

GIGABYTE™

AI TOP Utility

Version 4.2.0

User Manual

1. Installation and preparation	P.2~7
1-1. Environment preparation	P.2
1-2. Hardware condition	P.2~3
1-3. Graphic card driver installation	P.3~4
1-4. AI TOP Utility Software installation	P.4
1-5. Mount SSD for Linux (optional)	P.5
1-6. Multinode setting (optional)	P.5~7
2. Terms and functions description	P.7~14
2-1. Dashboard	P.7~8
2-2. Dataset	P.8~9
2-3. Experiment	P.9~12
2-3-1. LLM	P.9~12
2-3-2. LMM	P.12
2-4. Inference	P.12~13
2-5. RAG	P.13
2-6. Model Converter	P.13~14
2-7. Machine Learning	P.14

2-8. Settings	P.14
3. Operation tutorial	P.14~51
3-1. How to generate a well organized dataset from a rough dataset?	P.15~17
3-2. How to finetune a pre-trained LLM model based on a specified dataset?	P.17~23
3-3. How to fine-tune a pre-trained LLM model using multi-node cluster?	P.24~25
3-4. How to monitor the fine-tuning process in the Dashboard tab?	P.25~29
3-5. Inference	P.29~36
3-6. RAG	P.36~38
3-7. Model conversion	P.38~39
3-8. Machine Learning	P.39~45
3-9. Finetune Expand config syntax	P.45~51
4. Supported Models	P.51~53
4-1. Supported list of LLM models	P.51
4-2. Supported list of LMM models	P.52
4-3. Supported list of embedding models (RAG)	P.52
4-4. Download Model	P.52~53

1. Installation and Preparation

1-1. Environment preparation

- Linux installation: Ubuntu 24.04.2 LTS (Linux Kernel: 6.14.0-generic) installation.

Download ubuntu-24.04.2-desktop-amd64.iso file via: <https://releases.ubuntu.com/noble/>

- Windows WSL installation: please directly refer to section 1-4.

1-2. Hardware condition

AI TOP Utility only supports Gigabyte hardware.

1-2-1. Motherboard

- Other old GIGABYTE motherboards with the newest BIOS, such as
 - Intel :
 - Z890 Aorus Pro ICE
 - H810M S2H, Z790 Aorus Master X
 - B760 Gaming X AX
 - B660 Aorus Master
 - H610M D3W WIFI6
 - Z690 Aorus Master
 - AMD :
 - A620 A620M GAMING X 1.0
 - B650 B650 AORUS ELITE AX 1.2
 - X670 X670E AORUS MASTER 1.03
 - B840 B840M DS3H 1.0
 - B850 B850M EAGLE WIFI6E 1.1
 - X870E X870E AORUS XTREME X3D AI TOP 0.5
 - X870 X870 AORUS ELITE WIFI7 1.11
 - A620A A620I AX 2.01
- TRX50 AI TOP
Reference link: <https://www.gigabyte.com/Motherboard/TRX50-AI-TOP#kf>
- TRX50 AERO D
Reference link: <https://www.gigabyte.com/Motherboard/TRX50-AERO-D-rev-12#kf>
- W790 AI TOP
Reference link: <https://www.gigabyte.com/Motherboard/W790-AI-TOP#kf>
- Z890 AORUS XTREME AI TOP
Reference link: <https://www.gigabyte.com/Motherboard/Z890-AORUS-XTREME-AI-TOP#kf>
- Z890 AORUS MASTER AI TOP
Reference link: <https://www.gigabyte.com/Motherboard/Z890-AORUS-MASTER-AI-TOP#kf>

- Z890 AI TOP

Reference link: <https://www.gigabyte.com/Motherboard/Z890-AI-TOP#kf>

- X870E AORUS XTREME AI TOP

Reference link: <https://www.gigabyte.com/Motherboard/X870E-AORUS-XTREME-AI-TOP#kf>

- B850 AI TOP

Recommend minimum memory of DRAM: 64GB ~ 128GB

1-2-2. Graphics card

Reference link: <https://www.gigabyte.com/Graphics-Card>

- Gigabyte GeForce RTX 50 series: RTX 5090, RTX 5090 D, RTX 5080 SUPER, RTX 5080, RTX 5070 Ti SUPER, RTX 5070 Ti, RTX 5070 SUPER, RTX 5070, RTX5060 Ti.
- Gigabyte GeForce RTX 40 series: RTX 4090, RTX 4090 D, RTX 4080 SUPER, RTX 4080, RTX 4070 Ti SUPER, RTX 4070 Ti, RTX 4070 SUPER, RTX 4070, RTX 4060 Ti, RTX 4060.
- GIGABYTE Radeon™ AI PRO R9700 AI TOP 32G; Gigabyte Radeon Pro W7000 and Radeon RX 7000 series: Radeon Pro W7900, Radeon Pro W7800; Radeon RX 7900 XTX, Radeon RX 7900 XT, Radeon RX 7900 GRE.

1-2-3. SSD

For the OS SSD, we recommend that users use an M.2 SSD NVMe with 1TB ~ 2TB storage.

For the offloading SSD, AI TOP Utility supports ultra-durable SSDs to meet the high demands of read and write memory during the LLM training process. If no ultra-durable SSD is used in your PC, AI TOP Utility will not enable the SSD offloading memory options.

- GIGABYTE AI TOP 100E SSD 1TB (AI100E1TB)
- GIGABYTE AI TOP 100E SSD 2TB (AI100E2TB)
- GIGABYTE AI TOP 100E SSD 320G (AI100E320G-2B)
- GIGABYTE AI TOP 100E SSD 1TB (AI100E1TB-2B)
- GIGABYTE AI TOP 100E SSD 2TB (AI100E2TB-2B)
- Phison aiDAPTIVCache ai100 SSD

For safety, we highly recommend that users use the SSD models mentioned above for mounting SSDs for offloading options.

1-3. Graphic card driver installation:

1-3-1. If Nvidia graphic cards are used, please install:

- For **Ubuntu 24.04** :
NVIDIA driver version 570 installation + CUDA 12.9 installation.

For quick installation, please run the following command in the terminal:

```
bash nvidia_driver_cuda_2404.sh
```

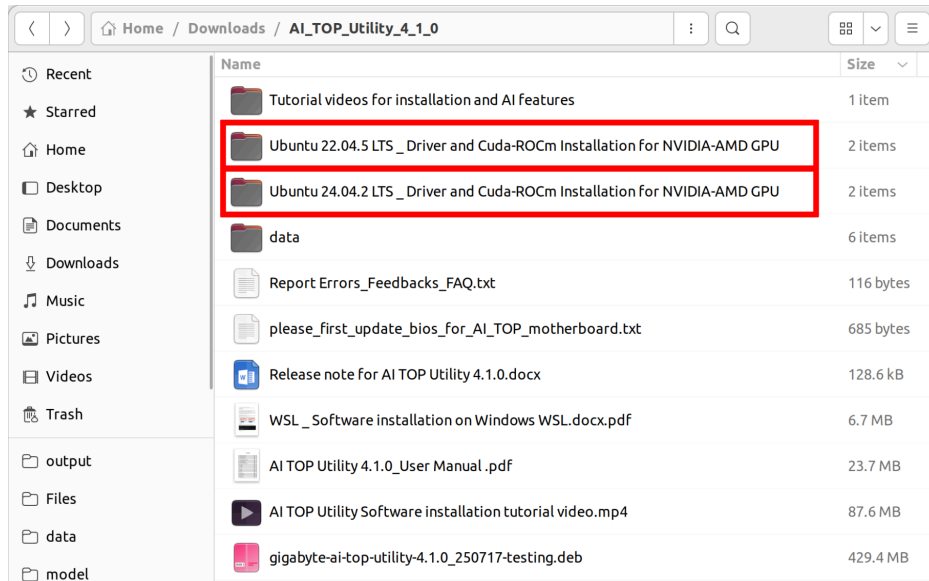
1-3-2. If AMD graphic cards are used, please install:

- For **Ubuntu 24.04** :

AMD driver version 6.4.60403-1 installation

For quick installation, please run the following command in the terminal:

```
bash amd_driver_rocm_2404.sh
```



*Note: Please reboot your PC after the driver installation above is finished.

1-4. AI TOP Utility Software installation

Please visit [this link](#) to download the installation package. If you are using **Windows** as your operating system, please refer to the "[WSL_Software_installation_on_Windows_WSL.pdf](#)".

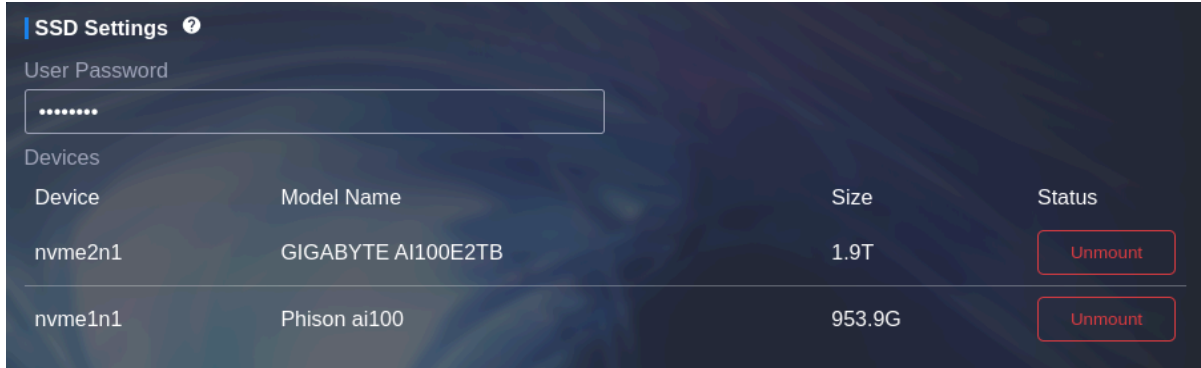
After downloading, copy the package to your PC's home directory, extract the .zip file. Double-click the .deb file and click **Install**. When the authentication window appears, enter your system password to continue. After a short while, the **Install** button will change to **Open**. Click it to launch a terminal window and the installation process will begin.

Please monitor the installation closely, as you will need to enter your system password when prompted.

After the installation is complete, create two new folders named "model" and "data" in your home directory. This step should be performed only once, immediately after installation. Do not repeat it during future updates to avoid overwriting or deleting your existing models, data, or checkpoints.

1-5. Mount SSD for Linux (optional):

In cases the model size is too large for VRAM and DRAM to handle during training, users need to offload the training memory to an NVMe SSD for additional capacity. To do this, users must first mount an ultra-durable NVMe SSD by entering their OS user password and clicking the 'Mount' button in the Settings tab.



If you want to format the SSD to EXT4 (e.g., Ubuntu)

```
sudo mkfs.ext4 /dev/sdb
```

This step **will erase all data** on the SSD. This formats the SSD /dev/sdb to the EXT4 filesystem.

1-6. Multinode setting (optional):

A multinode setup via SSH between PCs enables the machines to communicate and work together, often for distributed computing or task management. Here's a brief introduction on how to set it up between two PCs (**The usernames of both PCs must be the same!**):

1. Install SSH packages :

Open the terminal with a stable internet connection. Then, install the necessary packages for the multinode setup on both PCs.

```
sudo apt install openssh-server
```

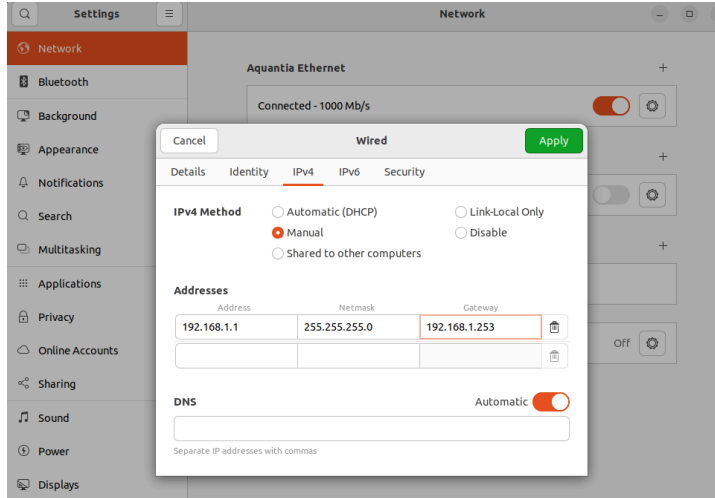
```
sudo apt-get install sshpass
```

2. Connect the two PCs using a network cable or Thunderbolt 4/5:

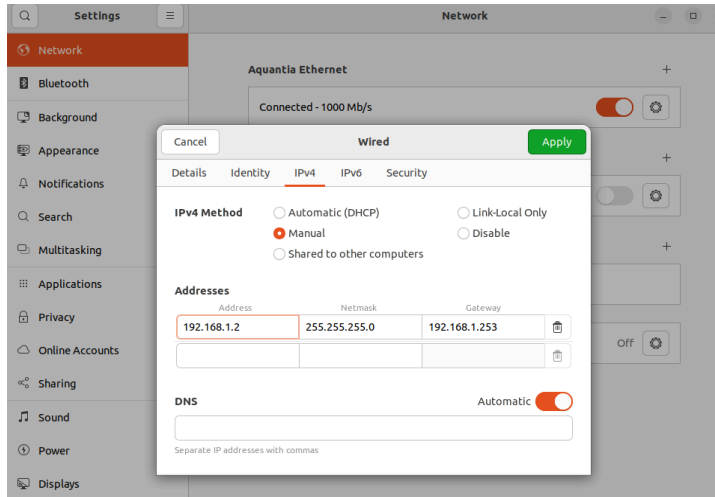
Disconnect each PC from the internet, then connect the two PCs using a network cable (CAT6 or higher) via the LAN port or a Thunderbolt 4/5 cable via the Type-C port on their respective motherboards.

3. Set the static IP address of the two PCs (First PC):

<Disconnect the internet>

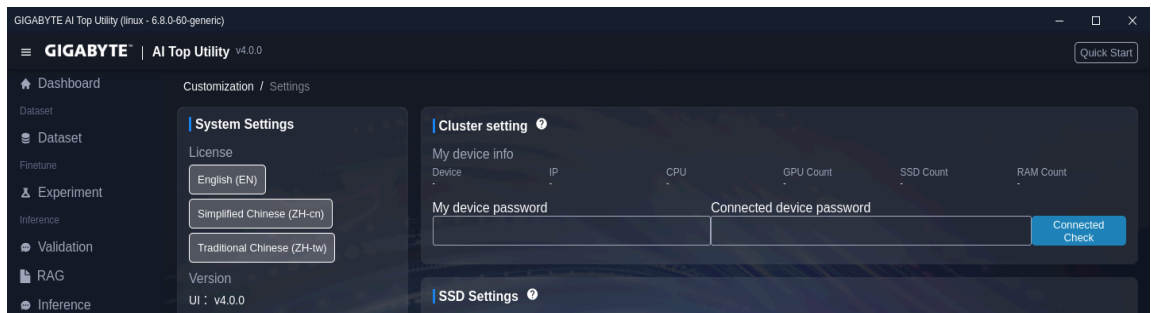


Set the static IP addresses of two PCs (Second PC):
<Disconnect internet>



4. Connect 2 PCs via SSH :

Enter the OS password for each PC and click 'Connect Check' to establish the connection between the two PCs via SSH. Please note that the passwords are used solely for connecting the two PCs while the internet is disconnected during multinode training. We do not collect or store end-user passwords for any purpose.





5. Now you can refer to the instructions in Section 3-3 about how to fine-tune a pre-trained LLM model using a multi-node cluster.

2. Terms and functions description

2-1. Dashboard

Experiment name	The name of your experiment job is defined in the Experiment tab.
Dataset	Name of dataset selected in Experiment tab.
ETA	Estimated time of accomplishment of the experiment job.
Accumulated Power Consumption (kWh)	The accumulated power consumption (kWh) is measured from the time the PC device is booted. For now, this feature only supports the Gigabyte GP-AP1600TM AI TOP PSU.
Current Power Consumption (W)	The current power consumption (W) of the PSU during a training job is the total capacity minus the power consumed. For now, this feature only supports Gigabyte GP-AP1600TM AI TOP PSU.
Progress bar	The progress bar shows the current completion percentage (%) of the experiment job.
Status	The status of the experiment job can be 'running,' 'stopped,' or 'draft' (not started).
Current GPU / CPU load	The usage (%) of GPU/CPU at the moment. Indicates the current percentage of GPU and CPU processing power utilization. This is often monitored to ensure that resources are being efficiently utilized and to diagnose performance bottlenecks.
DRAM usage	The loading state (of total memory) of System DRAM in the training process. Refers to the amount of Dynamic Random Access Memory (DRAM) being used. In the context of training, this is important as it affects the amount of data that can be processed in parallel.
NVMe SSD usage	The loading state (of total memory) of the NVMe SSD in the training process.

