AIX2024 SDK Installation Guide

: Compile & Run

(For Windows)

V1.0

서울대학교 차세대반도체 혁신공유대학

Xuan-Truong Nguyen (응웬트렁)

truongnx@snu.ac.kr

1. Overview

This AIX2024 SDK consists of C/C++ deep learning applications and frameworks based on the darknet framework [1]. This document demonstrates how to install the AIX2024 SDK and associated packages and run applications.

1.1. Directory structure (Next page)

The AIX2024 SDK package is “skeleton” that includes the third-party libraries, source codes, executable folders with the test datasets, weights, and executable scripts, Makefile, and the Visual Studio project.

skeleton

- **3rdparty : Third party library (Don't touch)**
  - lib/ Library
  - include/ Header files
  - dll/ Dynamic Link Library
  - CLBlast/ A modern, lightweight, performant and tunable OpenCL BLAS library implements BLAS routines: *basic linear algebra subprograms* (BLAS) operating on vectors and matrices.
- **src : Source code**
  - main.c //Main file
  - yolov2_forward_network.c // Do inference for an **FP32** model
  - yolov2_forward_network_quantized.c // Do inference for an i**nt8 quantized** model
  - additionally.c // All functions and utilities
  - …
- **bin : Executable files and bash scripts**
  - dataset/ test images and labels, make_list_cur.py, show_images.py, size_search.py, target.txt
  - weights/ output files when saving the model's parameters layer by layer
  - *.weights, *.cfg Files to save the parameters, and the structure of the model AIX2024
  - … Scripts for Unix/Linux (*.sh), scripts for Windows (*.cmd), executable files
- **obj : Object files when compiled on Unix/Linux (Don't care)**
- **Makefile : Makefile**
- ***.sln, *.cv* : Visual studio project**

2. Toolchains for AIX2024 SDK Installation

This chapter walks you through how to set up a development environment, install the AIX2024 SDK, and execute applications on the AIX2024 framework. The following prerequisites and requirements must be satisfied before installing the AIX2024 SDK. We will call these AIX2024 Toolchains. This chapter aims to teach you how to install pre-installed packages for the AIX2024.

The AIX2024 SDK has been tested on the following version of Windows 10, Visual Studio 2019, and Python 2.7.

- **Windows 10**
- **Python 2.7**
- **Visual Studio 2019**

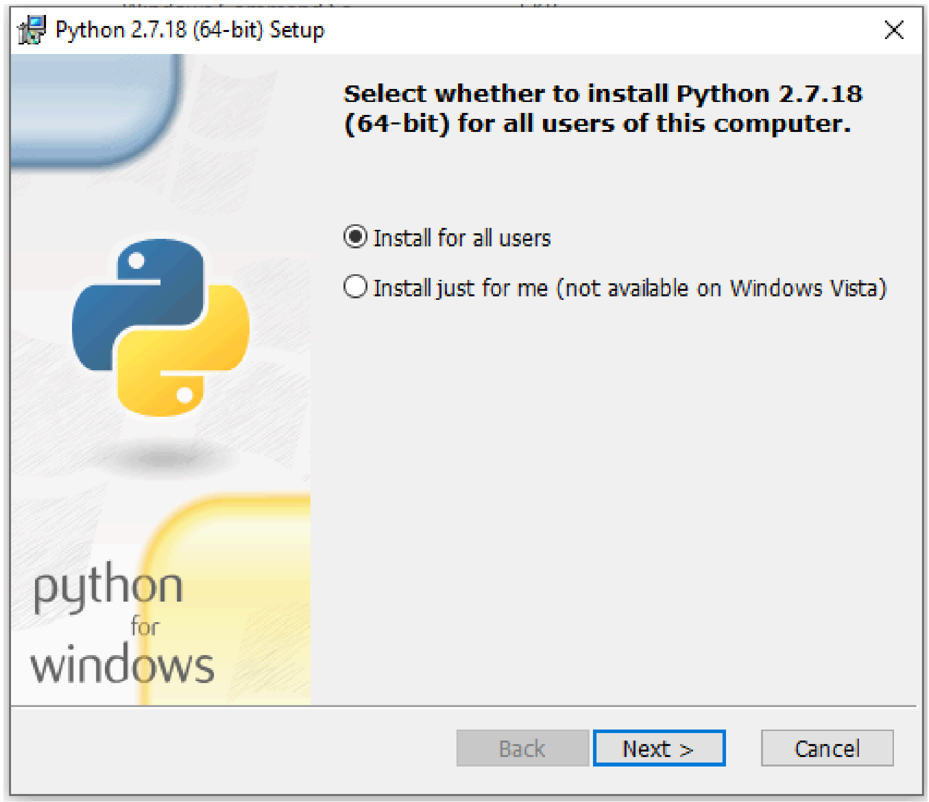
[Install required package]

- **Install Python on Windows:**
  1) You can go to https://www.python.org/downloads/release/python-2718/ to download Python2.7. The version is Windows x86-64 MSI installer

