Tutorial for

ANACONDA & MACBOOK APPLE SILICON SETUP

Aamil Khan Mahar

CS 437 – Deep Learning

Table of Contents

Anaconda

Introduction:

Anaconda is a comprehensive environment and package manager for Python that simplifies the management of multiple projects by isolating them in separate environments. It is particularly useful for developers, data scientists, and researchers who work on diverse projects with varying dependencies and Python versions. Instead of modifying your system-wide Python configuration—which can lead to conflicts—Anaconda enables you to create isolated virtual environments tailored to the specific requirements of each project.

With Anaconda, you can:

- Effortlessly manage multiple Python versions for different projects.
- Install, update, and remove libraries without affecting other environments.
- Reproduce project setups with ease, ensuring consistency across machines.
- Utilize specialized tools for data analysis, machine learning, web development, or scientific computing.

Whether you're working on legacy code requiring an older Python version or experimenting with cutting-edge libraries, Anaconda provides a robust, user-friendly framework to streamline your workflows and boost productivity.

Installation:

1. Go to https://www.anaconda.com/download and download the Anaconda software on your MacBook. (Ensure you select the Apple Silicon Mac version.)
2. Once the .pkg file has downloaded, run the installer by double-clicking on it.
3. After the installation is complete, open the VSCode terminal and type conda activate to confirm that Anaconda was installed successfully.

Usage:

Make sure to activate the environment before running the python files. You can use Conda directly from your terminal. Here are some key commands (pip commands work as well but after activating):

- Create a new virtual environment:
  Command: conda create --name <env_name> python=<version>
  Description: Replace <env_name> with your desired environment name and <version> with the required Python version.

- Activate an environment:
  Command: conda activate <env_name>
  Description: This switches your terminal to the specified environment.

- Deactivate the current environment:
  Command: conda deactivate
  Description: Exits the active environment and returns to the base environment.

- List all environments:
  Command: conda env list
  Description: Displays all virtual environments on your system.

- Install a package:
  Command: conda install <package_name>
  Description: Installs the specified package in the active environment.

- Remove a package:
  Command: conda remove <package_name>
  Description: Uninstalls the specified package from the active environment.

- Create an environment from a file:
  Command: conda env create -f environment.yml
  Description: Recreates an environment using a .yml file.

- Delete an environment:
  Command: conda env remove --name <env_name>
  Description: Deletes the specified virtual environment.

PyTorch MPS

Introduction:

MPS is like Cuda for Macbook, it accelerates the model training for PyTorch based models. PyTorch uses the new Metal Performance Shaders (MPS) backend for GPU training acceleration. This MPS backend extends the PyTorch framework, providing scripts and capabilities to set up and run operations on Mac.

Installation:

1. Open the VSCode terminal to use the commands attached below to install the Miniconda MPS script.

   Command 1: curl -O https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-arm64.sh

   Command 2: sh Miniconda3-latest-MacOSX-arm64.sh

2. Open the VSCode terminal and use the command attached below to install PyTorch completely.

   Command: conda install pytorch torchvision torchaudio -c pytorch-nightly

Usage:

1. MPS Setup in python notebook:

```
import torch

device = torch.device('mps' if torch.backends.mps.is_available() else 'cpu')

print('Device:', device) # Should print 'mps' if correctly configured
```

2. MPS Usage in python notebook:

```
class CNN():
    super().__init__()
    # Model Defination here

model = CNN().to(device)

# Now you can use the model on the MPS device
```

Numpy Accelerate

Introduction:

Numpy accelerate using Blas will make your numpy computations a bit faster and hopefully help your models train faster. After running the installation steps you'll be done with this

Installation:

1. Make sure to install anaconda before this and activate the environment you want this numpy inside and run the two commands below.

   Command 1: `conda config --add channels conda-forge`

   Command 2: `conda config --set channel_priority strict`

2. Install Numpy in this environment using the command given below.

   Command: `conda install numpy "blas=*=*accelerate*"`