User Guide :

How to enable Intel Neural Compute Stick2 in i.MX8MQ

1. Introduction

The Intel® Neural Compute Stick 2 (Intel® NCS 2) is Intel's newest deep learning inference development kit. Packed in an affordable USB-stick form factor, the Intel® NCS 2 is powered by latest VPU (vision processing unit) – the Intel® Movidius™ Myriad X, which includes an on-chip neural network accelerator called the Neural Compute Engine. With 16 SHAVE cores and a dedicated hardware neural network accelerator, the NCS 2 offers up to 8x performance improvement+ over the previous generation.
Ref: https://software.intel.com/en-us/articles/run-intel-openvino-models-on-intel-neural-compute-stick-2

The NCS 2 officially supported hardware platform is x86 PC and Raspberry Pi. In this guide, we will introduce how to implement in i.MX8MQ.

2. Download and flash i.MX8MQ Linux image

Please prepare an SD card, 16GB recommended.
Download i.MX8MQ Linux image in the link: https://www.nxp.com/support/developer-resources/software-development-tools/i.mx-developer-resources/i.mx-software-and-development-tool:IMX_SW
In i.MX BSP Updates and Release -> Linux
Linux L4.14.78 for i.MX 8MQuad GA has been tested. Other versions may work as well.
Unzip the downloaded image and flash in SD card.
You may use *dd* in Linux or win32diskimager(https://sourceforge.net/projects/win32diskimager/) in Windows to flash image.

3. Install Debian 9 rootfs and desktop

Download linaro-stretch-developer-20170626-41.tar.gz in the link:
https://releases.linaro.org/debian/images/developer-armhf/17.06/

In a Linux machine, check /dev before and after plugging in the SD card in a Linux machine,

```
$ ls /dev/sd*
/dev/sda /dev/sda1 /dev/sda1 /dev/sdb /dev/sdb1
/dev/sdb2
$ ls /dev/sd*
/dev/sda /dev/sda1 /dev/sda1 /dev/sdb /dev/sdb1
/dev/sdb2 /dev/sdc /dev/sdc1 /dev/sdc2

$ sudo umount /dev/sdc2
$ sudo mkfs.ext4 /dev/sdc2
y
$ tar -zxvf linaro-stretch-developer-20170626-41.tar.gz
$ mkdir rootfs
$ sudo mount /dev/sdc2 rootfs
$ sudo cp -a -r binary/* rootfs
$ sudo umount /dev/sdc2
```

Then Debian rootfs is successfully installed.

Boot i.MX8MQ evk board from SD card.
Install Xfce desktop:

```
$ sudo apt-get install task-xfce-desktop
```

Other desktops may work as well but not tested.
It may take some time. After successful install, reboot the system.

Change system time using:

```
$ sudo date -s “yyyy-mm-dd hour:minute:second”
```

4. Download and compile NCSDK.

There are two versions of NCSDK now, which correspond to two versions of Neural Compute Sticks. Source code can be found in: https://github.com/movidius/ncsdk.
Here we only introduce NCSDK2, the installation of NCSDK1 is similar.
Install git if you don't have it yet:

```
$ sudo apt-get install git
```

Use the following command to clone NCSDK2:

```
$ mkdir workspace
$ cd wordspace
$ git clone -b ncsdk2 https://github.com/movidius/ncsdk
```

If you have git verfication error: Server certificate verficatiob failed. Cafile:/ etc/ssl/certs/ca-certificates.crt CRLfile:none

Use this command:

```
$ git config --global http.sslverify false
```

Modify install.sh

```
$ vim ncsdk/install.sh
```

Add '--no-check-certificate' after 'wget'

As our system is Debian stretch, which is not officially supported by Intel, we need to hard code this script:

Uncomment os check in 'function check_supported_os()'

```
# check_supported_os - require install to be running on a supported OS
function check_supported_os()
{
	### Checking OS and version...
	# install package lsb-release if application lsb_release isn't installed
	RC=0
	command -v lsb_release > /dev/null || RC=$?
	[ $RC -ne 0 ] && exec_and_search_errors "$SUDO_PREFIX apt-get install -y lsb-release"
#		HARDWARE_PLATFORM=$(uname -i)
#		if [ "${HARDWARE_PLATFORM}" != "x86_64" ] ; then
#			echo -e "${RED} You are not running on a 64-bit OS version which is required.	Will exit${NC}"
#			exit 1
#		fi
	# require AVX support under Ubuntu OS
#		RC=0
#		grep -q avx /proc/cpuinfo || RC=$?
#		if [ ${RC} -ne 0 ] ; then
#			echo -e "${RED}Intel(R) Advanced Vector Extensions, Intel(R) AVX, support required but not detected.	Will exit${NC}"
#			exit 1
#		fi
#		[ "${VERBOSE}" = "yes" ] && echo "Installing on Ubuntu 16.04"
#	elif [ "${OS_DISTRO,,}" = "raspbian" ] && [ ${OS_VERSION} -ge 91 ]; then
#		[ "${VERBOSE}" = "yes" ] && echo "Installing on Raspbian Stretch"
#	elif [ "${OS_DISTRO,,}" = "raspbian" ] && [ ${OS_VERSION} -ge 80 ] && [ ${OS_VERSION} -lt 90 ]; then
#		echo -e "${RED} You are running Raspbian Jessie, which is not supported by NCSDK."
#		echo -e "Please upgrade to Raspbian Stretch and then install NCSDK."
#		echo -e "Error on line $LINENO${NC}"
#		exit 1
#	else
#		echo "Your current combination of Linux distribution and distribution version is not officially supported! Error on line $LINENO.	Will exit"
#		exit 1
#	fi
		return 0
	}
```

Also in 'function install_python_dependencies()'

```
# install_python_dependencies - install dependencies using pip2/pip3
function install_python_dependencies()
{
	# Note: If sudo is used and PIP_SYSTEM_INSTALL=yes (set in ncsdk.conf), pip packages
	#		will be installed in the systems directory, otherwise installed per user
	echo "Installing python dependencies"

#	if [ "${OS_DISTRO,,}" = "ubuntu" ] ; then
#		exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --trusted-host files.pythonhosted.org -r $DIR/requirements.txt"
		#NPS exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --trusted-host files.pythonhosted.org --upgrade numpy"
		# Install packages for python 2.x, required for NCSDK python API
#		exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --trusted-host files.pythonhosted.org Enum34>=1.1.6"
#	exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --trusted-host files.pythonhosted.org numpy==1.15"
		#NPS exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --trusted-host files.pythonhosted.org --upgrade numpy"
 # verify python3 import scipy._lib.decorator working, a potential problem on Ubuntu only.	First check python3 import scipy.
#		RC=0
#		python3 -c "import scipy" >& /dev/null || RC=$?
#		if [ ${RC} -ne 0 ] ; then
#			echo -e "${RED}Error, cannot import scipy into python3.	Error on line $LINENO.	Will exit${NC}"
#			exit 1
#		fi
#		RC=0
#		python3 -c "import scipy._lib.decorator" >& /dev/null || RC=$?
#		if [ ${RC} -ne 0 ] ; then
#			echo -e "${YELLOW}Problem importing scipy._lib.decorator into python3.	Attempting to fix${NC}"
			# Get the location of scipy to get the location of decorator.py
#			RC=0
#			SCIPY_FILE=$(python3 -c "import scipy; print(scipy.__file__)") || RC=$?
#			if [ ${RC} -eq 0 ] ; then
				# Get directory decorator.py is in from SCIPY_FILE. If decorator.py isn't a readable file, i.e. from a broken softlink, reinstall via apt
```

```
#                       [ ! -f $(dirname $SCIPY_FILE)/decorator.py ] &&
$SUDO_PREFIX apt install --reinstall python*-decorator
 #                      RC=0
 #                      python3 -c "import scipy._lib.decorator" >& /dev/null ||
RC=$?
 #                      if [ ${RC} -ne 0 ] ; then
 #                              echo -e "${RED}Error, cannot import
scipy._lib.decorator even after trying to fix this problem.    Error on line
$LINENO.    Will exit${NC}"
 #                              exit 1
 #                      else
 #                              echo "Resolved problem importing
scipy._lib.decorator into python3."
 #                              echo ""
 #                      fi
 #                  else
 #                      echo -e "${RED}Error in python3 importing scipy / printing
scipy.__file__.    Error on line $LINENO.    Will exit${NC}"
 #                      exit 1
 #                  fi
 #            fi
 #      elif [ "${OS_DISTRO,,}" = "raspbian" ] ; then
            # for Raspian, use apt with python3-* if available
            exec_and_search_errors "$SUDO_PREFIX apt-get $APT_QUIET install -
y $(cat "$DIR/requirements_apt_raspbian.txt")"
            exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --
trusted-host files.pythonhosted.org Cython"
            exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --
trusted-host files.pythonhosted.org graphviz"
            exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --
trusted-host files.pythonhosted.org scikit-image>=0.13.0,<=0.14.0"
            #NPS exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --
trusted-host files.pythonhosted.org --upgrade numpy"
            exec_and_search_errors "$PIP_PREFIX pip3 install $PIP_QUIET --
trusted-host files.pythonhosted.org pygraphviz Enum34>=1.1.6 numpy==1.15
networkx>=2.1,<=2.1"
            # Install packages for python 2.x, required for NCSDK python API
            exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --
trusted-host files.pythonhosted.org Enum34>=1.1.6"
            #NPS exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --
trusted-host files.pythonhosted.org --upgrade numpy"
        exec_and_search_errors "$PIP_PREFIX pip2 install $PIP_QUIET --trusted-
host files.pythonhosted.org numpy==1.15"
#       fi
```

```
$ sudo make install
```

