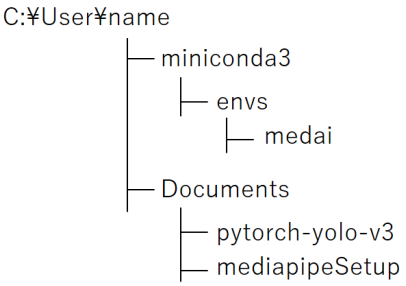
For YOLOv3

本ページは医療×AIの授業において，学習済みNNフレームワークのYOLOv3とMediaPipeをWindows上にPythonをインストールすることで，実行体験ができるためのノートです．

使用するPCはWindows11（10も可？）がインストールされていて，空き容量が5GB以上あるだけで良いですが，今後活用していくためには16GB以上のメモリやGPUなどが搭載されているPCが望ましいです．

既にMiniconda3は導入されているとして，作業を進めます（別途説明書があります）．医療AIのための仮想環境の名称は「medai」として，この中に動作環境を作り上げていきます．また，実行プログラムは「Documents」（マイドキュメント）に格納していくこととします．環境や各ファイルの配置は以下の通りになります．



なお，VScodeも環境構築された方はそちらからも動作することができますが，まずはコマンドプロンプトから動作させる方法で説明を進めていきます．

1. YOLOv3

原著論文：https://arxiv.org/abs/1804.02767

1.1 仮想環境の構築

Minicondaを導入すると，アプリ一覧に以下の様な「Anaconda Prompt」が現れていると思いますので，こちらをダブルクリックして起動します．

現れたコマンドプロンプト画面には以下の様な表示が出ていると思われます．頭の(　)には現在運用されている環境名が示されています．現在は大元の(base)になっています．

(base)の初期状態から，仮想環境(medai)を構築していきます．まずは，以下のように打ち込みましょう．

```
(base) C:¥Users¥YamashoMain>conda create -n medai python=3.10 pip
```

```
conda create -n medai python=3.10 pip
```

たくさんの実行命令が表示されますが，ここは悩まず(笑)，一番下のProceedで"ｙ"を入力してEnterを押します．

```
Proceed ([y]/n)? y
```

いろいろな処理が自動的に行われ，エラーが出ていなければ成功です．

1.2 パッケージのインストール

次に，作成した仮想環境(medai)に入ります．以下の様に打ち込み，先頭が(medai)になることを確認ください．なお，condaはMinicondaの命令となります．

```
(base) C:¥Users¥YamashoMain>conda activate medai
(medai) C:¥Users¥YamashoMain>
```

```
conda activate medai
```

この仮想環境(medai)に使用するパッケージをインストールしていきます．まずはデータベースのpandasと画像処理のopencvを導入します．ここでもインストールする一覧の下にProceedが現れますので，"ｙ"を入力してEnterを押します．

```
(medai) C:¥Users¥YamashoMain>conda install pandas opencv
```

```
conda install pandas opencv
```

```
Proceed ([y]/n)? y
```

続いて，pytorchを入れていきます．pytorchはFacebook（現Meta社）が開発したPython向けのオープンソース機械学習ライブラリです．ネット上で機械学習と言えばkerasやtensorflowが有名ですが，pytorchのほうが設定が楽なので，こちらで構築します．他のパッケージと同様に，下記に打ち込み，"ｙ"を入力してEnterを押します．ちょっと容量が多い（1.4GBぐらい）ので，インストールに5分ぐらいかかる場合もあります．気長に待ちましょう(笑)．

```
(medai) C:¥Users¥YamashoMain>conda install pytorch torchvision -c pytorch
```

```
conda install pytorch torchvision -c pytorch
```

```
Proceed ([y]/n)? y
```

💡 GPU(cuda)を使いたいときには，
conda install pytorch==2.5.1 torchvision==0.20.1 torchaudio==2.5.1 pytorch-cuda=12.1 -c pytorch -c nvidia
とします．詳しくは，https://frqux.hatenablog.com/entry/2023/02/07/062433

最後にmatplotlibとcythonをインストールします．matplotlibはグラフ生成，cythonはC言語インタプリタ―です．YOLOはリアルタイム処理を狙っていますので，高速な処理が必要な部分はC言語で製作されています．

```
(medai) C:¥Users¥YamashoMain>pip install matplotlib cython
```

```
pip install matplotlib cython
```