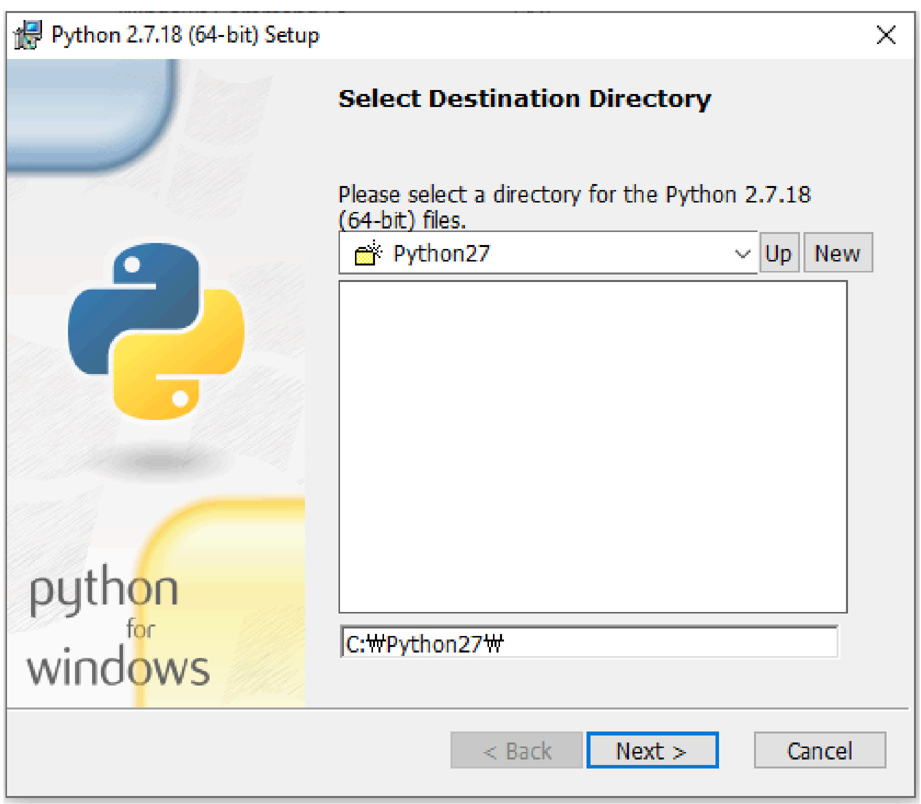
Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		38c84292658ed4456157195f1c9bcbe1	17539408	SIG
XZ compressed source tarball	Source release		fd6cc8ec0a78c44036f825e739f36e5a	12854736	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	ce98eeb7bdf806685adc265ec1444463	24889285	SIG
Windows debug information files	Windows		20b111ccfe8d06d2fe8c77679a86113d	25178278	SIG
Windows debug information files for 64-bit binaries	Windows		bb0897ea20fda343e5179d413d4a4a7c	26005670	SIG
Windows help file	Windows		b3b753dffe1c7930243c1c40ec3a72b1	6322188	SIG
Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64	a425c758d38f8e28b56f4724b499239a	20598784	SIG
Windows x86 MSI installer	Windows		db6ad9195b3086c6b4cefb9493d738d2	19632128	SIG

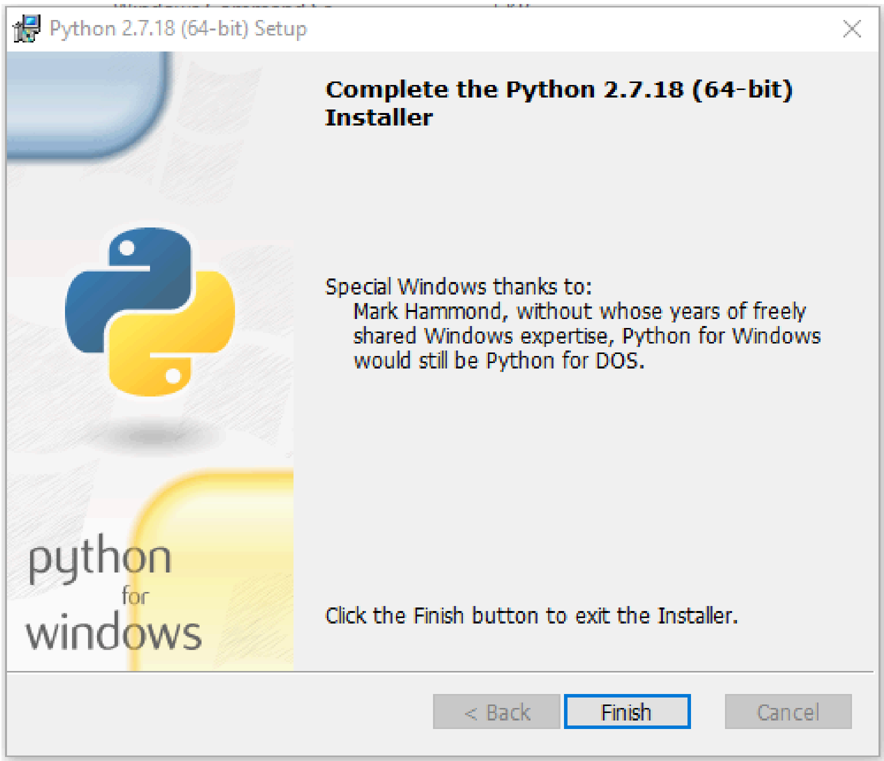
  2) Double-click on the downloaded file “python-2.7.18.amd64.msi” and click Next



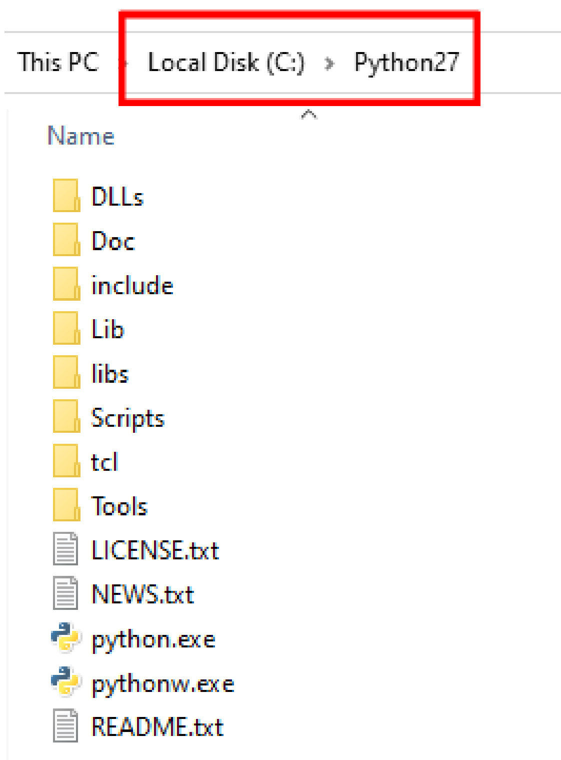
3) Select a directory for Python and click Next. The default setting is C:\Pthon27\



