EfficientDet 学習の行い方

システム開発部 韓 廷旻

# はじめに

本書は、EfficientDetで作成したデータをLinuxで学習させる方法について記したものである。

# 前提条件

* Linux環境で、GPUが使えること
* nvidiaドライバーがインストールされていること
* cuda9.2が使用できること

# 学習の流れ

本章では、作成した学習データをEfficientDetで学習させるまでの流れについて説明する。

まず、学習の流れについて、図 2.1に記す。

１．学習データの作成

２．学習データのエクスポート

3．GithubからClone

５．学習の開始

windowsで実施

Linuxで実施

4．ファイルの配置

図 . yolov3で学習させるまでの流れ

図 2.1中の青色はWindows環境、オレンジ色はLinuxで操作することを表している。

図 2.1について説明する。

## 学習データの作成

labeling\_v2.jarファイルを用いて、学習データを作成する。

学習データの作成は、次の流れになる。

1. 画像ファイルを用意する
2. 保存先データベースを作成する
3. カテゴリを登録する
4. ラベル付け

以降では、1.～4.について説明する。

### 画像ファイルを用意する

学習に使用する画像データを特定のフォルダに集めておく。複数のフォルダに分かれていてもよい。

本書では、画像ファイルが「d:\data\画像データ\images」にあるものとする(図 3.2)。

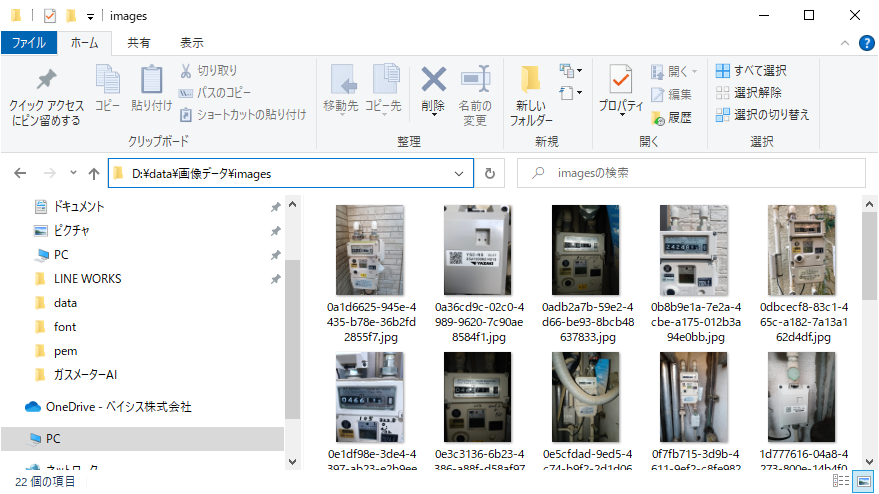


図 . 画像ファイルの準備

### 保存先データベースを作成する

labeling\_v2.jarファイルをダブルクリックして起動する(図 3.3)。

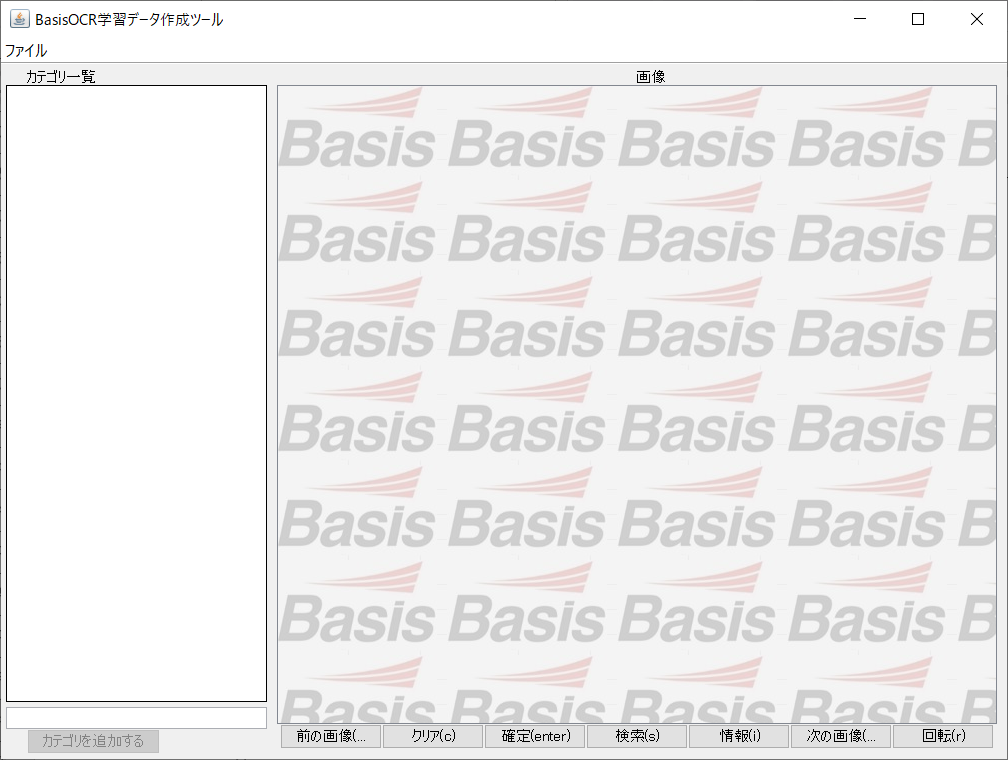


図 . labeling\_v2.jar起動画面

[ファイル]→[プロジェクトを作成]を選択する(図 3.4)。

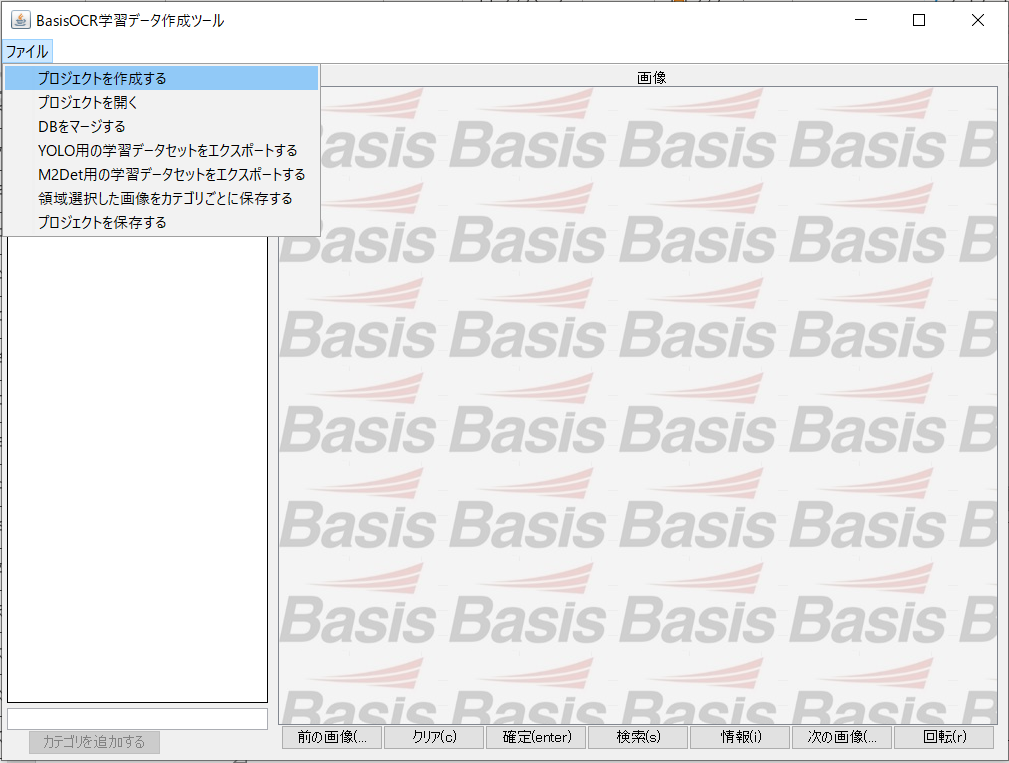


図 .　プロジェクトの作成

