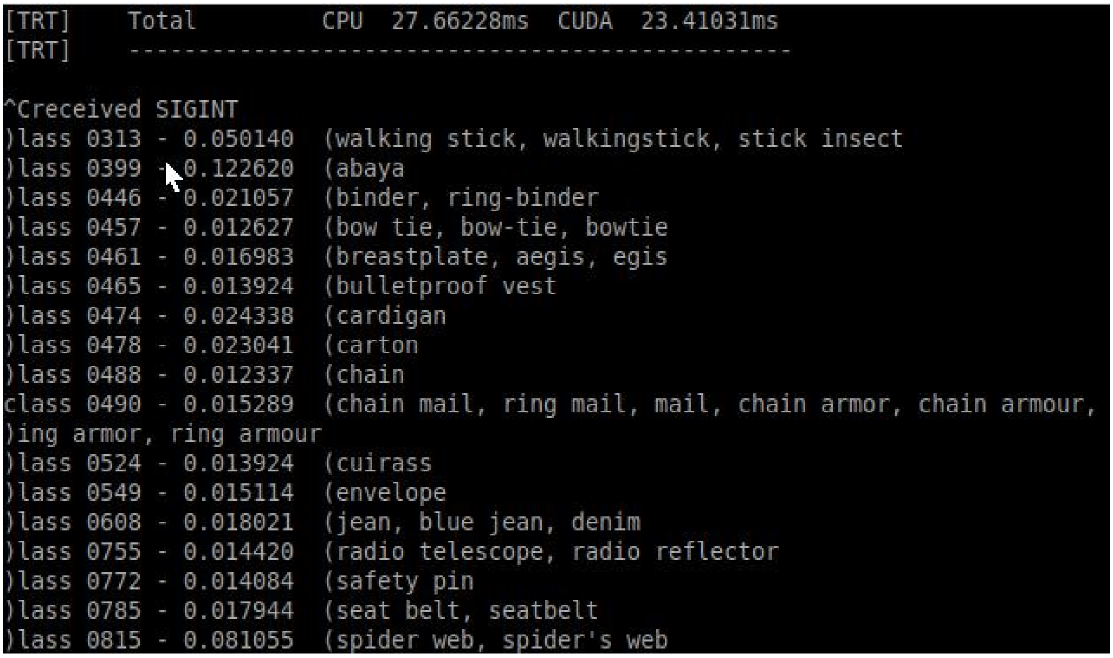
9.Using DetectNet camera Real-time detection

Run the object detection network on the real-time video source of the detectnet-camera Jetson onboard camera. Start it from the command line and the type of network you need:

```
$ ./detectnet-camera facenet          # Running using facial recognition network
$ ./detectnet-camera multiped    # Run using multi-level pedestrian/baggage detector
$ ./detectnet-camera pednet        # Run using original single-level pedestrian detector
$ ./detectnet-camera coco-bottle    # Detect bottle/soda can under the camera
$ ./detectnet-camera coco-dog                # Detecting dogs under the camera
$ ./detectnet-camera                    # By default, the program will run and use
```

Note: To get the best performance when running detectnet, increase the Jetson clock limit by running a script:

sudo ~/jetson_clocks.sh



```
[TRT]    Total         CPU  27.66228ms  CUDA  23.41031ms
[TRT]    ------------------------------------------------

^Creceived SIGINT
)lass 0313 - 0.050140  (walking stick, walkingstick, stick insect
)lass 0399 - 0.122620  (abaya
)lass 0446 - 0.021057  (binder, ring-binder
)lass 0457 - 0.012627  (bow tie, bow-tie, bowtie
)lass 0461 - 0.016983  (breastplate, aegis, egis
)lass 0465 - 0.013924  (bulletproof vest
)lass 0474 - 0.024338  (cardigan
)lass 0478 - 0.023041  (carton
)lass 0488 - 0.012337  (chain
class 0490 - 0.015289  (chain mail, ring mail, mail, chain armor, chain armour,
)ing armor, ring armour
)lass 0524 - 0.013924  (cuirass
)lass 0549 - 0.015114  (envelope
)lass 0608 - 0.018021  (jean, blue jean, denim
)lass 0755 - 0.014420  (radio telescope, radio reflector
)lass 0772 - 0.014084  (safety pin
)lass 0785 - 0.017944  (seat belt, seatbelt
)lass 0815 - 0.081055  (spider web, spider's web
```

In the above execution process, each time the first execution is performed, the update model will take a long time. You need to wait patiently, when you want to use it next time, you can use it directly.

