracker: '/tmp/pip-req-tracker-_wcd3ype'
```


■ apex-0.1をインストールする

MyDriveよりも上位にインストールされる（ログオフすると消える）



ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)

マイドライブ - Google ドライブ x Untitled0.ipynb - Colaboratory x +

← → ↺ https://colab.research.google.com/drive/1VIVkunGe-2eoEV1pUFncjY_SxpF2R3g#scrollTo=nGQftCfZkv2

ログイン - Google アカウ... フェールセーフ | 安全衛生... 首里城火事も緊急トラッ... フールブルーフ - 製品設...

CO Untitled0.ipynb ☆

ファイル 編集 表示 挿入 ランタイム ツール ヘルプ すべての変更を保存しました

RAM デスク

10 秒

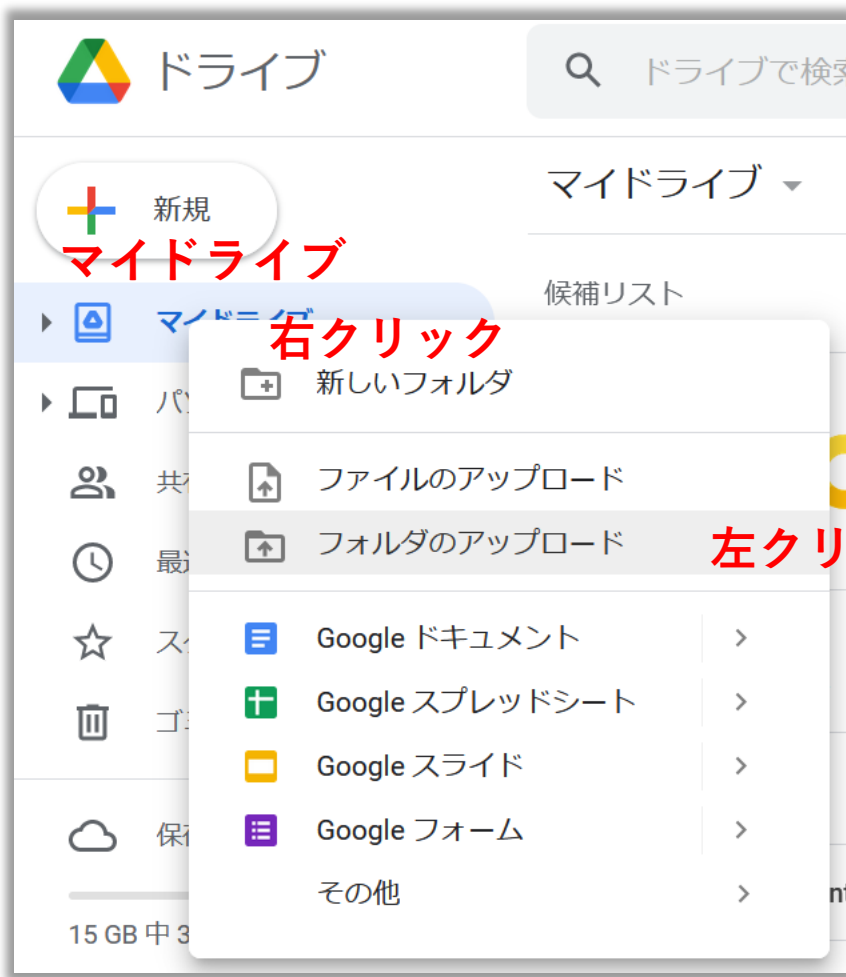
```
!!git clone https://github.com/NVIDIA/apex
!pip install -v --no-cache-dir --global-option="--cpp_ext" --global-option="--cuda_ext" ./apex
```

10 秒 完了時間: 7:13

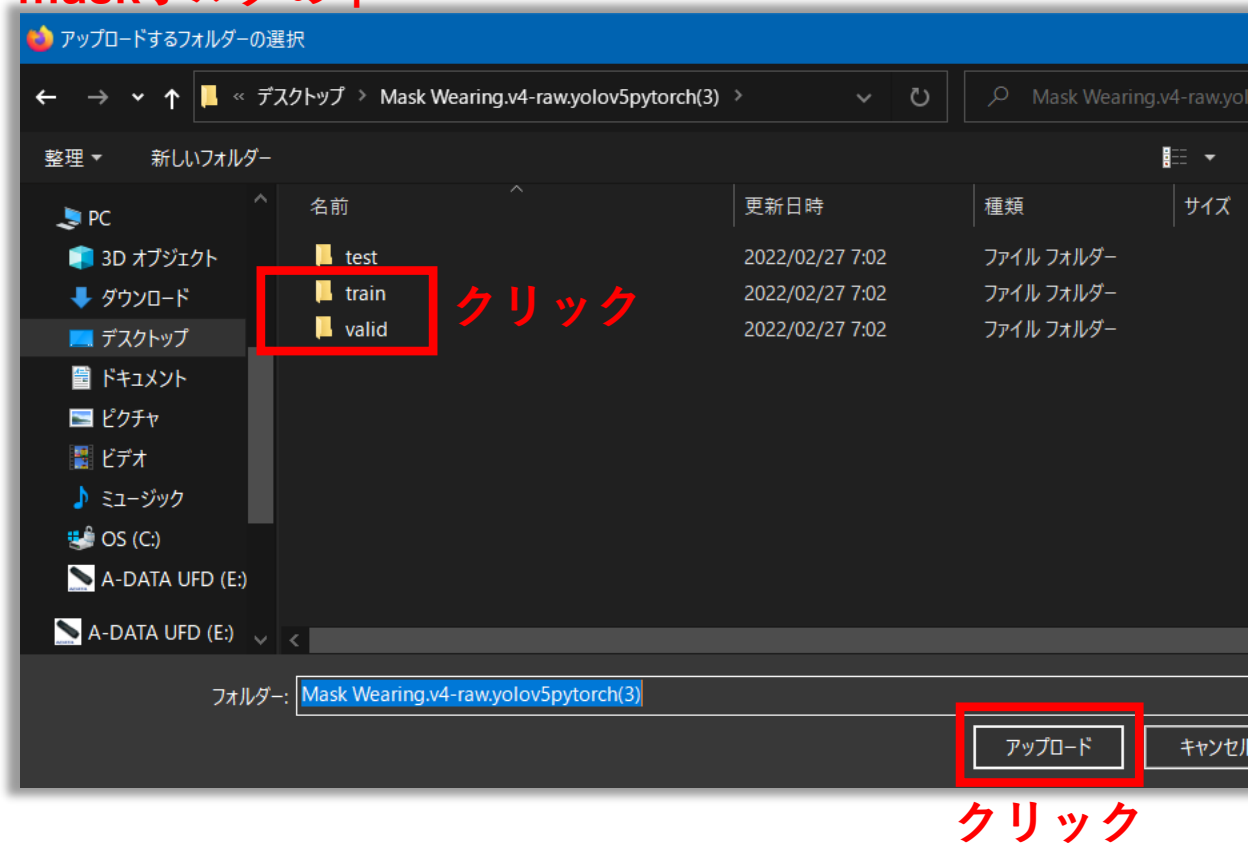
ここに入力して検索

7:16 2022/05/13

■ フォルダをアップロードする



maskホルダの中



■ ファイルをアップロードする

ドライブ

owner:me (type:application/vnd.google.colaboratory || type:application/vnd.google.colab)

検索結果

場所 ファイル形式 ユーザー 最終更新 タイトルのみ ToDo すべてクリア

関連度 ↓

Untitled0.ipynb

4 個のアップロード完了

yolov5x.pt	✓
yolov5s.pt	✓
yolov5m.pt	✓
yolov5l.pt	✓

15 GB 中 788 MB を使用

保存容量を購入

ここにを入力して検索

12°C 晴れのちくもり 8:13 2022/05/08

アップロード中

■ 4 種類の重みをダウンロード

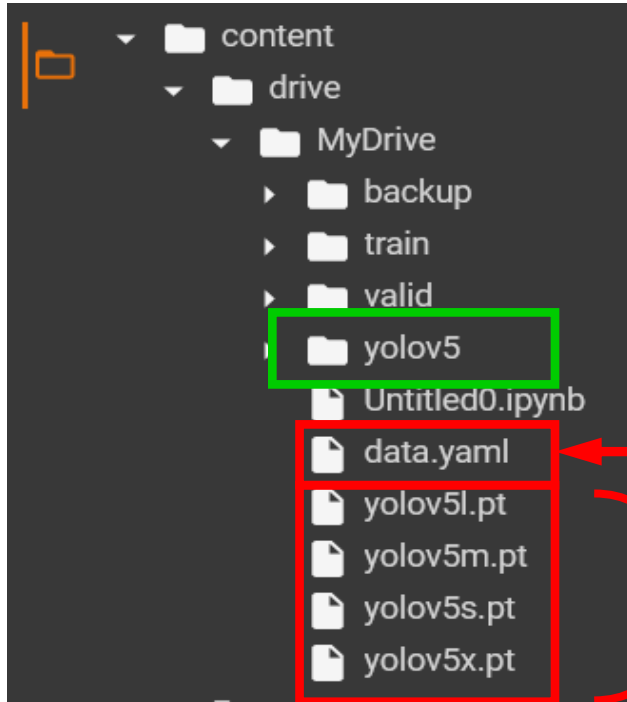
コード