4) Click Next → Next to install the package

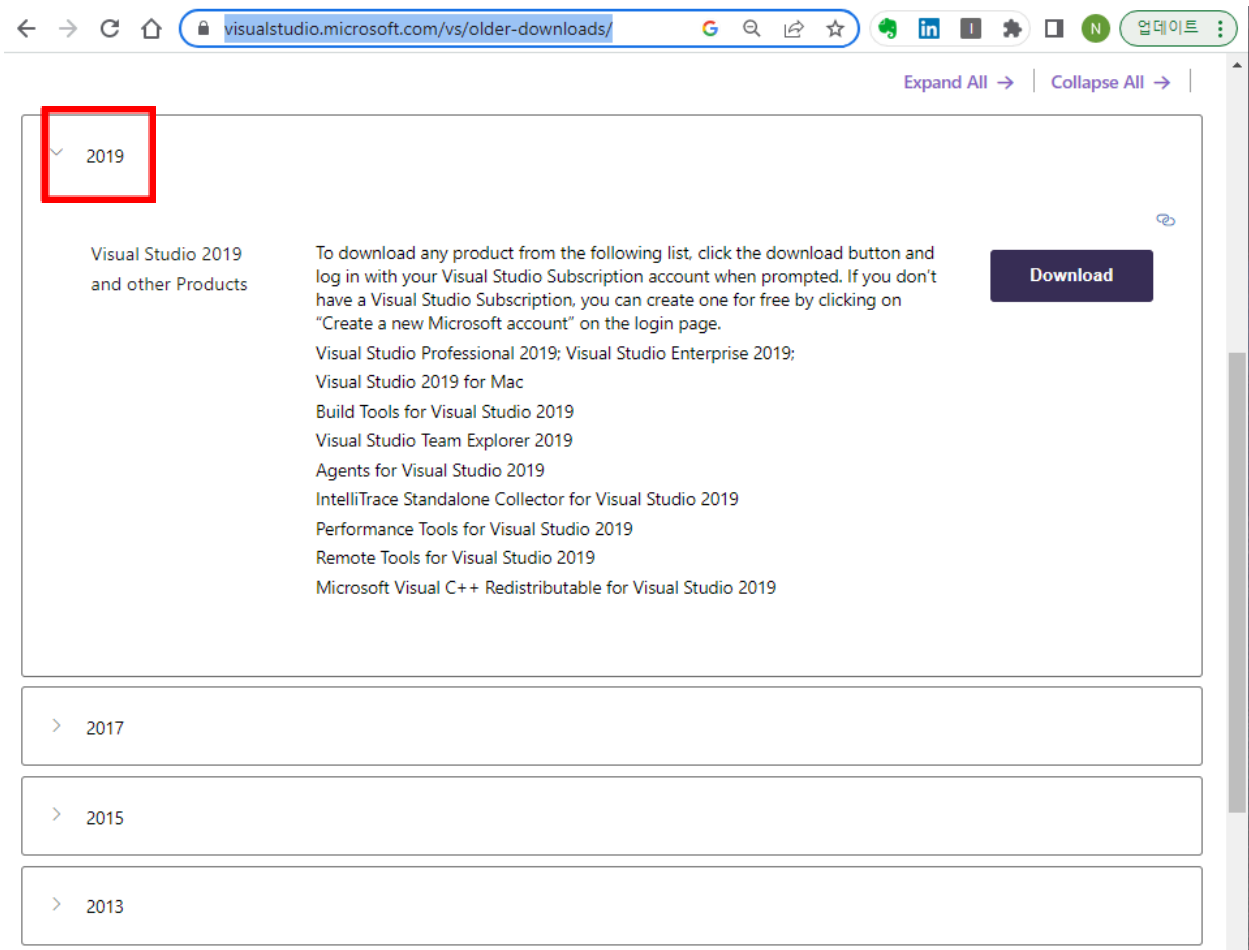


5) Finally, click Finish. Now, you can see C:/Python27



- **Install Visual Studio C/C++ on Windows**

1) Go to https://visualstudio.microsoft.com/vs/older-downloads/ to select version



2) Download and install Visual Studio 2019.

3) Go to Start > Visual Studio to check if the VS is installed.

3. AIX2024 SDK Installation Guide

This chapter elaborates on how to compile the AIX2024 SDK and run a model. Before going to this chapter, you must check Chapter 2 for the required packages. The flow consists of the following steps

1) *Generate the directories for the test images (Python)*
2) *Compile the code*
3) *Run the model(with Command Prompt or with Visual Studio projcet)*
4) *Verify the expected outputs*

Assume that the AIX2024 code is unzipped and stored in your Windows PC. For example, **the AIX2024 is located at C:\skeleton**. Before going to this step, make sure that Python and Visual Studio are installed on your Windows PC.

1) Generate the directories for the test images.

Note that this step is done only ONE time when you save the AIX2024 framework in your local directory. Go to start, type cmd, and click Enter to open Command Prompt. By default, the directory is C:\Users\User. Execute the following command line to access the folder “C:\skeleton”:

```
> cd ..\..\skeleton
```

After running those commands, you are supposed to see:

```
Command Prompt
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

C:₩Users₩User>cd ..₩..₩skeleton

C:₩skeleton>
```

Now, you can go to bin\dataset to generate the directories of the test images with your local folder. Execute the following commands:

```
> cd bin\dataset
> C:\skeleton\bin\dataset>python.exe make_list_cur.py
```

After Enter, you are supposed to see the screen on Command Prompt. In addition, if you open C:/skeleton/bin/dataset/target.txt, you will see that the file(target.txt) stores all directories of the test images with your local directory, for example, “C:/skeleton/bin/dataset”.

