

抜粋版 はじめてのAI転移学習キット

nvidia tao toolkitを使ってAI 開発スピードを 10 倍に加速

学習編 (TAO版)



スペクトラム・テクノロジー株式会社

<https://spectrum-tech.co.jp>

sales@spectrum-tech.co.jp

学習キット 目次

	ページ
• ubuntu運用マニュアル	
• ubuntuについて	4
• Linux基本コマンド	4
• ubuntu基本操作	6
• 日常運用(ウイルススキャン、更新)	7
• 学習キット 概念図	9
• 学習キット 全体像	10
• ハード概要	11
• ソフト概要	12
• TAO tool kit	
1. 概要	
① 概要	13
② 利用方法	13
③ 概念図	14
④ アプリ一覧	15
2. 事前準備と使い方	16
3. 事例: 物体認識	
① DetectNet_V2	18
② FasterRCNN	34
③ SSD	37
④ YOLOV3	40
⑤ YOLOV4	43
⑥ yolo_v4_tiny	46
⑦ RetinaNet	49
⑧ Efficientdet	52
⑨ DSSD	55

抜粋版のため目次
は一致しません

学習キット 目次

- TAO tool kit

1. 概要	13
2. 事前準備と使い方	16
3. 事例: 物体認識	18
4. 事例: セグメンテーション	
① MaskRCNN	58
② UNET	61
5. 個別事例	
⑫ action_recognition_net	64
⑬ Bpnet	67
⑭ image_classification	70
⑮ Emotionnet	73
⑯ Facenet	76
⑰ Fpenet	79
⑱ Gazenet	82
⑲ Gesturenet	85
⑳ Heartratenet	88
21 Lprnet	91
22 multitask_classification	94

抜粋版のため目次
は一致しません

Ubuntu運用マニュアル

1. Ubuntuについて

Linuxの中でも一番シェアの高いOSです。2004年にDebian系から派生。

2. Linux基本コマンド

① システム関係

- 起動: 電源を入れると自動で起動します。
- 再起動: `$ reboot`
又は、左上のメニューの「ゲストを再起動」
- 終了: `$ shutdown`
又は、左上のメニューの「ゲストをシャットダウン」
- ログアウト `$ exit`
ルートからログアウトします
- **日本語／英語の入力切替**: 半角／全角のボタン(ESCボタンの下)

Ubuntu運用マニュアル

2. Linux基本コマンド

② ディレクトリ操作、コピー、移動、削除

`masa@ubuntu:~$ cd /home/masa/Documents` ディレクトリの切り替え
`masa@ubuntu:~/home/masa/Documents$ ls` ファイルとディレクトリの表示(表示したら操作したいファイルを右クリックでコピーして操作します)
`masa@ubuntu:~$ cp` ファイル名 ディレクトリ 配下のディレクトリのファイルを別のディレクトリへコピー
`masa@ubuntu:~$ mv` ファイル名 ディレクトリ 配下のディレクトリのファイルを別のディレクトリへ移動
`masa@ubuntu:~$ rm` ファイル名 ファイルの削除
 便利な機能 `rm -help` コマンドのオプションが分からない場合は、ヘルプで問い合わせる。すべてのコマンド共通(マイナスを2個とhelp)

③ ユーザ権限、プロセス他

`masa@ubuntu:~$ su -` スーパーユーザ(root)に切り替え、パスワードを入力
`masa@Ubuntu:~$ sudo` ルート権限で各種コマンドを実施します。
`masa@ubuntu:~$ ps a` 現状の動いているプロセスを表示
`masa@ubuntu:~$ kill` 特定のプロセスを強制終了
`masa@ubuntu:~$ apt-get install pkg` パッケージのインストールなどに使用
`masa@ubuntu:~$ date` 日付、時間の設定を行います。
`masa@ubuntu:~$ leafpad /etc/network/interfaces` インタフェースに記述している内容を変更します。Viよりも使いやすいです。

④ モジュール、usb、メモリ、HDDなどの表示

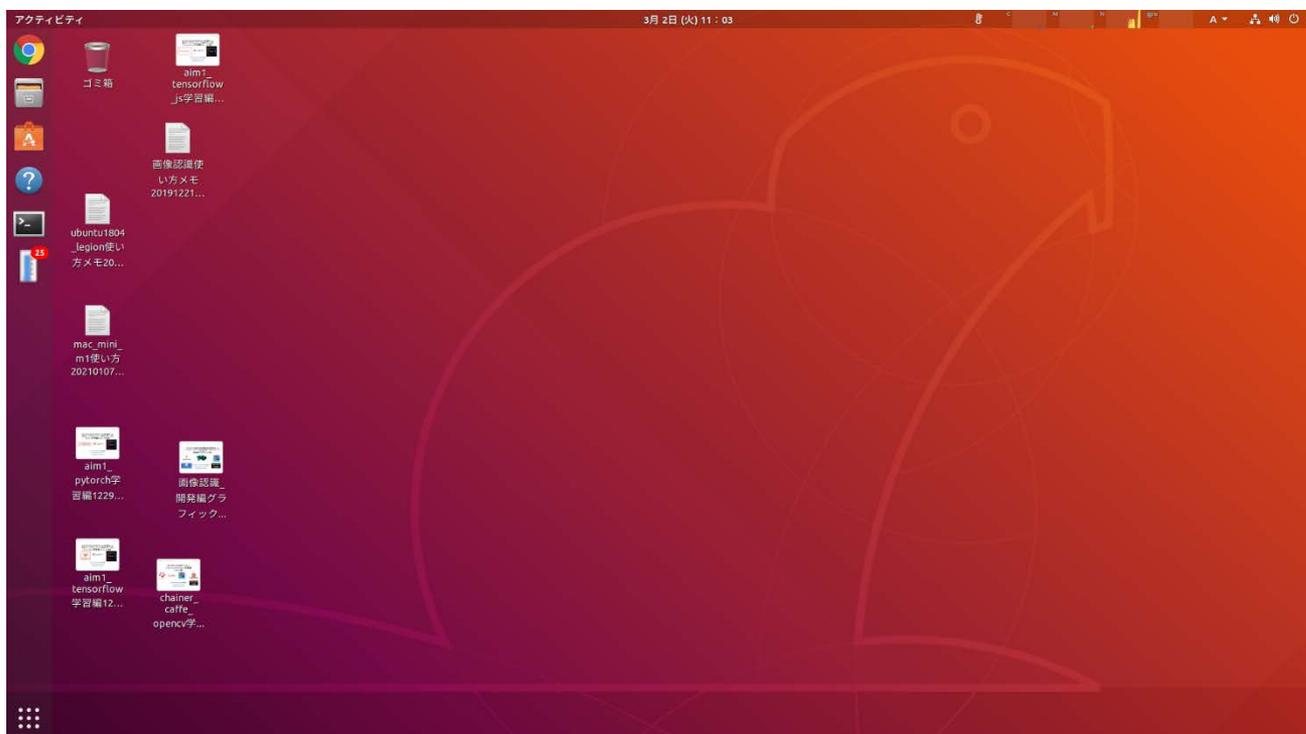
`masa@ubuntu:~$ lsmod` linuxのモジュールリスト表示
`masa@ubuntu:~$ lsusb` usbのデバイス表示
`masa@ubuntu:~$ free -mt` メモリ使用状態表示
`masa@ubuntu:~$ df` HDD(マイクロSD)の使用状態表示

