

OpenVINO™ 工具套件

[入门指南](#)

[操作步骤](#)

[指南](#)

[资源](#)

[性能信息](#)

[API 参考](#)

[资源概述](#)

[示例](#)

[图像分类 C++ 示例异步](#)

[图像分类 Python* 示例异步](#)

[Hello 分类 C++ 示例](#)

[图像分类 Python* 示例](#)

[Hello Reshape SSD C++ 示例](#)

[Hello NV12 输入分类 C++ 示例](#)

[Hello 查询设备 C++ 示例](#)

[Hello 查询设备 Python* 示例](#)

[对象检测 C++ 示例 SSD](#)

[对象检测 Python* 示例 SSD](#)

[自动语音识别 C++ 示例](#)

[神经风格迁移 C++ 示例](#)

[神经风格迁移 Python* 示例](#)

[性能指标评测 C++ 应用](#)

[性能指标评测 Python* 应用](#)

[演示](#)

[十字路口摄像头演示](#)

[交互式面部检测 C++ 演示](#)

[交互式人脸识别 Python* 演示](#)

[TensorFlow* 对象检测 Mask R-CNN 分割演示](#)

[多通道演示](#)

多渠道面部检测 C++ 演示

多通道人体姿态估计 C++ 演示

多通道对象检测 YOLO* V3 C++ 演示

使用 Faster R-CNN 的对象检测演示

面向 CenterNet Python* 的对象检测演示

对象检测 SSD C++ 演示，异步 API 性能展示

对象检测 SSD Python* 演示，异步 API 性能展示

安全障碍摄像头演示

图像分割 C++ 演示

单个人体姿态估计 Python* 演示

图像分割 Python* 演示

智能课堂演示

文本检测演示

文本辨认 Python* 演示

视线预估演示

人体姿态估计演示

3D 人体姿态估计 Python* 演示

行人追踪器演示

超高分辨率演示

对象检测 YOLO* V3 C++ 演示，异步 API 性能展示

对象检测 YOLO* V3 Python* 演示，异步 API 性能展示

动作识别演示

实例分割演示

3D 分割演示

图像检索 Python* 演示

多通道多人追踪 Python* 演示

语音库与语音识别演示

语音库

离线语音识别演示

现场语音识别演示

Kaldi* 统计语言模型转换工具

工具

精度检查器工具

[精度检查器示例](#)

[配置 Caffe* 启动程序](#)

[配置 OpenVINO 启动程序](#)

[配置 OpenCV* 启动程序](#)

[配置 MxNet* 启动程序](#)

[配置 TensorFlow* 启动程序](#)

[配置 TensorFlow* Lite 启动程序](#)

[配置 ONNX* Runtime 启动程序](#)

[配置 *PyTorch 启动程序](#)

[适配器](#)

[注释转换器](#)

[预处理程序](#)

[后处理程序](#)

[指标](#)

[面向精度检查器的定制评估器](#)

[阅读器](#)

[Caffe* 安装技巧](#)

[训练后优化](#)

[量化](#)

[DefaultQuantization 算法](#)

[AccuracyAwareQuantization 算法](#)

[使用训练后优化工具套件 API](#)

[定义配置文件](#)

[配置文件示例](#)

[模型下载器](#)

[交叉检查工具](#)

[编译工具](#)

[预训练模型 \(开放式 Model Zoo \)](#)

[对象检测模型](#)

[face-detection-adas-0001](#)

[face-detection-adas-binary-0001](#)

[face-detection-retail-0004](#)

face-detection-retail-0005
person-detection-retail-0002
person-detection-retail-0013
person-detection-action-recognition-0005
person-detection-action-recognition-0006
person-detection-action-recognition-teacher-0002
person-detection-raisinghand-recognition-0001
person-detection-asl-0001
pedestrian-detection-adas-0002
pedestrian-detection-adas-binary-0001
pedestrian-and-vehicle-detector-adas-0001
vehicle-detection-adas-0002
vehicle-detection-adas-binary-0001
person-vehicle-bike-detection-crossroad-0078
person-vehicle-bike-detection-crossroad-1016
product-detection-0001
vehicle-license-plate-detection-barrier-0106

对象识别模型

age-gender-recognition-retail-0013
head-pose-estimation-adas-0001
asl-recognition-0003
license-plate-recognition-barrier-0001
vehicle-attributes-recognition-barrier-0039
emotions-recognition-retail-0003
landmarks-regression-retail-0009
facial-landmarks-35-adas-0002
person-attributes-recognition-crossroad-0230
gaze-estimation-adas-0002