	Indicates how much of the NVMe (Non-Volatile Memory Express) SSD storage is being used. High usage could affect the speed at which data is read or written during training, especially when dealing with large datasets.
VRAM Loading state (%)	The loading state (% of total memory) of each GPU in the training process. The percentage of VRAM on the GPU currently in use. VRAM is crucial for storing the model, its weights, and the training data batches when utilizing GPUs.
DRAM Loading state (%)	The loading state (% of total memory) of System DRAM in the training process. Shows the percentage of DRAM being utilized. This metric helps in understanding if there are memory constraints that could be impacting the performance of the training process.
SSD Loading state (%)	The loading state (% of total memory) of NVMe SSD in the training process. The percentage of the SSD's capacity that is currently in use. It is important for understanding how much of the storage resource is occupied, which can influence data loading times during training.
CPU Loading state (%)	The average loading state (% of total memory) of all CPU cores in the training process. The percentage of CPU capacity currently in use. Monitoring this helps in understanding the CPU's role and load during the training process, particularly for tasks not offloaded to the GPU.
Training Configs	This encompasses all the settings that users have just set up in the experiment tab. This helps users to double check the configs with current hardware states in order to get better change in next training settings.
Loss	Monitor the training loss at every step. A metric that quantifies the difference between the predicted values by the model and the actual values in the training data. The goal of training is typically to minimize the loss, which is indicative of the model's performance and accuracy.
Logs	Monitor whatever happens in the training process and check error messages. These are records of events that occur during the training process. Logs include information about the training progress, performance metrics, error messages, and other diagnostic information. They are crucial for debugging and optimizing the training process.

2-2. Dataset

Dataset Folder	Select the folder of your raw datasets.
Create Dataset	Start by creating a well-organized dataset for the fine-tuning job in the

	Experiment tab.
Upload File	Select files of raw dataset you want to use for dataset creation.
Recommend LLM models	The system to recommend a list of LLM models may fit for data creation. Please download the model that you want to create the dataset in the next step.
Select LLM model	Select the LLM model you want to use for dataset creation.
Dataset Name	Select the name of your well-organized dataset to create.
No Limit of Q&A	Select the no limitation of the number of Q&A in your well-organized dataset.
Number of Q&A	In case of limitation, please fill the number of Q&A in your well-organized dataset.

2-3. Experiment

2-3-1. LLM

Experiment Name	Name of your fine-tuning experiment (job). Type a name of your fine-tuning experiment.
LLM Backbone	Refers to the pre-trained model architecture used as the starting point for fine-tuning. Select a pre-trained large language model (LLM) model that you want to process fine-tuning.
Output Directory	The folder path where the results of the fine-tuning (including the model weights, logs, and other outputs) are saved. Local path to save your model after fine-tuning.
Resume from checkpoint	Allows the fine-tuning process to start from a saved checkpoint rather than from scratch, useful for resuming interrupted training sessions. Continue fine-tuning your model from a specific checkpoint.
Datasets	Specifies the dataset used for training the model during fine-tuning. Select a dataset for fine-tuning process (txt, csv, json, jsonl).
Fine-tuning Strategy	The unique technique by Gigabyte to provide the easy-to-operate modes used for LLM fine-tuning. It is used for non-AI base users to quickly set up the fine-tuning settings without deep understanding of the configurations. Select a strategy for your fine-tuning process.
Standard	Medium training speed, medium precision.
Fast Speed	Fast training speed, lower precision.
High Precision	Slow training speed, high precision. Normally it chooses full fine-tuning methods and precision dtype fp16/bf16 above.
Clear	Clear all easy-to-operate mode settings and customize every config

	setting step by step.
Fine-tuning Type	Describes the fine-tuning method, including: full, freeze, (q)lora.
full	<p>Description: All parameters of the model are updated during the training process. This approach involves adjusting the weights of all layers of the model based on new training data.</p> <p>Advantages: High performance can be achieved on specific tasks or data sets because the model fully adapts to the nuances of new data.</p> <p>Disadvantages: Comprehensive fine-tuning requires significant computing resources and may lead to overfitting if the fine-tuning data set is small compared to the original training data set. It can also lead to “catastrophic forgetting,” where the model loses its ability to perform on the task it was originally trained on.</p>
freeze	<p>Description: Some layers of the model remain frozen during fine-tuning (i.e. their weights are not updated), while other layers are allowed to update.</p> <p>Advantages: Freezing layers reduces computational cost and the number of parameters that need to be updated. It also helps retain the general knowledge learned by the model and mitigates the risk of catastrophic forgetting.</p> <p>Disadvantages: Since only a part of the parameters of the model are updated, the model's ability to adapt to new tasks may be limited, which may be insufficient compared with comprehensive fine-tuning.</p>
Number of Trainable Layers	Specifies how many layers of the model are open to adjustments during the fine-tuning process.
Name Module Trainable	Name of trainable module when using freeze option: mlp or self_attention.
lora	<p>Description: LoRA modifies pre-trained models by introducing trainable low-rank matrices that adjust the model's existing weights instead of replacing them. These matrices are smaller and only modify the behavior of existing parameters of the model.</p> <p>Advantages: LoRA allows for efficient adaptation with less increase in trainable parameters, thus preserving the original advantages of the model while adapting to new tasks. It is more computationally efficient and reduces the risk of overfitting.</p> <p>Disadvantages: In some cases, LoRA may not offer the same flexibility as full fine-tuning.</p>
Lora Target	Specifies which parts of the model the LoRA adaptation targets. It refers to the modules (query, key, value, output, others.. projections) to apply the adapter to. The target modules will vary depending on each LLM model.
Lora Rank	<p>The dimension of the matrix decomposition used in LoRA.</p> <p>Its range follows the rule 2^n to optimize the learning principle of neural networks.</p>

	Range: [0, 2, 4, 8,16, 32, 64, 128, 256,...]
Lora Alpha	The scaling factor for the lora weights. Its range follows the rule 2^n to optimize the learning principle of neural networks. Range: [0, 2, 4, 8,16, 32, 64, 128, 256,...]
Lora dropout	The probability of applying dropout to the LoRA weights during training. Range: [0 ~ 1].
qlora	Quantized version of LoRA, likely involving fewer bits to represent the LoRA parameters, which could save memory and computation.
Quantization Bit	Defines the bit precision for model weights, typically to reduce model size or speed up inference. Convert the higher-precision floating-point numbers (32 bits, 16 bits) used to represent the model's parameters to lower-precision numbers (8 bits, 4 bits). Options: None, 4 bits, 8 bits.
Backbone Dtype	The datatype of the weights in the LLM backbone decides the precision of the model after fine-tuning. Options: fp16, bf16.
Learning rate	The learning rate is the speed at which the model updates its weights after processing each mini-batch of data. Range: [0 ~ 1].
LR Schedule Type	The strategy for adjusting the learning rate over between epochs or iterations as the training progresses. Options: cosine, linear(Only for LLM finetune)
Batch size	The number of training examples a mini-batch uses per single GPU during an iteration of the training model. Its range follows the rule 2^n to optimize the learning principle of neural networks. Range: [0, 2, 4, 8,16, 32, 64, 128, 256,...].
Epochs	The number of times the learning algorithm goes through the entire training dataset.
Save Checkpoint Strategy	The checkpoint saving strategy to adopt during training. Options: None, steps, epochs “None”: save no checkpoint until finishing training process “steps”: periodically save checkpoint at every n training steps “epochs”: periodically save checkpoint at every training epoch
Save Steps	Number of training steps before two checkpoint saves. Example: save_steps = 500 means periodically save checkpoint at every 500 training steps (500, 1000, 1500, ...) finishing training process
Max Length	The maximum length of the input sequences LLM used during model training. It decides the maximum length of the “input query” and “output answer” to inference the LLM. Normally 1000 tokens ~ 750 words by Open AI standard. Range: [512, 1024, 2048,...,128k] depends on the capability of pre-trained

	LLM.
GPUs	Number of GPUs used for training process. It automatically marks all the available GPUs in the current workstation PC. You can mark the GPUs you want.
Offloading Memory Strategy	The unique technique by Gigabyte to provide users multiple choices to offload model optimizer states, gradients, parameters and optionally activations to CPU or/and NVMe to optimize hardware capability and avoid out-of-memory issues.
Expand Config	The unique technique by Gigabyte to provide expert mode for professional LLM trainers who want to edit or add more training configs to fine-tune LLM in their way. Please refer to 3-9 Finetune Expand config syntax Example: --quantization_bit 8 \

2-3-2. LMM

Aside from the Dataset Settings, all other settings can be configured in the same way as mentioned in the section 2-3-1. LLM.

Dataset settings	All settings datasets used for finetuning a LMM model.
JSON Datasets	A collection of datasets stored in JSON format, typically containing annotations or metadata for text, images, audio, or video data to be used during model fine-tuning.

2-4. Inference

Inference Type	Select the type of inference: (1)Text to Text, (2)Text to Image, (3)Text to Video, (4)Image Text to Text.
Backbone Model	The model that you want to validate with GGUF or safetensors format.
System prompt (optional)	A set of instructions and guidelines is provided to the LLM to control its behavior, tone, and the overall context of its responses.
Maximum tokens	The maximum tokens for sentence length of query and answer.
Temperature	Controls the randomness and creativity of the model's output. It adjusts the probability distribution of the words the model considers when generating a response. Ranging from 0 to 1, higher values yield more diverse, creative responses.
Top p	Controls the diversity of the model's output using nucleus sampling. It limits the model's choices to the smallest set of words whose cumulative probability equals P . Ranging from 0 to 1, higher values yield more diverse, creative responses.

GPU Offload (GGUF)	Set up how many model layers mount to VRAM run by GPU .
CPU Threads (GGUF)	Set up use how many CPU threads to run the model layers mount to DRAM
Fine-tuning type (safetensors)	Describes the fine-tuning method, including: full, freeze, (q)lora.
Adaptar model (safetensors)	The adapter model that you finetuned via the Lora method.
Width	The width of the output generated by the text-to-image or text-to-video model.
Height	The height of the output generated by the text-to-image or text-to-video model.
FPS	The frames per second of the video generated by the text-to-video model.
Length of The Video (seconds)	The duration of the generated video in seconds.
Save path	The directory where the image or video will be saved.

2-5. RAG

Document Folder	The folder contains files in different formats, such as .txt, .wav, and .mp4.
Model	Select the LLM model for Q&A with RAG.
Text Embedding Model	The text embedding model, e.g., all-MiniLM-L6-v2.
Audio Embedding Model	The audio embedding model, e.g., larger_clap_general.
Image Embedding Model	The image embedding model, e.g., CLIP-ViT-B-32-laion2B-s34B-b79K.
Video Embedding Model	The video embedding model, e.g, CLIP-ViT-B-32-laion2B-s34B-b79K.
System prompt (optional)	A set of instructions and guidelines is provided to the LLM to control its behavior, tone, and the overall context of its responses.
Maximum New Tokens	The upper limit on the number of tokens the model can generate in a single response.

2-6. Model Converter

Model Folder	The directory that contains the original model files you want to convert. Supported formats include LLaVA, Hugging Face (.safetensors / .bin), and GGML (.ggmlv3).
Convert Format	The target format you want to convert the model into. Currently, only the GGUF format is supported.
Quantization Type	The numerical precision type to be applied to the output model. Available options include: f32 (32-bit float), f16 (16-bit float), bf16 (bfloat16), q8_0 , tq1_0 , tq2_0 (various quantized formats), and auto (automatically select a suitable quantization).

2-7. Machine Learning

(Project) Folder	The directory where the machine learning project is located. All ML projects will be stored in this folder.
Task Type	The type of machine learning task is the project. Currently supported types include: image classification , object detection , image segmentation , and OCR (Optical Character Recognition).
(Project) Name	The name of the project. The name can be modified later, but cannot be duplicated.
Python Version	The Python version used in the project. Currently, Python 3.10 is supported.
Go to Annotate	Annotation tools suitable for the selected task type will be provided to assist users in labeling their own datasets.

2-8. Settings

License	End user license agreement (EULA) in English, Simplified Chinese and Traditional Chinese version.
Version	The current version of software.
Cluster setting	This encompasses all the settings of each workstation (node) that users set up for multi-node training, including device name, IP, GPU count, SSD count, and RAM count.
SSD Settings	When training large AI models that exceed the capacity of VRAM and DRAM, users can offload memory to a high-performance NVMe SSD. This SSD offloading feature requires manual mounting.

3. Operation tutorial:

This is a step-by-step instruction for using the AI TOP Utility's Experiment & Dashboard to fine-tune a pretrained LLM with a specified dataset.

3-1. How to generate a well organized dataset from a rough dataset?

The dataset feature helps users automatically create a well-organized dataset that can be directly fine-tuned using an LLM backbone from a raw dataset. This feature recommends an LLM to generate a Q&A dataset based on specified factors such as dataset format, field, language, and the hardware capabilities of the user's PC. The dataset will be saved into dataset_info.json after the generation process is completed, allowing the user to quickly navigate to the Experiment tab and start fine-tuning the dataset using the LLM backbone.

- (1) Set dataset folder directory to be `"/home/{user}/data"` (the one created right after installing)
- (2) Click **"Create Dataset"** and **"Upload File"** to select the document files you want to use to generate a well-organized dataset. We support the following formats: TXT, DOC, JSON, CSV, XLSX, and PDF.

The image displays two screenshots of the 'Create Dataset' interface in a dark-themed application.

Left Screenshot: The 'Create Dataset' window is open. Under 'Upload File', a message states 'A file has been selected' and lists file paths: `["/home/z890/data/Huawei Pura 80 Ultra.txt", "/home/z890/data/iPhone 17.txt", "/home/z890/data/Samsung Galaxy Z Fold 7.txt"]`. The 'Recommended LLM model' section lists several options, with 'meta-llama/Llama-3.2-3B-Instruct' selected. The 'Select LLM model' section shows 'A folder has been selected' and the path `/home/z890/model/Llama-3.2-3B-Instruct`. The 'Dataset Name' field contains 'Phone_dataset'. The 'No Limit Q&A' section has 'True' selected. The 'Number of Q&A' field is set to '30'. A 'Submit' button is at the bottom right.

Right Screenshot: The same 'Create Dataset' window is shown, but a modal dialog box is overlaid in the center. The dialog has a progress bar at 36.7% and the text 'Creating dataset, please wait .'. A 'Stop' button is at the bottom of the dialog. The background window is dimmed.

(3) Recommend LLM model

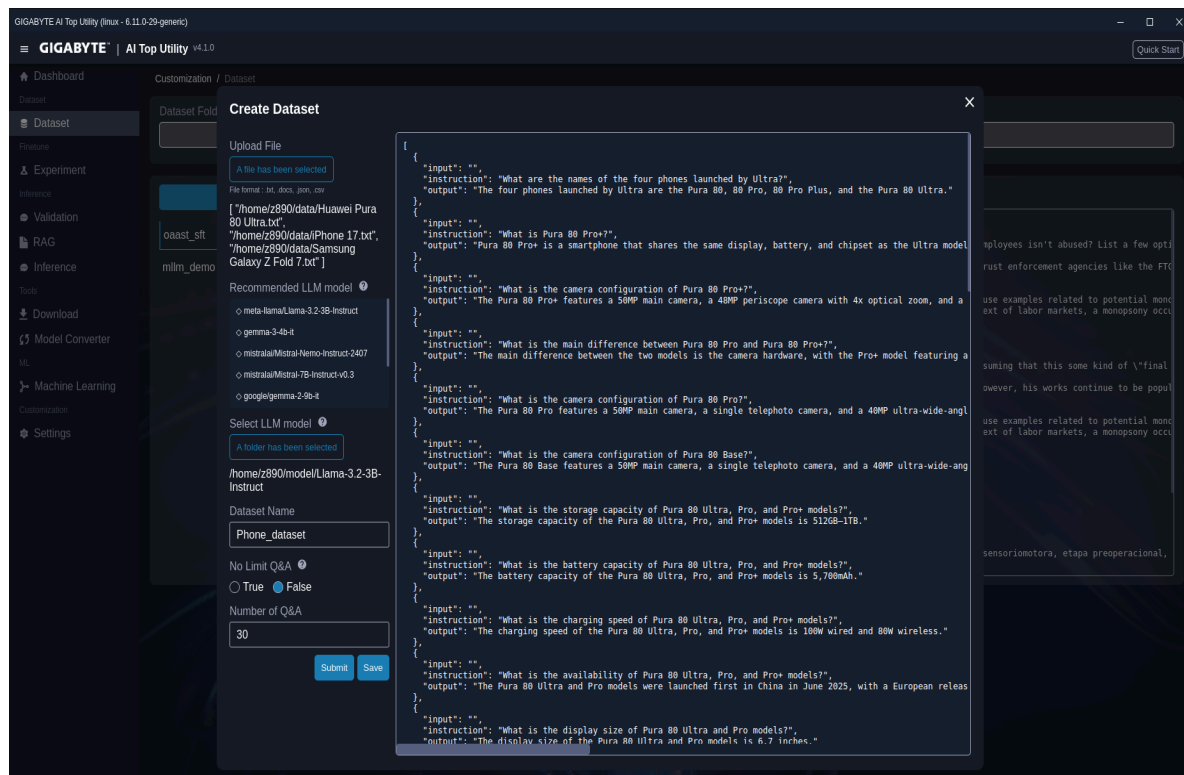
This feature will automatically recommend an LLM to generate a Q&A dataset based on specified factors such as dataset format, field, language, and the hardware capabilities of the user's PC.

(4) Select LLM model

Based on the recommended models, the user can download the LLM model from HuggingFace and store it in the "home/{user}/model" directory. Then, select it using the "**Please choose folder**" button.

(5) Next, input a name for your new dataset (e.g., "Phone_dataset") and specify the desired number of Q&A pairs. If you set "**No Limit Q&A**" to True, the system will generate up to 10,000 Q&A pairs.

(6) After clicking the **Submit** button, a well-organized dataset will be generated over a period of time.