Note: By default, Jetson's on-board CSI camera will be used as the video source.

If you want to use a USB webcam,
Similar to the previous detectnet-console example, these camera applications use detection networks, except that they process live video from the camera.
detectnet-camera accepts a variety of optional command line parameters, including:

- - - network flag, which changes the detection model in use (default is SSD-Mobilenet-v2).
- - - overlay flag, which can be a comma-separated combination of box, labels, conf, and none.
- The default value is --overlay = box, labels, conf display box, label and confidence values
- - - alpha sets the value of the alpha blending value to use when overriding (the default is 120).
- --threshold sets the value of the minimum detection threshold (default is 0.5).
- - - camera flag sets the camera device to be used
- Use MIPI CSI cameras by specifying the sensor index (0 or 1 etc.)
- V4L2   USB camera is used by specifying its /dev/video node (/dev/video0, , etc.).
- Default is to use MIPI CSI sensor 0 (--camera = 0)
- - - width and - - height flags set the camera resolution (default is 1280x720)
- Resolution should be set to a format supported by the camera.
- Query the available formats using:

```
sudo apt-get install v4l-utils
v4l2-ctl --list-formats-ext
```

You can combine these flags as needed, and there are other command line parameters available for loading custom models. Launch the application with the --help flag for more information, or see the Examples readme.

Here are some typical scenarios for start programs:

C ++

```
$ ./imagenet-camera - - network = facenet    # Use RESNET-18, default MIPI CSI camera (1280×720)
$ ./imagenet-camera - - camera = /dev /video1    # Use GoogleNet, V4L2 camera / dev/video1 (1280x720)
$ ./imagenet-camera - - width = 640 - - height = 480    # Use GoogleNet, default is MIPI CSI camera (640x480)
```

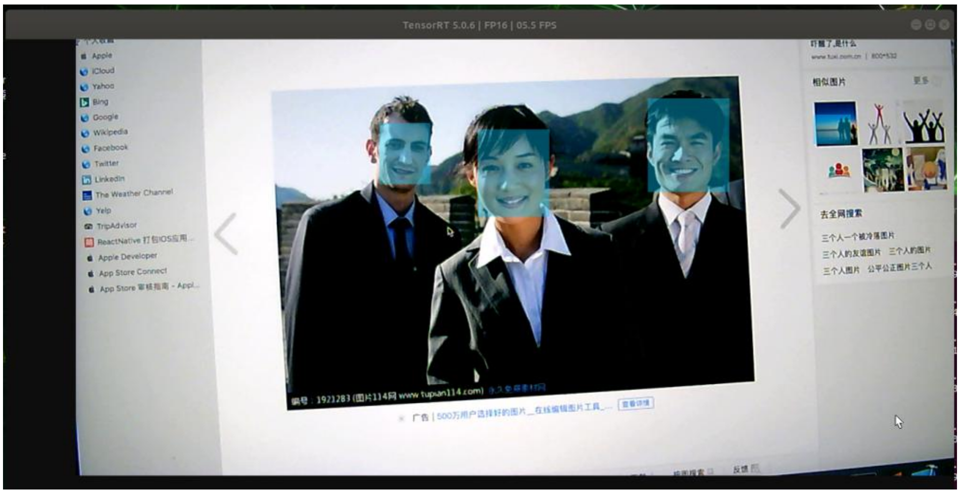
Python

```
$ python3 detectnet-camera.py --network=facenet      #Use FaceNet, defaultMIPI CSI camera (1280×720)
$ python3 detectnet-camera.py --network=facenet --camera=/dev/video    # Use RESNET-18, V4L2camera /dev/video1 (1280x720)
$ ./ imagenet-camera.py - - camera = /dev/video1     # Use GoogleNet, default is MIPI CSI camera (640x480)
```

If the desired object is not detected in the video feed, or you are getting false detections, try using the --threshold parameter to lower or increase the detection threshold (the default value is 0.5).

```
./detectnet-camera --network=facenet --threshold=0.6
```

After executing the first command,we can detect multiple faces. As shown below.


TensorRT 5.0.6 | FP16 | 05.5 FPS