A PROGRAMMING GUIDE

for the COURSE of

ARTIFICIAL INTELLIGENCE & NEURAL NETWORKS

2020

MENG ZHAOHUI

HOHAI UNIVERSITY

2020-10-01

****
****
* CHAPTER 1
* How to setup a ubuntu system
****
****

We will install the linux system of UBUNTU18.04 on a computer without any operating system.

****
* 1.1
* Download iso file
****

"ubuntu-18.04.4-desktop-amd64.iso"

****
* 1.2
* Make a startup disk(USB disk)
****

(1) windows : if you have a windows system, write iso file into a USB disk(>8GB) using Rufus(a software).

select : dd, MBR, UEFI

(2) ubuntu : if you have a linux(ubuntu) system, write iso file into a USB disk(>8GB) using Startup Disk Creator (a software in ubuntu system).

****
* 1.3
* install ubuntu
****

power on a computer and press <del> and entry the BIOS interface, select USB boot, and then reboot.

install ubuntu18.04, select NORMAL

define : computername, username, password

after done, then reboot

```
************************************
************************************
* CHAPTER 2
* Install software
************************************
************************************
```

```
**********************************
* 2.1
* install gcc  (necessary)
**********************************
```

open Terminal, then run command ($ is the prompt of the shell):

```
$ sudo apt-get install build-essential
```

after done, run command to verify:

```
$ 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.
```

```
**********************************
* 2.2
* install cmake  (necessary)
**********************************
```

```
$ sudo apt-get install cmake
```

after done, run command to verify:

```
$ cmake --version

cmake version 3.10.2
CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

```
******************************
* 2.3
* install opencv  (necessary)
******************************

command line :

$ sudo apt-get install libopencv-dev

after done, then command line :

$ dpkg -L libopencv-dev

result below :

/.
/usr
/usr/bin
/usr/bin/opencv_annotation
/usr/bin/opencv_createsamples
/usr/bin/opencv_interactive-calibration
/usr/bin/opencv_traincascade
/usr/bin/opencv_version
/usr/bin/opencv_visualisation
/usr/bin/opencv_waldboost_detector
/usr/include
/usr/include/opencv
/usr/include/opencv/cv.h
/usr/include/opencv/cv.hpp
/usr/include/opencv/cvaux.h
/usr/include/opencv/cvaux.hpp
/usr/include/opencv/cvwimage.h
/usr/include/opencv/cxcore.h
/usr/include/opencv/cxcore.hpp
/usr/include/opencv/cxeigen.hpp
/usr/include/opencv/cxmisc.h
/usr/include/opencv/highgui.h
/usr/include/opencv/ml.h
/usr/lib
/usr/lib/x86_64-linux-gnu
```

```
/usr/lib/x86_64-linux-gnu/pkgconfig
/usr/lib/x86_64-linux-gnu/pkgconfig/opencv.pc
/usr/share
/usr/share/OpenCV
/usr/share/OpenCV/OpenCVConfig-version.cmake
/usr/share/OpenCV/OpenCVConfig.cmake
/usr/share/OpenCV/OpenCVModules-release.cmake
/usr/share/OpenCV/OpenCVModules.cmake
/usr/share/doc
/usr/share/doc/libopencv-dev
/usr/share/doc/libopencv-dev/copyright
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/opencv_createsamples.1.gz
/usr/share/man/man1/opencv_haartraining.1.gz
/usr/share/man/man1/opencv_performance.1.gz
/usr/share/man/man1/opencv_traincascade.1.gz
/usr/share/doc/libopencv-dev/changelog.Debian.gz
```

```
**********************************
* 2.4
install synaptic (necessary)
**********************************

command line :

$ sudo apt-get install synaptic

after done, there is an icon of synaptic would be on the desktop
```

```
*******************************
* 2.5
* install cuda  (necessary)
*******************************

command line :
```

```
$ sudo apt-get install nvidia-cuda-toolkit

after done, then command line :

