Research Project Installation Manual

vanGoethem, Joren Maerten, Andreas

May 24, 2022

Contents

1 software

1.1 Python

Before anything, it is important to make sure python is installed, python 3.9.x is highly recommended. You can install python from the following url:

https://www.python.org/downloads/

Once python is installed you can double check the installation by opening up a terminal and entering the following command.

```
    python -V
```

1.2 CUDA

If you have an Nvidia graphics card, it is possible to run the model from there. To do this, a piece of software is required called CUDA. CUDA allows Nvidia gpu's to use their parallel computation prowess for the training and usage of AI models.

To install CUDA, you can follow the official Nvidia guide https://docs.nvidia.com/cuda/ cuda-installation-guide-microsoft-windows/index.html. To summarize it briefly, you will need two parts to install CUDA, the first is the installer software that will install the drivers necessary to use CUDA, and the second counterpart is the cuDNN library. In order to download cuDNN you will need an Nvidia account for verification, the guide tells you how and where to install everything. Once CUDA is installed you can test the installation with the following command.

```
    nvcc --version
```

Your output should be similar to this if the installation was successful:

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2021 NVIDIA Corporation
Built on Thu_Nov_18_09:45:30_PST_2021
Cuda compilation tools, release 11.5, V11.5.119
Build cuda_11.5.r11.5/compiler.30672275_0
```

2 Data Collector

2.1 setup

Setting up this repository is fairly simple, you can either clone the repo or download as a zip.

```
git clone https://github.com/Research-Project-Crypto/DataCollector.git
```

you will need a binance api key and secret to get started collecting crypto price data but this can be aquired for free by making a binance account and going to the Binance API management page[1], create an API-key and put it inside the crypto.py file.

collecting price data can take a long time depending on how much data you want, to prevent API bans there is timer between each api call. you can always stop the program when you think you have enough data.

To fetch reddit data you will also need to create an application on their site that has an api key and login. This can be found here[2]

3 Ticker Timescale-swap

The ticker timescale-swap application is used for changing the timescale of ticker data.

For example you'll most likely want to convert the most accurate data you have (1min ticker) to something less specific than that (5min / 10min / etc...)

3.1 Cloning

```
git clone https://github.com/Research-Project-Crypto/TickerTimescaleSwap.git --
    recursive
```

If you forgot to clone the repository with the recursive argument you always use the following command:

```
git submodule update --init --recursive
```

3.2 Build Dependencies

The following commands are for Arch based systems, they will give you an idea on what's required to port it to other systems.

```
pacman -S --noconfirm --needed gcc make
pacman -S --noconfirm --needed premake
```

3.3 Compiling the Application

```
generate build instructions
premake5 gmake2

compile the actual application
make config=release
```

3.4 Using the Application

3.4.1 Changing time rescale ratio

By default the time rescale is 1:60, currently you can't dynamically define this through command argument, so you'll have to to edit the code directly and recompile. To change the ratio go into the src/main.cpp file and locate where the processor.start() is called. If you pass a number in the .start method you can set the time rescale.

3.4.2 Arguments

Position	Argument
1	Data Input Folder
2	Data Output Folder

Example:

```
TickerTimescaleSwap data/input_folder data/output_folder
```

3.5 Verify Data Integrity

Included with this project is a python script with which you can verify the binary output data.

```
python3 scripts/binary_reader.py
```

It will loop over all the cells slowly, this mostly to shortly verify calculation mistakes in the program.

4 Data Preprocessor

4.1 cloning

```
git clone https://github.com/Research-Project-Crypto/DataPreprocessor.git --
    recursive
```

If you forgot to clone the repository with the recursive argument you always use the following command:

```
git submodule update --init --recursive
```

4.2 Build Dependencies

The following commands are for Arch based systems, they will give you an idea on what's required to port it to other systems.

```
pacman -S --noconfirm --needed gcc make
pacman -S --noconfirm --needed premake

yay -S ta-lib
# or
paru -S ta-lib
```

4.3 Compiling the Application

```
generate build instructions
premake5 gmake2

compile the actual application
make config=release
```

4.4 Using the Application

With Arguments

Position	**Argument**
1	Data Input Folder
2	Data Output Folder

Example:

```
DataProcessor data/input_folder data/output_folder
```

Downside of argument only. With argument only mode you are unable to specify the type of the input data, you can only parse CSV text data.

Using settings.json If you don't give any arguments the application will default to reading the settings from settings.json. only use the is_binary option when you used our time rescaler since this will output to binary files to save disk space.

```json
{
    "input": {
        "input_folder": "data/input",
        "is_binary": false
    },
    "output": {
        "output_folder": "data/output"
    }
}
```

4.5 Verify Data Integrity

Included with this project is a python script with which you can verify the binary output data.

```
python3 scripts/binary_reader.py
```

It will loop over all the cells slowly, this mostly to shortly verify calculation mistakes in the program.

5 Model Testing

5.1 Setting up local environment

Since certain pip packages don't work with the newest version of Python at the time of working on our project we were required to use an older version of Python. For this reason using a venv is recommended as it's easier to enforce the used Python version.

```
python3.9 -m venv .venv
source .venv/bin/activate
```

Then install the required pip packages for running the jupyter notebooks with the following command.

```
pip install -r requirements.txt
```

6 References

[1] Binance api management settings:
https://www.binance.com/en/my/settings/api-management
[2] reddit application page https://www.reddit.com/prefs/apps