Trainium JumpStart Fine-Tuning

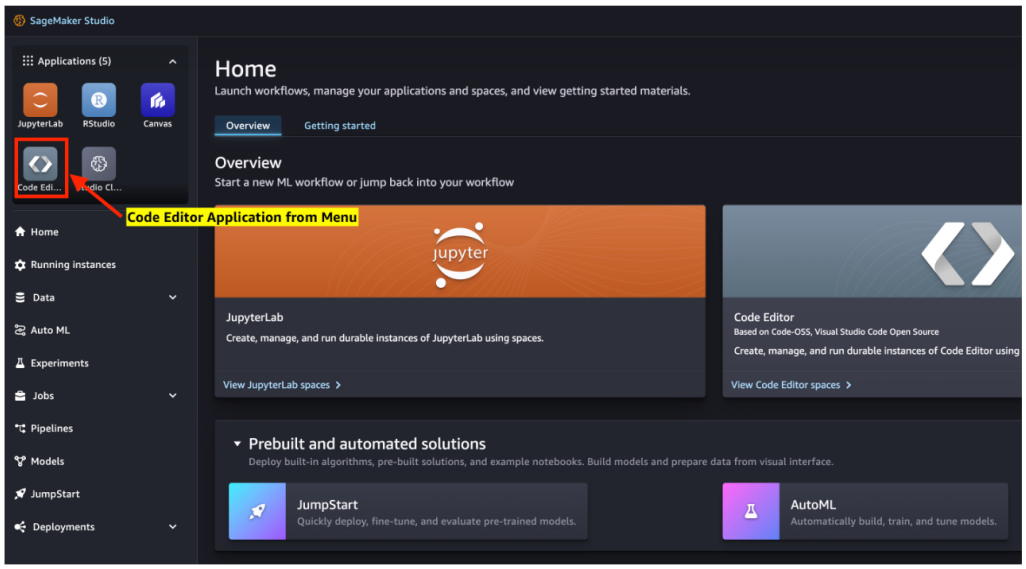
Important
Run with **Code Editor**: We're going to use scripts inside `lab-04-finetuning/trn1-finetune/*.py` notebook for this section

Contents

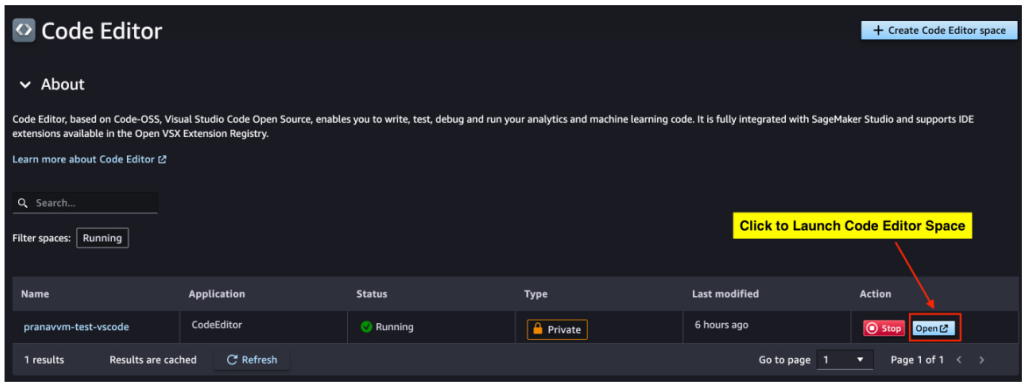
Code Editor for Fine-Tuning

For this lab, we're going to leverage SageMaker's Code Editor Open Source VSCode to fine tune a Llama 2 Model. To get to SageMaker's Code Editor,

1. Let's navigate back to Studio home



2. You should already have a Code Editor Space provisioned and running.



If you don't see a Code Editor Space, please navigate to the earlier "SageMaker Spaces: JupyterLab and Code Editor" Section of this Workshop and follow the instruction provided under `Create a VSCode Space`

Code Editor Syntax

Unlike JupyterLab, which is very heavily notebook driven and prepared for a visual workflow, Code Editor is meant for a more 'developer' styled workflow. In that spirit, we're going to be running our Fine-Tuning task in a script mode

Ensure Repo is cloned
Please ensure you have cloned the repository into your Code Editor space, as per instructions provided under SageMaker Spaces: JupyterLab and Code Editor section