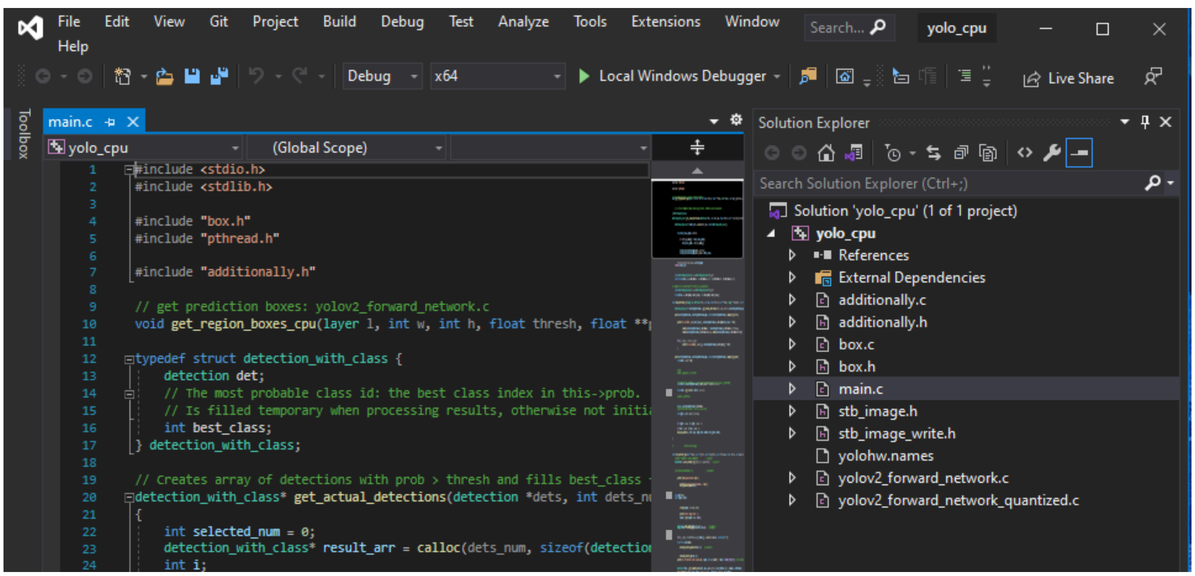
```
CAPP_testset_long_10012
CAPP_testset_long_10099
CAPP_testset_close_10120
CAPP_testset_close_10115
CAPP_testset_close_10054
CAPP_testset_close_10071
CAPP_testset_long_10039
CAPP_testset_close_10024
CAPP_testset_long_10001
CAPP_testset_long_10054

C:₩skeleton₩bin₩dataset>
```

target.txt

```
C:/skeleton/bin/dataset/CAPP_testset_close_10014.jpg
C:/skeleton/bin/dataset/CAPP_testset_long_10012.jpg
C:/skeleton/bin/dataset/CAPP_testset_long_10099.jpg
C:/skeleton/bin/dataset/CAPP_testset_close_10120.jpg
C:/skeleton/bin/dataset/CAPP_testset_close_10115.jpg
C:/skeleton/bin/dataset/CAPP_testset_close_10054.jpg
C:/skeleton/bin/dataset/CAPP_testset_close_10071.jpg
C:/skeleton/bin/dataset/CAPP_testset_long_10039.jpg
C:/skeleton/bin/dataset/CAPP_testset_close_10024.jpg
C:/skeleton/bin/dataset/CAPP_testset_long_10001.jpg
C:/skeleton/bin/dataset/CAPP_testset_long_10054.jpg
```

2) Open the visual studio project and compile the AIX2024 code

Now, you can go to C:\skeleton and double-click on **yolo_cpu.sln to** open the VS project for the AIX2024.

