Xilinx AI SDK Programming Guide

UG1355 (v2.0) August 13, 2019

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
08/13/2019 Version 2.0	
Editorial updates	Entire document
04/04/2019 Version 1.0	
General updates	Initial Xilinx release

Table of Contents

1 Overview **8**

1.1 The Xilinx AI SDK ... 8

1.1.1 The Xilinx AI SDK Block Diagram ... 8

1.1.2 The Xilinx AI SDK Features ... 9

1.2 How to Use ... 9

1.2.1 Table of Model Libraries ... 9

1.2.2 The basic Process ... 11

2 Class Documentation **12**

2.1 xilinx::ai::Classification Class Reference ... 12

2.1.1 Detailed Description ... 12

2.1.2 Member Function Documentation ... 13

2.1.2.1 create ... 13

2.1.2.2 lookup ... 13

2.1.2.3 run ... 13

2.1.2.4 getInputWidth ... 13

2.1.2.5 getInputHeight ... 14

2.2 xilinx::ai::ClassificationResult Struct Reference ... 14

2.2.1 Detailed Description ... 14

2.2.2 Member Data Documentation ... 14

2.2.2.1 scores ... 14

2.3 xilinx::ai::ClassificationResult::Score Struct Reference ... 14

2.3.1 Detailed Description ... 15

2.4 xilinx::ai::DpuTask Class Reference ... 15

2.4.1 Detailed Description ... 15

2.4.2 Member Function Documentation ... 15

2.4.2.1 create ... 15

2.4.2.2 run ... 16

2.4.2.3 setMeanScaleBGR ... 16

2.4.2.4 setImageBGR ... 16

2.4.2.5 setImageRGB ... 16

Send Feedback

2.4.2.6 getMean ... 16
2.4.2.7 getScale ... 17
2.4.2.8 getInputTensor ... 17
2.4.2.9 getOutputTensor ... 17
2.5 xilinx::ai::FaceDetect Class Reference ... 17
2.5.1 Detailed Description ... 18
2.5.2 Member Function Documentation ... 18
2.5.2.1 create ... 18
2.5.2.2 getInputWidth ... 19
2.5.2.3 getInputHeight ... 19
2.5.2.4 getThreshold ... 19
2.5.2.5 setThreshold ... 19
2.5.2.6 run ... 19
2.6 xilinx::ai::FaceDetectResult Struct Reference ... 20
2.6.1 Detailed Description ... 20
2.7 xilinx::ai::FaceDetectResult::BoundingBox Struct Reference ... 20
2.7.1 Detailed Description ... 20
2.8 xilinx::ai::FaceLandmark Class Reference ... 21
2.8.1 Detailed Description ... 21
2.8.2 Member Function Documentation ... 22
2.8.2.1 create ... 22
2.8.2.2 getInputWidth ... 22
2.8.2.3 getInputHeight ... 22
2.8.2.4 run ... 22
2.9 xilinx::ai::FaceLandmarkResult Struct Reference ... 23
2.9.1 Detailed Description ... 23
2.10 xilinx::ai::InputTensor Struct Reference ... 23
2.10.1 Detailed Description ... 23
2.11 xilinx::ai::MultiTask Class Reference ... 24
2.11.1 Detailed Description ... 24
2.11.2 Member Function Documentation ... 25
2.11.2.1 create ... 25
2.11.2.2 getInputWidth ... 25
2.11.2.3 getInputHeight ... 25
2.11.2.4 run_8UC1 ... 25
2.11.2.5 run_8UC3 ... 25
2.12 xilinx::ai::MultiTask8UC1 Class Reference ... 26
2.12.1 Detailed Description ... 26
2.12.2 Member Function Documentation ... 26
2.12.2.1 create ... 26

2.12.2.2 getInputWidth 27
2.12.2.3 getInputHeight 27
2.12.2.4 run 27
2.13 xilinx::ai::MultiTask8UC3 Class Reference 27
2.13.1 Detailed Description 28
2.13.2 Member Function Documentation 28
2.13.2.1 create 28
2.13.2.2 getInputWidth 28
2.13.2.3 getInputHeight 28
2.13.2.4 run 29
2.14 xilinx::ai::MultiTaskPostProcess Class Reference 29
2.14.1 Member Function Documentation 29
2.14.1.1 create 29
2.14.1.2 post_process_seg 30
2.14.1.3 post_process_seg_visualization 30
2.15 xilinx::ai::MultiTaskResult Struct Reference 30
2.15.1 Detailed Description 30
2.16 xilinx::ai::OpenPose Class Reference 31
2.16.1 Detailed Description 31
2.16.2 Member Function Documentation 32
2.16.2.1 create 32
2.16.2.2 run 33
2.16.2.3 getInputWidth 34
2.16.2.4 getInputHeight 34
2.17 xilinx::ai::OpenPoseResult Struct Reference 34
2.17.1 Detailed Description 34
2.17.2 Member Data Documentation 35
2.17.2.1 poses 35
2.18 xilinx::ai::OpenPoseResult::PosePoint Struct Reference 35
2.18.1 Detailed Description 35
2.18.2 Member Data Documentation 35
2.18.2.1 type 35
2.19 xilinx::ai::OutputTensor Struct Reference 35
2.19.1 Detailed Description 36
2.20 xilinx::ai::PoseDetect Class Reference 36
2.20.1 Detailed Description 36
2.20.2 Member Function Documentation 37
2.20.2.1 create 37
2.20.2.2 getInputWidth 37
2.20.2.3 getInputHeight 37

2.20.2.4 run 38
2.21 xilinx::ai::PoseDetectResult Struct Reference 38
2.21.1 Detailed Description 38
2.22 xilinx::ai::PoseDetectResult::Pose14Pt Struct Reference 38
2.22.1 Detailed Description 39
2.23 xilinx::ai::RefineDet Class Reference 39
2.23.1 Detailed Description 40
2.23.2 Member Function Documentation 41
2.23.2.1 create 41
2.23.2.2 run 41
2.23.2.3 getInputWidth 42
2.23.2.4 getInputHeight 42
2.24 xilinx::ai::RefineDetPostProcess Class Reference 42
2.24.1 Detailed Description 42
2.24.2 Member Function Documentation 43
2.24.2.1 create 43
2.24.2.2 refine_det_post_process 43
2.25 xilinx::ai::RefineDetResult Struct Reference 43
2.25.1 Detailed Description 43
2.26 xilinx::ai::RefineDetResult::BoundingBox Struct Reference 44
2.26.1 Detailed Description 44
2.27 xilinx::ai::Reid Class Reference 44
2.27.1 Detailed Description 44
2.27.2 Member Function Documentation 45
2.27.2.1 create 45
2.27.2.2 run 45
2.27.2.3 getInputWidth 45
2.27.2.4 getInputHeight 46
2.28 xilinx::ai::ReidResult Struct Reference 46
2.28.1 Detailed Description 46
2.29 xilinx::ai::RoadLine Class Reference 46
2.29.1 Detailed Description 47
2.29.2 Member Function Documentation 47
2.29.2.1 create 48
2.29.2.2 getInputWidth 49
2.29.2.3 getInputHeight 49
2.29.2.4 run 49
2.30 xilinx::ai::RoadLinePostProcess Class Reference 49
2.30.1 Detailed Description 50
2.30.2 Member Function Documentation 50

2.30.2.1 create 50
2.30.2.2 road_line_post_process 50
2.31 xilinx::ai::RoadLineResult Struct Reference 50
2.31.1 Detailed Description 51
2.32 xilinx::ai::RoadLineResult::Line Struct Reference 51
2.32.1 Detailed Description 51
2.32.2 Member Data Documentation 51
2.32.2.1 type 51
2.33 xilinx::ai::Segmentation Class Reference 51
2.33.1 Detailed Description 52
2.33.2 Member Function Documentation 52
2.33.2.1 create 53
2.33.2.2 getInputWidth 54
2.33.2.3 getInputHeight 54
2.33.2.4 run_8UC1 54
2.33.2.5 run_8UC3 54
2.34 xilinx::ai::Segmentation8UC1 Class Reference 55
2.34.1 Detailed Description 55
2.34.2 Member Function Documentation 56
2.34.2.1 create 56
2.34.2.2 getInputWidth 57
2.34.2.3 getInputHeight 57
2.34.2.4 run 57
2.35 xilinx::ai::Segmentation8UC3 Class Reference 57
2.35.1 Detailed Description 58
2.35.2 Member Function Documentation 58
2.35.2.1 create 58
2.35.2.2 getInputWidth 58
2.35.2.3 getInputHeight 58
2.35.2.4 run 59
2.36 xilinx::ai::SegmentationResult Struct Reference 59
2.36.1 Detailed Description 59
2.37 xilinx::ai::SSD Class Reference 60
2.37.1 Detailed Description 60
2.37.2 Member Function Documentation 62
2.37.2.1 create 62
2.37.2.2 run 62
2.37.2.3 getInputWidth 63
2.37.2.4 getInputHeight 63
2.38 xilinx::ai::SSDPostProcess Class Reference 63

2.38.1 Detailed Description 63
2.38.2 Member Function Documentation 64
2.38.2.1 create 64
2.38.2.2 ssd_post_process 64
2.39 xilinx::ai::SSDResult Struct Reference 64
2.39.1 Detailed Description 64
2.40 xilinx::ai::SSDResult::BoundingBox Struct Reference 65
2.40.1 Detailed Description 65
2.41 xilinx::ai::Tensor Struct Reference 65
2.41.1 Detailed Description 66
2.42 xilinx::ai::VehicleResult Struct Reference 66
2.42.1 Detailed Description 66
2.42.2 Member Data Documentation 66
2.42.2.1 label 66
2.43 xilinx::ai::YOLOv2 Class Reference 67
2.43.1 Detailed Description 67
2.43.2 Member Function Documentation 67
2.43.2.1 create 67
2.43.2.2 run 68
2.43.2.3 getInputWidth 68
2.43.2.4 getInputHeight 68
2.44 xilinx::ai::YOLOv2Result Struct Reference 68
2.44.1 Detailed Description 69
2.45 xilinx::ai::YOLOv2Result::BoundingBox Struct Reference 69
2.45.1 Detailed Description 69
2.46 xilinx::ai::YOLOv3 Class Reference 69
2.46.1 Detailed Description 70
2.46.2 Member Function Documentation 71
2.46.2.1 create 71
2.46.2.2 getInputWidth 71
2.46.2.3 getInputHeight 71
2.46.2.4 run 72
2.47 xilinx::ai::YOLOv3Result Struct Reference 73
2.47.1 Detailed Description 73
2.48 xilinx::ai::YOLOv3Result::BoundingBox Struct Reference 73
2.48.1 Detailed Description 74

Chapter 1

Overview

1.1 The Xilinx AI SDK

The Xilinx® AI SDK is a set of high-level libraries and APIs built for efficient AI inference with Deep-Learning Processor Unit (DPU). It provides an easy-to-use and unified interface by encapsulating many efficient and high-quality neural networks. This simplifies the use of deep-learning neural networks, even for users without knowledge of deep-learning or FPGAs. The Xilinx AI SDK allows users to focus more on the development of their applications, rather than the underlying hardware.

1.1.1 The Xilinx AI SDK Block Diagram

The Xilinx AI SDK block diagram is shown in the following figure.

