**RUNNING INTEL OPENVINO AND INFERENCES OF PRE-TRAINED MODELS ON WINDOW 10 AND UBUNTU**

**INTRODUCTION**

Intel OpenVINO™ (Open Visual Interface for Neural Network Optimization) Toolkit. It is mainly used for post-training optimization and fine-tuning of the parameters of the model.

**Prerequisites**

1. Python
2. Basic Knowledge of Machin Learning

**Intel® OpenVINO™**

 The OpenVINO™ Toolkit’s name comes from “Open Visual Inferencing and Neural Network Optimization”. It is largely focused around optimizing neural network inference and is open source.

It is developed by Intel® and supports quick inference through Intel® CPUs, GPUs, FPGAs, and a common API. OpenVINO™ may use its Model Optimizer to optimize inference models built with multiple different frames, such as TensorFlow or Caffe. Then, with this streamlined configuration, the Inference Engine may be used to speed up inference on the relevant hardware. There is already a broad variety of pre-trained models already deployed with Platform Optimizer.

**OpenVINO™ Toolkit**

* Enables deep learning inference on CNN-based edge devices
* Supports heterogeneous execution across an Intel® CPU (Central Processing Unit), Intel® Integrated Graphics Processing Unit (IGPU), Intel® FPGA (Field Programmable Gate Array), Intel® Neural Compute Stick 2 (NCS2) and Intel® Vision Accelerator Design with Intel® Movidius™ VPUs
* With the use of an easy-to-use library of computer vision functions and preoptimized kernels, it speeds time-to-market
* Includes optimized calls for computer vision standards, including OpenCV\* and OpenCL™

**System Requirements**

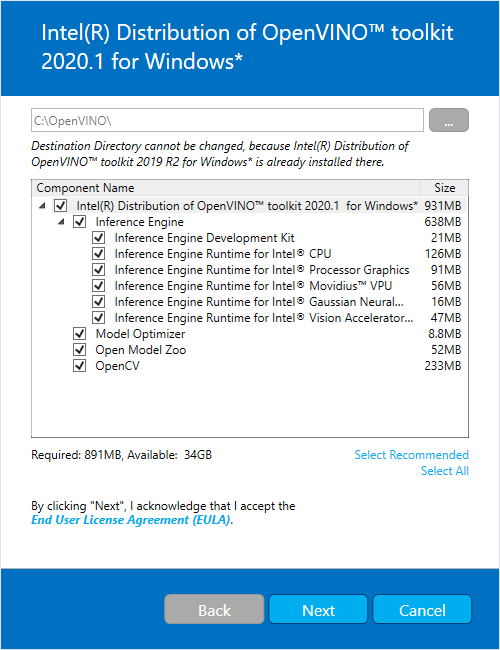
**Hardware**

* 6th to 10th generation Intel® Core™ processors and Intel® Xeon® processors
* Intel® Xeon® processor E family (formerly code named Sandy Bridge, Ivy Bridge, Haswell, and Broadwell)
* 3rd generation Intel® Xeon® Scalable processor (formerly code named Cooper Lake)
* Intel® Xeon® Scalable processor (formerly Skylake and Cascade Lake)
* Intel Atom® processor with support for Intel® Streaming SIMD Extensions 4.1 (Intel® SSE4.1)
* Intel Pentium® processor N4200/5, N3350/5, or N3450/5 with Intel® HD Graphics
* Intel® Neural Compute Stick 2
* Intel® Vision Accelerator Design with Intel® Movidius™ VPUs