「作成するデータベース」に「D:\data\画像データ\sample.db」を指定する。

「学習データ作成画像ディレクトリ」に「D:\data\画像データ\images」を指定する。

入力が終わったら、インポートボタンを押す(図 3.5)。

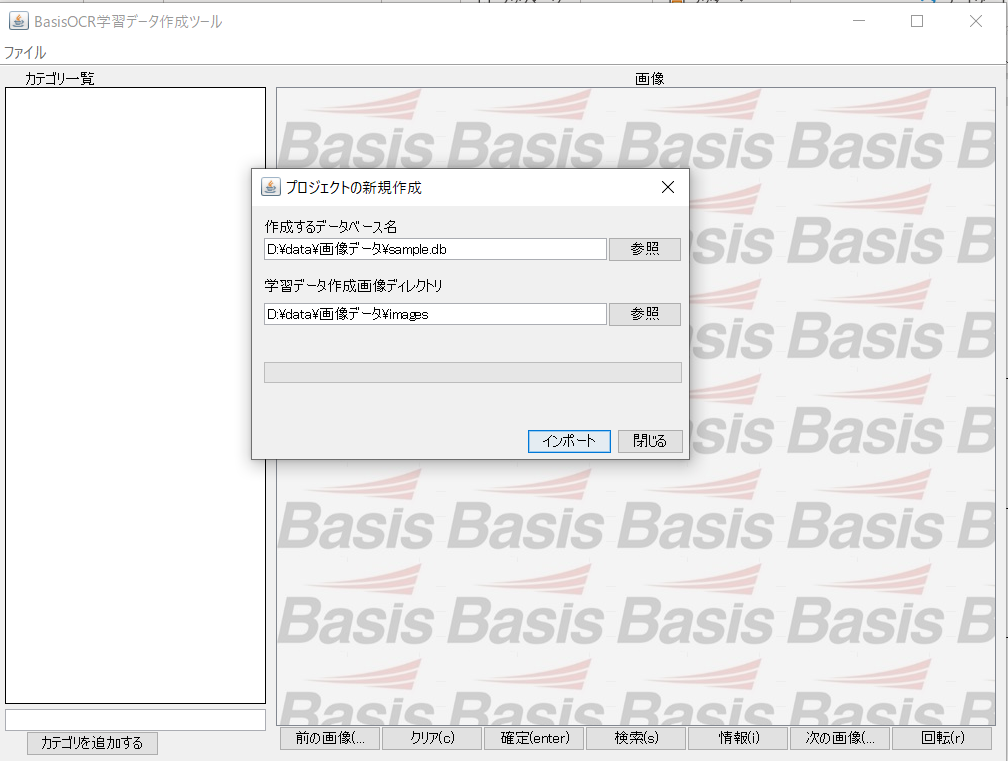


図 .　プロジェクトの作成

画像が表示されることを確認する(図 3.6)。



図 . インポート成功時の画面

また、「作成するデータベース」に指定したパス、「D:\data\画像データ\sample.db」が作成されていること確認する(図 3.7)。

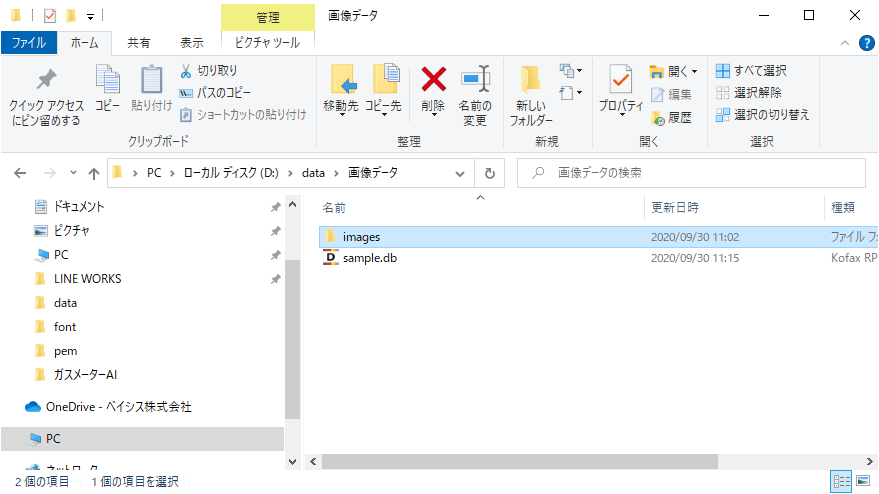


図 .　DB生成時のディレクトリの状態

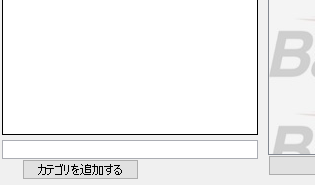
sample.dbには画像ファイルも含めてレコードに保存しているため、別の環境やパスに移動するときは、sample.dbだけを移動すればよく、imagesディレクトリは必要ない。