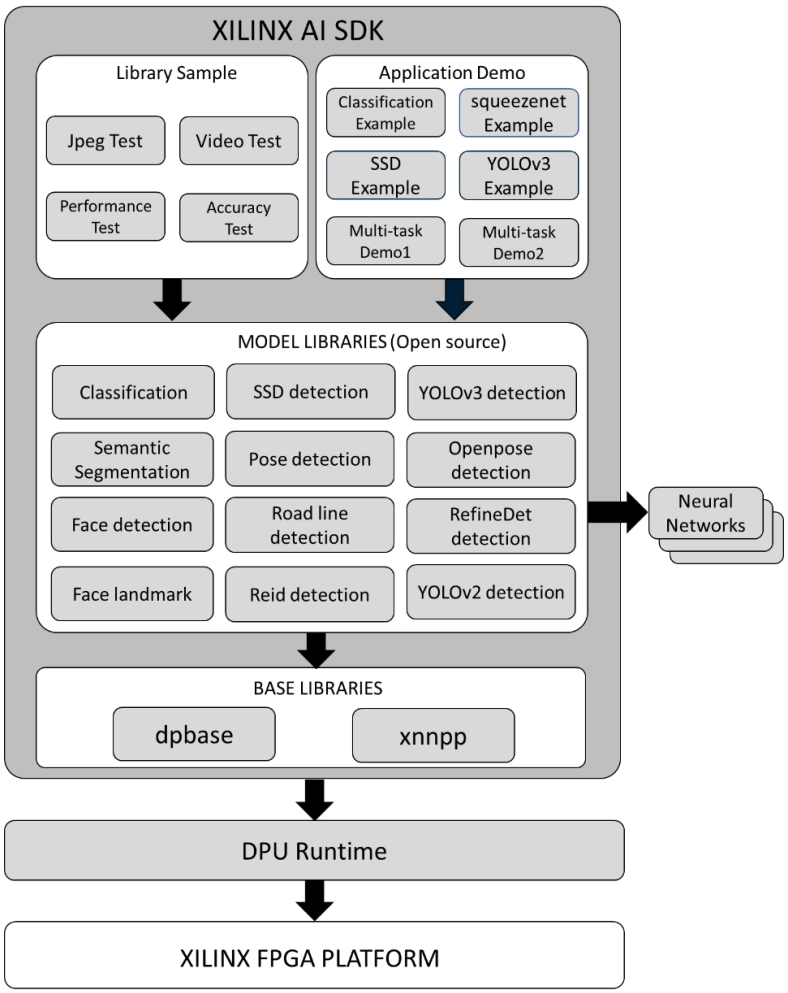


Figure 1.1: The Xilinx AI SDK Block Diagram

The core module of the Xilinx AI SDK includes the model libraries and the base libraries.

The model libraries implement most of the neural network deployment in the Xilinx AI model zoo. It mainly includes common types of networks, such as classification, detection, and segmentation. This module privides users with easy-to-use and fast development channels in a unified interface. Users can use existing model libraries or custom models that replace similar structures to combine market needs to achieve rapid customization if real business.

The base libraries implement encapsulation of the underlying DPU-related operations and the encapsulation of the post-processing acceleration function of each model library. Users can also use the dpbase library to add custom post-processing to deploy new networks.

1.1.2 The Xilinx AI SDK Features

The Xilinx AI SDK Features include:

- Full stack
- Optimized
- Open source
- Unified interface
- Practical application

1.2 How to Use

Development Language Using C++

First, the user needs to prepare the development board and cross-compilation environment. For detailed environment construction, please refer to the *Xilinx AI SDK User Guide* (UG1354).

During the development, you need to pay attention to the header files, library files, and model library files. These files in the development environment must match the version provided in the SDK.

The libraries running on the platform support ZCU102, ZCU104, and Ultra96.

Figure 1.2: Support FPGA Platform

1.2.1 Table of Model Libraries

Library Name	Header file/c++ classes/Library file/modle files	Description
dpfacedetect	xilinx/ai/facedetect.hpp xilinx::ai::FaceDetect libdpfacedetect.so* libdpumodeltiling_v6_320.so libdpumodeltiling_v6_640.so	face detection. including: FACE_DETECT_DENSE_BOX_320x320 FACE_DETECT_DENSE_BOX_640x360
dpssd	xilinx/ai/ssd.hpp xilinx::ai::SSD libdpssd.so* libdpumodelssd_vehicle_v3_480x360.so libdpumodelssd_traffic_480x360.so libdpumodelssd_pedestrian_640x360.so libdpumodelssd_mobilenet_v2_480x360.so libdpumodelssd_voc_300x300.so	object detection , including : SSD_ADAS_VEHICLE_V3_480x360 SSD_TRAFFIC_480x360 SSD_ADAS_PEDESTRIAN_640x360 SSD_MOBILENET_V2_480x360 SSD_VOC_300x300_TF
dpclassification	xilinx/ai/classification.hp xilinx::ai::Classification libdpclassification.so* libdpumodelresnet_50.so libdpumodelinception_v1.so libdpumodelinception_v2.so libdpumodelinception_v3.so libdpumodelinception_v4.so libdpumodelmobilenet_v2.so libdpumodelresnet_50_tf.so libdpumodelresnet_18_tf.so libdpumodelinception_v1_tf.so libdpumodelmobilenet_v1_tf.so libdpumodelmobilenet_v2_tf.so	image classification for ImageNet, including: CLASSIFICATION_RESNET_50 CLASSIFICATION_INCEPTION_V1 CLASSIFICATION_INCEPTION_V2 CLASSIFICATION_INCEPTION_V3 CLASSIFICATION_INCEPTION_V4 CLASSIFICATION_MOBILENET_V2 CLASSIFICATION_RESNET_50_TF CLASSIFICATION_RESNET_18_TF CLASSIFICATION_INCEPTION_V1_TF CLASSIFICATION_MOBILENET_V1_TF CLASSIFICATION_MOBILENET_V2_TF CLASSIFICATION_RESNET_18
dpyolov3	xilinx/ai/yolov3.hpp xilinx::ai::YOLOv3 libdpyolov3.so* libdpumodelyolov3_adas_512x256.so libdpumodelyolov3_adas_512x288.so libdpumodelyolov3_voc_416.so libdpumodelyolov3_voc_416x416_tf.so	for object detection, including: YOLOV3_ADAS_512x256 YOLOV3_ADAS_512x288 YOLOV3_VOC_416x416 YOLOV3_VOC_416x416_TF
dpsegmentatio	xilinx/ai/segmentation.hpp nxilinx::ai::Segmentation libdpsegmentation.so* libdpumodelfpn_deconv.so	for segmentation, including: SEGMENTATION_FPN
dprefinedet	xilinx/ai/refinedet.hpp xilinx::ai::RefineDet libdprefinedet.so* libdpumodelrefinedet_480x360.so libdpumodelrefinedet_480x360_10G.so libdpumodelrefinedet_480x360_5G.so	body detection, including : REFINE_DETECT_480x360 REFINE_DETECT_480x360_10G REFINE_DETECT_480x360_5G

dproadline	xilinx/ai/roadline.hpp xilinx::ai::RoadLine libdproadline.so* libdpumodelroadline.so libdpumodelroadline_deephi.so	roadline detection, including: ROAD_LINE_VPG
dpposedetect	xilinx/ai/posedetect.hpp xilinx::ai::PoseDetect libdpposedetect.so* libdpumodelpose2.so	14-pt gesture detection, including: POSE_DETECT
dpopenpose	xilinx/ai/openpose.hpp xilinx::ai::OpenPose libdpopenpose.so* libdpumodelopenpose_368x368.so libdpumodelopenpose_192x192.so	14-pt gesture detection, including: OPEN_POSE_368x368
dpyolov2	xilinx/ai/yolov2.hpp xilinx::ai::YOLOv2 libdpyolov2.so* libdpumodelyolov2_baseline.so libdpumodelyolov2_compress22G.so libdpumodelyolov2_compress24G.so libdpumodelyolov2_compress26G.so	object detection, including: YOLOV2_VOC_BASELINE YOLOV2_VOC_COMPRESS22G YOLOV2_VOC_COMPRESS24G YOLOV2_VOC_COMPRESS26G
dplandmark	xilinx/ai/facelandmark.hpp xilinx::ai::FaceLandmark libdpfacelandmark.so* libdpumodellandmark_attr_v1_7x.so	five key points detection, including: FACE_LANDMARK
dpreid	xilinx/ai/reid.hpp xilinx::ai::Reid libdpreid.so* libdpumodelreid.so	REID

1.2.2 The Basic Process

The basic process:

- Select an image (cv::Mat)
- Call the create method provided by the corresponding library to get class instance. If needed, set preprocess as `false`. The model will not subtract its mean and scale. Please use it only in the pre-minus means and scale.
- Call getInputWidth() and getInputHeight() to get the network need columns and rows of the input image.
- Resize image to inputWidth x inputHeight
- Call run() to get result of the network.

Chapter 2

Class Documentation

2.1 xilinx::ai::Classification Class Reference

Base class for detecting objects in the input image (cv::Mat).

`#include <xilinx/ai/classification.hpp>`

Public Member Functions

- **Classification** (const Classification &)=delete
- virtual xilinx::ai::ClassificationResult run (const cv::Mat &image)=0

  Function of get running result of the classification neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the classification network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the classification network (input image rows).

Static Public Member Functions

- static std::unique_ptr < Classification > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Classification.
- static const char * lookup (int index)

  Get the classification corresponding by index.

2.1.1 Detailed Description

Base class for detecting objects in the input image (cv::Mat).

Input is an image (cv::Mat).

Output is index and score of objects in the input image.

Sample code:

```
auto image = cv::imread("test.jpg");
auto network = xilinx::ai::Classification::create(
               xilinx::ai::CLASSIFICATION_RESNET_50,
               true);
auto result = network->run(image);
for (const auto &r : result.scores) {
```

```
    auto score = r.score;
    auto index = network->lookup(r.index);
}
```

2.1.2 Member Function Documentation

2.1.2.1 static std::unique_ptr<Classification> xilinx::ai::Classification::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class Classification.

Parameters

model_name	Model name.
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Classification class.

2.1.2.2 static const char* xilinx::ai::Classification::lookup (int *index*) [static]

Get the classification corresponding by index.

Parameters

index	The network result

Returns

Classification description, if index < 0, return empty string

2.1.2.3 virtual xilinx::ai::ClassificationResult xilinx::ai::Classification::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the classification neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

ClassificationResult.

2.1.2.4 virtual int xilinx::ai::Classification::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the classification network (input image cols).

Returns

InputWidth of the classification network

2.1.2.5 virtual int xilinx::ai::Classification::getInputHeight ( ) const [pure virtual]

Function to get InputHeight of the classification network (input image rows).

Returns

InputHeight of the classification network.

2.2 xilinx::ai::ClassificationResult Struct Reference

Struct of the result with the classification network.

`#include <xilinx/ai/nnpp/classification.hpp>`

Classes

- struct Score

  The struct of index and confidence for an object.

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- std::vector< Score > scores

2.2.1 Detailed Description

Struct of the result with the classification network.

2.2.2 Member Data Documentation

2.2.2.1 std::vector<Score> xilinx::ai::ClassificationResult::scores

A vector of objects width confidence in the first k, k defaults to 5 and can be modified through the model configutation file.

2.3 xilinx::ai::ClassificationResult::Score Struct Reference

The struct of index and confidence for an object.

`#include <xilinx/ai/nnpp/classification.hpp>`

Public Attributes

- int index

  Result's index in ImageNet.
- float score

  Confidence of this category.

2.3.1 Detailed Description

The struct of index and confidence for an object.

2.4 xilinx::ai::DpuTask Class Reference

Base class for run a DPU task.

`#include <xilinx/ai/dpu_task.hpp>`

Public Member Functions

- **DpuTask** (const DpuTask &other)=delete
- DpuTask & **operator=** (const DpuTask &rhs)=delete
- virtual void run ()=0

  Run the dpu task.
- virtual void setMeanScaleBGR (const std::vector< float > &mean, const std::vector< float > &scale)=0

  Set the mean/scale values.
- virtual void setImageBGR (const cv::Mat &img)=0

  Copy a input image in BGR format to the input tensor.
- virtual void setImageRGB (const cv::Mat &img)=0

  Copy a input image in RGB format to the input tensor.
- virtual std::vector< float > getMean ()=0

  Get the mean values.
- virtual std::vector< float > getScale ()=0

  Get the scale values.
- virtual std::vector
  < xilinx::ai::InputTensor > getInputTensor ()=0

  Get the input tensors.
- virtual std::vector
  < xilinx::ai::OutputTensor > getOutputTensor ()=0

  Get the output tensors.

Static Public Member Functions

- static std::unique_ptr< DpuTask > create (const std::string &kernal_name)

  A static method to create a DPU task.

2.4.1 Detailed Description

Base class for run a DPU task.

2.4.2 Member Function Documentation

2.4.2.1 static std::unique_ptr<DpuTask> xilinx::ai::DpuTask::create (const std::string & *kernal_name*) [static]

A static method to create a DPU task.

Parameters

kernel_name	The dpu kernel name. for example, if kernel_name is "resnet_50", the following dpu model files are searched. ./libdpumodelrestnet_50.so /usr/lib/libdpumodelrestnet_50.so

Returns

A DpuTask instance.