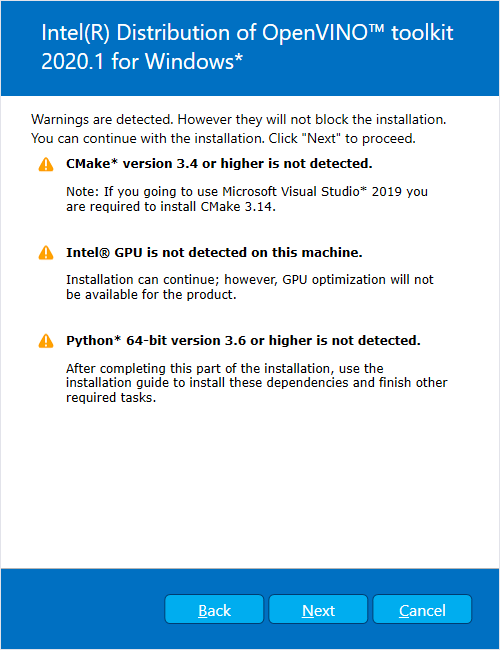
## Installation Steps

### Install the Intel® Distribution of OpenVINO™ toolkit Core Components

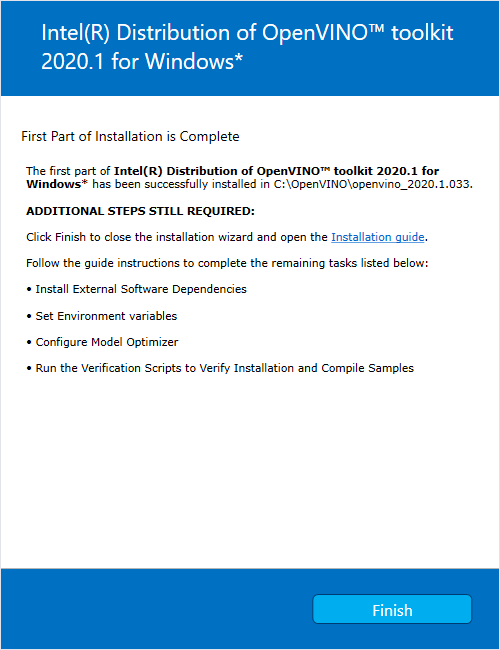
1. If you have not downloaded the Intel® Distribution of OpenVINO™ toolkit, [download the latest version](http://software.intel.com/en-us/openvino-toolkit/choose-download/free-download-windows). By default, the file is saved to the Downloads directory as w\_openvino\_toolkit\_p\_<version>.exe.
2. Go to the Downloads folder and double-click w\_openvino\_toolkit\_p\_<version>.exe. A window opens to let you choose your installation directory and components. The default installation directory is C:\Program Files (x86)\IntelSWTools\openvino\_<version>, for simplicity, a shortcut to the latest installation is also created: C:\Program Files (x86)\IntelSWTools\openvino. If you choose a different installation directory, the installer will create the directory for you:



1. Click **Next**.
2. You are asked if you want to provide consent to gather information. Choose the option of your choice. Click **Next**.
3. If you are missing external dependencies, you will see a warning screen. Write down the dependencies you are missing. **You need to take no other action at this time**. After installing the Intel® Distribution of OpenVINO™ toolkit core components, install the missing dependencies. The screen example below indicates you are missing two dependencies:



1. Click **Next**.
2. When the first part of installation is complete, the final screen informs you that the core components have been installed and additional steps still required:



1. Click **Finish** to close the installation wizard. A new browser window opens to the next section of the installation guide to set the environment variables. You are in the same document. The new window opens in case you ran the installation without first opening this installation guide.
2. If the installation indicated you must install dependencies, install them first. If there are no missing dependencies, you can go ahead and [set the environment variables](https://docs.openvinotoolkit.org/latest/openvino_docs_install_guides_installing_openvino_windows.html#set-the-environment-variables).

### Set the Environment Variables

You must update several environment variables before you can compile and run OpenVINO™ applications. Open the Command Prompt, and run the setupvars.bat batch file to temporarily set your environment variables:

cd C:\Program Files (x86)\IntelSWTools\openvino\bin\

setupvars.bat

## Configure the Model Optimizer

The Model Optimizer is a key component of the Intel® Distribution of OpenVINO™ toolkit. You cannot do inference on your trained model without running the model through the Model Optimizer. When you run a pre-trained model through the Model Optimizer, your output is an Intermediate Representation (IR) of the network. The IR is a pair of files that describe the whole model:

* .xml: Describes the network topology
* .bin: Contains the weights and biases binary data

The Inference Engine reads, loads, and infers the IR files, using a common API across the CPU, GPU, or VPU hardware.

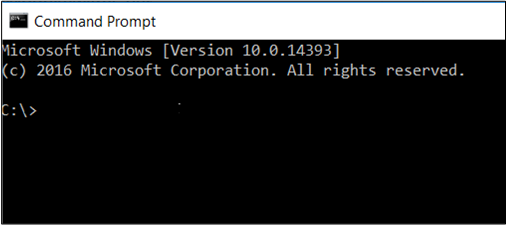
The Model Optimizer is a Python\*-based command line tool (mo.py), which is located in C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\model\_optimizer. Use this tool on models trained with popular deep learning frameworks such as Caffe\*, TensorFlow\*, MXNet\*, and ONNX\* to convert them to an optimized IR format that the Inference Engine can use.

