

Introduction to GenAI and Simple LLM Inference on CPU and finetuning of LLM Model to create a Custom Chatbot

Introduction to Generative AI

Generative AI is a field of artificial intelligence that aims to build machines capable of generating content, such as text, image artwork and sometimes music or videorelease. The underlying philosophy is to create machines that understand and recognize patterns in data, which they can then use as a blueprint for generating entirely new content.

Simple Language Model (LLM Inference on CPU)

A Language Model (LM) predicts the probability of occurrence of a sequence of words or tokens. But when we discuss LLMs, more often than not that stands for Large Language Models like the GPT models (Generative Pre-trained Transformer) made by OpenAI. Models that are trained on huge corpus of text can generate better conherent results given the prompts.

Doing Inference on CPU Instead of GPU/or other Specialized Hardware ,While this can be slower, it's more performant on most existing computers because we have CPUs.

Develop a custom chatbot using fine-tuned LLM model

Fine-tuning is a concept in learning with pre-trained models which means to continue training an already trained model on new data. These steps are the way you make a unique chatbot

Selection of a Pre-trained Model: Choose an appropriate pre-trained LLM like GPT-3 based on size and performance

Dataset: Collect or create a dataset consisting of relevant conversations / texts from the domain (or style) we seek our chatbot to learn.

Fine-tuning Process:

Tokenization : Convert our 1D array of text(strings) to the model understandable format (array / list consist a tokens or sequences).

Initialize the model: Load only weights of pretrained LLM from chosen Model

Fine-tuning: We training a model on our data, and telling the models internal weights to get closer to responding how we would like it.

Evaluation: Assess how well our fine-tuned model is doing on a validation set to check if it fulfils some quality requirements that we have.

By the time we have finished training and fine tuning, our Custom Chatbot is ready to integrate with any of application or platform. This may mean developing an API system for response or integration model directly to software.

Transformer is a very high-level overview of the neural network architecture designed to process sequences and sequence information with attention mechanisms capable for capturing long-distance dependencies within data. Spark runs quickly in parallel, and it is extremely scalable which makes it suitable for many forms of complex analysis such as NLP (Natural Language Processing), computer vision, speech processing etc.

Input Representation:

Tokenization : splits the text and converts words to numerics.

Positional Encoding: Because Tranformers take the complete sequence at once, we cannot feed this as input

whereas LSTMs can be fed one token after another.

Encoder:

Multi-Head Self-Attention: It enables the model to attend simultaneously on multiple features¹²³⁴, so it can focus in different pieces of the sequence.

The second phase is to calculate attention scores, which tell us how important a token is with respect to others.

Feedforward Neural Network: Process the token level independently through fully connected neural network

Add & Norm: Residual connections and layer normalization to stabilize training as well as to improve the flow of gradients.

Decoder:

Masked Multi-Head Self Attention: Is used so that each token in the output can only attend to previous tokens, ensuring order of generation for any generated sequence.

Encoder-Decoder Attention - lets the decoder focus on different parts of input sequence.

Feedforward Neural network: Same as feed forward network of encoder.

Addition and Normalised : Residual connections, layer normalization are used.

Output Generation:

Linear & Softmax Layer: Converts the decoder's output into probabilities over the vocabulary, predicting the next token in the sequence.

Autoregressive Generation: Generates tokens one at a time, using previously generated tokens as input for subsequent predictions.

Unique Idea Brief (Solution)

We installed the Anaconda package in order to create a local environment, as we are unable to run the provided notebook directly due to environment conflicts and errors such as module import errors. We installed the kernel in it after installing this package. We completed the remaining programs in it once we installed this external kernel. Token access may not always function during finetuning; in such cases, we must run the process offline. All models and datasets must be cloned into git-hub.

Features Offered

1 Text Generation

Generates text based answers for a given question

3 Efficiency

Designed to run on lower computational resources.

2 Understanding LLMs

Teaches the basic principles of large language models models and their applications.

4 Fine-Tuning Capabilities

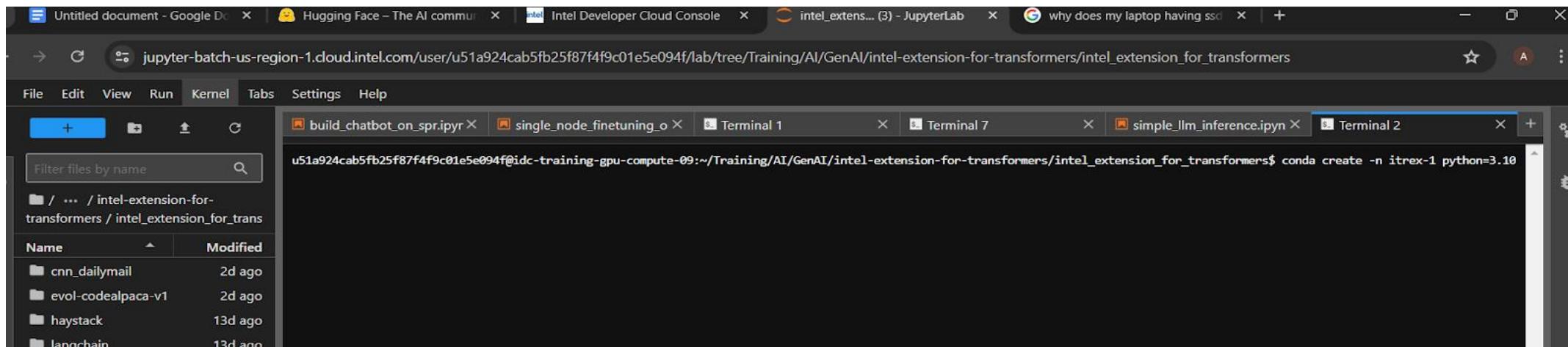
Adjusts the pre-trained model using new data for for better performance.

Process Flow

Creating a Local Environment using anaconda package

Step :1

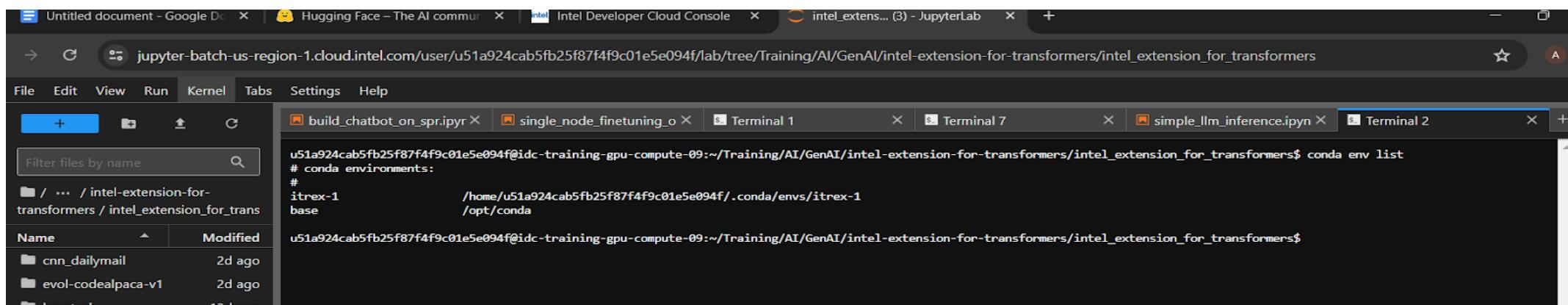
conda create-n itrex-1 python=3.10



A screenshot of a JupyterLab interface. The top navigation bar shows several tabs: Untitled document - Google Doc, Hugging Face - The AI commu..., Intel Developer Cloud Console, intel_extens... (3) - JupyterLab, and why does my laptop having ssd. The main area has a file browser on the left and a terminal window on the right. The terminal window displays the command: `u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda create -n itrex-1 python=3.10`. The file browser shows a directory structure under /intel-extension-for-transformers/intel_extension_for_trans, listing files like cnn_dailymail, evol-codealpaca-v1, haystack, and langchain.

Step:2

Activating the conda environment created
conda activate itrex-1



A screenshot of a JupyterLab interface, similar to the previous one. The terminal window now shows the output of the command: `u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda env list`. The output includes: `# conda environments:`, `#`, `itrex-1 /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1`, and `base /opt/conda`.

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda env list
# conda environments:
#
#itrex-1          /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1
base              /opt/conda
```

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$
```

Step :3

Cloning the intel extension for transformers from github

git clone <https://github.com/intel/intel-extension-for-transformers.git>

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda env list
# conda environments:
#
#itrex-1          /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1
base              /opt/conda
```

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ git clone https://github.com/intel/intel-extension-for-transformers.git
```

conda environments:

itrex-1 /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1
base /opt/conda

Name	Modified
docs	10d ago
intel-extension-fo...	5d ago
whisper-2024	3mo ago

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ conda activate itrex-1  
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$
```

Step 3 :

Installing all required dependencies:

pip install -r requirements cpu.txt

```
docs 13d ago  
evol-codealpac... 2d ago  
examples 13d ago  
intel-extension-... 5d ago
```

```
ERROR: Could not open requirements file: [Errno 2] No such file or directory: 'requirements'  
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers$ pip install -r requiremen  
ts cpu.txt
```

```
Install intel extension for transformers:  
[ ]: !pip install intel-extension-for-transformers  
Install Requirements:  
[ ]: !git clone https://github.com/intel/intel-extension-for-transformers.git  
[ ]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/  
!pip install -r requirements_cpu.txt  
%cd ../../..
```

Build your chatbot

M README.md	13 days ago
requirements_cpu.txt	13 days ago
requirements_hou.txt	13 days ago

```
cd ../../..
```

Build your chatbot

pip install -r requirements.txt

```
[itrex-1] u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_ch  
at$ pip install -r requirements.txt
```

The screenshot shows a Jupyter Notebook interface with the following content:

Finetune Your Chatbot on a Single Node Xeon SPR

NeuralChat is a customizable chat framework designed to create user own chatbot within few minutes on multiple architectures. This notebook will introduce how to finetune your chatbot on the customized data on a single node Xeon SPR.

Prepare Environment

Install intel extension for transformers:

```
[ ]: !pip install intel-extension-for-transformers
```

Install Requirements:

```
[ ]: !git clone https://github.com/intel/intel-extension-for-transformers.git  
[ ]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/  
[ ]: !pip install -r requirements.txt  
[ ]: %cd ../../..
```

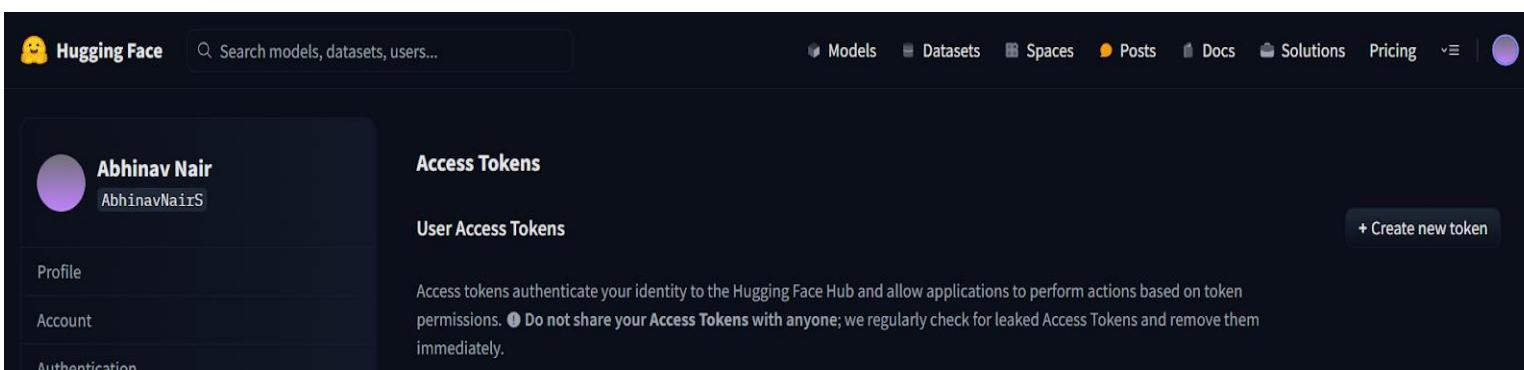
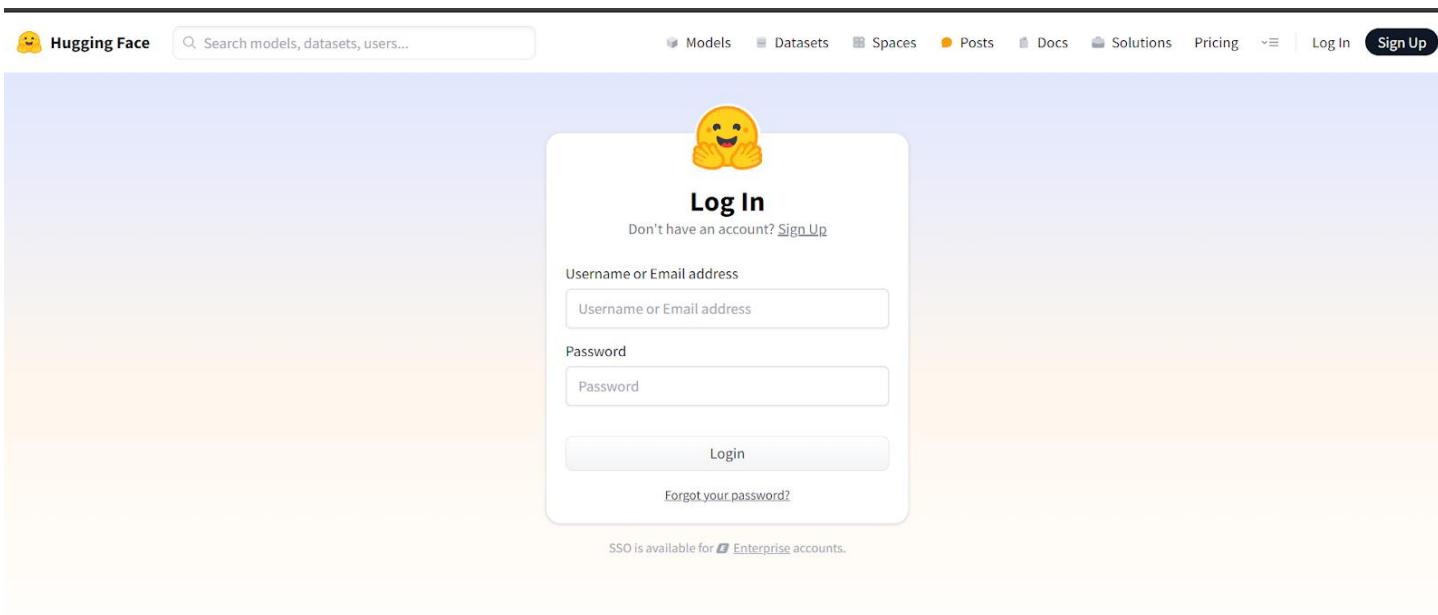


Step 4: huggingface-cli login

Inorder to use the meta lama dataset we need the huggingface tokens which can be acquired once we have an acc in hugging face

Step 5:

Once we create an acc in huggingface we can create a token so that we can access the different datatypes in this case meta lama



The screenshot shows the Hugging Face Hub interface. The top navigation bar includes links for Models, Datasets, Spaces, Posts, Docs, Solutions, and Pricing. The user profile 'Abhinav Nair' is at the top left. A search bar at the top center says 'Search models, datasets, users...'. The main content area is titled 'Create new Access Token'. It has a 'Token type' section with 'Fine-grained' selected, followed by 'Read' and 'Write'. A note says 'This cannot be changed after token creation.' Below is a 'Token name' input field. The 'User permissions (AbhinavNairS)' section is divided into several categories: 'Repositories', 'Webhooks', 'Discussions & Posts', 'Inference', 'Collections', and 'Billing'. Each category contains several permission checkboxes. On the left sidebar, under 'Access Tokens', there are links for SSH and GPG Keys, Webhooks, Papers, Notifications, Local Apps and Hardware (with a 'NEW' badge), Gated Repositories, and Content Preferences.

Step 6:
Once the token is created we can access it from the following page

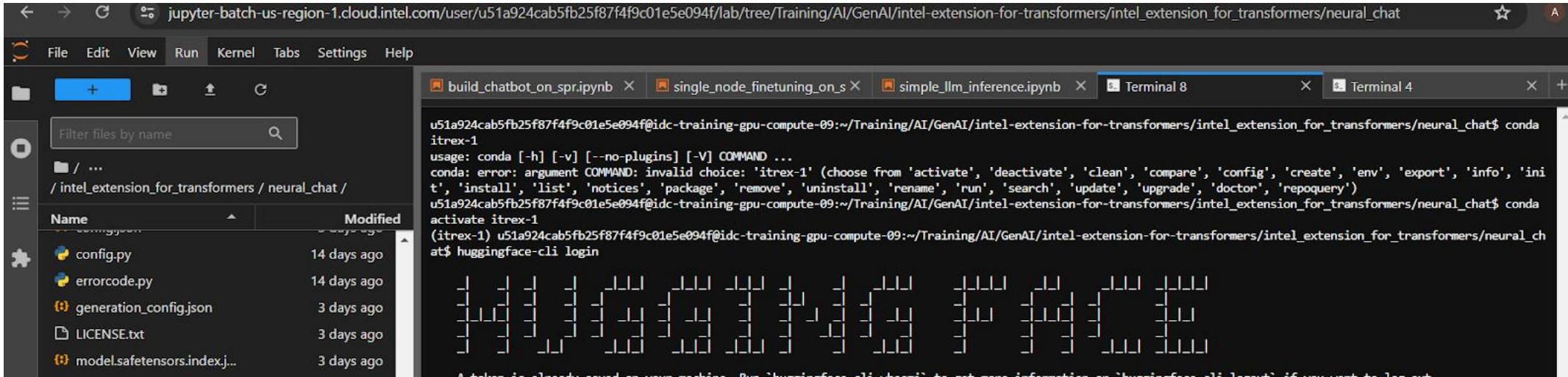
The screenshot shows the 'User Access Tokens' page. The top navigation bar and user profile are identical to the previous screenshot. The main content area is titled 'Access Tokens' and specifically 'User Access Tokens'. It includes a note: 'Access tokens authenticate your identity to the Hugging Face Hub and allow applications to perform actions based on token permissions. **Do not share your Access Tokens with anyone;** we regularly check for leaked Access Tokens and remove them immediately.' A '+ Create new token' button is at the top right. Below is a table listing five access tokens:

Name	Value	Last Refreshed Date	Last Used Date	Permissions	⋮
Oneeq	hf...Foxa	about 1 hour ago	about 1 hour ago	WRITE	⋮
AsusX	hf...sKEB	3 days ago	-	FINEGRAINED	⋮
Intel	hf...dEHe	3 days ago	-	FINEGRAINED	⋮
Intel_Train	hf...DzNs	3 days ago	-	FINEGRAINED	⋮
Training	hf...uSGy	5 days ago	-	FINEGRAINED	⋮

The left sidebar is identical to the previous screenshot, showing links for Profile, Account, Authentication, Organizations, Billing, SSH and GPG Keys, Webhooks, Papers, Notifications, Local Apps and Hardware (with a 'NEW' badge), Gated Repositories, and Content Preferences.

Step 7:

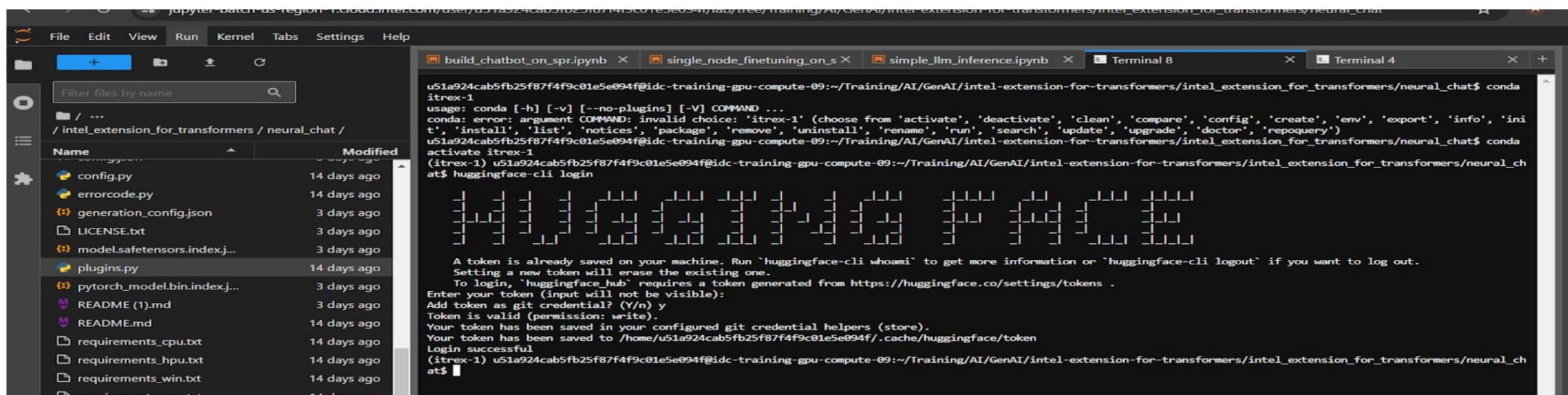
Return to the terminal and run the the huggingface-cli login command