(7) Click the "**Save**" button once the generation process is complete. The dataset (e.g., "Phone_dataset") will be automatically enrolled in "/home/{user}/data/dataset_info.json", allowing the user to quickly navigate to the Experiment tab to begin fine-tuning the dataset using an LLM backbone.

```

dataset_info.json
~/data

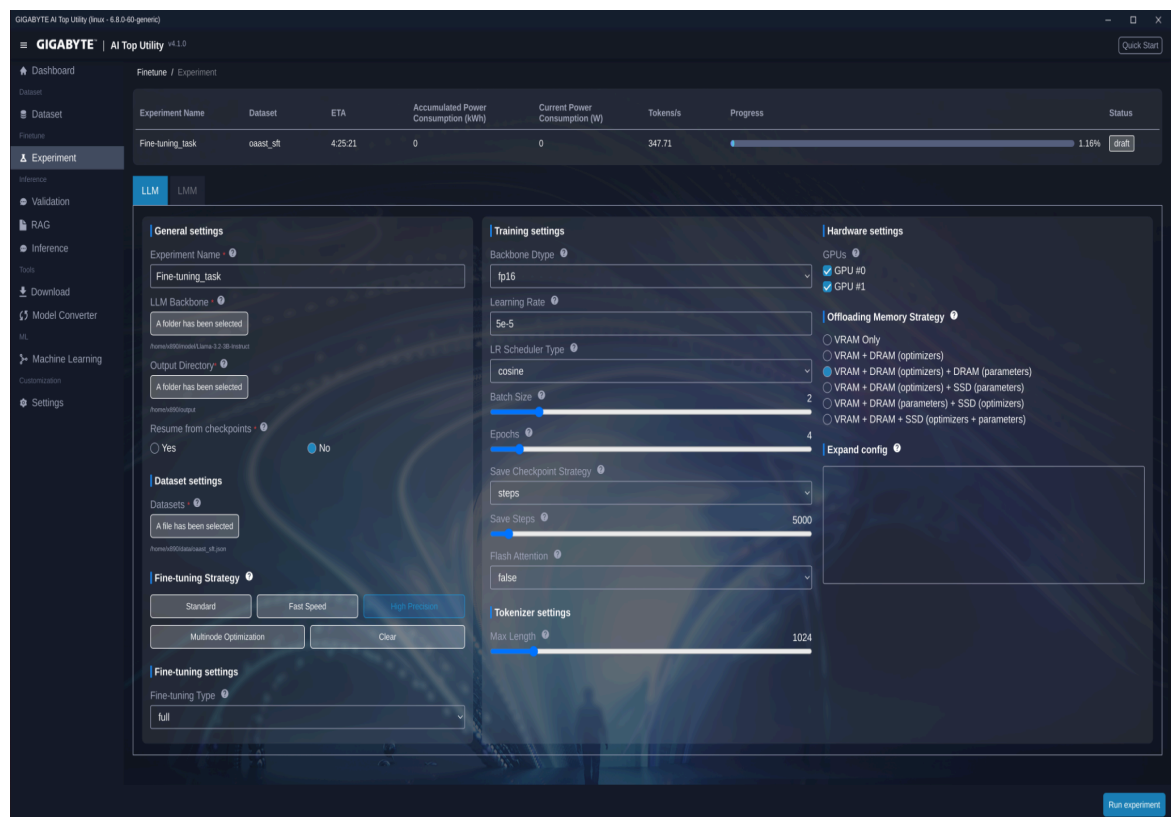
{
  "Phone_dataset": {
    "file_name": "Phone_dataset.json",
    "file_sha1": "0820be60225ff09abb23815d5b61617549eb8760",
    "columns": {
      "prompt": "instruction",
      "query": "input",
      "response": "output"
    }
  },
  "oaast_sft": {
    "file_name": "oaast_sft.json",
    "file_sha1": "7baf5d43e67a91f9bbdf4e400dbe033b87e9757e",
    "columns": {
      "prompt": "instruction",
      "query": "input",
      "response": "output",
      "history": "history"
    }
  }
},

```

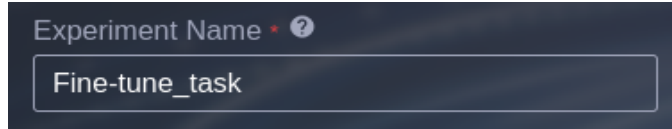
3-2. How to finetune a pre-trained LLM model based on a specified dataset?

Fine-tuning a pre-trained LLM model with a custom dataset

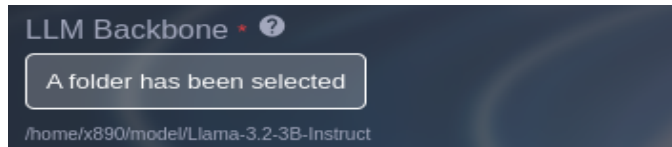
- (1) Click the **"Experiment"** tab to begin setting up a new fine-tuning experiment.



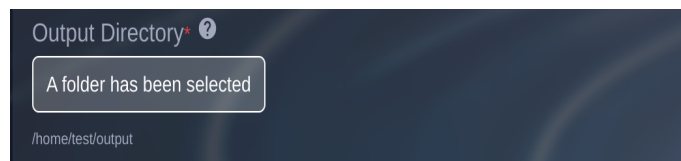
- (2) In **"General Setting"**, choose the **"Experiment name"** and enter any name you prefer.



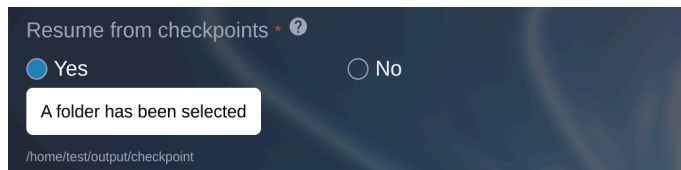
- (3) Select the "**LLM Backbone**". The default is Llama-3.2-3B-Instruct, but you can choose the LLM you target for your fine-tuning project. (Please refer to section 4 to choose your target LLM Backbone that AI TOP Utility supports default configs for easy settings.)



- (4) Select the "**Output directory**" where the fine-tuned model will be saved.



- (5) If you want to resume from a checkpoint, select "**Yes**" in "**Resume from checkpoint**" and search in the "**Select folder**" which stores your previous checkpoint. (If there is no checkpoint, please skip this step.)



- (6) In "**Dataset settings**", click "**Select File**" to choose the dataset for your fine-tuning process. The LLM backbone will learn from this dataset.



- (7) The dataset has to be enrolled in "**/home/{user}/data/dataset_info.json**". If your dataset is created by our "**Create Dataset**" function, you can now move on to (8). Otherwise, please

modify the “dataset_info.json” file and provide your dataset definition in this file.

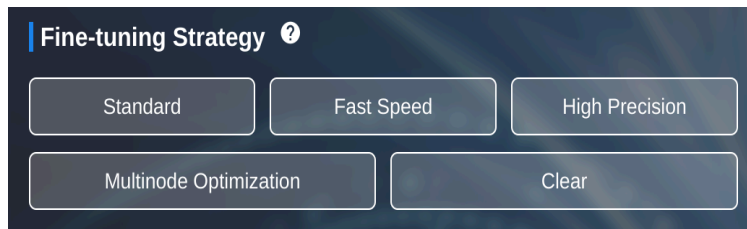
```
{
  "Phone_dataset": {
    "file_name": "Phone_dataset.json",
    "file_sha1": "0820be60225ff09abb23815d5b61617549eb8760",
    "columns": {
      "prompt": "instruction",
      "query": "input",
      "response": "output"
    }
  },
  "oaast_sft": {
    "file_name": "oaast_sft.json",
    "file_sha1": "7baf5d43e67a91f9bbdf4e400dbe033b87e9757e",
    "columns": {
      "prompt": "instruction",
      "query": "input",
      "response": "output",
      "history": "history"
    }
  }
},
```

```
``json
"dataset_name": {
  "file_name": "the name of the dataset file in this directory. (required if
above are not specified)",
  "file_sha1": "the SHA-1 hash value of the dataset file. (optional, does not affect training)",
  "subset": "the name of the subset. (optional, default: None)",
  "folder": "the name of the folder of the dataset repository on the Hugging Face hub. (optional,
default: None)",
  "ranking": "whether the dataset is a preference dataset or not. (default: false)",
  "formatting": "the format of the dataset. (optional, default: alpaca, can be chosen from
{alpaca, share
gpt})",
  "columns (optional)": {
    "prompt": "the column name in the dataset containing the prompts. (default: instruction)",
    "query": "the column name in the dataset containing the queries. (default: input)",
    "response": "the column name in the dataset containing the responses. (default: output)",
    "history": "the column name in the dataset containing the histories. (default: None)",
    "messages": "the column name in the dataset containing the messages. (default:
conversations)",
    "system": "the column name in the dataset containing the system prompts. (default: None)",
```

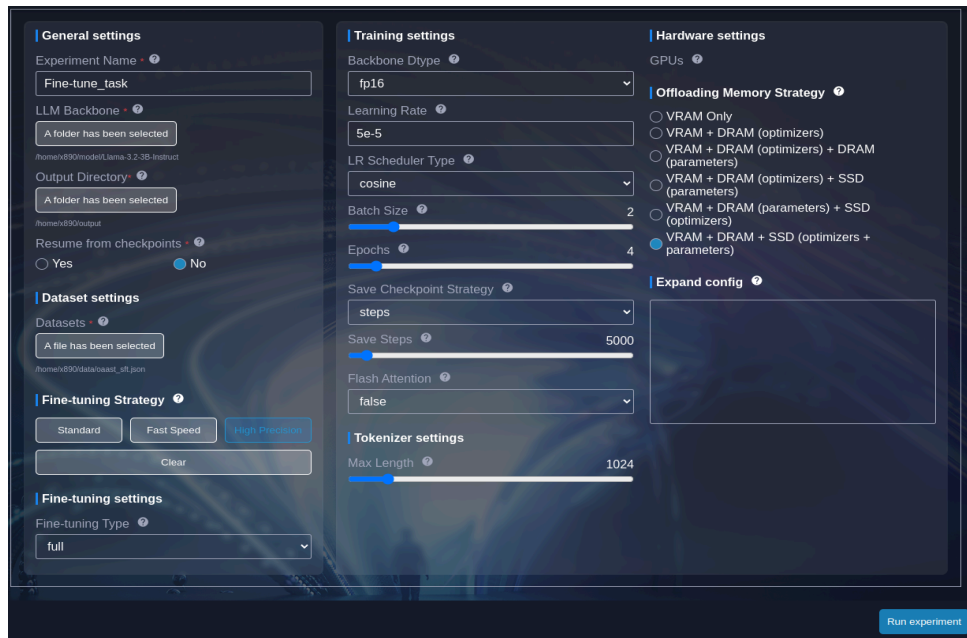


```
"tools": "the column name in the dataset containing the tool description. (default: None)"
},
"tags (optional, used for the sharegpt format)": {
  "role_tag": "the key in the message represents the identity. (default: from)",
```

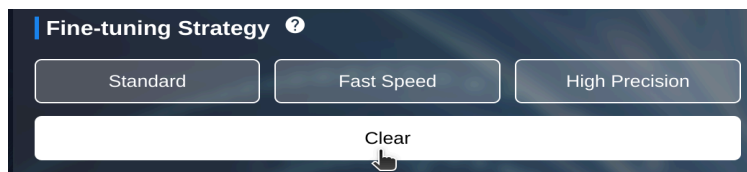
- (8) AI TOP Utility provides easy-to-operate modes for non-AI-based users to select their fine-tuning strategies: Standard, Fast Speed, and High Precision. Select one that matches your preferences.



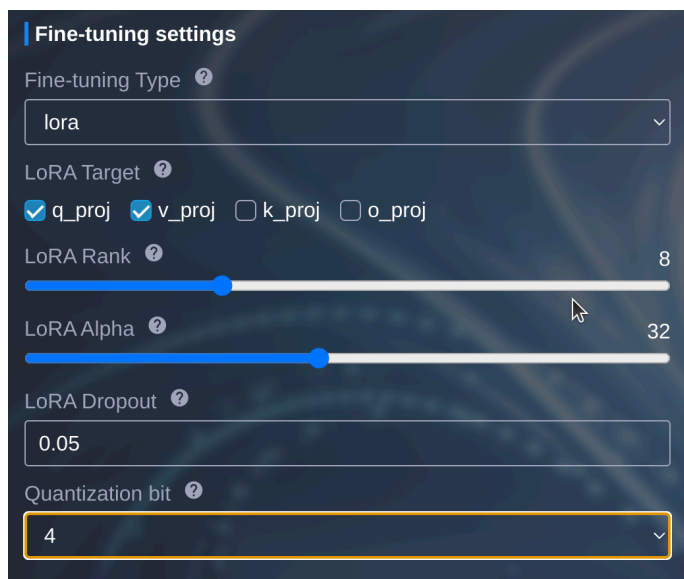
- **Standard:** offers medium speed and medium precision.
 - **Fast Speed:** provides fast speed with acceptable precision.
 - **High Precision:** ensures high precision.
 - **Multinode Optimization:** enables advanced distributed training for large-scale LLM models when utilizing two or more GPUs.
- (9) After selecting one of these three strategies, all remaining settings in the experiment tab will be automatically configured. Simply click "**Run Experiment**" at the bottom-right corner to start the fine-tuning process.



- (10) If you are a professional LLM trainer or have basic knowledge in this field, you can choose the **"Clear"** option to customize all fine-tuning settings yourself.



- (11) In **"Fine-tuning settings"**, select a fine-tuning method: Full, Freeze, or (Q)LoRA. Each method affects training memory requirements and the quality of the LLM after fine-tuning. Hover over the question marks next to each setting title for more details.



- (12) In **"Training settings"**, set up the hyperparameters for the training process.

Training settings

Backbone Dtype [?]
fp16

Learning Rate [?]
0.00005

LR Scheduler Type [?]
cosine

Batch Size [?]
2

Epochs [?]
1

Save Checkpoint Strategy [?]
steps

Save Steps [?]
1500

Flash Attention [?]
true

- **Backbone Type:** Choose between fp16 and bf16.
- **Learning Rate:** Set the speed at which the model updates its weights after processing each mini-batch of data.
- **LR Schedule Type:** Choose the type of learning rate schedule to adjust the learning rate between epochs or iterations as training progresses.
- **Batch Size, Number of Epochs, and Save Checkpoint Strategy:** Configure these parameters for your training process.
- **Save Steps:** Specify the number of updates steps between two checkpoint saves.

(13) In “**Tokenizer Settings**”, choose the "Max Length" of the input sequence the LLM uses during model training. This sets the maximum length in words for both the input query and the LLM's response.

Tokenizer settings

Max Length [?]
2048

(14) In “**Hardware Settings**”, select the GPU(s) you want to use for the training process.

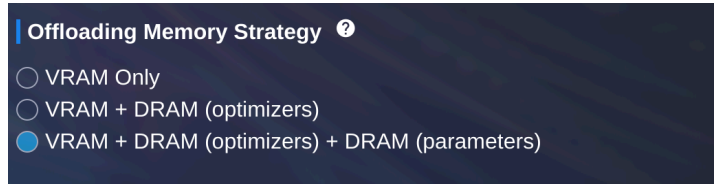
Hardware settings

GPUs [?]

☒ GPU #0

☒ GPU #1

- (15) In **"Offloading Memory Strategy"**, this advanced feature allows you to fine-tune small LLM models like GPT-2 using only VRAM, or fine-tune medium/large LLM models like 13B, 30B, and 70B by offloading memory to system DRAM and SSD, thus breaking through hardware memory limitations.



Offloading Memory Strategy ?

- ☐ VRAM Only
- ☐ VRAM + DRAM (optimizers)
- ☒ VRAM + DRAM (optimizers) + DRAM (parameters)

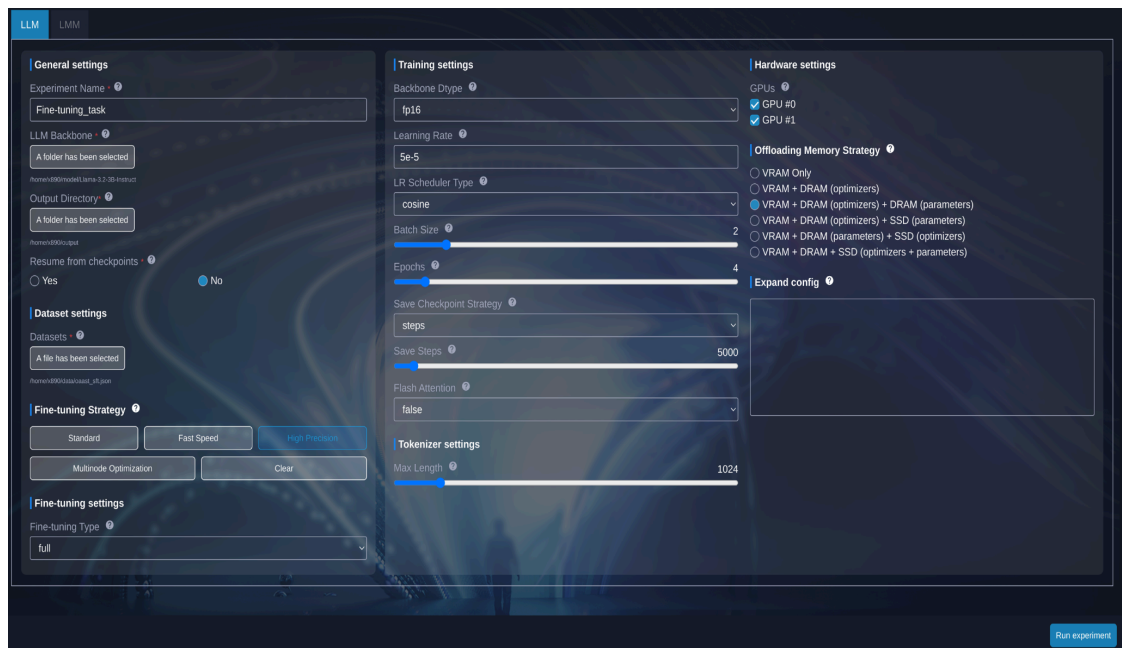
- (16) In **"Expand config"**, you may adjust current configurations or add more configurations to the fine-tuning script, e.g., `--quantization_bit 8 \`. (Please refer to section 3-9 to check all the possible config syntaxes for this "expand config" settings.)