重识别模型

person-reidentification-retail-0031
person-reidentification-retail-0103
person-reidentification-retail-0107
person-reidentification-retail-0200
face-reidentification-retail-0095

语义分割模型

road-segmentation-adas-0001
semantic-segmentation-adas-0001

实例分割模型

instance-segmentation-security-0050
instance-segmentation-security-0083
instance-segmentation-security-0010

人体姿态估计模型

human-pose-estimation-0001

图像处理

single-image-super-resolution-1032

single-image-super-resolution-1033

text-image-super-resolution-0001

文本检测

text-detection-0003

text-detection-0004

文本识别

text-recognition-0012

handwritten-score-recognition-0003

文本辨认

text-spotting-0001-detector

text-spotting-0001-recognizer-decoder

text-spotting-0001-recognizer-encoder

动作识别模型

driver-action-recognition-adas-0002-encoder

driver-action-recognition-adas-0002-decoder

action-recognition-0001-encoder

action-recognition-0001-decoder

图像检索

image-retrieval-0001

压缩模型

resnet18-xnor-binary-onnx-0001

resnet50-binary-0001

resnet18-xnor-binary-onnx-0001

对象检测 C++ 示例 SSD

本文档

[操作步骤](#)

[运行](#)

[示例输出](#)

[另请参阅](#)

该主题展示了如何运行对象检测示例应用，该应用利用对象检测网络（如 SSD-VGG）在英特尔® 处理器和英特尔® 核芯显卡上执行推理。

注：该主题介绍了如何使用对象检测示例 SSD 的 C++ 实现。有关 Python* 实现，请参阅[对象检测 Python* 示例 SSD](#)。

操作步骤

启动时，示例应用读取命令行参数，并将网络和图像加载至推理引擎设备。推理完成后，应用创建一张输出图像，并以标准输出流输出数据。

注：默认情况下，推理引擎示例和演示希望输入采用 BGR 通道顺序。如果您经过训练的模型采用 RGB 顺序，您需要手动重新排列示例或演示应用中的默认通道顺序，或者使用模型优化器工具（指定 `--reverse_input_channels` 参数）重新转换模型。有关参数的更多信息，请参阅[使用常规转换参数转换模型](#)的何时反转输入通道顺序一节。

运行

使用 `-h` 选项运行应用可生成下列使用信息：

```
1 ./object_detection_sample_ssd -h
2 InferenceEngine:
3   API version .....<version>
4   Build .....<number>
5
6 object_detection_sample_ssd [OPTION]
7 Options:
8
9  -h                Print a usage message.
10 -i "<path>"        Required.Path to an .bmp image.
11 -m "<path>"        Required.Path to an .xml file with a trained model.
12 -l "<absolute_path>" Required for CPU custom layers.Absolute path to a shared library with the
    kernels implementations.
13   Or
14 -c "<absolute_path>" Required for GPU custom kernels.Absolute path to the .xml file with the
    kernels descriptions.
15 -d "<device>"      Optional.Specify the target device to infer on (the list of available devices is
    shown below).Default value is CPU.Use "-d HETERO:<comma-separated_devices_list>" format to
    specify HETERO plugin.Sample will look for a suitable plugin for device specified
16 -p_msg            Optional.Enables messages from a plugin
```

使用空选项列表运行应用程序将生成上述使用信息和错误消息。

您可以使用公共或预训练模型运行示例。如欲下载预训练模型，请使用 OpenVINO [模型下载器](#) 或访问 <https://download.01.org/opencv/>。

注：使用经过训练的模型运行示例之前，请确保已使用 [模型优化器工具](#) 将模型转换为推理引擎格式 (*.xml + *.bin)。

例如，如要使用 OpenVINO™ 工具套件人员检测 SSD 模型在 CPU 上执行推理，可运行以下命令：

```
1 ./object_detection_sample_ssd -i <path_to_image>/inputImage.bmp -m <path_to_model>person-detection-retail-0013.xml -d CPU
```

或

```
1 ./object_detection_sample_ssd -i <path_to_image>/inputImage.jpg -m <path_to_model>person-detection-retail-0002.xml -d CPU
```

示例输出

应用输出一张图像 (out_0.bmp)，将检测到的对象用矩形圈出。它向标准输出流输出检测对象的分类列表、各自的置信度和矩形坐标。

另请参阅

- [使用推理引擎示例](#)
- [模型优化器](#)
- [模型下载器](#)

有关编译器优化的更多完整信息，请参阅我们的 [优化声明](#)

支持

[英特尔® OpenVINO™ 工具套件分发版的英特尔® 开发人员专区论坛](#)

Cookie

英特尔 Cookie 和类似技术声明