ここまでで，必要なパッケージのインストールは終了しました．続いて，動作プログラムを導入していきます．コマンドプロンプト命令「cd」でディレクトリを変更しましょう．今回はDocumentsフォルダに動作プログラムを入れていきます．

```
(medai) C:¥Users¥YamashoMain>cd Documents

(medai) C:¥Users¥YamashoMain¥Documents>
```

動作プログラムはgitで導入します．gitはプログラムのソースコードやファイルの変更履歴を記録・追跡・管理するための分散型バージョン管理システムで，動作プログラム一式を簡単に導入することができます．

```
(medai) C:¥Users¥YamashoMain¥Documents>git clone https://github.com/ayooshkathuria/pytorch-yolo-v3.git
```

```
git clone https://github.com/ayooshkathuria/pytorch-yolo-v3.git
```

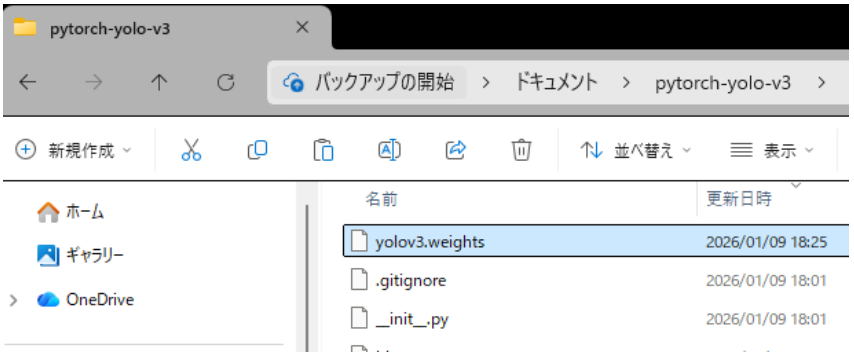
コマンドプロンプト命令「dir」で，Documentsフォルダ下に「pytorch-yolo-v3」というフォルダができていることを確認してください．そして，「cd」で「pytorch-yolo-v3」フォルダに入りましょう．

```
(medai) C:¥Users¥YamashoMain¥Documents>cd pytorch-yolo-v3

(medai) C:¥Users¥YamashoMain¥Documents¥pytorch-yolo-v3>
```

「pytorch-yolo-v3」フォルダにいろいろプログラムや設定ファイルがインストールされていることがわかると思います．続いて，yolo-v3の学習済みの重みをダウンロードします．ネット上にはwgetで取得すると書いてありましたが，wgetの保存先がわかりにくいので，直接ダウンロードして，作業フォルダに直接入れます．以下にアクセスしたらyolov3.weightsが自動的にダウンロードされます．**そのファイルを「pytorch-yolo-v3」フォルダに格納しましょう．**

```
https://pjreddie.com/media/files/yolov3.weights
```



1.3 プログラムの修正

ここまでで環境設定と動作プログラムの入手は完了しました．しかし，Pythonプログラムは環境とのマッチングなどで動かないことがあります．ここでは先駆者さんたちがそのトラブルを回避している事例を使わせて頂き，無事動作させることとします．

＜予めわかっている対策1＞

pytorch-yolo-v3において，そのままではRuntimeErrorが発生します．どうも`"util.py"`というファイルが古いようで，一旦消去し新たに下のリンクのutil.pyに差し替えることで回避できるようです。以下のリンクにアクセスしてください．

https://github.com/ayooshkathuria/pytorch-yolo-v3/blob/3aa94c3fba787ea99e07d0f5c6e2668df0d1baa7/util.py

そして，このutil.pyの，Rawの2つ隣の「download raw file」をクリックしてダウンロードします．ダウンロードしたutil.pyを「pytorch-yolo-v3」フォルダで上書き保存します．

＜予めわかっている対策2＞

「pytorch-yolo-v3」はLinux環境で開発されているため，Windows環境では演算結果を保存するフォルダの指定が異なります（以下のBugFix参照）．

https://github.com/ayooshkathuria/pytorch-yolo-v3/pull/65

「pytorch-yolo-v3」フォルダにある，「detect.py」をメモ帳で開きます．そして以下の処理をします．

- 上部のimportが並んでいる部分の下に，「import platform」を追記します
- 下部にある"det_names = "と書かれている行の頭に＃をつけて（コメントアウト），以下のif，else文を挿入します．Pythonではインデント（字下げ）で行の返還を表しますので，if, else文の位置や字下げ幅を他の行と同じようにします．

```
 （上部）
import platform
...
 （下部）
```

```
if platform.system() == 'Windows':
    det_names = pd.Series(imlist).apply(lambda x: "{}\\det_{}".format(arg
s.det,x.split("\\")[-1]))
else:
    det_names = pd.Series(imlist).apply(lambda x: "{}/det_{}".format(args.
det,x.split("/")[-1]))
```