### Model Optimizer Configuration Steps

You can configure the Model Optimizer either for all supported frameworks at once or for one framework at a time.

#### Configure the Model Optimizer for all supported frameworks at the same time:

1. Open a command prompt. To do so, type cmd in your **Search Windows** box and then press **Enter**. Type commands in the opened window:



1. Go to the Model Optimizer prerequisites directory.

cd C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\model\_optimizer\install\_prerequisites

1. Run the following batch file to configure the Model Optimizer for Caffe\*, TensorFlow\*, MXNet\*, Kaldi\*, and ONNX\*:

install\_prerequisites.bat

#### Configure the Model Optimizer for each framework separately:

1. Go to the Model Optimizer prerequisites directory:

cd C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\model\_optimizer\install\_prerequisites

1. Run the batch file for the framework you will use with the Model Optimizer. You can use more than one:
   * For **Caffe**:

install\_prerequisites\_caffe.bat

* + For **TensorFlow**:

install\_prerequisites\_tf.bat

* + For **MXNet**:

install\_prerequisites\_mxnet.bat

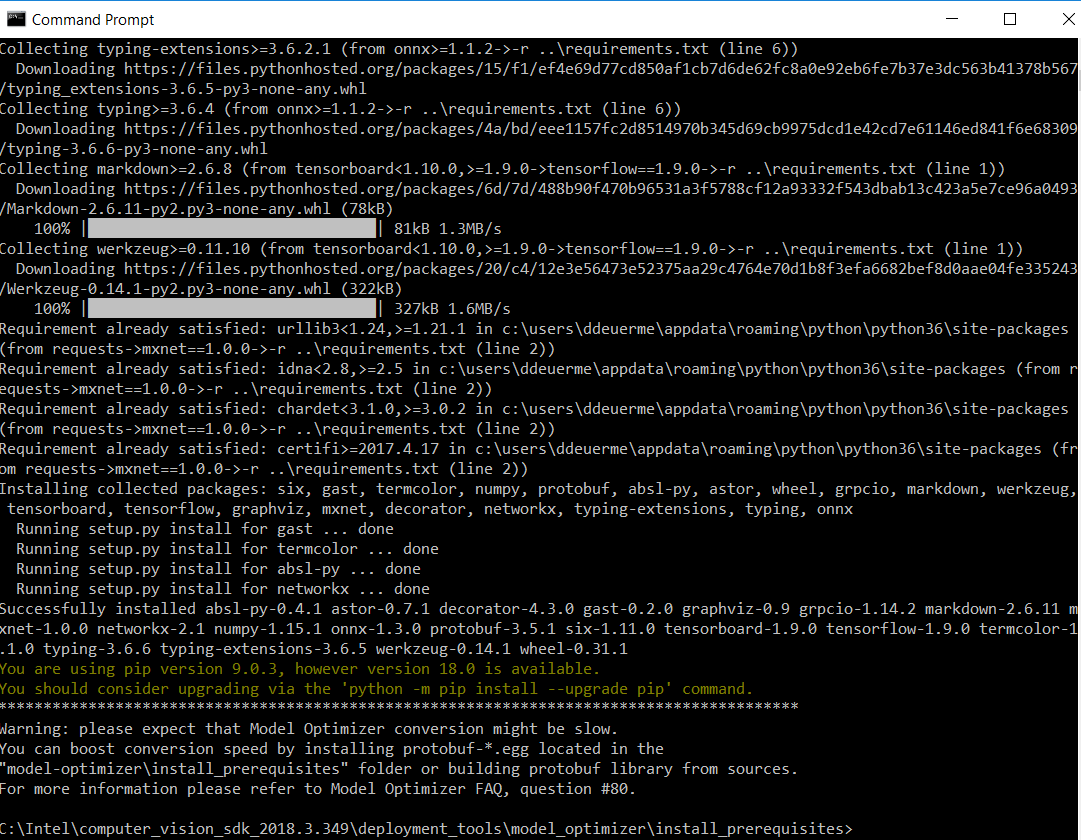
* + For **ONNX**:

install\_prerequisites\_onnx.bat

* + For **Kaldi**:

install\_prerequisites\_kaldi.bat

The Model Optimizer is configured for one or more frameworks. Success is indicated by a screen similar to this:



You are ready to use two short demos to see the results of running the Intel Distribution of OpenVINO toolkit and to verify your installation was successful.