Ubuntu運用マニュアル



3. 基本操作

① 表示画面と内容



主に使用するもの

- ・ブラウザ: Chrome
- ・フォルダ: Documents内に必要なファイルがあります。
- ・コマンド: コマンド画面を立ち上げて、python3のプログラムを動作させます。

Ubuntu運用マニュアル

4. 日常運用

① セキュリティ対策(アンチウイルス更新、スキャン)

- アンチウイルス対策として無料のclamAVをインストールしてます。
- 手動での運用を基本としています。

```
masa@ubuntu: ~  
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)  
masa@ubuntu:~$ sudo freshclam  
Fri Jul 12 09:58:58 2019 -> ClamAV update process started  
2019  
Fri Jul 12 09:58:58 2019 -> ^Your ClamAV installation is OK!  
Fri Jul 12 09:58:58 2019 -> ^Local version: 0.100.3 Recommended  
Fri Jul 12 09:58:58 2019 -> DON'T PANIC! Read https://www.clamav.net/  
pgrading-clamav  
Fri Jul 12 09:58:58 2019 -> main.cvd is up to date (version: 100, f-  
level: 60, builder: sigmgr)  
Fri Jul 12 09:58:58 2019 -> daily.cld is up to date (version: 19, f-  
level: 63, builder: rayman)  
Fri Jul 12 09:58:58 2019 -> bytecode.cvd is up to date (version: 328, stgs: 94,  
f-level: 63, builder: neo)  
masa@ubuntu:~$ sudo clamscan --infected --remove --recursive
```

パターンファイル更新

```
$ sudo freshclam
```

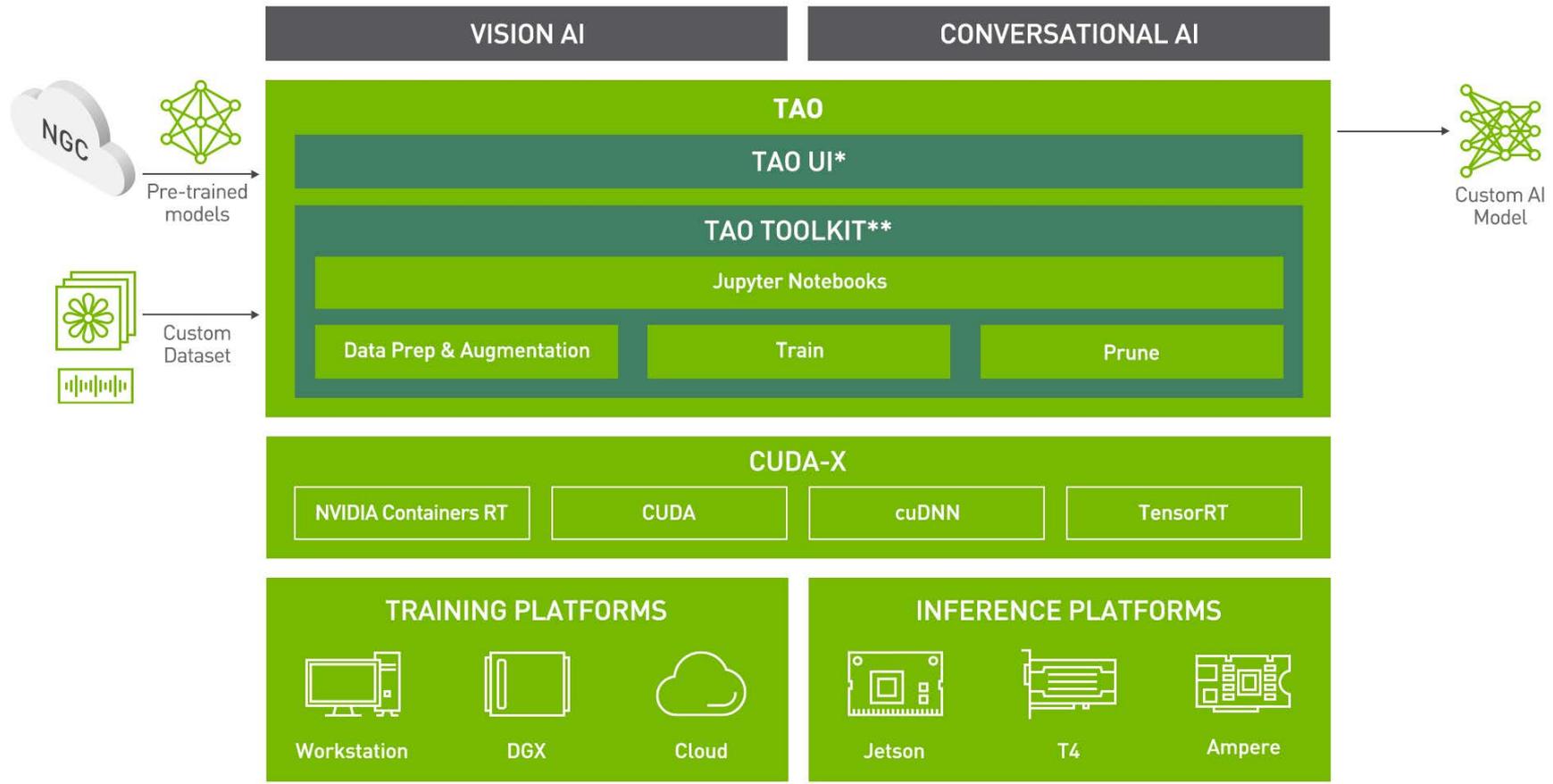
手動スキャン時にも更新されます

手動でスキャン

```
$ sudo clamscan --infected --remove --recursive
```

自動化可能ですが、バックグラウンドで重くなる可能性大。コマンド入力後時間がかかります。

学習キット 概念図(TAO版)



* Coming Soon

** Formerly Transfer Learning Toolkit

学習キット 全体像(TAO版)

ハードウェア

CPU



8 core cpu, 32GB RAM

+

GPU



Nvidia A100,
V100, RTX30シリーズに対応

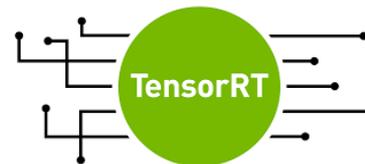
USBメモリ



OS



AI系



TensorFlow



ソフトウェア



画像系



プログラム言語



グラフィック



ハードウェア概要

①必要なハードウェア仕様

ハードウェアの概要です。

区分	プロダクト	メーカー	備考
USBメモリ (tao版)	512GB USB3.0 TAO 関連ソフト		
お客様準備品			
PC本体	cpu	Intel i9, i7, amd Ryzen 9,7など 8 core cpu	GPUが搭載できるものに限ります。
	GPU	A100, V100 RTX30シリーズ	nvidia
	メモリ	32GB以上	
	SSD	512GB以上	巨大なdatabaseをインストールするため1TBが望ましい