変更したら，必ず「detect.py」を上書き保存してください．

<予めわかっている対策3>

pytorch-yolo-v3は開発が古いので，新しいOpenCV(今は4.12)に対応できていません（以下のBugFix参照）．
https://github.com/ayooshkathuria/pytorch-yolo-v3/issues/151

再び，「pytorch-yolo-v3」フォルダにある「detect.py」をメモ帳で開きます．そしてdef writeの部分の下にある"c1 =", "c2="文を以下のプログラムに書き換えます．

```
（下部 def writeの部分）
c1 = tuple(map(torch.Tensor.item, x[1:3].int()))
c2 = tuple(map(torch.Tensor.item, x[3:5].int()))
```

変更したら，必ず「detect.py」を上書き保存してください．

1.4 動作確認

インストールや修正が完了したら，いよいよ動作させましょう．プログラムは「detect.py」です．これは「pytorch-yolo-v3」フォルダにあります．コマンドプロンプト画面で，現在の位置が「pytorch-yolo-v3」であることを確認しましょう．

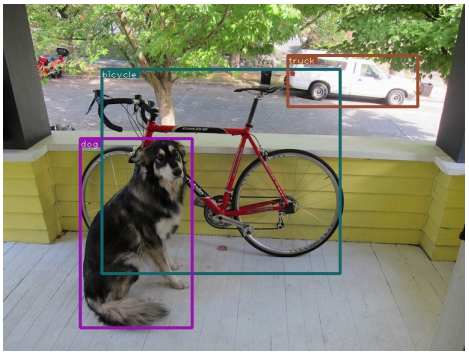
```
(medai) C:¥Users¥YamashoMain¥Documents¥pytorch-yolo-v3>
```

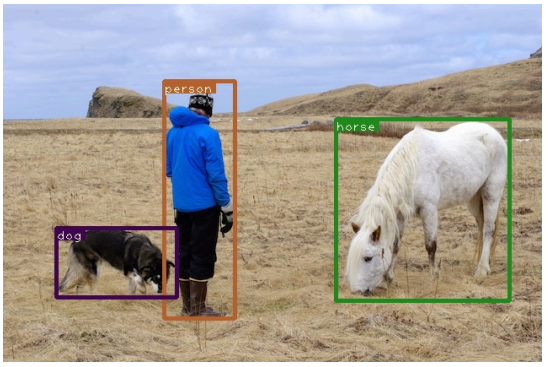
そして，以下の文章で動作させましょう．この命令において，認識処理する画像は「imgs」に入っています．独自の画像で確認するときには，「imgs」フォルダに新たに格納して置いて下さい（.jpgか.pngファイルに限定）．

```
(medai) C:¥Users¥YamashoMain¥Documents¥pytorch-yolo-v3>python detect.py --images imgs --det results
```

```
python detect.py --images imgs --det results
```

演算結果はresults フォルダに格納されていますので，確認ください．





なお，yolo-v3では以下の物体が認識できるように学習されています．

クラス ID	クラス名	クラス ID	クラス名	クラス ID	クラス名	クラス ID	クラス名
0	人	20	ゾウ	40	ワイングラス	60	ダイニングテーブル
1	自転車	21	クマ	41	カップ	61	トイレ
2	車	22	シマウマ	42	フォーク	62	テレビ
3	バイク	23	キリン	43	ナイフ	63	ノートパソコン
4	飛行機	24	リュックサック	44	スプーン	64	マウス
5	バス	25	傘	45	ボウル	65	リモコン
6	電車	26	ハンドバッグ	46	バナナ	66	キーボード
7	トラック	27	ネクタイ	47	リンゴ	67	携帯電話
8	船	28	スーツケース	48	サンドウィッチ	68	電子レンジ
9	信号機	29	フリスビー	49	オレンジ	69	オーブン
10	消火栓	30	スキー	50	ブロッコリー	70	トースター
11	ストップサイン	31	スノーボード	51	キャロット	71	シンク
12	パーキングメーター	32	スポーツボール	52	ホットドッグ	72	冷蔵庫
13	ベンチ	33	カイト	53	ピザ	73	本
14	鳥	34	野球バット	54	ドーナツ	74	時計
15	猫	35	野球グラブ	55	ケーキ	75	花瓶
16	犬	36	スケートボード	56	椅子	76	ハサミ
17	馬	37	サーフボード	57	ソファー	77	テディベア
18	羊	38	テニスラケット	58	鉢植え	78	ヘアードライヤー
19	牛	39	ボトル	59	ベッド	79	歯ブラシ