2.4.2.2 virtual void xilinx::ai::DpuTask::run ( ) [pure virtual]

Run the dpu task.

Note

Before invoking this function. An input data should be properly copied to input tensors, via `setImageBGR` or `setImageRGB`.

2.4.2.3 virtual void xilinx::ai::DpuTask::setMeanScaleBGR (const std::vector< float > & *mean*, const std::vector< float > & *scale*) [pure virtual]

Set the mean/scale values.

Note

By default, no mean-scale processing, after invoking this function, mean-scale processing is enabled. You cannot turn it off after enabling.

Parameters

mean	Mean, Normalization is used.
scale	Scale, Normalization is used.

2.4.2.4 virtual void xilinx::ai::DpuTask::setImageBGR (const cv::Mat & *img*) [pure virtual]

Copy a input image in BGR format to the input tensor.

Parameters

img	The input image (cv::Mat).

2.4.2.5 virtual void xilinx::ai::DpuTask::setImageRGB (const cv::Mat & *img*) [pure virtual]

Copy a input image in RGB format to the input tensor.

Parameters

img	The input image(cv::Mat).

2.4.2.6 virtual std::vector<float> xilinx::ai::DpuTask::getMean ( ) [pure virtual]

Get the mean values.

Returns

Mean values

2.4.2.7 virtual std::vector<float> xilinx::ai::DpuTask::getScale ( ) [pure virtual]

Get the scale values.

Returns

Scale values

2.4.2.8 virtual std::vector<xilinx::ai::InputTensor> xilinx::ai::DpuTask::getInputTensor ( ) [pure virtual]

Get the input tensors.

Returns

The input tensors

2.4.2.9 virtual std::vector<xilinx::ai::OutputTensor> xilinx::ai::DpuTask::getOutputTensor ( ) [pure virtual]

Get the output tensors.

Returns

The output tensors.

2.5 xilinx::ai::FaceDetect Class Reference

Base class for detecting the position of faces in the input image (cv::Mat).

`#include <xilinx/ai/facedetect.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0
  Function to get InputWidth of the facedetect network (input image cols).
- virtual int getInputHeight () const =0
  Function to get InputHeigth of the facedetect network (input image rows).
- virtual float getThreshold () const =0
  Function to get detect threshold.
- virtual void setThreshold (float threshold)=0
  Function of update detect threshold.
- virtual FaceDetectResult run (const cv::Mat &img)=0
  Function of get running result of the facedetect network.

Static Public Member Functions

- static std::unique_ptr
  < FaceDetect > create (const std::string &model_name, bool need_preprocess=true)
  Factory function to get instance of derived classes of class FaceDetect.

Protected Member Functions

- **FaceDetect** (const FaceDetect &)=delete
- FaceDetect & **operator=** (const FaceDetect &)=delete

2.5.1 Detailed Description

Base class for detecting the position of faces in the input image (cv::Mat).

Input is an image (cv::Mat).

Output is a vector of position and score for faces in the input image.

Sample code:

```
auto image = cv::imread("test_faces.jpg");
auto network = xilinx::ai::FaceDetect::create(
               xilinx::ai::FACE_DETCTE_DENSE_BOX_640x360,
               true);
auto result = network->run(image);
for (const auto &r : result) {
   auto score = r.score;
   auto x = r.x * image.cols;
   auto y = r.y * image.rows;
   auto width = r.width * image.cols;
   auto height = rz.height * image.rows;
}
```

Display of the facedetect model results:

Figure 2.1: facedetect result image

2.5.2 Member Function Documentation

2.5.2.1 static std::unique_ptr<FaceDetect> xilinx::ai::FaceDetect::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get instance of derived classes of class FaceDetect.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of FaceDetect class.

2.5.2.2 virtual int xilinx::ai::FaceDetect::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the facedetect network (input image cols).

Returns

InputWidth of the facedetect network

2.5.2.3 virtual int xilinx::ai::FaceDetect::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the facedetect network (input image rows).

Returns

InputHeight of the facedetect network.

2.5.2.4 virtual float xilinx::ai::FaceDetect::getThreshold ( ) const [pure virtual]

Function to get detect threshold.

Returns

The detect threshold , the value range from 0 to 1.

2.5.2.5 virtual void xilinx::ai::FaceDetect::setThreshold (float *threshold*) [pure virtual]

Function of update detect threshold.

Note

The detection results will filter by detect threshold (score >= threshold).

Parameters

threshold	The detect threshold,the value range from 0 to 1.

2.5.2.6 virtual FaceDetectResult xilinx::ai::FaceDetect::run (const cv::Mat & *img*) [pure virtual]

Function of get running result of the facedetect network.

Parameters

img	Input Data ,input image (cv::Mat) need to be resized to InputWidth and InputHeight required by the network.

Returns

The detection result of the face detect network , filter by score >= det_threshold

2.6 xilinx::ai::FaceDetectResult Struct Reference

Struct of the result with the facedetect network.

`#include <xilinx/ai/nnpp/facedetect.hpp>`

Classes

- struct BoundingBox

  The coordinate and confidence of a face.

Public Attributes

- int width

  Width of a input image.
- int height

  Height of a input image.
- std::vector< BoundingBox > rects

  All faces, filtered by confidence >= detect threshold.

2.6.1 Detailed Description

Struct of the result with the facedetect network.

2.7 xilinx::ai::FaceDetectResult::BoundingBox Struct Reference

The coordinate and confidence of a face.

`#include <xilinx/ai/nnpp/facedetect.hpp>`

Public Attributes

- float x

  x-coordinate , x is normalized relative to the input image cols ,the value range from 0 to 1.
- float y

  y-coordinate , y is normalized relative to the input image rows ,the value range from 0 to 1.
- float width

  face width , width is normalized relative to the input image cols , the value range from 0 to 1.
- float height

  face height , heigth is normalized relative to the input image rows ,the value range from 0 to 1.
- float score

  face confidence, the value range from 0 to 1.

2.7.1 Detailed Description

The coordinate and confidence of a face.

2.8 xilinx::ai::FaceLandmark Class Reference

Base class for detecting five key points, gender, age and score from a face image (cv::Mat).

`#include <xilinx/ai/facelandmark.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the landmark network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the landmark network (input image rows).
- virtual FaceLandmarkResult run (const cv::Mat &input_image)=0

  Function of get running result of the face landmark network.

Static Public Member Functions

- static std::unique_ptr
  < FaceLandmark > create (const std::string &model_name=FACE_LANDMARK, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class FaceLandmark.

Protected Member Functions

- **FaceLandmark** (const FaceLandmark &other)=delete

2.8.1 Detailed Description

Base class for detecting five key points and score from a face image (cv::Mat). Input a face

image (cv::Mat).

Output score, five key points of the face.

Note

Usually the input image contains only one face, when contains multiple faces will return the highest score.

Sample code:

```
cv:Mat image = cv::imread("test_face.jpg");
auto landmark = xilinx::ai::FaceLandmark::create();
auto result = landmark->run(image);
float score = result.score;
auto points = result.points;
for(int i = 0; i< 5 ; ++i){
    auto x = points[i].frist * image.cols;
    auto y = points[i].second * image.rows;
}
```

Display of the landmark model results:

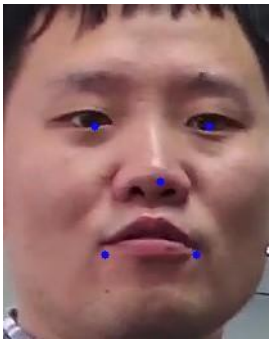


Figure 2.2: FaceLandmark Result Image

2.8.2 Member Function Documentation

2.8.2.1 static std::unique_ptr<FaceLandmark> xilinx::ai::FaceLandmark::create (const std::string & *model_name* = FACE_LANDMARK, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class FaceLandmark.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of FaceLandmark class.

2.8.2.2 virtual int xilinx::ai::FaceLandmark::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the landmark network (input image cols).

Returns

InputWidth of the face landmark network.

2.8.2.3 virtual int xilinx::ai::FaceLandmark::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the landmark network (input image rows).

Returns

InputHeight of the face landmark network.

2.8.2.4 virtual FaceLandmarkResult xilinx::ai::FaceLandmark::run (const cv::Mat & *input_image*) [pure virtual]

Function of get running result of the face landmark network.

Set data of a face(e.g data of cv::Mat) and get the five key points , gender and age .

Parameters

input_image	Input data of input image (cv::Mat) of detected by the facedetect network and resized as inputwidth and inputheight.

Returns

The struct of FaceLandmarkResult

2.9 xilinx::ai::FaceLandmarkResult Struct Reference

Struct of the result returned by the landmark network.

`#include <xilinx/ai/nnpp/facelandmark.hpp>`

Public Attributes

- std::array< std::pair< float, float >, 5 > points

  Five key points coordinate, this array of <x,y> has 5 elements ,x / y is normalized relative to width / height, the value range from 0 to 1.

2.9.1 Detailed Description

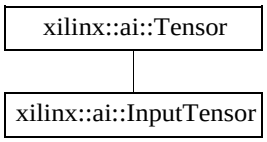
Struct of the result returned by the landmark network.

2.10 xilinx::ai::InputTensor Struct Reference

The actual data of input tensor.

`#include <xilinx/ai/tensor.hpp>`

Inheritance diagram for xilinx::ai::InputTensor:



Public Attributes

- uintptr_t phy_addr

  The start physical address of this tensor.
- void * data

  The start pointer of this Tensor.

2.10.1 Detailed Description

The actual data of input tensor.

2.11 xilinx::ai::MultiTask Class Reference

Base class for ADAS MuiltTask from a image (cv::Mat).

`#include <xilinx/ai/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run_8UC1 (const cv::Mat&image)=0

  Function of get running result from the MultiTask network.
- virtual MultiTaskResult run_8UC3 (const cv::Mat&image)=0

  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr< MultiTask > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Multitask.

Protected Member Functions

- **MultiTask** (const MultiTask&)=delete

2.11.1 Detailed Description

Base class for ADAS MuiltTask from an image (cv::Mat).

Input an image (cv::Mat).

Output is a struct of MultiTaskResult include segmentation results, detection detection results and vehicle towards;

Sample code:

```
auto det = xilinx::ai::MultiTask::create(xilinx::ai::MULTITASK);
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run_8UC3(image);
cv::imwrite("res.jpg",result.segmentation);
```

Display of the multitask model results:

Figure 2.3: Multitask Visualization Result Image

2.11.2 Member Function Documentation

2.11.2.1 static std::unique_ptr<MultiTask> xilinx::ai::MultiTask::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get an instance of derived classes of class Multitask.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Multitask class.

2.11.2.2 virtual int xilinx::ai::MultiTask::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

2.11.2.3 virtual int xilinx::ai::MultiTask::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

2.11.2.4 virtual MultiTaskResult xilinx::ai::MultiTask::run_8UC1 (const cv::Mat & *image*) [pure virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC1 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

2.11.2.5 virtual MultiTaskResult xilinx::ai::MultiTask::run_8UC3 (const cv::Mat & *image*) [pure virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC3 of the MultiTaskResult.segmentation.

Parameters

image	Input image;

Returns

The struct of MultiTaskResult

2.12 xilinx::ai::MultiTask8UC1 Class Reference

Base class for ADAS MuiltTask8UC1 from a image (cv::Mat).

`#include <xilinx/ai/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const

  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const

  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run (const cv::Mat&image)

  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr
  < MultiTask8UC1 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class MultiTask8UC1.

Protected Member Functions

- **MultiTask8UC1** (std::unique_ptr< MultiTask > multitask)
- **MultiTask8UC1** (const MultiTask8UC1 &)=delete

2.12.1 Detailed Description

Base class for ADAS MuiltTask8UC1 from a image (cv::Mat).

Input is an image (cv::Mat).

Output is struct MultiTaskResult include segmentation results, detection results and vehicle towards; The result cv::Mat type is CV_8UC1

Sample code:

```
auto det = xilinx::ai::MultiTask8UC1::create(xilinx::ai::MULTITASK);
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run(image);
cv::imwrite("res.jpg",result.segmentation);
```

2.12.2 Member Function Documentation

2.12.2.1 static std::unique_ptr<MultiTask8UC1> xilinx::ai::MultiTask8UC1::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [inline], [static]

Factory function to get an instance of derived classes of class MultiTask8UC1.

