

Guide 1: Engaging with the INGENIOUS Project

This tutorial presents an overview of working with a Deep Learning software developed by Dr. Stephen Brown during his involvement in two research initiatives at the Great Basin Center for Geothermal Energy (GBCGE) at the University of Nevada, Reno. The primary objective is to apply machine learning (ML) techniques to create an algorithmic method for identifying new geothermal systems in the Great Basin region. This project builds upon the achievements of the Nevada geothermal play fairway project. The motivation behind this approach is the potential for an empirical, algorithm-based method to more effectively scale and perform than the original workflow used in play fairway analysis by dynamically estimating the influence assigned weights of various parameters. Key activities of the project include expanding the training dataset with additional sites (both positive and negative examples), converting the data into ML-friendly formats, and the development, as well as testing, of the ML algorithms and their results.

1.1 Download and Install the Environment

To effectively run the deep learning notebooks for this project, it's essential to install a specialized Python environment. The required environment, named 'GeothermalML2023', is available on GitHub. Start by downloading the environment file from the following URL: [GeothermalML2023 on GitHub](#).

Before proceeding with this environment installation, we recommend using the Mamba distribution of Python. Mamba is a fast, behavior-driven development (BDD) testing framework for Python, inspired by RSpec, focusing on behavior specifications to enhance readability and maintainability.

For beginners, the Miniforge distribution is suggested. In case an older version of Mamba is required, opt for the Mambaforge distribution. Notably, Miniforge comes pre-configured with the conda-forge channel, though this can be modified to accommodate any preferred channel. Download Miniforge from the following link: [Miniforge on GitHub](#).

M	Miniforge3-Windows-x86_64	12/14/2023 9:52 AM	Application	65,462 KB
---	---------------------------	--------------------	-------------	-----------

Once you have downloaded the necessary files, proceed to install the downloaded Miniforge distribution on your system. Ensure that you have at least 30GB of available disk space for the environment and software installation.

Creating the Environment

Open Miniforge Prompt: Begin by launching the Miniforge Prompt on your system.

Create a New Environment: In the Miniforge Prompt, type the following command:

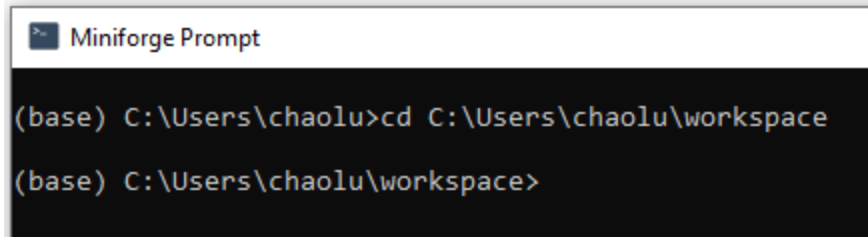
```
mamba create -n GeothermalML2023
```

This creates a new environment titled 'GeothermalML2023'. Feel free to replace 'GeothermalML2023' with a name more suited to your project.

Updating the Environment

Prepare the Environment File: Ensure that the downloaded 'GeothermalML2023.yml' file is in your current working directory. Alternatively, navigate to the directory where you have stored this file.

Example:



```
Miniforge Prompt
(base) C:\Users\chaolu>cd C:\Users\chaolu\workspace
(base) C:\Users\chaolu\workspace>
```

Update the Environment: In the Miniforge Prompt, execute the following command to update your environment:

```
mamba env update --name GeothermalML2023 --file GeothermalML2023.yml
```

Replace 'GeothermalML2023' with the name of your environment if you have chosen a different name.

Activate the Environment

To activate the environment, enter in the Miniforge Prompt:

```
conda activate GeothermalML2023
```

Deactivate the Environment

To deactivate the environment, type the following in the Miniforge Prompt:

```
conda deactivate
```

1.2 Install Python Integrated Development Environments

Python Integrated Development Environments (IDEs) are comprehensive software applications that offer a range of facilities for software development. These environments typically include a code editor, debugger, and build automation tools. By consolidating essential tools into a single, user-friendly platform, IDEs significantly enhance productivity. In this section, we will introduce two popular Python IDEs.