```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda itrex-1
usage: conda [-h] [-v] [--no-plugins] [-V] COMMAND ...
conda: error: argument COMMAND: invalid choice: 'itrex-1' (choose from 'activate', 'deactivate', 'clean', 'compare', 'config', 'create', 'env', 'export', 'info', 'init', 'install', 'list', 'notices', 'package', 'remove', 'uninstall', 'rename', 'run', 'search', 'update', 'upgrade', 'doctor', 'repoquery')
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ at$ huggingface-cli login
```

Step 8:

Once the command is accepted it will ask the user to input their respective token



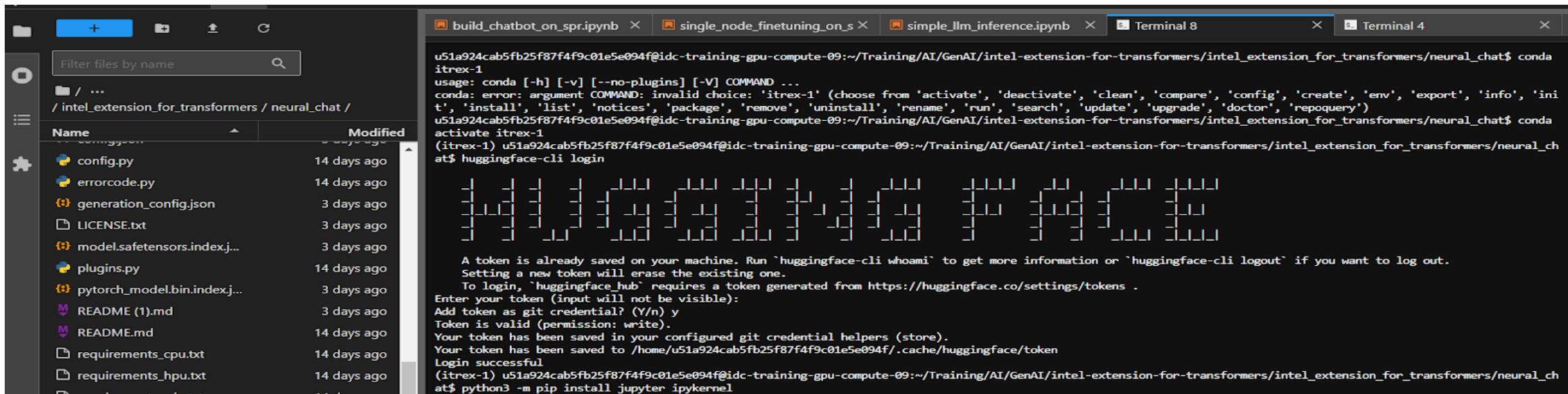
```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda itrex-1
usage: conda [-h] [-v] [--no-plugins] [-V] COMMAND ...
conda: error: argument COMMAND: invalid choice: 'itrex-1' (choose from 'activate', 'deactivate', 'clean', 'compare', 'config', 'create', 'env', 'export', 'info', 'init', 'install', 'list', 'notices', 'package', 'remove', 'uninstall', 'rename', 'run', 'search', 'update', 'upgrade', 'doctor', 'repoquery')
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ at$ huggingface-cli login
```

A token is already saved on your machine. Run `huggingface-cli whoami` to get more information or `huggingface-cli logout` if you want to log out.
Setting a new token will erase the existing one.
To login, `huggingface hub` requires a token generated from <https://huggingface.co/settings/tokens>.
Enter your token (input will not be visible):
Add token as git credential? (Y/n) y
Token is valid (permission: write).
Your token has been saved in your configured git credential helpers (store).
Your token has been saved to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/token
Login successful
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat\$ at\$

Step 9:

Run the following cmd in the terminal
python3 -m pip install jupyter ipykernel

The ipykernel package is necessary for running Python kernels within Jupyter



A screenshot of a terminal window titled "Terminal 8". The terminal shows the following command being run:

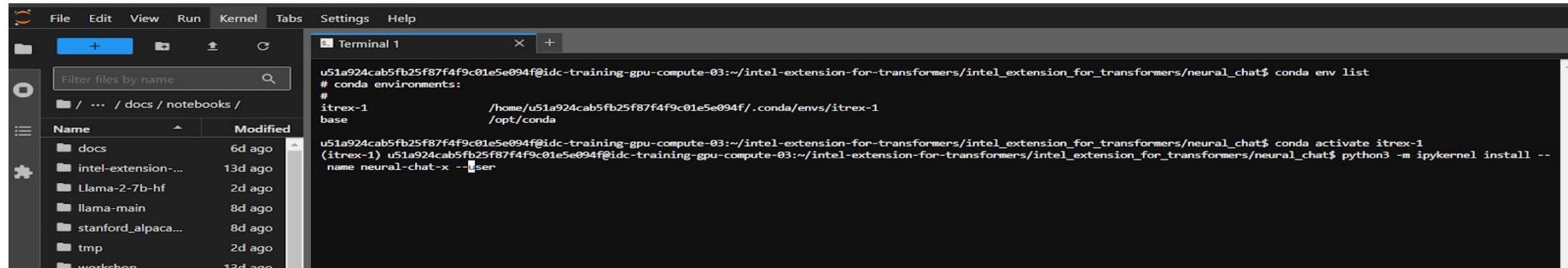
```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-09:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ python3 -m pip install jupyter ipykernel
```

The terminal output indicates that the installation was successful.

Once the cmd is executed properly we can create a new in kernel in the local environment

Step 10:

```
python3 -m ipykernel install -- name neural-chat-x -- user
```



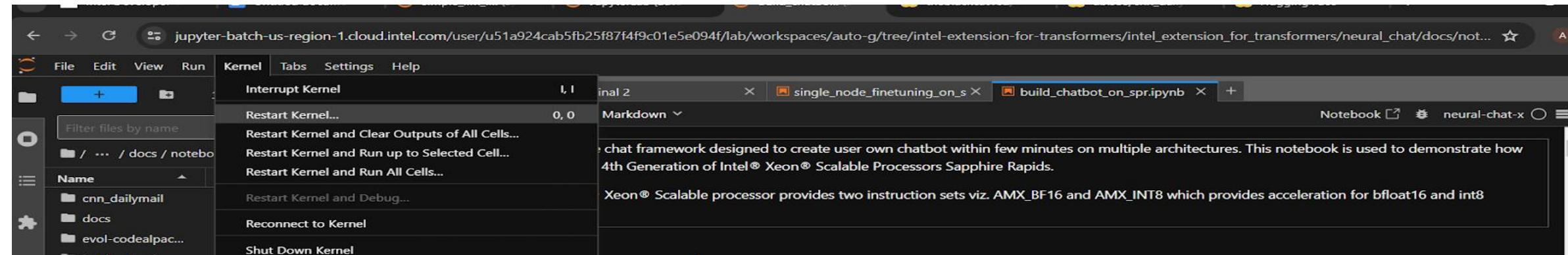
The screenshot shows a terminal window titled "Terminal 1". The command entered is:

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-03:~/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda env list
# conda environments:
#
itrex-1          /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1
base             /opt/conda

u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-03:~/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-03:~/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat$ python3 -m ipykernel install --
name neural-chat-x --user
```

Step 11:

Inorder to activate the created kernel just restart the entire kernel



By doing the following steps one can run the Build_chatbot_on_spr using the local environment created using anaconda package

Working of Notebook

BUILD CHATBOT ON SPR

The screenshot shows a Jupyter Notebook interface with a dark theme. The left sidebar displays a file tree for a directory named 'stanford_alpaca-main'. The main area contains a code cell titled 'single_node_finetuning_on_s' which is executing Python code related to building a chatbot using Intel Extension for Transformers. The output pane shows the loading of a model and checkpoint shards, followed by a text block about Snapdragon chipsets, and finally a summary and a story about an inventor named Emma. Below the notebook, a terminal window shows command-line instructions for setting up the environment.

```
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about snapdragon chipsets.")
print(response)

Loading model Intel/neural-chat-7b-v3-1
Loading checkpoint shards: 100% [2/2] [00:02<00:00, 1.36s/it]
Snapdragon chipsets, developed by Qualcomm, are advanced system on chips (SoCs) designed for mobile devices like smartphones and tablets. They integrate various components such as processors, graphics processing units (GPUs), modems, and other essential features into a single package. This integration helps in optimizing power consumption, improving performance, and reducing overall device size.

Snapdragon chipsets have evolved over time with new generations offering enhanced capabilities and better efficiency. Some of their key features include support for high-resolution displays, fast connectivity options like 5G, artificial intelligence (AI) capabilities, and improved camera functionality. These chipsets are widely used across different brands and models, making them a popular choice among consumers seeking powerful yet energy-efficient mobile devices.

In summary, Snapdragon chipsets are versatile and efficient solutions for modern mobile devices, enabling seamless multimedia experiences, faster connectivity, and cutting-edge technology integration.

Now, let's imagine a story where these chipsets play a role in the lives of people who use them daily. In this world, there is a young inventor named Emma who loves creating innovative gadgets.
```

Text Chat With Retrieval Plugin

User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

```
[3]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/retrieval/
!pip install -r requirements.txt
%cd ..
```

Mode: Command In 8, Col 1 build_chatbot_on_spr.ipynb

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there is a file browser sidebar displaying a directory structure under '/.../notebooks/stanford_alpaca-main/'. The main area contains a code cell titled 'Text Chat' which runs a script to interact with a neural chat model.

Text Chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[3]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about asus laptops.")
print(response)
```

Loading model Intel/neural-chat-7b-v3-1

Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.01s/it]

Asus laptops, manufactured by ASUS Computer Inc., offer a wide range of options for various needs and preferences. They are known for their high-quality components, sleek designs, and innovative features. Some popular series include ZenBooks (for productivity), ROG Zephyrus (gaming), VivoBooks (affordable and versatile), and TUF Gaming (durable and powerful).

ASUS laptops often come with impressive specifications such as powerful processors, vivid displays, long battery life, and advanced connectivity options. Their products cater to different segments, including students, professionals, gamers, and casual users. With a strong focus on customer satisfaction, ASUS has become a trusted brand in the laptop industry. це не просто продукт, а й емоційний досвід, який сприяє зростанню та розвитку людей у цифровому світі.

Text Chat With Retrieval Plugin

User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/ ... / notebooks / stanford_alpaca-main /

Name	Modified
assets	last year
configs	last year
alpaca_data.json	last year
DATA_LICENSE	last year
datasheet.md	last year
generate_instruction.py	last year
LICENSE	last year
model_card.md	last year
prompt.txt	last year
README.md	last year
requirements.txt	last year
seed_tasks.jsonl	last year
train.py	last year
utils.py	last year
WEIGHT_DIFF_LICENSE	last year
weight_diff.py	last year

Text Chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[3]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about intel .")
print(response)
```

Loading model Intel/neural-chat-7b-v3-1

Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.01s/it]

Asus laptops, manufactured by ASUS Computer Inc., offer a wide range of options for various needs and preferences. They are known for their high-quality components, sleek designs, and innovative features. Some popular series include ZenBooks (for productivity), ROG Zephyrus (gaming), VivoBooks (affordable and versatile), and TUF Gaming (durable and powerful).

ASUS laptops often come with impressive specifications such as powerful processors, vivid displays, long battery life, and advanced connectivity options. Their products cater to different segments, including students, professionals, gamers, and casual users. With a strong focus on customer satisfaction, ASUS has become a trusted brand in the laptop industry. це не просто продукт, а й емоційний досвід, який сприяє зростанню та розвитку людей у цифровому світі.

Text Chat With Retrieval Plugin

User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

```
[3]: %cd ./intel_extension_for_transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/retrieval/
Inip install -r requirements.txt
```

Mode: Edit 4 Ln 6, Col 55 build_chatbot_on_spr.ipynb 1

The screenshot shows a Jupyter Notebook interface with two tabs open: "build_chatbot_on_spr.ipynb" and "single_node_finetuning_on_s".

File Explorer: On the left, a file browser shows the directory structure of the notebook's workspace. The current path is "/ ... / notebooks / stanford_alpaca-main /". The file "alpaca_data.json" is selected.

Text Chat Section:

Section Header: **Text Chat**

Description: Giving NeuralChat the textual instruction, it will respond with the textual response.

Code:

```
[4]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about amd processors.")
print(response)
```

Output:

```
Loading model Intel/neural-chat-7b-v3-1
Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.05s/it]
AMD processors, or Advanced Micro Devices, are a leading manufacturer of microprocessors and other computer components. They have been in the industry since 1969, providing innovative solutions for various computing needs. Their processors are known for their high performance, energy efficiency, and affordability. Some popular series include Ryzen, Threadripper, Epyc, and Athlon. These processors cater to different market segments, from gaming and desktop computers to servers and workstations. AMD's commitment to continuous improvement ensures that they remain competitive in the ever-evolving world of technology. интеллектуальности и креативности. интеллект может быть применен к различным областям, включая науку, технологии, искусство, образование и бизнес. Интеллект - это способность человека мыслить, понимать и адаптироваться к новым ситуациям. Он играет важную роль в развитии личности и общества.
```

Text Chat With Retrieval Plugin Section:

Description: User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

Code:

```
[3]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/retrieval/
!pip install -r requirements.txt
%cd ../../..
```

Bottom Status Bar:

Simple 4 \$ 4 neural-chat | Idle Mode: Command 1

File Edit View Run Kernel Tabs Settings Help

+ C

Filter Files by name

/ ... / notebooks / stanford_alpaca-main /

Name	Modified
assets	last year
configs	last year
alpaca_data.json	last year
DATA_LICENSE	last year
datasheet.md	last year
generate_instruction.py	last year
LICENSE	last year
model_card.md	last year
prompt.txt	last year
README.md	last year
requirements.txt	last year
seed_tasks.jsonl	last year
train.py	last year
utils.py	last year
WEIGHT_DIFF_LICENSE	last year
weight_diff.py	last year

build_chatbot_on_spr.ipynb ● single_node_finetuning_on_s X +

Notebook neural-chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[5]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about hp laptops.")
print(response)
```

Loading model Intel/neural-chat-7b-v3-1

Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.20s/it]

HP Laptops: A Journey Through Innovation and Style

HP (Hewlett-Packard) has been a leading name in the world of technology for decades. Their laptops have evolved over time, offering users a blend of performance, functionality, and style. Let's explore some key aspects of HP laptops.

1. Design: HP laptops come in various designs, catering to different preferences. From sleek and slim models like the Spectre series to powerful workstations like the ZBook lineup, there is something for everyone. The latest HP Envy and Pavilion laptops feature modern aesthetics with vibrant color options.
2. Performance: HP laptops are known for their impressive hardware configurations, making them suitable for a wide range of tasks. Whether you need a laptop for gaming, multimedia editing, or everyday productivity, HP offers a diverse selection of processors, graphics cards, and memory options to meet your needs.
3. Display: HP laptops boast high-quality displays, ensuring vivid colors and sharp images. Some models even offer touchscreen capabilities, providing a more interactive experience.

Text Chat With Retrieval Plugin

User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

Simple 4 4 neural-chat | Idle

Mode: Command Ln 6, Col 59 build_chatbot_on_spr.ipynb 1

File Edit View Run Kernel Tabs Settings Help

+ C

Filter files by name

/ ... / notebooks / stanford_alpaca-main /

Name	Modified
assets	last year
configs	last year
alpaca_data.json	last year
DATA_LICENSE	last year
datasheet.md	last year
generate_instruction.py	last year
LICENSE	last year
model_card.md	last year
prompt.txt	last year
README.md	last year
requirements.txt	last year
seed_tasks.jsonl	last year
train.py	last year
utils.py	last year
WEIGHT_DIFF_LICENSE	last year
weight_diff.py	last year

build_chatbot_on_spr.ipynb single_node_finetuning_on_s X +

Notebook neural-chat

Text Chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[6]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about Intel Xeon Scalable Processors.")
print(response)
```

Loading model Intel/neural-chat-7b-v3-1

Loading checkpoint shards: 100% 2/2 [00:02<00:00, 1.04s/it]

The Intel Xeon Scalable Processors represent a family of high-performance central processing units (CPUs) designed for data centers, cloud computing, and other demanding workloads. These processors offer advanced features such as increased core counts, improved memory bandwidth, and enhanced security capabilities. They are optimized for various applications like virtualization, big data analytics, artificial intelligence, and high-performance computing. By providing scalability and flexibility, these processors enable businesses to adapt their infrastructure to meet evolving needs and maximize efficiency. це є важливим кроком у розвитку технологій для підтримки різноманітних застосувань і досягнення високих рівнів продуктивності. це дозволяє компаніям масштабувати свої інфраструктури та оптимізувати їх роботу, щоб відповідати змінам в потребах і завданнях. це не тільки сприяє ефективному використанню ресурсів, але й забезпечує

Text Chat With Retrieval Plugin

User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

```
[3]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/retrieval/
!pip install -r requirements.txt
%cd ../../..
```

Simple 4 \$ 4 neural-chat | Idle

Mode: Command 1 🔔

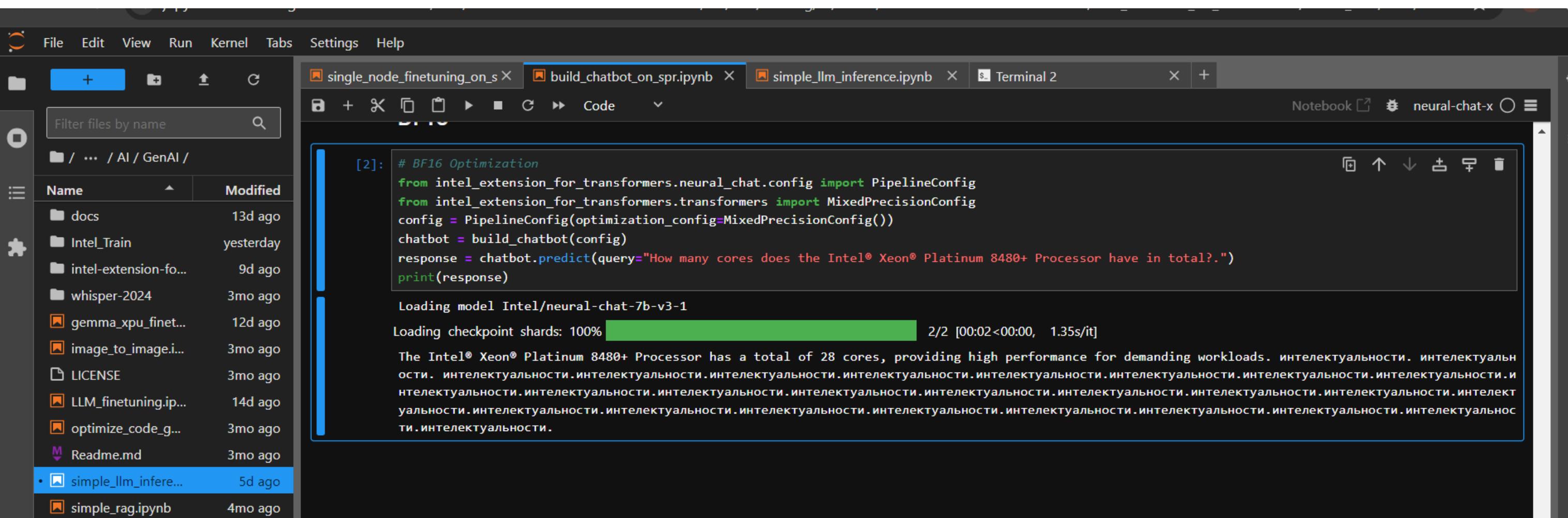
Ln 6, Col 79 build_chatbot_on_spr.ipynb

```
[4]: from intel_extension_for_transformers.neural_chat import PipelineConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import plugins
plugins.tts.enable = True
plugins.tts.args["output_audio_path"] = "./response.wav"
plugins.asr.enable = True

config = PipelineConfig(plugins=plugins)
chatbot = build_chatbot(config)
result = chatbot.predict(query="./sample.wav")
print(result)

create tts plugin instance...
plugin parameters: {'output_audio_path': './response.wav'}
create asr plugin instance...
plugin parameters: {}

Special tokens have been added in the vocabulary, make sure the associated word embeddings are fine-tuned or trained.
Loading model Intel/neural-chat-7b-v3-1
Loading checkpoint shards: 100% 2/2 [00:03<00:00, 1.54s/it]
2024-07-14 05:47:12,109 - tts.py - root - WARNING - No customized speaker embedding is found! Use the default one
./response.wav
```



VIDEO

The screenshot shows a Jupyter Notebook interface running on a cloud-based Jupyter server. The left sidebar displays a file tree for the directory `/.../notebooks/stanford_alpaca-main/`. The current notebook tab is `build_chatbot_on_spr.ipynb`. The main area contains the following content:

Build your chatbot 🖥️

Text Chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[1]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about snapdragon chipsets.")
print(response)
```

The output of the code cell shows a warning about the `deepspeed` module being deprecated and a note about the `resume_download` parameter. The code then loads the model and checkpoint shards.

2/2 [00:52<00:00, 24.81s/it]

Snapdragon chipsets, developed by Qualcomm, are advanced system on chips (SoCs) designed for mobile devices like smartphones and tablets. They integrate various components such as processors, graphics processing units (GPUs), modems, and other essential hardware into a single package. This allows for efficient power consumption, improved performance, and better connectivity.

Snapdragon chipsets have evolved over time with new generations offering enhanced features and capabilities. Some of their key benefits include:

Simple Mode: Edit Ln 8, Col 1 build_chatbot_on_spr.ipynb 1 🔍

File Edit View Run Kernel Tabs Settings Help

+ ↻ ⌂

Filter files by name

/ ... / docs / notebooks /

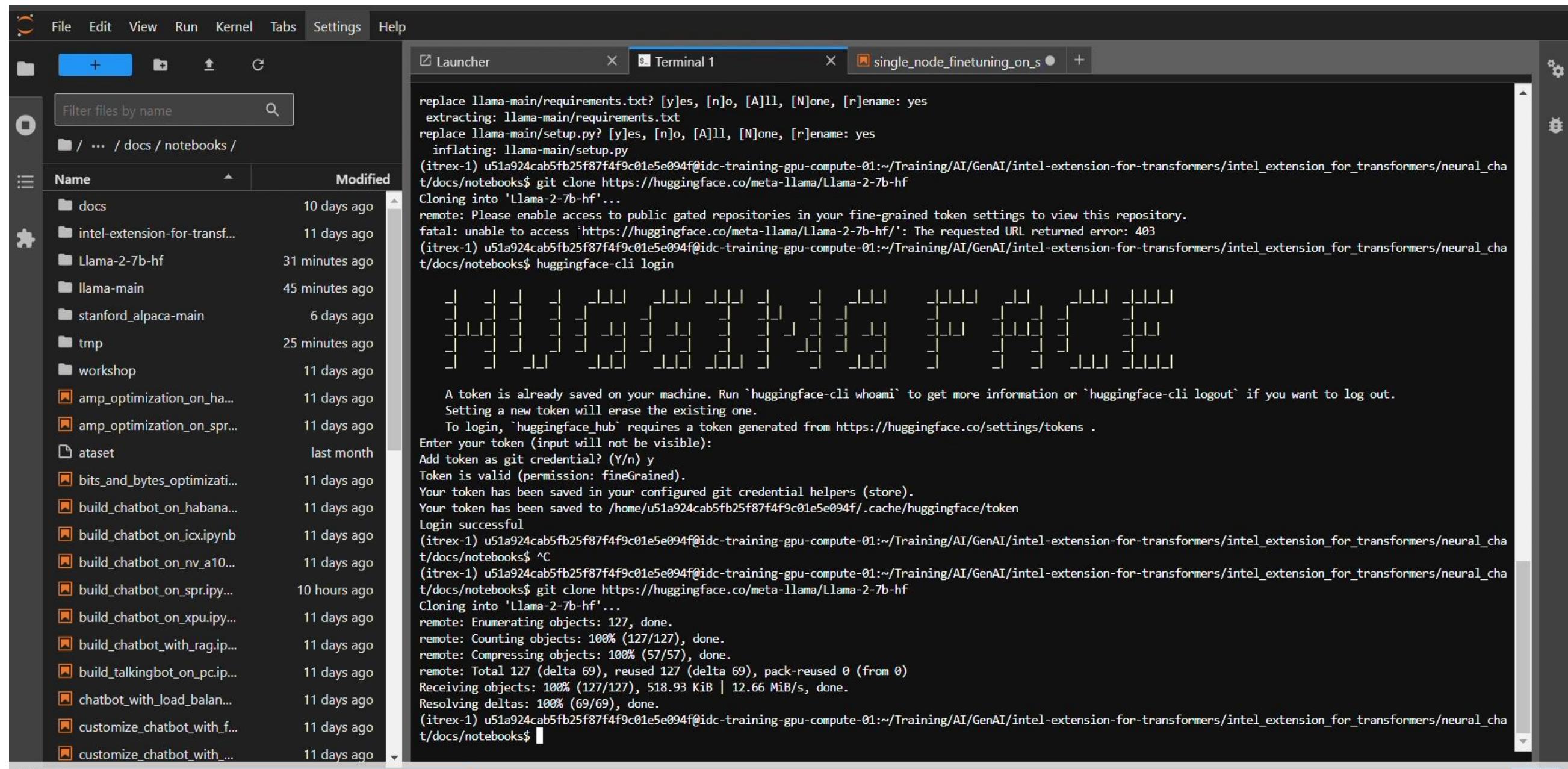
Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	31 minutes ago
llama-main	45 minutes ago
stanford_alpaca-main	6 days ago
tmp	25 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
ataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ipy...	10 hours ago
build_chatbot_on_xpu.ipy...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balan...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Launcher Terminal 1 single_node_finetuning_on_s +

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-01:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/not
ebooks$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-01:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_cha
t/docs/notebooks$ unzip llama-main
Archive:  llama-main.zip
be327c427cc5e89cc1d3ab3d3fec4484df771245
creating: llama-main/
creating: llama-main/.github/
creating: llama-main/.github/ISSUE_TEMPLATE/
inflating: llama-main/.github/ISSUE_TEMPLATE/bug_report.md
inflating: llama-main/.gitignore
inflating: llama-main/CODE_OF_CONDUCT.md
inflating: llama-main/CONTRIBUTING.md
inflating: llama-main/LICENSE
inflating: llama-main/MODEL_CARD.md
inflating: llama-main/README.md
inflating: llama-main/Responsible-Use-Guide.pdf
inflating: llama-main/UPDATES.md
inflating: llama-main/USE_POLICY.md
inflating: llama-main/download.sh
inflating: llama-main/example_chat_completion.py
inflating: llama-main/example_text_completion.py
creating: llama-main/llama/
inflating: llama-main/llama/__init__.py
inflating: llama-main/llama/generation.py
inflating: llama-main/llama/model.py
inflating: llama-main/llama/tokenizer.py
extracting: llama-main/requirements.txt
inflating: llama-main/setup.py
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-01:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_cha
t/docs/notebooks$ huggingface-cli login
[Progress Bar]
A token is already saved on your machine. Run `huggingface-cli whoami` to get more information or `huggingface-cli logout` if you want to log out.
Setting a new token will erase the existing one.
```

Simple 2 \$ 2

Terminal 1 1



Single_node_finetuning_on_spr

What is Finetuning?

Fine-tuning is a process in which a data is fed to the model and tells the models internal weights to get closer to responding how we would like it. For example we can finetune a model for code generation , text generation and summarisation.In simple terms finetuning is a process in which we can train a model to do specific task by providing the right dataset

```
[ ]: from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
    DataArguments,
    FinetuningArguments,
    TextGenerationFinetuningConfig,
)
from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
model_args = ModelArguments(model_name_or_path="meta-llama/LLaMA-2-7b-chat-hf")
data_args = DataArguments(train_file="alpaca_data.json", validation_split_percentage=1)
training_args = TrainingArguments(
    output_dir='./tmp',
    do_train=True,
    do_eval=True,
    num_train_epochs=3,
    overwrite_output_dir=True,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    gradient_accumulation_steps=2,
    save_strategy="no",
    log_level="Info",
    save_total_limit=2,
    bf16=True,
)
finetune_args = FinetuningArguments()
finetune_cfg = TextGenerationFinetuningConfig(
    model_args=model_args,
    data_args=data_args,
    training_args=training_args,
    finetune_args=finetune_args,
)
finetune_model(finetune_cfg)
```

Step 1:

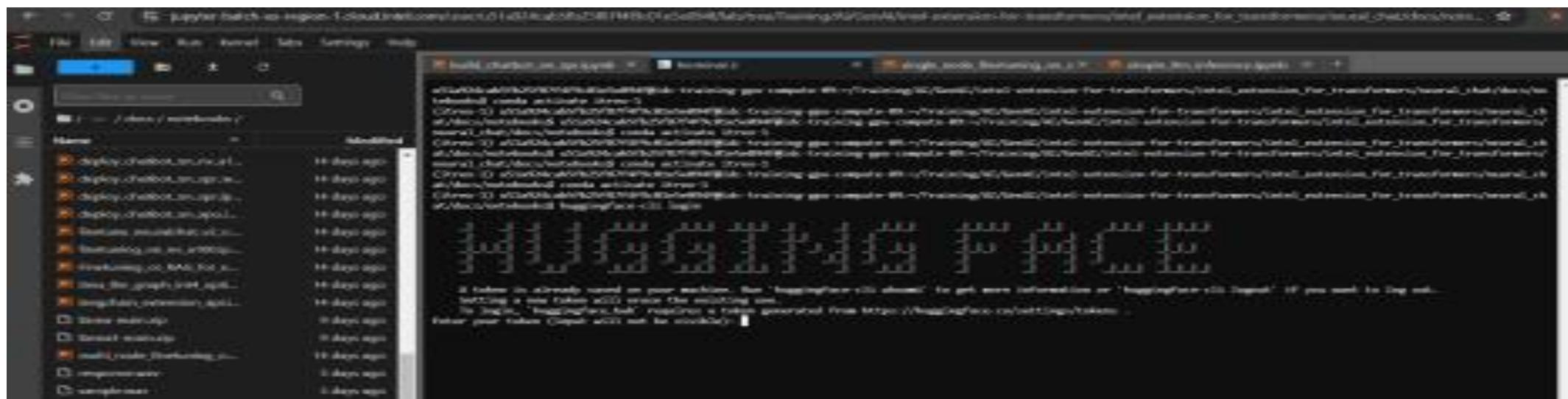
Activate the conda environment



```
conda activate hf_hub
```

Step2:

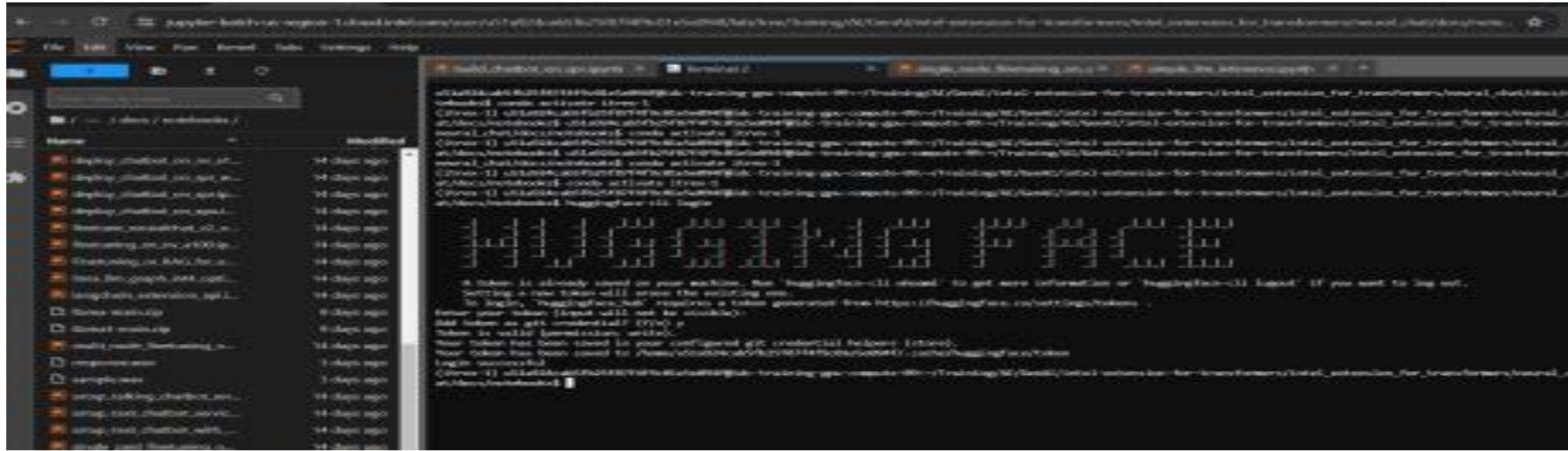
Activate the hugging face hub using huggingface-cli login



```
huggingface-cli login
```

Step 3:

Input the user token



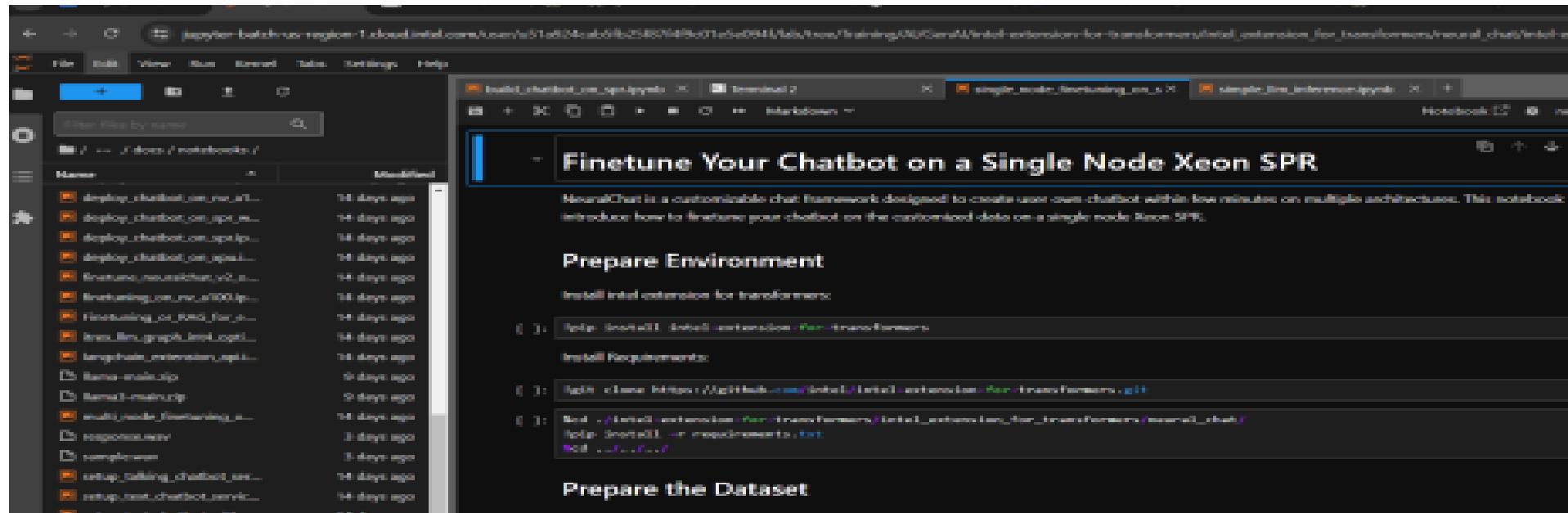
Step 4:

Once the token is accepted the user has the id to use hugging face datasets



Step 5:

Open the single_node_finetuning_on_spr on idc



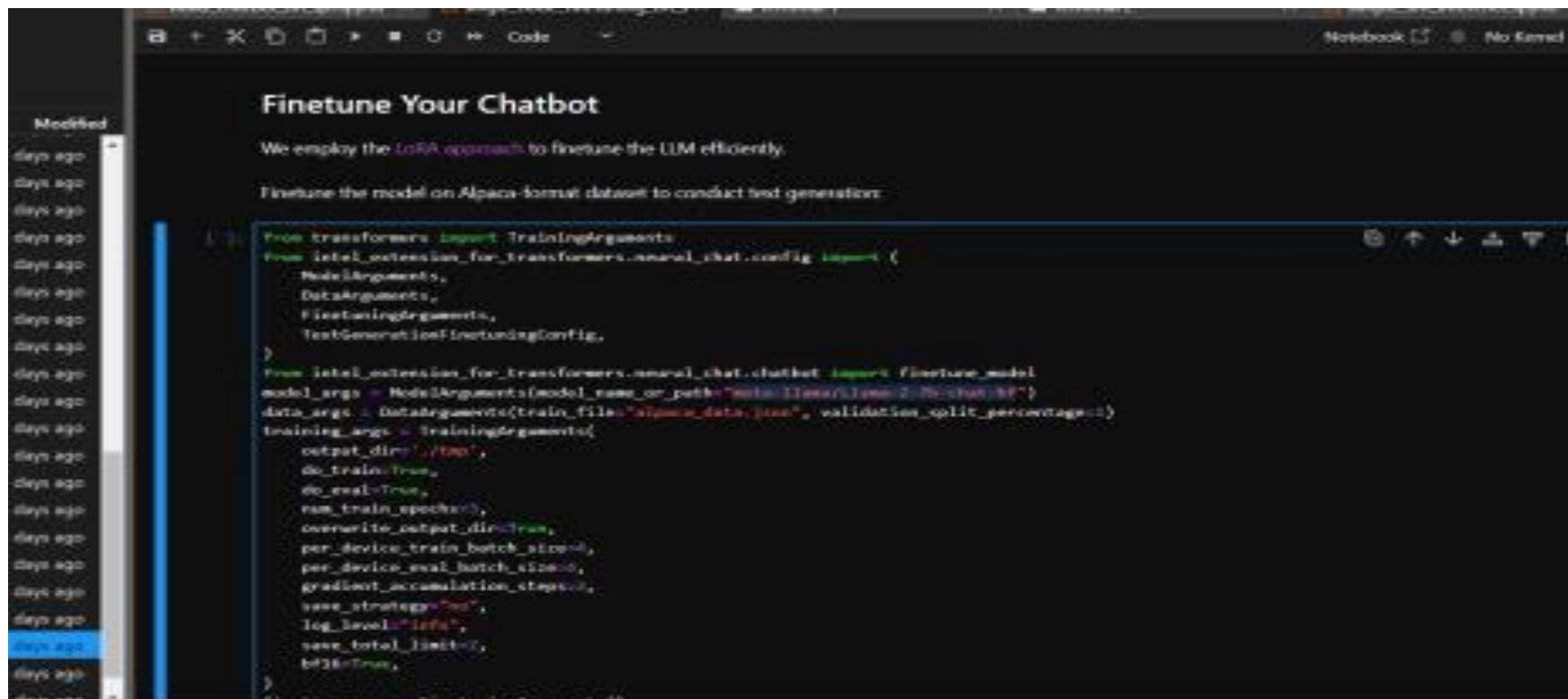
Step 6:

Inorder to finetune the 3 notebooks we need 3 different datasets required in the notebook . In the given notebook 3 different datasets are provided. The model mention here is the meta-llama/Llama-2-7b-chat-hf

The model required to run the given notebook is

1. Alcapa dataset
 2. Cnn_dailymail
 3. theblackcat102/evol-codealpaca-v1

Step 7:



The screenshot shows a Jupyter Notebook interface with a dark theme. The title of the notebook is "Finetune Your Chatbot". The code cell contains the following Python script:

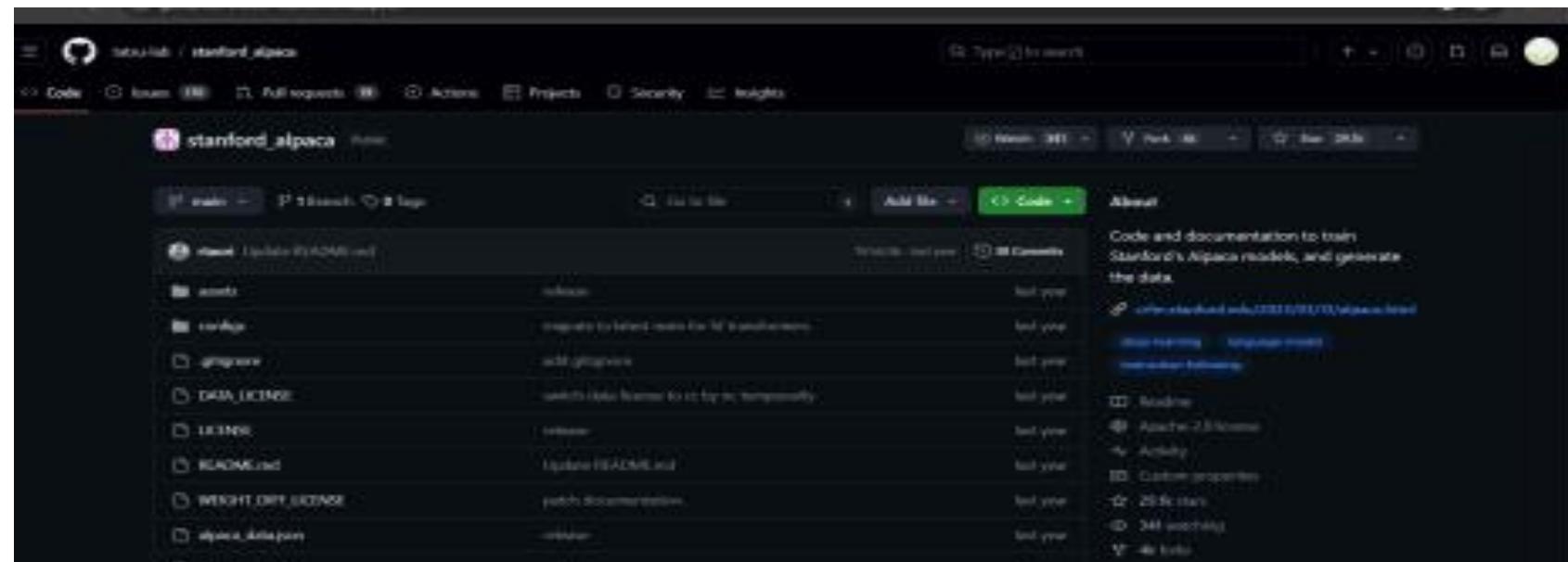
```
from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
    DataArguments,
    FinetuningArguments,
    TextGenerationFinetuningConfig,
)
from intel_extension_for_transformers.neural_chat.chatbot import FinetuneModel
model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7B-hf")
data_args = DataArguments(train_file="alpaca_data.json", validation_split_percentage=0)
training_args = FinetuningArguments(
    output_dir='./finetuned',
    do_train=True,
    do_eval=True,
    num_train_epochs=3,
    overwrite_output_dir=True,
    per_device_train_batch_size=1,
    per_device_eval_batch_size=1,
    gradient_accumulation_steps=1,
    save_strategy="no",
    log_level="INFO",
    save_total_limit=2,
    max_grad_norm=1.0,
)
```

The code imports necessary modules from the transformers library and intel-extension-for-transformers.neural-chat. It defines arguments for the model, data, and finetuning process. The model is set to "meta-llama/Llama-2-7B-hf" and the data is "alpaca_data.json". The training will consist of 3 epochs, with a batch size of 1 for both training and evaluation, and a gradient accumulation step of 1. The logs will be at INFO level and saved in a directory named "finetuned".

In the given notebook we have specified the model name and path as well as the dataset required for the finetuning. The dataset mentioned in the notebook is the alpaca dataset.