```
!bash weights/download_weights.sh
```

yolov5l.pt, yolov5m.pt, yolov5s.pt, yolov5x.ptの 4 種類の重みがダウンロードされます。

yolov5l.yaml, yolov5m. yaml, yolov5s. yaml, yolov5x. yamlの 4 つはyolov5のサブディレクトリであるmodelsの中にあります。

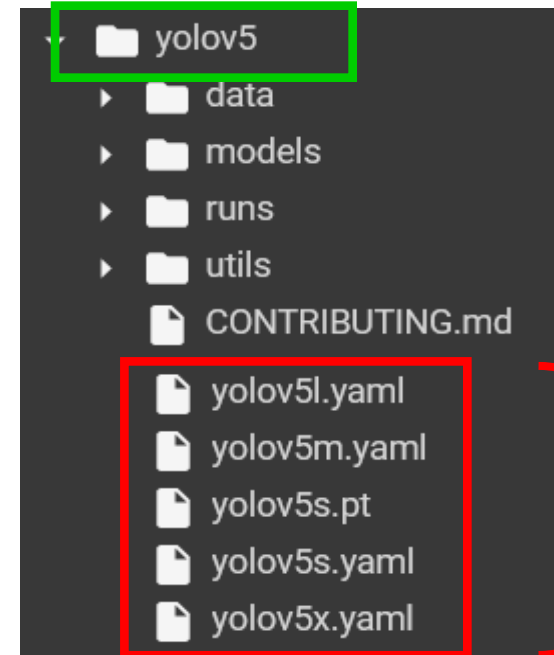
■ ファイルをアップロードする



Mydriveの配下に入る
(ログオフで消えることはない)

data.yaml
確認

.pt
確認



.yaml
確認

yolov5の配下に入る
(ログオフで消えることはない)

■ 学習する

コード **train.py**に至るpathを指定する

```
# Train YOLOv5s on coco128 for 3 epochs
#!python train.py --img 416 --batch 16 --epochs 100 --data data.yaml --cfg
yolov5s.yaml --weights yolov5s.pt --nosave --cache
!python /content/drive/MyDrive/yolov5/train.py --img 416 --batch 16 --
epochs 100 --data /content/drive/MyDrive/data.yaml --cfg
/content/drive/MyDrive/yolov5/yolov5s.yaml
```

応答

```
train: weights=yolov5/yolov5s.pt, cfg=yolov5x.yaml, data=data.yaml,
hyp=yolov5/data/hyps/hyp.scratch-low.yaml, epochs=500, batch_size=16,
(途中省略)
github: up to date with https://github.com/ultralytics/yolov5 ✓
YOLOv5 🚀 v6.1-11-g63ddb6f torch 1.10.0+cu111 CUDA:0 (Tesla K80,
11441MiB)
```

<https://laboratory.kazuuu.net/i-used-yolo-v5-with-google-colaboratory/>

■学習開始する

epoch500の場合

エポック開始
(0/499)

```
+ コード + テキスト
21         [-1, 10] 1 0 models.common.Conv [040, 040, 3, 2]
22         [-1, 10] 1 0 models.common.Concat [1]
23         [-1, 4] 4 19676160 models.common.C3 [1280, 1280, 4, False]
... 24 [17, 20, 23] 1 47103 models.yolo.Detect [2, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [116, 90, 156, 198, 373, 32]]]
Model Summary: 567 layers, 86224543 parameters, 86224543 gradients, 204.2 GFLOPs