**[再開する場合]**

bodt\_m2detを閉じ、再開する場合は、「ファイル」→「プロジェクトを開く」を選択し、sample.dbを指定すれば復元できる。

### カテゴリを登録する

カテゴリを登録するには、カテゴリ入力テキストボックスにカテゴリ名を入力し、カテゴリ追加ボタンを押下すると、登録できる(図 3.8)。



カテゴリ名を入力

図 . カテゴリ登録

**ただし、カテゴリ名に日本語を入力しないこと。**日本語で登録すると、EfficientDetが予測時に日本語を認識できずエラーになってしまう。

例では「顧客名」は「Customer」、「住所」は「Address」、「検針値」は「Usage」として追加している(図 3.9)。

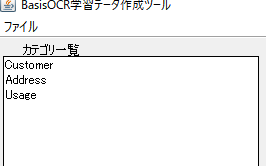


図 . カテゴリ登録例

### ラベル付け

検出させたい領域をドラッグする(図 3.10)。



図 . 領域の作成

領域をドラッグした後、Enterキーを押すと、領域が確定し、矩形が黄色に変わる(図 3.11)。

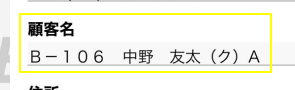


図 . 領域の確定

次に黄色の矩形のカテゴリを指定する。

黄色のボックスの中でマウス右クリックすると、カテゴリ一覧が表示される。表示されたカテゴリの中から、該当カテゴリ名を選択すると、矩形が緑色に変わり、カテゴリ名が表示される(図 3.12)。

