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

- 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

自动语音识别 C++ 示例

本文档

本主题展示了如何运行语音示例应用，该应用展示了基于 Kaldi* 神经网络和语音特征向量的声学模型推理。

操作步骤

启动时，该应用会读取命令行参数，并将 Kaldi 训练的神经网络和 Kaldi ARK 语音特征向量文件加载至推理引擎插件。然后对存储于 ARK 输入文件中的所有语音发音执行推理。根据 `-bs` 参数，以 1 - 8 帧的批次处理上下文窗口语音帧。该示例不支持跨语音的批处理。完成推理后，应用创建一个 ARK 输出文件。如果提供了 `-r` 选项，将为每个语音发音提供误差统计，如上所述。

GNA 特定细节

量化

如果选择了 GNA 设备（如使用 `-d GNA` 标记），GNA 推理引擎插件将模型和输入特征向量序列量化为整数表示，然后执行推理。多个参数控制神经网络量化。`-q` 标记确定量化模型。支持 3 种模式：静态、动态和用户定义。在静态量化模式中，对 ARK 输入文件中的首个发音进行动态范围扫描。将首个发音的最大输入值扩大到 16384（15 位）所需的比例系数（浮点标量乘法器）用于所有后续输入。量化神经网络，以适应扩大后的输入动态范围。用户定义 - 在用户定义量化模式中，用户可通过 `-sf` 标记指定用于静态量化的比例系数。在动态量化模式中，计算每个输入批次的比例系数，然后对该批次进行推理。利用高效的流程对输入和网络进行动态（重新）量化。

`-qb` 标记为 GNA 插件提供了一个与面向所有层的首选目标权重分辨率相关的提示。例如，指定 `-qb 8` 后，插件将尽可能在网络中使用 8 位权重。请注意，由于 GNA

硬件的限制，不可能一直使用 8 位权重。例如，卷积层通常使用 16 位权重（GNA 硬件版本 1 和 2）。该限制将从 GNA 硬件版本 3 及更高版本中消除。

执行模式

通过 `-d` 标记可支持多种执行模式。如果设备设为 `CPU` 模式，所有计算将通过 CPU 插件在 CPU 设备上执行。如果设备设为 `GNA_AUTO`，将使用 GNA 硬件（如果可用）并安装驱动程序。否则，在 fast-but-not-bit-exact 模式下模拟 GAN 设备。如果设备设为 `GNA_HW`，将使用 GNA 硬件（如果可用）并安装驱动程序。否则，将出现错误。如果设备设为 `GNA_SW`，将在 fast-but-not-bit-exact 模式下模拟 GNA 设备。如果设备设为 `GNA_SW_EXACT`，将在 bit-exact 模式下模拟 GNA 设备。

加载和保存模型

GNA 插件支持通过 `-rg` 和 `-wg` 标记加载和保存 GNA 优化的模型（非 IR）。因此，可以避免在运行时进行完整模型量化的开销。GNA 插件还支持通过 `-we` 标记面向英特尔® 语音支持开发人员套件和 Amazon Alexa* Premium 远场语音开发套件导出兼容固件的嵌入式模型图像（仅保存）。

这些选项不仅能从 GNA 模型文件中直接执行推理，还支持：

- 将 IR 格式转换成 GNA 格式模型文件（`-m`、`-wg`）
- 将 IR 格式转换成嵌入式格式模型文件（`-m`、`-we`）
- 将 GNA 格式转换成嵌入式格式模型文件（`-rg`、`-we`）

运行

使用 `-h` 选项运行应用可生成下列使用信息：

```
$ ./speech_sample -h
InferenceEngine:
    API version ............<version>
    Build ..................<number>

speech_sample [OPTION]
Options:

    -h                    Print a usage message.
    -i "<path>"           Required.Paths to an .ark files.Example of usage:<file1.ark,file2.ark> or <file.ark>.
    -m "<path>"           Required.Path to an .xml file with a trained model (required if -rg is missing).
    -o "<path>"           Optional.Output file name (default name is "scores.ark").
    -l "<absolute_path>"  Required for CPU custom layers.Absolute path to a shared library with the kernel implementations.
```

```
14   -d "<device>"          Optional.Specify a target device to infer on.CPU, GPU, GNA_AUTO, GNA_HW,
   GNA_SW, GNA_SW_EXACT and HETERO with combination of GNA
15    as the primary device and CPU as a secondary (e.g. HETERO:GNA,CPU) are supported.The list of
   available devices is shown below.The sample will look for a suitable plugin for device specified.
16   -p                  Optional.Plugin name.For example, GPU.If this parameter is set, the sample will
   look for this plugin only
17   -pc                 Optional.Enables performance report
18   -q "<mode>"          Optional.Input quantization mode:  "static" (default), "dynamic", or "user" (use
   with -sf).
19   -qb "<integer>"       Optional.Weight bits for quantization:  8 or 16 (default)
20   -sf "<double>"        Optional.Input scale factor for quantization (use with -q user).
21   -bs "<integer>"       Optional.Batch size 1-8 (default 1)
22   -r "<path>"           Optional.Read reference score .ark file and compare scores.
23   -rg "<path>"          Optional.Read GNA model from file using path/filename provided (required if -
   m is missing).
24   -wg "<path>"          Optional.Write GNA model to file using path/filename provided.
25   -we "<path>"          Optional.Write GNA embedded model to file using path/filename provided.
26   -nthreads "<integer>"   Optional.Number of threads to use for concurrent async inference
   requests on the GNA.
27   -cw_l "<integer>"      Optional.Number of frames for left context windows (default is 0).Works
   only with context window networks.
28                    If you use the cw_l or cw_r flag, then batch size and nthreads arguments are ignored.
29   -cw_r "<integer>"      Optional.Number of frames for right context windows (default is 0).Works
   only with context window networks.
30                    If you use the cw_r or cw_l flag, then batch size and nthreads arguments are ignored.
```