Expand config ?

```
--quantization_bit 8 \
```

- (17) After completing all settings, click the **"Run Experiment"** button at the bottom right corner to start fine-tuning the LLM.



LLM LLM

General settings

Experiment Name ?

Fine-tuning_task

LLM Backbone ?

A folder has been selected

Output Directory ?

A folder has been selected

Resume from checkpoints ?

☐ Yes ☒ No

Dataset settings

Datasets ?

A file has been selected

Fine-tuning Strategy ?

Standard Fast Speed High Precision

Multinode Optimization Clear

Fine-tuning settings

Fine-tuning Type ?

full

Training settings

Backbone Dtype ?

fp16

Learning Rate ?

5e-5

LR Scheduler Type ?

cosine

Batch Size ?

2

Epochs ?

4

Save Checkpoint Strategy ?

steps

Save Steps ?

5000

Flash Attention ?

false

Tokenizer settings

Max Length ?

1024

Hardware settings

GPUs ?

☒ GPU #0 ☒ GPU #1

Offloading Memory Strategy ?

- ☐ VRAM Only
- ☐ VRAM + DRAM (optimizers)
- ☒ VRAM + DRAM (optimizers) + DRAM (parameters)
- ☐ VRAM + DRAM (optimizers) + SSD (parameters)
- ☐ VRAM + DRAM (parameters) + SSD (optimizers)
- ☐ VRAM + DRAM + SSD (optimizers + parameters)

Expand config ?

Run experiment

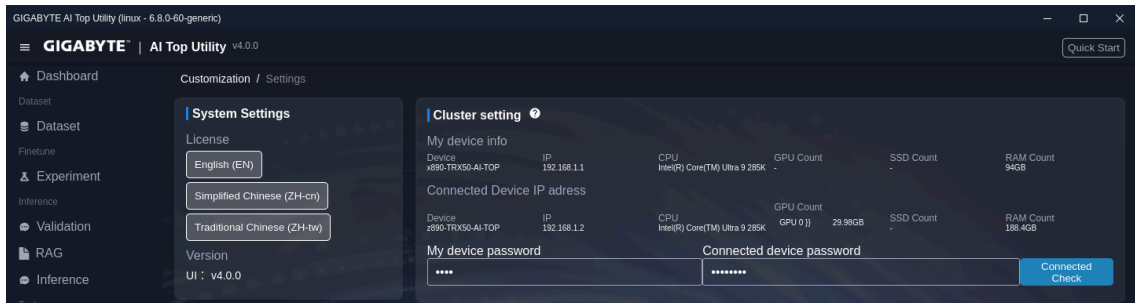
3-3. How to fine-tune a pre-trained LLM model using multi-node cluster?

- (1) Click the **"Experiment"** tab to begin setting up a new fine-tuning experiment.

Aside from the Cluster Settings, all other settings can be configured in the same way as mentioned in Section 3-2.

- (2) Set up **"Cluster Settings"**

Navigate to the "Settings" tab in the menu bar, then click the "Connect Check" button in the Cluster setting. All the information for each node will be displayed below if the connection is successfully established.



- (3) Navigate back to the **"Experiment"** tab and select **Cluster (connection device: 1)** in Cluster Settings, and click **Run experiment**.



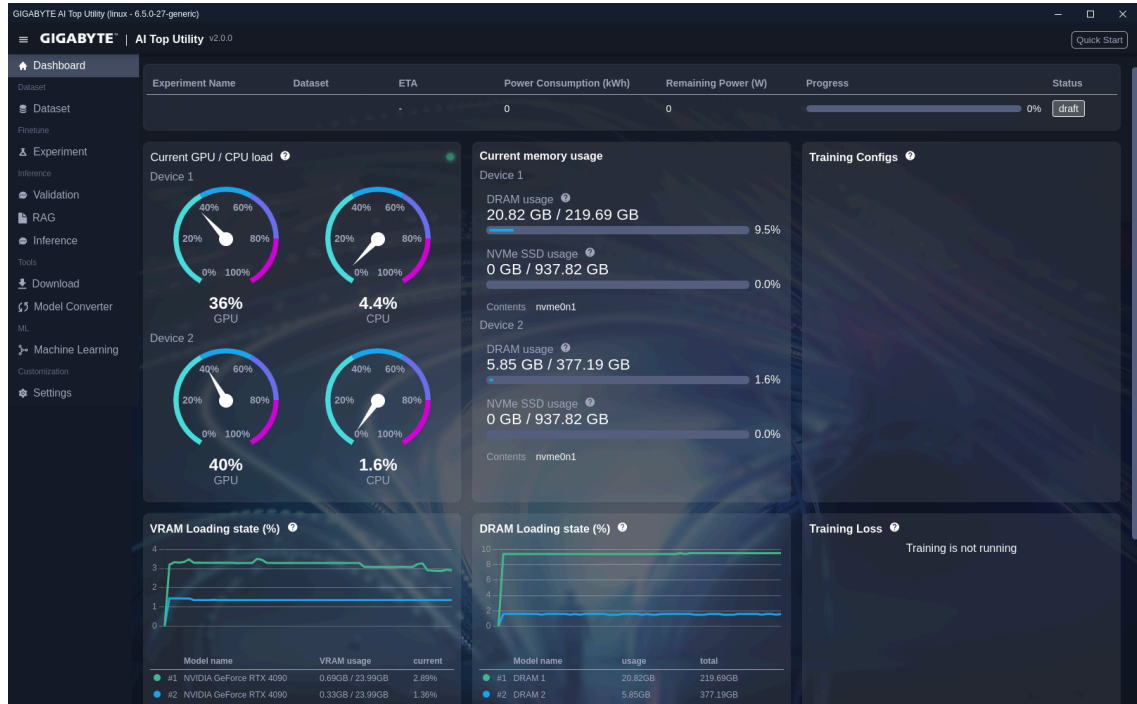
- (4) Go to the **"Dashboard"** tab to monitor the progress of your fine-tuning job with multi-node settings.

The Dashboard enables real-time monitoring of hardware status across multiple devices, including Device 1 (Master Server) and Device 2 (Client Node). This functionality is vital for efficiently managing and optimizing parallel training across multiple nodes for Large Language

Models (LLMs). By ensuring a smooth and synchronized workflow, it helps maximize the performance and effectiveness of the parallel LLM training setup.

Device 1 (Master Server): Serves as the central hub for monitoring all connected devices.

Device 2 (Client Node): Operates as a subordinate node participating in parallel tasks alongside the master server.



3-4. How to monitor the fine-tuning process in the Dashboard tab?

- (1) Click the "**Dashboard**" tab to monitor the fine-tuning process. The dashboard allows you to track experiment progress, configurations, quality, and logs, as well as observe the real-time status of all hardware components.



- (2) The top bar displays the current experiment name, estimated time of accomplishment (ETA), accumulated power consumption, current power consumption, tokens/s, progress percentage, and the overall status of your experiment job.

Experiment Name	Dataset	ETA	Accumulated Power Consumption (kWh)	Current Power Consumption (W)	Tokens/s	Progress	Status
Fine-tuning_task	oaast_sft	6 days, 15:50:37	0	0	118.86	0.08%	Running

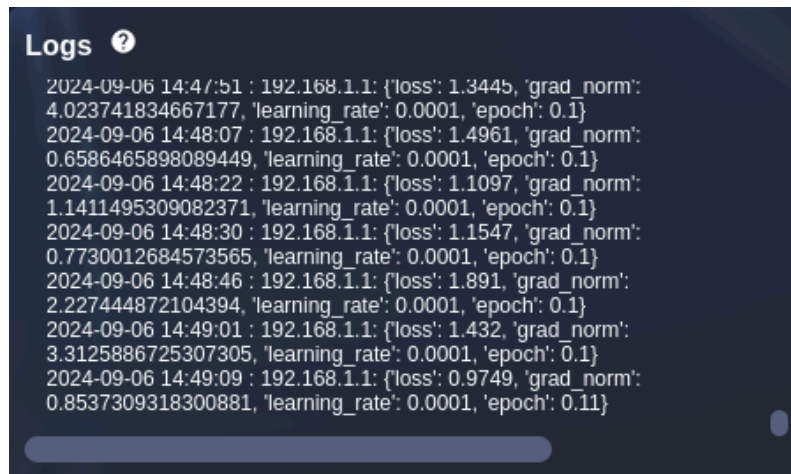
- (3) **Training Configs:** Double-check all configuration settings for the current experiment.

Training Configs ?	
Num Gpus	1
Num Nodes	1
Stage	sft
Model Name Or Path	"/home/z890/model/Llama-3.2-3B-Inst ruct"
Dataset Dir	/home/z890/data
Dataset	"oaast_sft"
Max Length	1024
Finetuning Type	full
Output Dir	"/home/z890/output/2025-06-11-16-1 1-30/output/"
Per Device Train Batch Size	?

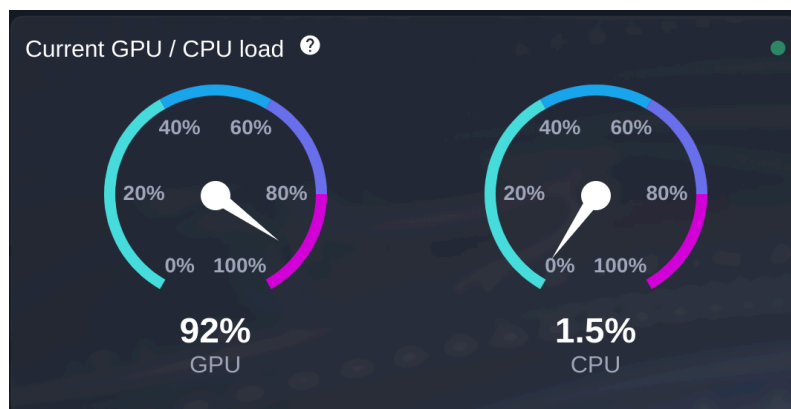
- (4) **Training Loss:** Monitor training loss of every step to ensure your model training is converging.



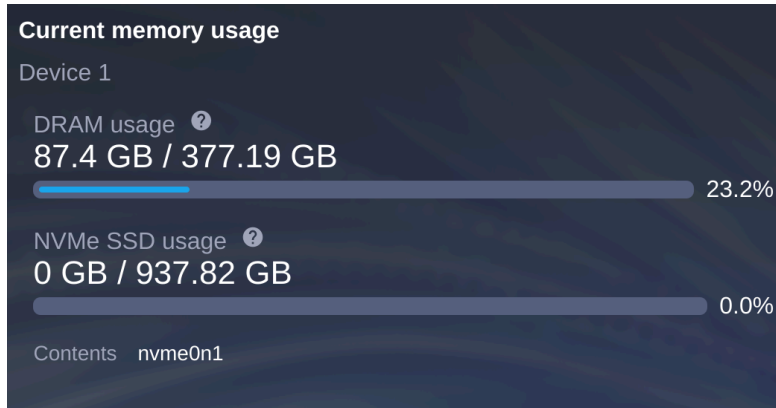
- (5) **Logs:** Track events and error messages during the training process.



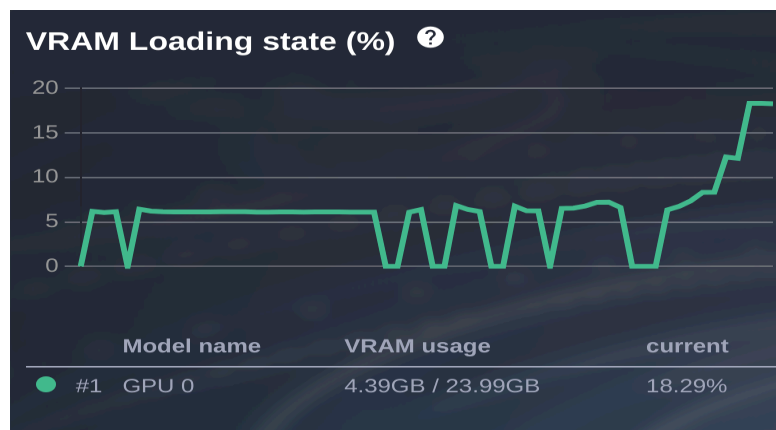
- (6) **Current GPU/CPU Load:** View the efficiency (%) of GPUs and CPU in real-time.



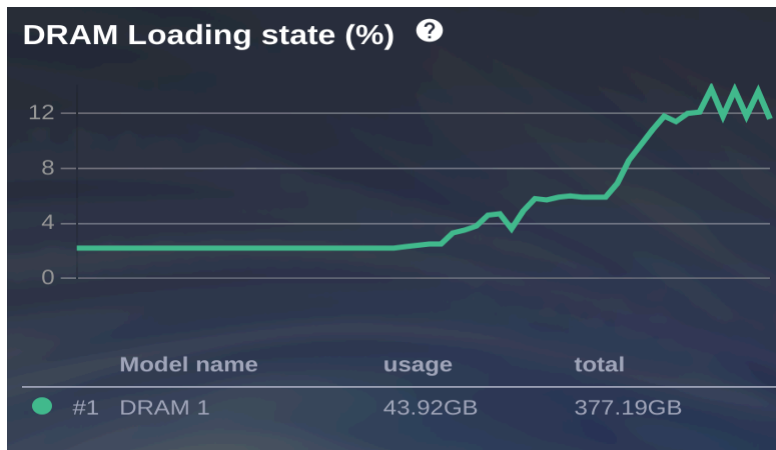
- (7) **Current Memory Usage:** Check the loading state of system DRAM and SSD.



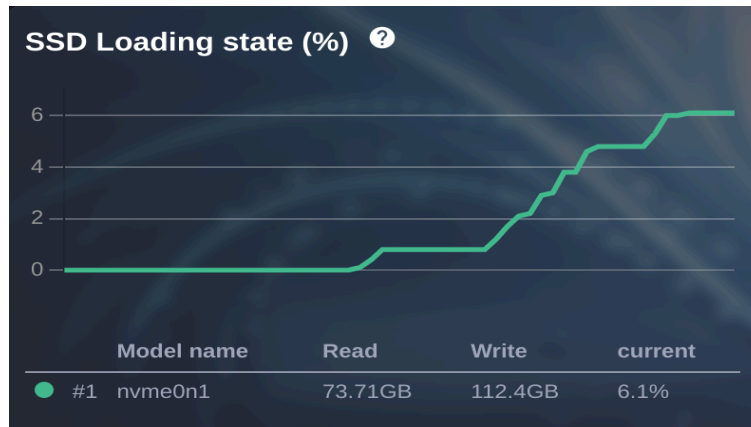
(8) **VRAM Loading State (%)**: Displays the loading state of each GPU during training.



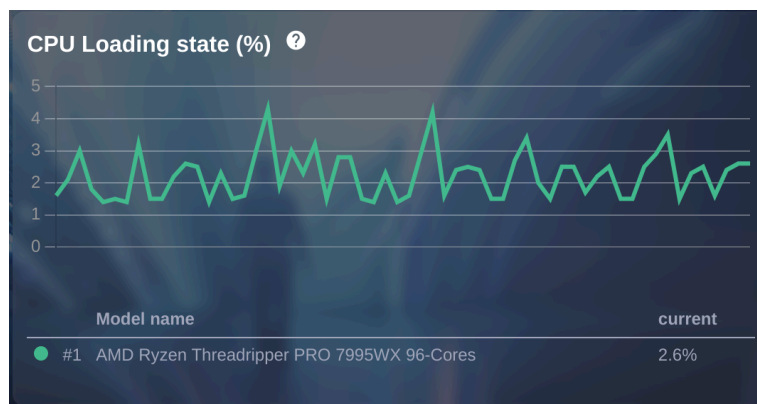
(9) **DRAM Loading State (%)**: Shows the loading state of the system DRAM if memory is offloaded to it.



(10) **SSD Loading State (%)**: Indicates the loading state of the NVMe SSD if memory is offloaded to it.



(11) **CPU Loading State (%)**: Shows the average loading state of all CPU cores during training.