With error installing tensorflow:

```
$ sudo vim ~/workspace/ncsdk/ncsdk.conf
INSTALL_TENSORFLOW=no
```

With error cannot stst '/opt/movidius/NCSDK/ncsdk-aarch64/tk': No such file or directory

```
$ cd ~/workspace/ncsdk
$ sudo tar -zxvf NCSDK-2.10.01.01.tar.gz
$ cd NCSDK-2.10.01.01
$ sudo cp -r ncsdk-arm71l ncsdk-aarch64
$ cd ..
$ mv NCSDK-2.10.01.01.tar.gz NCSDK-2.10.01.01.tar.gz.bak
$ tar -zcvf NCSDK-2.10.01.01.tar.gz NCSDK-2.10.01.01

$ cd ~/workspace/ncsdk
$ sudo make install
```

NCSDK2 will be successfully installed.

5. Install OPENVINO Toolkit

The Intel Distribution of OpenVINO Toolkit is necessary for NCSDK2.
For details, refer to https://software.intel.com/en-us/articles/OpenVINO-Install-Linux
and https://software.intel.com/en-us/articles/OpenVINO-Install-RaspberryPI

Download OpenVINO Toolkit for Raspbian from:
https://download.01.org/openvinotoolkit/2018_R5/packages/l_openvino_toolkit_ie_p_2018.5.445.tgz
Copy it to workspace.

```
$ cd ~/workspace
$ tar -xf l_openvino_toolkit_ie_p_<version>.tgz
```

Modify setupvars.sh in inference_engine_vpu_arm/bin/

```
$ sed -i "s|<INSTALLDIR>|$(pwd)/inference_engine_vpu_arm|"
inference_engine_vpu_arm/bin/setupvars.sh
$ sudo vim inference_engine_vpu_arm/bin/setupvars.sh
In line 63, change "Raspbian" to "Debian"
```

Then set the environment variables:

```
$ source inference_engine_vpu_arm/bin/setupvars.sh
$ sudo vim
inference_engine_vpu_arm/deployment_tools/inference_engine/share/Inferen
ceEngineConfig.cmake
```

Add the code in red:

```
elseif(CMAKE_SYSTEM_PROCESSOR STREQUAL "armv7l")
        set(_ARCH armv7l)
elseif(CMAKE_SYSTEM_PROCESSOR STREQUAL "aarch64")
        set(_ARCH aarch64)

elseif (${os_name} STREQUAL "Raspbian 9")
    set(_OS_PATH "raspbian_9/")
elseif (${os_name} STREQUAL "Debian 9")
    set(_OS_PATH "raspbian_9/")
```

```
$ sudo vim
inference_engine_vpu_arm/deployment_tools/inference_engine/samples/CMa
keLists.txt
```