### Run the Inference Pipeline Verification Script

To run the script, start the demo\_security\_barrier\_camera.bat file while still in the console:

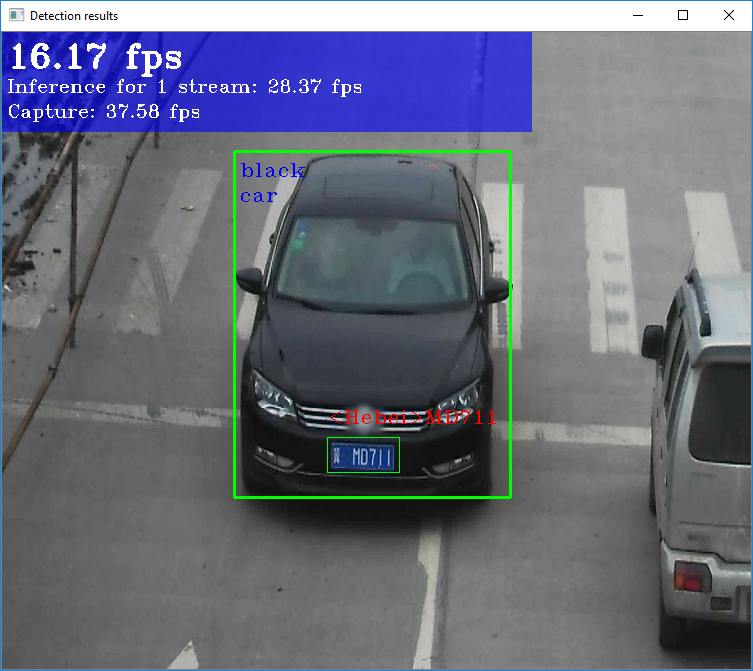
demo\_security\_barrier\_camera.bat

This script downloads three pre-trained model IRs, builds the [Security Barrier Camera Demo](https://docs.openvinotoolkit.org/latest/omz_demos_security_barrier_camera_demo_README.html) application, and runs it with the downloaded models and the car\_1.bmp image from the demo directory to show an inference pipeline. The verification script uses vehicle recognition in which vehicle attributes build on each other to narrow in on a specific attribute.

First, an object is identified as a vehicle. This identification is used as input to the next model, which identifies specific vehicle attributes, including the license plate. Finally, the attributes identified as the license plate are used as input to the third model, which recognizes specific characters in the license plate.

When the demo completes, you have two windows open:

* A console window that displays information about the tasks performed by the demo
* An image viewer window that displays a resulting frame with detections rendered as bounding boxes, similar to the following:



Close the image viewer window to end the demo.

In this section, you saw a preview of the Intel® Distribution of OpenVINO™ toolkit capabilities.

To perform inference on Intel® Vision Accelerator Design with Intel® Movidius™ VPUs, the following additional installation steps are required:

1. If your Intel® Vision Accelerator Design with Intel® Movidius™ VPUs card requires SMBUS connection to PCIe slot (Raw video data card with HW version Fab-B and before), install the SMBUS driver:
   1. Go to the <INSTALL\_DIR>\deployment\_tools\inference-engine\external\hddl\SMBusDriver directory, where <INSTALL\_DIR> is the directory in which the Intel Distribution of OpenVINO toolkit is installed.
   2. Right click on the hddlsmbus.inf file and choose **Install** from the pop up menu.
2. Download and install [Visual C++ Redistributable for Visual Studio 2017](https://www.microsoft.com/en-us/download/details.aspx?id=48145)

You are done installing your device driver and are ready to use your Intel® Vision Accelerator Design with Intel® Movidius™ VPUs.

After configuration is done, you are ready to run the verification scripts with the HDDL Plugin for your Intel® Vision Accelerator Design with Intel® Movidius™ VPUs.

1. Open a command prompt window.
2. Go to the Inference Engine demo directory:

cd C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\demo\

1. Run the Image Classification verification script. If you have access to the Internet through the proxy server only, please make sure that it is configured in your environment.

demo\_squeezenet\_download\_convert\_run.bat -d HDDL

1. Run the Inference Pipeline verification script:

demo\_security\_barrier\_camera.bat -d HDDL.

