## Set up the development environment

#### 1.1 install ANACONDA

Visit URL https://www.anaconda.com/distribution/#download-section

Select "Linux"-->"64-Bit (x86) Installer", download the installation script: Anaconda3-2020.11-Linux-x86\_64.sh (Just need to download the latest version, no specific version requirements)

Run the script to install:

$ bash Anaconda3-2020.11-Linux-x86\_64.sh

When the installation page prompts "**Do you wish the installer to initialize Anaconda3 by running conda init?**", it is recommended, select "yes".

When the installation page displays "**Thank you for installing Anaconda3!**", the installation is successful.

Reopen the terminal or execute the following command to make the installation take effect immediately:

$source ~/.bashrc

#### 1.2 Neural Processing SDK for AI Setup

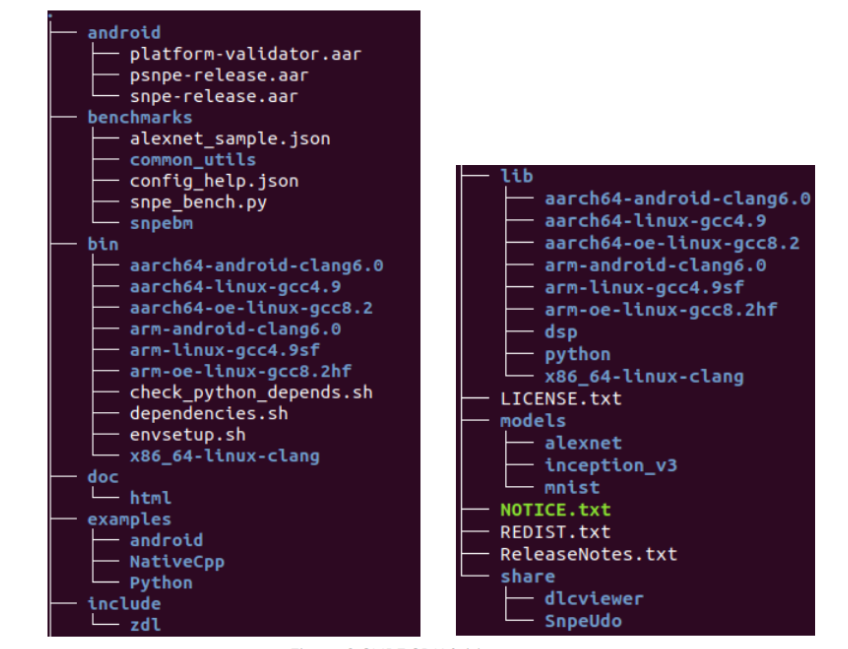
Visit Qualcomm website: (register, log in to Qualcomm account)

https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk

In the ‘Neural Processing SDK for AI -> Tools & Resources’ page,

Select the Qualcomm Neural Processing SDK for AI v1.47. version to download to the local, and unzip the file

SDK File structure:



**Configuration environment and dependencies:**

1. Run the dependencies script to check the system for Ubuntu package dependencies. It will ask to install ones that are missing. Install the missing packages.

$source snpe-X.Y.Z/bin/dependencies.sh

2. Run the python dependency checker to check the system for python package dependencies.

Install the missing packages.

$source snpe-X.Y.Z/bin/check\_python\_depends.sh

python package versions tested with the SDK are:

numpy v1.16.5

sphinx v2.2.1

scipy v1.3.1

matplotlib v3.0.3

skimage v0.15.0

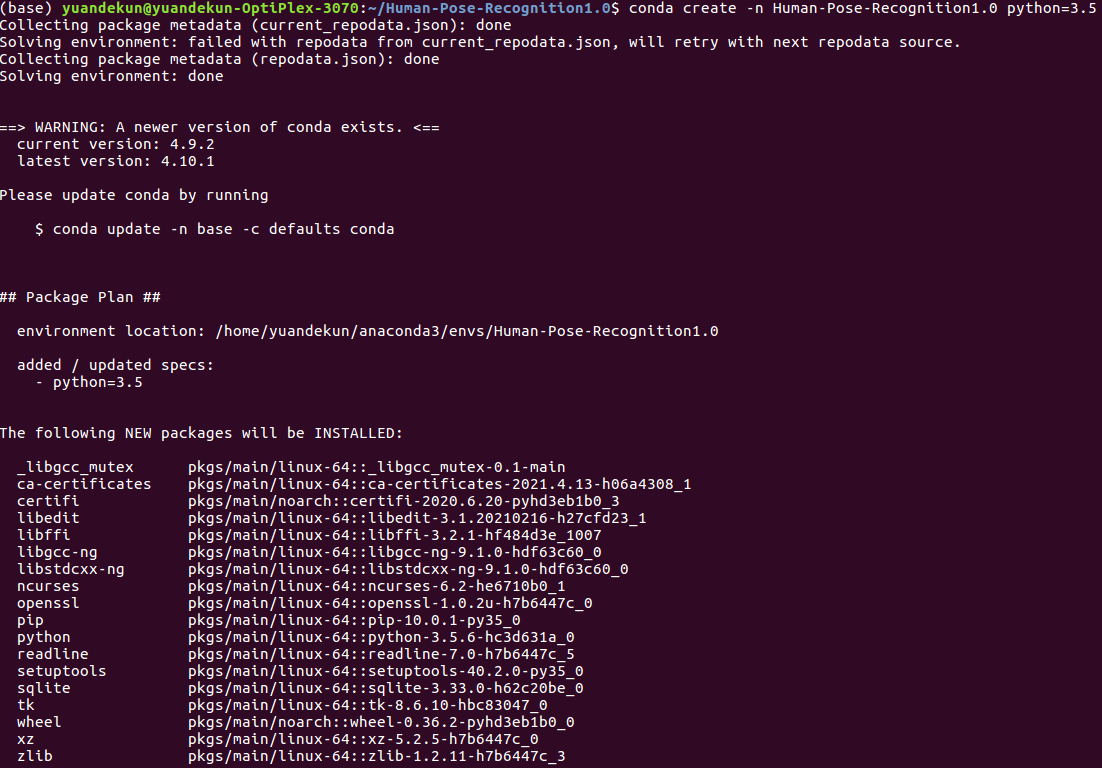
protobuf v3.6.0

pyyaml v5.1

#### 1.3 Set up Human-Pose-Recognition1.0 Development environment

1.Create a 3.5 virtual environment

$conda create -n Human-Pose-Recognition1.0 python=3.5



1. Activate the virtual environment

conda activate Human-Pose-Recognition1.0

1. Download pre-training models:

There are three types of data pre-training models, coco, mipi and body\_25, we

select the body\_25 model since it has higher inference accuracy and faster speed.

pre-training models:<http://posefs1.perception.cs.cmu.edu/OpenPose/models/pose/body_25/pose_iter_584000.caffemodel>

prototxt :<https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/models/pose/body_25/pose_deploy.prototxt>

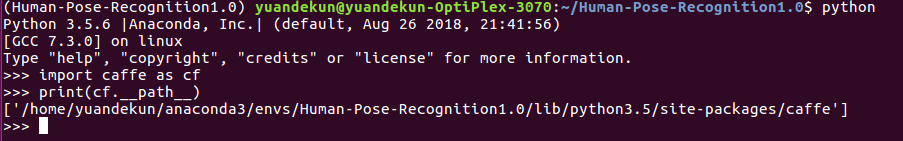
1. Install caffe

conda install caffe

1. Define $SNPE\_ROOT in .bashrc file.

export SNPE\_ROOT=/home/yuandekun/bin/snpe-1.47.0.2501

1. Configure the Caffe environment:
2. Set the $CAFFE\_DIR



CAFFE\_DIR=/home/yuandekun/anaconda3/envs/Human-Pose-Recognition1.0/lib/python3.5/site-packages/caffe