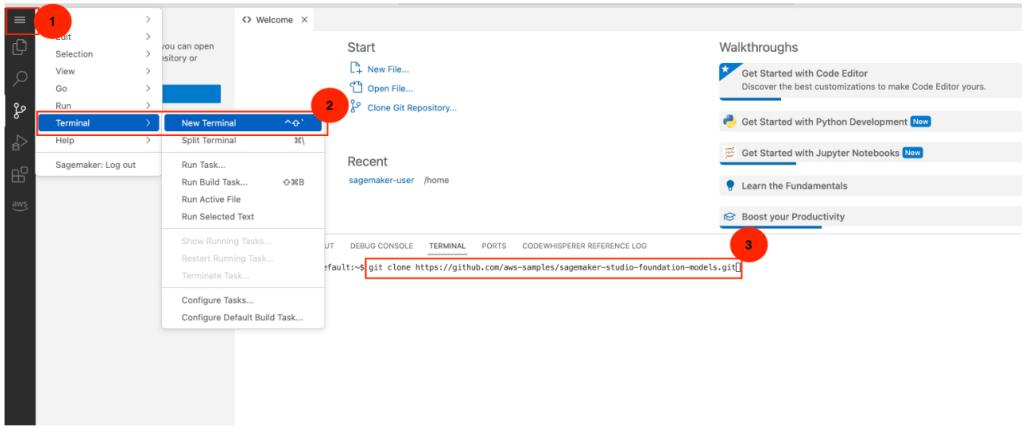
If you haven't cloned the repository into Code Editor Space, the follow the instructions below to do so,

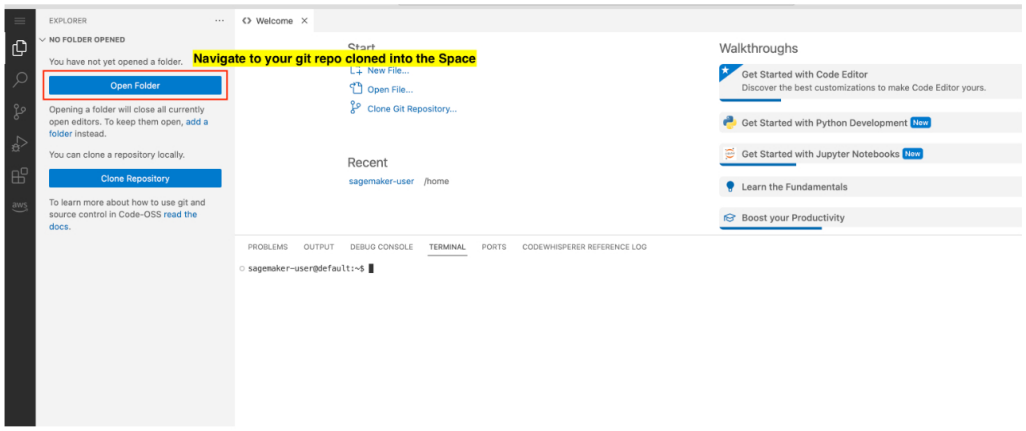
1. Launch a new terminal
2. Paste the git repo into the terminal to clone the repository into your Space

```
git clone https://github.com/aws-samples/sagemaker-studio-foundation-models.git
```

3. Navigate to `lab-04-finetuning/trn1-finetune/` like below from your File Explorer or from your terminal,

```
cd ~/sagemaker-studio-foundation-models/lab-04-finetuning/trn1-finetune/
```





Dataset

You can fine-tune on the dataset with `domain adaptation` format or `instruction tuning` format. Below are the instructions for how the training data should be formatted for input to the model.

To fine tune an LLM you are going to need a directory containing either a JSON lines (`.jsonl`) or text (`.txt`) formatted file.

- For JSON lines (JSONL) file, each line is a dictionary, representing a dictionary. The key in dictionary (each line) has to be 'text'.
- The number of files under train directory should equal to one.

In this demo, we will use a subset of Dolly dataset in an instruction tuning format. Dolly dataset contains roughly 15,000 instruction following records for various categories such as question answering, summarization, information extraction etc. It is available under Apache 2.0 license. We will select the summarization examples for fine-tuning. You can extend this lab fine-tuning example to any dataset you prefer.

