AIX2024 SDK Installation Guide

: Compile & Run

(For LINUX / MacOS)

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 Ubuntu and Python.

- **Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-142-generic x86_64)**
- **Python version: 3.7.9**
- **GCC version: 7.5.0**

[install required package]

1. **Check Python version**

```
$ python3 –version
Python 3.7.9
```

2. **Check GCC version**

```
$ gcc –version
gcc (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0
Copyright (C) 2017 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.
```

Note: By the way, the AIX2024 is supposed to work on other versions. If you have any issue related to the package versions, please contact us.

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) *Execute the scripts*
4) *Verify the expected outputs*

Assume that the AIX2024 code is unzipped and stored in your Linux/Unix PC. For example, the AIX2024 is located at /home/truongnx/aix2024/skeleton/

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. Execute the following command lines:

```
$ cd bin/dataset
$ python3 make_list_cur.py
```

After running those commands, you are supposed to see:

```
CAPP_testset_long_10063
CAPP_testset_close_10031
CAPP_testset_close_10052
CAPP_testset_long_10109
CAPP_testset_long_10007
CAPP_testset_close_10016
CAPP_testset_long_10083
CAPP_testset_close_10075
CAPP_testset_close_10101
CAPP_testset_long_10076
CAPP_testset_long_10042
CAPP_testset_long_10068
CAPP_testset_long_10107
CAPP_testset_close_10073
CAPP_testset_close_10088
CAPP_testset_long_10113
CAPP_testset_close_10062
CAPP_testset_long_10036
(base) truongnx@marlin:~/aix2023/skeleton/bin/dataset$
```

The command executes **make_list_cur.py** to generate all directories of the test images with your local directory and save them to **target.txt**. Note that only jpeg images are included in the file. Now, you can view “target.txt” which stores the directories of all test images.

```
$ vi target.txt
```

```
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_long_10098.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10054.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_long_10041.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10096.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10107.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_long_10075.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10027.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10098.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_long_10057.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10012.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10035.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10048.jpg
/home/truongnx/aix2023/skeleton/bin/dataset/CAPP_testset_close_10003.jpg
"target.txt" 229L, 16602C
```

To close the file, you use a combination of keys: “Esc” → “:” → “q” → “!”, which closes the file without any modification.

Now, you can go back to the main folder ***skeleton/*** by using the following command:

```
$ cd ../../
```

2) Compile the AIX2024

*****NOTE: Make sure that you are at skeleton/ before compiling the code.***

- Clean all object files in obj/ and the executable file bin/darknet using the command line:

```
$ make clean
```

- Execute the command line to compile the AIX2024 code:

```
$ make
```

After compilation, you are supposed to see the following screen. And the executable file **"darknet"** is generated and stored in bin/.

```
gcc  -Wall -Wfatal-errors -Ofast -c ./src/yolov2_forward_network_quantized.c -o obj/yolov2_forward_ne
twork_quantized.o
./src/yolov2_forward_network_quantized.c: In function ‘gemm_nn_int8_int32’:
./src/yolov2_forward_network_quantized.c:11:32: warning: integer overflow in expression [-Woverflow]
 #define MAX_VAL_32 (256*256*256*256/2 - 1) // 31-bit (1-bit sign)
                                ^
./src/yolov2_forward_network_quantized.c:104:47: note: in expansion of macro ‘MAX_VAL_32’
             C[i*ldc + j] += max_abs(c_tmp[j], MAX_VAL_32);
                                               ^~~~~~~~~~
./src/yolov2_forward_network_quantized.c: In function ‘forward_convolutional_layer_q’:
./src/yolov2_forward_network_quantized.c:140:0: warning: ignoring #pragma omp parallel [-Wunknown-pra
gmas]
     #pragma omp parallel for

./src/yolov2_forward_network_quantized.c: In function ‘do_quantization’:
./src/yolov2_forward_network_quantized.c:288:26: warning: unused variable ‘weights_size’ [-Wunused-va
riable]
             size_t const weights_size = l->size*l->size*l->c*l->n;
                          ^~~~~~~~~~~~
./src/yolov2_forward_network_quantized.c: In function ‘save_quantized_model’:
./src/yolov2_forward_network_quantized.c:333:26: warning: unused variable ‘filter_size’ [-Wunused-var
iable]
             size_t const filter_size = l->size*l->size*l->c;
                          ^~~~~~~~~~~
In file included from ./src/yolov2_forward_network_quantized.c:1:0:
At top level:
./src/additionally.h:130:17: warning: ‘activate_array’ defined but not used [-Wunused-function]
     static void activate_array(float *x, const int n, const ACTIVATION a)
                 ^~~~~~~~~~~~~~
gcc  -Wall -Wfatal-errors -Ofast obj/main.o obj/additionally.o obj/box.o obj/yolov2_forward_network.o
 obj/yolov2_forward_network_quantized.o -o bin/darknet -lm -pthread
(base) truongnx@marlin:~/aix2023/skeleton$
```