#### 2)Run script:

$ cd $SNPE\_ROOT

$ source bin/envsetup.sh -c $CAFFE\_DIR

## Show Human-Pose-Recognition1.0

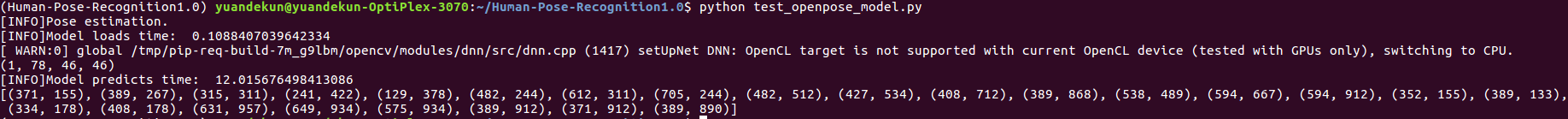
1. cd <project dir>

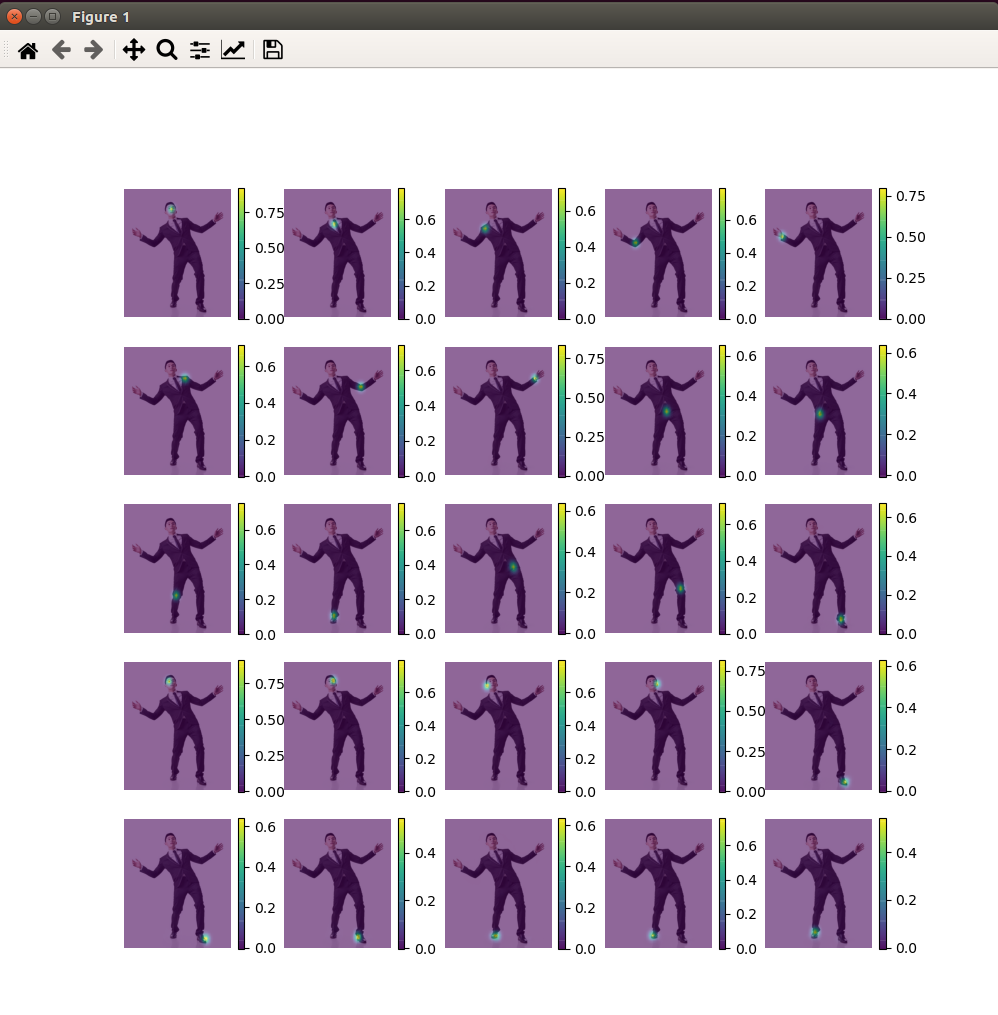


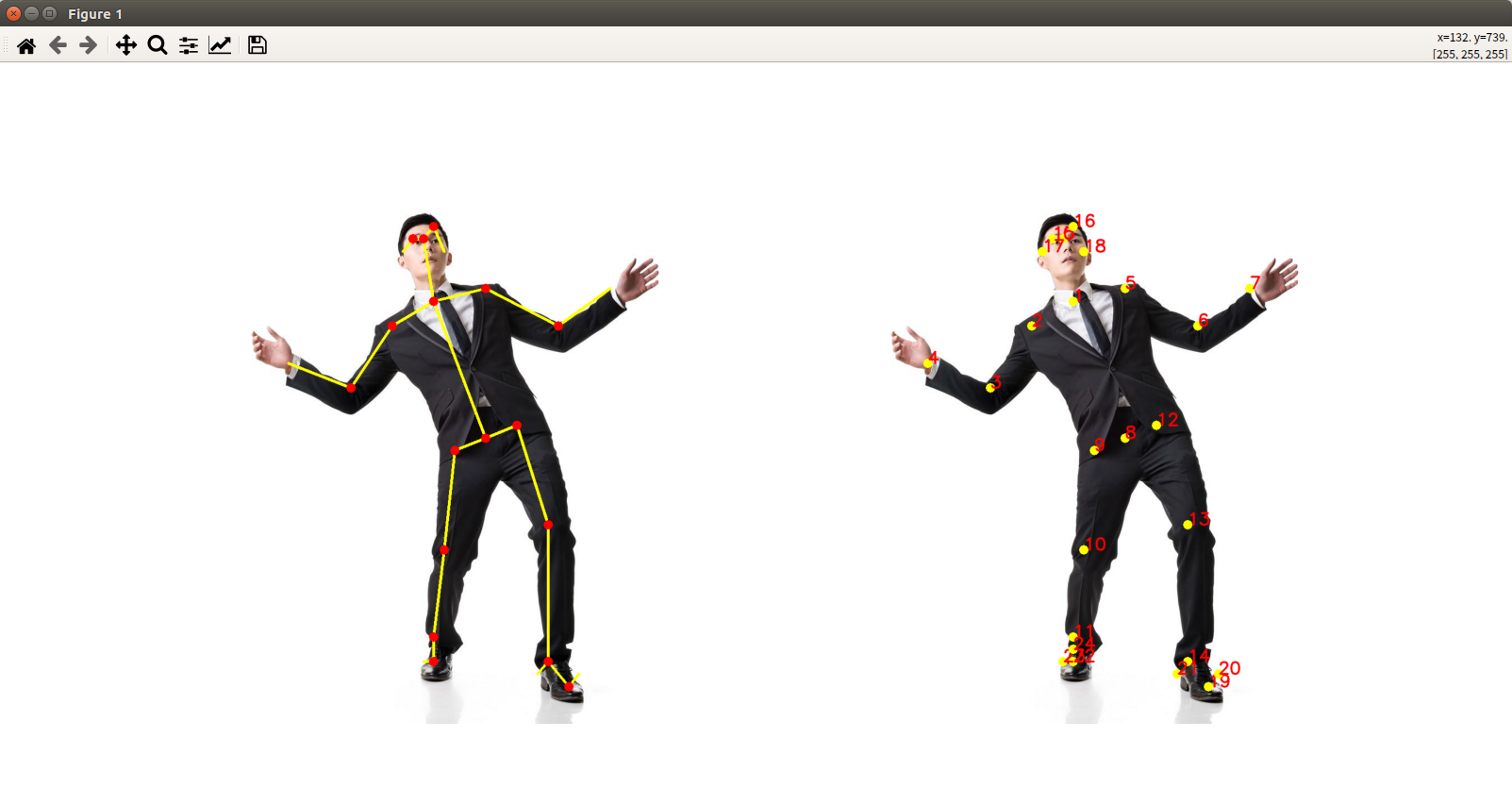
1. Test whether this model is available(caffe format)

$ python test\_openpose\_model.py

Result:







## Convert to DLC file

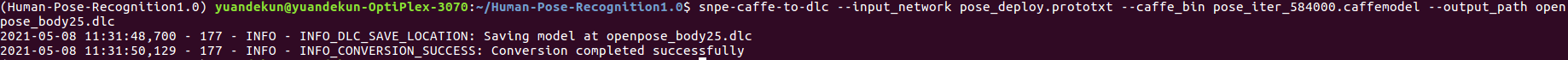
#### Convert pb to DLC

$snpe-caffe-to-dlc --input\_network pose\_deploy.prototxt --caffe\_bin pose\_iter\_584000.caffemodel --output\_path openpose\_body25.dlc

--input\_network INPUT\_NETWORK, Path to the source framework model.

--caffe\_bin CAFFE\_BIN INPUT\_NETWORK, Input caffe binary file containing the weight data.