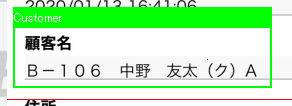


図 . カテゴリを指定

他の項目に対しても、同様の操作を繰り返し、必要なラベリングを行う(図 3.13)。

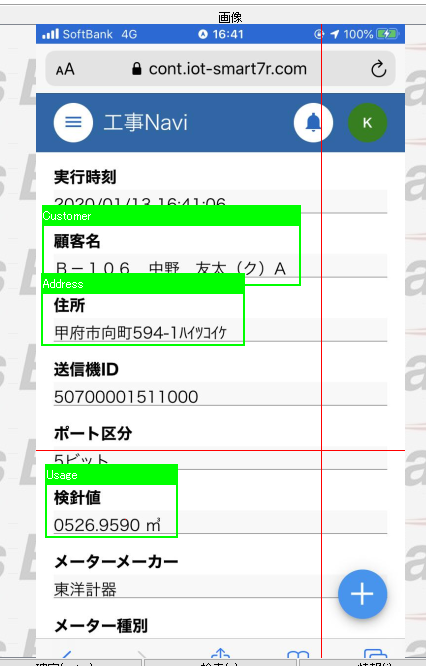


図 . ラベリング完了画面

次の画像にラベリングを行う場合は、右下の「次の画像」ボタンを押すか<n>キーを押すことで、次に進む。

保存する場合は、「ファイル」→「プロジェクトを保存する」を実行すること。

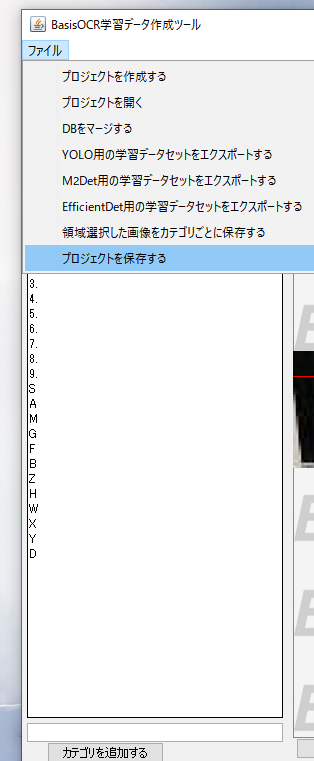


図 . プロジェクトを保存する

## 学習データのエクスポート

作成した学習データをEfficientDetが学習できるデータ形式にエクスポートする。

エクスポートする場合は、「ファイル」→「EfficientDet用の学習データセットをエクスポートする」を選択する(図 3.15)。

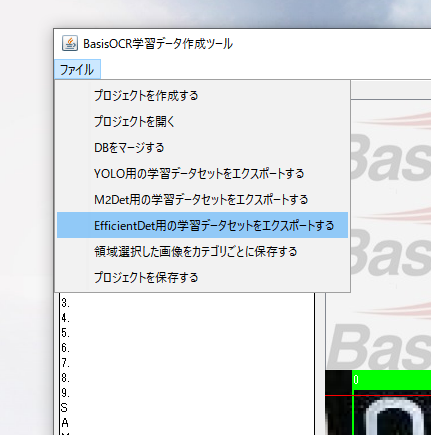


図 . EfficientDet用の学習データをエクスポートする

出力先ディレクトリを聞かれるので、ここでは「C:\Users\PC726\Desktop\output」を指定する。

出力が終わると、次のディレクトリが作成される。

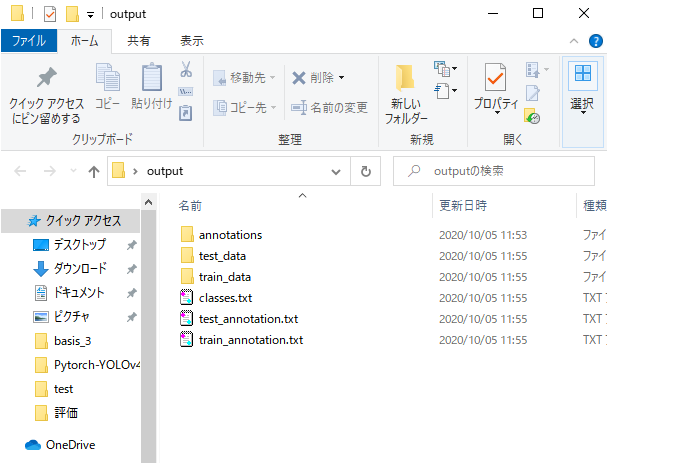


図 . 出力ファイル

## JSONファイルの作成

convertToJSON.pyファイルを「C:\Users\PC726\Desktop\output」に配置する。

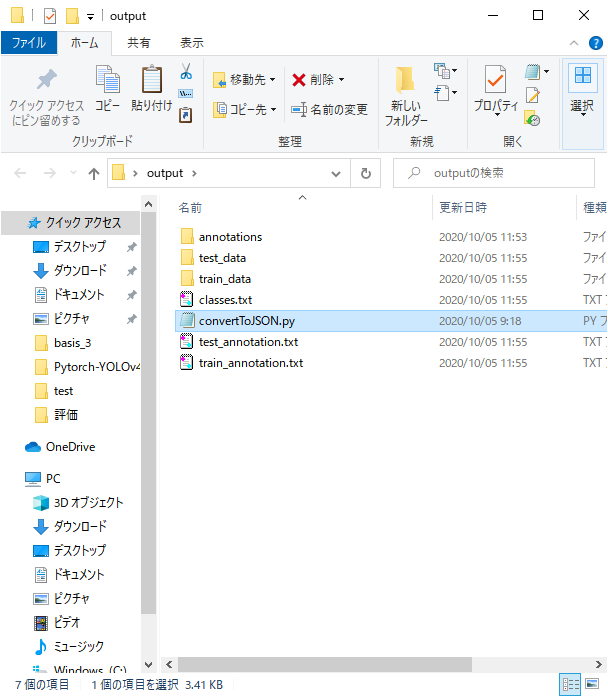


図 3.17 convertToJSON.pyファイルの配置

cmdを開いてconvertToJSON.pyファイルを実行する。実行が終わったらannotationsフォルダにinstances\_train\_data.jsonとinstances\_test\_data.jsonファイルが生成される。