We're going to move this data to `s3` to ensure our training job is able to access this data for training. However, you also have the option to,

1. Use EFS as data source
2. Use FSXLustre as data source

Fine-Tuning on AWS Trainium, Deploy on Inferentia2 and Validate Model Inference

We're going to use Llama 2 model already available on JS for fine-tuning.

For Trainium based fine-tuning

For Trainium based fine-tuning

```
model_id = "meta-textgenerationneuron-llama-2-13b"`
model_version = "1.*"
```

You can kick start a fine-tuning job using JumpStart SageMaker API and your custom dataset, with just few lines of code.

Fine-Tuning on AWS Trainium using Code Editor Terminal

Navigate to `lab-04-finetuning/trn1-finetune/` like below from your File Explorer or from your terminal,

```
cd ~/sagemaker-studio-foundation-models/lab-04-finetuning/trn1-finetune/
```

We are going to be running `fine_tune_llama2_trn1.py` script from our terminal but before that, we have a setup process to install some pre-requisite packages. Please run the following commands in sequence,

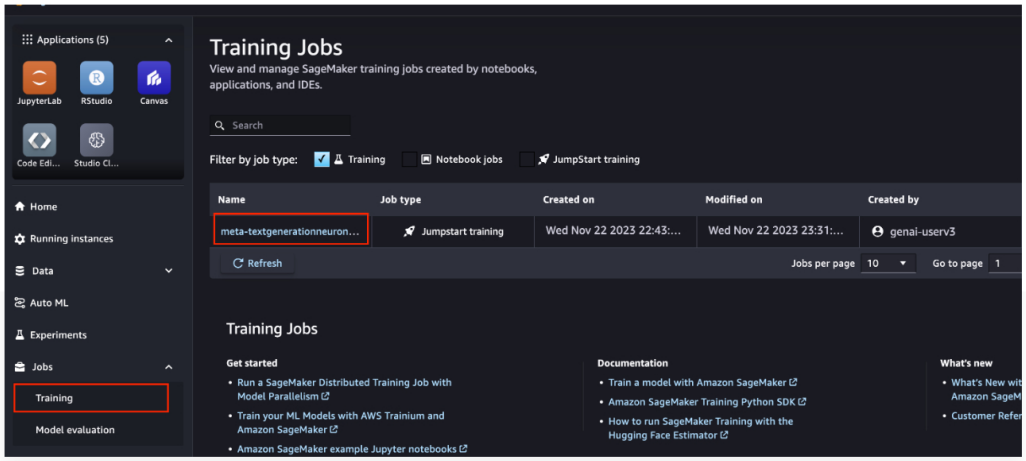
1. Setup

```
sh setup.sh
```

2. Kick off fine tuning job

```
python3 fine_tune_llama2_trn1.py \
    --dataset_name=databricks/databricks-dolly-15k \
    --js_hf_model_id=meta-textgenerationneuron-llama-2-13b \
    --js_hf_model_version=1.* \
    --max_steps=2 \
    --lr=0.0001 \
    --batch_size=1000 \
    --instance_type=ml.trn1.32xlarge
```

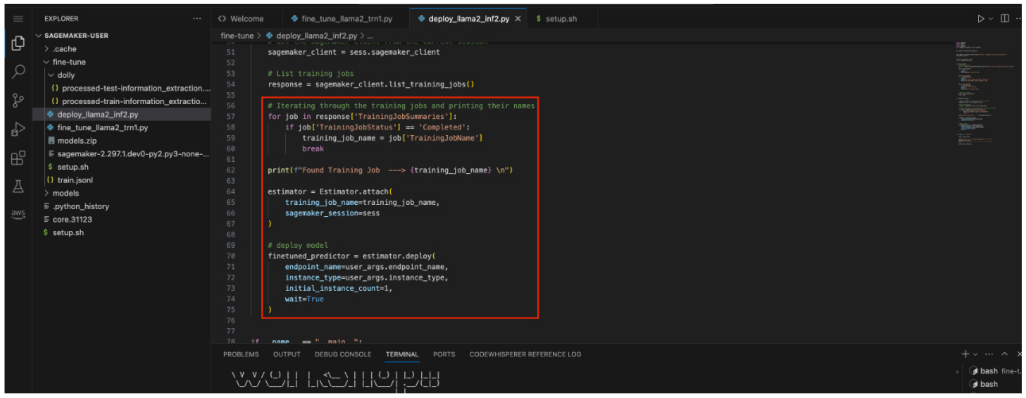
3. View your training job `In-Progress` or `Completed` from here



Deploy Model on Inferentia2

Once the training job is complete, we can simply `attach` to our training job and simply call `.deploy()` to host our fine-tuned model to a SageMaker endpoint.

```
python3 deploy_llama2_inf2.py \
    --endpoint_name=ft-meta-llama2-13b-neuron-chat-tg-ep \
    --instance_type=ml.inf2.24xlarge
```



Once the endpoint is `In Service` we can being inference tasks using this new fine-tuned endpoint!