2. ソフトウェア概要

①ソフトウェア一覧

ソフトウェアの概要です。

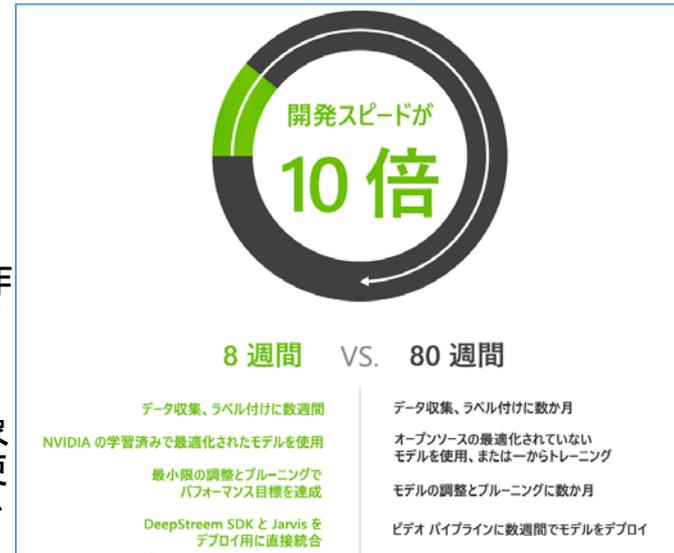
区分	ソフト名	バージョン	備考
OS	ubuntu	18.04.3 LTS	
GPU用	cuDNN	8.2.4+cuda11.4	Nvidia用,搭載するGPUに依存 本キットでは、コンテナで使用 するため参考
プログラム言語	python3	3.6.9	仮想化で使用
TAO	TAO tool kit	3.0.21.8	
コンテナ関連	Docker-ce	>19.03.5	
	nvidia-container-toolkit	>1.3.0-1	
	Nvidia-driver	>465	
AI用プログラム	tensorRT	8.4	
	tensorflow	1.15	Taoコンテナで使用
	Pytorch	1.8	
	onnx	1.8.1	
各種モジュール	Jupyter notebook、 matplotlibなど多数のpipラ イブラリ		

Tao toolkit

1. 概要

① 概要

- お金をかけて AI の専門知識を習得しなくても、AI 開発スピードを **10 倍** に加速できます。トレーニングをスピードアップして、精度の高い、高性能のドメイン別 AI モデルをすばやく作成できます。
- ビジネスの課題を解決するための AI/ML モデルを一から作成するのは、お金と時間がかかります。**転移学習** は、学習済みの特徴量を、既存のニューラル ネットワーク モデルから新しいモデルに抽出するときによく使われる手法です。**NVIDIA TAO Toolkit** は AI/DL フレームワークの複雑さを抽象化する AI ツールキットです。高品質の学習済みモデルを使用するため、わずかな量のデータでも運用品質のモデルをすばやく構築できます。



② 利用方法

- TAO を使用すると、NVIDIA の運用品質の学習済みモデルをそのままデプロイするか、コンピュータービジョンや対話型 AI などのさまざまなユースケースに合わせてモデルを微調整して使用できます。TAO は、AI 作成のための UI ベースのガイド付きワークフロー、**TAO (Train, Adapt and Optimize)** プラットフォームの中核的なコンポーネントです。
- <https://docs.nvidia.com/tao/tao-toolkit/text/overview.html>

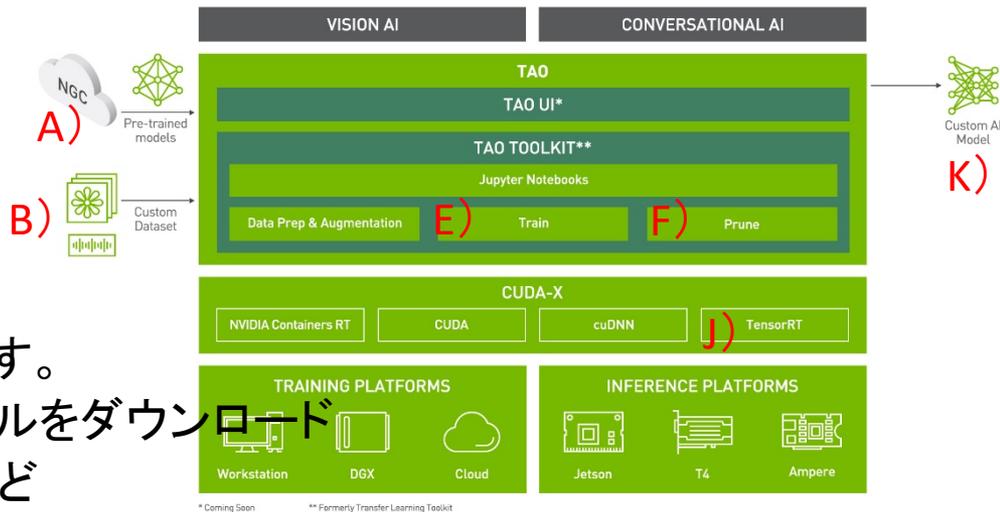
Tao toolkit

1. 概要

③ 概念図

以下の手順で転移学習を行います。

- A) ngcで作成済の学習済みモデルをダウンロード
- B) カスタムデータを設定:cocoなど
- C) カスタムデータをtensorflow用tfrecordに変換
- D) Tfrecordを使って学習
- E) 学習したモデルの評価: **時間がかかります**(モデルによりrtx3080で10時間など)
- F) モデルのプルーン化(prune): 切り落とし
- G) プルーン化モデルの再学習: **時間がかかります**
- H) 再学習モデルの再評価
- I) 推論(inference)
- J) モデルのエクスポート(int8最適化、tensorRT変換)
- K) エクスポートモデルでの推論