Step 8:

Click the alpaca dataset in the notebook . Once its clicked it will redirect us to a github rep having the alpaca dataset



Step 9:

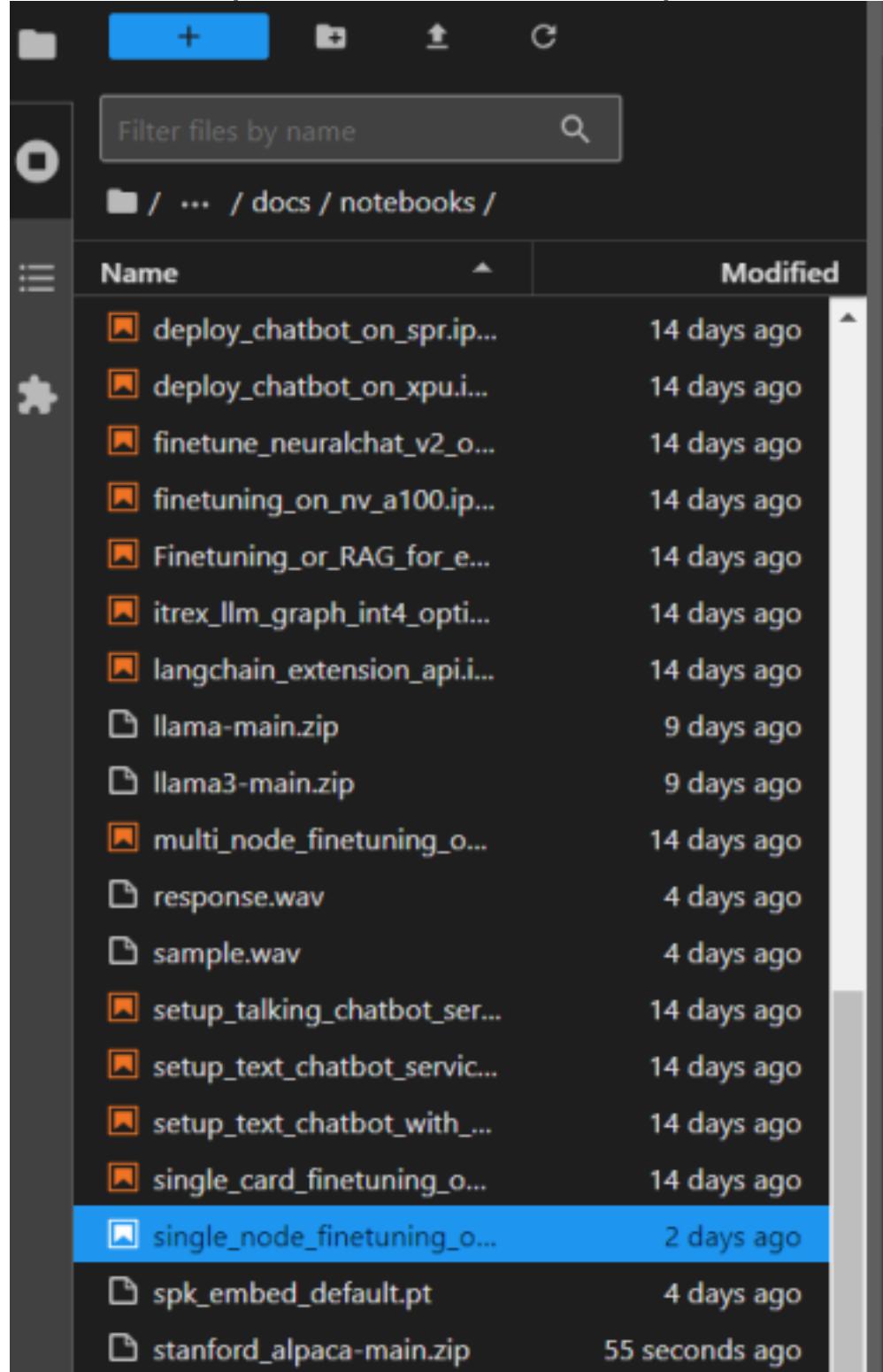
Inorder to use the dataset we need to download it as a zip file

Step 10: Once the file is downloaded it needs to be uploaded in idc or the local terminal for unzipping the zip file .

alpaca_data	30-06-2024 22:10	JSON Source File	22,241 KB
stanford_alpaca-main	30-06-2024 21:31	Compressed (zipp...)	9,335 KB

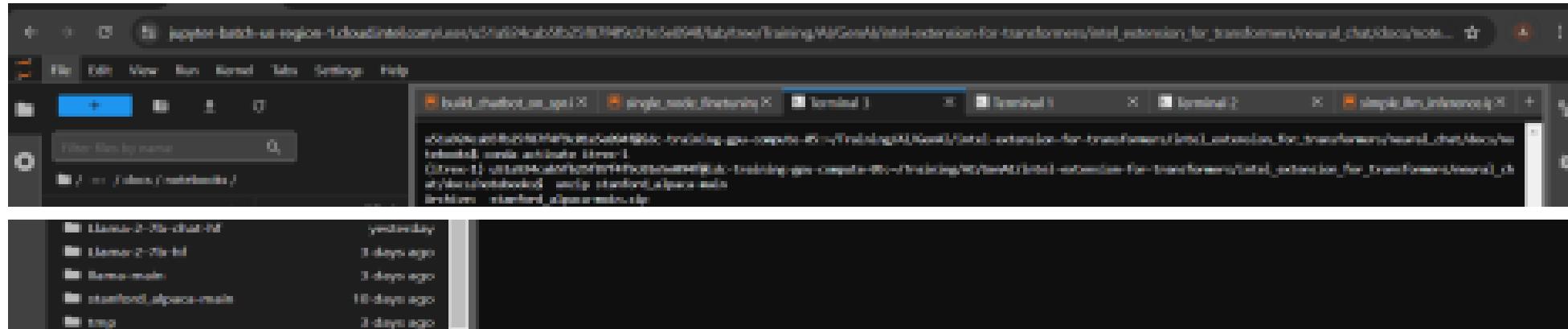
Step 11:

Once the file is uploaded we can unzip it in the terminal in idc server



Step 12:

Unzip it in the idc terminal



Step 13:

Run the notebook

```
[ ]: from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
    DataArguments,
    FinetuningArguments,
    TextGenerationFinetuningConfig,
)
from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
model_args = ModelArguments(model_name_or_path="meta-llma/llama-2-7b-chat-hf")
data_args = DataArguments(train_file="alpaca_data.json", validation_split_percentage=1)
training_args = TrainingArguments(
    output_dir='./tmp',
    do_train=True,
    do_eval=True,
    num_train_epochs=3,
    overwrite_output_dir=True,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    gradient_accumulation_steps=2,
    save_strategy="no",
    log_level="info",
    save_total_limit=2,
    bf16=True,
)
finetune_args = FinetuningArguments()
finetune_cfg = TextGenerationFinetuningConfig(
    model_args=model_args,
    data_args=data_args,
    training_args=training_args,
    finetune_args=finetune_args,
)
finetune_model(finetune_cfg)
```

Step 14:

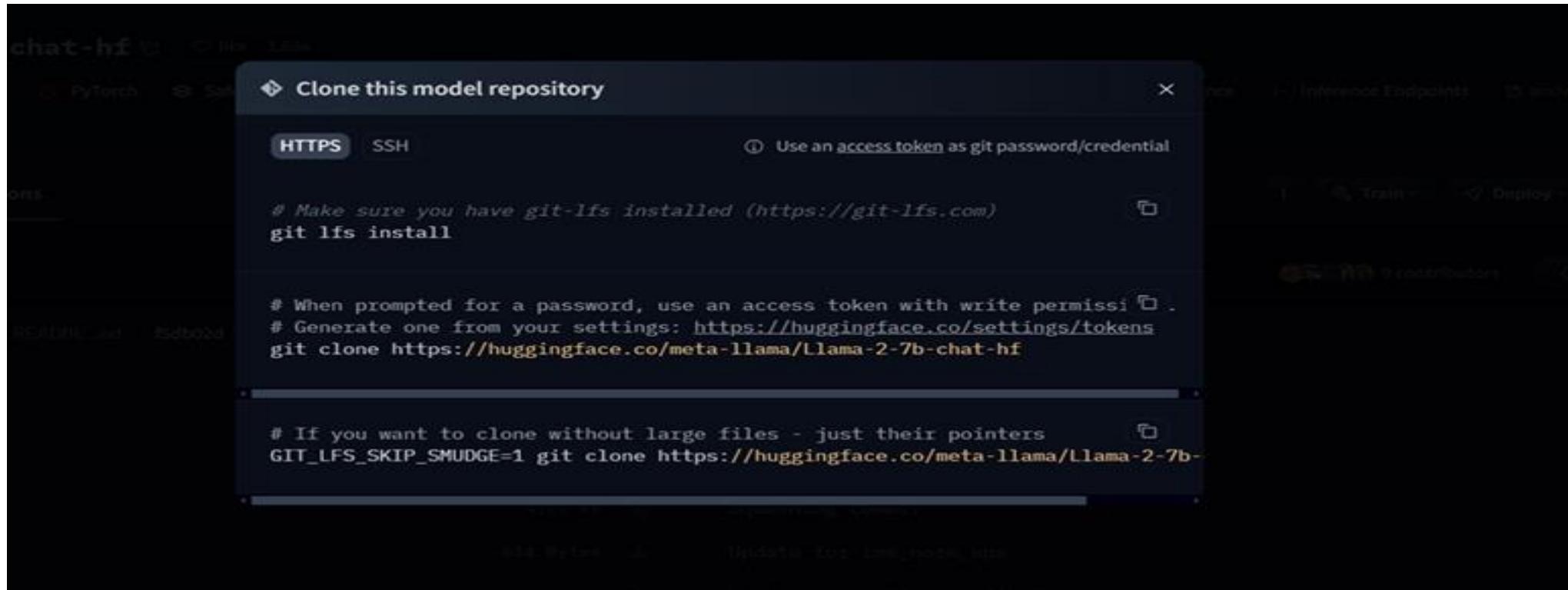
In certain cases the notebook may show an error saying the huggingface token is not able to connect to the meta-llama/Llama-2-7b-chat-hf model. The error msg indicates

```
warnings.warn(  
2024-07-05 05:37:19,930 - chatbot.py - intel_extension_for_transformers.neural_chat.chatbot - ERROR - Exception: We couldn't connect to 'https://huggingface.co' to load this file, couldn't find it in the cached files and it looks like meta-llama/Llama-2-7b-chat-hf is not the path to a directory containing a file named config.json.  
Checkout your internet connection or see how to run the library in offline mode at 'https://huggingface.co/docs/transformers/installation/offline-mode'.  
2024-07-05 05:37:19,931 - error_utils.py - intel_extension_for_transformers.neural_chat.utils.error_utils - ERROR - neuralchat_error: L0  
FA finetuning failed
```

The only way to run the notebook is by loading the meta lama mode in offline mode ; inorder to do so we need to download the entire files of lama model

File	Last Commit
README.md	3 months ago
.gitattributes	13 months ago
LICENSE.txt	13 months ago
README.txt	3 months ago
.gitmodules	13 months ago
config.json	3 months ago
generation_config.json	13 months ago
model-00001-v1-00002.safetensors	13 months ago

Or we can clone the entire meta lama repo



In order to clone this repo we need a Hugging face token with write permission

Name	Value	Last Refreshed Date	Last Used Date	Permissions
Oneeq	hf...NZfJ	about 11 hours ago	about 3 hours ago	WRITE
AsusX	hf...sKEB	3 days ago	-	FINEGRAINED
Intel	hf...dEHe	4 days ago	-	FINEGRAINED

Step 16:

Now we need to login to the hugging face id using the write permitted token

```
u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-18:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks$ conda activate itrex-1
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-18:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks
$ huggingface-cli login
[REDACTED]
A token is already saved on your machine. Run `huggingface-cli whoami` to get more information or `huggingface-cli logout` if you want to log out.
Setting a new token will erase the existing one.
To login, `huggingface_hub` requires a token generated from https://huggingface.co/settings/tokens .
Enter your token (input will not be visible):
Add token as git credential? (Y/n) y
Token is valid (permission: write).
Your token has been saved in your configured git credential helpers (store).
Your token has been saved to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/token
Login successful.
```

Step 17:

Since meta lama model repo huge we cannot clone it directly ; in order to clone it in windows we need to install git lfs :

(Git large file storage) which can be done by manually installing it from the git website

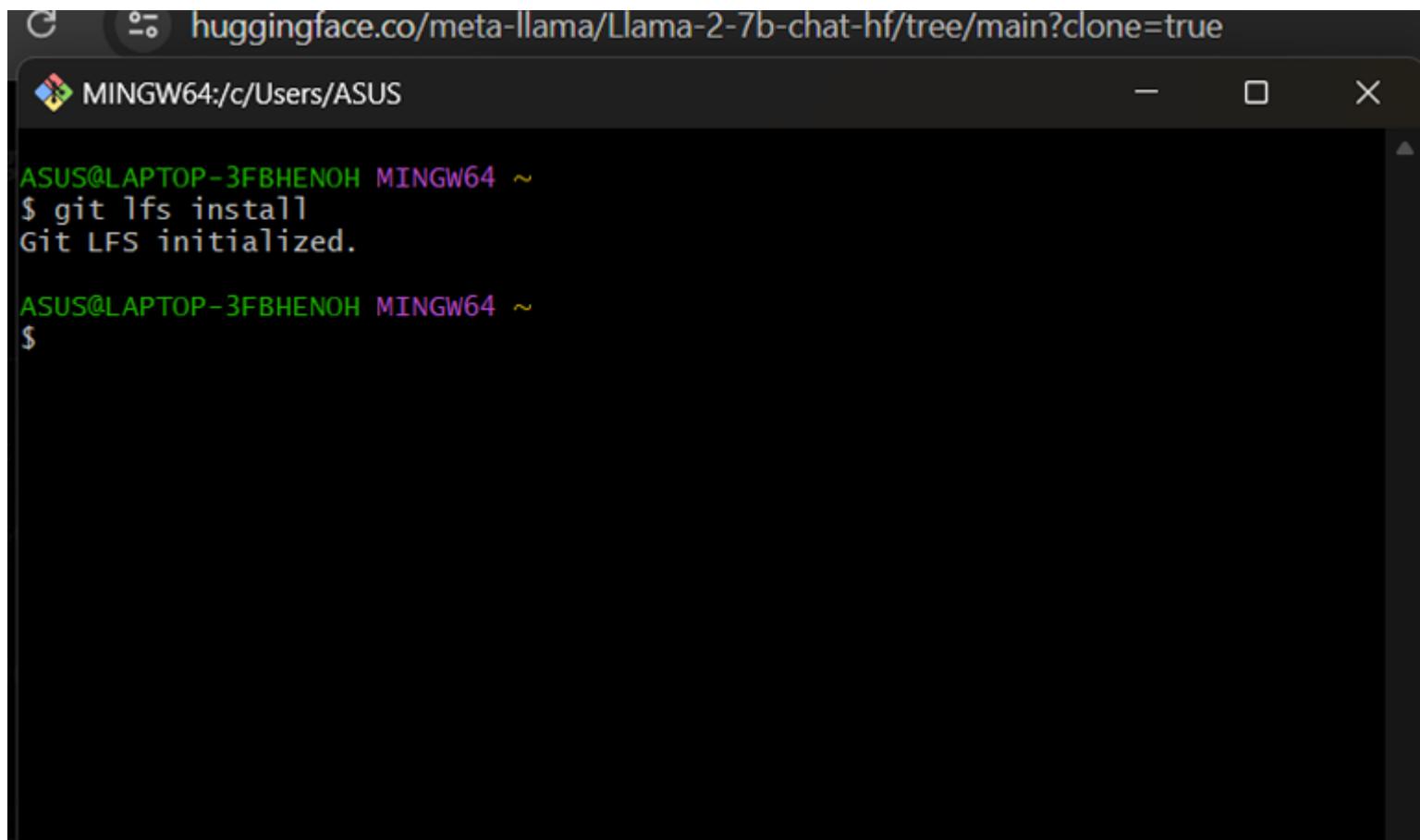
The screenshot shows the official website for Git Large File Storage (LFS). At the top, there's a navigation bar with links to Docs, Discussions, Wiki, Installation, Releases, and Source. Below the navigation, there's a heading "An open source Git extension for versioning large files". A diagram illustrates the workflow: a "Local" computer (represented by a monitor icon) connects to a "Remote" server (represented by a stack of three red boxes). Both the local and remote sides have "code" components, with arrows indicating bidirectional communication between them. At the bottom left, there's a blue button labeled "Download v3.5.1 (Windows)".

Or by clone git lfs repo into the terminal

```
Login successful
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-18:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks
$ git clone https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
fatal: destination path 'Llama-2-7b-chat-hf' already exists and is not an empty directory.
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-18:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks
$ git clone https://github.com/git-lfs/git-lfs.git
Cloning into 'git-lfs'...
remote: Enumerating objects: 49113, done.
remote: Counting objects: 100% (294/294), done.
remote: Compressing objects: 100% (161/161), done.
remote: Total 49113 (delta 149), reused 243 (delta 133), pack-reused 48819
Receiving objects: 100% (49113/49113), 19.30 MiB | 20.74 MiB/s, done.
Resolving deltas: 100% (33940/33940), done.
(itrex-1) u51a924cab5fb25f87f4f9c01e5e094f@idc-training-gpu-compute-18:~/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks
$ 
```

Step 18:

Once the installation or cloning is completely done In case of manual installation we may need to activate lfs in git bash



The screenshot shows a terminal window titled "MINGW64:c/Users/ASUS". The command \$ git lfs install is entered, followed by the output "Git LFS initialized.".

```
C  huggingface.co/meta-llama/Llama-2-7b-chat-hf/tree/main?clone=true
MINGW64:c/Users/ASUS
ASUS@LAPTOP-3FBHENOH MINGW64 ~
$ git lfs install
Git LFS initialized.

ASUS@LAPTOP-3FBHENOH MINGW64 ~
$ 
```

In case of cloning lfs ; we can directly clone the meta lama model

```
■ Llama-2-7b-chat-hf 2d ago Add token as git credential? (Y/n) y
■ Llama-2-7b-hf 3d ago Token is valid (permission: write).
■ Llama-main 3d ago Your token has been saved in your configured git credential helpers (store).
■ stanford_alpaca... 10d ago Your token has been saved to /home/ubuntu2calinfb25f87f9fc81a5a094f/.cache/huggingface/token
■ tmp 3d ago Login successful
(stanford-1) u51a824cab5fb25f87f4f9c81a5a094f@dc-training-gpu-compute-18:~/training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/docs/notebooks
$ git clone https://huggingface.co/meta-Llama/Llama-2-7b-chat-hf
fatal: destination path 'Llama-2-7b-chat-hf' already exists and is not an empty directory.
```

Step 19:

Once the model is completely cloned or installed we can run the notebook with the alpaca dataset in the correct directory . Similarly we can finetune 3 notebooks by accessing the hugging face tokens or by manually cloning or installing the required datasets in the correct directories

Finetuning chatbot

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there is a file browser pane showing a directory structure under '/ ... / docs / notebooks /'. The main pane displays a notebook titled 'single_node_finetuning_on_s.ipynb' with the following content:

```
Finetune Your Chatbot

We employ the LoRA approach to finetune the LLM efficiently.

Finetune the model on Alpaca-format dataset to conduct text generation:

[*]: from transformers import TrainingArguments
      from intel_extension_for_transformers.neural_chat.config import (
          ModelArguments,
          DataArguments,
          FinetuningArguments,
          TextGenerationFinetuningConfig,
      )
      from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
      model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-hf")
      data_args = DataArguments(train_file="./stanford_alpaca-main/alpaca_data.json", validation_split_percentage=1)
      training_args = TrainingArguments(
          output_dir='./tmp',
          do_train=True,
          do_eval=True,
          num_train_epochs=3,
          overwrite_output_dir=True,
          per_device_train_batch_size=4,
          per_device_eval_batch_size=4,
          gradient_accumulation_steps=2,
          save_strategy="no",
          log_level="info",
          save_total_limit=2,
          bf16=True,
      )
      finetune_args = FinetuningArguments()
      finetune_cfg = TextGenerationFinetuningConfig()
```

The bottom status bar indicates 'Mode: Edit' and 'Ln 20, Col 24'.

File Edit View Run Kernel Tabs Settings Help

+ ↻ ⌂

Filter files by name

/ ... / docs / notebooks /

Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	32 minutes ago
llama-main	46 minutes ago
stanford_alpaca-main	6 days ago
tmp	25 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
ataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ip...	10 hours ago
build_chatbot_on_xpu.ip...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balan...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Launcher Terminal 1 single_node_finetuning_on_s +

Notebook neural-chat-x

```
training_args=training_args,
finetune_args=finetune_args,
)
finetune_model(finetune_cfg)
```

2024-07-07 18:42:37.141590: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.

2024-07-07 18:42:37.170828: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:479] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT when one has already been registered

2024-07-07 18:42:37.209290: E external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:10575] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered

2024-07-07 18:42:37.209435: E external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1442] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered

2024-07-07 18:42:37.238159: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations. To enable the following instructions: AVX2 AVX512F AVX512_VNNI AVX512_BF16 AVX512_FP16 AVX_VNNI AMX_TILE AMX_INT8 AMX_BF16 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.