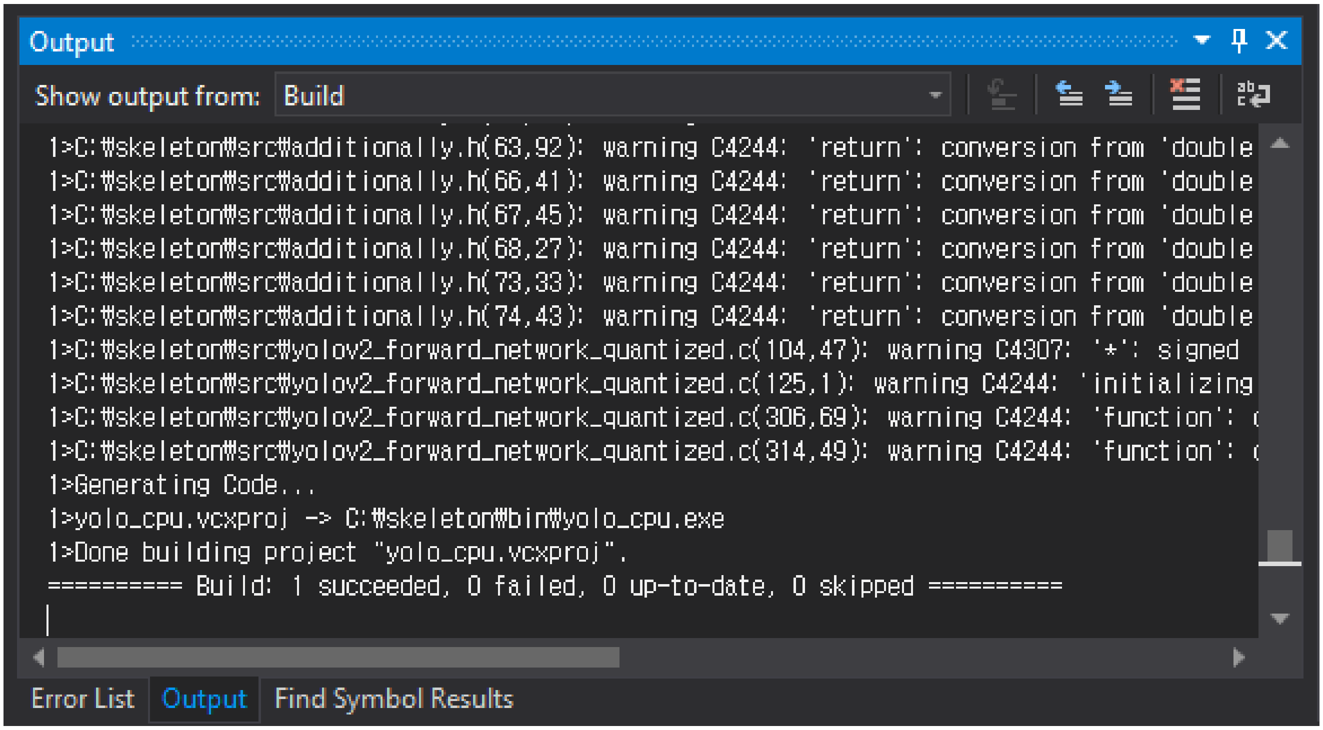


- Compile the VS project

**NOTE: Before compiling the code, you must open "additionally.c" and change the lines 3716 and 3718 as follows to locate the directories for target.txt and yolohw.names.

```
3714        char *name_list = option_find_str(options, "names", "yolohw.names");
3715    #else          // 2022.02.19: Fixed the directories for debugging the code on Windows
3716        char* valid_images = option_find_str(options, "valid", "C:₩₩skeleton₩₩bin₩₩dataset₩₩target.txt");
3717        char* difficult_valid_images = option_find_str(options, "difficult", NULL);
3718        char* name_list = option_find_str(options, "names", "C:₩₩skeleton₩₩bin₩₩yolohw.names");
3719    #endif
3720        char **names = get_labels(name_list);
```

- Click Build → Clean solution.
- Now, click Build → Build Solution, and you can see the output window as follows. The executable file **yolo_cpu.exe** is generated and stored in **C:\skeleton\bin.**



3) **Run the model AIX2024 with Command Prompt**

Now, let's go back to Command Prompt. **Make sure that you are at C:\skeleton\bin before executing the script.** If you are at C:\skeleton\bin\dataset, you can type the command to go back to C:\skeleton\bin

```
> cd ..
```

- **Before executing the script files, you must check if the script files are as follows**:
    - File **bin\script-wins-aix2024-test-all.cmd**

    yolo_cpu.exe detector map yolohw.names aix2024.cfg aix2024.weights -thresh 0.24

    - File **bin\script-wins-aix2024-test-all-quantized.cmd**

    yolo_cpu.exe detector map yolohw.names aix2024.cfg aix2024.weights -thresh 0.24 -quantized

Now, you can run the script with Command Prompt to test the model.

- Run the **full-precision** model and calculate the mAP by the command:

```
> script-wins-aix2024-test-all.cmd
```

This command uses the default “C:\skeleton\bin\dataset\target.txt” generated at Step 1 and the name list of 60 product items stored in the file “C:\skeleton\bin\yolohw.names”. Next, it loads the model architecture from aix2024.cfg and then loads the parameters from aix2024.weights.

```
C:\skeleton\bin>yolo_cpu.exe detector map yolohw.names aix2024.cfg aix2024.weigh
ts -thresh 0.24
valid: Using default 'C:/skeleton/bin/dataset/target.txt'
names: Using default 'C:/skeleton/bin/yolohw.names'
layer     filters    size              input                output
    0 conv     16  3 x 3 / 1   256 x 256 x   3   ->   256 x 256 x  16 0.057 BF
    1 max          2 x 2 / 2   256 x 256 x  16   ->   128 x 128 x  16
    2 conv     32  3 x 3 / 1   128 x 128 x  16   ->   128 x 128 x  32 0.151 BF
    3 max          2 x 2 / 2   128 x 128 x  32   ->    64 x  64 x  32
    4 conv     64  3 x 3 / 1    64 x  64 x  32   ->    64 x  64 x  64 0.151 BF
    5 max          2 x 2 / 2    64 x  64 x  64   ->    32 x  32 x  64
    6 conv    128  3 x 3 / 1    32 x  32 x  64   ->    32 x  32 x 128 0.151 BF
    7 max          2 x 2 / 2    32 x  32 x 128   ->    16 x  16 x 128
    8 conv    256  3 x 3 / 1    16 x  16 x 128   ->    16 x  16 x 256 0.151 BF
    9 max          2 x 2 / 2    16 x  16 x 256   ->     8 x   8 x 256
   10 conv    512  3 x 3 / 1     8 x   8 x 256   ->     8 x   8 x 512 0.151 BF
   11 max          2 x 2 / 1     8 x   8 x 512   ->     8 x   8 x 512
   12 conv    256  1 x 1 / 1     8 x   8 x 512   ->     8 x   8 x 256 0.017 BF
   13 conv    512  3 x 3 / 1     8 x   8 x 256   ->     8 x   8 x 512 0.151 BF
   14 conv    195  1 x 1 / 1     8 x   8 x 512   ->     8 x   8 x 195 0.013 BF
   15 yolo
   16 route  12
   17 conv    12[redacted]/ 1     8 x   8 x 256   ->     8 x   8 x 128 0.004 BF
   18 upsample [redacted]2x     8 x   8 x 128   ->    16 x  16 x 128
   19 route  18 8
   20 conv    195  1 x 1 / 1    16 x  16 x 384   ->    16 x  16 x 195 0.038 BF
   21 yolo
Total BFLOPS 1.035
Loading weights from aix2024.weights...
Done!
```