Add the code in red:

```
if(CMAKE_SYSTEM_PROCESSOR STREQUAL "armv7l")
    set (ARCH armv7l)
if(CMAKE_SYSTEM_PROCESSOR STREQUAL "aarch64")
    set (ARCH aarch64)
```

```
$ cd
~/workspace/inference_engine_vpu_arm/inference_engine/lib/raspbian_9/
$ sudo cp -r armv7l/ aarch64
$ cd
~/workspace/inference_engine_vpu_arm/deployment_tools/inference_engine/l
ib/raspbian_9/
$ sudo cp -r armv7l/ aarch64
```

Add USB rules:

```
$ sudo usermod -a -G users "$(whoami)"
$ sh inference_engine_vpu_arm/install_dependencies/install_NCS_udev_rules
```

Build demos:

```
$ cd inference_engine_vpu_arm/deployment_tools/inference_engine/samples
$ ./build_samples.sh
```

The built demo can be found in: ~/inference_engine_samples_build/aarch64/Release/

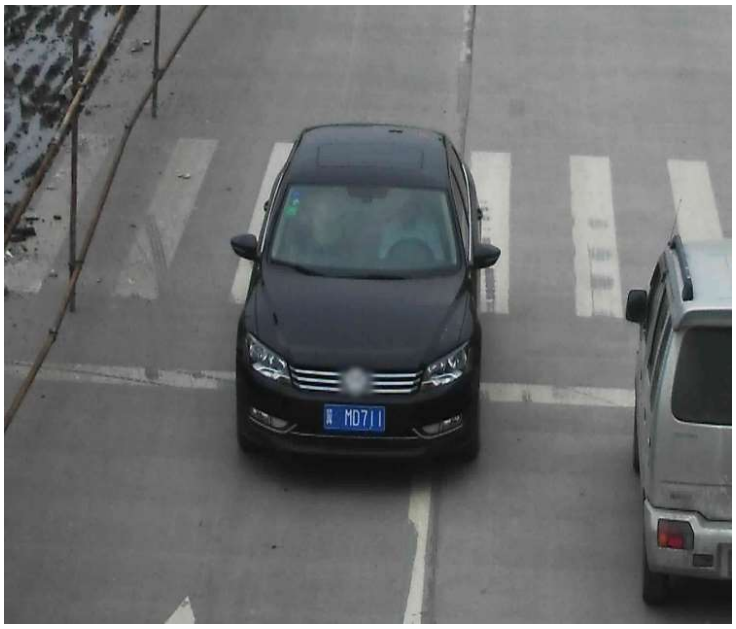
6. Run the demo

The example details can be found in https://software.intel.com/en-us/articles/OpenVINO-IE-Samples#image-classification
We will introduce some basic usages.
Car detection:

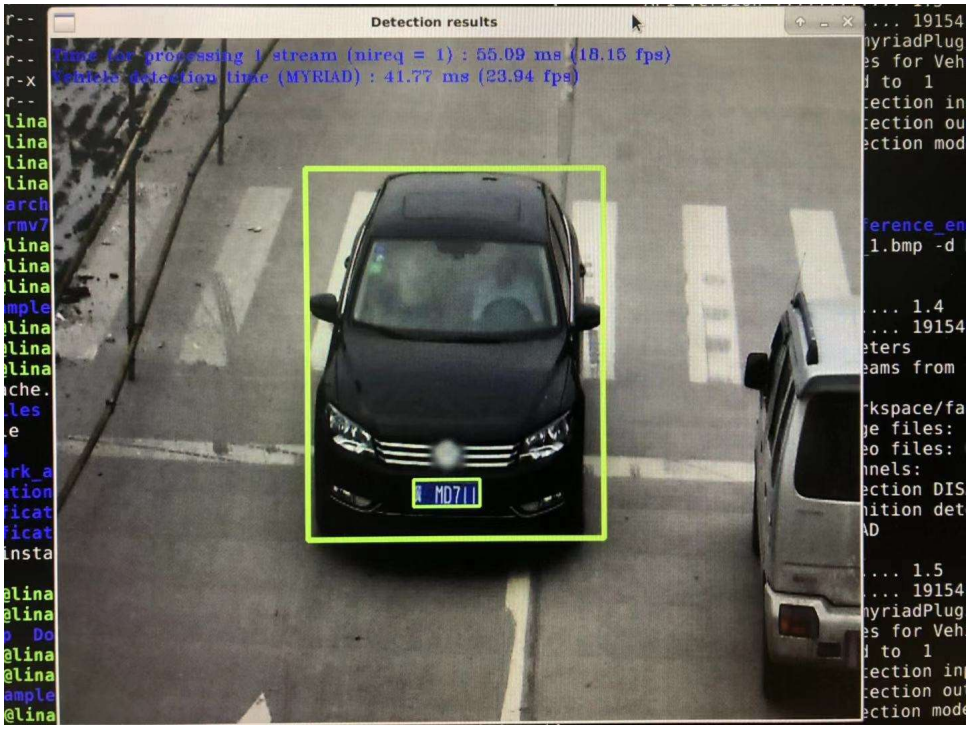
```
$ cd workspace
$ mkdir car_recog_demo
$ cd car_recognition_demo
$ wget
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/vehicle-license-plate-detection-barrier-0106/FP16/vehicle-license-plate-detection-barrier-0106.xml
$ wget
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/vehicle-license-plate-detection-barrier-0106/FP16/vehicle-license-plate-detection-barrier-0106.bin
Copy car_1.bmp to this directory
$ cd inference_engine_samples_build/aarch64/Release/
$ ./security_barrier_camera_demo -i
~/workspace/car_recognition_demo/car_1.bmp -d MYRIAD -m
~/workspace/car_recognition_demo/vehicle-license-plate-detection-barrier-0106.xml
```