Transferred 57/745 items from yolov5/yolov5s.pt
Scaled weight_decay = 0.0005
optimizer: SGD with parameter groups 123 weight (no decay), 126 weight, 126 bias
albumentations: version 1.0.3 required by YOLOv5, but version 0.1.12 is currently installed
train: Scanning '/content/yolov5/./train/labels' images and labels...105 found, 0 missing, 0 empty, 0 corrupt: 100% 105/105 [00:00<00:00, 1110.96it/s]
train: New cache created: /content/yolov5/./train/labels.cache
val: Scanning '/content/yolov5/./valid/labels' images and labels...29 found, 0 missing, 0 empty, 0 corrupt: 100% 29/29 [00:00<00:00, 425.81it/s]
val: New cache created: /content/yolov5/./valid/labels.cache
Plotting labels to yolov5/runs/train/exp3/labels.jpg...

AutoAnchor: 5.70 anchors/target, 0.999 Best Possible Recall (BPR). Current anchors are a good fit to dataset ✓
Image sizes 416 train, 416 val
Using 2 dataloader workers
Logging results to yolov5/runs/train/exp3
Starting training for 500 epochs...

Epoch  gpu_mem  box  obj  cls  labels  img_size
0/499   6.61G   0.1114  0.06958  0.02803   166     416: 71% 5/7 [00:26<00:09, 4.78s/it]
```

■強制的に切断された例①

epoch500の場合

CPUでの作業を余儀なくされる

エポック中断
(307/499)

+ コード + テキスト

301/499

7.08G

0.03588

0.04208

0.002658

124

416: 100% 7/7 [00:29<00:00, 4.15s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.74s/it]

all

29

162

0.827

0.63

0.697

0.367

Epoch

gpu_mem

box

obj

cls

labels

img_size

302/499

7.08G

0.03479

0.03848

0.00256

179

416: 100% 7/7 [00:28<00:00, 4.14s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.76s/it]

all

29

162

0.743

0.657

0.707

0.357

Epoch

gpu_mem

box

obj

cls

labels

img_size

303/499

7.08G

0.03486

0.04111

0.002911

110

416: 100% 7/7 [00:29<00:00, 4.15s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.75s/it]

all

29

162

0.733

0.736

0.672

0.349

Epoch

gpu_mem

box

obj

cls

labels

img_size

304/499

7.08G

0.03079

0.03848

0.002706

98

416: 100% 7/7 [00:28<00:00, 4.13s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.75s/it]

all

29

162

0.739

0.745

0.687

0.362

Epoch

gpu_mem

box

obj

cls

labels

img_size

305/499

7.08G

0.03551

0.04202

0.003374

167

416: 100% 7/7 [00:28<00:00, 4.14s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.75s/it]

all

29

162

0.66

0.775

0.686

0.331

Epoch

gpu_mem

box

obj

cls

labels

img_size

306/499

7.08G

0.03398

0.04114

0.002499

80

416: 100% 7/7 [00:29<00:00, 4.15s/it]

Class

Images

Labels

P

R

mAP@.5

mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.74s/it]

all

29

162

0.684

0.67

0.644

0.352

Epoch

gpu_mem

box

obj

cls

labels

img_size

307/499

7.08G

0.03315

0.0366

0.002414

157

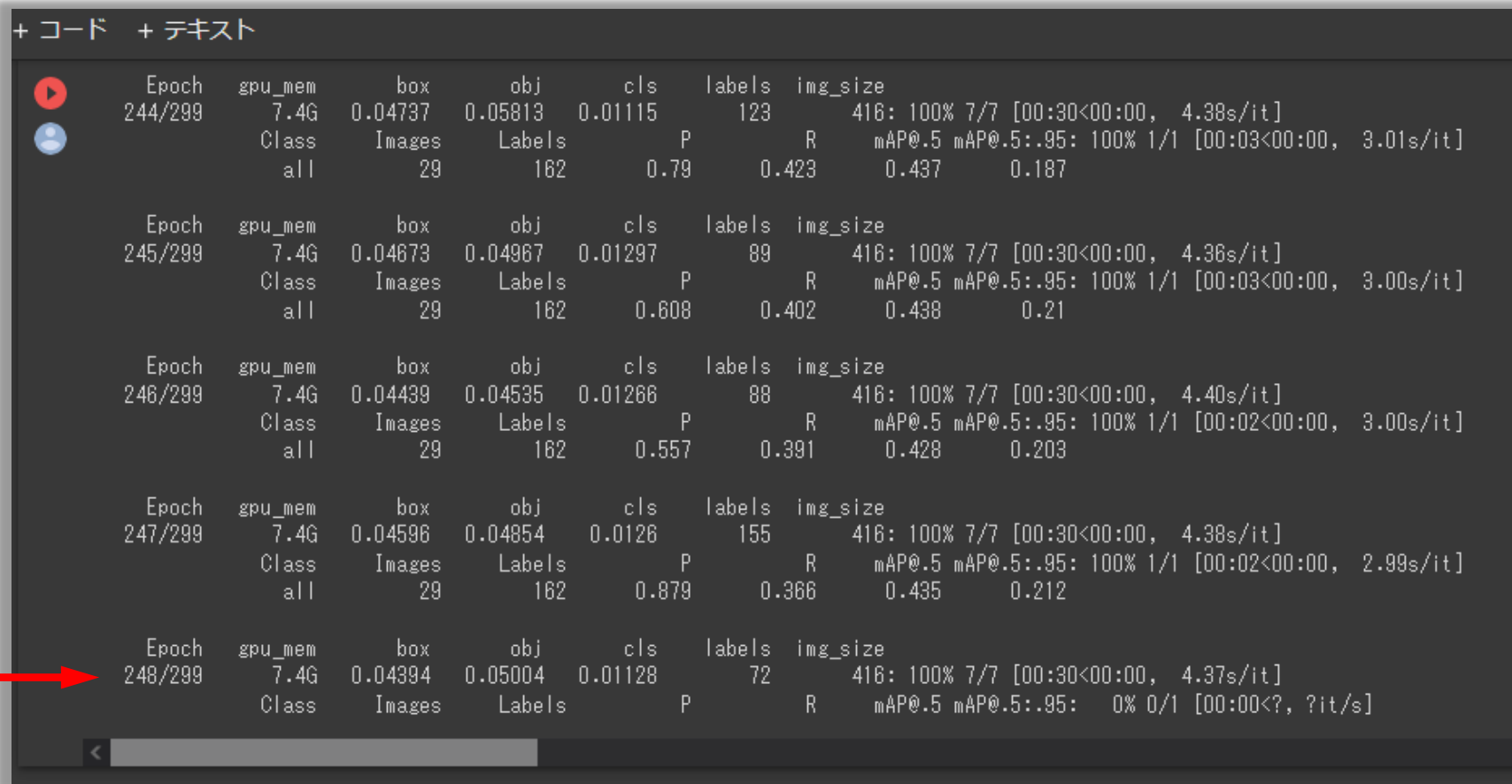
416: 71% 5/7 [00:22<00:08, 4.38s/it]

■強制的に切断された例②

epoch300の場合

CPUでの作業を余儀なくされる

エポック中断
(248/299)



Epoch	gpu_mem	box	obj	cls	labels	img_size
244/299	7.4G	0.04737	0.05813	0.01115	123	416: 100% 7/7 [00:30<00:00, 4.38s/it]
Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100% 1/1 [00:03<00:00, 3.01s/it]
all	29	162	0.79	0.423	0.437	0.187
Epoch	gpu_mem	box	obj	cls	labels	img_size
245/299	7.4G	0.04673	0.04967	0.01297	89	416: 100% 7/7 [00:30<00:00, 4.36s/it]
Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100% 1/1 [00:03<00:00, 3.00s/it]
all	29	162	0.608	0.402	0.438	0.21
Epoch	gpu_mem	box	obj	cls	labels	img_size
246/299	7.4G	0.04439	0.04535	0.01266	88	416: 100% 7/7 [00:30<00:00, 4.40s/it]
Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100% 1/1 [00:02<00:00, 3.00s/it]
all	29	162	0.557	0.391	0.428	0.203
Epoch	gpu_mem	box	obj	cls	labels	img_size
247/299	7.4G	0.04596	0.04854	0.0126	155	416: 100% 7/7 [00:30<00:00, 4.38s/it]
Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 100% 1/1 [00:02<00:00, 2.99s/it]
all	29	162	0.879	0.366	0.435	0.212
Epoch	gpu_mem	box	obj	cls	labels	img_size
248/299	7.4G	0.04394	0.05004	0.01128	72	416: 100% 7/7 [00:30<00:00, 4.37s/it]
Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95: 0% 0/1 [00:00<?, ?it/s]

■ 学習成功の場合

epoch100の場合