2024-07-07 18:42:39.934240: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1/lib/python3.10/site-packages/transformers/training_args.py:1489: FutureWarning: using `no_cuda` is deprecated and will be removed in version 5.0 of 🤗 Transformers. Use `use_cpu` instead

warnings.warn(

2024-07-07 18:42:41,902 - _logger.py - intel_extension_for_transformers.transformers.llm.finetuning - WARNING - Process rank: 0, device: cpu

distributed training: True, 16-bits training: True

2024-07-07 18:42:41,905 - finetuning.py - intel_extension_for_transformers.transformers.llm.finetuning - INFO - Training/evaluation parameters TrainingArguments(

_n_gpu=0,

accelerator_config={'split_batches': False, 'dispatch_batches': None, 'even_batches': True, 'use_seedable_sampler': True, 'non_blocking': False, 'gradient_accumulation_kwargs': None},

adafactor=False,

adam_beta1=0.9,

adam_beta2=0.999,

adam_epsilon=1e-08,

auto_find_batch_size=False

Simple 2 2 neural-chat-x | Idle Mode: Edit 21, Col 22 single_node_finetuning_on_s.ipynb 1 🔍

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with icons for file operations like creating (+), deleting (-), moving (up), and copying (C). Below this is a "Filter files by name" search bar and a list of files and folders in the current directory. The list includes:
 - docs
 - intel-extension-for-transf...
 - Llama-2-7b-hf
 - llama-main
 - stanford_alpaca-main
 - tmp
 - workshop
 - amp_optimization_on_ha...
 - amp_optimization_on_spr...
 - ataset
 - bits_and_bytes_optimizati...
 - build_chatbot_on_habana...
 - build_chatbot_on_icx.ipynb
 - build_chatbot_on_nv_a10...
 - build_chatbot_on_spr.ip...
 - build_chatbot_on_xpu.ip...
 - build_chatbot_with_rag.ip...
 - build_talkingbot_on_pc.ip...
 - chatbot_with_load_balan...
 - customize_chatbot_with_f...
 - customize_chatbot_with_...
- Terminal:** In the center, there is a terminal window titled "single_node_finetuning_on_spr.ipynb". It displays log output from a TensorFlow session, including warnings about TensorRT and FutureWarning regarding `no_cuda` usage. The log also shows training parameters and optimizer settings.
- Code Editor:** On the right, there is a code editor window titled "single_node_finetuning_on_spr.ipynb". The code is identical to the log output in the terminal, showing the configuration for a single-node finetuning process.

At the bottom, the status bar shows "Simple" mode, 2 cells, 2 outputs, and the notebook title "neural-chat-x | Idle". The bottom right corner has a small notification icon.

```
operations, rebuild TensorFlow with the appropriate compiler flags.
2024-07-07 18:42:39.934240: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
/home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1/lib/python3.10/site-packages/transformers/training_args.py:1489: FutureWarning:
g: using `no_cuda` is deprecated and will be removed in version 5.0 of 🤗 Transformers. Use `use_cpu` instead
warnings.warn(
2024-07-07 18:42:41,902 - _logger.py - intel_extension_for_transformers.transformers.llm.finetuning - WARNING - Process rank: 0, device: cpu
distributed training: True, 16-bits training: True
2024-07-07 18:42:41,905 - finetuning.py - intel_extension_for_transformers.transformers.llm.finetuning - INFO - Training/evaluation parameters TrainingArguments(
_n_gpu=0,
accelerator_config={'split_batches': False, 'dispatch_batches': None, 'use_seadable_sampler': True, 'non_blocking': False, 'gradient_accumulation_kwargs': None},
adafactor=False,
adam_beta1=0.9,
adam_beta2=0.999,
adam_epsilon=1e-08,
auto_find_batch_size=False,
batch_eval_metrics=False,
bf16=True,
bf16_full_eval=False,
data_seed=None,
dataloader_drop_last=False,
dataloader_num_workers=0,
dataloader_persistent_workers=False,
dataloader_pin_memory=False,
dataloader_prefetch_factor=None,
ddp_backend=None,
ddp_broadcast_buffers=None,
ddp_bucket_cap_mb=None,
ddp_find_unused_parameters=None,
ddp_timeout=1800,
debug=[],
deepspeed=None,
disable_tqdm=False,
dispatch_batches=None,
```

A screenshot of a Jupyter Notebook interface. The top navigation bar includes File, Edit, View, Run, Kernel, Tabs, Settings, and Help. The left sidebar features a file browser with a search bar labeled "Filter files by name". The main area contains three tabs: "Launcher", "Terminal 1", and "single_node_finetuning_on_s...". The "single_node_finetuning_on_s..." tab is active and displays a large amount of Python code related to finetuning a model. The code includes parameters like adam_beta1=0.9, adam_beta2=0.999, adam_epsilon=1e-08, auto_find_batch_size=False, batch_eval_metrics=False, bf16=True, bf16_full_eval=False, data_seed=None, dataloader_drop_last=False, dataloader_num_workers=0, dataloader_persistent_workers=False, dataloader_pin_memory=False, dataloader_prefetch_factor=None, ddp_backend=None, ddp_broadcast_buffers=None, ddp_bucket_cap_mb=None, ddp_find_unused_parameters=None, ddp_timeout=1800, debug=[], deepspeed=None, disable_tqdm=False, dispatch_batches=None, do_eval=True, do_predict=False, do_train=True, eval_accumulation_steps=None, eval_delay=0, eval_do_concat_batches=True, eval_steps=None, eval_strategy=no, evaluation_strategy=None, fp16=False, fp16_backend=auto, fp16_full_eval=False, fp16_opt_level=01, fsdp=[], and max_grad_norm=0.0. The bottom right corner shows a timestamp of 00:51.

```
adam_beta1=0.9,
adam_beta2=0.999,
adam_epsilon=1e-08,
auto_find_batch_size=False,
batch_eval_metrics=False,
bf16=True,
bf16_full_eval=False,
data_seed=None,
dataloader_drop_last=False,
dataloader_num_workers=0,
dataloader_persistent_workers=False,
dataloader_pin_memory=False,
dataloader_prefetch_factor=None,
ddp_backend=None,
ddp_broadcast_buffers=None,
ddp_bucket_cap_mb=None,
ddp_find_unused_parameters=None,
ddp_timeout=1800,
debug=[],
deepspeed=None,
disable_tqdm=False,
dispatch_batches=None,
do_eval=True,
do_predict=False,
do_train=True,
eval_accumulation_steps=None,
eval_delay=0,
eval_do_concat_batches=True,
eval_steps=None,
eval_strategy=no,
evaluation_strategy=None,
fp16=False,
fp16_backend=auto,
fp16_full_eval=False,
fp16_opt_level=01,
fsdp=[],
```

A screenshot of a Jupyter Notebook interface. On the left, there is a file browser showing a directory structure under "/ ... / docs / notebooks /". The list includes various notebooks and directories like "docs", "intel-extension-for-transf...", "Llama-2-7b-hf", etc. In the center, there is a terminal window titled "single_node_finetuning_on_s..." showing Python code related to training parameters. On the right, there is a code editor window titled "single_node_finetuning_on_s..." containing the same code. The bottom status bar shows "Mode: Edit" and the file path "single_node_finetuning_on_spr.ipynb".

```
fp16_backend=auto,  
fp16_full_eval=False,  
fp16_opt_level=01,  
fsdp=[],  
fsdp_config={'min_num_params': 0, 'xla': False, 'xla_fsdp_v2': False, 'xla_fsdp_grad_ckpt': False},  
fsdp_min_num_params=0,  
fsdp_transformer_layer_cls_to_wrap=None,  
full_determinism=False,  
gradient_accumulation_steps=2,  
gradient_checkpointing=False,  
gradient_checkpointing_kwargs=None,  
greater_is_better=None,  
group_by_length=False,  
half_precision_backend=auto,  
hub_always_push=False,  
hub_model_id=None,  
hub_private_repo=False,  
hub_strategy=every_save,  
hub_token=<HUB_TOKEN>,  
ignore_data_skip=False,  
include_inputs_for_metrics=False,  
include_num_input_tokens_seen=False,  
include_tokens_per_second=False,  
jit_mode_eval=False,  
label_names=None,  
label_smoothing_factor=0.0,  
learning_rate=5e-05,  
length_column_name=length,  
load_best_model_at_end=False,  
local_rank=0,  
log_level=info,  
log_level_replica=warning,  
log_on_each_node=True,  
logging_dir=./tmp/runs/Jul07_18-42-36_idc-training-gpu-compute-01,  
logging_first_step=False,  
logging_nan_inf_filter=True,
```

A screenshot of a Jupyter Notebook interface. On the left, there is a file browser showing a list of notebooks and other files in the directory `/ ... / docs / notebooks /`. The list includes `docs`, `intel-extension-for-transf...`, `Llama-2-7b-hf`, `llama-main`, `stanford_alpaca-main`, `tmp`, `workshop`, `amp_optimization_on_ha...`, `amp_optimization_on_spr...`, `ataset`, `bits_and_bytes_optimizati...`, `build_chatbot_on_habana...`, `build_chatbot_on_icx.ipynb`, `build_chatbot_on_nv_a10...`, `build_chatbot_on_spr.ipynb`, `build_chatbot_on_xpu.ipynb`, `build_chatbot_with_rag.ip...`, `build_talkingbot_on_pc.ip...`, `chatbot_with_load_balan...`, `customize_chatbot_with_f...`, and `customize_chatbot_with_...`. The files are sorted by **Modified** date.

The central part of the interface is a terminal window titled `single_node_finetuning_on_s...` which contains the following Python code:

```
fsdp_min_num_params=0,  
fsdp_transformer_layer_cls_to_wrap=None,  
full_determinism=False,  
gradient_accumulation_steps=2,  
gradient_checkpointing=False,  
gradient_checkpointing_kwargs=None,  
greater_is_better=None,  
group_by_length=False,  
half_precision_backend=auto,  
hub_always_push=False,  
hub_model_id=None,  
hub_private_repo=False,  
hub_strategy=every_save,  
hub_token=<HUB_TOKEN>,  
ignore_data_skip=False,  
include_inputs_for_metrics=False,  
include_num_input_tokens_seen=False,  
include_tokens_per_second=False,  
jit_mode_eval=False,  
label_names=None,  
label_smoothing_factor=0.0,  
learning_rate=5e-05,  
length_column_name=length,  
load_best_model_at_end=False,  
local_rank=0,  
log_level=info,  
log_level_replica=warning,  
log_on_each_node=True,  
logging_dir=./tmp/runs/Jul07_18-42-36_idc-training-gpu-compute-01,  
logging_first_step=False,  
logging_nan_inf_filter=True,  
logging_steps=500,  
logging_strategy=steps,  
lr_scheduler_kwargs={},  
lr_scheduler_type=linear,  
max_grad_norm=1.0,
```

The bottom status bar shows the mode is **Edit**, the line and column are **Ln 21, Col 22**, the notebook is **single_node_finetuning_on_spr.ipynb**, and there is a single notification icon.

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there is a file browser window titled "Launcher" showing a list of notebooks in the directory "/ ... / docs / notebooks /". The list includes various notebooks such as "docs", "intel-extension-for-transf...", "Llama-2-7b-hf", "llama-main", "stanford_alpaca-main", "tmp", "workshop", and many others starting with "build_chatbot_on_". The "workshop" notebook is currently selected. On the right, there is a code editor window titled "single_node_finetuning_on_spr.ipynb" which contains the following Python code:

```
lr_scheduler_type=linear,
max_grad_norm=1.0,
max_steps=-1,
metric_for_best_model=None,
mp_parameters=,
neftune_noise_alpha=None,
no_cuda=True,
num_train_epochs=3,
optim=adammw_torch,
optim_args=None,
optim_target_modules=None,
output_dir='./tmp',
overwrite_output_dir=True,
past_index=-1,
per_device_eval_batch_size=4,
per_device_train_batch_size=4,
prediction_loss_only=False,
push_to_hub=False,
push_to_hub_model_id=None,
push_to_hub_organization=None,
push_to_hub_token=<PUSH_TO_HUB_TOKEN>,
ray_scope=last,
remove_unused_columns=True,
report_to=['tensorboard'],
restore_callback_states_from_checkpoint=False,
resume_from_checkpoint=None,
run_name='./tmp',
save_on_each_node=False,
save_only_model=False,
save_safetensors=True,
save_steps=500,
save_strategy=no,
save_total_limit=2,
seed=42,
skip_memory_metrics=True,
split_batches=None,
```

The status bar at the bottom indicates "Mode: Edit" and "Ln 21, Col 22 single_node_finetuning_on_spr.ipynb".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / docs / notebooks /`. The tree lists several notebooks and other files, such as `docs`, `intel-extension-for-transf...`, `Llama-2-7b-hf`, `llama-main`, `stanford_alpaca-main`, `tmp`, `workshop`, `amp_optimization_on_ha...`, `amp_optimization_on_spr...`, `ataset`, `bits_and_bytes_optimizati...`, `build_chatbot_on_habana...`, `build_chatbot_on_icx.ipynb`, `build_chatbot_on_nv_a10...`, `build_chatbot_on_spr.ipynb`, `build_chatbot_on_xpu.ipynb`, `build_chatbot_with_rag.ip...`, `build_talkingbot_on_pc.ip...`, `chatbot_with_load_balan...`, `customize_chatbot_with_f...`, and `customize_chatbot_with_...`. A search bar at the top of the sidebar allows filtering by file name.
- Terminal:** In the center, there is a terminal window titled "single_node_finetuning_on_s...". It displays Python code related to model configuration and download parameters. A progress bar at the bottom indicates a download of `config.json` from `614/614 [00:00<00:00, 39.5kB/s]`.
- Code Editor:** On the right, there is a code editor window titled "neural-chat-x". It shows a snippet of Python code, likely part of a notebook, with syntax highlighting for variables like `resume_from_checkpoint`, `run_name`, and `save_steps`.

The status bar at the bottom provides information about the notebook's state: "Simple" mode, 2 cells, 2 outputs, and the title "neural-chat-x | Idle". The status bar also indicates the current mode is "Edit", the line and column numbers (Ln 21, Col 22), and the notebook name ("single_node_finetuning_on_spr.ipynb").

```
resume_from_checkpoint=None,
run_name='./tmp',
save_on_each_node=False,
save_only_model=False,
save_safetensors=True,
save_steps=500,
save_strategy=no,
save_total_limit=2,
seed=42,
skip_memory_metrics=True,
split_batches=None,
tf32=None,
torch_compile=False,
torch_compile_backend=None,
torch_compile_mode=None,
torchdynamo=None,
tpu_metrics_debug=False,
tpu_num_cores=None,
use_cpu=True,
use_ipex=False,
use_legacy_prediction_loop=False,
use_mps_device=False,
warmup_ratio=0.0,
warmup_steps=0,
weight_decay=0.0,
)
/home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132: FutureWarning: `resume_download` is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want to force a new download, use `force_download=True`.
    warnings.warn(
config.json: 100% 614/614 [00:00<00:00, 39.5kB/s]

[INFO|configuration_utils.py:733] 2024-07-07 18:42:42,960 >> loading configuration file config.json from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/config.json
[INFO|configuration_utils.py:796] 2024-07-07 18:42:42,967 >> Model config LlamaConfig {
```

File Edit View Run Kernel Tabs Settings Help

+ ↻ ⌂ ⌄ ⌅

Filter files by name

/ ... / docs / notebooks /

Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	32 minutes ago
llama-main	46 minutes ago
stanford_alpaca-main	6 days ago
tmp	25 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
ataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ip...	10 hours ago
build_chatbot_on_xpu.ip...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balan...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Launcher Terminal 1 single_node_finetuning_on_s

```
[INFO|configuration_utils.py:733] 2024-07-07 18:42:42,960 >> loading configuration file config.json from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/config.json
[INFO|configuration_utils.py:796] 2024-07-07 18:42:42,967 >> Model config LlamaConfig {
    "_name_or_path": "meta-llama/Llama-2-7b-chat-hf",
    "architectures": [
        "LlamaForCausalLM"
    ],
    "attention_bias": false,
    "attention_dropout": 0.0,
    "bos_token_id": 1,
    "eos_token_id": 2,
    "hidden_act": "silu",
    "hidden_size": 4096,
    "initializer_range": 0.02,
    "intermediate_size": 11008,
    "max_position_embeddings": 4096,
    "mlp_bias": false,
    "model_type": "llama",
    "num_attention_heads": 32,
    "num_hidden_layers": 32,
    "num_key_value_heads": 32,
    "pretraining_tp": 1,
    "rms_norm_eps": 1e-05,
    "rope_scaling": null,
    "rope_theta": 10000.0,
    "tie_word_embeddings": false,
    "torch_dtype": "float16",
    "transformers_version": "4.41.2",
    "use_cache": true,
    "vocab_size": 32000
}
```

tokenizer_config.json: 100% 1.62k/1.62k [00:00<00:00, 166kB/s]

tokenizer_model: 100% 500k/500k [00:00<00:00, 25.9MB/s]

Simple 2 \$ 2 neural-chat-x | Idle Saving completed Mode: Command 1

File Edit View Run Kernel Tabs Settings Help

+ ↻

Filter files by name

/ ... / docs / notebooks /

Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	32 minutes ago
llama-main	46 minutes ago
stanford_alpaca-main	6 days ago
tmp	25 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
dataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ip...	10 hours ago
build_chatbot_on_xpu.ip...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balan...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Launcher Terminal 1 single_node_finetuning_on_spr.ipynb

tokenizer_config.json: 100% 1.02K/1.02K [00:00<00:00, 100kB/s]

tokenizer.model: 100% 500k/500k [00:00<00:00, 26.9MB/s]

special_tokens_map.json: 100% 414/414 [00:00<00:00, 41.5kB/s]

tokenizer.json: 100% 1.84M/1.84M [00:00<00:00, 11.3MB/s]

```
[INFO|tokenization_utils_base.py:2108] 2024-07-07 18:42:44,144 >> loading file tokenizer.model from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer.model
[INFO|tokenization_utils_base.py:2108] 2024-07-07 18:42:44,146 >> loading file added_tokens.json from cache at None
[INFO|tokenization_utils_base.py:2108] 2024-07-07 18:42:44,147 >> loading file special_tokens_map.json from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/special_tokens_map.json
[INFO|tokenization_utils_base.py:2108] 2024-07-07 18:42:44,147 >> loading file tokenizer_config.json from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer_config.json
[INFO|tokenization_utils_base.py:2108] 2024-07-07 18:42:44,148 >> loading file tokenizer.json from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer.json
Using custom data configuration default-d16f3e0b8ba7c184
2024-07-07 18:42:44,624 - builder.py - datasets.builder - INFO - Using custom data configuration default-d16f3e0b8ba7c184
Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
2024-07-07 18:42:44,631 - info.py - datasets.info - INFO - Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
Generating dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
2024-07-07 18:42:44,667 - builder.py - datasets.builder - INFO - Generating dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
Downloading and preparing dataset json/default to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092...
2024-07-07 18:42:44,671 - builder.py - datasets.builder - INFO - Downloading and preparing dataset json/default to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092...
```

Saving completed

Mode: Command

Ln 21, Col 22 single_node_finetuning_on_spr.ipynb 1

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is "... / docs / notebooks /". The tree lists several notebooks and other files, such as "docs", "intel-extension-for-transf...", "Llama-2-7b-hf", "Llama-main", "stanford_alpaca-main", "tmp", "workshop", "amp_optimization_on_ha...", "amp_optimization_on_spr...", "ataset", "bits_and_bytes_optimizati...", "build_chatbot_on_habana...", "build_chatbot_on_icx.ipynb", "build_chatbot_on_nv_a10...", "build_chatbot_on_spr.ipynb", "build_chatbot_on_xpu.ipynb", "build_chatbot_with_rag.ip...", "build_talkingbot_on_pc.ip...", "chatbot_with_load_balan...", "customize_chatbot_with_f...", and "customize_chatbot_with_...".
- Terminal:** In the center, there is a terminal window titled "single_node_finetuning_on_s...". It displays command-line logs related to dataset loading and preparation, including:

```
[INFO]tokenization_utils_base.py:2108] 2024-07-07 18:42:44,144 >> loading file tokenizer.model from cache at /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer.model
[INFO]tokenization_utils_base.py:2108] 2024-07-07 18:42:44,146 >> loading file added_tokens.json from cache at None
[INFO]tokenization_utils_base.py:2108] 2024-07-07 18:42:44,147 >> loading file special_tokens_map.json from cache at /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/special_tokens_map.json
[INFO]tokenization_utils_base.py:2108] 2024-07-07 18:42:44,147 >> loading file tokenizer_config.json from cache at /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer_config.json
[INFO]tokenization_utils_base.py:2108] 2024-07-07 18:42:44,148 >> loading file tokenizer.json from cache at /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/tokenizer.json
Using custom data configuration default-d16f3e0b8ba7c184
2024-07-07 18:42:44,624 - builder.py - datasets.builder - INFO - Using custom data configuration default-d16f3e0b8ba7c184
Loading Dataset Infos from /home/u51a924cab5fb25f87f49c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
2024-07-07 18:42:44,631 - info.py - datasets.info - INFO - Loading Dataset Infos from /home/u51a924cab5fb25f87f49c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
Generating dataset json (/home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
2024-07-07 18:42:44,667 - builder.py - datasets.builder - INFO - Generating dataset json (/home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
Downloading and preparing dataset json/default to /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092...
2024-07-07 18:42:44,671 - builder.py - datasets.builder - INFO - Downloading and preparing dataset json/default to /home/u51a924cab5fb25f87f49c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092...
Downloading took 0.0 min
2024-07-07 18:42:44,683 - download_manager.py - datasets.download.download_manager - INFO - Downloading took 0.0 min
Checksum Computation took 0.0 min
2024-07-07 18:42:44,701 - download_manager.py - datasets.download.download_manager - INFO - Checksum Computation took 0.0 min
Generating train split
2024-07-07 18:42:44,721 - builder.py - datasets.builder - INFO - Generating train split
```

- Code Editor:** On the right, there is a code editor window titled "neural-chat-x". The status bar indicates "Saving completed".
- Bottom Status Bar:** The status bar shows "Mode: Command" and the current file path "single_node_finetuning_on_spr.ipynb".

File Edit View Run Kernel Tabs Settings Help

Launcher Terminal 1 single_node_finetuning_on_s

2024-07-07 18:42:44,721 - builder.py - datasets.builder - INFO - Generating train split

Generating train split: 52002/0 [00:00<00:00, 73761.36 examples/s]

Unable to verify splits sizes.