Parameters

model_name	Model name
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of MultiTask8UC1 class.

2.12.2.2 virtual int xilinx::ai::MultiTask8UC1::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

2.12.2.3 virtual int xilinx::ai::MultiTask8UC1::getInputHeight ( ) const [inline], [virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

2.12.2.4 virtual MultiTaskResult xilinx::ai::MultiTask8UC1::run (const cv::Mat & *image*) [inline], [virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC1 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

2.13 xilinx::ai::MultiTask8UC3 Class Reference

Base class for ADAS MuiltTask8UC3 from a image (cv::Mat).

`#include <xilinx/ai/multitask.hpp>`

Public Member Functions

- virtual int getInputWidth () const

  Function to get InputWidth of the multitask network (input image cols).
- virtual int getInputHeight () const

  Function to get InputHight of the multitask network (input image rows).
- virtual MultiTaskResult run (const cv::Mat&image)

  Function of get running result from the MultiTask network.

Static Public Member Functions

- static std::unique_ptr
  < MultiTask8UC3 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class MultiTask8UC3.

Protected Member Functions

- **MultiTask8UC3** (std::unique_ptr< MultiTask > multitask)
- **MultiTask8UC3** (const MultiTask8UC3 &)=delete

2.13.1 Detailed Description

Base class for ADAS MuiltTask8UC3 from a image (cv::Mat).

Input is an image (cv::Mat).

Output is struct MultiTaskResult include segmentation results, detection results and vehicle towards; The result cv::Mat type is CV_8UC3

Sample code:

```
auto det = xilinx::ai::MultiTask8UC3::create(xilinx::ai::MULITASK);
auto image = cv::imread("sample_multitask.jpg");
auto result = det->run(image);
cv::imwrite("res.jpg",result.segmentation);
```

2.13.2 Member Function Documentation

2.13.2.1 static std::unique_ptr<MultiTask8UC3> xilinx::ai::MultiTask8UC3::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [inline], [static]

Factory function to get a instance of derived classes of class MultiTask8UC3.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of MultiTask8UC3 class.

2.13.2.2 virtual int xilinx::ai::MultiTask8UC3::getInputWidth ( ) const [inline], [virtual]

Function to get InputWidth of the multitask network (input image cols).

Returns

InputWidth of the multitask network.

2.13.2.3 virtual int xilinx::ai::MultiTask8UC3::getInputHeight ( ) const [inline], [virtual]

Function to get InputHight of the multitask network (input image rows).

Returns

InputHeight of the multitask network.

2.13.2.4 virtual MultiTaskResult xilinx::ai::MultiTask8UC3::run (const cv::Mat & *image*) [inline],[virtual]

Function of get running result from the MultiTask network.

Note

The type is CV_8UC3 of the MultiTaskResult.segmentation.

Parameters

image	Input image

Returns

The struct of MultiTaskResult

2.14 xilinx::ai::MultiTaskPostProcess Class Reference

Public Member Functions

- virtual MultiTaskResult post_process_seg ()=0

  The post-processing function of the multitask which stored the original segmentation classes.
- virtual MultiTaskResult post_process_seg_visualization ()=0

  The post-processing function of the multitask which return a result include segmentation image mapped to color.

Static Public Member Functions

- static std::unique_ptr < MultiTaskPostProcess > create (const std::vector < std::vector < xilinx::ai::InputTensor > > &input_tensors, const std::vector < std::vector < xilinx::ai::OutputTensor > > &output_tensors, const xilinx::ai::proto::DpuModelParam &config)

  Factory function to get a instance of derived classes of MultiTaskPostProcess.

Protected Member Functions

- **MultiTaskPostProcess** (const MultiTaskPostProcess &)=delete
- MultiTaskPostProcess & **operator=** (const MultiTaskPostProcess &)=delete

2.14.1 Member Function Documentation

2.14.1.1 static std::unique_ptr<MultiTaskPostProcess> xilinx::ai::MultiTaskPostProcess::create (const std::vector< std::vector< xilinx::ai::InputTensor >> & *input_tensors*, const std::vector< std::vector< xilinx::ai::OutputTensor >> & *output_tensors*, const xilinx::ai::proto::DpuModelParam & *config*) [static]

Factory function to get a instance of derived classes of MultiTaskPostProcess.

Parameters

input_tensors	A vector of all input-tensors in the network. Usage: input_tensors[kernel_index][input_tensor-_index].
output_tensors	A vector of all output-tensors in the network. Usage: output_tensors[kernel_index][output_-index].
config	The dpu model configuration information.

Returns

The struct of MultiTaskResult.

2.14.1.2 virtual MultiTaskResult xilinx::ai::MultiTaskPostProcess::post_process_seg ( ) [pure virtual]

The post-processing function of the multitask which stored the original segmentation classes.

Returns

The struct of SegmentationResult.

2.14.1.3 virtual MultiTaskResult xilinx::ai::MultiTaskPostProcess::post_process_seg_visualization ( ) [pure virtual]

The post-processing function of the multitask which return a result include segmentation image mapped to color.

Returns

The struct of SegmentationResult.

2.15 xilinx::ai::MultiTaskResult Struct Reference

Struct of the result returned by the MultiTask network, when you need to visualize.

`#include <xilinx/ai/nnpp/multitask.hpp>`

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- std::vector< VehicleResult > vehicle

  Detection result of SSD task.
- cv::Mat segmentation

  Segmentation result to visualize , cv::Mat type is CV_8UC1 or CV_8UC3.

2.15.1 Detailed Description

Struct of the result returned by the MultiTask network, when you need to visualize.

2.16 xilinx::ai::OpenPose Class Reference

Base class for detecting poses of people.

`#include <xilinx/ai/openpose.hpp>`

Public Member Functions

- **OpenPose** (const OpenPose &)=delete
- virtual OpenPoseResult run (const cv::Mat &image)=0

  Function of get running result of the openpose neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the openpose network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the openpose network (input image rows).

Static Public Member Functions

- static std::unique_ptr< OpenPose > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class OpenPose.

2.16.1 Detailed Description

Base class for detecting poses of people.

Input is an image (cv:Mat).

Output is a OpenPoseResult.

Sample code :

```
 auto image = cv::imread(argv[1]);
 if (image.empty()) {
   std::cerr << "cannot load " << argv[1] << std::endl;
   abort();
 }
 auto det = xilinx::ai::OpenPose::create(xilinx::ai; int width = det-
 >getInputWidth();
 int height = det->getInputHeight();
 vector<vector<int>> limbSeq = {{0,1}, {1,2}, {2,3}, {3,4}, {1,5}, {5,6},
{6,7}, {1,8}, ¥ {8,9}, {9,10}, {1,11}, {11,12}, {12,13}}; float scale_x =
float(image.cols) / float(width); float scale_y = float(image.rows) /
float(height); auto results = det->run(image); for(size_t k = 1; k <
results.poses.size(); ++k){ for(size_t i = 0; i < results.poses[k].size();
++i){ if(results.poses[k][i].type == 1){ results.poses[k][i].point.x *=
scale_x; results.poses[k][i].point.y *= scale_y; cv::circle(image,
results.poses[k][i].point, 5, cv::Scalar(0, 255, 0), -1);
        }
    }
    for(size_t i = 0; i < limbSeq.size(); ++i){
        Result a = results.poses[k][limbSeq[i][0]];
        Result b = results.poses[k][limbSeq[i][1]];
        if(a.type == 1 && b.type == 1){
            cv::line(image, a.point, b.point, cv::Scalar(255, 0, 0), 3, 4);
        }
    }
 }
```

Display of the openpose model results:

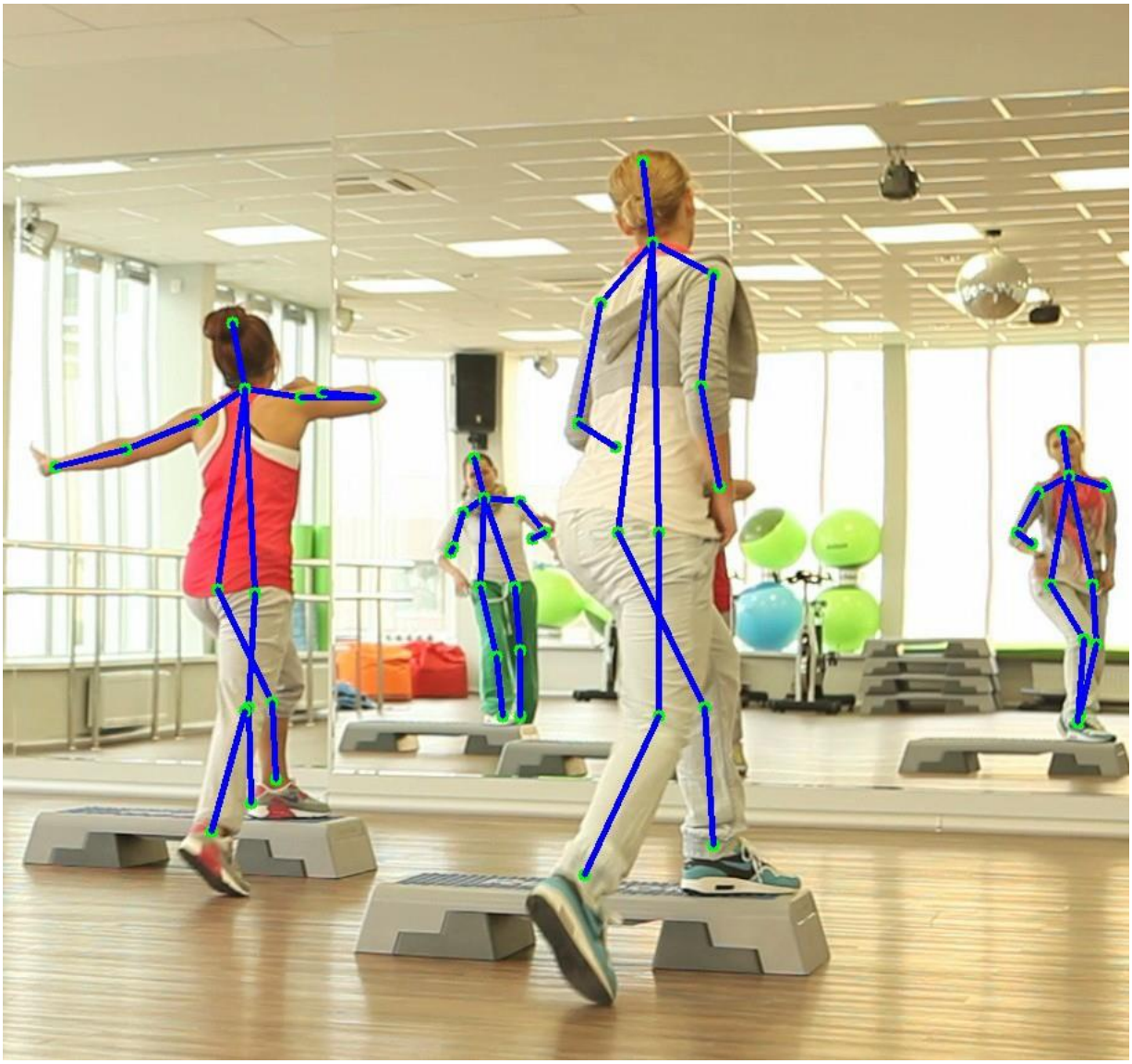


Figure 2.4: openpose result image

2.16.2 Member Function Documentation

2.16.2.1 static std::unique_ptr<OpenPose> xilinx::ai::OpenPose::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class OpenPose.

Parameters

need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of OpenPose class.

2.16.2.2 virtual OpenPoseResult xilinx::ai::OpenPose::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the openpose neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

OpenPoseResult.

2.16.2.3 virtual int xilinx::ai::OpenPose::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the openpose network (input image cols).

Returns

InputWidth of the openpose network

2.16.2.4 virtual int xilinx::ai::OpenPose::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the openpose network (input image rows).

Returns

InputHeight of the openpose network.

2.17 xilinx::ai::OpenPoseResult Struct Reference

Result with the openpose network.

`#include <xilinx/ai/nnpp/openpose.hpp>`

Classes

- struct PosePoint

  Struct of a coordinate point and the point type.

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- std::vector< std::vector < PosePoint > > poses

2.17.1 Detailed Description

Result with the openpose network.