Validate Model Inference

It is recommended to run evaluation on your new fine-tuned endpoint and your JumpStart endpoint and compare the responses. You should see an improvement in the model's responses.

Below is a table of sample responses from Llama 2 model pre and post fine-tuning.

Inputs	Ground Truth	Non Fine Tuned Model Response	Fine Tuned Model Response
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\nBased on the paragraph about the Korean War, what is the name of the new sovereign state created in the north?\n\n### Input:\nIn 1910, Imperial Japan annexed Korea, where it ruled for 35 years until its surrender at the end of World War II on 15 August 1945. The United States and the Soviet Union divided Korea along the 38th parallel into two zones of occupation. The Soviets administered the northern zone and the Americans administered the southern zone. In 1948, as a result of Cold War tensions, the occupation zones became two sovereign states. A socialist state, the Democratic People's Republic of Korea, was established in the north under the totalitarian communist leadership of Kim Il-sung, while a capitalist state, the Republic of Korea, was established in the south under the autocratic leadership of Syngman Rhee. Both governments of the two new Korean states claimed to be the sole legitimate government of all of Korea, and neither accepted the border as permanent.\n\n### Response:\n	Democratic People's Republic of Korea	The name of the new sovereign state created in the north is the Democratic People's Republic of Korea.	North Korea\n\n
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\nDoes "outbreeding" or "inbreeding" benefit the offspring more?\n\n### Input:\nExogamy often results in two individuals that are not closely genetically related marrying each other; that is, outbreeding as opposed to inbreeding. In moderation, this benefits the offspring as it reduces the risk of the offspring inheriting two copies of a defective gene. Increasing the genetic diversity of the offspring improves the chances of offspring reproducing, up until the fourth-cousin level of relatedness; however, reproduction between individuals on the fourth-cousin level of relatedness decreases evolutionarily fitness.\n\n### Response:\n	"Outbreeding" is more beneficial to the offspring as it reduces the risk of inheriting defective genes, increasing the offspring's ability to reproduce.	Inbreeding is more beneficial to the offspring than outbreeding. Inbreeding is the mating of two individuals that are closely related. Inbreeding is more beneficial to the offspring than outbreeding because it reduces the risk of the offspring inheriting two copies of a defective gene. Increasing the genetic diversity of the offspring improves the chances of offspring reproducing, up until the fourth-cousin level of relatedness;	Outbreeding benefits the offspring more.\n\n
Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\nWhich programming languages did Netscape try to add to their browser in order to make it more dynamic? \n\n### Input:\nThe first popular web browser with a graphical user interface, Mosaic, was released in 1993. Accessible to non-technical people, it played a prominent role in the rapid growth of the nascent World Wide Web. The lead developers of Mosaic then founded the Netscape corporation, which released a more polished browser, Netscape Navigator, in 1994. This quickly became the most-used.\n\nDuring these formative years of the Web, web pages could only be static, lacking the capability for dynamic behavior after the page was loaded in the browser. There was a desire in the flourishing web development scene to remove this limitation, so in 1995, Netscape decided to add a scripting language to Navigator. They pursued two routes to achieve this: collaborating with Sun Microsystems to embed the Java programming language, while also hiring Brendan Eich to embed the Scheme language.\n\n### Response:\n	The programming languages Netscape tried to add to their browser to make it more dynamic were Java and Scheme.\n\nIn 1995, Netscape pursued two routes to add dynamic behavior to their browser, Netscape Navigator:\n1. Collaborating with Sun Microsystems to embed the Java programming language\n2. Hiring Brendan Eich to embed the Scheme language	Netscape tried to add the Java programming language and the Scheme language to their browser in order to make it more dynamic.	Netscape attempted to add the Java and Scheme programming languages to their browser in order to add dynamic behavior after the page was loaded.\n\n

Lab 4 Complete!
Congratulations, You've successfully completed Lab 4!

Previous Next