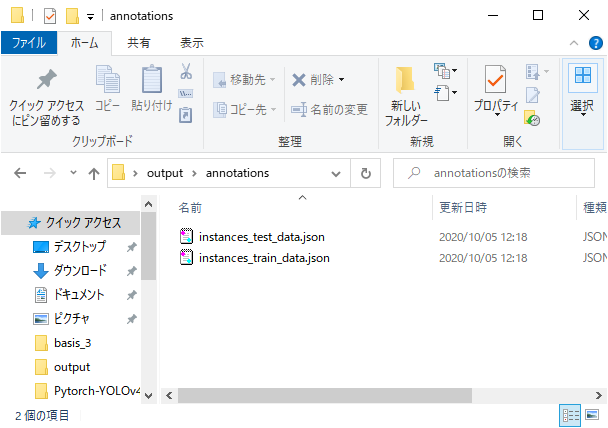


図 3.18 jsonファイルの生成

## LINUX上でGit Clone

本節以降は、Linuxで操作します。例では、/home/share/jmhanの下で作業します。

まず、/home/share/jmhanディレクトリを作成してGithubからCloneする。

|  |
| --- |
| mkdir /home/share/jmhan  cd /home/share/jmhan  git clone https://github.com/zylo117/Yet-Another-EfficientDet-Pytorch.git |

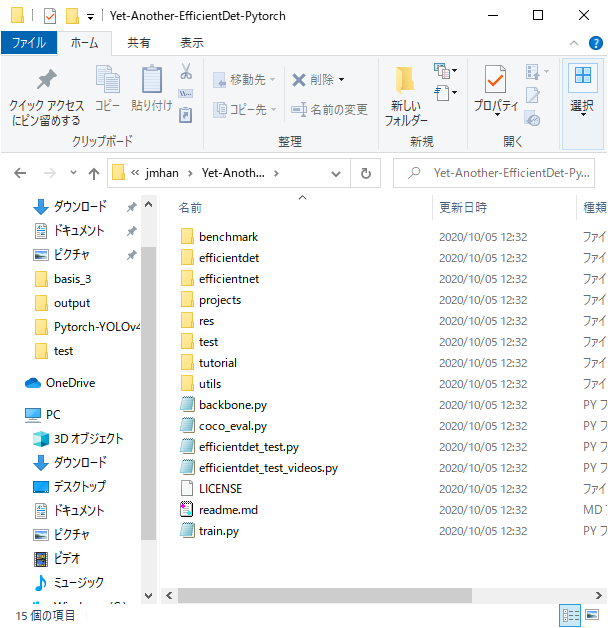


図 3.19 EfficientDetフォルダ

Cloneが終わったらdatasetsディレクトリを生成する。

|  |
| --- |
| mkdir /home/share/jmhan/Yet-Another-EfficientDet-Pytorch/datasets |

## ファイルの配置

次にdatasetsの下にoutputのファイルを配置する(図 3.17)。

なお、Linuxへのファイル転送はsambaサーバを設定して使用している。

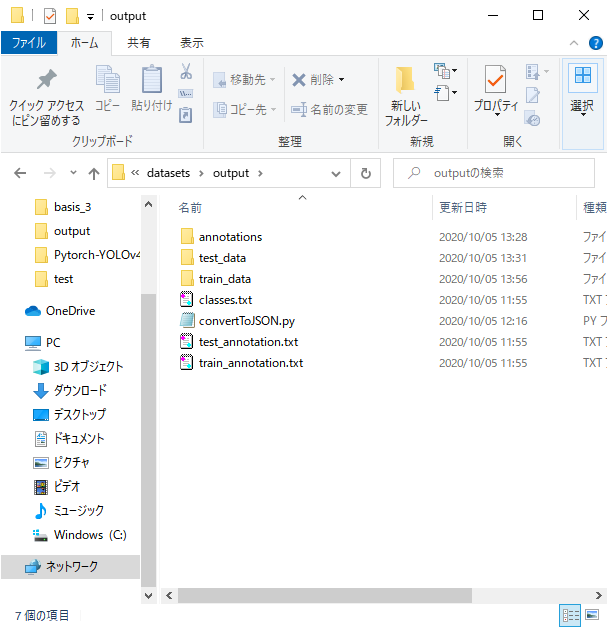


図 .20 Yet-Another-EfficientDet-Pytorch/datasetsにファイルを配置した図

## Pretrained Weightsファイルダウンロード

次のコマンドを実行して、weightsディレクトリを作成します。

|  |
| --- |
| mkdir /home/share/jmhan/Yet-Another-EfficientDet-Pytorch/weights |