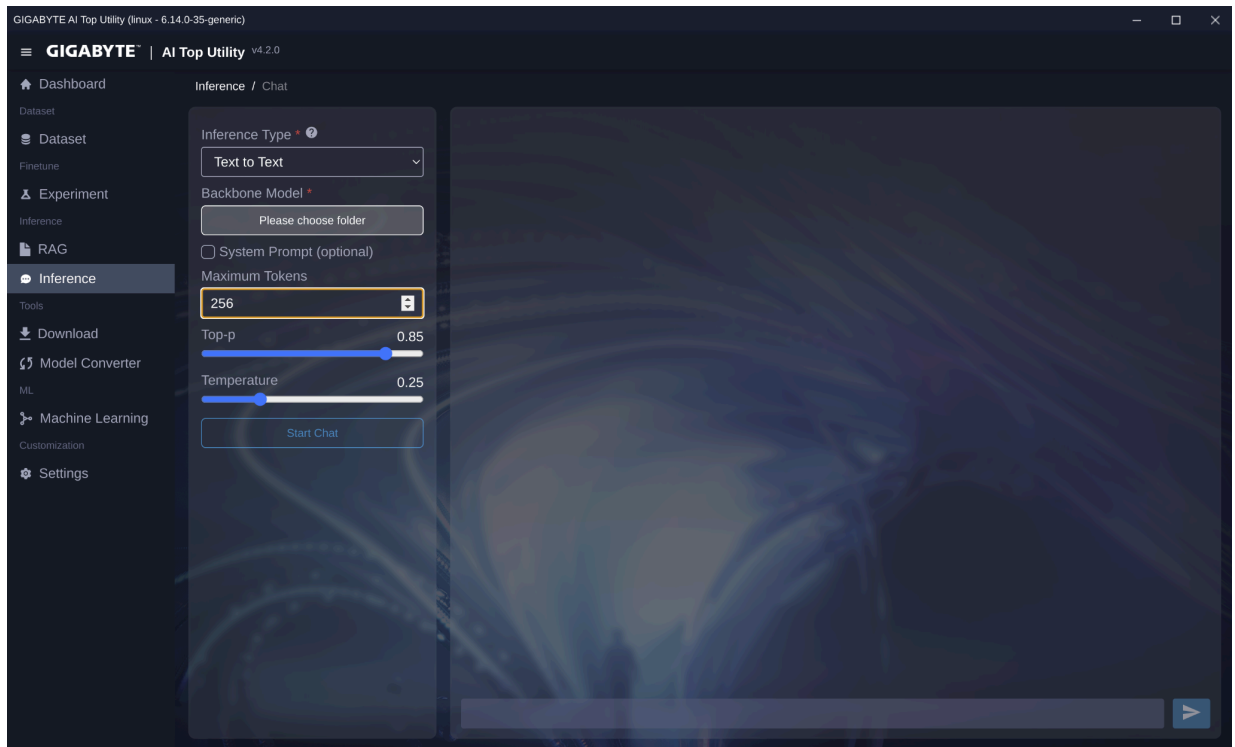


3-5. Inference

Our built-in inference tool lets you run models in either **Safetensors** or **GGUF** format. You can use models which are **downloaded** from the **Download** tab, **fine-tuned** in the **Experiment** tab or **converted** from Safetensors in the **"Model Converter"** tab. We support various inference types, including:

1. Text to Text
2. Text to Image
3. Text to Video
4. Image Text to Text.

(1) Click the **"Inference"** tab



(2) Choose **"Inference Type"**.

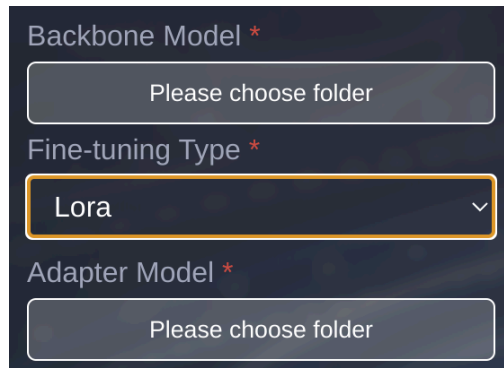
❖ Suppose we choose **Text to Text**.

(3) Under the **"Backbone Model"**, please select one of the text-to-text models, e.g. **"Llama-3.1-8B-Instruct"**

(4) Configure model options

a. If you're using the **safetensors** format model.

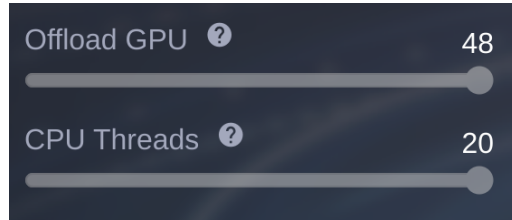
- Pick the fine-tuning type: **Full**, **Freeze**, or **LoRA**. If you have not fine-tuned it, choose **Full**.
- If you choose **LoRA**, select the corresponding adapter in **Adapter Model**.



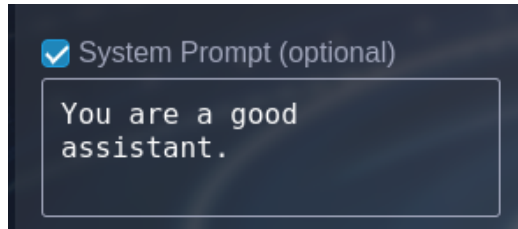
b. If you're using the **GGUF** format model.

- In **Offload GPU**, specify how many layers to keep in **VRAM (GPU)**.

- Set the number of **CPU threads** for the layers that stay in **DRAM**.



- (5) Tick **System Prompt** and enter a prompt if you need one (Optional).



- (6) Adjust **Maximum Tokens**, **Temperature**, and **Top-p** settings to control generation.

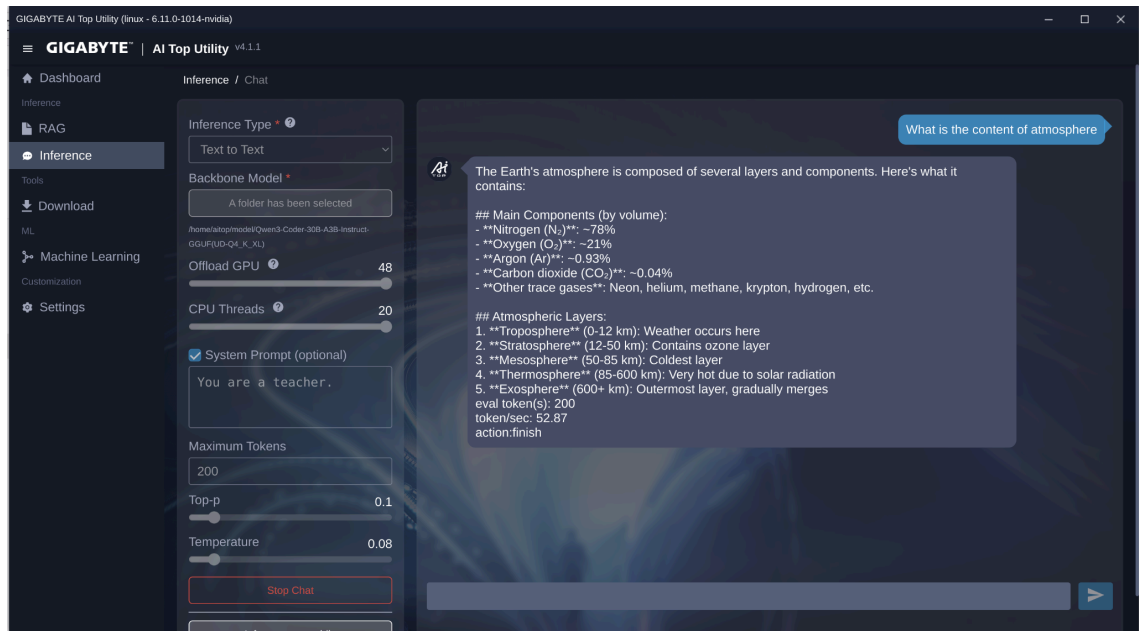


Maximum tokens: This determines the maximum number of tokens (words or word parts) that the model can generate in its output. A higher limit allows for longer responses, while a lower limit restricts the length.

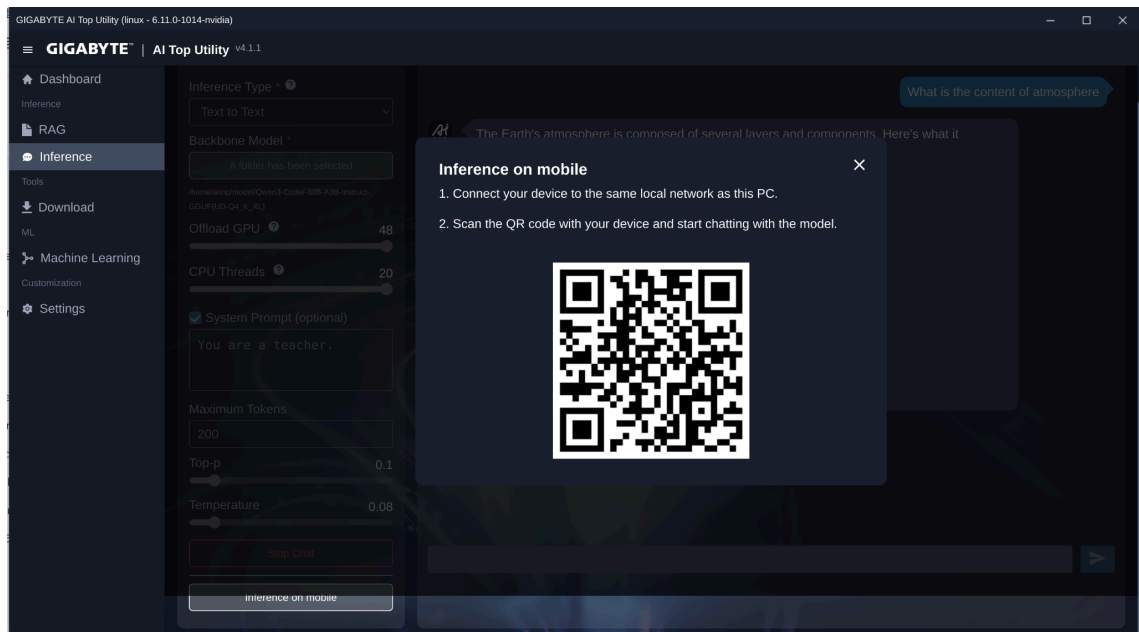
Top-p (nucleus sampling): This controls the probability mass from which the model selects its next token. A value of 1 means all possible tokens are considered, while a lower value (e.g., 0.9) narrows the choices to tokens that collectively account for 90% of the probability, making the output more focused.

Temperature: This controls the randomness of the model's output. A higher temperature (e.g., 1.0) makes the output more diverse and creative, while a lower value (e.g., 0.2) makes the output more deterministic and focused.

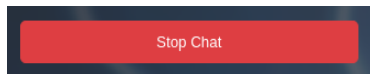
- (7) Click "Start Chat" to load the model. Then you can chat with the model.



- (8) To run inference on another device, click **"Inference on Mobile"** and make sure the device is on the **same local network** as your PC.



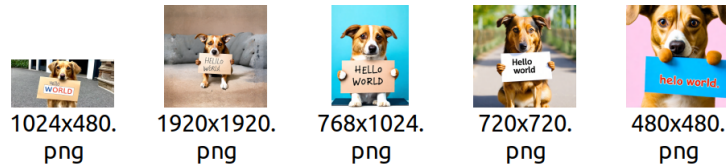
- (9) Click the "Stop Chat" button if you want to stop the inference.



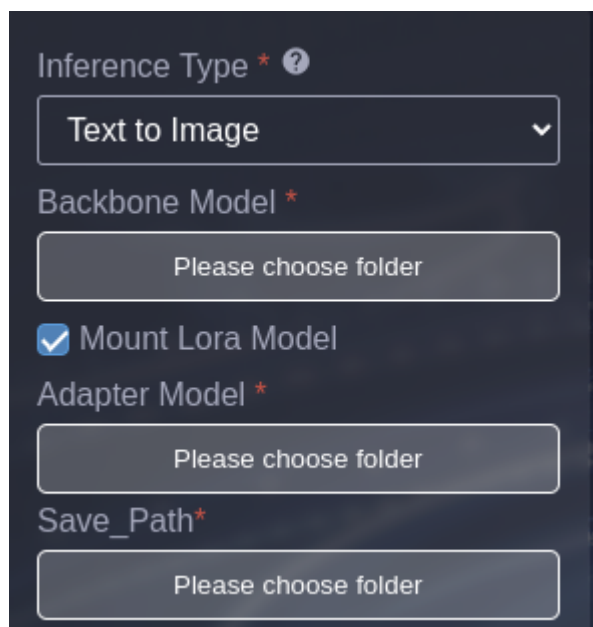
❖ Suppose we choose **Text to Image**.

- (3) Under the **"Backbone Model"**, please select one of the text-to-image models, e.g., **"stable-diffusion-3-medium-diffusers"** (requires at least 23GB of VRAM).

- (4) You can set the width and height attributes of the image and input any content for the image you wish to generate. However, the model may impose certain restrictions. Please refer to the model's Hugging Face page for more details. Usually, the height and width must be multiples of 16. You may try these sizes of image to begin with: 1024x480, 1920x1920, 768x1024, 720x720, 480x480.

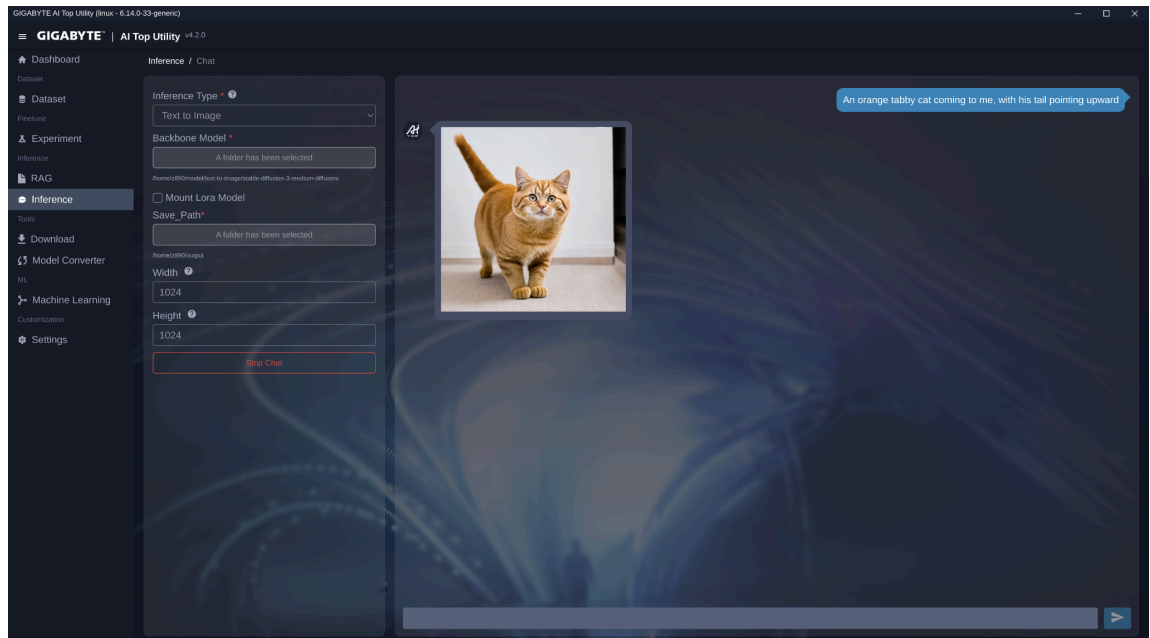


* To mount a LoRA model, click “Mount LoRA Model” and select the desired LoRA file.

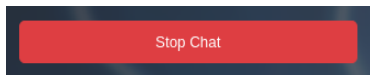
A screenshot of a dark-themed configuration window. It contains several sections: 'Inference Type' with a dropdown menu set to 'Text to Image'; 'Backbone Model' with a button labeled 'Please choose folder'; 'Mount Lora Model' with a checked checkbox; 'Adapter Model' with a button labeled 'Please choose folder'; and 'Save_Path' with a button labeled 'Please choose folder'.

Make sure the LoRA model you select is compatible with your chosen backbone model.

- (5) Click "Start Chat" to load the model. Then, you can input any prompt for the image you wish to generate. Refer to the model's page for best prompting practice.

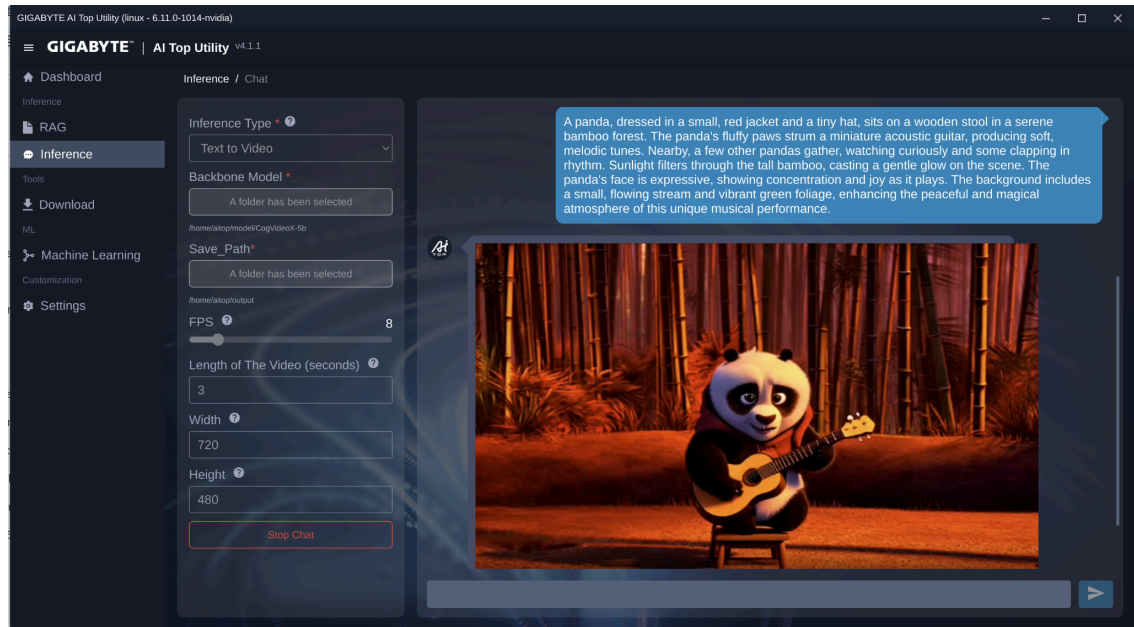


- (6) Click the "Stop Chat" button if you want to stop the inference.