2024-07-07 18:42:45,609 - info_utils.py - datasets.utils.info_utils - INFO - Unable to verify splits sizes.

Dataset json downloaded and prepared to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092. Subsequent calls will reuse this data.

2024-07-07 18:42:45,622 - builder.py - datasets.builder - INFO - Dataset json downloaded and prepared to /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092. Subsequent calls will reuse this data.

Using custom data configuration default-d16f3e0b8ba7c184

2024-07-07 18:42:46,072 - builder.py - datasets.builder - INFO - Using custom data configuration default-d16f3e0b8ba7c184

Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json

2024-07-07 18:42:46,074 - info.py - datasets.info - INFO - Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json

Overwrite dataset info from restored data version if exists.

2024-07-07 18:42:46,090 - builder.py - datasets.builder - INFO - Overwrite dataset info from restored data version if exists.

Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092

2024-07-07 18:42:46,096 - info.py - datasets.info - INFO - Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092

Found cached dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)

2024-07-07 18:42:46,103 - builder.py - datasets.builder - INFO - Found cached dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)

Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092

2024-07-07 18:42:46,104 - info.py - datasets.info - INFO - Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092

Using custom data configuration default-d16f3e0b8ba7c184

2024-07-07 18:42:46,358 - builder.py - datasets.builder - INFO - Using custom data configuration default-d16f3e0b8ba7c184

Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json

Simple 2 \$ 2 neural-chat-x | Idle Mode: Command ✓ Ln 21, Col 22 single_node_finetuning_on_spr.ipynb 1 🔔

File Edit View Run Kernel Tabs Settings Help

Launcher Terminal 1 single_node_finetuning_on_s

Filter files by name

/ ... / docs / notebooks /

Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	32 minutes ago
llama-main	46 minutes ago
stanford_alpaca-main	6 days ago
tmp	25 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
ataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ip...	10 hours ago
build_chatbot_on_xpu.ip...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balanc...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Using custom data configuration default-d16f3e0b8ba7c184
2024-07-07 18:42:46,358 - builder.py - datasets.builder - INFO - Using custom data configuration default-d16f3e0b8ba7c184
Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
2024-07-07 18:42:46,365 - info.py - datasets.info - INFO - Loading Dataset Infos from /home/u51a924cab5fb25f87f4f9c01e5e094f/Training/AI/GenAI/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/datasets/src/datasets/packaged_modules/json
Overwrite dataset info from restored data version if exists.
2024-07-07 18:42:46,372 - builder.py - datasets.builder - INFO - Overwrite dataset info from restored data version if exists.
Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092
2024-07-07 18:42:46,376 - info.py - datasets.info - INFO - Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092
Found cached dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
2024-07-07 18:42:46,386 - builder.py - datasets.builder - INFO - Found cached dataset json (/home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092)
Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092
2024-07-07 18:42:46,390 - info.py - datasets.info - INFO - Loading Dataset info from /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092
model.safetensors.index.json: 100% 26.8k/26.8k [00:00<00:00, 2.16MB/s]
[INFO|modeling_utils.py:3474] 2024-07-07 18:42:47,119 >> loading weights file model.safetensors from cache at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d4590/model.safe tensors.index.json
Downloading shards: 100% 2/2 [01:52<00:00, 52.50s/it]
model-00001-of-00002.safetensors: 100% 9.98G/9.98G [00:31<00:00, 333MB/s]
model-00002-of-00002.safetensors: 100% 3.50G/3.50G [00:11<00:00, 311MB/s]
[INFO|modeling_utils.py:1519] 2024-07-07 18:44:39,293 >> Instantiating LlamaForCausalLM model under default dtype torch.bfloat16.
[INFO|configuration_utils.py:962] 2024-07-07 18:44:39,300 >> Generate config GenerationConfig {
 "bos_token_id": 1

Mode: Command

Ln 21, Col 22 single_node_finetuning_on_s.ipynb 1

File Edit View Run Kernel Tabs Settings Help

+ ↻ ⌂ ⌄ ⌅ ⌆ ⌇

Filter files by name

/ ... / docs / notebooks /

Name	Modified
docs	10 days ago
intel-extension-for-transf...	11 days ago
Llama-2-7b-hf	32 minutes ago
llama-main	46 minutes ago
stanford_alpaca-main	6 days ago
tmp	26 minutes ago
workshop	11 days ago
amp_optimization_on_ha...	11 days ago
amp_optimization_on_spr...	11 days ago
ataset	last month
bits_and_bytes_optimizati...	11 days ago
build_chatbot_on_habana...	11 days ago
build_chatbot_on_icx.ipynb	11 days ago
build_chatbot_on_nv_a10...	11 days ago
build_chatbot_on_spr.ipny...	10 hours ago
build_chatbot_on_xpu.ipny...	11 days ago
build_chatbot_with_rag.ip...	11 days ago
build_talkingbot_on_pc.ip...	11 days ago
chatbot_with_load_balan...	11 days ago
customize_chatbot_with_f...	11 days ago
customize_chatbot_with_...	11 days ago

Launcher Terminal 1 single_node_finetuning_on_s

Notebook neural-chat-x

```
[INFO|modeling_utils.py:1519] 2024-07-07 18:44:39,293 >> Instantiating LlamaForCausalLM model under default dtype torch.bfloat16.  
[INFO|configuration_utils.py:962] 2024-07-07 18:44:39,300 >> Generate config GenerationConfig {  
    "bos_token_id": 1,  
    "eos_token_id": 2  
}  
  
Loading checkpoint shards: 100% 2/2 [01:42<00:00, 47.10s/it]  
  
[INFO|modeling_utils.py:4280] 2024-07-07 18:46:22,280 >> All model checkpoint weights were used when initializing LlamaForCausalLM.  
  
[INFO|modeling_utils.py:4288] 2024-07-07 18:46:22,282 >> All the weights of LlamaForCausalLM were initialized from the model checkpoint  
at meta-llama/Llama-2-7b-chat-hf.  
If your task is similar to the task the model of the checkpoint was trained on, you can already use LlamaForCausalLM for predictions without further training.  
  
generation_config.json: 100% 188/188 [00:00<00:00, 16.9kB/s]  
  
[INFO|configuration_utils.py:917] 2024-07-07 18:46:22,678 >> loading configuration file generation_config.json from cache at /home/u51a9  
24cab5fb25f87f4f9c01e5e094f/.cache/huggingface/hub/models--meta-llama--Llama-2-7b-chat-hf/snapshots/f5db02db724555f92da89c216ac04704f23d  
4590/generation_config.json  
[INFO|configuration_utils.py:962] 2024-07-07 18:46:22,686 >> Generate config GenerationConfig {  
    "bos_token_id": 1,  
    "do_sample": true,  
    "eos_token_id": 2,  
    "max_length": 4096,  
    "pad_token_id": 0,  
    "temperature": 0.6,  
    "top_p": 0.9  
}  
  
Map: 100% 51482/51482 [01:46<00:00, 488.63 examples/s]  
  
Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e8  
9e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092/cache-5951a22f72ce2067.arrow  
2024-07-07 18:46:27,559 - arrow_dataset.py - datasets.arrow_dataset - INFO - Caching processed dataset at /home/u51a924cab5fb25f87f4f9c0  
1e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb109
```

Simple 2 \$ 2 neural-chat-x | Idle Mode: Command Ln 21, Col 22 single_node_finetuning_on_spr.ipynb 1 ⌂

File Edit View Run Kernel Tabs Settings Help

Launcher Terminal 1 single_node_finetuning_on_s

Map: 100% 51482/51482 [01:46<00:00, 488.63 examples/s]

Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092/cache-5951a22f72ce2067.arrow
2024-07-07 18:46:27,559 - arrow_dataset.py - datasets.arrow_dataset - INFO - Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092/cache-5951a22f72ce2067.arrow

Map: 100% 520/520 [00:01<00:00, 459.20 examples/s]

Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092/cache-54dea6f49b358050.arrow
2024-07-07 18:48:13,043 - arrow_dataset.py - datasets.arrow_dataset - INFO - Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/json/default-d16f3e0b8ba7c184/0.0.0/f4e89e8750d5d5ffbef2c078bf0ddfedef29dc2faff52a6255cf513c05eb1092/cache-54dea6f49b358050.arrow
2024-07-07 18:48:13,119 - finetuning.py - intel_extension_for_transformers.transformers.llm.finetuning.finetuning - INFO - Using data collator of type DataCollatorForSeq2Seq
trainable params: 4,194,304 || all params: 6,742,609,920 || trainable%: 0.06220594176090199

[INFO|trainer.py:641] 2024-07-07 18:48:13,546 >> Using cpu_amp half precision backend
[INFO|trainer.py:2078] 2024-07-07 18:48:14,764 >> ***** Running training *****
[INFO|trainer.py:2079] 2024-07-07 18:48:14,766 >> Num examples = 51,482
[INFO|trainer.py:2080] 2024-07-07 18:48:14,767 >> Num Epochs = 3
[INFO|trainer.py:2081] 2024-07-07 18:48:14,768 >> Instantaneous batch size per device = 4
[INFO|trainer.py:2084] 2024-07-07 18:48:14,768 >> Total train batch size (w. parallel, distributed & accumulation) = 8
[INFO|trainer.py:2085] 2024-07-07 18:48:14,769 >> Gradient Accumulation steps = 2
[INFO|trainer.py:2086] 2024-07-07 18:48:14,770 >> Total optimization steps = 19,305
[INFO|trainer.py:2087] 2024-07-07 18:48:14,774 >> Number of trainable parameters = 4,194,304

[21/19305 21:05 < 356:43:15, 0.02 it/s, Epoch 0.00/3]

Step Training Loss

```
[ ]: from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
```

Simple 2 \$ 2 neural-chat-x | Idle Mode: Command Ln 21, Col 22 single_node_finetuning_on_s 1

VIDEO

The screenshot shows a Jupyter Notebook interface running on Intel Developer Cloud. The left sidebar displays a file tree with various notebooks and files. The main notebook content is titled "Finetune Your Chatbot on a Single Node Xeon SPR". It contains sections for "Prepare Environment" and "Prepare the Dataset", along with terminal commands for installation and dataset selection.

Finetune Your Chatbot on a Single Node Xeon SPR

NeuralChat is a customizable chat framework designed to create user own chatbot within few minutes on multiple architectures. This notebook will introduce how to finetune your chatbot on the customized data on a single node Xeon SPR.

Prepare Environment

Install intel extension for transformers:

```
[ ]: !pip install intel-extension-for-transformers
```

Install Requirements:

```
[ ]: !git clone https://github.com/intel/intel-extension-for-transformers.git
```

```
[ ]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/
[ ]: !pip install -r requirements.txt
[ ]: %cd ../../..
```

Prepare the Dataset

We select 3 kind of datasets to conduct the finetuning process for different tasks.

1. Text Generation (General domain instruction): We use the Alpaca dataset from Stanford University as the general domain dataset to fine-tune the model. This dataset is provided in the form of a JSON file, `alpaca_data.json`. In Alpaca, researchers have manually crafted 175 seed tasks to guide `text-davinci-003` in generating 52K instruction data for diverse tasks.

Mode: Edit Simple 2 \$ 2 neural-chat-x | Busy Ln 3, Col 20 single_node_finetuning_on_spr.ipynb 1 🔍

SINGLE NODE FINETUNING SUMMARISATION

The screenshot shows a Jupyter Notebook interface with the following components:

- File Bar:** File, Edit, View, Run, Kernel, Tabs, Settings, Help.
- Toolbar:** Includes icons for file operations like New, Open, Save, and Run, along with a search bar labeled "Filter files by name".
- File Explorer:** Shows the directory structure: / ... / GenAI / intel-extension-for-transformers /. The list includes conda_meta, docker, docs, examples, intel_extension_for_transfo..., tests, whisper-2024, workflows, env_gpu.sh, LICENSE, MANIFEST.in, README.md, requirements-cpu.txt, requirements.txt, SECURITY.md, setup.py, and third_party_programs.txt. Most files were modified 11 days ago, except for whisper-2024 which was modified 3 months ago.
- Terminal 1:** A terminal window displaying Python code for finetuning a model. The code imports TrainingArguments, ModelArguments, DataArguments, FinetuningArguments, and TextGenerationFinetuningConfig from the transformers and intel_extension_for_transformers.neural_chat.config modules. It then defines model_args, data_args, and training_args. It also defines finetune_args and finetune_cfg. Finally, it calls finetune_model(finetune_cfg). The code is as follows:

```
from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
    DataArguments,
    FinetuningArguments,
    TextGenerationFinetuningConfig,
)
from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-chat-hf")
data_args = DataArguments(dataset_name="cnn_dailymail", dataset_config_name="3.0.0")
training_args = TrainingArguments(
    output_dir='./tmp',
    do_train=True,
    do_eval=True,
    num_train_epochs=3,
    overwrite_output_dir=True,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    gradient_accumulation_steps=2,
    save_strategy="no",
    log_level="info",
    save_total_limit=2,
    bf16=True
)
finetune_args = FinetuningArguments(task='summarization')
finetune_cfg = TextGenerationFinetuningConfig(
    model_args=model_args,
    data_args=data_args,
    training_args=training_args,
    finetune_args=finetune_args,
)
finetune_model(finetune_cfg)
```

The bottom status bar indicates "Mode: Edit" and "Ln 32, Col 29".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes several sub-directories and files, such as `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. Files are listed with their names and last modified dates.
- Terminal:** In the center, there is a terminal window titled "Terminal 1". It displays command-line output related to TensorFlow and Intel Extension for Transformers. Key logs include:
 - TensorFlow custom operations warning: `2024-07-08 06:42:54.809761: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.`
 - cuFFT factory registration error: `2024-07-08 06:42:54.837589: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:479] Unable to register cuFFT factory: Attempting to register factory for plugin cuFFT when one has already been registered`
 - cuDNN factory registration error: `2024-07-08 06:42:54.876218: E external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:10575] Unable to register cuDNN factory: Attempting to register factory for plugin cuDNN when one has already been registered`
 - cuBLAS factory registration error: `2024-07-08 06:42:54.876290: E external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1442] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered`
 - TensorFlow CPU instruction optimization: `2024-07-08 06:42:54.898686: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations. To enable the following instructions: AVX2 AVX512F AVX512_VNNI AVX512_BF16 AVX512_FP16 AVX_VNNI AMX_TILE AMX_INT8 AMX_BF16 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.`
 - TensorRT warning: `2024-07-08 06:42:56.603958: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT /home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1/lib/python3.10/site-packages/transformers/training_args.py:1489: FutureWarning: using `no_cuda` is deprecated and will be removed in version 5.0 of 🤗 Transformers. Use `use_cpu` instead`
 - Logger warning: `2024-07-08 06:42:57,826 - _logger.py - intel_extension_for_transformers.transformers.llm.finetuning - WARNING - Process rank: 0, device: cpu`
 - Distributed training information: `distributed training: True, 16-bits training: True`
 - Training parameters: `2024-07-08 06:42:57,828 - finetuning.py - intel_extension_for_transformers.transformers.llm.finetuning.finetuning - INFO - Training/evaluation parameters TrainingArguments(_n_gpu=0, accelerator_config={'split_batches': False, 'dispatch_batches': None, 'use_seedable_sampler': True, 'non_blocking': False, 'gradient_accumulation_kwargs': None}, adafactor=False, adam_beta1=0.9, adam_beta2=0.999, adam_epsilon=1e-08)`
- Notebook Area:** On the right, there are tabs for "single_node_finetuning_on_spr.ipynb" and "simple_llm_inference.ipynb". The status bar at the bottom indicates "Mode: Edit" and "Ln 32, Col 29".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / GenAI / intel-extension-for-transformers /`. The sidebar includes a search bar labeled "Filter files by name" and a sorting section with "Name" and "Modified".
- Terminal:** In the center, there is a terminal window titled "Terminal 1" showing Python code related to finetuning. The code includes parameters like `gradient_accumulation_kwargs`, `adafactor`, `adam_beta1`, `adam_beta2`, `adam_epsilon`, `auto_find_batch_size`, `batch_eval_metrics`, `bf16`, `bf16_full_eval`, `data_seed`, `dataloader_drop_last`, `dataloader_num_workers`, `dataloader_persistent_workers`, `dataloader_pin_memory`, `dataloader_prefetch_factor`, `ddp_backend`, `ddp_broadcast_buffers`, `ddp_bucket_cap_mb`, `ddp_find_unused_parameters`, `ddp_timeout`, `debug`, `deepspeed`, `disable_tqdm`, `dispatch_batches`, `do_eval`, `do_predict`, `do_train`, `eval_accumulation_steps`, `eval_delay`, `eval_do_concat_batches`, `eval_steps`, `eval_strategy`, `evaluation_strategy`, `fp16`, `fp16_backend`, and `fp16 full eval`.
- Notebook Tabs:** At the top, there are tabs for "single_node_finetuning_on_spr.ipynb" and "simple_llm_inference.ipynb".
- Code Editor:** On the right, there is a code editor window titled "neural-chat-x" showing the same Python code as the terminal.

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes sub-directories like `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, and files like `env.gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. Most files were modified 11 days ago, except for `whisper-2024` which was modified 3 months ago.
- Terminal:** In the center, there is a terminal window titled "Terminal 1". It displays a large block of Python code, likely configuration or command-line arguments for a training process. The code includes parameters like `fp16_full_eval=False`, `fp16_opt_level=01`, `fsdp=[], fsdp_config={'min_num_params': 0, 'xla': False, 'xla_fsdp_v2': False, 'xla_fsdp_grad_ckpt': False}, fsdp_min_num_params=0, fsdp_transformer_layer_cls_to_wrap=None, full_determinism=False, gradient_accumulation_steps=2, gradient_checkpointing=False, gradient_checkpointing_kwargs=None, greater_is_better=None, group_by_length=False, half_precision_backend=auto, hub_always_push=False, hub_model_id=None, hub_private_repo=False, hub_strategy=every_save, hub_token=<HUB_TOKEN>, ignore_data_skip=False, include_inputs_for_metrics=False, include_num_input_tokens_seen=False, include_tokens_per_second=False, jit_mode_eval=False, label_names=None, label_smoothing_factor=0.0, learning_rate=5e-05, length_column_name=length, load_best_model_at_end=False, local_rank=0, log_level=info, log_level_replica=warning, log_on_each_node=True, logging_dir='./tmp/runs/Jul08_06-42-54_idc-training-gpu-compute-26', logging_first_step=False, logging_nan_inf_filter=True, logging_steps=500}`.
- Notebook Tab:** At the top, there are tabs for "single_node_finetuning_on_spr.ipynb" and "simple_llm_inference.ipynb".
- Bottom Status Bar:** The status bar at the bottom shows "Simple" mode, "Mode: Edit", "Ln 32, Col 29", "single_node_finetuning_on_spr.ipynb", and a notification icon.

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. Most files were modified 11 days ago, except for `whisper-2024` which was modified 3 months ago.
- Terminal:** In the center, there is a terminal window titled "Terminal 1". It displays a block of Python code related to training or inference parameters. The code includes variables like `local_rank=0`, `log_level=info`, `log_level_replica=warning`, `log_on_each_node=True`, `logging_dir=./tmp/runs/Jul08_06-42-54_idc-training-gpu-compute-26`, `logging_first_step=False`, `logging_nan_inf_filter=True`, `logging_steps=500`, `logging_strategy=steps`, `lr_scheduler_kwargs={}`, `lr_scheduler_type=linear`, `max_grad_norm=1.0`, `max_steps=-1`, `metric_for_best_model=None`, `mp_parameters=`, `neftune_noise_alpha=None`, `no_cuda=True`, `num_train_epochs=3`, `optim=adammw_torch`, `optim_args=None`, `optim_target_modules=None`, `output_dir=./tmp`, `overwrite_output_dir=True`, `past_index=-1`, `per_device_eval_batch_size=4`, `per_device_train_batch_size=4`, `prediction_loss_only=False`, `push_to_hub=False`, `push_to_hub_model_id=None`, `push_to_hub_organization=None`, `push_to_hub_token=<PUSH_TO_HUB_TOKEN>`, `ray_scope=last`, `remove_unused_columns=True`, `report_to=['tensorboard']`, `restore_callback_states_from_checkpoint=False`, and `resume_from_checkpoint=None`.
- Notebook:** On the right, there is a notebook tab titled "single_llm_inference.ipynb". Below it, a status bar shows "Mode: Edit" and "Ln 32, Col 29".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, there is a sidebar with a file tree. The root directory is `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes sub-directories like `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, and files like `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. A blue vertical bar highlights the `requirements-cpu.txt` file.
- Terminal:** In the center, there is a terminal window titled "Terminal 1". It displays Python code related to model configuration, including parameters like `remove_unused_columns=True`, `report_to=['tensorboard']`, and `resume_from_checkpoint=None`. It also shows a warning message about the `resume_download` parameter being deprecated. The terminal output ends with a configuration file path: `/home/u51a924cab5fb25f87f4f9c01e5e094f/.conda/envs/itrex-1/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132: FutureWarning: `resume_download` is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want to force a new download, use `force_download=True`.`
- Notebook:** On the right, there is a "Notebook" tab with a status indicator "neural-chat-x".
- Bottom Status Bar:** The bottom bar shows the notebook name "neural-chat-x | Unknown", the mode "Edit", line and column numbers "Ln 32, Col 29", and the current file "single_node_finetuning_on_spr.ipynb".

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, a sidebar displays a file tree under the path `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes several sub-directories and files such as `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. The files are sorted by **Name** and **Modified**.
- Terminal:** The central panel contains a terminal window titled "Terminal 1". It displays the contents of the `g.json` file, which defines a `LlamaConfig` object with various parameters. Below the JSON, several INFO-level log messages from `tokenization_utils_base.py` are shown, indicating the loading of tokenizer and added tokens files.
- Notebook:** A tab labeled "single_node_finetuning_on_spr.ipynb" is visible at the top, indicating the current notebook being used.
- Bottom Status Bar:** The status bar at the bottom provides information about the kernel ("neural-chat-x"), mode ("Edit"), line and column numbers ("Ln 32, Col 29"), and the current file ("single_node_finetuning_on_spr.ipynb").