- Now, you can go to bin/ to check if "darknet" is generated.

```
$ cd bin/
$ ll
```

```
(base) truongnx@marlin:~/aix2023/skeleton/bin$ ll
total 10332
drwxrwxr-x 4 truongnx truongnx    4096  1월  15 13:49 ./
drwxrwxr-x 8 truongnx truongnx    4096  1월  15 13:40 ../
-rwxrwxr-x 1 truongnx truongnx  312448  1월  15 13:49 darknet*
drwxrwxr-x 2 truongnx truongnx   28672  1월  15 13:33 dataset/
-rw-rw-r-- 1 truongnx truongnx 2486319  1월  10 13:41 predictions.png
-rw-rw-r-- 1 truongnx truongnx   82944  1월  10 13:41 pthreadVC2.dll
-rw-rw-r-- 1 truongnx truongnx       0  1월  10 13:41 target.txt
-rwxrwxr-x 1 truongnx truongnx     352  1월  10 13:41 tiny-yolo-aix2023-int8.sh*
-rwxrwxr-x 1 truongnx truongnx     112  1월  10 13:41 tiny-yolo-aix2023-int8-test.sh*
-rwxrwxr-x 1 truongnx truongnx     203  1월  10 13:41 tiny-yolo-aix2023.sh*
drwxrwxr-x 2 truongnx truongnx    4096  1월  10 13:41 weights/
-rw-rw-r-- 1 truongnx truongnx     222  1월  10 13:41 yolo_cpu.cmd
-rw-rw-r-- 1 truongnx truongnx  389632  1월  10 13:41 yolo_cpu.exe
-rw-rw-r-- 1 truongnx truongnx  991608  1월  10 13:41 yolo_cpu.ilk
-rw-rw-r-- 1 truongnx truongnx     244  1월  10 13:41 yolo_cpu_int8.cmd
-rw-rw-r-- 1 truongnx truongnx     246  1월  10 13:41 yolo_cpu_int8_test.cmd
-rw-rw-r-- 1 truongnx truongnx 1143878  1월  10 13:41 yolo_cpu.iobj
-rw-rw-r-- 1 truongnx truongnx  275856  1월  10 13:41 yolo_cpu.ipdb
-rw-rw-r-- 1 truongnx truongnx  733184  1월  10 13:41 yolo_cpu.pdb
-rw-rw-r-- 1 truongnx truongnx     115  1월  10 13:41 yolohw.data
-rw-rw-r-- 1 truongnx truongnx    1323  1월  10 13:41 yolohw.names
-rw-rw-r-- 1 truongnx truongnx    3257  1월  10 13:41 yolov4-tiny-aix2023.cfg
-rw-rw-r-- 1 truongnx truongnx 4059884  1월  10 13:41 yolov4-tiny-aix2023.weights
(base) truongnx@marlin:~/aix2023/skeleton/bin$
```

3) Execute the scripts

NOTE: Make sure that you are at skeleton/bin/ before executing the script.

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

```
$ script-unix-aix2024-test-all.sh
```

➔ This command uses the default "dataset/target.txt" generated at Step 1 and the name list of 60 product items stored in the file "yolohw.names". Next, it loads the model architecture from aix2024.cfg and then loads the parameters from aix2024.weights.

```
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    128  1 x 1 / 1     8 x   8 x 256   ->     8 x   8 x 128 0.004 BF
   18 upsample           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
```

4) Verify the expected outputs

Finally, it executes the model on 229 test images and then 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 = 24, name = reeses_pieces,     ap = 100.00 %
class_id = 25, name = clif_crunch_peanut_butter,         ap = 73.53 %
class_id = 26, name = mom_to_mom_butternut_squash_pear,
ap = 90.05 %
class_id = 27, name = pop_tararts_strawberry,    ap = 91.21 %
class_id = 28, name = quaker_big_chewy_chocolate_chip,   ap = 77.93 %
class_id = 29, name = spam,      ap = 61.59 %
class_id = 30, name = coffee_mate_french_vanilla,        ap = 57.76 %
class_id = 31, name = pepperidge_farm_milk_chocolate_macadamia_cookies,   ap = 74.59 %
class_id = 32, name = kitkat_king_size,          ap = 60.20 %
class_id = 33, name = snickers,          ap = 11.17 %
class_id = 34, name = toblerone_milk_chocolate,          ap = 41.16 %
class_id = 35, name = clif_z_bar_chocolate_chip,         ap = 97.71 %
class_id = 36, name = nature_valley_crunchy_oats_n_honey,        ap = 86.78 %
class_id = 37, name = ritz_crackers,     ap = 100.00 %
class_id = 38, name = palmolive_orange,          ap = 87.18 %
class_id = 39, name = crystal_hot_sauce,         ap = 85.22 %
class_id = 40, name = tapatio_hot_sauce,         ap = 66.87 %
class_id = 41, name = nabisco_nilla_wafers,      ap = 85.76 %
class_id = 42, name = pepperidge_farm_milano_cookies_double_chocolate,    ap = 94.20 %
class_id = 43, name = campbells_chicken_noodle_soup,     ap = 99.47 %
class_id = 44, name = frappuccino_coffee,        ap = 91.53 %
class_id = 45, name = chewy_dips_chocolate_chip,         ap = 64.94 %
class_id = 46, name = chewy_dips_peanut_butter,          ap = 89.97 %
class_id = 47, name = nature_vally_fruit_and_nut,        ap = 92.30 %
class_id = 48, name = cheerios,          ap = 96.27 %
class_id = 49, name = lindt_excellence_cocoa_dark_chocolate,     ap = 81.82 %
class_id = 50, name = hersheys_symphony,         ap = 100.00 %
class_id = 51, name = campbells_chunky_classic_chicken_noodle,   ap = 94.65 %
class_id = 52, name = martinellis_apple_juice,   ap = 79.72 %
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 = 0.80, recall = 0.70, F1-score = 0.74
 for thresh = 0.24, TP = 2058, FP = 508, FN = 901, average IoU = 59.16 %

 mean average precision (mAP) = 0.817559, or 81.76 %
Total Detection Time: 36.000000 Seconds
bo@BoGram14:/mnt/c/skeleton/bin$
```

- Run the **int8 quantized** model and calculate the mAP by the command:

```
$ script-unix-aix2024-test-all-quantized.sh
```

→ 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.

```
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    128  1 x 1 / 1     8 x   8 x 256   ->     8 x   8 x 128 0.004 BF
   18 upsample           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!
```

```
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.sh
```

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

References

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