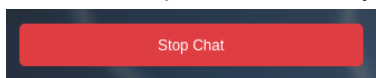


❖ Suppose we choose **Text-to-video**.

- (3) Under the "**Backbone Model**", please select one of the text-to-video models, e.g., "**CogVideoX-5b**" (requires at least 32GB of VRAM).
- (4) You can set the width and height attributes of the video. However, the model may impose certain restrictions. For example, CogVideoX-5b is limited to a height of 480 and a width of 720. Please refer to the model's Hugging Face page for more details.
- (5) You can set the attribute of FPS and the Length of The Video (seconds). Again, the model may impose certain restrictions. Please refer to the model's Hugging Face page for more details.
- (6) Click "Start Chat" to load the model. Then, you can input any prompt for the video you wish to generate. Refer to the model's page for best prompting practice. If the FPS attribute is set to 8 and the video length is 3 seconds, it will take approximately 2 to 3 minutes to process.



- (7) Click the "Stop Chat" button if you want to stop the inference.

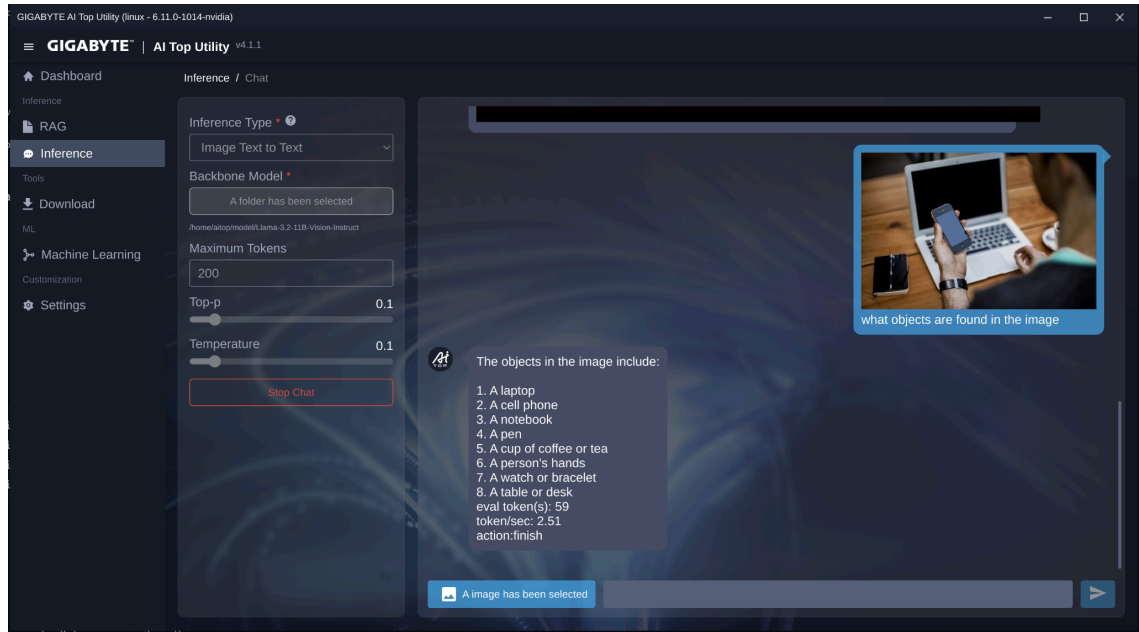


❖ Suppose we choose **Image Text to Text**.

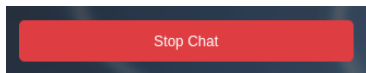
- (3) Under the "**Backbone Model**", please choose one of the image-text-to-text models, such as "**Llama-3.2-11B-Vision-Instruct**" (requires at least 24GB of VRAM).
- (4) Adjust **Maximum Tokens**, **Temperature**, and **Top-p** settings to control generation.



- (5) Click "Start Chat" to load the model. Then, you can choose the image you want the LLM to analyze and input your question.



(6) Click the "Stop Chat" button if you want to stop the inference.



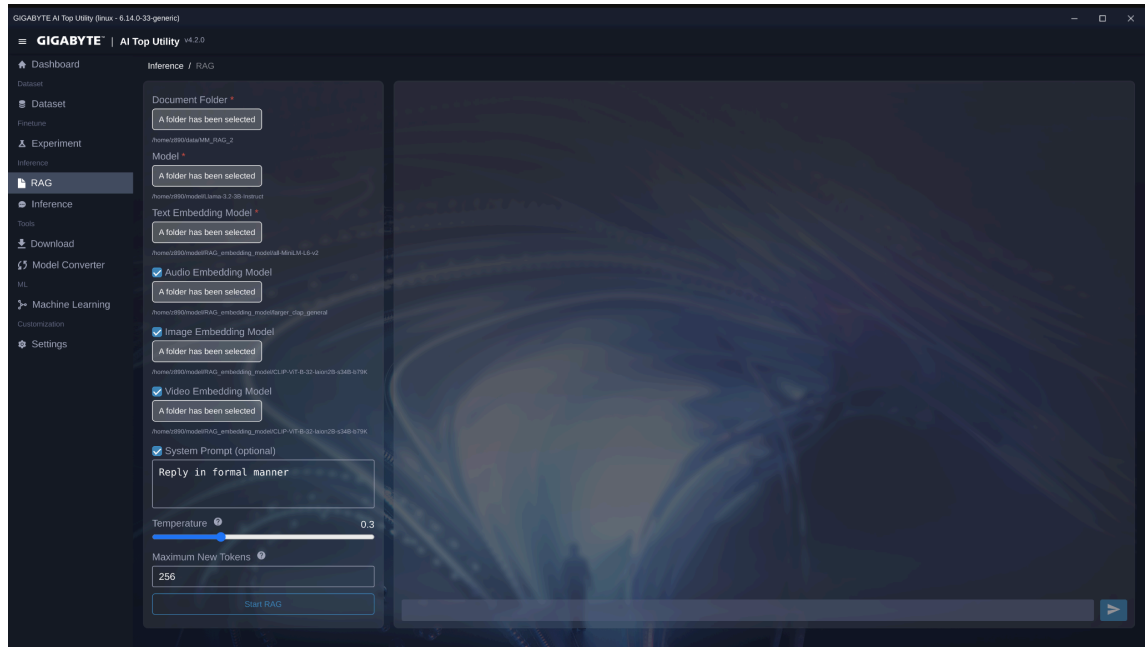
3-6. RAG

Here's a step-by-step explanation of how Retrieval-Augmented Generation (RAG) works with a vector database and a Large Language Model (LLM):

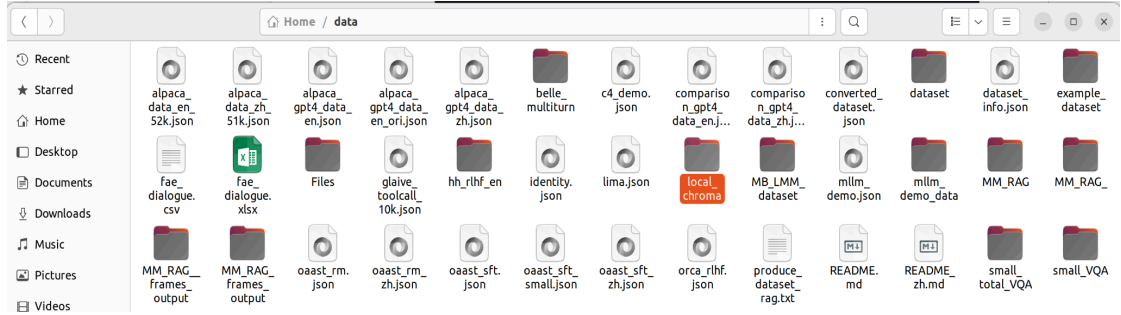
- **Document Encoding:** Documents are first processed and encoded into high-dimensional vectors using techniques such as embeddings. These vectors represent the semantic content of the documents and are stored in a vector database.
- **Query Encoding:** When a query is received, it is also converted into a vector using the same encoding method used for the documents.
- **Retrieval:** The query vector is compared against the vectors in the vector database. This comparison is typically done using similarity metrics like cosine similarity or dot product. The database retrieves the most similar documents or pieces of information based on the query vector.
- **Contextual Integration:** The retrieved documents are then used to provide context to the LLM. This means the LLM has access to relevant information that it can use to generate a response.
- **Generation:** The LLM processes the query along with the context from the retrieved documents. It generates a response that incorporates the relevant information to ensure accuracy and relevance.
- **Output:** The final output is a response generated by the LLM that is informed by both the query and the relevant documents retrieved from the vector database.

Let start with Multi-Modal RAG feature:

- (1) Click the "RAG" tab



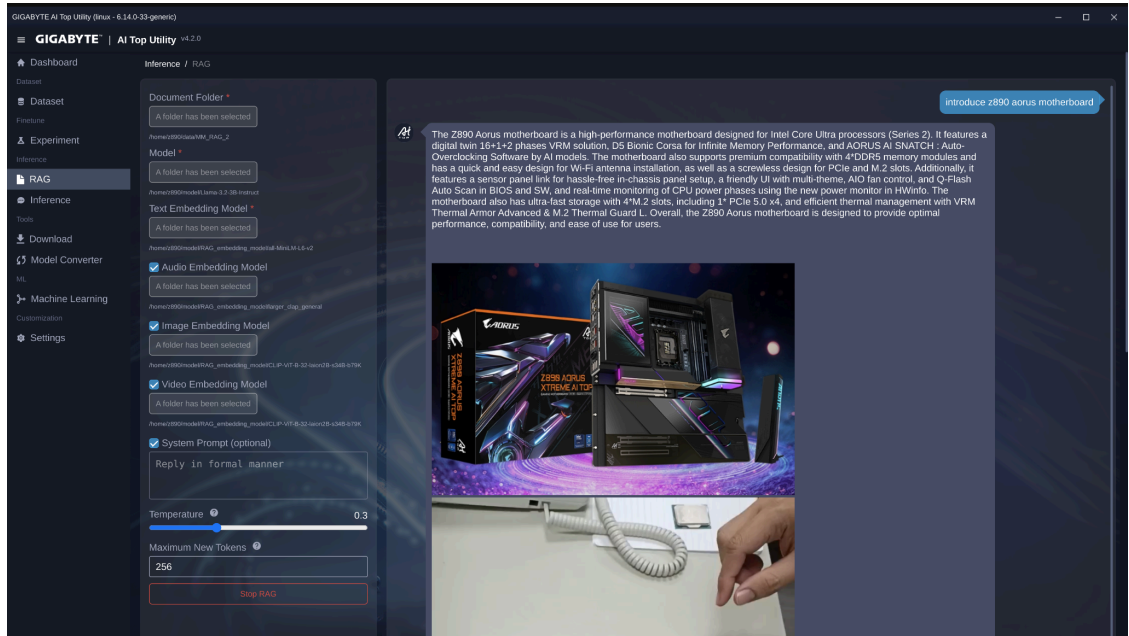
- (2) Under the “Document Folder,” select the directory containing the files you want to add to ChromaDB. We support the following file types: Images (.png/.jpg), Audio (.wav), Video (.mp4), and Documents (.txt, .json, .csv, .pdf, .docx, .xlsx).
- (3) Under the “Model”, Select the model you want to apply to RAG.
- (4) Under the “Text Embedding Model” section, select the text embedding model you want to use to convert the text into embedding vectors and save them into the ChromaDB text collection.
- (5) Under the “Audio Embedding Model” section, select the audio embedding model. If selected, the model will later return the audio that is most similar to your input.
- (6) Under the “Image Embedding Model” section, select the image embedding model. If selected, the model will later return the image that is most similar to your input.
- (7) Under the “Video Embedding Model” section, select the video embedding model. If selected, the model will later return the video that is most similar to your input.
- (8) Then, the Multi-Modal RAG will convert these files into embedding vectors and save them into ChromaDB (at ~/data/local_chroma).



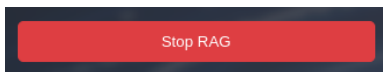
(9) Under the “System prompt”, set up the prompt for your inference chat, if desired. This is an optional setting.

(10) Set the **Temperature** and **Maximum New Tokens** to control the generation process.

(11) Start asking a question in the dialog box.



(12) Click "Stop RAG" if you want to stop asking for documents.

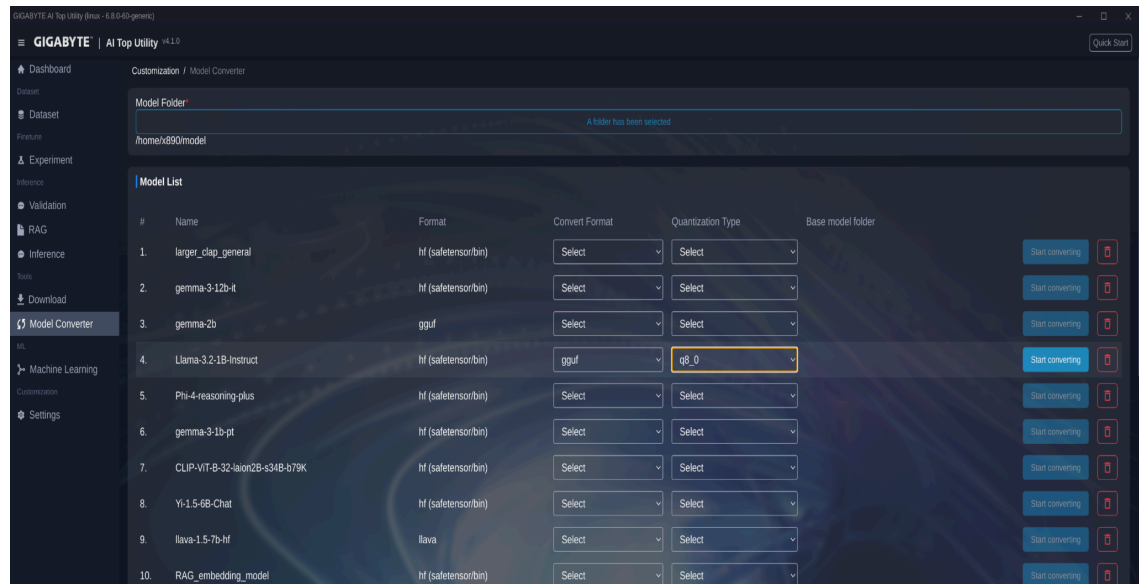


3-7. Model conversion

(1) Click the **"Model Converter"** tab.

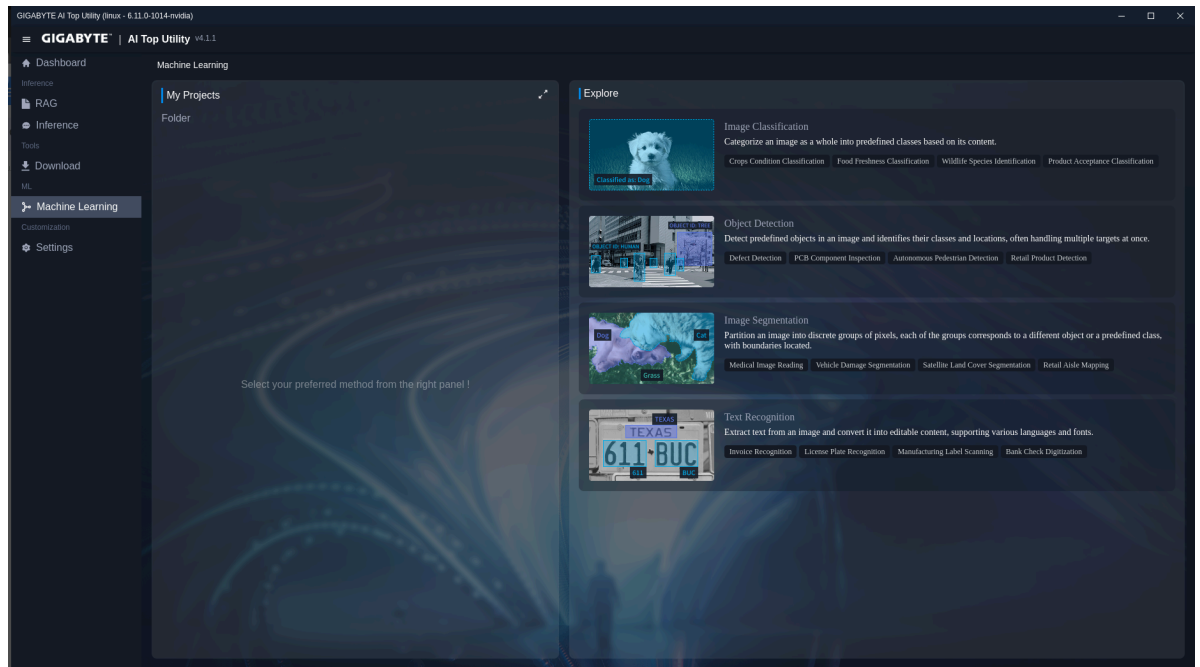
(2) Under the **"Model Folder"**, select the directory (usually `/home/{user}/model`) containing the models you wish to convert. The following model types are supported: **Hugging Face** (`.safetensors/.bin`), and **GGML** (`.ggmlv3`).