$ nvcc --version

nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2017 NVIDIA Corporation
Built on Fri_Nov__3_21:07:56_CDT_2017
Cuda compilation tools, release 9.1, V9.1.85




**********************************
* 2.6
install nvidia drivers (necessary)
**********************************

install nvidia-driver-390 / or 430 or 435, by Synaptic

(1) open Synaptic,

(2) press <Reload> button on the top menu bar,

(3) press <Search> button on the top menu bar, search "nvidia-driver-390", or other
keywords like this,

(4) select and mark,

(5) press <Apply> button on the top menu bar, to download and install.


after done, then command line :

hhcv@hhcv:~$ nvidia-smi

result below :
```

```
Fri May 29 09:49:57 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 390.132              Driver Version: 390.132                     |
|-------------------------------+----------------------+----------------------+
| GPU  Name       Persistence-M| Bus-Id       Disp.A  | Volatile Uncorr. ECC   |
| Fan  Temp  Perf  Pwr:Usage/Cap|       Memory-Usage | GPU-Util  Compute M.    |
|===============================+======================+======================|
|   0  GeForce GT 710     Off  | 00000000:01:00.0 N/A |               N/A       |
| 40%   38C    P0    N/A /  N/A |    404MiB /  1994MiB |    N/A     Default       |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU      PID   Type   Process name                             Usage      |
|=============================================================================|
|   0                Not Supported                                            |
+-----------------------------------------------------------------------------+
```

```
************************************
* 2.7
* install VScode (necessary / or other code editor)
************************************
```

download deb package : code_1.44.2-1587059832_amd64.deb

run command:

```
$ sudo dpkg -i ~/Downloads/code_1.44.2-1587059832_amd64.deb
```

after done, there is an icon of vscode would be on the desktop

```
************************************
* 2.8
install latex (optional)
************************************
```

```
$ sudo apt-get install texlive-full

$ sudo apt-get install texmaker

after done, there is a texmaker icon would be on the desktop

or run in command line ($ is a prompt for command):

$ texmaker




**********************************
* 2.9
* install emacs (optional)
**********************************

$sudo apt-get install emacs

after done, there are two emacs icons(terminal & GUI) would be on the desktop

or run in command line ($ is a prompt for command):

$ emacs




**********************************
* 2.10
* install ffmpeg (optional)
**********************************

$sudo apt-get install ffmpeg

after done, run command:

$ ffplay ~/meng/AI2019/camera_all_in_one_cuda_1.avi

## ffplay video-file-name(path & filename)
```

```
************************************
************************************
* CHAPTER 3
* A test for using opencv
************************************
************************************




**********************************
* 3.1
* code file
**********************************

code file is a file of c++ program.

code file and makefile are both in the directory of ~/meng/AI2019/

file name : AIcourse2019.cpp
file directory : ~/meng/AI2019/



*******************************
* start of code file (only a test)
*******************************

/**
* 2019-08-04
*/

#include <iostream>
#include <string>
#include <stdio.h>

#include <stdlib.h>

#include <opencv2/opencv.hpp>

using namespace std;
using namespace cv;

//////////
```

```
// main  //
///////////

int main()
{

  printf("hello !\n");

  Mat img = imread("wannan.jpg");

  imshow("image", img);

  waitKey();



  cv::VideoCapture capture_file1("camera_all_in_one_cuda_1.avi");

  cv::Mat src_matFrame1;

  cv::namedWindow("ORIGINAL1");

  bool read_valid_frame1 = true;

  while(1)
    {

      read_valid_frame1 = capture_file1.read(src_matFrame1);

      if (!read_valid_frame1) break;

      imshow(""""ORIGINAL1", src_matFrame1);

      char c = cvWaitKey(20);
      if (c == 27) break;
    }



  cv::VideoCapture capture_file2("201809261329_003477AA.MP4");

  cv::Mat src_matFrame2;

  cv::namedWindow("ORIGINAL2");
```

```
  bool read_valid_frame2 = true;

  while(1)
    {

      read_valid_frame2 = capture_file2.read(src_matFrame2);

      if (!read_valid_frame2) break;

      imshow(""ORIGINAL2", src_matFrame2);

      char c = cvWaitKey(20);
      if (c == 27) break;
    }


  return 0;