```
+ コード + テキスト
97/99 7.4G 0.07203 0.06784 0.01555 140 416: 100% 7/7 [00:30<00:00, 4.40s/it]
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 1/1 [00:03<00:00, 3.41s/it]
all 29 162 0.755 0.222 0.24 0.0867

Epoch gpu_mem box obj cls labels img_size
98/99 7.4G 0.07109 0.07182 0.01932 61 416: 100% 7/7 [00:30<00:00, 4.40s/it]
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 1/1 [00:03<00:00, 3.43s/it]
all 29 162 0.782 0.215 0.242 0.0886

Epoch gpu_mem box obj cls labels img_size
99/99 7.4G 0.07238 0.06984 0.016 68 416: 100% 7/7 [00:30<00:00, 4.39s/it]
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 1/1 [00:03<00:00, 3.49s/it]
all 29 162 0.795 0.194 0.233 0.0789

100 epochs completed in 1.068 hours.
Optimizer stripped from yolov5/runs/train/exp/weights/last.pt, 175.1MB
Optimizer stripped from yolov5/runs/train/exp/weights/best.pt, 175.1MB

Validating yolov5/runs/train/exp/weights/best.pt...
Fusing layers...
Model Summary: 476 layers, 87205423 parameters, 0 gradients, 217.1 GFLOPs
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 1/1 [00:04<00:00, 4.43s/it]
all 29 162 0.782 0.215 0.242 0.0886
mask 29 142 0.564 0.43 0.459 0.169
no-mask 29 20 1 0 0.0259 0.00771