Spyder

Spyder (Scientific PYthon Development EnviRonment) is an open-source IDE tailored for scientific programming in Python. It enjoys widespread popularity among data scientists,

researchers, and professionals in scientific computation. To install Spyder using the Miniforge Prompt, follow these steps:

Activate Your Environment: Ensure that your desired environment is active. You can activate it with the command:

```
conda activate GeothermalML2023
```

Install Spyder: With your environment activated, install Spyder by executing:

```
conda install spyder
```

Launch Spyder: Once successfully installed, you can start Spyder by typing the command:

```
spyder
```

This process will install Spyder within your specified conda environment, ready for your scientific programming tasks.

JupyterLab

JupyterLab is an interactive, web-based development environment renowned for its versatility in handling Jupyter notebooks, code, and data. It is particularly favored in the fields of data science, scientific research, and educational sectors for its intuitive interface and powerful data visualization capabilities. To install JupyterLab using the Miniforge Prompt, follow these steps:

Activate Your Environment: Before installing JupyterLab, make sure your desired conda environment is active. Activate your environment using the command:

```
conda activate GeothermalML2023
```

Install JupyterLab: With the environment activated, install JupyterLab by running:

```
conda install -c conda-forge jupyterlab
```

Launch JupyterLab: After the installation is complete, start JupyterLab with the command:

```
jupyter lab
```

Upon executing this command, JupyterLab will open in your default web browser, providing you with a robust platform for developing and visualizing your Python projects.

1.3 Computing on Central Processing Unit or Graphics Processing Units

In our project, we utilize PyTorch due to its flexibility, ease of use, and robust performance. It's particularly beneficial for developing and iterating on complex models, allowing for rapid prototyping and experimentation. The dynamic nature of its computation graph and its seamless integration with Python have made it an invaluable tool in our machine learning and AI development efforts.

PyTorch can be installed and used on various operating systems. Depending on your system and computing requirements, your experience with PyTorch may vary in terms of processing time. Visit [PyTorch's official installation page](#) to find the section suitable for your computer.

Installing on Windows

In this example, we demonstrate the installation process on a Windows computer equipped with an NVIDIA GPU and CUDA. If your machine has a CUDA-enabled GPU, follow these steps for installation, verification, and debugging:

Activate Your Environment: Before installing PyTorch, ensure your desired conda environment is active. Activate your environment using the command:

```
conda activate GeothermalML2023
```

Install PyTorch: With the environment activated, install PyTorch by running:

```
pip3 install torch torchvision torchaudio --index-url
```

```
https://download.pytorch.org/whl/cu118
```

Note: This command varies depending on your system and the CUDA version required.

Verification: To check if your GPU driver and CUDA are enabled and accessible by PyTorch, run the following commands in your Python console:

```
import torch
torch.cuda.is_available()
```

If the result is **'True'**, your GPU is ready for processing.

If the result is **'False'**, and your computer does not support CUDA or a GPU, use the following code to set CPU as your device:

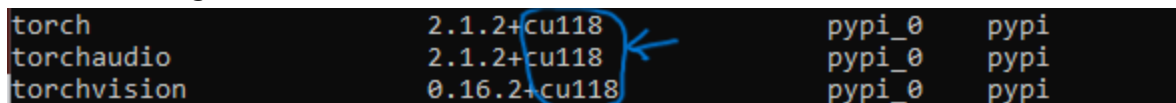
```
DEVICE = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
```

If the result is **'False'**, but your computer is capable of using CUDA or a GPU, proceed to the debugging steps below.

Debugging: Ensure your conda environment is active. Then, type the command:

```
conda list
```

Check the versions of your torch packages in the environment. You should see something similar to the figure shown below:



torch	2.1.2+cu118	pypi_0	pypi
torchaudio	2.1.2+cu118	pypi_0	pypi
torchvision	0.16.2+cu118	pypi_0	pypi

If your torch version does not include **'cu118'**, you may have installed the CPU version of torch.

Uninstall your current torch version using the command:

```
pip uninstall torch torchvision torchaudio
```

Then reinstall the CUDA version of torch using the command:

```
pip3 install torch torchvision torchaudio --index-url
```

```
https://download.pytorch.org/whl/cu118
```

After this, verify the installation again.