}

*******************************
* end of code file (only a test)
*******************************
```

```
***********************************
* 3.2
* make file
***********************************

make file is a file of commands of compile and link, etc.

code file and makefile are both in the directory of ~/meng/AI2019/

file name : makefile
file directory : ~/meng/AI2019/
```

```
******************************
* start of make file (only a test)
******************************


INCLUDE=$(shell pkg-config opencv --cflags)
## get string --> -I/usr/include/opencv

INCLUDE2=-I/usr/include/opencv2

LIB=$(shell pkg-config opencv --libs)
##  get  string  -->    -lopencv_shape  -lopencv_stitching  -lopencv_superres
-lopencv_videostab -lopencv_aruco
-lopencv_bgsegm    -lopencv_bioinspired    -lopencv_ccalib    -lopencv_datasets
-lopencv_dpm -lopencv_face -lopencv_freetype
-lopencv_fuzzy    -lopencv_hdf    -lopencv_line_descriptor    -lopencv_optflow
-lopencv_video -lopencv_plot -lopencv_reg
-lopencv_saliency          -lopencv_stereo          -lopencv_structured_light
-lopencv_phase_unwrapping -lopencv_rgbd -lopencv_viz
-lopencv_surface_matching  -lopencv_text  -lopencv_ximgproc  -lopencv_calib3d
-lopencv_features2d -lopencv_flann
-lopencv_xobjdetect     -lopencv_objdetect     -lopencv_ml     -lopencv_xphoto
-lopencv_highgui -lopencv_videoio -lopencv_imgcodecs
-lopencv_photo -lopencv_imgproc -lopencv_core


OBJECTS=AIcourse2019.o
SOURCE=AIcourse2019.cpp

BIN=AIcourse2019


#$(BIN) : $(OBJECTS)
#   g++ -o $(BIN) $(OBJECTS) $(INCLUDE) $(INCLUDE2)

$(BIN) : $(OBJECTS)
    g++ -o $(BIN) $(OBJECTS) $(INCLUDE) $(INCLUDE2) $(LIB)


$(OBJECTS) : $(SOURCE)
    g++ -c $(SOURCE)
```

```
*******************************
* end of make file (only a test)
*******************************
```

```
***********************************
* 3.3
* make
***********************************
```

command line :

```
make
```

```
*******************************
* 3.4
* list file in current dir
*******************************
```

command line :

```
ll
```

result below :

```
m1@hhcv:~/meng/AI2019$ ll
total 109628
drwxr-xr-x 2 m1 m1     4096 8月   5 21:27  ./
drwxr-xr-x 3 m1 m1     4096 8月   4 20:52  ../
-rw-r--r-- 1 m1 m1 104857600 9月  26  2018  201809261329_003477AA.MP4         ## video file
-rwxr-xr-x 1 m1 m1    59000 8月   5 15:17  AIcourse2019*                   ## run file : after
make
-rw-rw-r-- 1 m1 m1     1201 8月   5 15:17  AIcourse2019.cpp                 ## source file
-rw-r--r-- 1 m1 m1   121536 8月   5 15:17  AIcourse2019.o                   ## object file :
after make
-rw-r--r-- 1 m1 m1  7071388 1月  30  2015  camera_all_in_one_cuda_1.avi       ## video file
-rw-r--r-- 1 m1 m1     1142 8月   5 14:40  makefile                        ## make file
-rw-rw-r-- 1 m1 m1    99356 8月   5 14:48  wannan.jpg                       ## picture file
```

```
*******************************
* 3.5
* run
*******************************

command line :

~/meng/AI2019/AIcourse2019

result : show picture, show video file, show video file.




