Lab 1 on embedded Artificial Intelligence on microcontroler

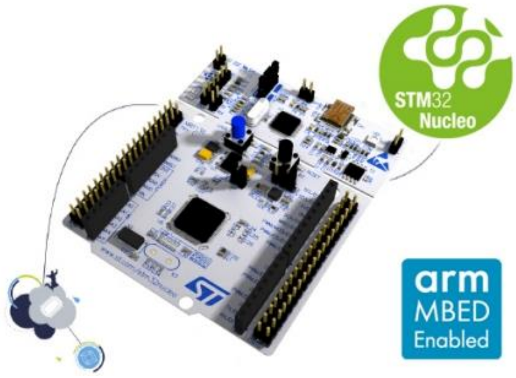
Polytech Nice Sophia

During this lab, we will prepare the software environment needed for the programmation of AI-based algorithms onto MCU (MicroControler Unit). This environment will be used during next Labs.

Hardware target

The hardware target is a Nucleo 64 board, equipped with a STM32L476 MCU. You will also need a USB type A to USB mini-B cable. Please look if you have a cable of this type, we don't have enough yet.

This MCU is based on the ARM Cortex M4 architecture and runs at a frequency of 80 MHz. The board provides 1 MB Flash and 128 KB SRAM.



Software environment

The following installation guide is targeting a **windows 10** operating system and will also use **docker**. See annex for installation on Linux.

You must be sure to have abour 10 GB of free disk space for the installation of the tools. If you need space, you can for example uninstall Quartus that is not needed for the last step of the course.

We will mainly use the following software tools that you can download from a high bandwidth connection:

STM32CubeIDE 1.6.1 [746 Mo for the installer and 2,1 Go for the IDE] with the password ***Sensors*** from

https://unice-my.sharepoint.com/:u:/g/personal/benoit_miramond_unice_fr/ES4aQV6xR5FJnJdF_uhKHx0Bz4Sm4MQD-LBOX7iG1TfHew?e=5C7iz6

Docker desktop [512 Mo] from https://hub.docker.com/editions/community/docker-ce-desktop-windows

(no registration)

Note: Since the last version, Docker works with Windows 10 Pro, Entreprise, Education, or Windows Home.

Installation steps

Part1. Learn Neural Networks with Docker (each time you will train a CNN)

1. Connect to moodle, and download the provided TensorFlow script to train (during the lab) a predefined neural network
2. Open a windows powershell and go to the directory where you get the TF script and type the following command that will download the tensorflow image the first time you type it

```
docker.exe run -it --rm -p 8888:8888 -e JUPYTER_ENABLE_LAB=yes –v $((pwd).tostring()+"/d1-notebooks:/home/jovyan/test")  jupyter/tensorflow-notebook
```

If you have the following error : docker: invalid reference format, please type by hand the preceding command (no copy/paste)

3. Accept the request for access rights from Docker
4. Copy the link beginning by http://127.0.0.1... provided in the powershell and paste it in a web browser.
   (Tested with Firefox 74.0)
5. Open the script provided in the test directory.
6. You can just take a look to the code. Explanations will come during the course and the lab.

Part 2. Configure STM32CubeIDE and STM CubeAI (only once)

1. From the executable file downloaded, run the installer with the default settings.
2. Accept the installation of drivers (3 times).
3. Run STM32CubeIDE.
4. Create a new project for STM32 micro-controler (new STM32 Project).
5. Click on Board Selector, search for L476 and then select the board nucleo L476RG. Click next.
6. Finalize the project creation (finish, Yes, Yes). It will download the board support package for this board and it will take few minutes the first time.
7. Click on Software Packs → Select Components → STMicroelectronics.X-CUBE-AI → Install
8. Accept license
9. Click on Software Packs → Select Components → STMicroelectronics.X-CUBE-AI → Artificial Intelligence X-CUBE-AI → Core
10. Click on Software Packs → Select Components → STMicroelectronics.X-CUBE-AI → Device Application → Application → Validation
11. Click OK.
    - This new package contains the software part that will translate the abstract convolutional neural network model (CNN) coming from TensorFlow and learned in TensorFlow into a compressed model coded in C and C++.
12. CubeMX perspective → Category "Connectivity" → "USART2"
13. "Configuration" panel → "Parameter Settings" tab → "Basic Parameters" → "Baud Rate" → "912600”
14. Unnecessary, should be configured by default. Otherwise, category " Additional Software" on the left menu → "STMicroelectronics.X-CUBE-AI"

- "Configuration" panel → "Platform Settings" tab → "COM Port", "IPs or Components" drop-down "USART: asynchronous", "Found Solutions" drop-down USART2

Part 3. Program the CNN on the board with STM32CubeIDE (each time you will deploy a CNN on the board)

15. "Configuration" panel → "Add network"
    - "Choose model…" drop-down → "Keras" "Saved model"
16. Select also the following:
    - the type of CNN model we are using: Keras
    - select the file with extension .h5 for the description of the pre-trained CNN in your TF folder (step 1.6)
    - the compression level: None (others to be tested during **Lab 2)**
    - validation inputs: select the file X_test from the directory where you trained your model in step 1.6.
    - validation outputs: select the file Y_test from the same directory
17. (Optional: (we will come back on the following steps in Lab 2)
    - Select Analyze to verify that you get the correct files.
    - Select Validation on Desktop to get the details of the CNN layer per layer
18. Select “Validation on target” to download the code on target, verify the configuration:
    - Use communication port: Automatic
    - Bauds: 912600 = enabled
    - **Tick the box “enabled”** in the part « Automatic compilation and download » without changing the default settings: ST-Link | PA2 | PA3 | STM32CubeIDE | SWD
19. Wait for the injection of the test vectors through the serial connection (10 000 vectors).
    - You will get the result of the inference phase of the CNN at the end of the injection.
    - Warning: the window printing the test vectors is freezing sometimes but is running in background
    - Warning: don’t put your PC on automatic standby during this step
20. If you get the result of the execution on the target, you are ready for **Lab 2**. See you next week!

Annex 1 - Installation guide under Linux

## Tensorflow installation on Linux

### Install Docker

In the terminal:

```
sudo apt update
sudo apt install docker.io
sudo usermod -G docker -a $USER
```

Reboot

### Download notebooks

In the terminal:

```
git clone https://gitlab.miaounyan.eu/d1/d1-notebooks
```

### Run JupyterLab with Python3/Tensorflow 2.1 on Docker

In the terminal:

```
setfacl -R -m u:1000:rwX,d:u:1000:rwX,d:u:$(whoami):rwX d1-notebooks
docker run -it --rm -p 8888:8888 -e JUPYTER_ENABLE_LAB=yes –v "$(pwd)"/d1-notebooks:/home/jovyan/test jupyter/tensorflow-notebook
```

Open link `127.0.0.1/[…]` in the browser

Open `d1-notebooks/MNIST_LeNet-5.ipynb`

Click Cells menu → Run all

## STM32CubeIDE installation on Linux

```
sudo apt install libncurses5 make gcc g++ wish # deps for STM32CubeAI desktop validation
```

https://www.st.com/en/development-tools/stm32cubeide.html

GET SOFTWARE STM32CubeIDE-DEB STM32CubeIDE Debian Linux Installer 1.6.1

Extract the zip and Right click on extracted file → Properties → Permissions → Check "Allow executing file as program"

```
sudo Downloads/st-stm32cubeide_XXX.deb_bundle.sh (replace XXX by the name of the file)
```

Space to scroll license until the end, I ACCEPT type 'y'

Run STM32CubeIDE from the application menu