- (3) Under **"Convert Format"**, choose your desired output format. Currently, only **GGUF** is supported.
- (4) Select the quantization type for the output model. Supported types include: **f32**, **f16**, **bf16**, **q8_0**, **tq1_0**, **tq2_0**, and **auto**.
- (5) Click the **"Start Converting"** button to begin the conversion process. This may take approximately **15 to 30 seconds**.

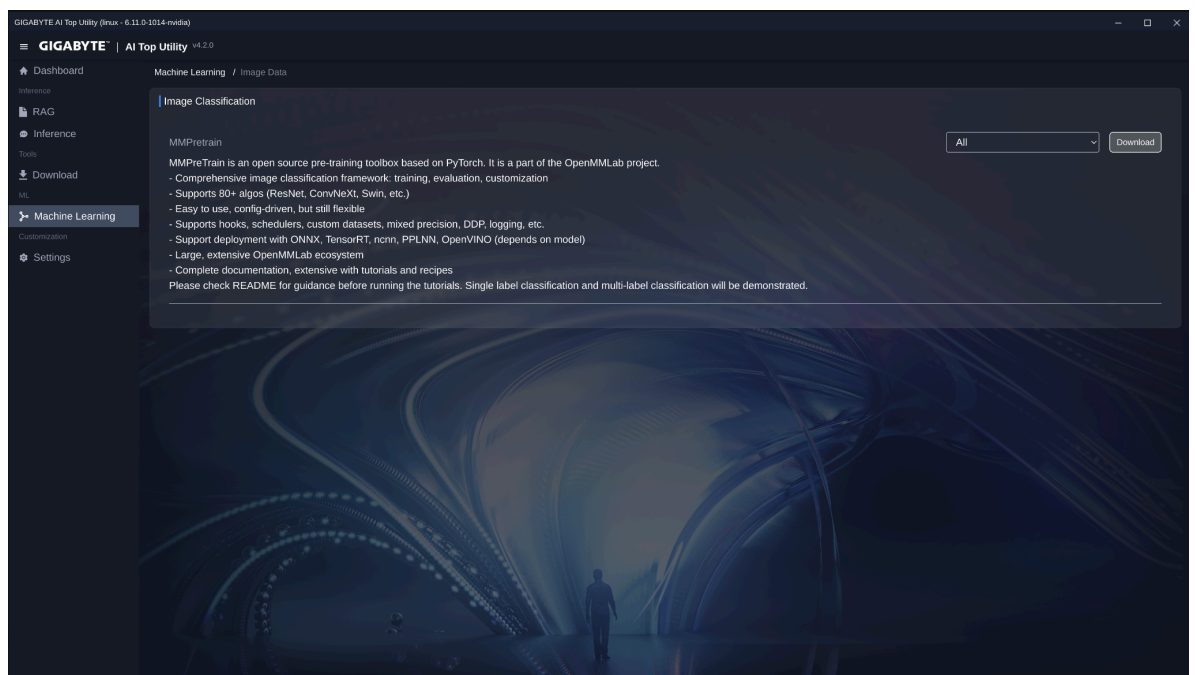


3-8. Machine Learning

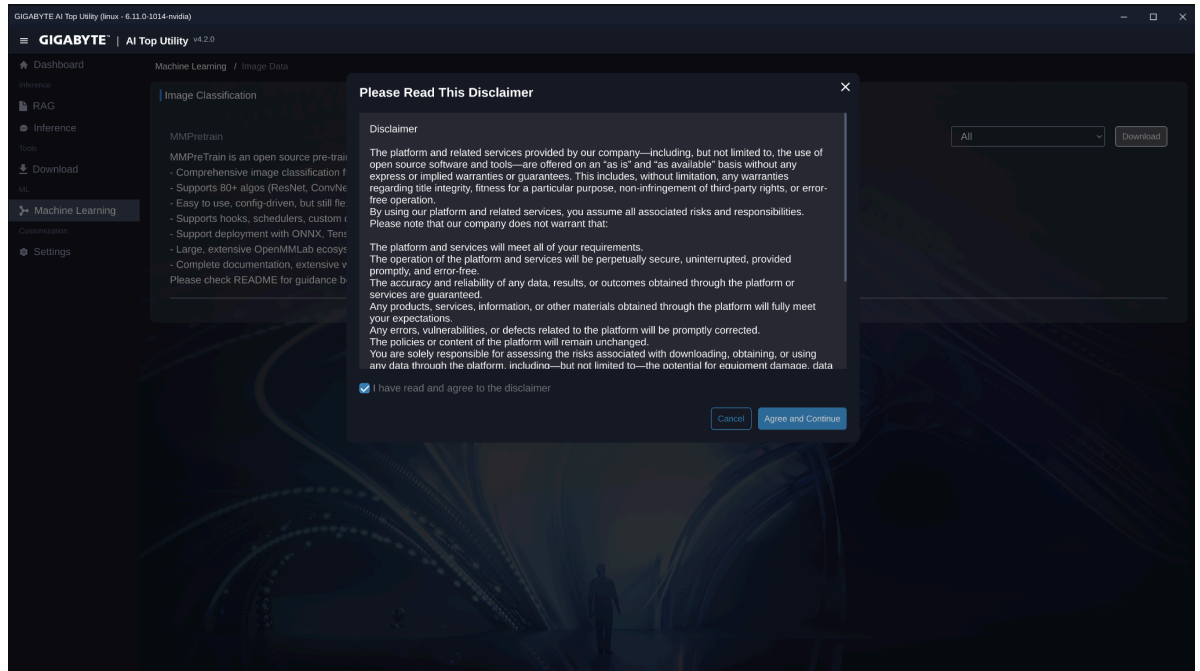
- (1) Click the **"Machine Learning"** tab. Choose the machine learning task type you want to work on. The currently supported types include: **image classification**, **object detection**, **image segmentation**, and **OCR** (Optical Character Recognition).



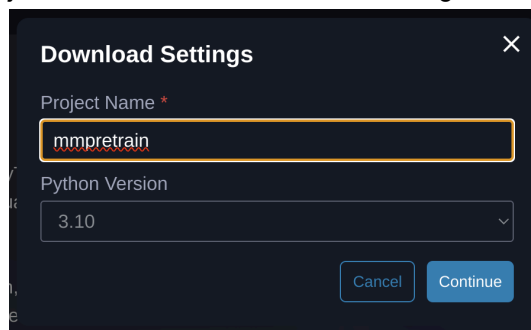
(2) Click “Select Model” and set it to “All”.



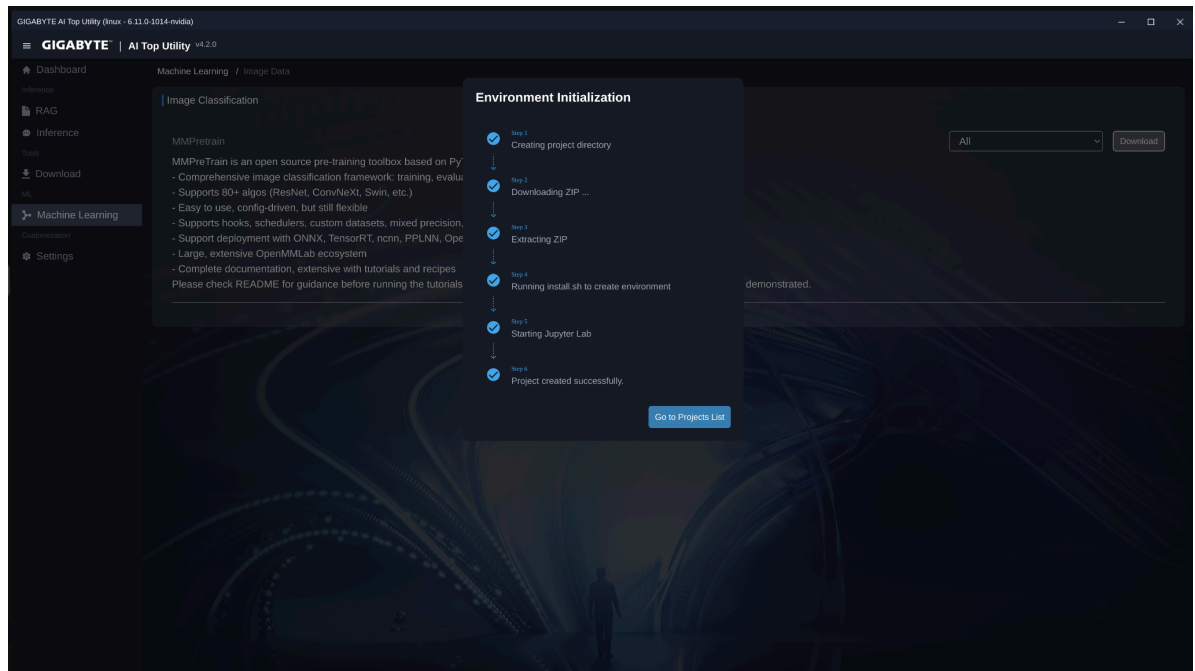
(3) Click Download. A disclaimer will appear — read it carefully, tick the “I have read” box, then press Agree.



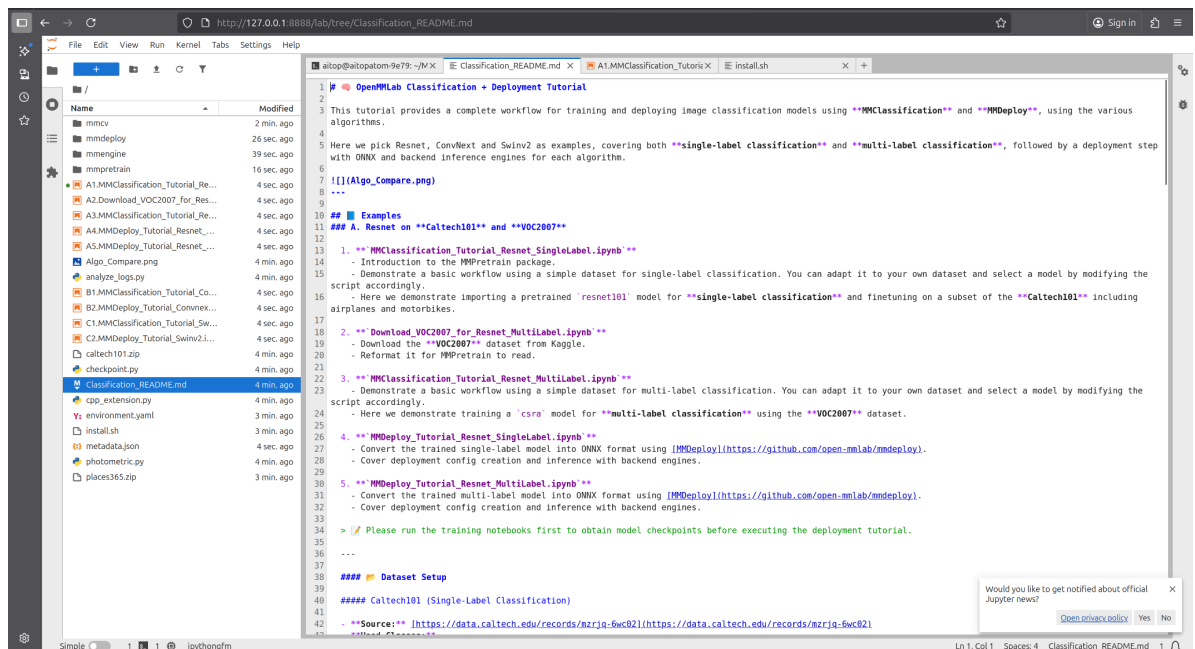
- (4) Enter a name for your project. The name can be modified later, but it cannot be duplicated. Click "Continue". The project download and installation will begin.



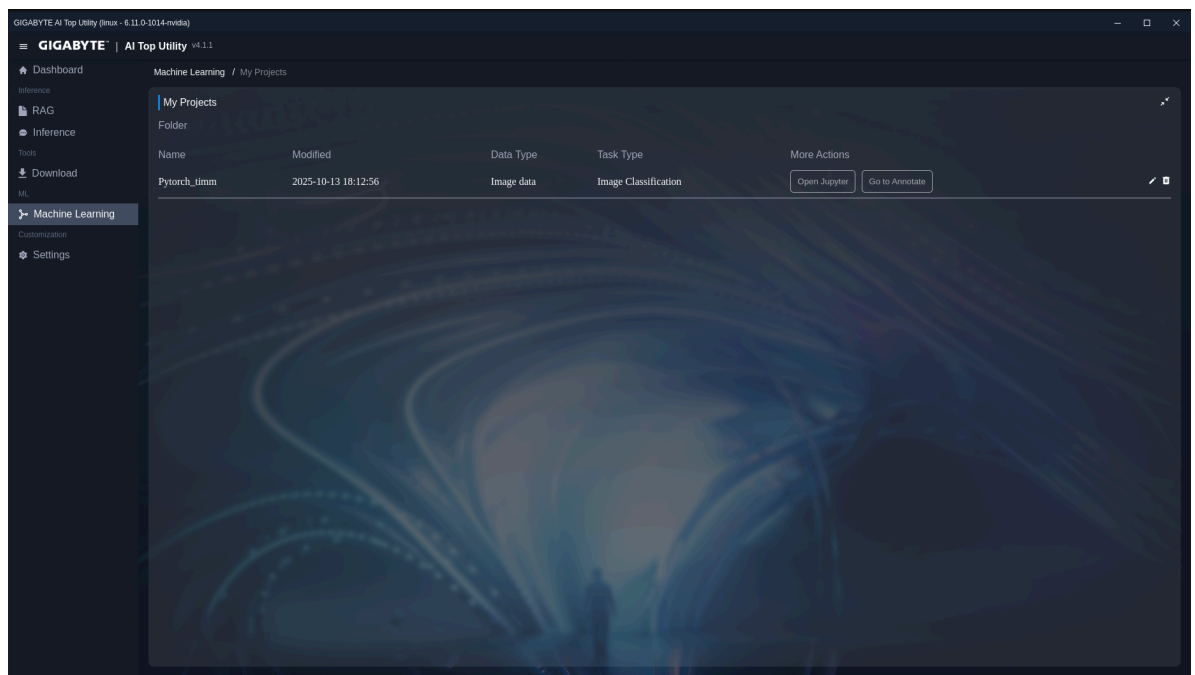
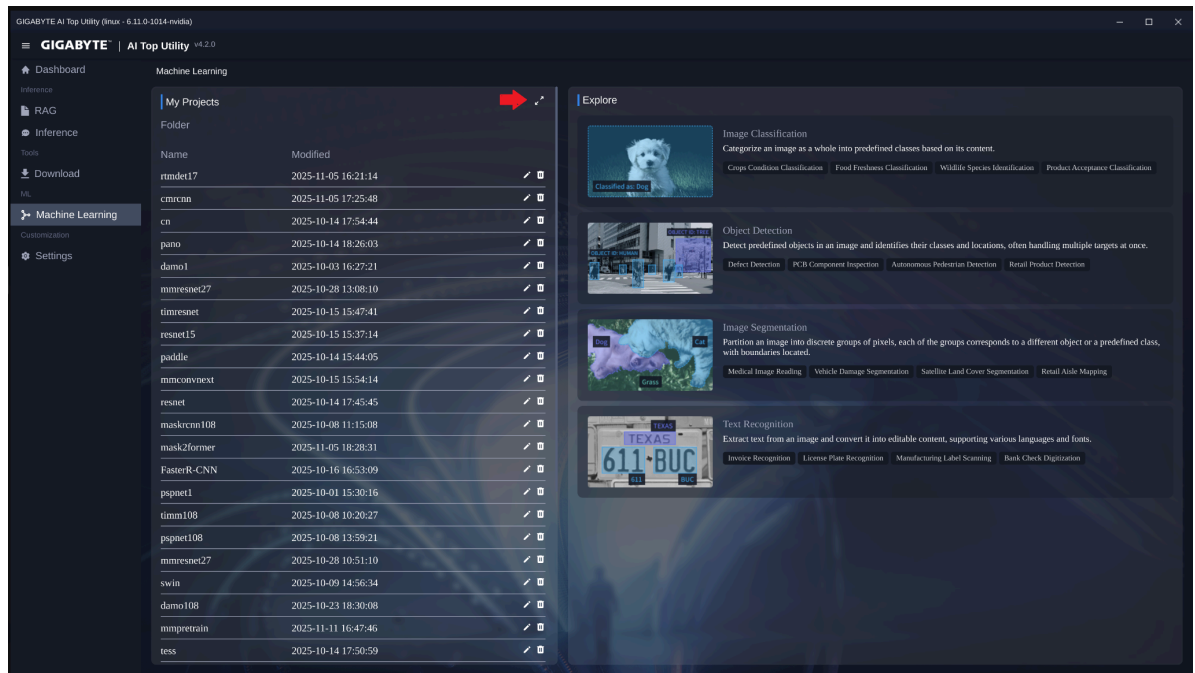
- (5) The process may take up to 60 minutes, depending on your internet connection.