＜参考サイト＞

https://qiita.com/konan/items/f29ac144f960d0f6a8ec

https://qiita.com/daiarg/items/03c55c82fdc6e623bf07

https://dlbb1.blogspot.com/2020/03/yolo3windowspc.html

https://github.com/eriklindernoren/PyTorch-YOLOv3

https://qiita.com/ys_dirard/items/a82b4ad229c89bb4ba7c

2. MediaPipe

Googleが提供するオープンソースの機械学習フレームワークで、画像や動画から顔、手、姿勢、物体などをリアルタイムで検出・認識・追跡できるツールです．YOLOと同様にpython3.10環境で動作しますので，(medai)の環境下で動作を確認します．

なお，MediaPipeでは**顔検出、手検出、ポーズ検出**が可能ですが，全て画像上の処理になっていますので，別途Webカメラを用意ください．ノートPCで実施する方は内蔵のカメラで十分です．

原著論文：https://arxiv.org/abs/1804.02767

2.1 パッケージのインストール

仮想環境は(medai)ですので，コマンドプロンプトの操作によって，以下の様な表示が出るまで，フォルダ移動をしてください．

```
(base) C:¥Users¥YamashoMain>conda activate medai
(medai) C:¥Users¥YamashoMain>
```

上記フォルダになったら，

```
(medai) C:¥Users¥YamashoMain>pip install mediapipe==0.10.14
```

```
pip install mediapipe==0.10.14
```

再び，フォルダ移動して，Documentsまでやってきます．ここに動作プログラム一式をインストールしていきます．プログラムはYOLOと同様，gitにて転送してきます．

```
(medai) C:¥Users¥YamashoMain>cd Documents
(medai) C:¥Users¥YamashoMain¥Documents>git clone https://github.com/tech-life-hacking/MediapipeSetup.git
```

```
git clone https://github.com/tech-life-hacking/MediapipeSetup.git
```

エクスプローラなどを使い，必ず，Documents（マイドキュメント）フォルダにMediapipeSetupという新しいフォルダができていて，その中に実行プログラムなどが格納されていることを確認してください．

2.2 動作確認

まずはDocuments（マイドキュメント）フォルダから，コマンドプロンプトの"cd"コマンドを用い，MediapipeSetupフォルダに移動します．

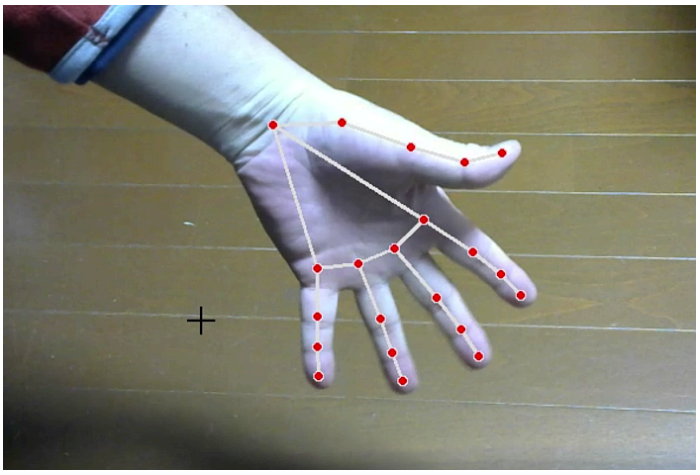
```
(medai) C:¥Users¥YamashoMain¥Documents>cd MediapipeSetup
```

```
cd MediapipeSetup
```

続いて，下記の命令を記載して，Enterで実行します．必ずPCにカメラが付いているかを御確認下さい．

```
(medai) C:¥Users¥YamashoMain¥Documents¥MediapipeSetup>python hands.py
```

```
python hands.py
```



このような表示が出てきたら成功です．終了は「ESC」キーを押します．それでも停止しない場合には，Ctrl+C で強制終了させてもかまいません．

＜参考サイト＞

https://www.techlife-hacking.com/?p=705

https://developers.google.com/mediapipe/solutions/vision/hand_landmarker