Results saved to yolov5/runs/train/exp
```

結果格納先

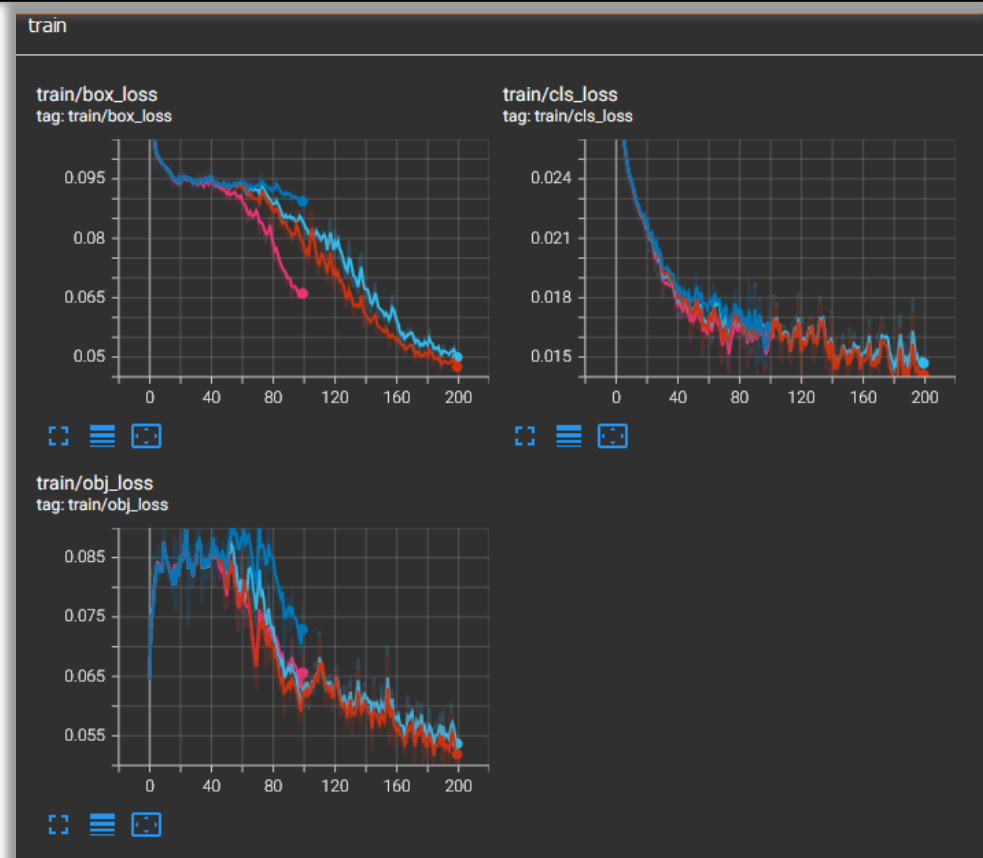
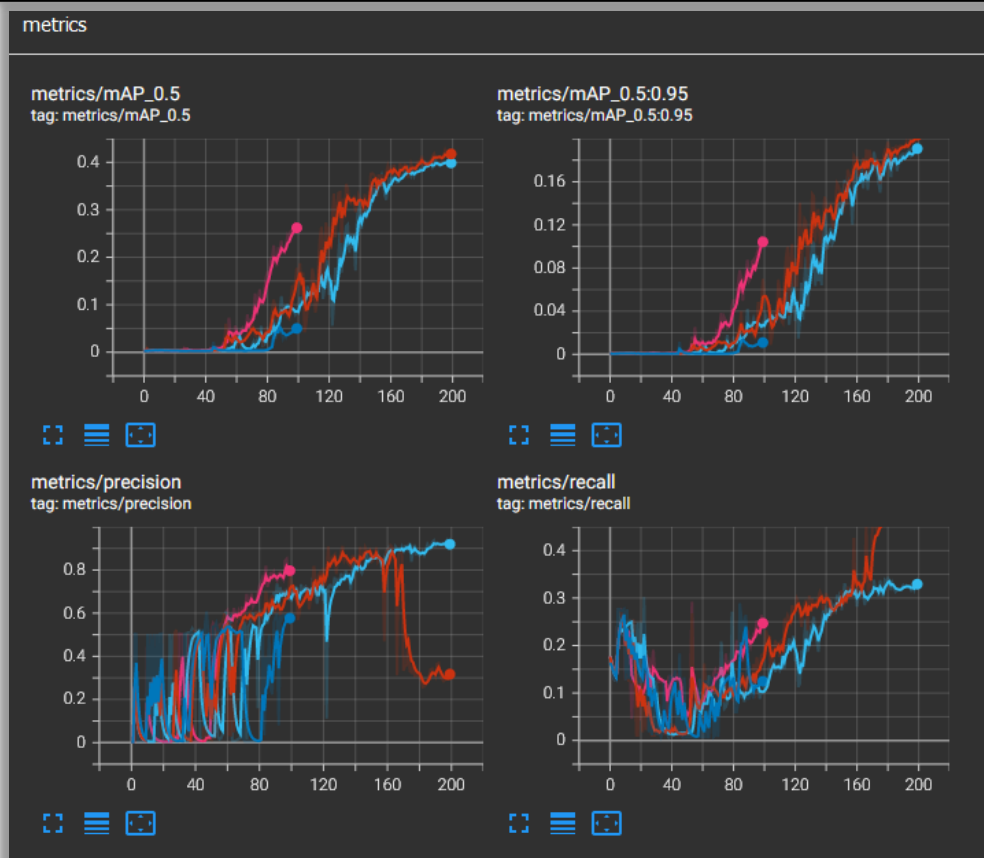


Results saved to yolov5/runs/train/exp

■ グラフにする

コード

```
# Start tensorboard (optional)
%load_ext tensorboard
%tensorboard --logdir "/content/drive/My Drive/yolov5/runs"
```

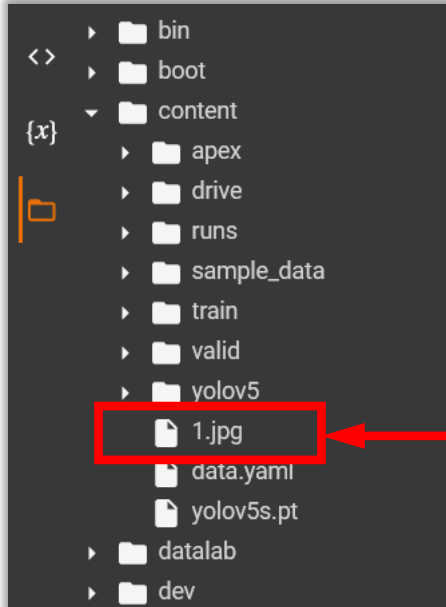


<https://laboratory.kazuuu.net/i-used-yolo-v5-with-google-colaboratory/>

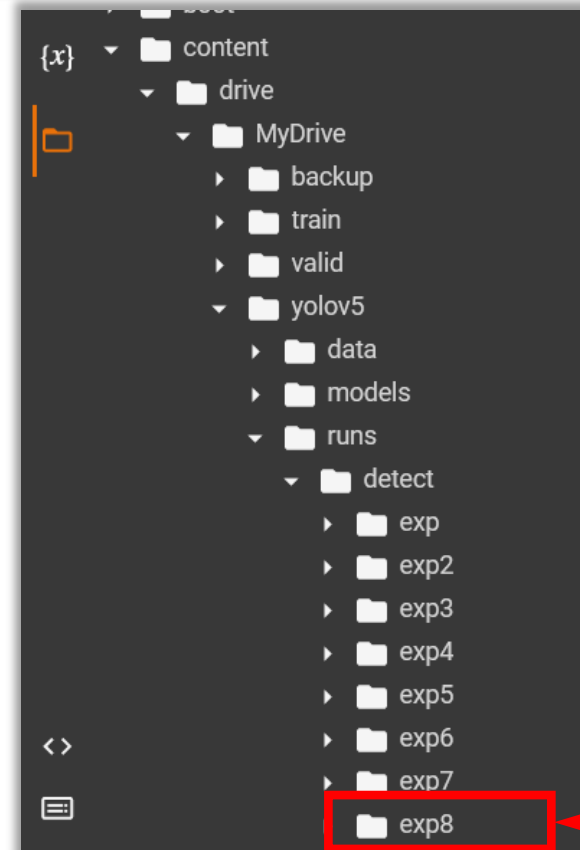
■推論（検出）する

コード