The screenshot shows a Jupyter Notebook interface with the following components:

- File Browser:** On the left, a sidebar displays a tree view of files and folders under the path `/ ... / GenAI / intel-extension-for-transformers /`. The list includes `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. Most files were modified 11 days ago, except for `whisper-2024` which was modified 3 months ago.
- Terminal:** A terminal window titled "Terminal 1" is open, showing log output from a process named `odel`. The log details the loading of various files from a cache directory, including `added_tokens.json`, `special_tokens_map.json`, `tokenizer_config.json`, and `tokenizer.json`. It also shows the dataset info being overwritten and found in the cache. The log concludes with the instantiation of the `LLamaForCausalLM` model under default dtype `torch.bfloat16`.
- Notebook:** The main area contains two tabs: `single_node_finetuning_on_spr.ipynb` and `simple_llm_inference.ipynb`. The status bar at the bottom indicates the notebook is in "Edit" mode, on line 32, column 29, and the current tab is `single_node_finetuning_on_spr.ipynb`.

The screenshot shows a Jupyter Notebook interface with the following components:

- File Explorer:** On the left, it displays a file tree under the path `/ ... / GenAI / intel-extension-for-transformers /`. The tree includes several directories and files such as `conda_meta`, `docker`, `docs`, `examples`, `intel_extension_for_transfo...`, `tests`, `whisper-2024`, `workflows`, `env_gpu.sh`, `LICENSE`, `MANIFEST.in`, `README.md`, `requirements-cpu.txt`, `requirements.txt`, `SECURITY.md`, `setup.py`, and `third_party_programs.txt`. The files are sorted by name and modified date.
- Terminal:** The main area contains two terminal tabs:
 - single_node_finetuning_on_spr.ipynb:** Shows log output for loading a checkpoint. It includes messages about weights being used for initialization, configuration file loading from cache, and generation configuration parameters like `bos_token_id`, `do_sample`, `max_length`, `pad_token_id`, `temperature`, and `top_p`.
 - Map:** Shows progress at 100% for processing a dataset, with a rate of 79.34 examples/s. It also logs dataset caching and loading processes.
- Notebook:** A tab labeled "neural-chat-x" is visible in the top right corner.

The screenshot shows a Jupyter Notebook interface with a dark theme. On the left, there is a file browser pane displaying a directory structure under 'GenAI / intel-extension-for-transformers/'. The main area contains a terminal window and several code cells.

Terminal 1:

```
[INFO|trainer.py:2084] 2024-07-08 07:38:05,097 >> Total train batch size (w. parallel, distributed & accumulation) = 8
[INFO|trainer.py:2085] 2024-07-08 07:38:05,097 >> Gradient Accumulation steps = 2
[INFO|trainer.py:2086] 2024-07-08 07:38:05,098 >> Total optimization steps = 107,667
[INFO|trainer.py:2087] 2024-07-08 07:38:05,105 >> Number of trainable parameters = 4,194,304
[ 18/107667 59:39 < 6689:03:44, 0.00 it/s, Epoch 0.00/3]
```

Notebook Tabs: single_node_finetuning_on_s.ipynb (active), simple_llm_inference.ipynb

Code Cells:

```
[ ]: from transformers import TrainingArguments
from intel_extension_for_transformers.neural_chat.config import (
    ModelArguments,
    DataArguments,
    FinetuningArguments,
    TextGenerationFinetuningConfig,
)
from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-hf")
data_args = DataArguments(dataset_name="theblackcat102/evol-codealpaca-v1")
training_args = TrainingArguments(
    output_dir='./tmp',
    do_train=True,
    do_eval=True,
    num_train_epochs=3,
    overwrite_output_dir=True,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    gradient_accumulation_steps=2,
    save_strategy="no",
    log_level="info",
    save_total_limit=2,
    bf16=True
```

Toolbar: File, Edit, View, Run, Kernel, Tabs, Settings, Help

Status Bar: Mode: Command, neural-chat-x | Unknown, Ln 32, Col 29, single_node_finetuning_on_s.ipynb, 1

The screenshot shows a Jupyter Notebook interface with the following components:

- File Bar:** File, Edit, View, Run, Kernel, Tabs, Settings, Help.
- File Browser:** Shows a list of files and folders in the directory `/.../GenAI/intel-extension-for-transformers/`. The list includes:
 - conda_meta (modified 11 days ago)
 - docker (modified 11 days ago)
 - docs (modified 11 days ago)
 - examples (modified 11 days ago)
 - intel_extension_for_transfo... (modified 11 days ago)
 - tests (modified 11 days ago)
 - whisper-2024 (modified 3 months ago)
 - workflows (modified 11 days ago)
 - env_gpu.sh (modified 11 days ago)
 - LICENSE (modified 11 days ago)
 - MANIFEST.in (modified 11 days ago)
 - README.md (modified 11 days ago)
 - requirements-cpu.txt (modified 11 days ago)
 - requirements.txt (modified 11 days ago)
 - SECURITY.md (modified 11 days ago)
 - setup.py (modified 11 days ago)
 - third_party_programs.txt (modified 11 days ago)
- Terminal 1:** Displays the output of a command, likely related to model finetuning or inference. The output includes logs from `datasets.arrow_dataset`, `finetuning.py`, and `trainer.py`. Key lines include:

```
Caching processed dataset at /home/u51a924cab5fb25f87f4f9c0aa90b8bee7c28b81fa3fa6223d/cache-09eb8bddb0322802.arrow
2024-07-08 06:43:17,973 - arrow_dataset.py - datasets.arrow_dataset - INFO - Caching processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/cnn_dailymail/3.0.0/0.0.0/96df5e686bee6baa90b8bee7c28b81fa3fa6223d/cache-09eb8bddb0322802.arrow
Loading cached processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/cnn_dailymail/3.0.0/0.0.0/96df5e686bee6baa90b8bee7c28b81fa3fa6223d/cache-752542bc1ab8f84f.arrow
2024-07-08 07:38:04,108 - arrow_dataset.py - datasets.arrow_dataset - INFO - Loading cached processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/cnn_dailymail/3.0.0/0.0.0/96df5e686bee6baa90b8bee7c28b81fa3fa6223d/cache-752542bc1ab8f84f.arrow
Loading cached processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/cnn_dailymail/3.0.0/0.0.0/96df5e686bee6baa90b8bee7c28b81fa3fa6223d/cache-11efac47977e9948.arrow
2024-07-08 07:38:04,135 - arrow_dataset.py - datasets.arrow_dataset - INFO - Loading cached processed dataset at /home/u51a924cab5fb25f87f4f9c01e5e094f/.cache/huggingface/datasets/cnn_dailymail/3.0.0/0.0.0/96df5e686bee6baa90b8bee7c28b81fa3fa6223d/cache-11efac47977e9948.arrow
2024-07-08 07:38:04,136 - finetuning.py - intel_extension_for_transformers.transformers.llm.finetuning.finetuning - INFO - Using data collator of type DataCollatorForSeq2Seq
trainable params: 4,194,304 || all params: 6,742,609,920 || trainable%: 0.06220594176090199
[INFO]trainer.py:641] 2024-07-08 07:38:04,487 >> Using cpu_amp half precision backend
[INFO]trainer.py:804] 2024-07-08 07:38:05,054 >> The following columns in the training set don't have a corresponding argument in `PeftModelForCausalLM.forward` and have been ignored: decoder_input_ids, decoder_attention_mask, decoder_labels. If decoder_input_ids, decoder_attention_mask, decoder_labels are not expected by `PeftModelForCausalLM.forward`, you can safely ignore this message.
[INFO]trainer.py:2078] 2024-07-08 07:38:05,094 >> ***** Running training *****
[INFO]trainer.py:2079] 2024-07-08 07:38:05,095 >> Num examples = 287,113
[INFO]trainer.py:2080] 2024-07-08 07:38:05,095 >> Num Epochs = 3
[INFO]trainer.py:2081] 2024-07-08 07:38:05,096 >> Instantaneous batch size per device = 4
[INFO]trainer.py:2084] 2024-07-08 07:38:05,097 >> Total train batch size (w. parallel, distributed & accumulation) = 8
[INFO]trainer.py:2085] 2024-07-08 07:38:05,097 >> Gradient Accumulation steps = 2
[INFO]trainer.py:2086] 2024-07-08 07:38:05,098 >> Total optimization steps = 107,667
[INFO]trainer.py:2087] 2024-07-08 07:38:05,105 >> Number of trainable parameters = 4,194,304
[ 18/107667 59:39 < 6689:03:44, 0.00 it/s, Epoch 0/0/3]
```
- Notebook Area:** Shows tabs for `single_node_finetuning_on_s.ipynb` and `simple_llm_inference.ipynb`.
- Bottom Status Bar:** Shows the mode (Command), line number (Ln 32, Col 29), file name (`single_node_finetuning_on_s.ipynb`), and a bell icon.

<https://jupyter-batch-us-region-1.cloud.intel.com/hub/user-redirect/lab/tree/Training/AI/GenAI/intel-extension-for-transformers>

Architecture Diagram

Build your chatbot

Text Chat

Giving NeuralChat the textual instruction, it will respond with the textual response.

```
[8]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat import build_chatbot, PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
print(chatbot)
response = chatbot.predict(query="Tell me about Intel Xeon Scalable Processors.")
print(response)
```

```
+-----+  
|       User  
| (query)  
|       |  
|       v  
+-----+  
|       Chatbot Interface  
| (intel_extension_for_transformers)  
+-----+  
|       |  
|       v  
+-----+  
|       Build Chatbot Function  
| - build_chatbot(config)  
+-----+  
|       |  
|       v  
+-----+  
|       Pipeline Configuration Object  
| - PipelineConfig  
| - MixedPrecisionConfig  
+-----+  
|       |  
|       v
```

```
|  
|       v  
+-----+  
|           Chatbot Model  
+-----+  
|           |  
|           v  
+-----+  
|           Prediction Function  
| - chatbot.predict(query)  
+-----+  
|           |  
|           v  
+-----+  
|           Response Generation  
| - Response: "..."  
+-----+  
|           |  
|           v  
+-----+  
|           Output to User  
| - print(response)  
+-----+
```

Text Chat With Retrieval Plugin



User could also leverage NeuralChat Retrieval plugin to do domain specific chat by feeding with some documents like below:

```
[ ]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/retrieval/  
!pip install -r requirements.txt  
%cd ../../../../../../
```

```
[ ]: !mkdir docs  
%cd docs  
!curl -OL https://raw.githubusercontent.com/intel/intel-extension-for-transformers/main/intel_extension_for_transformers/neural_chat/assets/docs/  
!curl -OL https://raw.githubusercontent.com/intel/intel-extension-for-transformers/main/intel_extension_for_transformers/neural_chat/assets/docs/  
!curl -OL https://raw.githubusercontent.com/intel/intel-extension-for-transformers/main/intel_extension_for_transformers/neural_chat/assets/docs/  
%cd ..
```

```
[ ]: from intel_extension_for_transformers.neural_chat import PipelineConfig  
from intel_extension_for_transformers.neural_chat import build_chatbot  
from intel_extension_for_transformers.neural_chat import plugins  
plugins.retrieval.enable=True  
plugins.retrieval.args["input_path"]="./docs/"  
config = PipelineConfig(plugins=plugins)  
chatbot = build_chatbot(config)  
response = chatbot.predict("How many cores does the Intel® Xeon® Platinum 8480+ Processor have in total?")  
print(response)
```

```
+-----+  
|       User  
|       (query)  
|       |  
|       v  
+-----+  
|   Chatbot Interface  
|   (intel_extension_for_  
|   transformers)  
+-----+  
|       |  
|       v  
+-----+  
|   Setup Environment  
|   - Change Directory  
|   - Install Packages  
|   - Create `docs` Folder  
|   - Download Documents  
+-----+  
|       |  
|       v  
+-----+  
|   Configure and Build  
|   Chatbot  
+-----+  
|       |  
|       v
```

```
|       |  
|       v  
+-----+  
|   |   Prediction Function  
|   |   - Process Query  
+-----+  
|       |  
|       v  
+-----+  
|   |   Retrieve Data from  
|   |   Documents  
+-----+  
|       |  
|       v  
+-----+  
|   |   Generate Response  
+-----+  
|       |  
|       v  
+-----+  
|   |   Output Response to User  
+-----+  
|       |  
+-----+
```

Voice Chat with ASR & TTS Plugin

In the context of voice chat, users have the option to engage in various modes: utilizing input audio and receiving output audio, employing input audio and receiving textual output, or providing input in textual form and receiving audio output.

For the Python API code, users have the option to enable different voice chat modes by setting ASR and TTS plugins enable or disable.

```
[ ]: %cd ./intel-extension-for-transformers/intel_extension_for_transformers/neural_chat/pipeline/plugins/audio/
!pip install -r requirements.txt
%cd ../../../../../../../
```

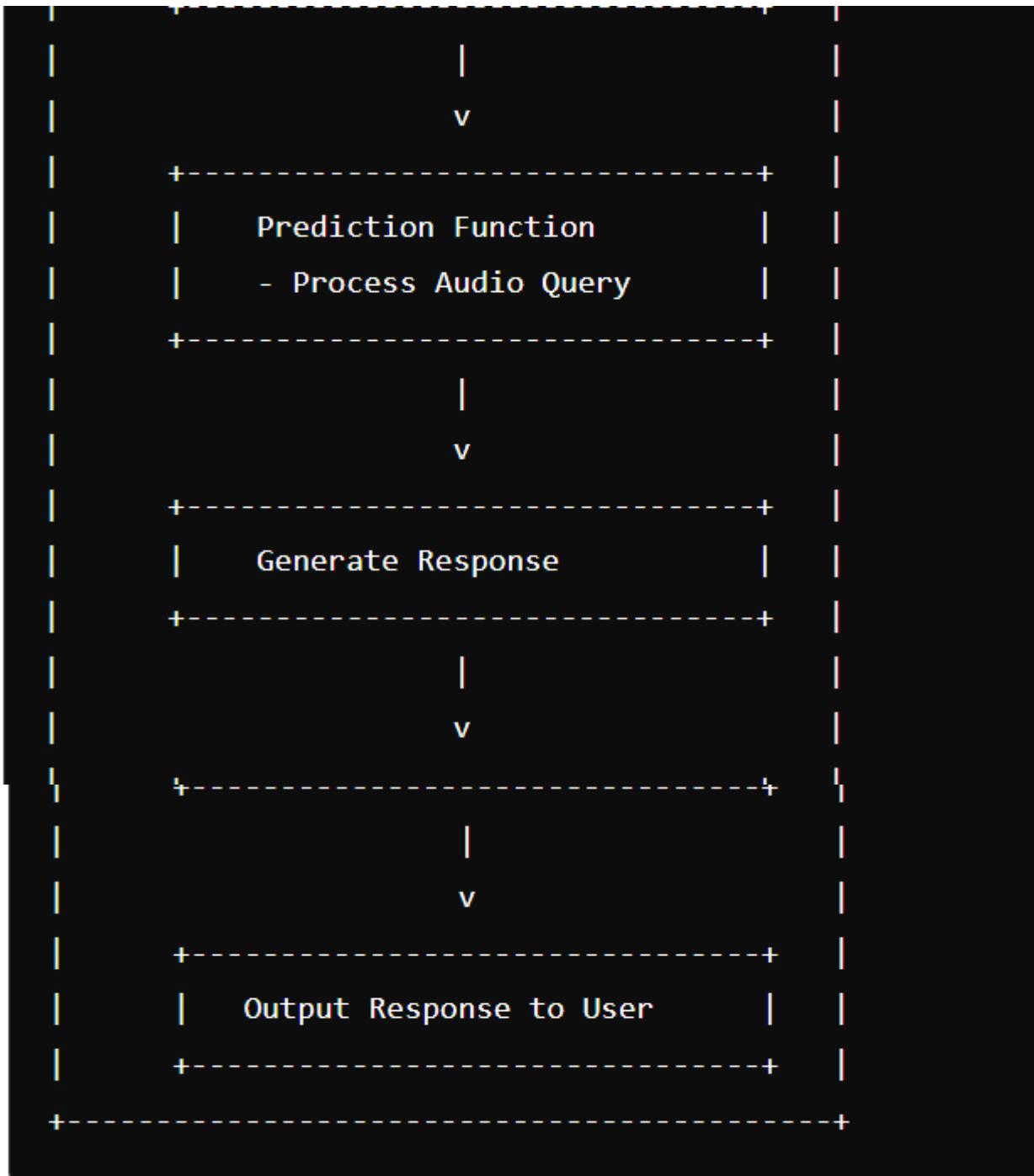
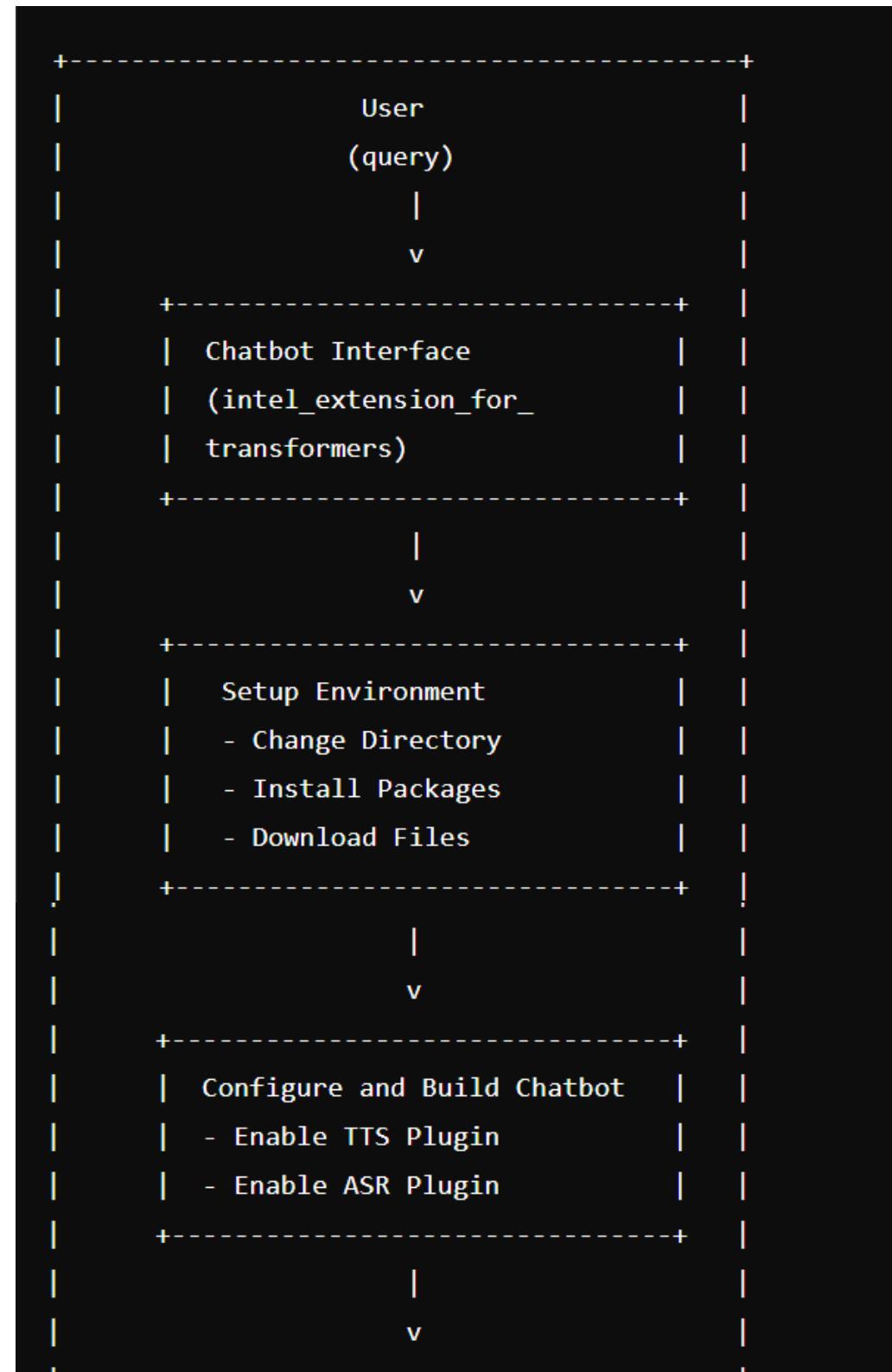


```
[ ]: !curl -OL https://raw.githubusercontent.com/intel/intel-extension-for-transformers/main/intel_extension_for_transformers/neural_chat/assets/speak
!curl -OL https://raw.githubusercontent.com/intel/intel-extension-for-transformers/main/intel_extension_for_transformers/neural_chat/assets/audio
◀ ━━━━━━ ▶
```



```
[ ]: from intel_extension_for_transformers.neural_chat import PipelineConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import plugins
plugins.tts.enable = True
plugins.tts.args["output_audio_path"] = "./response.wav"
plugins.asr.enable = True

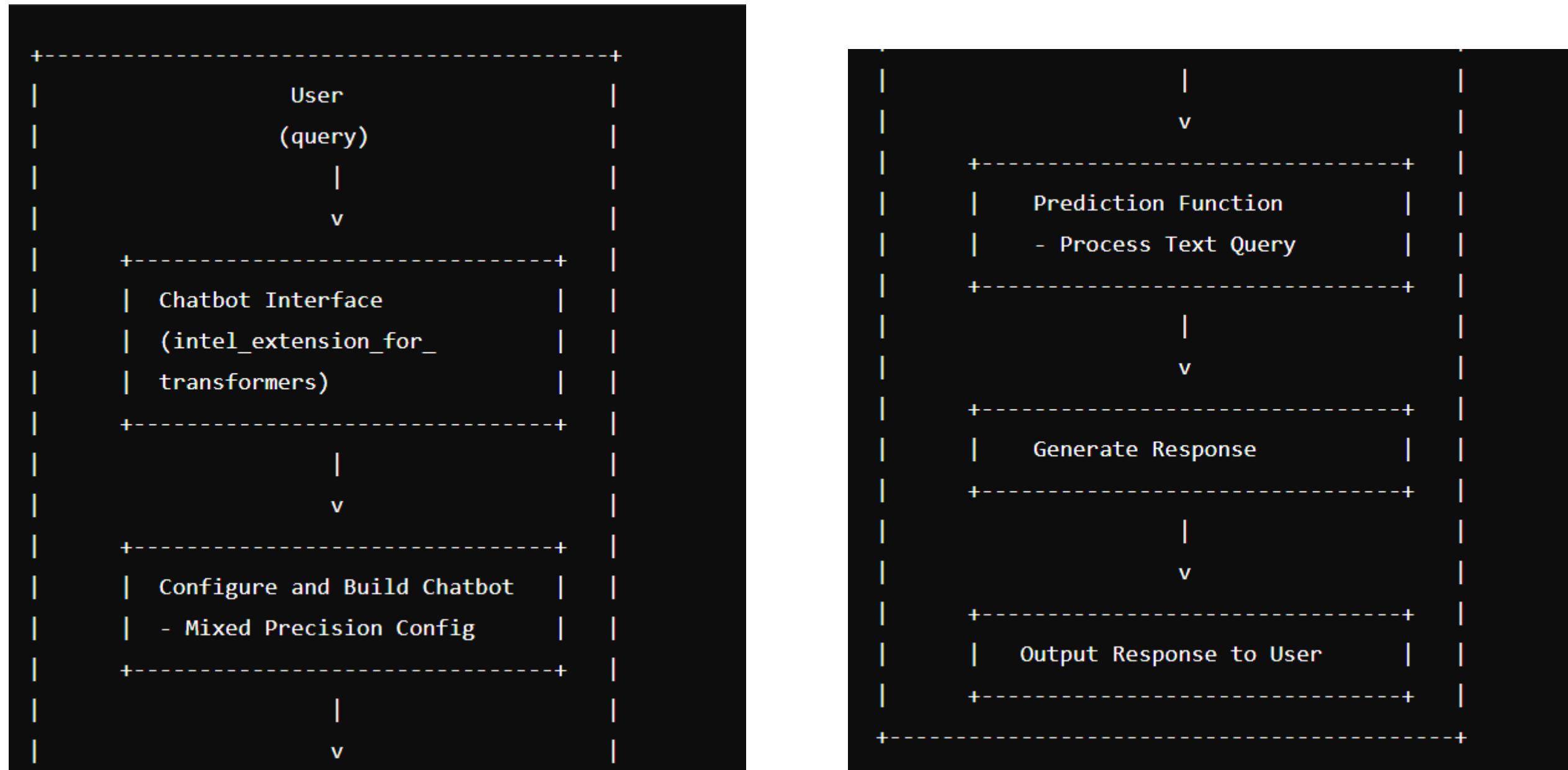
config = PipelineConfig(plugins=plugins)
chatbot = build_chatbot(config)
result = chatbot.predict(query="./sample.wav")
print(result)
```