<https://github.com/zylo117/Yet-Another-EfficientDet-Pytorch>からPretrained Weightファイルをweightsディレクトリにダウンロードする。

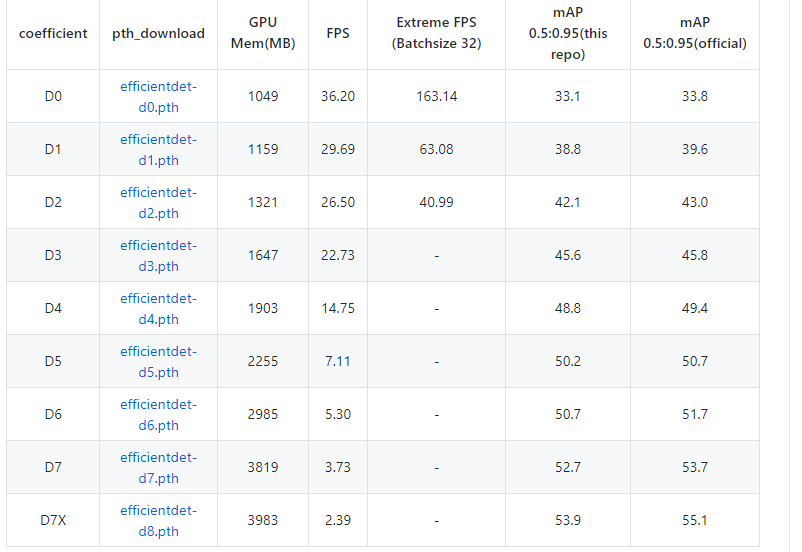


図 3.21 Pretrained Weightファイル

192.168.10.145のPCはGPUメモリ不足の問題でeffcientdet-d2.pthファイルを使った方がよい。

## ymlファイルの作成

ymlファイルはyolov3のcfgファイルみたいに学習の設定を保存するファイル

次のコマンドを実行して、ymlファイルコピーする。

|  |
| --- |
| cd /home/share/jmhan/Yet-Another-EfficientDet-Pytorch/projects  cp shape.yml output.yml |

コピーして、output.ymlファイルを開いて次の内容を作成。

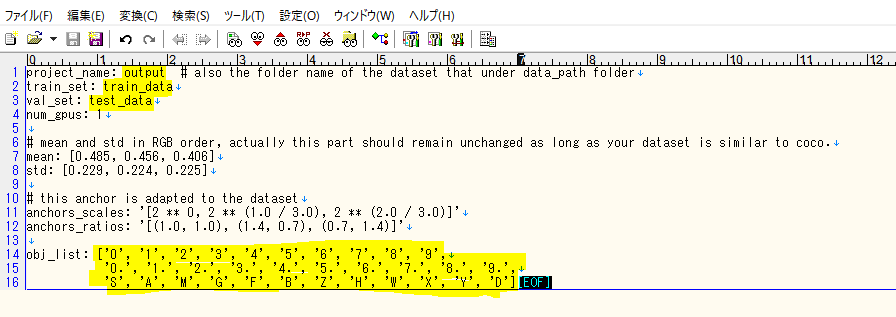


図 3.21 ymlファイルの作成

|  |
| --- |
| project\_name : output(datasetsディレクトリの下のディレクトリ名)  train\_set : train\_data  val\_set : test\_data  obj\_list : ここにはclassを入力 |

## 学習の開始

次のコマンドを実行して、学習を行う。

|  |
| --- |
| cd /home/share/jmhan/Yet-Another-EfficientDet-Pytorch  python3 train.py -c 2 -p output --batch\_size 2 --lr 1e-5 --num\_epochs 10 --load\_weights weights/efficientdet-d2.pth |

実行が進んでいくと/home/share/jmhan/Yet-Another-EfficientDet-Pytorch/logsに学習データが保存されてく。

―以上－