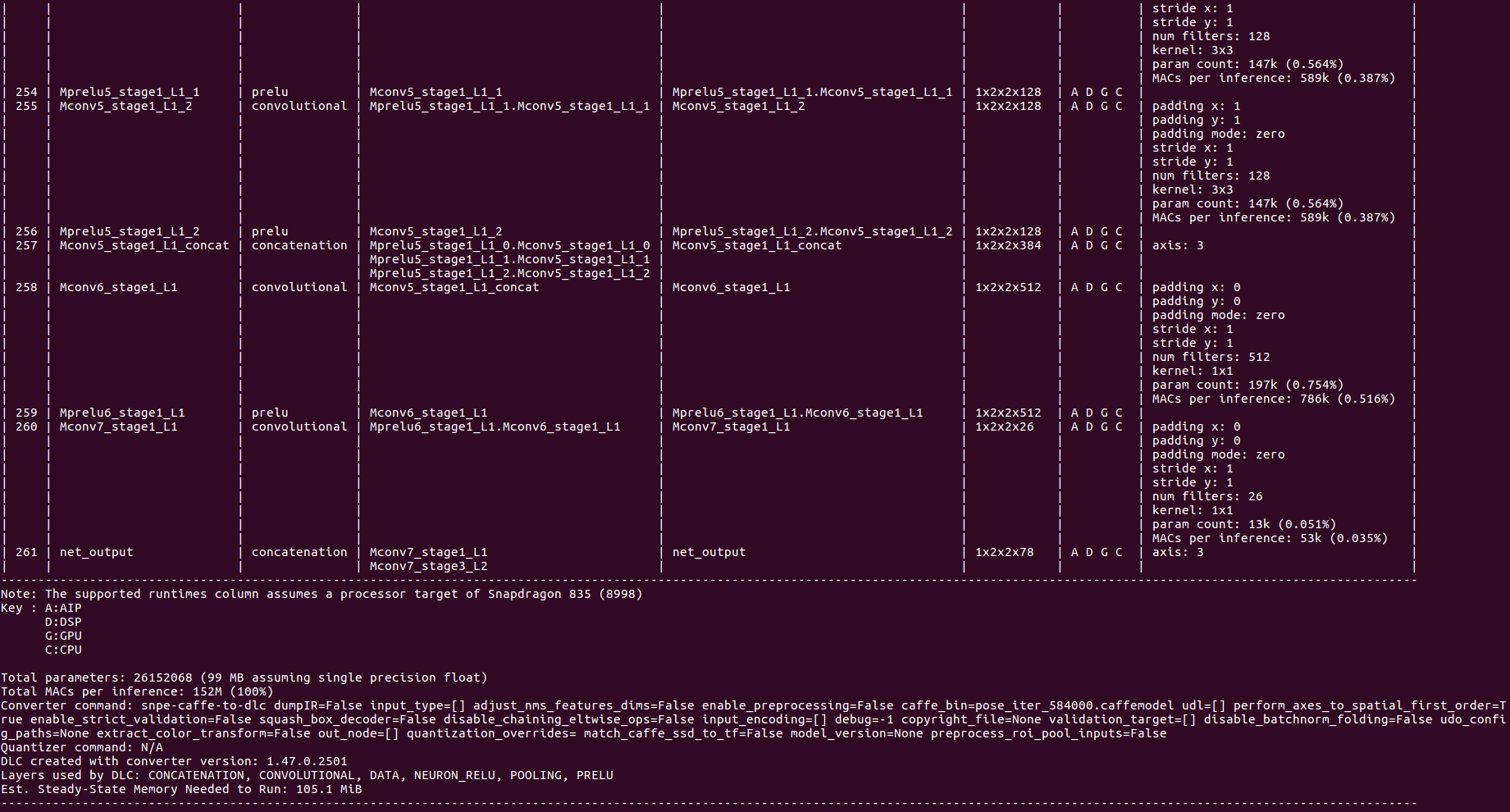
--output\_path output dlc file



#### Check DLC

1. Run snpe-dlc-info to check network layer information

$snpe-dlc-info -i openpose\_body25.dlc



1. Run snpe-dlc-viewer to check input, output layer

snpe-dlc-viewer -i openpose\_body25.dlc