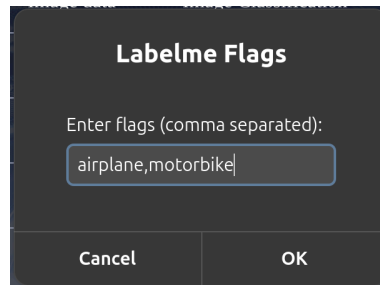
- (6) Once it is completed, a Jupyter Notebook will pop up automatically. Read the README and follow the instructions within the notebook to start learning how to build a machine learning project.



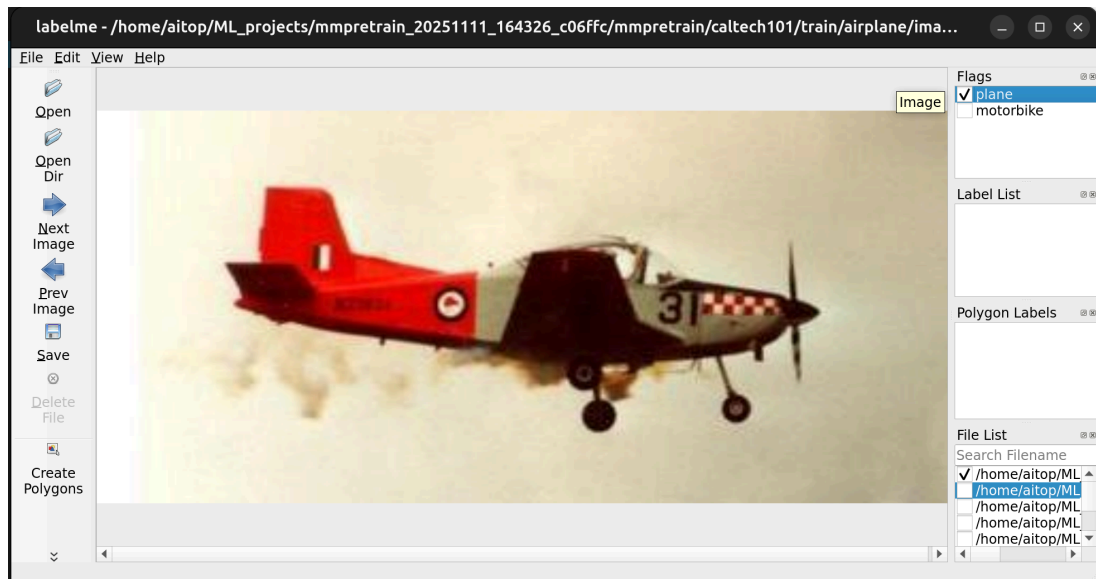
- (7) Click the button pointed by the arrow to display all existing projects saved on your device.



- (8) If you want to label your image dataset, click “Go to Annotate”. This will take you to the LabelMe or Labellmg interface, where you can make annotations.
- (9) If you are doing classification labeling, a box will pop out. Enter your class names. Separate them with commas, and do not enter any space. Then click “OK”.



Click “Open Dir” to set the directory and click corresponding class(es) under Flags.



(10) If you are doing object detection labeling, click “Open Dir” to set the directory, click “Create RectBox”, frame the object, and name the object.



- (11) If you are doing segmentation labeling, click “Open Dir” to set the directory, click “Create Polygons”, frame the region, and name the region. Repeat until each area of image is annotated as a certain class. You are recommended to label in the order of from far to near (e.g. sky > building > road > foreground object).



3-9. Finetune Expand config syntax

```
--model_name_or_path MODEL_NAME_OR_PATH
    Path to the model weight or identifier from huggingface.co/models or
    modelscope.cn/models. (default: None)
--adapter_name_or_path ADAPTER_NAME_OR_PATH
    Path to the adapter weight or identifier from huggingface.co/models. (default:
None)
--cache_dir CACHE_DIR
    Where to store the pre-trained models downloaded from huggingface.co or
    modelscope.cn. (default: None)
--quantization_bit QUANTIZATION_BIT
    The number of bits to quantize the model using bitsandbytes. (default: None)
--quantization_type {fp4,nf4}
    Quantization data type to use in int4 training. (default: nf4)
--double_quantization [DOUBLE_QUANTIZATION]
    Whether or not to use double quantization in int4 training. (default: True)
--flash_attn {off,sdpa,fa2,auto}
```

Enable FlashAttention for faster training and inference. (default: auto)

--use_cache [USE_CACHE]
Whether or not to use KV cache in generation. (default: True)

--no_use_cache Whether or not to use KV cache in generation. (default: False)

--export_dir EXPORT_DIR
Path to the directory to save the exported model. (default: None)

--export_size EXPORT_SIZE
The file shard size (in GB) of the exported model. (default: 1)

--template TEMPLATE Which template to use for constructing prompts in training and inference. (default: None)

--dataset DATASET The name of the provided dataset(s) to use. Use commas to separate multiple datasets. (default: None)

--dataset_dir DATASET_DIR
Path to the folder containing the datasets. (default: data)

--split SPLIT Which dataset split to use for training and evaluation. (default: train)

--cutoff_len CUTOFF_LEN
The cutoff length of the tokenized inputs in the dataset. (default: 1024)

--overwrite_cache [OVERWRITE_CACHE]
Overwrite the cached training and evaluation sets. (default: False)

--preprocessing_num_workers PREPROCESSING_NUM_WORKERS
The number of processes to use for the pre-processing. (default: None)

--max_samples MAX_SAMPLES
For debugging purposes, truncate the number of examples for each dataset. (default: None)

--val_size VAL_SIZE Size of the development set, should be an integer or a float in range $[0,1)$. (default: 0.0)

--output_dir OUTPUT_DIR
The output directory where the model predictions and checkpoints will be written. (default: None)

--overwrite_output_dir [OVERWRITE_OUTPUT_DIR]
Overwrite the content of the output directory. Use this to continue training if output_dir points to a checkpoint directory. (default: False)

--do_train [DO_TRAIN]
Whether to run training. (default: False)

--do_eval [DO_EVAL] Whether to run eval on the dev set. (default: False)

--do_predict [DO_PREDICT]

Whether to run predictions on the test set. (default: False)

--evaluation_strategy {no,steps,epoch}

The evaluation strategy to use. (default: no)

--prediction_loss_only [PREDICTION_LOSS_ONLY]

When performing evaluation and predictions, only returns the loss. (default: False)

--per_device_train_batch_size PER_DEVICE_TRAIN_BATCH_SIZE

Batch size per GPU/TPU/MPS/NPU core/CPU for training. (default: 8)

--per_device_eval_batch_size PER_DEVICE_EVAL_BATCH_SIZE

Batch size per GPU/TPU/MPS/NPU core/CPU for evaluation. (default: 8)

--per_gpu_train_batch_size PER_GPU_TRAIN_BATCH_SIZE

Deprecated, the use of `--per_device_train_batch_size` is preferred. Batch size per GPU/TPU core/CPU for training. (default: None)

--per_gpu_eval_batch_size PER_GPU_EVAL_BATCH_SIZE

Deprecated, the use of `--per_device_eval_batch_size` is preferred. Batch size per GPU/TPU core/CPU for evaluation. (default: None)

--gradient_accumulation_steps GRADIENT_ACCUMULATION_STEPS

Number of update steps to accumulate before performing a backward/update pass. (default: 1)

--eval_accumulation_steps EVAL_ACCUMULATION_STEPS

Number of prediction steps to accumulate before moving the tensors to the CPU. (default: None)

--eval_delay EVAL_DELAY

Number of epochs or steps to wait for before the first evaluation can be performed, depending on the evaluation_strategy. (default: 0)

--learning_rate LEARNING_RATE

The initial learning rate for AdamW. (default: 5e-05)

--weight_decay WEIGHT_DECAY

Weight decay for AdamW if we apply some. (default: 0.0)

--max_grad_norm MAX_GRAD_NORM

Max gradient norm. (default: 1.0)

--num_train_epochs NUM_TRAIN_EPOCHS

Total number of training epochs to perform. (default: 3.0)

--max_steps MAX_STEPS

If > 0: set total number of training steps to perform. Override num_train_epochs. (default: -1)

--lr_scheduler_type {linear, cosine, cosine_with_restarts, polynomial, constant, constant_with_warmup, inverse_sqrt, reduce_lr_on_plateau, cosine_with_min_lr}
The scheduler type to use. (default: linear)

--lr_scheduler_kwargs LR_SCHEDULER_KWARGS
Extra parameters for the lr_scheduler such as {'num_cycles': 1} for the cosine with hard restarts. (default: {})

--warmup_ratio WARMUP_RATIO
Linear warmup over warmup_ratio fraction of total steps. (default: 0.0)

--warmup_steps WARMUP_STEPS
Linear warmup over warmup_steps. (default: 0)

--logging_dir LOGGING_DIR
Tensorboard log dir. (default: None)

--logging_strategy {no,steps,epoch}
The logging strategy to use. (default: steps)

--logging_first_step [LOGGING_FIRST_STEP]
Log the first global_step (default: False)

--logging_steps LOGGING_STEPS
Log every X update steps. Should be an integer or a float in range '[0,1)'. If smaller than 1, it will be interpreted as a ratio of total training steps. (default: 500)

--save_strategy {no,steps,epoch}
The checkpoint save strategy to use. (default: steps)

--save_steps SAVE_STEPS
Save checkpoint every X update steps. Should be an integer or a float in range '[0,1)'. If smaller than 1, it will be interpreted as a ratio of total training steps. (default: 500)

--bf16 [BF16] Whether to use bf16 (mixed) precision instead of 32-bit. Requires Ampere or higher NVIDIA architecture or using CPU (use_cpu) or Ascend NPU.

--fp16 [FP16] Whether to use fp16 (mixed) precision instead of 32-bit (default: False)

--fp16_opt_level FP16_OPT_LEVEL
For fp16: Apex AMP optimization level selected in ['O0', 'O1', 'O2', and 'O3']. See details at <https://nvidia.github.io/apex/amp.html> (default: O1)

--half_precision_backend {auto,apex,cpu_amp}
The backend to be used for half precision. (default: auto)

--bf16_full_eval [BF16_FULL_EVAL]

Whether to use full bfloat16 evaluation instead of 32-bit. This is an experimental API and it may change. (default: False)

`--fp16_full_eval [FP16_FULL_EVAL]`

Whether to use full float16 evaluation instead of 32-bit (default: False)

`--tf32 TF32` Whether to enable tf32 mode, available in Ampere and newer GPU architectures. This is an experimental API and it may change. (default: None)

`--local_rank LOCAL_RANK`

For distributed training: local_rank (default: -1)

`--dataloader_num_workers DATALOADER_NUM_WORKERS`

Number of subprocesses to use for data loading (PyTorch only). 0 means that the data will be loaded in the main process. (default: 0)

`--deepspeed DEEPSPEED`

Enable deepspeed and pass the path to deepspeed json config file (e.g. `'ds_config.json'`) or an already loaded json file as a dict (default: None)

`--resume_from_checkpoint RESUME_FROM_CHECKPOINT`

The path to a folder with a valid checkpoint for your model. (default: None)

`--ddp_timeout DDP_TIMEOUT`

Overrides the default timeout for distributed training (value should be given in seconds). (default: 1800)

`--torch_compile [TORCH_COMPILE]`

If set to `'True'`, the model will be wrapped in `'torch.compile'`. (default: False)

The rank of GaLore gradients. (default: 16)

`--ref_model_adapters REF_MODEL_ADAPTERS`

Path to the adapters of the reference model. (default: None)

`--ref_model_quantization_bit REF_MODEL_QUANTIZATION_BIT`

The number of bits to quantize the reference model. (default: None)

`--reward_model REWARD_MODEL`

Path to the reward model used for the PPO training. (default: None)

`--reward_model_adapters REWARD_MODEL_ADAPTERS`

Path to the adapters of the reward model. (default: None)

`--reward_model_quantization_bit REWARD_MODEL_QUANTIZATION_BIT`

The number of bits to quantize the reward model. (default: None)

`--reward_model_type {lora,full,api}`

The type of the reward model in PPO training. Lora model only supports lora training. (default: lora)

`--additional_target ADDITIONAL_TARGET`
 Name(s) of modules apart from LoRA layers to be set as trainable and saved in the final checkpoint. (default: None)

`--lora_alpha LORA_ALPHA`
 The scale factor for LoRA fine-tuning (default: $\text{lora_rank} * 2$). (default: None)

`--lora_dropout LORA_DROPOUT`
 Dropout rate for the LoRA fine-tuning. (default: 0.0)

`--lora_rank LORA_RANK`
 The intrinsic dimension for LoRA fine-tuning. (default: 8)

`--lora_target LORA_TARGET`
 Name(s) of target modules to apply LoRA. Use commas to separate multiple modules. Use "all" to specify all the linear modules. LLaMA choices: ["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj"], BLOOM & Falcon & ChatGLM choices: ["query_key_value", "dense", "dense_h_to_4h", "dense_4h_to_h"], Baichuan choices: ["W_pack", "o_proj", "gate_proj", "up_proj", "down_proj"], Qwen choices: ["c_attn", "attn.c_proj", "w1", "w2", "mlp.c_proj"], InternLM2 choices: ["wqkv", "wo", "w1", "w2", "w3"], Others choices: the same as LLaMA. (default: all)

`--create_new_adapter [CREATE_NEW_ADAPTER]`
 Whether or not to create a new adapter with randomly initialized weight. (default: False)

`--name_module_trainable NAME_MODULE_TRAINABLE`
 Name of trainable modules for partial-parameter (freeze) fine-tuning. Use commas to separate multiple modules. Use "all" to specify all the available modules. LLaMA choices: ["mlp", "self_attn"], BLOOM & Falcon & ChatGLM choices: ["mlp", "self_attention"], Qwen choices: ["mlp", "attn"], InternLM2 choices: ["feed_forward", "attention"], Others choices: the same as LLaMA. (default: all)

`--num_layer_trainable NUM_LAYER_TRAINABLE`
 The number of trainable layers for partial-parameter (freeze) fine-tuning. (default: 2)

`--stage {pt,sft,rm,ppo,dpo,orpo}`
 Which stage will be performed in training. (default: sft)

`--finetuning_type {lora,freeze,full}`
 Which fine-tuning method to use. (default: lora)

`--use_llama_pro [USE_LLAMA_PRO]`

Whether or not to make only the parameters in the expanded blocks trainable.
(default: False)

--plot_loss [PLOT_LOSS]
Whether or not to save the training loss curves. (default: False)

--do_sample [DO_SAMPLE]
Whether or not to use sampling, use greedy decoding otherwise. (default: True)

4. Supported Models

4-1. Supported list of LLM models

Users need to download LLM backbone models from the Hugging Face website or another open-source platform before fine-tuning. We recommend that users conduct thorough research on how to fine-tune each LLM backbone model by setting training configurations and then try to fine-tune using AI TOP Utility. If you are a professional LLM trainer, please use the 'Clear' option and the 'Expand Config' feature to create your own fine-tuning settings.

Model	Model size	Template
Qwen/Qwen3	4B/8B Qwen/Qwen3-4B Qwen/Qwen3-8B	Qwen
Llama 3	1B/3B/8B/70B Meta-Llama-3-8B-Instruct Meta-Llama-3.2-1B-Instruct Meta-Llama-3.2-3B-Instruct Meta-Llama-3.3-70B-Instruct	llama3
Gemma/Gemma 2	2B/9B gemma-2-2b-it gemma-2-9b-it	gemma
Mistral/Mixtral	7B Mistral-7B-Instruct-v0.1 Mistral-7B-Instruct-v0.2 Mistral-7B-Instruct-v0.3	Mistral

4-2. Supported list of LMM models

Model	Task	Model size	Template
Stabilityai	Text-to-image	Stable-diffusion-3-medium-diffusers stabilityai/stable-diffusion-3.5-medium	-
black-forest-lab	Text-to-image	FLUX.1-dev FLUX.1-schnell	-
zai-org	Text-to-video	CogVideoX-2b CogVideoX-5b	-
Qwen-VL	Image-text-to-text	Qwen2.5-VL-3B-Instruct Qwen2.5-VL-7B-Instruct	Qwen-VL
Llama 3	Image-text-to-text	Meta-Llama-3.2-11B-Vision-Instruct (22.35G)	Llama3
LLaVA-1.5	Image-text-to-text	7B llava-1.5-7b-hf	Vicuna

4-3. Supported list of embedding models (RAG)

Model	Task	Model size	Template
Sentence Transformers	text embedding model	all-MiniLM-L6-v2	-
laion	audio embedding model	larger_clap_general	-
laion	Image embedding model	CLIP-ViT-B-32-laion2B-s34B-b79K	-
laion	video embedding model	CLIP-ViT-B-32-laion2B-s34B-b79K	-

4-4. Download Model

The “model” folder in the Home directory is for locating your LLM/LMM backbone models. Before downloading, you need to create a Hugging Face account and generate a User Access Token to

download LLM/LMM backbone models. You can refer to the [User access tokens](#) page for guidance on creating one. Be sure to store your Access Token securely so you do not lose it.

The software provides a convenient feature to help users download LLM/LMM models from Hugging Face without needing to run command line syntax.

Note: Some models (e.g, Llama family models) require users to agree to share contact information to access models in order to download from Hugging Face, so you need to visit the model's page to submit a request and grant approval.