```
!python /content/drive/MyDrive/yolov5/detect.py --source 1.jpg --weights  
/content/drive/MyDrive/yolov5/runs/detect/exp8/weights/best.pt --img 416  
Image(filename='inference/output/1.jpg', width=419)
```



推論画像



学習済best.pt
ファイル

<https://laboratory.kazuuu.net/i-used-yolo-v5-with-google-colaboratory/>

■推論（検出）する

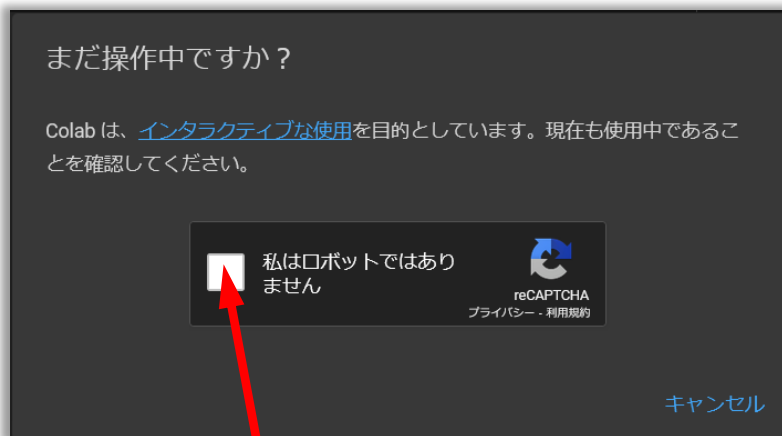


epocs=100

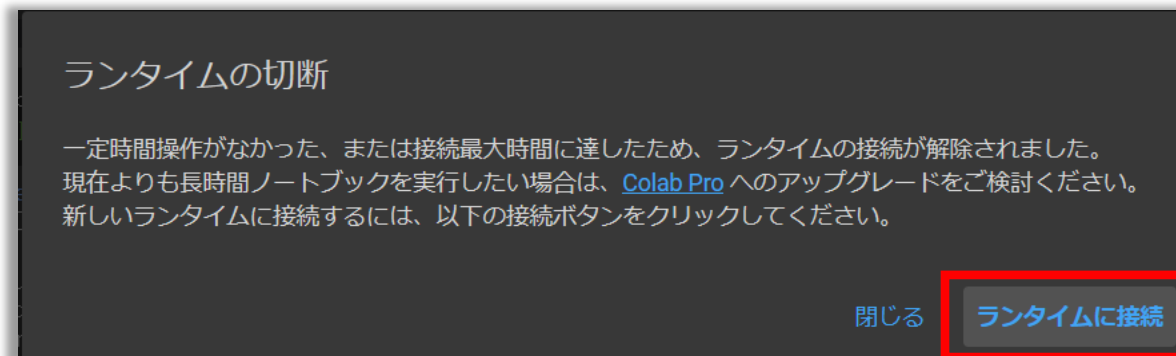


epocs=300

■ こんなメッセージが出たら

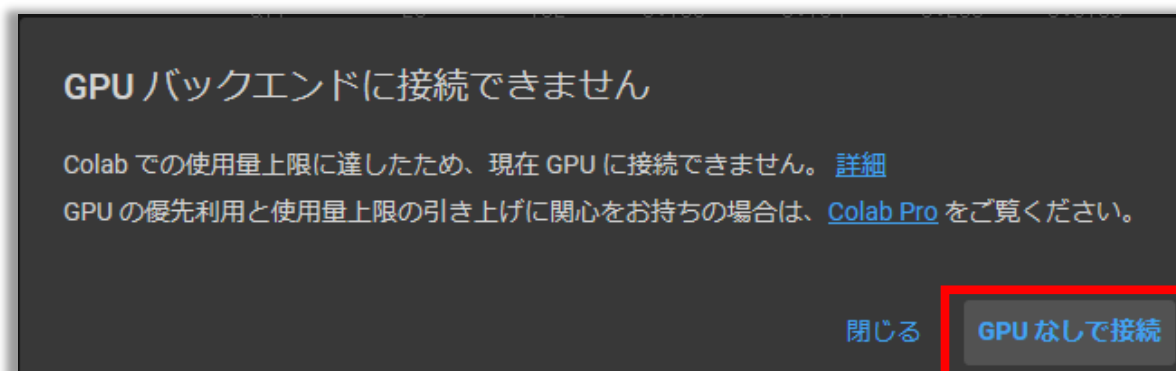


クリックすると指定した画像を選択照合の後、復旧する



30分でも出ることもある

クリックすると復旧する



日本時間10:00-18:00
が繋がりやすい

クリックすると
CPU接続になる

■ こんなメッセージが出たら

500epochを322epochで打ち切られた

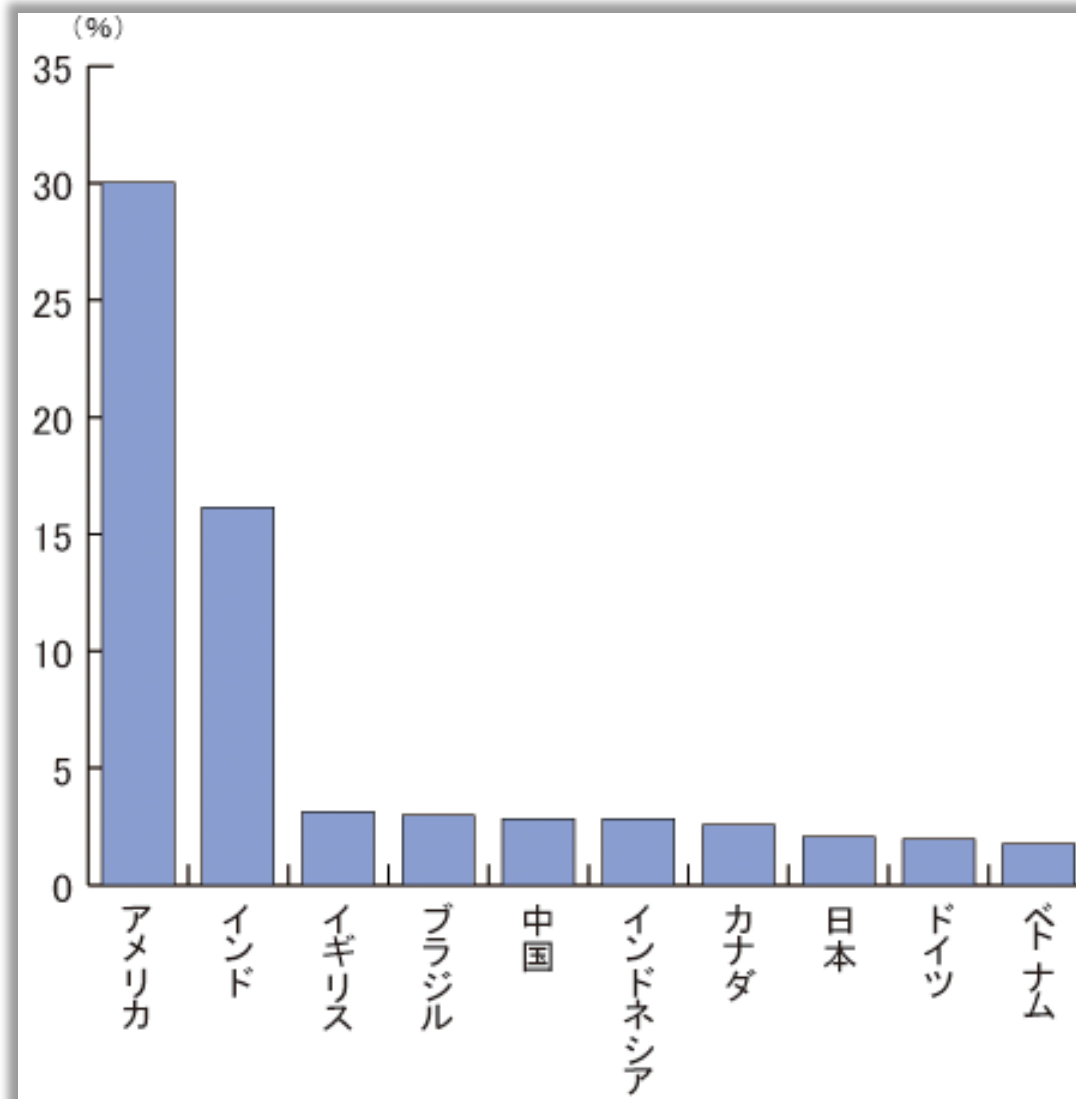
```
Epoch 322/499  gpu_mem 2.01G  box 0.01724  obj 0.01047  cls 0.001255  labels 66  img_size 416: 100% 55/55 [00:16<00:00, 3.33it/s]
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 8/8 [00:01<00:00, 4.03it/s]
all 250 452 0.631 0.509 0.535 0.385
Stopping training early as no improvement observed in last 100 epochs. Best results observed at epoch 222. best model saved as best.pt
To update EarlyStopping(patience=100) pass a new patience value, i.e. `python train.py --patience 300` or use `--patience 0` to disable EarlyStopping.