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。

模型准备

您可以利用以下模型优化器命令将 Kaldi nnet1 或 nnet2 神经网络转换为英特尔 IR 格式：

```
1  $ python3 mo.py --framework kaldi --input_model wsj_dnn5b_smbr.nnet --counts
   wsj_dnn5b_smbr.counts --remove_output_softmax
```

假设模型优化器 (mo.py)、Kaldi 训练的神经网络 (wsj_dnn5b_smbr.nnet) 和 Kaldi 类 counts 文件 (wsj_dnn5b_smbr.counts) 在工作目录中，该命令将生成包含 wsj_dnn5b_smbr.xml 和 wsj_dnn5b_smbr.bin 的英特尔 IR 网络。

用户可使用以下预训练模型：

- wsj_dnn5b_smbr
- rm_lstm4f
- rm_cnn4a_smbr

所有这些模型都可从 https://download.01.org/openvinotoolkit/models_contrib/speech/kaldi 或使用 OpenVINO 模型下载器下载。

语音推理

创建 IR 后，您可以利用以下命令在搭载 GNA 协处理器（或模拟库）的英特尔® 处理器上执行推理：

```
$ ./speech_sample -d GNA_AUTO -bs 2 -i wsj_dnn5b_smbr_dev93_10.ark -m
  wsj_dnn5b_smbr_fp32.xml -o scores.ark -r wsj_dnn5b_smbr_dev93_scores_10.ark
```

此处，假设与输入特征文件 (wsj_dnn5b_smbr_dev93_10.ark) 对应的 Kaldi 生成的浮点参考神经网络分数 (wsj_dnn5b_smbr_dev93_scores_10.ark) 可用于比较。

注：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)。

示例输出

所有发音的对数似然比序列存储在 Kaldi ARK 文件 scores.ark 中。如果使用了 -r 选项，将针对每个发音生成一个分数误差统计报告，如下所示：

```
Utterance 0:4k0c0301
  Average inference time per frame:6.26867 ms
      max error:0.0667191
      avg error:0.00473641
   avg rms error:0.00602212
    stdev error:0.00393488
```

Kaldi* 语音识别管道中的示例使用

利用 Kaldi s5 方案和 Kaldi Nnet (nnet1) 框架准备本示例中使用的华尔街日报 DNN 模型。通过将 speech_sample 替换为 Kaldi 的 nnet-forward 命令可识别语音。由于 speech_sample 尚未使用管道，运行 Kaldi 语音识别管道时，有必要将临时文件用于说话人转换特征向量和分数。以下操作假设已根据 s5 方案执行特征提取，并且 Kaldi 源代码树中的工作目录为 egs/wsj/s5。

1. 基于 final.feature_transform 中指定的特征转换和 feats.scp 中指定的特征文件准备说话人转换特征：

   ```
   nnet-forward --use-gpu=no final.feature_transform "ark,s,cs:copy-feats scp:feats.scp ark:-
     |" ark:feat.ark
   ```

2. 利用 speech_sample 对特征集进行打分：

```
./speech_sample -d GNA_AUTO -bs 8 -i feat.ark -m wsj_dnn5b_smbr_fp32.xml -o
scores.ark
```

3. 运行 Kaldi 解码器，以生成最优文本假设并基于 WFST (HCLG.fst)、词汇 (words.txt) 和 TID/PID 映射 (final.mdl) 选择最可能的文本：

```
latgen-faster-mapped --max-active=7000 --max-mem=50000000 --beam=13.0 --lattice-
beam=6.0 --acoustic-scale=0.0833 --allow-partial=true --word-symbol-table=words.txt
final.mdl HCLG.fst ark:scores.ark ark:-| lattice-scale --inv-acoustic-scale=13 ark:- ark:- |
lattice-best-path --word-symbol-table=words.txt ark:- ark,t:-  > out.txt &
```

4. 运行误字率工具，根据词汇 (words.txt) 和参考脚本 (test_filt.txt) 检查准确性：

```
cat out.txt | utils/int2sym.pl -f 2- words.txt | sed s:<UNK>::g | compute-wer --text --
mode=present ark:test_filt.txt ark,p:-
```

另请参阅

- 使用推理引擎示例
- 模型优化器
- 模型下载器