You will have the following log and the recognized car:

```
[ INFO ] InferenceEngine:
        API version ............ 1.4
        Build ................... 19154
[ INFO ] Parsing input parameters
[ INFO ] Capturing video streams from the video files or loading images
[ INFO ] Files were added: 1
[ INFO ]         /home/linaro/workspace/faceexample/car_1.bmp
[ INFO ] Number of input image files: 1
[ INFO ] Number of input video files: 0
[ INFO ] Number of input channels:       1
[ INFO ] Vehicle Attribs detection DISABLED
[ INFO ] License Plate Recognition detection DISABLED
[ INFO ] Loading plugin MYRIAD

        API version ............ 1.5
        Build ................... 19154
        Description ....... myriadPlugin
[ INFO ] Loading network files for VehicleDetection
[ INFO ] Batch size is forced to    1
[ INFO ] Checking Vehicle Detection inputs
[ INFO ] Checking Vehicle Detection outputs
[ INFO ] Loading Vehicle Detection model to the MYRIAD plugin
[ INFO ] Start inference
```

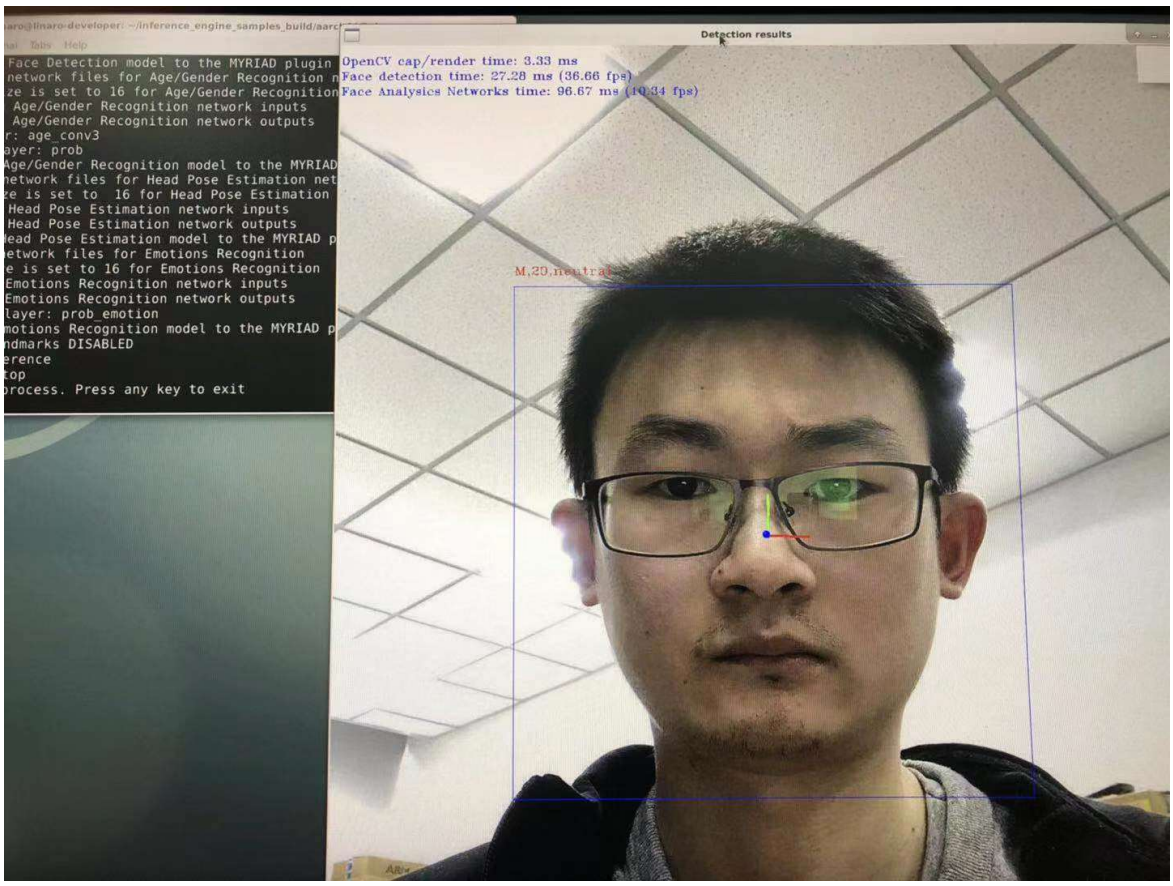


car_1.bmp


Detection results
Time for processing 1 stream (nireq = 1) : 55.09 ms (18.15 fps)
Vehicle detection time (MYRIAD) : 41.77 ms (23.94 fps)
MD711

Another example: interactive face detection

```
$ cd workspace
$ mkdir faceexample
$ cd faceexample

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/face-detection-retail-0004/FP16/face-detection-retail-0004.xml

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/face-detection-retail-0004/FP16/face-detection-retail-0004.bin

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/age-gender-recognition-retail-0013/FP16/age-gender-recognition-retail-0013.xml