323 epochs completed in 1.709 hours.
Optimizer stripped from drive/MyDrive/yolov5/runs/train/exp2/weights/last.pt, 14.4MB
Optimizer stripped from drive/MyDrive/yolov5/runs/train/exp2/weights/best.pt, 14.4MB

Validating drive/MyDrive/yolov5/runs/train/exp2/weights/best.pt...
Fusing layers...
YOLOv5s summary: 224 layers, 7064698 parameters, 0 gradients
Class Images Labels P R mAP@.5 mAP@.5:.95: 100% 8/8 [00:03<00:00, 2.29it/s]
all 250 454 0.786 0.478 0.556 0.408
Ambulance 250 64 0.959 0.728 0.817 0.696
Bus 250 46 0.647 0.565 0.624 0.49
Car 250 238 0.654 0.353 0.417 0.278
Motorcycle 250 46 0.781 0.478 0.508 0.277
Truck 250 60 0.889 0.267 0.415 0.301
Results saved to drive/MyDrive/yolov5/runs/train/exp2
```

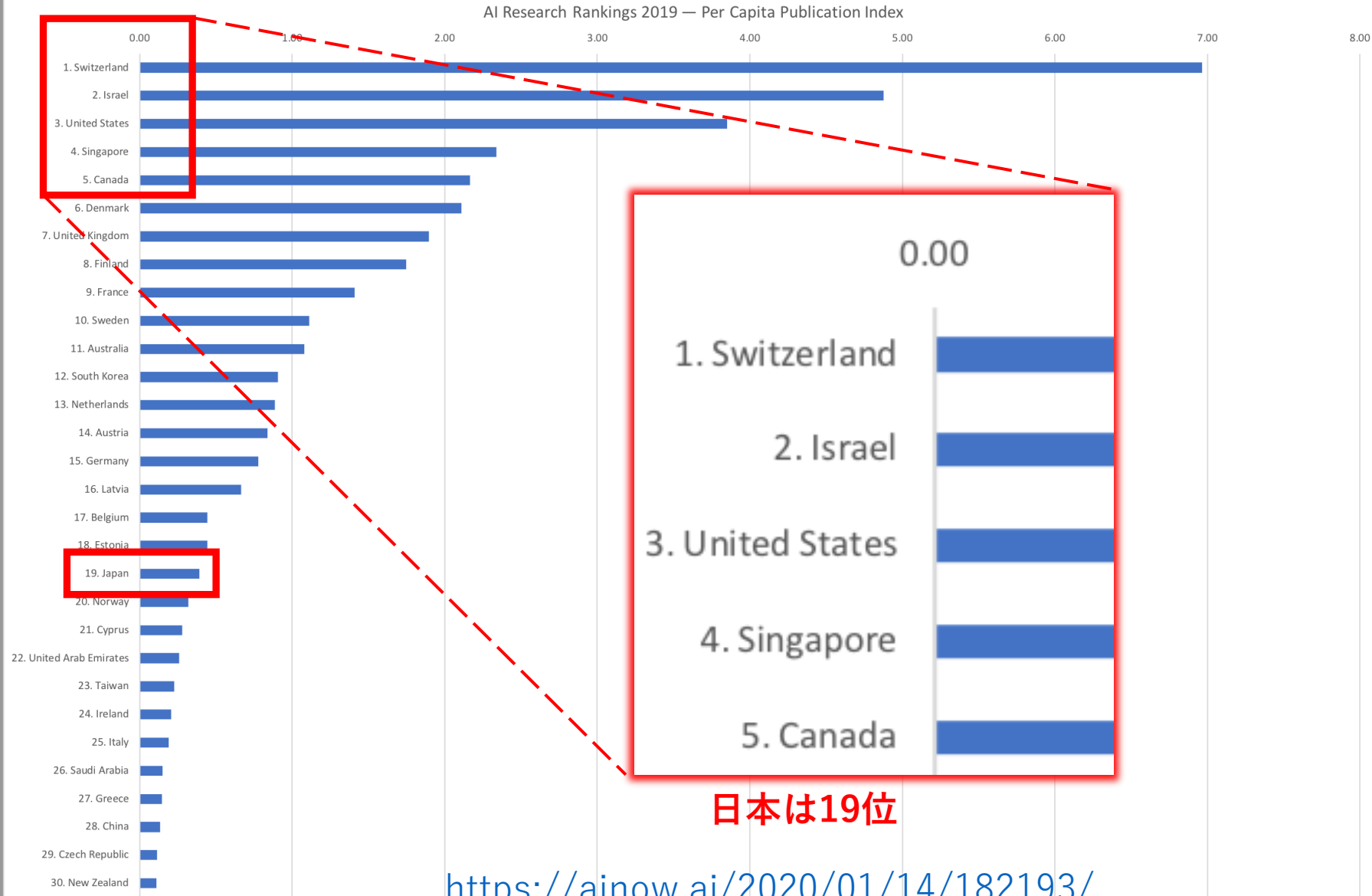
打ち切りを防ぐ方法

■ 「Google+」 の国別ユーザー数ランキング



ユーザーローカルによる
独自調査結果 (2011)

■ AI研究ランキング2019



■ 日本との時差

アメリカ（時差14時間）

現地時間	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
日本時間	14	15	16	17	18	19	20	21	22	23	0	1	2	3	4	5	6	7	8	9	10	11	12	13
UTC	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	0	1	2	3	4

インド（時差3時間30分）

現地時間	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
日本時間	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	0	1	2
UTC	18	19	20	21	22	23	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