別売: [TensorRT](#)
学習・開発キット
を参照ください

1. 概要

④. アプリ一覧

id	区分	アプリ名	概要	tao版	deepstream版 先頭数字は、目次、 二つ目が細分
1	物体検出	DetectNet_V2	車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、学習モデルを作成。その後、prune、量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現	●	●5_7
2		FasterRCNN	同上のデータセット、事前学習モデルで、Faster RCNNモデルで学習する。正解率は、高いが、処理が遅い	●	●3_6
3		SSD	SSD(single shot detector)は、画像中の物体を単一のディープニューラルネットワークで検出する。一般には、認識精度も高く高速といわれているが？結果は、反対。	●	●3_5
4		YOLOV3	darknetで有名なyolo、高速認識が売り	●	●3_7
5		YOLOV4	darknetで有名なyolo、高速認識が売り、新たな開発者が担当	●	
6		yolo_v4_tiny	darknetで有名なyolo、高速認識が売り、新たな開発者が担当、小型モデル版(軽い)	●	
7		RetinaNet	一般的には、2段階で検出していたものを、速度を維持したままで、精度が高い一段階検出モデルができなかと考え、RetinaNetが発表されました	●	
8		efficientdet	efficient detを使った物体認識	●	
9		DSSD	deconvolutional single shot detector(DSSD)	●	
10	セグメンテーション	MaskRCNN	物体のセグメンテーションして、背景をマスクする。	▲エラー	●5_6
11		UNET	医療系の画像を使って、血管などをセグメンテーションする	●	
12	個別事例	action_recognition_net	ビデオから行動認識を行います	▲エラー	▲3_17エラー
13		bpnet	body pose netは、からだの目、耳、手首などの位置を検出します。	●	
14		image_classification	物体の検出します	●	▲5_8エラー
15		emotionnet	感情分類	●	▲5_3エラー
16		facenet	顔検出	●	
17		fpenet	顔の部位検出	●	
18		gazenet	顔から視線を推測	●	▲5_4エラー
19		gesturenet	指の動作を検出	●	▲5_5エラー
20		heartratenet	心電図による推測	▲データ未入手	
21		lprnet	ナンバープレートの文字認識	●	▲4_8エラー
22		multitask_classification	ファッション用品の分類	●	
23		dashcamnet	車種、メーカー名などを識別		●4_3
24		FaceDetectIR	顔識別		●4_4
25		VehicleMakeNet	車種、メーカーの認識		●4_5
26		vehicletypenet	車種の認識: クーペ、suv、ban		●4_6
27		PeopleNet-ResNet34	人認識: deepstreamer使用		●4_2
28		TrafficCamNet	車両認識: deepstreamer使用		●4_1
29		LPD	ナンバープレート検出		▲4_7エラー
30		Facial Landmark	顔の細部の位置を認識		●5_2
31		PeopleSegNet	人体セグメンテーション		●5_9
32		People Semantic Segmentation	人体セグメンテーション		●5_10
33		2D Body Pose Estimation	2D人体のポーズ認識		●5_1
34		imagedata-multistream	2画面ストリーミング		●2_12

deepstream版の
エラーは、engine生
成の問題。Tao-
converterのエラー
不明

2. 事前準備と使い方

入カコマンド

```
$ source /home/masa/venv_py36/bin/activate
```

```
$ tao --help
```

① python仮想環境

- 設定編で準備したvenv_py36で全て動作させます。

```
$ source /home/masa/venv_py36/bin/activate
```

```
(venv_py36) masa@ubuntu2 :$
```

② TAO tool kitの動作方法

- 動作方法には以下の二つがあります。

- https://docs.nvidia.com/tao/tao-toolkit/text/tao_toolkit_quick_start_guide.html

A) Jupyter notebook (推奨)

- 基本的にjupyter notebookで動作させます。全体の流れがわかるので理解しやすいです。

B) Tao cli

- Tao の個別コマンドを使って使用。
- .tao_mounts.jsonを準備してから操作、jupyter notebookで解説しています。

```
$ tao --help      ヘルプコマンドで内容を確認
```

```
$ tao list        動作中のtao コンテナを確認できます
```

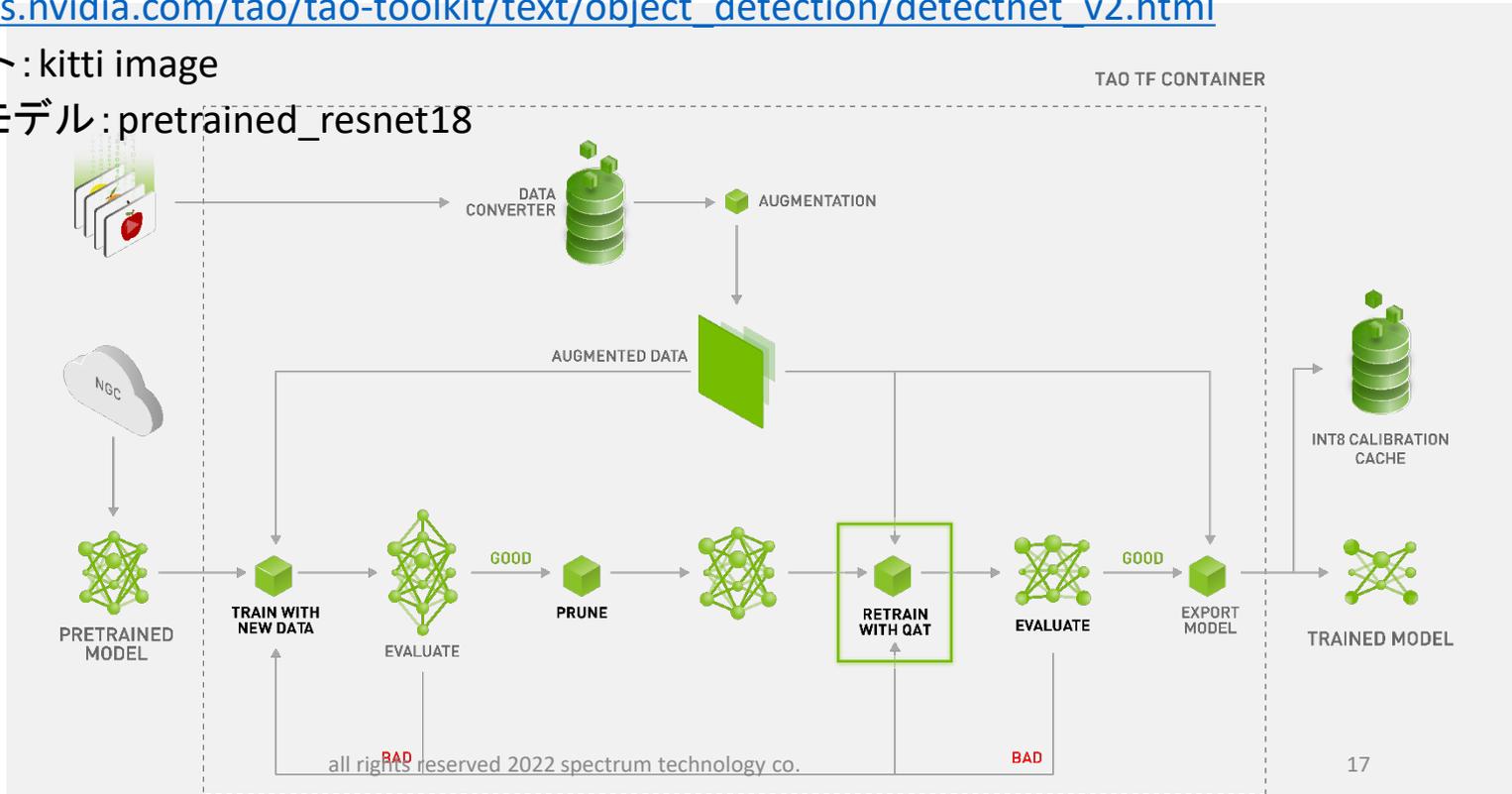
TAO Tool kit

3.事例：物体認識

① detectnet_v2：物体認識

- 概要：車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、[detectnet_v2](https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html)アルゴリズムを使い学習モデルを作成。その後、prune, 量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現。
- https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html
- データセット：kitti image
- 事前学習モデル：pretrained_resnet18