4) Verify the expected outputs

Finally, it executes the model on 229 test images and calculates the mAP. You are supposed to see “**mean average precision (mAP) = 0.817559, or 81.76%**”. Depending on your PC specifications, it may take more or less execution.

```
class_id = 53, name = dove_pink,         ap = 74.48 %
class_id = 54, name = dove_white,        ap = 88.93 %
class_id = 55, name = david_sunflower_seeds,     ap = 95.94 %
class_id = 56, name = monster_energy,    ap = 44.72 %
class_id = 57, name = act_ii_butter_lovers_popcorn,      ap = 86.10 %
class_id = 58, name = coca_cola_glass_bottle,    ap = 81.61 %
class_id = 59, name = twix,      ap = 85.90 %
 for thresh = 0.24, precision = 0.74, recall = 0.64, F1-score = 0.69
 for thresh = 0.24, TP = 1895, FP = 671, FN = 1064, average IoU = 55.01 %

 mean average precision (mAP) = 0.817559, or 81.76 %
Total Detection Time: 10.000000 Seconds
```

Then, let's run the quantized model. un the **int8 quantized** model and calculate the mAP by the command:

```
> script-unix-aix2024-test-all-quantized.cmd
```

→ Now, you can see some similar outputs as that of the full-precision model. However, after loading the model, it prints out the default quantization multipliers for an input image or input feature maps, weights, and biases.

```
Multipler     Input    Weight    Bias
 CONV0:         128      16      2048
 CONV2:         16       64      1024
 CONV4:         16       64      1024
 CONV6:         16       64      1024
 CONV8:         16       64      1024
 CONV10:        16       64      1024
 CONV12:        16       64      1024
 CONV13:        16       64      1024
 CONV14:        16       64      1024
 CONV17:        16       64      1024
 CONV20:        16       64      1024
```

Finally, it executes the **int8 quantized** model on 229 test images and then calculates the mAP. You are supposed to see "**mean average precision (mAP) = 0.550382, or 55.04 %**". Depending on your PC specifications, it may take more or less execution.

Since we used the default multiplier for quantization, the mAP result is currently poor. It's your job to improve it by following the 'Quantization Manual.pdf' and Tutorial 03 on Quantization."

```
class_id = 38, name = palmolive_orange,          ap = 69.79 %
class_id = 39, name = crystal_hot_sauce,         ap = 32.63 %
class_id = 40, name = tapatio_hot_sauce,         ap = 61.68 %
class_id = 41, name = nabisco_nilla_wafers,      ap = 89.12 %
class_id = 42, name = pepperidge_farm_milano_cookies_double_chocolate,   ap = 73.35 %
class_id = 43, name = campbells_chicken_noodle_soup,     ap = 45.40 %
class_id = 44, name = frappuccino_coffee,        ap = 80.00 %
class_id = 45, name = chewy_dips_chocolate_chip,         ap = 35.23 %
class_id = 46, name = chewy_dips_peanut_butter,          ap = 72.98 %
class_id = 47, name = nature_vally_fruit_and_nut,        ap = 24.98 %
class_id = 48, name = cheerios,          ap = 87.83 %
class_id = 49, name = lindt_excellence_cocoa_dark_chocolate,     ap = 59.94 %
class_id = 50, name = hersheys_symphony,         ap = 98.99 %
class_id = 51, name = campbells_chunky_classic_chicken_noodle,   ap = 66.98 %
class_id = 52, name = martinellis_apple_juice,   ap = 32.10 %
class_id = 53, name = dove_pink,         ap = 22.91 %
class_id = 54, name = dove_white,        ap = 58.06 %
class_id = 55, name = david_sunflower_seeds,     ap = 71.00 %
class_id = 56, name = monster_energy,    ap = 10.08 %
class_id = 57, name = act_ii_butter_lovers_popcorn,      ap = 47.84 %
class_id = 58, name = coca_cola_glass_bottle,    ap = 65.94 %
class_id = 59, name = twix,      ap = 37.09 %
 for thresh = 0.24, precision = 0.64, recall = 0.23, F1-score = 0.34
 for thresh = 0.24, TP = 672, FP = 380, FN = 2287, average IoU = 44.83 %

 mean average precision (mAP) = 0.550382, or 55.04 %
Total Detection Time: 9.000000 Seconds
```

You can also run a script to test the AIX 2024 model on one image.

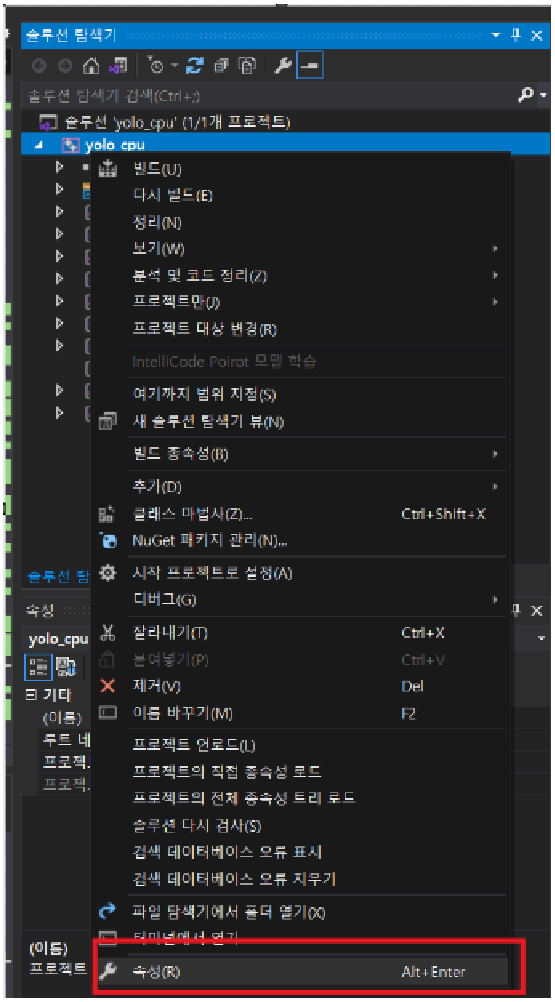
```
$ script-unix-aix2024-test-one.cmd
```

