YOLOv8設定マニュアル

Last updated: 2023/06/26 17:36:09

公式HP

Google Colaboratory

この方法が高速

前提

- Googleアカウントをもっていること
- Google Driveを利用していること

実行（物体認識）

- Google Colaboratoryにログイン
- 「ファイル」→「ノートブックを新規作成」
- 「ランタイム」→「ランタイムのタイプを変更」→ハードウェアアクセラレーターを「GPU」にする
- 以下のコードを実行する

```
from google.colab import drive
drive.mount('/content/drive')
%cd ./drive/MyDrive
```

- Google Driveへの接続許可が求められるので，許可する．

```
!git clone https://github.com/ultralytics/ultralytics
%cd ultralytics

!pip install -r requirements.txt

from ultralytics import YOLO
model = YOLO("yolov8x.pt")

results = model("https://ultralytics.com/images/bus.jpg", save=True)
```

- Google Deiveのultralytics\runs\detect\predictフォルダの中に実行結果（jpgファイル）が保存される．

Windows11

前提

- Anacondaがインストールされていること

インストール

- 適当なフォルダで，ターミナルを用いて以下を実行

```
conda create -n yolov8 python=3.11
conda activate yolov8
git clone https://github.com/ultralytics/ultralytics
cd ultralytics
pip install -r requirements.txt
```

実行（物体認識）

- ultralyticsのフォルダに，以下のプログラムをtest.pyとして保存する．

```
from ultralytics import YOLO

model = YOLO("yolov8x.pt")

results = model("https://ultralytics.com/images/bus.jpg", save=True,
save_txt=True, save_conf=True)
boxes = results[0].boxes
for box in boxes:
    print(box.data)
print(results[0].names)
```

- test.pyを実行する．
- runs\detectフォルダの中のpredict?フォルダの中に実行結果（jpgファイル）が保存される．?は整数で，実行するごとに増加する．
- predict?/labelsフォルダには認識結果を記録したテキストファイルが生成される．各行に，物体ID，始点x, 始点y, 終点x, 終点y, 信頼度，が記録されている．
- 認識結果はresults[0].boxes[i].dataにも格納されている．
- 物体IDの一覧はresults[0].namesで得られる．
- modelの第1引数を動画（mp4ファイル）にしても，そのまま解析できる．フレームjの結果はresults[j]に格納される．

実行（骨格推定）

- ultralyticsのフォルダに，以下のプログラムをtest.pyとして保存する．

```
from ultralytics import YOLO

model = YOLO("yolov8x-pose.pt")

results = model("https://ultralytics.com/images/bus.jpg", save=True,
save_txt=True, save_conf=True)
keypoints = results[0].keypoints
```

```
print(keypoints.data)
```

- test.pyを実行する．
- runs\poseフォルダの中のpredict?フォルダの中に実行結果（jpgファイル）が保存される．?は整数で，実行するごとに増加する．
- predict?/labelsフォルダには認識結果を記録したテキストファイルが生成される．各行に，物体ID，始点x, 始点y, 終点x, 終点y, キーポイント1x, キーポイント1y, キーポイント2x, キーポイント2y, ..., 信頼度，が記録されている．
- 認識結果はresults[0].keypoints.dataにも格納されている．
- キーポイントの対応関係は以下の通り．

```
"categories": [
    {
        "supercategory": "person",
        "id": 1,
        "name": "person",
        "keypoints": [
            "nose","left_eye","right_eye","left_ear","right_ear",
            "left_shoulder","right_shoulder",
            "left_elbow","right_elbow","left_wrist","right_wrist",
            "left_hip","right_hip",
            "left_knee","right_knee","left_ankle","right_ankle"
        ],
        "skeleton": [
            [16,14],[14,12],[17,15],[15,13],[12,13],[6,12],[7,13],[6,7],
            [6,8],[7,9],[8,10],[9,11],[2,3],[1,2],[1,3],[2,4],[3,5],[4,6],[5,7]
        ]
    },
    ...
]
```

- modelの第1引数を動画（mp4ファイル）にしても，そのまま解析できる．フレームjの結果はresults[j]に格納される．

GPUの利用

- GPUが利用可能であるかの確認は以下のコードを実行し，Trueであれば利用可能である．

```
import torch
print(torch.cuda.is_available())
```

- 利用可能でない場合は，以下の手順を行う．
    - GPUの型番を確認する．タスクバーを右クリックし，タスクマネージャを起動し，パフォーマンス→GPUで確認する．GPUがNVIDIA製でなければ利用できない．
    - ここからGPUに対応する最新のNVIDIAドライバをダウンロードし，インストールする．

    - ここからCUDA Toolkit 11.8をダウンロードし，インストールする．
    - PytorchのGPU版をインストールする．

```
pip3 install torch==2.0.1 torchvision --extra-index-url
https://download.pytorch.org/whl/cu118
```

    - GPUが利用可能であるか，再確認する．

参考文献

- つくもちブログ，YOLOv8まとめ，https://tt-tsukumochi.com/archives/6275
- MS COCO datasetのフォーマットのまとめ，https://qiita.com/kHz/items/8c06d0cb620f268f4b3e