2.17.2 Member Data Documentation

2.17.2.1 std::vector<std::vector<PosePoint> > xilinx::ai::OpenPoseResult::poses

A vector of pose, pose is represented by a vector of PosePoint. Joint points are arranged in order 0: head, 1: neck, 2: L_shoulder, 3:L_elbow, 4: L_wrist, 5: R_shoulder, 6: R_elbow, 7: R_wrist, 8: L_hip, 9:L_knee, 10: L_ankle, 11: R_hip, 12: R_knee, 13: R_ankle

2.18 xilinx::ai::OpenPoseResult::PosePoint Struct Reference

Struct of a coordinate point and the point type.

`#include <xilinx/ai/nnpp/openpose.hpp>`

Public Attributes

- int type = 0

  Point type.
- cv::Point2f point

  Coordinate point.

2.18.1 Detailed Description

Struct of a coordinate point and the point type.

2.18.2 Member Data Documentation

2.18.2.1 int xilinx::ai::OpenPoseResult::PosePoint::type = 0

Point type.

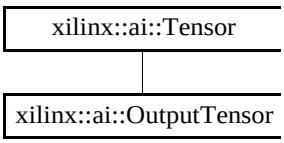
- 1 : "valid"
- 3 : "invalid"

2.19 xilinx::ai::OutputTensor Struct Reference

The actual data of output tensor.

`#include <xilinx/ai/tensor.hpp>`

Inheritance diagram for xilinx::ai::OutputTensor:



Public Attributes

- uintptr_t phy_addr

  The start physical address of this tensor.
- void * data

  The start pointer of this tensor.

2.19.1 Detailed Description

The actual data of output tensor.

2.20 xilinx::ai::PoseDetect Class Reference

Base class for detecting a pose from a input image (cv::Mat).

`#include <xilinx/ai/posedetect.hpp>`

Public Member Functions

- **PoseDetect** (const PoseDetect&)=delete
- virtual int getInputWidth () const =0

  Function to get InputWidth of the PoseDetect network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the PoseDetect network (input image rows).
- virtual PoseDetectResult run (const cv::Mat &image)=0

  Function of get running result of the posedetect neuron network.

Static Public Member Functions

- static std::unique_ptr
  < PoseDetect > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class PoseDetect.

2.20.1 Detailed Description

Base class for detecting a pose from a input image (cv::Mat).

Note

Support detect a signle pose.

Input an image (cv::Mat).

Output is a struct of PoseDetectResult, include 14 point.

Sample code:

```
auto det = xilinx::ai::PoseDetect::create("posedetect");
auto image = cv::imread("sample.jpg");
auto results = det->run(image);
for(auto result: results.pose14pt) {
    std::cout << result << std::endl;
}
```

Display of the posedetect model results:

Figure 2.5: pose detect image

2.20.2 Member Function Documentation

2.20.2.1 static std::unique_ptr<PoseDetect> xilinx::ai::PoseDetect::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class PoseDetect.

Parameters

model_name	Model name .
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of PoseDetect class.

2.20.2.2 virtual int xilinx::ai::PoseDetect::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the PoseDetect network (input image cols).

Returns

InputWidth of the PoseDetect network.

2.20.2.3 virtual int xilinx::ai::PoseDetect::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the PoseDetect network (input image rows).

Returns

InputHeight of the PoseDetect network.

2.20.2.4 virtual PoseDetectResult xilinx::ai::PoseDetect::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the posedetect neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

PoseDetectResult.

2.21 xilinx::ai::PoseDetectResult StructReference

Struct of the result returned by the posedetect network.

`#include <xilinx/ai/nnpp/posedetect.hpp>`

Classes

- struct Pose14Pt

  A pose , represented by 14 coordinate points.

Public Types

- using Point = cv::Point2f

  A coordinate point.

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- Pose14Pt pose14pt

  The pose of input image.

2.21.1 Detailed Description

Struct of the result returned by the posedetect network.

2.22 xilinx::ai::PoseDetectResult::Pose14Pt Struct Reference

A pose , represented by 14 coordinate points.

`#include <xilinx/ai/nnpp/posedetect.hpp>`

Public Attributes

- Point right_shoulder

  R_shoulder coordinate.
- Point right_elbow

  R_elbow coordinate.
- Point right_wrist

  R_wrist coordinate.
- Point left_shoulder

  L_shoulder coordinate.
- Point left_elbow

  L_elbow coordinate.
- Point left_wrist

  L_wrist coordinate.
- Point right_hip

  R_hip coordinate.
- Point right_knee

  R_knee coordinate.
- Point right_ankle

  R_ankle coordinate.
- Point left_hip

  L_hip coordinate.
- Point left_knee

  L_knee coordinate.
- Point left_ankle

  L_ankle coordinate.
- Point head

  head coordinate
- Point neck

  neck coordinate

2.22.1 Detailed Description

A pose , represented by 14 coordinate points.

2.23 xilinx::ai::RefineDet Class Reference

Base class for detecting pedestrian in the input image (cv::Mat).

`#include <xilinx/ai/refinedet.hpp>`

Public Member Functions

- **RefineDet** (const RefineDet &)=delete
- virtual RefineDetResult run (const cv::Mat &image)=0

  Function of get running result of the RefineDet neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the refinedet network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the refinedet network (input image rows).

Static Public Member Functions

- static std::unique_ptr< RefineDet > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class RefineDet.

2.23.1 Detailed Description

Base class for detecting pedestrian in the input image (cv::Mat).

Input is a image (cv::Mat).

Output is position and score of pedestrian in the input image.

Sample code:

```
auto set = getenv("REFINEDET");
string type =
    xilinx::ai::REFINE_DETECT_480x360;
if (set && string(set) == "480x360_10G") {
  type = xilinx::ai::REFINE_DETECT_480x360_10G;
}
if (set && string(set) == "480x360_5G") {
  type = xilinx::ai::REFINE_DETECT_480x360_5G;
}
auto det = xilinx::ai::RefineDet::create(type);
auto image_file = string(argv[1]);
auto image = cv::imread(image_file);
cout << "load image" << endl;
if (image.empty()) {
  cerr << "cannot load " << argv[1] << endl;
  abort();
}

auto results = det->run(image);

auto img = image.clone();
for (auto &box : results.bboxes) {
    float x = box.x * (img.cols);
    float y = box.y * (img.rows);
    int xmin = x;
    int ymin = y;
    int xmax = x + (box.width) * (img.cols);
    int ymax = y + (box.height) * (img.rows);
    float score = box.score;
    xmin = std::min(std::max(xmin, 0), img.cols);
    xmax = std::min(std::max(xmax, 0), img.cols);
    ymin = std::min(std::max(ymin, 0), img.rows);
    ymax = std::min(std::max(ymax, 0), img.rows);

    cv::rectangle(img, cv::Point(xmin, ymin), cv::Point(xmax, ymax),
                  cv::Scalar(0, 255, 0), 1, 1, 0);
}
auto out = image_file.substr(0, image_file.size() - 4) + "_out.jpg";
LOG(INFO) << "write result to " << out;
cv::imwrite(out, img);
```

Display of the refinedet model results:

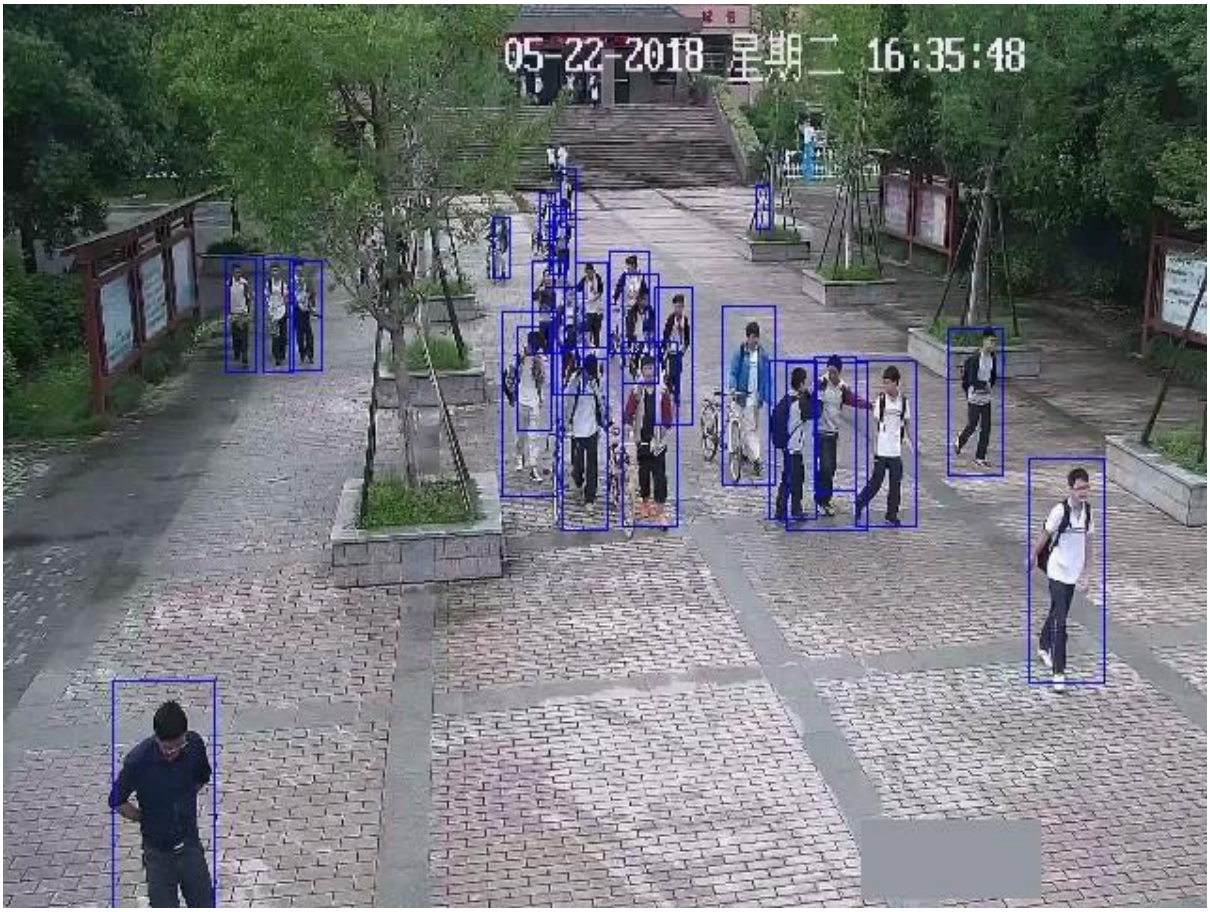


Figure 2.6: refinedet result image

2.23.2 Member Function Documentation

2.23.2.1 static std::unique_ptr<RefineDet> xilinx::ai::RefineDet::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class RefineDet.

Parameters

model_name	
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of RefineDet class.

2.23.2.2 virtual RefineDetResult xilinx::ai::RefineDet::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the RefineDet neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

A Struct of RefineDetResult.

2.23.2.3 virtual int xilinx::ai::RefineDet::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the refinedet network (input image cols).

Returns

InputWidth of the refinedet network

2.23.2.4 virtual int xilinx::ai::RefineDet::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the refinedet network (input image rows).

Returns

InputHeight of the refinedet network.

2.24 xilinx::ai::RefineDetPostProcess Class Reference

Class of the refinedet post-process, it will initialize the parameters once instead of compute them every time when the program execute.

`#include <xilinx/ai/nnpp/refinedet.hpp>`

Public Member Functions

- virtual RefineDetResult refine_det_post_process ()=0

  Run refinedet post-process.

Static Public Member Functions

- static std::unique_ptr < RefineDetPostProcess > create (const std::vector< std::vector< xilinx::ai::InputTensor >> &input_tensors, const std::vector< std::vector< xilinx::ai::OutputTensor >> &output_tensors, const xilinx::ai::proto::DpuModelParam &config)

  Create an RefineDetPostProcess object.

Protected Member Functions

- **RefineDetPostProcess** (const RefineDetPostProcess &)=delete
- RefineDetPostProcess & **operator=** (const RefineDetPostProcess &)=delete

2.24.1 Detailed Description

Class of the refinedet post-process, it will initialize the parameters once instead of compute them every time when the program execute.