## Run the Image Classification Sample Application

**IMPORTANT**: This section requires that you have [Run the Verification Scripts to Verify Installation](https://docs.openvinotoolkit.org/latest/openvino_docs_install_guides_installing_openvino_windows.html#run-the-demos). This script builds the Image Classification sample application and downloads and converts the required Caffe\* Squeezenet model to an IR.

In this section you will run the Image Classification sample application, with the Caffe\* Squeezenet1.1 model on three types of Intel® hardware: CPU, GPU and VPUs.

Image Classification sample application binary file was automatically built and the FP16 model IR files are created when you [Ran the Image Classification Verification Script](https://docs.openvinotoolkit.org/latest/openvino_docs_install_guides_installing_openvino_windows.html#run-the-image-classification-verification-script).

The Image Classification sample application binary file located in the C:\Users\<username>\Documents\Intel\OpenVINO\inference\_engine\_samples\_build\intel64\Release\ directory. The Caffe\* Squeezenet model IR files (.bin and .xml) are located in the in the C:\Users\<username>\Documents\Intel\OpenVINO\openvino\_models\ir\public\squeezenet1.1\FP16\ directory.

**NOTE**: If you installed the Intel® Distribution of OpenVINO™ toolkit to the non-default installation directory, replace C:\Program Files (x86)\IntelSWTools with the directory where you installed the software.

To run the sample application:

1. Set up environment variables:

cd C:\Program Files (x86)\IntelSWTools\openvino\bin\setupvars.bat

1. Go to the samples build directory:

cd C:\Users<username>\Documents\Intel\OpenVINO\inference\_engine\_samples\_build\intel64\Release

1. Run the sample executable with specifying the car.png file from the demo directory as an input image, the IR of your FP16 model and a plugin for a hardware device to perform inference on.

**NOTE**: Running the sample application on hardware other than CPU requires performing [additional hardware configuration steps](https://docs.openvinotoolkit.org/latest/openvino_docs_install_guides_installing_openvino_windows.html#optional-steps).

* + For CPU:

classification\_sample\_async.exe -i "C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\demo\car.png" -m "C:\Users<username>\Documents\Intel\OpenVINO\openvino\_models\ir\public\squeezenet1.1\FP16\squeezenet1.1.xml" -d CPU

* + For GPU:

classification\_sample\_async.exe -i "C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\demo\car.png" -m "C:\Users<username>\Documents\Intel\OpenVINO\openvino\_models\ir\public\squeezenet1.1\FP16\squeezenet1.1.xml" -d GPU

* + For VPU (Intel® Neural Compute Stick 2):

classification\_sample\_async.exe -i "C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\demo\car.png" -m "C:\Users<username>\Documents\Intel\OpenVINO\openvino\_models\ir\public\squeezenet1.1\FP16\squeezenet1.1.xml" -d MYRIAD

* + For VPU (Intel® Vision Accelerator Design with Intel® Movidius™ VPUs):

classification\_sample\_async.exe -i "C:\Program Files (x86)\IntelSWTools\openvino\deployment\_tools\demo\car.png" -m "C:\Users<username>\Documents\Intel\OpenVINO\openvino\_models\ir\public\squeezenet1.1\FP16\squeezenet1.1.xml" -d HDDL

For information on Sample Applications, see the [Inference Engine Samples Overview](https://docs.openvinotoolkit.org/latest/_docs_IE_DG_Samples_Overview.html).

Congratulations, you have finished the installation of the Intel® Distribution of **OpenVINO™ toolkit for Windows\*.**

**IN UBUNTU**

To install openVINO software **First Install OpenCV and CMake**

1- Before installing any software in Linux be sure to update your repositories with

$ sudo apt update && upgrade

2- Install the required dependencies:

$ sudo apt install build-essential cmake git pkg-config libgtk-3-dev \ libavcodec-dev libavformat-dev libswscale-dev libv4l-dev \ libxvidcore-dev libx264-dev libjpeg-dev libpng-dev libtiff-dev \ gfortran openexr libatlas-base-dev python3-dev python-numpy \ libtbb2 libtbb-dev libdc1394–22-dev

3- Create an opencv\_build directory, cd into it and then clone the OpenCV and OpenCV contrib repositories:

$ mkdir ~/opencv\_build && cd ~/opencv\_build

$ git clone <https://github.com/opencv/opencv.git>

$ git clone <https://github.com/opencv/opencv_contrib.git>

4- Once the download is complete, create a build directory and cd into it:

$ cd ~/opencv\_build/opencv

$ mkdir build && cd build

5- Now setup the build with CMake:

$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE \ -D CMAKE\_INSTALL\_PREFIX=/usr/local \ -D INSTALL\_C\_EXAMPLES=ON \ -D INSTALL\_PYTHON\_EXAMPLES=ON \ -D OPENCV\_GENERATE\_PKGCONFIG=ON \ -D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_build/opencv\_contrib/modules \ -D BUILD\_EXAMPLES=ON ..