全体図



TAO Tool kit

3.事例：物体認識

① detectnet_v2：物体認識

- 概要：車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、学習モデルを作成。その後、prune, 量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現。

• https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html

• データセット:kitti image

• 事前学習モデル: pretrained_resnet18

ディレクトリは各自
違います

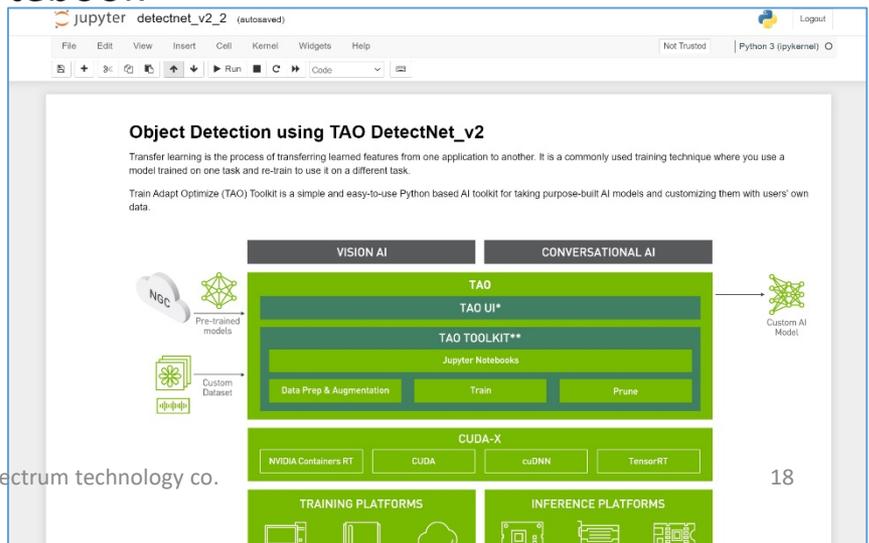
```
$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/detectnet_v2
```

```
(venv_py36) masa@ubuntu2 :$ jupyter notebook
```

detectnet_v2_2.ipynbを選択

オリジナルは、
detectnet_v2.ipyn
bです。

```
入カコマンド  
$ $ cd  
/home/masa/Documents/tao/cv_samples_v1.3.0/det  
ectnet_v2  
$ jupyter notebook
```



TAO Tool kit

3.事例：物体認識

① detectnet_v2: 物体認識

- 概要: 車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、detectnet_v2アルゴリズムを使って学習モデルを作成。その後、prune, 量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現。
- https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html

detectnet_v2_2.ipynbを選択

When using the purpose-built pretrained models from NGC, please make sure to set the \$KEY environment variable to the key as mentioned in the model overview. Failing to do so, can lead to errors when trying to load them as pretrained models.

The following notebook requires the user to set an env variable called the \$LOCAL_PROJECT_DIR as the path to the users workspace. Please note that the dataset to run this notebook is expected to reside in the \$LOCAL_PROJECT_DIR/data, while the TAO experiment generated collaterals will be output to \$LOCAL_PROJECT_DIR/detectnet_v2. More information on how to set up the dataset and the supported steps in the TAO workflow are provided in the subsequent cells.

Note: Please make sure to remove any stray artifacts/files from the \$USER_EXPERIMENT_DIR or \$DATA_DOWNLOAD_DIR paths as mentioned below, that may have been generated from previous experiments. Having checkpoint files etc may interfere with creating a training graph for a new experiment.

Note: This notebook currently is by default set up to run training using 1 GPU. To use more

```
In [1]: # Setting up env variables for cleaner command line commands.
import os

%env KEY=lit_encode
%env NTAO_GPU=1
%env USER_EXPERIMENT_DIR=/workspace/tao-experiments/detectnet_v2
%env DATA_DOWNLOAD_DIR=/workspace/tao-experiments/data

# Set this path if you don't run the notebook from the samples directory.
%env NOTEBOOK_ROOT=/tao-samples/detectnet_v2

# Please define this local project directory that needs to be mapped to the TAO docker
# The dataset expected to be present in $LOCAL_PROJECT_DIR/data, while the results
# in this notebook will be stored at $LOCAL_PROJECT_DIR/detectnet_v2
# PLEASE MAKE SURE TO UPDATE THIS PATH!
os.environ["LOCAL_PROJECT_DIR"] = "/home/masa/Documents/tao"

os.environ["LOCAL_DATA_DIR"] = os.path.join(
    os.getenv("LOCAL_PROJECT_DIR", os.getcwd()),
    "data"
)

os.environ["LOCAL_EXPERIMENT_DIR"] = os.path.join(
    os.getenv("LOCAL_PROJECT_DIR", os.getcwd()),
    "detectnet_v2"
)

# The sample spec files are present in the same path as the downloaded samples.
os.environ["LOCAL_SPECS_DIR"] = os.path.join(
    os.getenv("NOTEBOOK_ROOT", os.getcwd()),
    "specs"
)

%env SPECS_DIR=/workspace/tao-experiments/specs

# Showing list of specification files.
!ls -l $LOCAL_SPECS_DIR

env: KEY=lit_encode
```

ローカルプロジェクト・ディレクトリ設定、各自変更のこと

jupyter detectnet_v2_2 (autosaved)

```
In [2]: # Mapping up the local directories to the TAO docker.
import json
mounts_file = os.path.expanduser("~/tao_mounts.json")

# Define the dictionary with the mapped drives
drive_map = {
    "Mounts": [
        # Mapping the data directory
        {
            "source": os.environ["LOCAL_PROJECT_DIR"],
            "destination": os.getenv("LOCAL_EXPERIMENT_DIR",
                "/workspace/tao-experiments")
        },
        # Mapping the specs directory,
        {
            "source": os.environ["LOCAL_SPECS_DIR"],
            "destination": os.getenv("SPECS_DIR",
                "/workspace/tao-experiments/specs")
        }
    ]
}

# Writing the mounts file.
with open(mounts_file, "w") as mfile:
    json.dump(drive_map, mfile, indent=4)
```

.tao_mounts.json 設定

```
In [3]: !cat ~/tao_mounts.json

{
  "Mounts": [
    {
      "source": "/home/masa/Documents/tao",
      "destination": "/workspace/tao-experiments"
    },
    {
      "source": "/home/masa/Documents/tao/cv_samples_v1.3.0/detectnet_v2/specs",
      "destination": "/workspace/tao-experiments/specs"
    }
  ]
}
```

all rights reserved 2022 spectrum technology co.