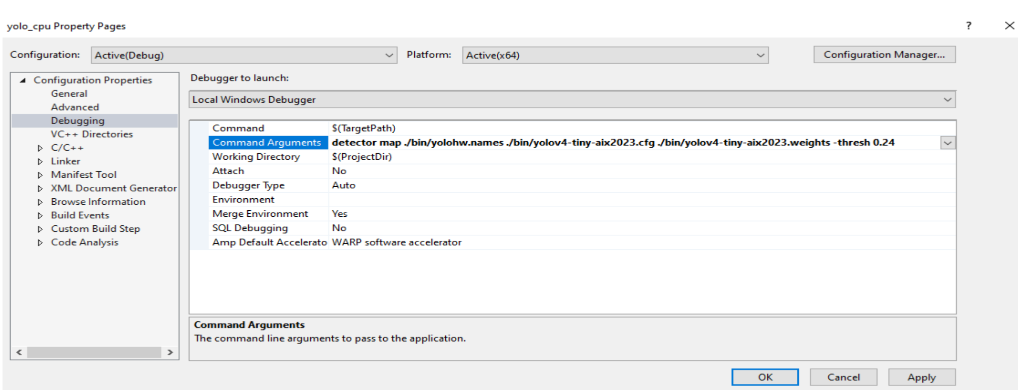
```
$ script-unix-aix2024-test-one-quantized.cmd
```

```
 Quantization!

Multipler    Input    Weight    Bias
 CONV0:        128     16      2048
 CONV2:         16      64      1024
 CONV4:         16      64      1024
 CONV6:         16      64      1024
 CONV8:         16      64      1024
 CONV10:        16      64      1024
 CONV12:        16      64      1024
 CONV13:        16      64      1024
 CONV14:        16      64      1024
 CONV17:        16      64      1024
 CONV20:        16      64      1024

 Saving quantized model...

 Saving quantized weights, bias, and scale for CONV00
 Saving quantized weights, bias, and scale for CONV02
 Saving quantized weights, bias, and scale for CONV04
 Saving quantized weights, bias, and scale for CONV06
 Saving quantized weights, bias, and scale for CONV08
 Saving quantized weights, bias, and scale for CONV10
 Saving quantized weights, bias, and scale for CONV12
 Saving quantized weights, bias, and scale for CONV13
 Saving quantized weights, bias, and scale for CONV14
 Saving quantized weights, bias, and scale for CONV17
 Saving quantized weights, bias, and scale for CONV20
test01.jpg: Predicted in 0.031000 seconds.
mom_to_mom_sweet_potato_corn_apple: 50% (left_x:  202   top_y:  544   width:  193   height:  311)
pringles_bbq: 78%       (left_x:  374   top_y:  270   width:  177   height:  340)
dr_pepper: 89%  (left_x:  643   top_y:  438   width:  135   height:  136)
cheeze_it: 49%  (left_x:  944   top_y:  281   width:  215   height:  304)
white_rain_body_wash: 35%       (left_x: 1207   top_y:  275   width:  191   height:  326)
Not compiled with OpenCV, saving to test01-det-quantized.png instead
```

- **Run the model AIX2024 with the Visual Studio project**

- Command Arguments



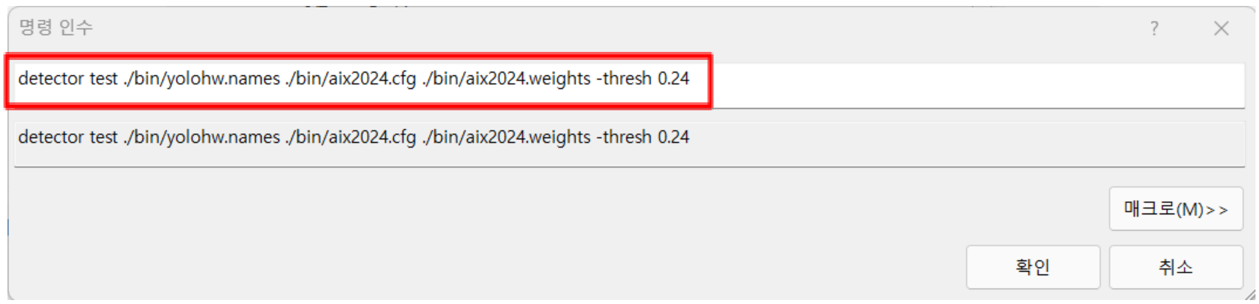


- To run the **full-precision** model,

Right-click on Solution yolo_cpu → Choose Properties → Debugging → Check if Command Arguments are as follows □ **click F5** to run the code

[Command Arguments]

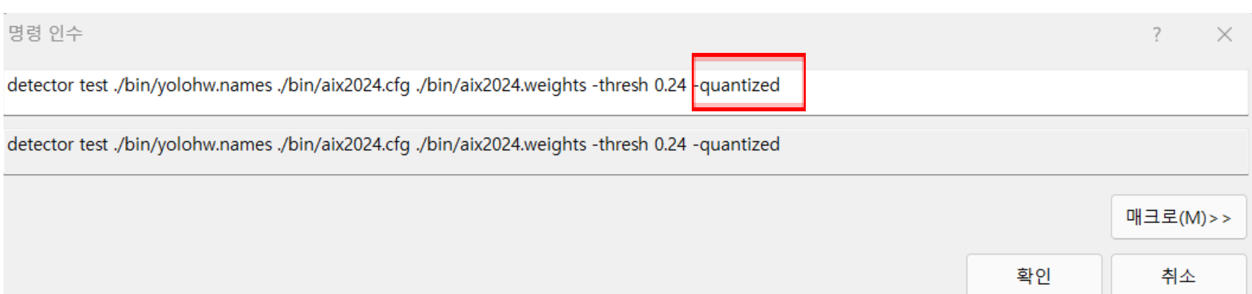
detector map yolohw.names aix2024.cfg aix2024.weights -thresh 0.24