*******************************
* 3.6
* why use shell command in makefile
*******************************


command line:

pkg-config opencv --cflags

result :

-I/usr/include/opencv

that is, sentence in makefile, INCLUDE=$(shell pkg-config opencv --cflags),
shell command could get string "-I/usr/include/opencv", then assign to varible
INCLUDE.



command line:

pkg-config opencv --libs

result :

-lopencv_shape    -lopencv_stitching    -lopencv_superres    -lopencv_videostab
```

```
-lopencv_aruco -lopencv_bgsegm
-lopencv_bioinspired    -lopencv_ccalib    -lopencv_datasets    -lopencv_dpm
-lopencv_face -lopencv_freetype -lopencv_fuzzy
-lopencv_hdf    -lopencv_line_descriptor    -lopencv_optflow    -lopencv_video
-lopencv_plot -lopencv_reg -lopencv_saliency
-lopencv_stereo      -lopencv_structured_light      -lopencv_phase_unwrapping
-lopencv_rgbd -lopencv_viz -lopencv_surface_matching
-lopencv_text    -lopencv_ximgproc    -lopencv_calib3d    -lopencv_features2d
-lopencv_flann -lopencv_xobjdetect -lopencv_objdetect
-lopencv_ml      -lopencv_xphoto      -lopencv_highgui      -lopencv_videoio
-lopencv_imgcodecs -lopencv_photo -lopencv_imgproc -lopencv_core
```

that is, sentence in makefile, LIB=$(shell pkg-config opencv --libs), shell command could also get string "-lopencv_shape -lopencv_stitching -lopencv_superres ...", then assign to varible LIB.

```
************************************
************************************
* CHAPTER 4
* GVNN2019 : an example neural networks
************************************
************************************
```

GVNN2019 is a collection of functions for an example neural networks.

it contains many functions to deal with video files from two data sets, <hmdb51> and <ucf101>.

or another drive video set, <hhdrv>.

```
**********************************
* 4.1
* main dir
**********************************
```

/home/m1/meng/

or

/home/hhcv/meng/

or other directory you enjoy. this main dir can be settled by yourself.

you can modify this main directory in the header file : aliveBase.h

```
constexpr char* DIR_HOME_MENG = (char*)"/home/hhcv/meng/";
```

and in the makefile : makefile

```
DIR_MAIN = /home/hhcv/meng/gvnn2019/
```

gvnn2019/ is a sub directory of the main directory, this name must be preserved.

```
************************************
* 4.2
* data dir
************************************

two video sets, <hmdb51> and <ucf101>, you can download by youself.

/home/m1/meng/hmdb51/
or
/home/m1/meng/ucf101/
or
/home/m1/meng/hhdrv/



************************************
* 4.3
* sourse code dir
************************************

/home/m1/meng/gvnn2019/src/


************************************
* 4.4
* object file dir
************************************

/home/m1/meng/gvnn2019/obj/

"makefile" is in this directory



************************************
* 4.5
* run parameter file
************************************

"GVNN_run_para_linux.txt" is in the directory "/home/m1/meng/gvnn2019/"
```

****
* 4.6
* functions
****

There are many functions in <gvnn2019>, the selection of these functions is under control of <basePara->para_RUN_PROCEDURE>,
which could be modified in the text file <"gvnn2019/GVNN_run_para_linux.txt">.

****
* 4.7
* more notes
****

In course of <Neural Networks 2019>, we focus on the function of <basePara->para_RUN_PROCEDURE == 78 or 79>,

<78> is for video set of hmdb51, and <79> is for ucf101.

By now, we have completed the tasks of getting low level super pixels for every frame of every video in the two video sets, and displaying.

Other tasks will coming soon.

In course of <Artificial Intelligence 2020>, ...

****
* 4.8
* home works
****

Basically, you can read and explain the code mentioned above, or others, <31/32 or 65/66> be recommended.

You can do more works, if you can find some modifies of the functions.