19

1. Install the TAO launcher

The TAO launcher is a python package distributed as a python wheel listed in the 'nvidia-pyindex' python index. You may install the launcher by executing the

TAO Tool kit

3.事例：物体認識

① detectnet_v2：物体認識

- 概要：車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、学習モデルを作成。その後、prune, 量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現。
- https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html
- detectnet_v2_2.ipynbを選択

E. Inference using QAT engine

Run inference and visualize detections on test images, using the exported TensorRT engine from [Section C](#).

```
In [16]: tao detectnet_v2 inference -e $SPECIES_DIR/detectnet_v2_inference_kitti_e2lt_out.txt \
        -o $USER_EXPERIMENT_DIR/tao_infer_testing_out \
        -i $DATA_DOWNLOAD_DIR/testing_image_2 \
        -k KEY
```

2022-04-14 18:58:16,330 [INFO] root: Restiry: [nvcr.io]
 2022-04-14 18:58:16,372 [INFO] tti.components.instance_handler.local_instance: Running command in container
 tti v2 21 tti tti 15 4ms
 2022-04-14 18:58:16,383 [WARNING] tti.components.docker_handler.docker_handler: Docker will run the commands as root. If you would like to retain your local host permissions, please add the "user" UID/GID in the DockerOptions portion of the "/home/nvidia/.tao_mounts.json" file. You can obtain your users UID and GID by using the "id -u" and "id -g" commands on the terminal.
 Using TensorFlow backend.
 Using TensorFlow backend.
 WARNING:tensorflow:Deprecation warnings have been disabled. Set TF_ENABLE_DEPRECATION_WARNINGS=1 to re-enable.
 2022-04-14 09:58:20,477 [INFO] iba.detectnet_v2.spec.handler.spec_loader: Merging specification from /workspace/tao_experiments/specs/detectnet_v2_inference_kitti_e2lt_out.txt
 2022-04-14 09:58:20,485 [INFO] __main__: Creating output inference directory
 2022-04-14 09:58:20,486 [INFO] __main__: Overlay images will be saved in the output path.
 2022-04-14 09:58:20,489 [INFO] iba.detectnet_v2.inferencer.built_inferencer: Constructing inferencer
 2022-04-14 09:58:20,922 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Reading from engine file at: /workspace/tao_experiments/detectnet_v2_experiment_dir_final/resnet18_detector_out.trt.int8
 2022-04-14 09:58:21,622 [INFO] __main__: Initialized model
 2022-04-14 09:58:21,630 [INFO] __main__: Commencing inference
 100% |#####| 470/470 [07:44:00.00, 1.01it/s]
 2022-04-14 10:06:06,530 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Clearing input buffers.
 2022-04-14 10:06:06,531 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Clearing output buffers.
 2022-04-14 10:06:06,532 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Clearing tensorrt runtime.
 2022-04-14 10:06:06,533 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Clearing tensorrt context.
 2022-04-14 10:06:06,533 [INFO] iba.detectnet_v2.inferencer.trt_inferencer: Clearing tensorrt engine.
 2022-04-14 10:06:06,538 [INFO] __main__: Inference complete
 2022-04-14 19:06:07,430 [INFO] tti.components.docker_handler.docker_handler: Stopping container.

推論と可視化

```
In [20]: # visualize the first 12 inferred images.
OUTPUT_PATH = "tti_infer_testing_out/images_annotated" # relative path from $USER_EXPERIMENT_DIR
COLS = 4 # number of columns in the visualizer grid
IMAGES = 12 # number of images to visualize.
visualize_images(OUTPUT_PATH, num_cols=COLS, num_images=IMAGES)
```

推論結果の可視化

ll rights reserved 2022 spectrum technology co.

TAO Tool kit

3.事例：物体認識

① detectnet_v2: 物体認識

- 概要: 車、バン、トラック、歩行者、サイクリストのデータからresnet18の学習モデルを使って、検出対象を3個(車、サイクリスト、歩行者)に絞り、学習モデルを作成。その後、prune, 量子化などを使い、最終的にtensorRT用のモデルとして出力。Taoで最初に使ったモデルに比べて、12倍に高速化し、正解率もほぼ同じを実現。
- https://docs.nvidia.com/tao/tao-toolkit/text/object_detection/detectnet_v2.html
- detectnet_v2_2.ipynbを選択
- 認識率と推定時間の各モデル別推移

Tao当初 評価結果		tao平均推論時間 (sec): ①	prune後の再学習結果		量子化後の評価結果		tensorRT後の評価結果		tensorRT後の平均推論時間(sec): ②	高速化倍率: ①/②	tao学習、prune後再学習時間の合計 (hour)
class name	average precision (in %)		class name (in %)	average precision	class name (in %)	average precision	class name (in %)	average precision			
car	79.5795	0.005441	car	80.2562	car	75.3572	car	74.9865	0.000436	12.48	4
cyclist	80.3184		cyclist	81.5315	cyclist	77.1818	cyclist	77.9812			
pedestrian	65.9508		pedestrian	65.9992	pedestrian	65.6729	pedestrian	66.0452			