2.24.2 Member Function Documentation

2.24.2.1 static std::unique_ptr<RefineDetPostProcess> xilinx::ai::RefineDetPostProcess::create (const std::vector< std::vector< xilinx::ai::InputTensor >> & *input_tensors*, const std::vector< std::vector< xilinx::ai::OutputTensor >> & *output_tensors*, const xilinx::ai::proto::DpuModelParam & *config*) [static]

Create an RefineDetPostProcess object.

Parameters

input_tensors	A vector of all input-tensors in the network. Usage: input_tensors[input_tensor_index].
output_tensors	A vector of all output-tensors in the network. Usage: output_tensors[output_index].
config	The dpu model configuration information.

Returns

An unique printer of RefineDetPostProcess.

2.24.2.2 virtual RefineDetResult xilinx::ai::RefineDetPostProcess::refine_det_post_process ( ) [pure virtual]

Run refinedet post-process.

Returns

The struct of RefineDetResult.

2.25 xilinx::ai::RefineDetResult Struct Reference

Struct of the result with the refinedet network.

`#include <xilinx/ai/nnpp/refinedet.hpp>`

Classes

- struct BoundingBox

  Struct of a object coordinate and confidence.

Public Attributes

- int width

  Width of the input image.
- int height

  Height of the input image.
- std::vector< BoundingBox > bboxes

  The vector of BoundingBox.

2.25.1 Detailed Description

Struct of the result with the refinedet network.

2.26 xilinx::ai::RefineDetResult::BoundingBox Struct Reference

Struct of a object coordinate and confidence.

`#include <xilinx/ai/nnpp/refinedet.hpp>`

Public Attributes

- float x

  x-coordinate , x is normalized relative to the input image cols ,the value range from 0 to 1.
- float y

  y-coordinate , y is normalized relative to the input image rows ,the value range from 0 to 1.
- float width

  body width , width is normalized relative to the input image cols , the value range from 0 to 1.
- float height

  body height , heigth is normalized relative to the input image rows , the value range from 0 to 1.
- float score

  body detection confidence, the value range from 0 to 1.

2.26.1 Detailed Description

Struct of a object coordinate and confidence.

2.27 xilinx::ai::Reid Class Reference

Base class for detecting roadline from a image (cv::Mat).

`#include <xilinx/ai/reid.hpp>`

Public Member Functions

- **Reid** (const Reid &)=delete
- virtual ReidResult run (const cv::Mat &image)=0

  Function of get running result of the reid neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the reid network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the reid network (input image rows).

Static Public Member Functions

- static std::unique_ptr< Reid > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Reid.

2.27.1 Detailed Description

Base class for detecting roadline from a image (cv::Mat).

Input is an image (cv::Mat).

Output road line type and points maked road line.

Note

The input image size is 640 x 480.

Sample code:

```
if(argc < 3){
    cerr<<"need two images"<<endl;
    return -1;
}
Mat imgx = imread(argv[1]);
if(imgx.empty()){
    cerr<<"can't load image! "<<argv[1]<<endl;
    return -1;
}
Mat imgy = imread(argv[2]);
if(imgy.empty()){
    cerr<<"can't load image! "<<argv[2]<<endl;
    return -1;
}
auto det = xilinx::ai::Reid::create(xilinx::ai::REID);
Mat featx = det->run(imgx).feat;
Mat featy = det->run(imgy).feat;
double dismat= cosine_distance(featx, featy);
printf("dismat : %.3lf ¥n", dismat);
```

2.27.2 Member Function Documentation

2.27.2.1 static std::unique_ptr<Reid> xilinx::ai::Reid::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class Reid.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Reid class.

2.27.2.2 virtual ReidResult xilinx::ai::Reid::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the reid neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

ReidResult.

2.27.2.3 virtual int xilinx::ai::Reid::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the reid network (input image cols).

Returns

InputWidth of the reid network

2.27.2.4 virtual int xilinx::ai::Reid::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the reid network (input image rows).

Returns

InputHeight of the reid network.

2.28 xilinx::ai::ReidResult Struct Reference

Result with the reid network.

`#include <xilinx/ai/nnpp/reid.hpp>`

Public Attributes

- int width
  Width of input image.
- int height
  Height of input image.
- cv::Mat feat
  The feature of input image.

2.28.1 Detailed Description

Result with the reid network.

2.29 xilinx::ai::RoadLine Class Reference

Base class for detecting roadline from a image (cv::Mat).

`#include <xilinx/ai/roadline.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0
  Function to get InputWidth of the roadline network (input image cols).
- virtual int getInputHeight () const =0
  Function to get InputHight of the roadline network (input image rows).
- virtual RoadLineResult run (const cv::Mat &image)=0
  Function of get running result of the RoadLine network.

Static Public Member Functions

- static std::unique_ptr< RoadLine > create (const std::string &model_name, bool need_preprocess=true)
  Factory function to get a instance of derived classes of class RoadLine.

Protected Member Functions

- **RoadLine** (const RoadLine &)=delete

2.29.1 Detailed Description

Base class for detecting roadline from a image (cv::Mat).

Input is a image (cv::Mat).

Output road line type and points maked road line.

Note

The input image size is 640x480

Sample code:

```
auto det = xilinx::ai::RoadLine::create(ROAD_LINE_DEEPHI);
auto image = cv::imread(argv[1]);
//    Mat image;
//    resize(img, image, Size(640, 480));
if (image.empty()) {
  cerr << "cannot load " << argv[1] << endl;
  abort();
}

vector<int> color1 = {0, 255, 0, 0, 100, 255};
vector<int> color2 = {0, 0, 255, 0, 100, 255};
vector<int> color3 = {0, 0, 0, 255, 100, 255};

RoadLineResult results = det->run(image);
for (auto &line : results.lines) {
  vector<Point> points_poly = line.points_cluster;
  // for (auto &p : points_poly) {
  //  std::cout << p.x << " " << (int)p.y << std::endl;
  //}
  int type = line.type < 5 ? line.type : 5;
  if (type == 2 && points_poly[0].x < image.rows * 0.5)
    continue;
  cv::polylines(image, points_poly, false,
                Scalar(color1[type], color2[type], color3[type]), 3, CV_AA,
                0);
}
```

Display of the roadline model results:

Figure 2.7: roadline result image

2.29.2 Member Function Documentation

2.29.2.1 static std::unique_ptr<RoadLine> xilinx::ai::RoadLine::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class RoadLine.

Parameters

model_name	String of model name
need_-preprocess	normalize with mean/scale or not, default value is true.

Returns

An instance of RoadLine class.

2.29.2.2 virtual int xilinx::ai::RoadLine::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the roadline network (input image cols).

Returns

InputWidth of the roadline network.

2.29.2.3 virtual int xilinx::ai::RoadLine::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the roadline network (input image rows).

Returns

InputHeight of the roadline network.

2.29.2.4 virtual RoadLineResult xilinx::ai::RoadLine::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the RoadLine network.

Parameters

image	Input data , input image (cv::Mat) need to resized as 640x480.

Returns

The struct of RoadLineResult

2.30 xilinx::ai::RoadLinePostProcess Class Reference

Class of the roadline post-process, it will initialize the parameters once instead of compute them every time when the program execute.

`#include <xilinx/ai/nnpp/roadline.hpp>`

Public Member Functions

- virtual RoadLineResult road_line_post_process (int inWidth, int inHetght)=0

  Run roadline post-process.

Static Public Member Functions

- static std::unique_ptr < RoadLinePostProcess > create (const std::vector< xilinx::ai::InputTensor > &input_tensors, const std-::vector< xilinx::ai::OutputTensor > &output_tensors, const xilinx::ai::proto::DpuModelParam &config)

  Create an RoadLinePostProcess object.

Protected Member Functions

- **RoadLinePostProcess** (const RoadLinePostProcess &)=delete
- RoadLinePostProcess & **operator=** (const RoadLinePostProcess &)=delete

2.30.1 Detailed Description

Class of the roadline post-process, it will initialize the parameters once instead of compute them every time when the program execute.

2.30.2 Member Function Documentation

2.30.2.1 static std::unique_ptr<RoadLinePostProcess> xilinx::ai::RoadLinePostProcess::create (const std::vector< xilinx::ai::InputTensor > & *input_tensors*, const std::vector< xilinx::ai::OutputTensor > & *output_tensors*, const xilinx::ai::proto::DpuModelParam & *config*) [static]

Create an RoadLinePostProcess object.

Parameters

input_tensors	A vector of all input-tensors in the network. Usage: input_tensors[input_tensor_index].
output_tensors	A vector of all output-tensors in the network. Usage: output_tensors[output_index].
config	The dpu model configuration information.

Returns

An unique printer of RoadLinePostProcess.

2.30.2.2 virtual RoadLineResult xilinx::ai::RoadLinePostProcess::road_line_post_process (int *inWidth*, int *inHetght*) [pure virtual]

Run roadline post-process.

Returns

The struct of RoadLineResult.

2.31 xilinx::ai::RoadLineResult Struct Reference

Struct of the result returned by the roadline network.

`#include <xilinx/ai/nnpp/roadline.hpp>`

Classes

- struct Line

  Struct of the result returned by the roadline network.

Public Attributes

- int width

  Width of input image.
- int height

Height of input image.

- std::vector< Line > lines

  the vector of line

2.31.1 Detailed Description

Struct of the result returned by the roadline network.

2.32 xilinx::ai::RoadLineResult::Line Struct Reference

Struct of the result returned by the roadline network.

`#include <xilinx/ai/nnpp/roadline.hpp>`

Public Attributes

- int type
- std::vector< cv::Point > points_cluster

  point clusters, make line from these.

2.32.1 Detailed Description

Struct of the result returned by the roadline network.

2.32.2 Member Data Documentation

2.32.2.1 int xilinx::ai::RoadLineResult::Line::type

road line type, the value range from 0 to 3.

- `0` : background
- `1` : white dotted line
- `2` : white solid line
- `3` : yollow line

2.33 xilinx::ai::Segmentation Class Reference

Base class for Segmentation.

`#include <xilinx/ai/segmentation.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the segmentation network (input image cols).

- virtual int getInputHeight () const =0

  Function to get InputHight of the segmentation network (input image rows).

- virtual SegmentationResult run_8UC1 (const cv::Mat &image)=0

Function of get running result of the segmentation network.

- virtual SegmentationResult run_8UC3 (const cv::Mat &image)=0

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation.

Protected Member Functions

- **Segmentation** (const Segmentation &)=delete

2.33.1 Detailed Description

Base class for Segmentation.

Input is an image (cv:Mat).

Output is result of running the Segmentation network.

Sample code :

```
   auto det
=xilinx::ai::Segmentation::create(xilinx::ai::SEGMENTATION_FPN, true);

   auto img= cv::imread("sample_segmentation.jpg");
   int width = det->getInputWidth();
   int height = det->getInputHeight();
   cv::Mat image;
   cv::resize(img, image, cv::Size(width, height), 0, 0,
              cv::INTER_LINEAR);
   auto result = det->run_8UC1(image);
   for (auto y = 0; y < result.segmentation.rows; y++) {
     for (auto x = 0; x < result.segmentation.cols; x++) {
           result.segmentation.at<uchar>(y,x) *= 10;
       }
   }
   cv::imwrite("segres.jpg",result.segmentation);

   auto resultshow = det->run_8UC3(image);
   resize(resultshow.segmentation, resultshow.segmentation,
cv::Size(resultshow.cols * 2, resultshow.rows * 2));
   cv::imwrite("sample_segmentation_visualization_result.jpg",resultshow.segmentation);
```

Figure 2.8: segmentation visulization result image

2.33.2 Member Function Documentation

2.33.2.1 static std::unique_ptr<Segmentation> xilinx::ai::Segmentation::create (const std::string & *model_name,* bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class Segmentation.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Segmentation class.

2.33.2.2 virtual int xilinx::ai::Segmentation::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

2.33.2.3 virtual int xilinx::ai::Segmentation::getInputHeight ( ) const [pure virtual]

Function to get InputHight of the segmentation network (input image rows).

Returns

InputHeight of the segmentation network.

2.33.2.4 virtual SegmentationResult xilinx::ai::Segmentation::run_8UC1 (const cv::Mat & *image*) [pure virtual]

Function of get running result of the segmentation network.

Note

The type of CV_8UC1 of the Reuslt's segmentation.