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/age-gender-recognition-retail-0013/FP16/age-gender-recognition-retail-0013.bin

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/emotions-recognition-retail-0003/FP16/emotions-recognition-retail-0003.xml

$ wget --no-check-certificate
https://download.01.org/openvinotoolkit/2018_R5/open_model_zoo/emotions-recognition-retail-0003/FP16/emotions-recognition-retail-0003.bin

Put a face image into this directory.

$ cd inference_engine_samples_build/aarch64/Release/
$ ./interactive_face_detection_demo -i face.jpg -d MYRIAD -m
~/workspace/faceexample/face-detection-retail-0004.xml -d_ag MYRIAD -m_ag
~/workspace/faceexample/age-gender-recognition-retail-0013.xml -d_em
MYRIAD -m_em    ~/ workspace/faceexample/emotions-recognition-retail-0003.xml -d_hp MYRIAD -m_hp ~/ workspace/faceexample/head-pose-estimation-adas-0001.xml
```



With L4.14.78, you can use a CSI camera. Leave the parameter “-i”, camera can be opened after the command:

```
$ cd inference_engine_samples_build/aarch64/Release/
$ ./interactive_face_detection_demo    -d MYRIAD -m
~/workspace/faceexample/face-detection-retail-0004.xml -d_ag MYRIAD -m_ag
~/workspace/faceexample/age-gender-recognition-retail-0013.xml -d_em
MYRIAD -m_em    ~/ workspace/faceexample/emotions-recognition-retail-
0003.xml -d_hp MYRIAD -m_hp ~/ workspace/faceexample/head-pose-
estimation-adas-0001.xml
```

7. Performance

The performance is compared between Intel Neural Stick 2 and x86 PC:

	Intel Neural Compute Stick2	PC (Ubuntu18.04, 8 cores Intel i7@2.5GHz)
Face detection	40fps	90fps
Age analysis	40fps	50fps
Emotion analysis	32fps	50fps
Head position analysis	26fps	53fps
Age + emotion + head position	10fps	6.7fps
Human position detection	3.58fps	9.4fps