Low Precision Optimization

BF16

```
[ ]: # BF16 Optimization
from intel_extension_for_transformers.neural_chat.config import PipelineConfig
from intel_extension_for_transformers.transformers import MixedPrecisionConfig
config = PipelineConfig(optimization_config=MixedPrecisionConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about Intel Xeon Scalable Processors.")
print(response)
```

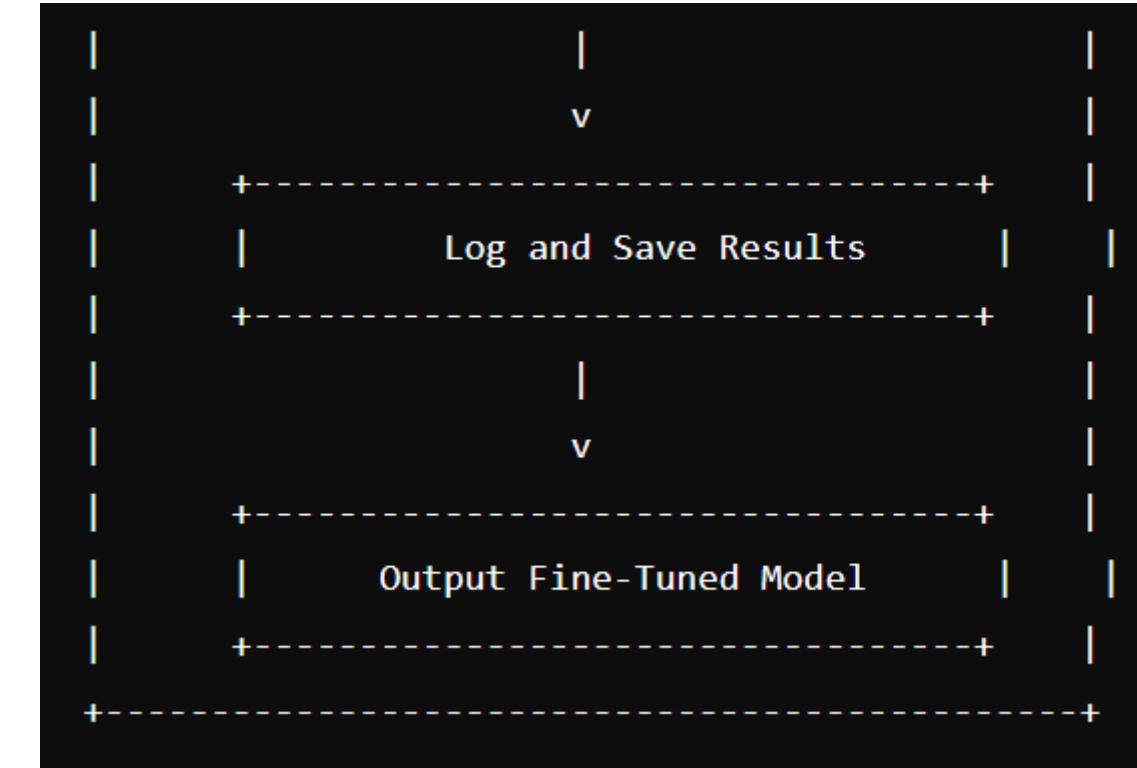
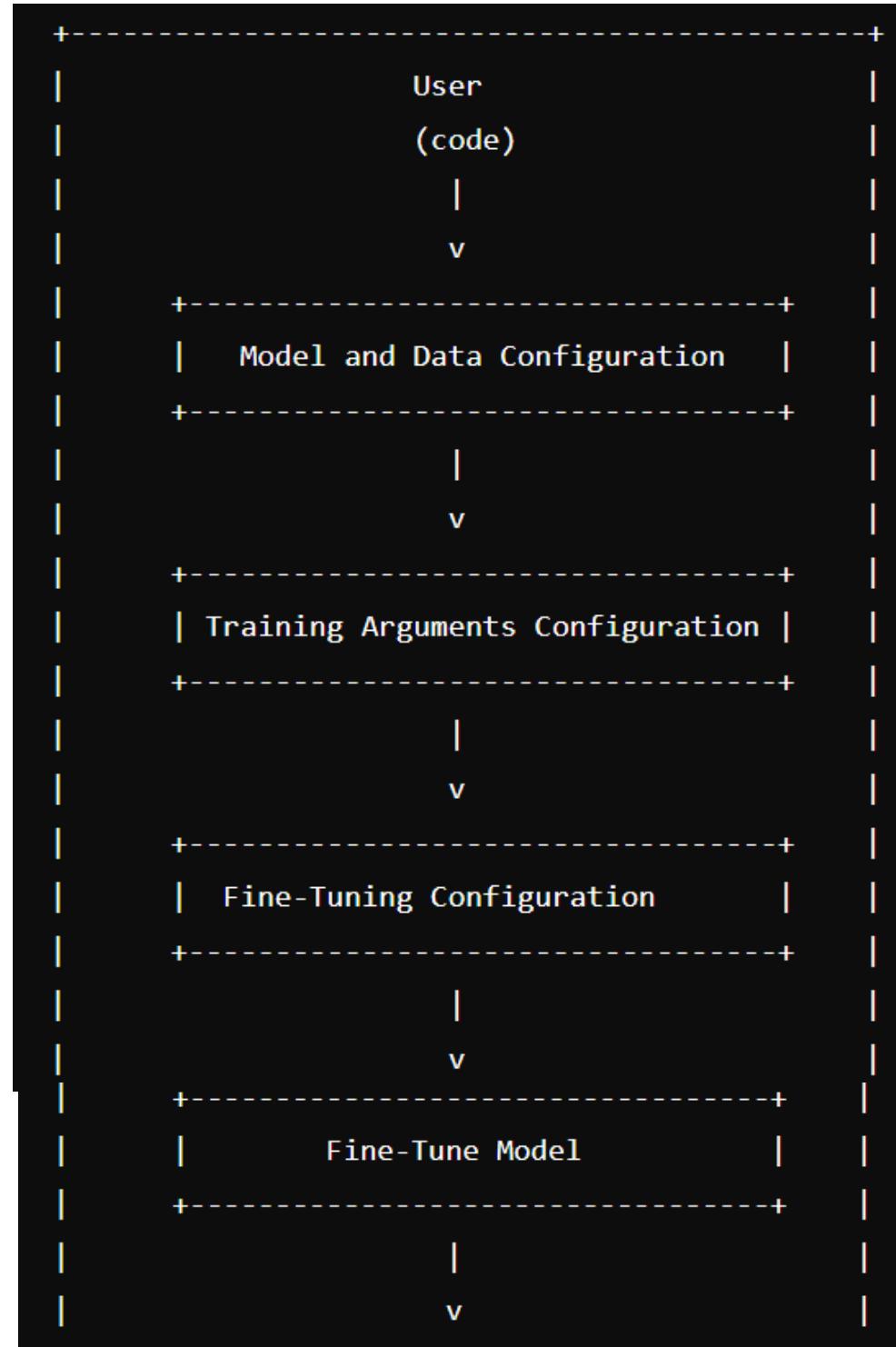


Finetune Your Chatbot

We employ the LoRA approach to finetune the LLM efficiently.

Finetune the model on Alpaca-format dataset to conduct text generation:

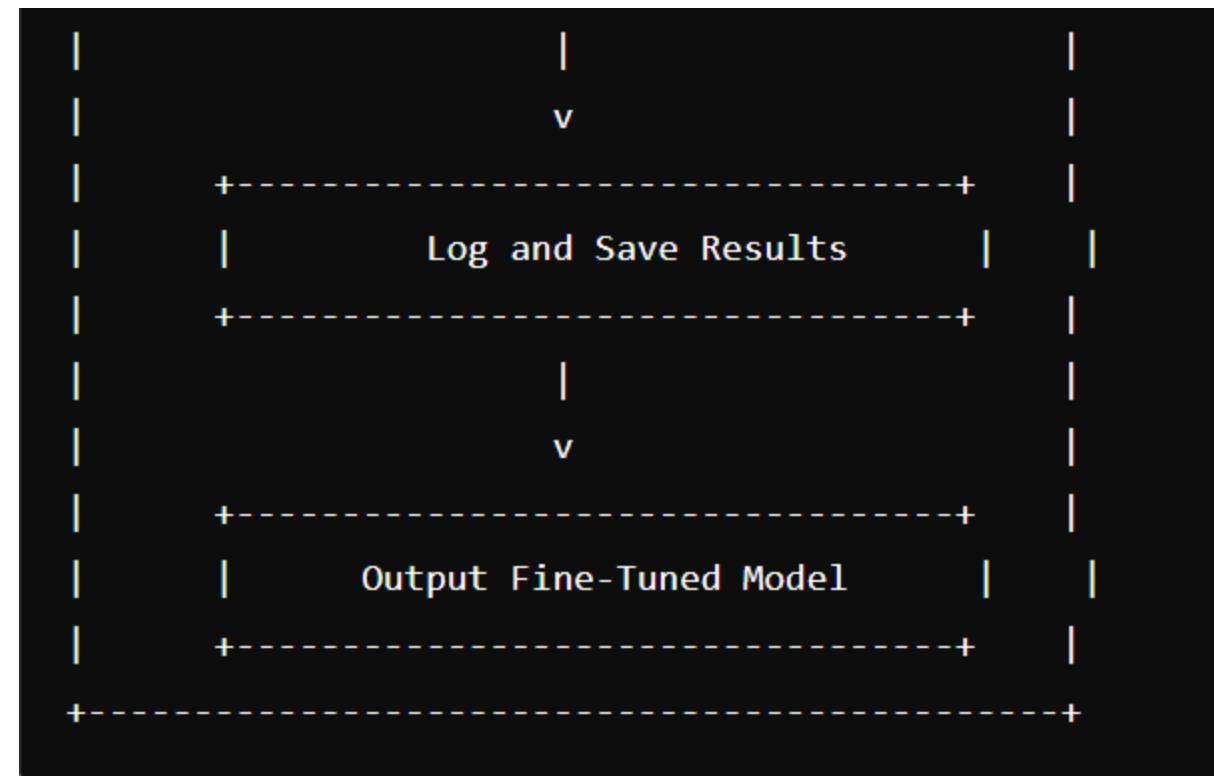
```
[ ]: from transformers import TrainingArguments
      from intel_extension_for_transformers.neural_chat.config import (
          ModelArguments,
          DataArguments,
          FinetuningArguments,
          TextGenerationFinetuningConfig,
      )
      from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
      model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-chat-hf")
      data_args = DataArguments(train_file="alpaca_data.json", validation_split_percentage=1)
      training_args = TrainingArguments(
          output_dir='./tmp',
          do_train=True,
          do_eval=True,
          num_train_epochs=3,
          overwrite_output_dir=True,
          per_device_train_batch_size=4,
          per_device_eval_batch_size=4,
          gradient_accumulation_steps=2,
          save_strategy="no",
          log_level="info",
          save_total_limit=2,
          bf16=True,
      )
      finetune_args = FinetuningArguments()
      finetune_cfg = TextGenerationFinetuningConfig(
          model_args=model_args,
          data_args=data_args,
          training_args=training_args,
          finetune_args=finetune_args,
      )
      finetune_model(finetune_cfg)
```



Finetune the model on the summarization task:

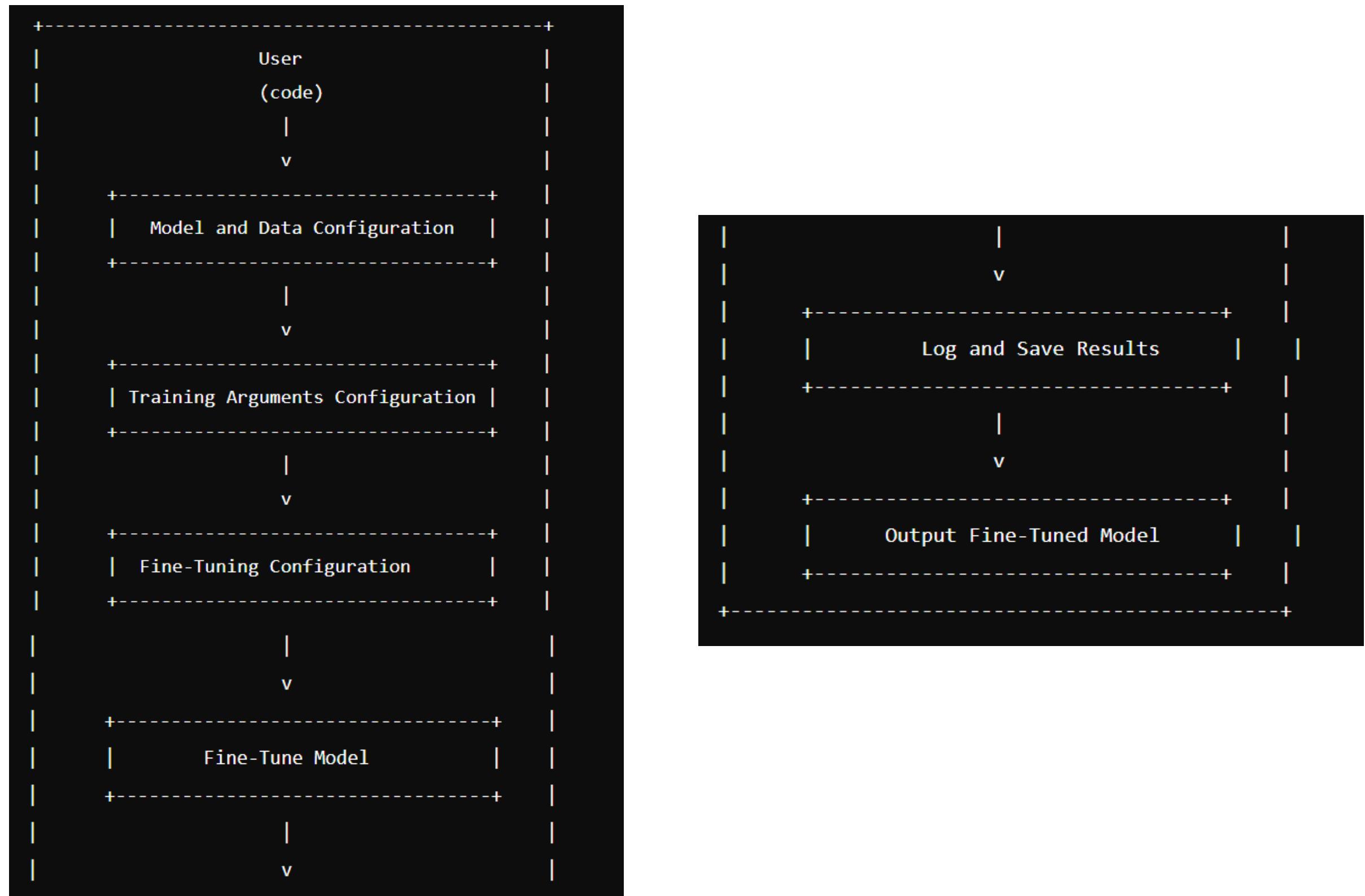
```
[ ]: from transformers import TrainingArguments
      from intel_extension_for_transformers.neural_chat.config import (
          ModelArguments,
          DataArguments,
          FinetuningArguments,
          TextGenerationFinetuningConfig,
      )
      from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
      model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-chat-hf")
      data_args = DataArguments(dataset_name="cnn_dailymail", dataset_config_name="3.0.0")
      training_args = TrainingArguments(
          output_dir='./tmp',
          do_train=True,
          do_eval=True,
          num_train_epochs=3,
          overwrite_output_dir=True,
          per_device_train_batch_size=4,
          per_device_eval_batch_size=4,
          gradient_accumulation_steps=2,
          save_strategy="no",
          log_level="info",
          save_total_limit=2,
          bf16=True
      )
      finetune_args = FinetuningArguments(task='summarization')
      finetune_cfg = TextGenerationFinetuningConfig(
          model_args=model_args,
          data_args=data_args,
          training_args=training_args,
          finetune_args=finetune_args,
      )
```

```
finetune_model(finetune_cfg)
```



Finetune the model on the code generation task:

```
[ ]: from transformers import TrainingArguments
      from intel_extension_for_transformers.neural_chat.config import (
          ModelArguments,
          DataArguments,
          FinetuningArguments,
          TextGenerationFinetuningConfig,
      )
      from intel_extension_for_transformers.neural_chat.chatbot import finetune_model
      model_args = ModelArguments(model_name_or_path="meta-llama/Llama-2-7b-chat-hf")
      data_args = DataArguments(dataset_name="theblackcat102/evol-codealpaca-v1")
      training_args = TrainingArguments(
          output_dir='./tmp',
          do_train=True,
          do_eval=True,
          num_train_epochs=3,
          overwrite_output_dir=True,
          per_device_train_batch_size=4,
          per_device_eval_batch_size=4,
          gradient_accumulation_steps=2,
          save_strategy="no",
          log_level="info",
          save_total_limit=2,
          bf16=True
      )
      finetune_args = FinetuningArguments(task='code-generation')
      finetune_cfg = TextGenerationFinetuningConfig(
          model_args=model_args,
          data_args=data_args,
          training_args=training_args,
          finetune_args=finetune_args,
      )
      finetune_model(finetune_cfg)
```



Technologies Used

Intel® Developer Cloud (IDC)

Intel extensions for Transformers
Git-Hub

Intel Server in college

Intel unnati Lab in college



Team Members and Contribution

ABHINAV NAIR S

Working of entire notebook and Report writing.

ARDRA A NAIR

Working of notebook and Report writing

ESA MARIA SHYJU

Report writing

CHAITHANYA P SUNIL

Report writing

SREERAM R NAIR

Report writing

Conclusion

In this report, we explored the fundamental concepts of Generative AI, focusing on the use of simple language model (LLM) interfaces on CPUs and the process of fine-tuning LLMs to create custom chatbots . By understanding the basics of how LLMs function and their applications, we recognized the potential these models hold in transforming various domains through intelligent automation and personalized user interactions. Initially, we delved into the mechanics of LLMs, discussing their architecture, capabilities, and limitations, especially when operating on CPU-based systems. This foundation set the stage for practical applications, highlighting how even with hardware constraints, these models can deliver valuable results. The core of our study involved the fine-tuning process, where we adapted a pre-trained LLM to cater to specific needs. This customization not only demonstrated the flexibility of LLMs but also emphasized the importance of domain-specific data and fine-tuning techniques to enhance the model's performance in targeted applications. By creating a custom chatbot , we illustrated a tangible outcome of this process, showcasing how tailored interactions can significantly improve user experiences