最初の推論より、12倍高速化

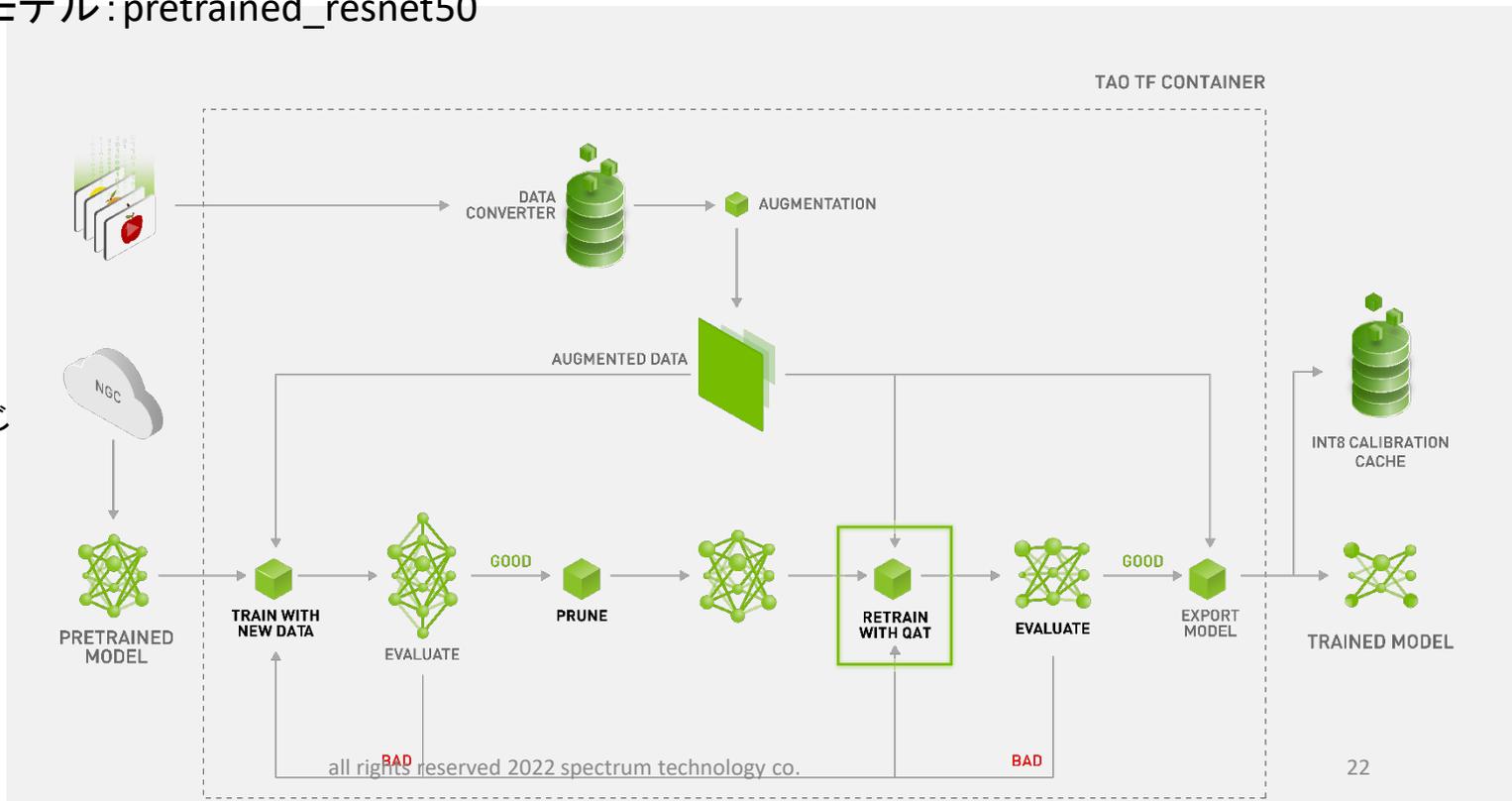
4.事例：セグメンテーション

⑪ unet: セグメンテーション

- 概要: 医療系の画像を使って、血管などをセグメンテーションする。[Unet_isbi](#)を参照
- https://docs.nvidia.com/tao/tao-toolkit/text/semantic_segmentation/unet.html
- データセット: isbi
- 事前学習モデル: pretrained_resnet50

全体図

Detectnetと同じ



TAO Tool kit

4. 事例：セグメンテーション

⑪ unet: セグメンテーション

- 概要: 医療系の画像を使って、血管などをセグメンテーションする。[Unet_isbi](#)を参照
- https://docs.nvidia.com/tao/tao-toolkit/text/semantic_segmentation/unet.html
- データセット: isbi
- 事前学習モデル: pretrained_resnet50

\$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/unet

(venv_py36) masa@ubuntu2 :\$ jupyter notebook

unet_isbi2.ipynbを選択

Jupyterの各処理は省略

入カコマンド

\$ cd

/home/masa/Documents/tao/cv_samples_v1.3.0/unet

\$ jupyter notebook

Binary Semantic Segmentation using TAO UNET

Transfer learning is the process of transferring learned features from one application to another. It is a commonly used training technique where you use a model trained on one task and re-train to use it on a different task.

Train Adapt Optimize (TAO) Toolkit is a simple and easy-to-use Python based AI toolkit for taking purpose-built AI models and customizing them with users' own data.

The diagram illustrates the TAO Toolkit architecture, showing the flow from pre-trained models and custom datasets through the TAO Toolkit (TAO UI*, TAO TOOLKIT**, Jupyter Notebooks) to a Custom AI Model. The toolkit is supported by CUDA-X (NVIDIA Containers RT, CUDA, cuDNN, TensorRT) and runs on various training and inference platforms (Workstation, DGX, Cloud, Jetson, TX, Ampere).

Learning Objectives

In this notebook, you will learn how to leverage the simplicity and convenience of TAO to:

- Take a pre-trained resnet18 model and train a ResNet-18 UNet model on the ISBI dataset
- Run Inference on the trained model and visualize the inferences

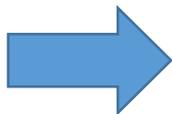
4. 事例：セグメンテーション

⑪ unet: セグメンテーション

- 概要: 医療系の画像を使って、血管などをセグメンテーションする。[Unet_isbi](#)を参照
- https://docs.nvidia.com/tao/tao-toolkit/text/semantic_segmentation/unet.html

unet_isbi2.ipynbを選択

tao当初評価結果	tao平均推論時間(sec): ①	prune後の再学習結果	量子化後の評価結果	tensorRT後の評価結果	tensorRT後の平均推論時間(sec):②	高速化倍率:①/②	tao学習、prune後再学習時間の合計(hour)
Recall : 0.8171335756778717 Precision: 0.831825852394104 F1 score: 0.8241343816116111 Mean IOU: 0.7151066660881042	Throughput Avg: 75.287 img/s Latency Avg: 48.422 ms Latency 90%: 59.762 ms Latency 95%: 61.934 ms Latency 99%: 66.18 ms	Recall : 0.8249054551124573 Precision: 0.8128768801689148 F1 score: 0.8186391778106556 Mean IOU: 0.7068162858486176			10/10 [00:00<00:00, 77961.04it/s]	1036.03	0.333333



TAO Tool kit 5.個別事例

入カコマンド

```
$ cd
```

```
/home/masa/Documents/tao/cv_samples_v1.3.0/bpnet
```

```
$ jupyter notebook
```

⑬ BodyPoseNet

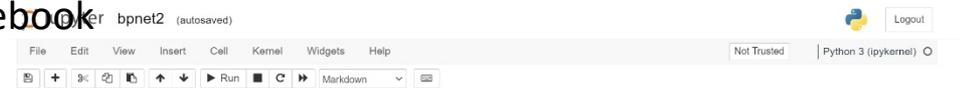
- 概要: body pose netは、からだの目、耳、手首などの位置を検出します。
- https://docs.nvidia.com/tao/tao-toolkit/text/bodypose_estimation/bodyposenet.html
- データセット: coco
- 事前学習モデル: bodyposenet_vtrainable_v1-2.0

```
$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/bpnet
```

```
(venv_py36) masa@ubuntu2 :$ jupyter notebook
```

bpnet2.ipynbを選択

Jupyterの各処理は省略,学習時間は、28時間



Bodypose Estimation using TAO BodyposeNet

Transfer learning is the process of transferring learned features from one application to another. It is a commonly used training technique where you use a model trained on one task and re-train to use it on a different task.

Train Adapt Optimize (TAO) Toolkit is a simple and easy-to-use Python based AI toolkit for taking purpose-built AI models and customizing them with users' own data.



TAO Tool kit 5.個別事例

⑰ Fpenet

- 概要: 顔の部位(Eyes, Nose, Mouth, Eyebrows, Chin, HP, Pupil, Ears,)を検出します。
- https://docs.nvidia.com/tao/tao-toolkit/text/facial_landmarks_estimation/facial_landmarks_estimation.html