■ エラーメッセージ

ローカルPCで環境構築する場合

エラー

AttributeError: module 'tensorflow' has no attribute 'io'

対策

`pip install tensorflow-io`

<https://stackoverflow.com/questions/60146023/attributeerror-module-tensorflow-has-no-attribute-io>

エラー

BrokenPipeError: [Errno 32] Broken pipe

対策

この状態は正常であり、メッセージは単に情報を表示しているだけです

<https://docs.oracle.com/cd/E19455-01/806-2720/msgs-127/index.htm>

|

■ エラーメッセージ

ローカルPCで環境構築する場合

エラー

[WinError 1455] ページング ファイルが小さすぎるため、この操作を完了できません。

対策

仮想メモリを増やす

チェック外す

カスタムサイズを選択

サイズを記入



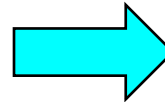
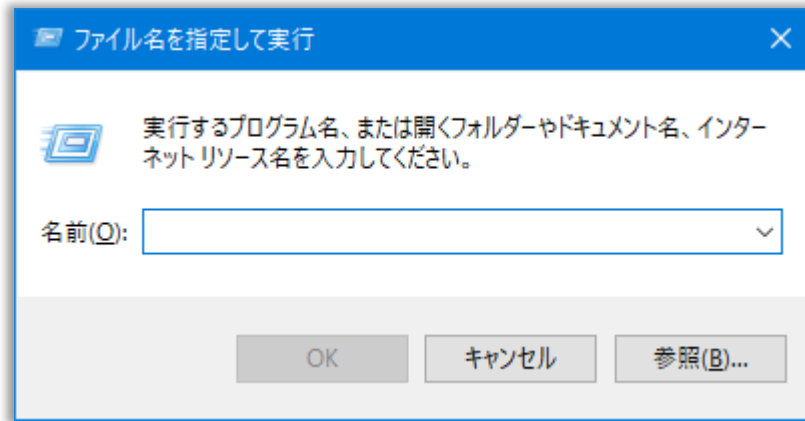
■不具合

エラー

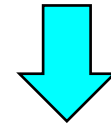
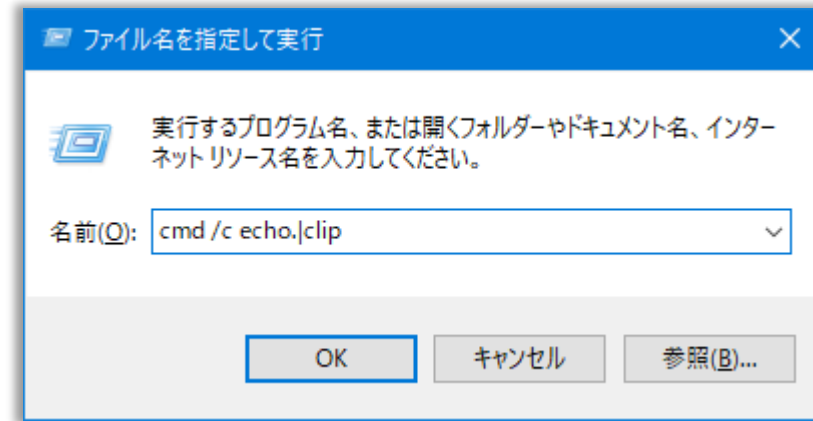
コピー & ペーストができない（できても内容が違う）

対策

Windowsキー + Rを押して、
「ファイル名を指定して実行」
ダイアログを出す



入力欄に「cmd /c echo.|clip」と入力



Enterキー押下

■参考文献

- (1) YOLO V5 を Google Colab で学習する方法
<https://ai-coordinator.jp/yolov5-google-colab>
- (2) Google Colaboratory で YOLOv5 を使用しオブジェクト検出する
<https://laboratory.kazuuu.net/i-used-yolo-v5-with-google-colaboratory/>
- (3) YOLOv5 でマスク検出のトレーニングさせてみる
<https://konchangakita.hatenablog.com/entry/2020/08/17/220000>
- (4) 人工知能／機械学習を学ぶ際の開発環境の注意点
<https://webbigdata.jp/what-is-ai/page-6804/page-8359/page-7728>