- To run the **int8 quantized** model, you must modify the command as follows

[Command Arguments]

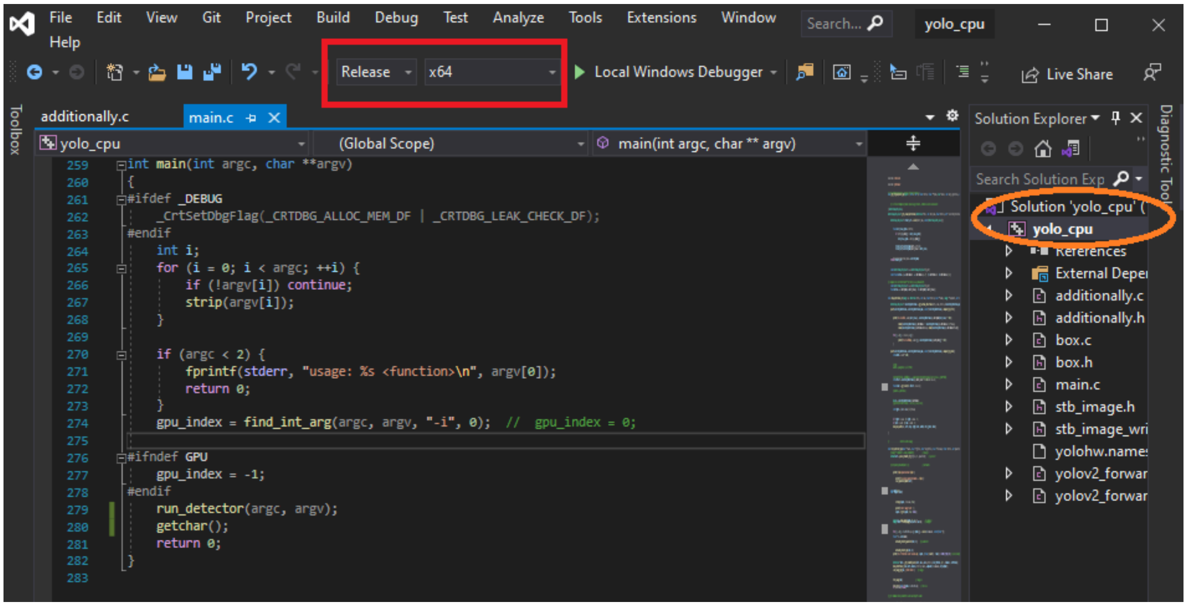
detector map yolohw.names aix2024.cfg aix2024.weights -thresh 0.24 -quantized



5) Debug vs Release

You can speed up the execution time by using **Release** instead of **Debug.**

1) Change Debug to **Release**



2) In the Release mode, you cannot use F5 as Debug. Instead, you can evaluate the program with Command Prompt.

- Before accessing the Command Prompt, it is necessary to first "Build (Ctrl+Shift+B)" in release mode. In doing so, you can observe that the size of the newly built .exe file has been reduced, compared to that of Debug mode.
- Run the <u>**full-precision**</u> model and calculate the mAP by the command:

```
> script-wins-aix2024-test-all.cmd
```

This command loads the model architecture from aix2024.cfg and the parameters from aix2024.weights. Finally, it executes the model on 229 test images and then calculates the mAP. You are supposed to see “**mean average precision (mAP) (mAP) =**

0.817559, or 81.76%". Depending on your PC specifications, it may take more or less execution. However, it should be >10x faster than that of Debug.

- (Click Enter and) Run the **int8 quantized** model and calculate the mAP by the command:

```
> script-wins-aix2024-test-all-quantized.cmd
```

→ After loading the model, it prints out the default quantization multipliers for an input image or input feature maps, weights, and biases. Finally, it executes the **int8 quantized** model on 229 test images and then calculates the mAP. You are supposed to see "**mean average precision (mAP) = 0.550382, or 55.04 %**". Depending on your PC specifications, it may take more or less execution. However, it should be 10x-20x faster than that of Debug.

```
 Quantization!

Multipler    Input    Weight    Bias
 CONV0:        128     16       2048
 CONV2:        16      64       1024
 CONV4:        16      64       1024
 CONV6:        16      64       1024
 CONV8:        16      64       1024
 CONV10:       16      64       1024
 CONV12:       16      64       1024
 CONV13:       16      64       1024
 CONV14:       16      64       1024
 CONV17:       16      64       1024
 CONV20:       16      64       1024

 Saving quantized model...

 Saving quantized weights, bias, and scale for CONV00
 Saving quantized weights, bias, and scale for CONV02
 Saving quantized weights, bias, and scale for CONV04
 Saving quantized weights, bias, and scale for CONV06
 Saving quantized weights, bias, and scale for CONV08
 Saving quantized weights, bias, and scale for CONV10
 Saving quantized weights, bias, and scale for CONV12
 Saving quantized weights, bias, and scale for CONV13
 Saving quantized weights, bias, and scale for CONV14
 Saving quantized weights, bias, and scale for CONV17
 Saving quantized weights, bias, and scale for CONV20
test01.jpg: Predicted in 0.031000 seconds.
mom_to_mom_sweet_potato_corn_apple: 50% (left_x:  202   top_y:  544   width:  193   height:  311)
pringles_bbq: 78%       (left_x:  374   top_y:  270   width:  177   height:  340)
dr_pepper: 89%  (left_x:  643   top_y:  438   width:  135   height:  136)
cheeze_it: 49%  (left_x:  944   top_y:  281   width:  215   height:  304)
white_rain_body_wash: 35%       (left_x: 1207   top_y:  275   width:  191   height:  326)
Not compiled with OpenCV, saving to test01-det-quantized.png instead
```

References

[1]. https://github.com/pjreddie/darknet