Parameters

image	Input data of input image (cv::Mat).

Returns

a result include segmentation output data.

2.33.2.5 virtual SegmentationResult xilinx::ai::Segmentation::run_8UC3 (const cv::Mat & *image*) [pure virtual]

Function of get running result of the segmentation network.

Note

The type of CV_8UC3 of the Reuslt's segmentation.

Parameters

image	Input data of input image (cv::Mat).

Returns

a result include segmentation image and shape;.

2.34 xilinx::ai::Segmentation8UC1 Class Reference

The Class of Segmentation8UC1, this class run function return a cv::Mat with the type is cv_8UC1 Sample code :

```
#include <xilinx/ai/segmentation.hpp>
```

Public Member Functions

- int getInputWidth () const

  Function to get InputWidth of the segmentation network (input image cols).
- int getInputHeight () const

  Function to get InputHight of the segmentation network (input image cols).
- SegmentationResult run (const cv::Mat &image)

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation8UC1 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation8UC1.

Protected Member Functions

- **Segmentation8UC1** (std::unique_ptr< Segmentation > segmentation)
- **Segmentation8UC1** (const Segmentation8UC1 &)=delete

2.34.1 Detailed Description

The Class of Segmentation8UC1, this class run function return a cv::Mat with the type is cv_8UC1 Sample code :

```
   auto det =
xilinx::ai::Segmentation8UC1::create(xilinx::ai::SEGMENTATION_FPN);
  auto img = cv::imread("sample_segmentation.jpg");
   int width = det->getInputWidth();
   int height = det->getInputHeight();
   cv::Mat image;
   cv::resize(img, image, cv::Size(width, height), 0, 0,
              cv::INTER_LINEAR);
   auto result = det->run(image);
   for (auto y = 0; y < result.segmentation.rows; y++) {
     for (auto x = 0; x < result.segmentation.cols; x++) {
           result.segmentation.at<uchar>(y,x) *= 10;
       }
   }
   cv::imwrite("segres.jpg",result.segmentation);
```

2.34.2 Member Function Documentation

2.34.2.1 static std::unique_ptr<Segmentation8UC1> xilinx::ai::Segmentation8UC1::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class Segmentation8UC1.

Parameters

model_name	Model name
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Segmentation8UC1 class.

2.34.2.2 int xilinx::ai::Segmentation8UC1::getInputWidth ( ) const

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

2.34.2.3 int xilinx::ai::Segmentation8UC1::getInputHeight ( ) const

Function to get InputHight of the segmentation network (input image cols).

Returns

InputHeight of the segmentation network.

2.34.2.4 SegmentationResult xilinx::ai::Segmentation8UC1::run (const cv::Mat & *image*)

Function of get running result of the segmentation network.

Note

The result cv::Mat of the type is CV_8UC1.

Parameters

image	Input data of the image (cv::Mat)

Returns

A Struct of SegmentationResult ,the result of segmentation network.

2.35 xilinx::ai::Segmentation8UC3 Class Reference

The Class of Segmentation8UC3, this class run function return a cv::Mat with the type is cv_8UC3 Sample code :

```
#include <xilinx/ai/segmentation.hpp>
```

Public Member Functions

- int getInputWidth () const

  Function to get InputWidth of the segmentation network (input image cols).
- int getInputHeight () const

  Function to get InputWidth of the segmentation network (input image cols).
- SegmentationResult run (const cv::Mat &image)

  Function of get running result of the segmentation network.

Static Public Member Functions

- static std::unique_ptr
  < Segmentation8UC3 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class Segmentation8UC3.

Protected Member Functions

- **Segmentation8UC3** (std::unique_ptr< Segmentation > segmentation)
- **Segmentation8UC3** (const Segmentation8UC3 &)=delete

2.35.1 Detailed Description

The Class of Segmentation8UC3, this class run function return a cv::Mat with the type is cv_8UC3 Sample code :

```
   auto det =
xilinx::ai::Segmentation8UC3::create(xilinx::ai::SEGMENTATION_FPN);
  auto img = cv::imread("sample_segmentation.jpg");

   int width = det->getInputWidth();
   int height = det->getInputHeight();
   cv::Mat image;
   cv::resize(img, image, cv::Size(width, height), 0, 0,
              cv::INTER_LINEAR);
   auto result = det->run(image);
   cv::imwrite("segres.jpg",result.segmentation);
```

2.35.2 Member Function Documentation

2.35.2.1 static std::unique_ptr<Segmentation8UC3> xilinx::ai::Segmentation8UC3::create (const std::string & *model_name*, bool *need_preprocess* = `true`) [static]

Factory function to get a instance of derived classes of class Segmentation8UC3.

Parameters

model_name	Model name
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of Segmentation8UC3 class.

2.35.2.2 int xilinx::ai::Segmentation8UC3::getInputWidth ( ) const

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

2.35.2.3 int xilinx::ai::Segmentation8UC3::getInputHeight ( ) const

Function to get InputWidth of the segmentation network (input image cols).

Returns

InputWidth of the segmentation network.

2.35.2.4 SegmentationResult xilinx::ai::Segmentation8UC3::run (const cv::Mat & *image*)

Function of get running result of the segmentation network.

Note

The result cv::Mat of the type is CV_8UC1.

Parameters

image	Input data of the image (cv::Mat)

Returns

SegmentationResult The result of segmentation network.

2.36 xilinx::ai::SegmentationResult Struct Reference

Struct of the result returned by the segementation network.

`#include <xilinx/ai/nnpp/segmentation.hpp>`

Public Attributes

- int **width**
- int **height**
- cv::Mat **segmentation**

2.36.1 Detailed Description

Struct of the result returned by the segementation network.

FPN Num of segmentation classes

- `0` : "unlabeled"
- `1` : "ego vehicle"
- `2` : "rectification border"
- `3` : "out of roi"
- `4` : "static"
- `5` : "dynamic"
- `6` : "ground"
- `7` : "road"
- `8` : "sidewalk"
- `9` : "parking"
- `10` : "rail track"
- `11` : "building"
- `12` : "wall"
- `13` : "fence"

- 14 : "guard rail"
- 15 : "bridge"
- 16 : "tunnel"
- 17 : "pole"
- 18 : "polegroup"

2.37 xilinx::ai::SSD Class Reference

Base class for detecting position of vehicle,pedestrian and so on.

`#include <xilinx/ai/ssd.hpp>`

Public Member Functions

- virtual xilinx::ai::SSDResult run (const cv::Mat &img)=0

  Function of get result of the ssd neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the SSD network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the SSD network (input image rows).

Static Public Member Functions

- static std::unique_ptr< SSD > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class SSD.

Protected Member Functions

- **SSD** (const SSD &)=delete

2.37.1 Detailed Description

Base class for detecting position of vehicle,pedestrian and so on.

Input is an image (cv:Mat).

Output is a struct of detection results, named SSDResult.

Sample code :

```
  Mat img = cv::imread("sample_ssd_TRAFFIC_480x360.jpg");
  auto ssd = xilinx::ai::SSD::create(xilinx::ai::SSD_TRAFFIC_480x360,true);
  auto results = ssd->run(img);
  for(const auto &r : results.bboxes){
     auto label = r.label;
     auto x = r.x * img.cols;
     auto y = r.y * img.rows;
     auto width = r.width * img.cols;
     auto heigth = r.height * img.rows;
     auto score = r.score;
     std::cout << "RESULT: " << label << "¥t" << x << "¥t" << y << "¥t" <<
width
        << "¥t" << height << "¥t" << score << std::endl;
  }
```

Display of the ssd_TRAFFIC_480x360 model results:

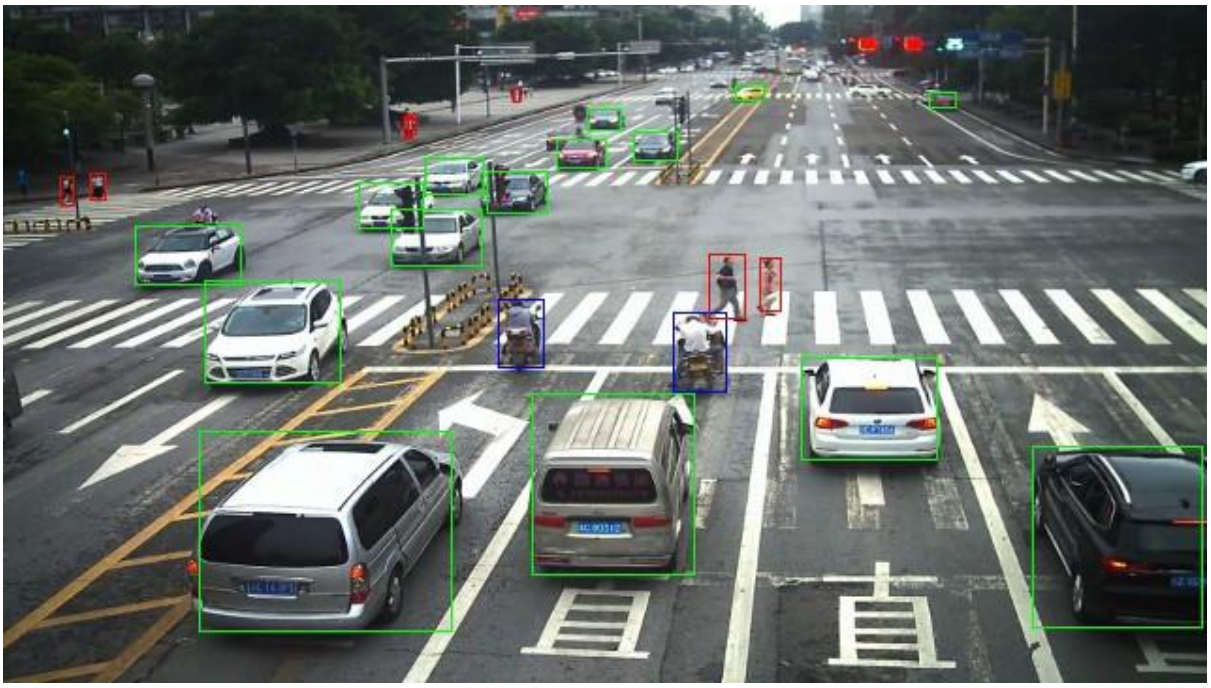


Figure 2.9: ssd_TRAFFIC_480x360 detection result

Display of the ADAS_VEHICLE_V3_480x360 model results:

Figure 2.10: ssd_ADAS_VEHICLE_V3_480x360 detection result

2.37.2 Member Function Documentation

2.37.2.1 static std::unique_ptr<SSD> xilinx::ai::SSD::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class SSD.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of SSD class.

2.37.2.2 virtual xilinx::ai::SSDResult xilinx::ai::SSD::run (const cv::Mat & *img*) [pure virtual]

Function of get result of the ssd neuron network.

Parameters

img	Input data of input image (cv::Mat).

Returns

SSDResult.

2.37.2.3 virtual int xilinx::ai::SSD::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the SSD network (input image cols).

Returns

InputWidth of the SSD network.

2.37.2.4 virtual int xilinx::ai::SSD::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the SSD network (input image rows).

Returns

InputHeight of the SSD network.

2.38 xilinx::ai::SSDPostProcess Class Reference

Class of the ssd post-process, it will initialize the parameters once instead of compute them every time when the program execute.

`#include <xilinx/ai/nnpp/ssd.hpp>`

Public Member Functions

- virtual SSDResult ssd_post_process ()=0

  The post-processing of function of the ssd network.

Static Public Member Functions

- static std::unique_ptr < SSDPostProcess > create (const std::vector < xilinx::ai::InputTensor > &input_tensors, const std::vector < xilinx::ai::OutputTensor > &output_tensors, const xilinx::ai::proto::DpuModelParam &config)

  Create an SSDPostProcess object.

Protected Member Functions

- **SSDPostProcess** (const SSDPostProcess &)=delete
- SSDPostProcess & **operator=** (const SSDPostProcess &)=delete

2.38.1 Detailed Description

Class of the ssd post-process, it will initialize the parameters once instead of compute them every time when the program execute.

2.38.2 Member Function Documentation

2.38.2.1 static std::unique_ptr<SSDPostProcess> xilinx::ai::SSDPostProcess::create (const std::vector< xilinx::ai::InputTensor > & *input_tensors*, const std::vector< xilinx::ai::OutputTensor > & *output_tensors*, const xilinx::ai::proto::DpuModelParam & *config*) [static]

Create an SSDPostProcess object.