データセット: AFW

事前学習モデル: fpenet:trainable_v1.0

\$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/fpenet

(venv_py36) masa@ubuntu2 :\$ jupyter notebook

fpenet2.ipynbを選択

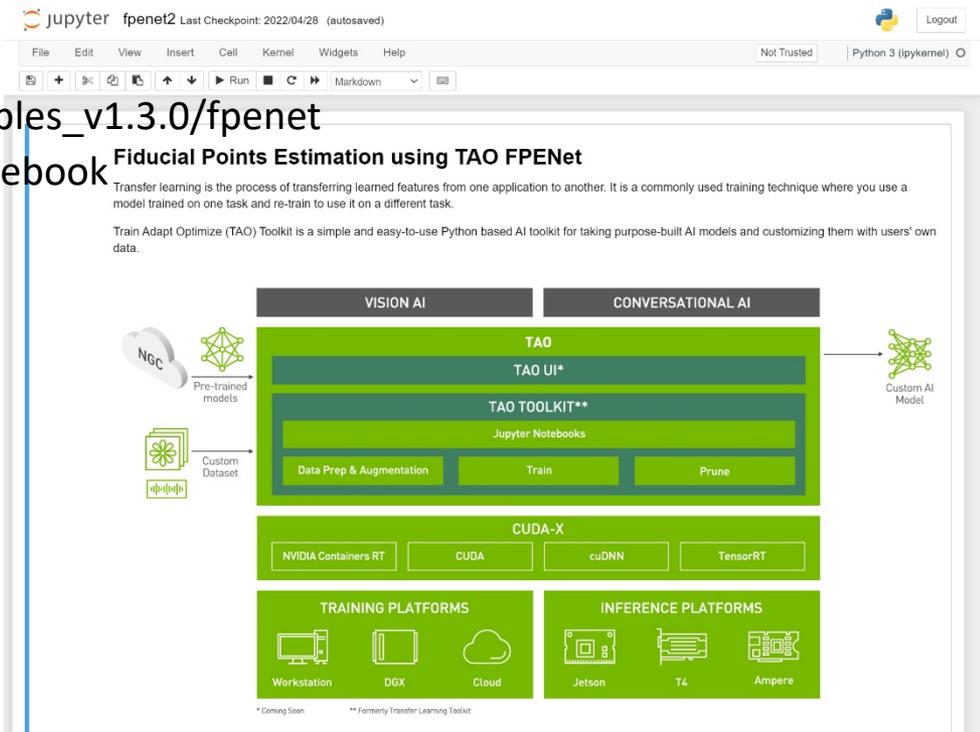
Jupyterの各処理は省略

入カコマンド

\$ cd

/home/masa/Documents/tao/cv_samples_v1.3.0/fpenet

\$ jupyter notebook



TAO Tool kit 5.個別事例

入カコマンド

```
$ cd  
/home/masa/Documents/tao/cv_samples_v1.3.0/gazenet  
$ jupyter notebook
```

⑱ gazenet

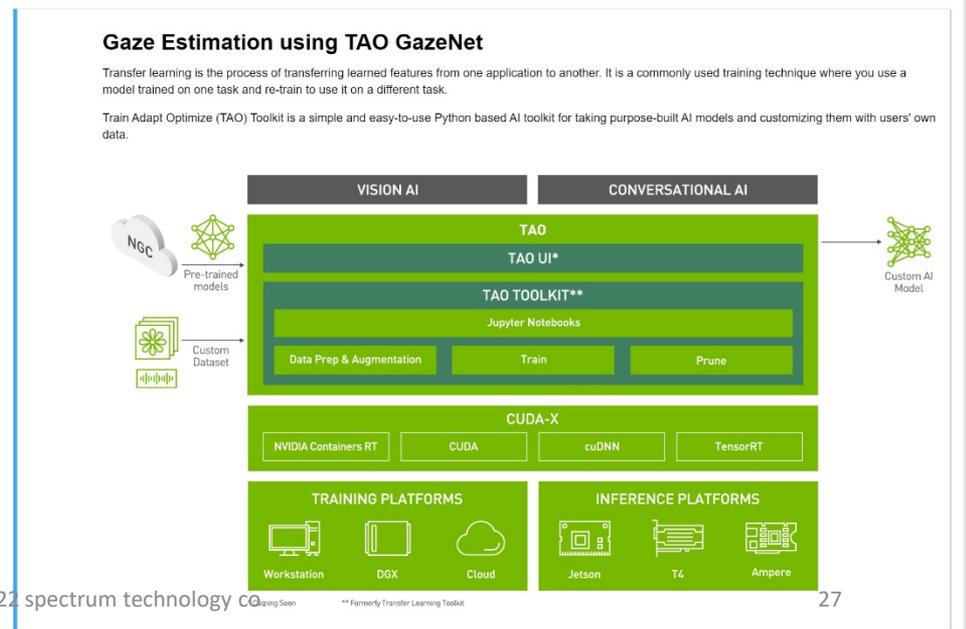
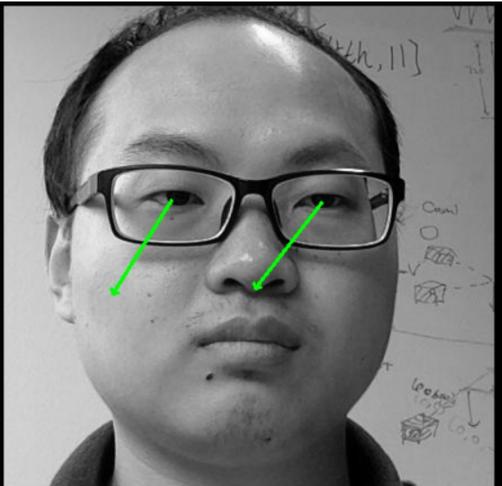
- 概要: 顔から視線を推測。
- https://docs.nvidia.com/tao/tao-toolkit/text/gaze_estimation/gaze_estimation.html
- データセット: MPIIFaceGaze
- 事前学習モデル: gazenet:trainable_v1.0

\$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/gazenet

(venv_py36) masa@ubuntu2 :\$ jupyter notebook

gazenet2.ipynbを選択

Jupyterの各処理は省略



TAO Tool kit 5.個別事例

21 lprnet

- 概要: ナンバープレートの認識。
- https://docs.nvidia.com/tao/tao-toolkit/text/character_recognition/lprnet.html
- データセット: OpenALPR benchmark
- 事前学習モデル: pretrained_lprnet_baseline18

```
$ cd /home/masa/Documents/tao/cv_samples_v1.3.0/lprnet
```

```
(venv_py36) masa@ubuntu2 :$ jupyter notebook
```

lprnet2.ipynbを選択

Jupyterの各処理は省略



読み取り結果

wts-lg-000076.jpg:4GLS168

入カコマンド

```
$ cd
```

```
/home/masa/Documents/tao/cv_samples_v1.3.0/lprnet
```

```
$ jupyter notebook
```

US版

License Plate Recognition using TAO LPRNet

Transfer learning is the process of transferring learned features from one application to another. It is a commonly used training technique where you use a model trained on one task and re-train to use it on a different task.

Train Adapt Optimize (TAO) Toolkit is a simple and easy-to-use Python based AI toolkit for taking purpose-built AI models and customizing them with users' own data.

The diagram shows the TAO architecture. It is divided into VISION AI and CONVERSATIONAL AI. The TAO layer includes TAO UI* and TAO TOOLKIT**. The TAO TOOLKIT** layer includes Jupyter Notebooks, Data Prep & Augmentation, Train, and Prune. The TAO layer is supported by CUDA-X, which includes NVIDIA Containers RT, CUDA, cuDNN, and TensorRT. The TAO layer is supported by TRAINING PLATFORMS (Workstation, DGX, Cloud) and INFERENCE PLATFORMS (Jetson, T4, Ampere).