6- Start compiling with the make and then install, then verify if it was installed correctly:

$ make -j(number of cores your CPU has), e.g. make -j2 or make -j4

$ sudo make install

$ python3 -c "import cv2; print(cv2.\_\_version\_\_)

# Now to install OpenVINO

To install OpenVINO

I will summarize the steps here:

1- Make sure to download OpenVINO from the link I provided.

2- Change directories to your downloads directory with $ cd ~/Downloads

3- The file is saved in the directory as l\_openvino\_toolkit\_p\_<version>.tgz

4- Untar the file using the following command:

$ tar -xvf l\_openvino\_toolkit\_p\_<version>.tgz

5- Then move into the l\_openvino\_toolkit\_p\_<version> directory:

$ cd l\_openvino\_toolkit\_p\_<version>

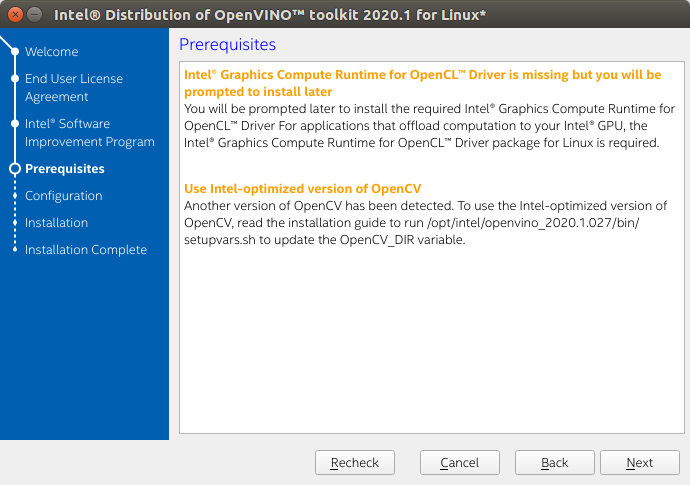
6- Now, you can install either using a GUI or CLI Installation using the following commands:

$ sudo ./install.sh

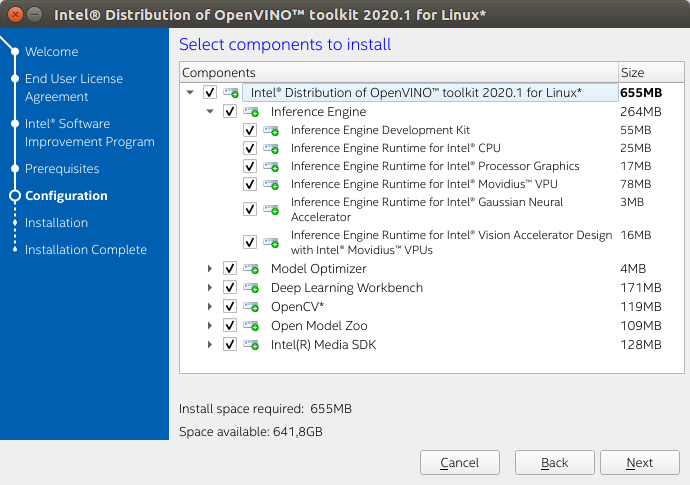
or

$ sudo ./install\_GUI.sh

7- Follow the onscreen instructions and it will tell you what needs to be installed or what you will be prompted to install later.



8- You can customize or choose a default installation. Choose default if you’re new to this.



9- Openvino will be installed on the /opt/intel/openvino\_<version>

directory

10- The first part of the installation will be complete, next you have to set up the variables, configure the model optimizer and add dependencies.

11- Change to the install\_dependencies directory with $ cd /opt/intel/openvino/install\_dependencies

12- Run the script to download the needed dependencies with the following:

$ sudo -E ./install\_openvino\_dependencies.sh

13- Next set your variables with the

$ source /opt/intel/openvino/bin/setupvars.sh command.