Parameters

input_tensors	A vector of all input-tensors in the network. Usage: input_tensors[input_tensor_index].
output_tensors	A vector of all output-tensors in the network. Usage: output_tensors[output_index].
config	The dpu model configuration information.

Returns

An unique printer of SSDPostProcess.

2.38.2.2 virtual SSDResult xilinx::ai::SSDPostProcess::ssd_post_process ( ) [pure virtual]

The post-processing of function of the ssd network.

Returns

The struct of SSDResult.

2.39 xilinx::ai::SSDResult Struct Reference

Struct of the result returned by the ssd neuron network.

`#include <xilinx/ai/nnpp/ssd.hpp>`

Classes

- struct BoundingBox

  Struct of an object coordinate ,confidence and classification.

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- std::vector< BoundingBox > bboxes

  All objects, a vector of BoundingBox.

2.39.1 Detailed Description

Struct of the result returned by the ssd neuron network.

2.40 xilinx::ai::SSDResult::BoundingBox StructReference

Struct of an object coordinate ,confidence and classification.

`#include <xilinx/ai/nnpp/ssd.hpp>`

Public Attributes

- int label

  Classification.
- float score

  Confidence.
- float x

  x-coordinate, x is normalized relative to the input image cols ,the value range from 0 to 1.
- float y

  y-coordinate ,y is normalized relative to the input image rows ,the value range from 0 to 1.
- float width

  width, width is normalized relative to the input image cols ,the value range from 0 to 1.
- float height

  height, height is normalized relative to the input image rows ,the value range from 0 to 1.

2.40.1 Detailed Description

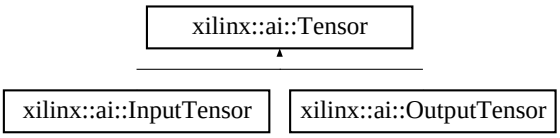
Struct of an object coordinate ,confidence and classification.

2.41 xilinx::ai::Tensor StructReference

The basic abstract structure of neural network layer.

`#include <xilinx/ai/tensor.hpp>`

Inheritance diagram for xilinx::ai::Tensor:



Public Attributes

- uintptr_t logic_addr

  The logic address of this Tensor.
- size_t size

  The total size of this tensor's data.
- size_t height

  The height of this tensor.
- size_t width

  The width of this tensor.
- size_t channel

  The channel of this tensor.

- int fixpos

  The fixed position of this tensor, the value range from 0 to 7.
- DataType dtype

  This tensor's data type.
- std::string name

  name for debug purpose

2.41.1 Detailed Description

The basic abstract structure of neural network layer.

2.42 xilinx::ai::VehicleResult StructReference

A struct to define detection result of MultiTask.

`#include <xilinx/ai/nnpp/multitask.hpp>`

Public Attributes

- int label
- float score

  confidence of this target
- float x

  x-coordinate, x is normalized relative to the input image cols ,the value range from 0 to 1.
- float y

  y-coordinate ,y is normalized relative to the input image rows ,the value range from 0 to 1.
- float width

  width, width is normalized relative to the input image cols ,the value range from 0 to 1.
- float height

  height, height is normalized relative to the input image rows ,the value range from 0 to 1.
- float angle

  the angle between the target vehicle and ourself.

2.42.1 Detailed Description

A struct to define detection result of MultiTask.

2.42.2 Member Data Documentation

2.42.2.1 int xilinx::ai::VehicleResult::label

number of classes

- `0` : "background"
- `1` : "person"
- `2` : "car"
- `3` : "truck"
- `4` : "bus"

- 5 : "bike"
- 6 : "sign"
- 7 : "light"

2.43 xilinx::ai::YOLOv2 Class Reference

Base class for detecting objects in the input image(cv::Mat). Input is an image(cv::Mat). Output is position of the objects in the input image. Sample code:

```
#include <xilinx/ai/yolov2.hpp>
```

Public Member Functions

- **YOLOv2** (const YOLOv2 &)=delete
- virtual YOLOv2Result run (const cv::Mat &image)=0

  Function of get running result of the YOLOv2 neuron network.
- virtual int getInputWidth () const =0

  Function to get InputWidth of the YOLOv2 network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the YOLOv2 network (input image rows).

Static Public Member Functions

- static std::unique_ptr< YOLOv2 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class YOLOv2.

2.43.1 Detailed Description

Base class for detecting objects in the input image(cv::Mat). Input is an image(cv::Mat). Output is position of the objects in the input image. Sample code:

```
 auto img = cv::imread("test_image.jpg");
 auto model = xilinx::ai::YOLOv2::create(xilinx::ai::YOLOV2_VOC_BASELINE);
 auto result = model->run(img);
 for (const auto &bbox : result.bboxes) {
   int label = bbox.label;
   float xmin = bbox.x * img.cols + 1;
   float ymin = bbox.y * img.rows + 1;
   float xmax = xmin + bbox.width * img.cols;
   float ymax = ymin + bbox.height * img.rows;
   if (xmax > img.cols)
     xmax = img.cols;
   if (ymax > img.rows)
     ymax = img.rows;
   float confidence = bbox.score;

   cout << "RESULT: " << label << "¥t" << xmin << "¥t" << ymin << "¥t" << xmax
       << "¥t" << ymax << "¥t" << confidence << "¥n";
   rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(0, 255, 0), 1,
             1, 0);
}
```

2.43.2 Member Function Documentation

2.43.2.1 static std::unique_ptr<YOLOv2> xilinx::ai::YOLOv2::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class YOLOv2.

Parameters

model_name	Model name
need_-preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of YOLOv2 class.

2.43.2.2 virtual YOLOv2Result xilinx::ai::YOLOv2::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the YOLOv2 neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

A Struct of YOLOv2Result.

2.43.2.3 virtual int xilinx::ai::YOLOv2::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the YOLOv2 network (input image cols).

Returns

InputWidth of the YOLOv2 network

2.43.2.4 virtual int xilinx::ai::YOLOv2::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the YOLOv2 network (input image rows).

Returns

InputHeight of the YOLOv2 network.

2.44 xilinx::ai::YOLOv2Result Struct Reference

```
#include <xilinx/ai/nnpp/yolov2.hpp>
```

Classes

- struct BoundingBox

  Struct of an object coordinate ,confidence and classification.

Public Attributes

- int width

  Width of input image.
- int height

  Height of input image.
- std::vector< BoundingBox > bboxes

  All objects.

2.44.1 Detailed Description

Struct of the result returned by the yolov2 network.

2.45 xilinx::ai::YOLOv2Result::BoundingBox Struct Reference

Struct of an object coordinate ,confidence and classification.

`#include <xilinx/ai/nnpp/yolov2.hpp>`

Public Attributes

- int label

  classification.
- float score

  confidence, the range from 0 to 1.
- float x

  x-coordinate, x is normalized relative to the input image cols, its value range from 0 to 1.
- float y

  y-coordinate, y is normalized relative to the input image rows, its value range from 0 to 1.
- float width

  width, width is normalized relative to the input image cols, its value from 0 to 1.
- float height

  height, height is normalized relative to the input image rows, its value range from 0 to 1.

2.45.1 Detailed Description

Struct of an object coordinate ,confidence and classification.

2.46 xilinx::ai::YOLOv3 Class Reference

Base class for detecting objects in the input image (cv::Mat).

`#include <xilinx/ai/yolov3.hpp>`

Public Member Functions

- virtual int getInputWidth () const =0

  Function to get InputWidth of the YOLOv3 network (input image cols).
- virtual int getInputHeight () const =0

  Function to get InputHeigth of the YOLOv3 network (input image rows).
- virtual YOLOv3Result run (const cv::Mat &image)=0

  Function of get running result of the YOLOv3 neuron network.

Static Public Member Functions

- static std::unique_ptr< YOLOv3 > create (const std::string &model_name, bool need_preprocess=true)

  Factory function to get a instance of derived classes of class YOLOv3.

Protected Member Functions

- **YOLOv3** (const YOLOv3 &)=delete

2.46.1 Detailed Description

Base class for detecting objects in the input image (cv::Mat).

Input is an image (cv::Mat).

Output is position of the pedestrians in the input image.

Sample code:

```
   auto yolo =
xilinx::ai::YOLOv3::create(xilinx::ai::YOLOV3_ADAS_512x256, true);
   Mat img = cv::imread("test.jpg");

   auto results = yolo->run(img);

   for(auto &box : results.bboxes){
     int label = box.label;
     float xmin = box.x * img.cols + 1;
     float ymin = box.y * img.rows + 1;
     float xmax = xmin + box.width * img.cols;
     float ymax = ymin + box.height * img.rows;
     if(xmin < 0.) xmin = 1.;
     if(ymin < 0.) ymin = 1.;
     if(xmax > img.cols) xmax = img.cols;
     if(ymax > img.rows) ymax = img.rows;
     float confidence = box.score;

     cout << "RESULT: " << label << "¥t" << xmin << "¥t" << ymin << "¥t"
          << xmax << "¥t" << ymax << "¥t" << confidence << "¥n";
     if (label == 0) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(0, 255, 0),
                 1, 1, 0);
     } else if (label == 1) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(255, 0, 0),
                 1, 1, 0);
     } else if (label == 2) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax), Scalar(0, 0, 255),
                 1, 1, 0);
     } else if (label == 3) {
       rectangle(img, Point(xmin, ymin), Point(xmax, ymax),
                 Scalar(0, 255, 255), 1, 1, 0);
     }

   }
   imwrite("result.jpg", img);
```

Display of the yolov3_ADAS_512x256 model results:

Figure 2.11: out image

2.46.2 Member Function Documentation

2.46.2.1 static std::unique_ptr<YOLOv3> xilinx::ai::YOLOv3::create (const std::string & *model_name*, bool *need_preprocess* = true) [static]

Factory function to get a instance of derived classes of class YOLOv3.

Parameters

model_name	Model name
need_preprocess	Normalize with mean/scale or not, default value is true.

Returns

An instance of YOLOv3 class.

2.46.2.2 virtual int xilinx::ai::YOLOv3::getInputWidth ( ) const [pure virtual]

Function to get InputWidth of the YOLOv3 network (input image cols).

Returns

InputWidth of the YOLOv3 network

2.46.2.3 virtual int xilinx::ai::YOLOv3::getInputHeight ( ) const [pure virtual]

Function to get InputHeigth of the YOLOv3 network (input image rows).

Returns

InputHeight of the YOLOv3 network.

2.46.2.4 virtual YOLOv3Result xilinx::ai::YOLOv3::run (const cv::Mat & *image*) [pure virtual]

Function of get running result of the YOLOv3 neuron network.

Parameters

image	Input data of input image (cv::Mat).

Returns

YOLOv3Result.

2.47 xilinx::ai::YOLOv3Result Struct Reference

Struct of the result returned by the yolov3 neuron network.

`#include <xilinx/ai/nnpp/yolov3.hpp>`

Classes

- struct BoundingBox

Public Attributes

- int width

  Width of input image.
- int height

  Height of output image.
- std::vector< BoundingBox > bboxes

  All objects, The vector of BoundingBox .

2.47.1 Detailed Description

Struct of the result returned by the yolov3 neuron network.

Note

VOC dataset category:string label[20] = {"aeroplane", "bicycle", "bird", "boat", "bottle", "bus","car", "cat", "chair", "cow", "diningtable", "dog", "horse", "motorbike","person", "pottedplant", "sheep", "sofa", "train", "tv-monitor"};
ADAS dataset category : string label[3] = {"car", "person", "cycle"};

2.48 xilinx::ai::YOLOv3Result::BoundingBox Struct Reference

`#include <xilinx/ai/nnpp/yolov3.hpp>`

Public Attributes

- int label

  classification.
- float score

  confidence, the range from 0 to 1.
- float x

  x-coordinate, x is normalized relative to the input image cols, its value range from 0 to 1.
- float y

y-coordinate, y is normalized relative to the input image rows, its value range from 0 to 1.

- float width

  width, width is normalized relative to the input image cols, its value from 0 to 1.

- float height

  height, height is normalized relative to the input image rows, its value range from 0 to 1.

2.48.1 Detailed Description

Struct of detection result with a object