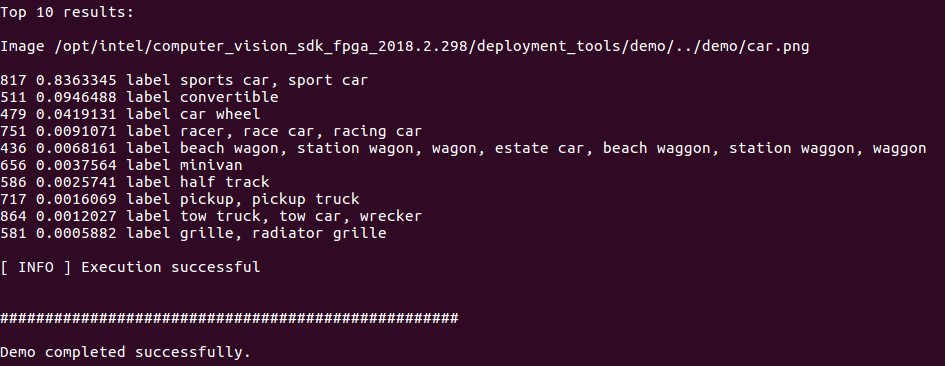
Optionally you can add it to your .bashrc file in the last line. Use nano or vi to edit the file. Just add that line at the end and source it.

14- Next, configure the model optimizer by changing to the prerequisites directory with $ cd /opt/intel/openvino/deployment\_tools/model\_optimizer/install\_prerequisites to start the process. Then run the script to configure the model optimizer with **$ sudo ./install\_prerequisites.sh**. If you want to install only a specified model use the **$ sudo ./install\_prequisites\_<model name>.sh** command instead.

15- To verify installation, go to the Inference Engine demo directory with $ cd /opt/intel/openvino/deployment\_tools/demo.

16- Then run the image classification verification script with $ ./demo\_squeezenet\_download\_convert\_run.sh.

17- If it says execution successful then it has run correctly.



18- Next, run the Inference Pipeline verification script with $ ./demo\_security\_barrier\_camera.sh. When it completes you should get an image that displays the resulting frame with detections rendered as bounding boxes and text. Your framerate will vary depending on the machine you have. Close the window to verify the installation.

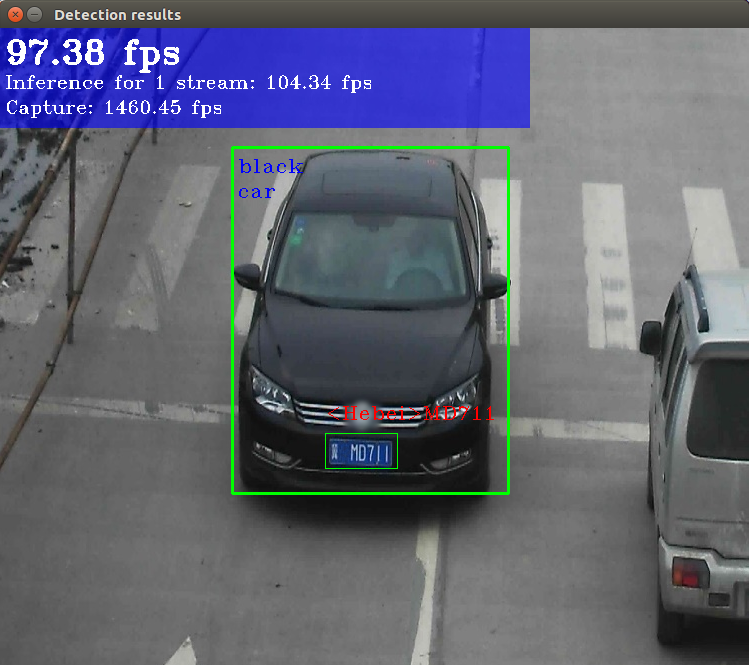


Image source = <https://docs.openvinotoolkit.org/latest/inference_pipeline_script_lnx.png>

18- Now OpenVINO installation completed.

**REFERENCES**

https://www.c-sharpcorner.com/article/intel-openvino/

https://docs.openvinotoolkit.org/latest/openvino\_docs\_install\_guides\_installing\_openvino\_windows.html.