Generative AI and AIoT (GenAIoT) Coding Skills Education

Lab Session 5.4

Image generation and multimodal LLM on Jetson AGX Orin

Overview:

Image generation is one of the interesting and trending AI applications useful for several tasks including digital twin, biomedical engineering and soon. In this lab, we are going to use a stable-diffusion model to generate images using the Jetson AGX Orin developer kit. We will also experiment with a multimodal Vision Transformer (LLava).

Required:

1. Jetson AGX Orin developer kit
2. Internet connection
3. Assume you have flashed the board with Jetpack 6

Part 1: Generate images with stable-diffusion-webui

Jetson container provides some useful docker images that can be used on Jetson boards, we would use the stable-diffusion-webui to try generating images with a text prompt.

Procedure:

(Reference: https://www.jetson-ai-lab.com/tutorial_stable-diffusion.html)

1. Run the stable-diffusion-webui (Jetson container will pull the image if not found). Use *jetson-containers run* and *autotag* tools to automatically pull or build a compatible container image. The container has a default run command (`CMD`) that will automatically start the webserver.

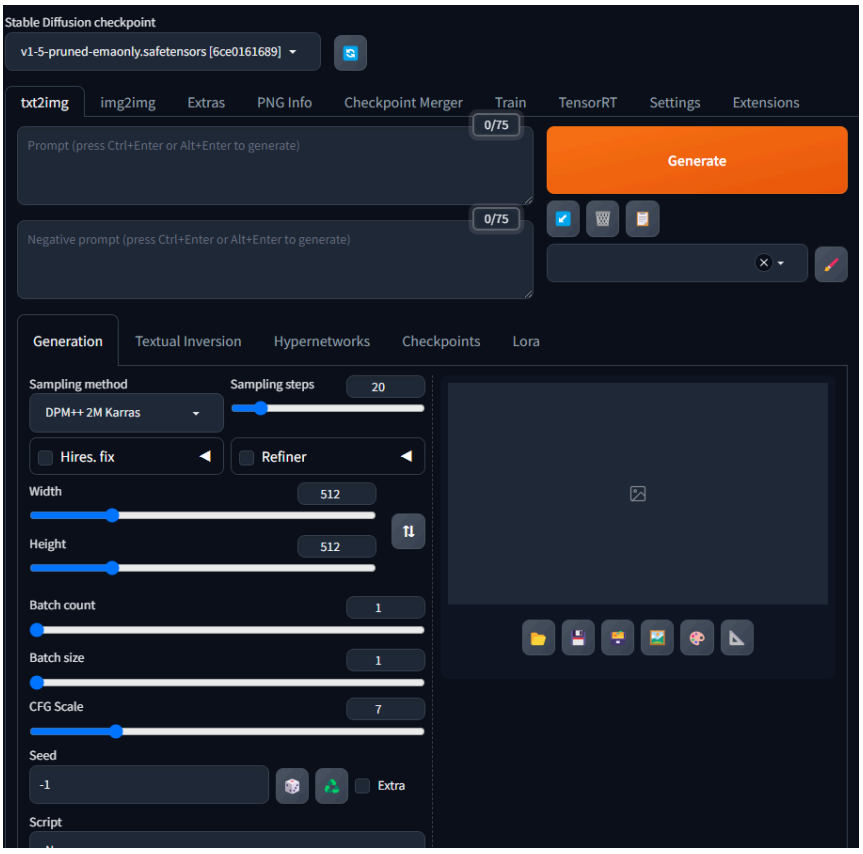
```
>> jetson-containers run $(autotag stable-diffusion-webui)
```

```
Commit hash: cf2772fab0af5573da775e7437e6acdca424f26e
Launching Web UI with arguments: --data=/data/models/stable-diffusion --enable-insecure-extension-acce
ss --xformers --listen --port=7860
Style database not found: /data/models/stable-diffusion/styles.csv
Downloading: "https://huggingface.co/runwayml/stable-diffusion-v1-5/resolve/main/v1-5-pruned-emaonly.s
afetensors" to /data/models/stable-diffusion/models/Stable-diffusion/v1-5-pruned-emaonly.safetensors

100%|| 3.97G/3.97G [01:46<00:00, 40.1MB/s]
/opt/stable-diffusion-webui/extensions-builtin/stable-diffusion-webui-tensorrt/ui_trt.py:64: GradioDep
recationWarning: The `style` method is deprecated. Please set these arguments in the constructor inste
ad.
  with gr.Row().style(equal_height=False):
Calculating sha256 for /data/models/stable-diffusion/models/Stable-diffusion/v1-5-pruned-emaonly.safet
ensors: Running on local URL:  http://0.0.0.0:7860

To create a public link, set `share=True` in `launch()`.
Startup time: 124.1s (prepare environment: 3.4s, import torch: 5.1s, import gradio: 2.0s, setup paths:
 1.8s, initialize shared: 0.3s, other imports: 1.6s, setup codeformer: 0.2s, list SD models: 107.1s, l
oad scripts: 1.1s, create ui: 0.8s, gradio launch: 0.6s).
6ce0161689b3853acaa03779ec93eafe75a02f4ced659bee03f50797806fa2fa
Loading weights [6ce0161689] from /data/models/stable-diffusion/models/Stable-diffusion/v1-5-pruned-em
aonly.safetensors
Creating model from config: /opt/stable-diffusion-webui/configs/v1-inference.yaml
vocab.json: 100%| 961k/961k [00:00<00:00, 2.37MB/s]
merges.txt: 100%| 525k/525k [00:00<00:00, 20.5MB/s]
special_tokens_map.json: 100%| 389/389 [00:00<00:00, 845kB/s]
tokenizer_config.json: 100%| 905/905 [00:00<00:00, 2.02MB/s]
config.json: 100%| 4.52k/4.52k [00:00<00:00, 9.62MB/s]
Applying attention optimization: xformers... done.
Model loaded in 15.8s (calculate hash: 6.3s, load weights from disk: 0.4s, create model: 3.9s, apply w
eights to model: 4.0s, apply half(): 0.2s, load textual inversion embeddings: 0.5s, calculate empty pr
ompt: 0.3s).
```

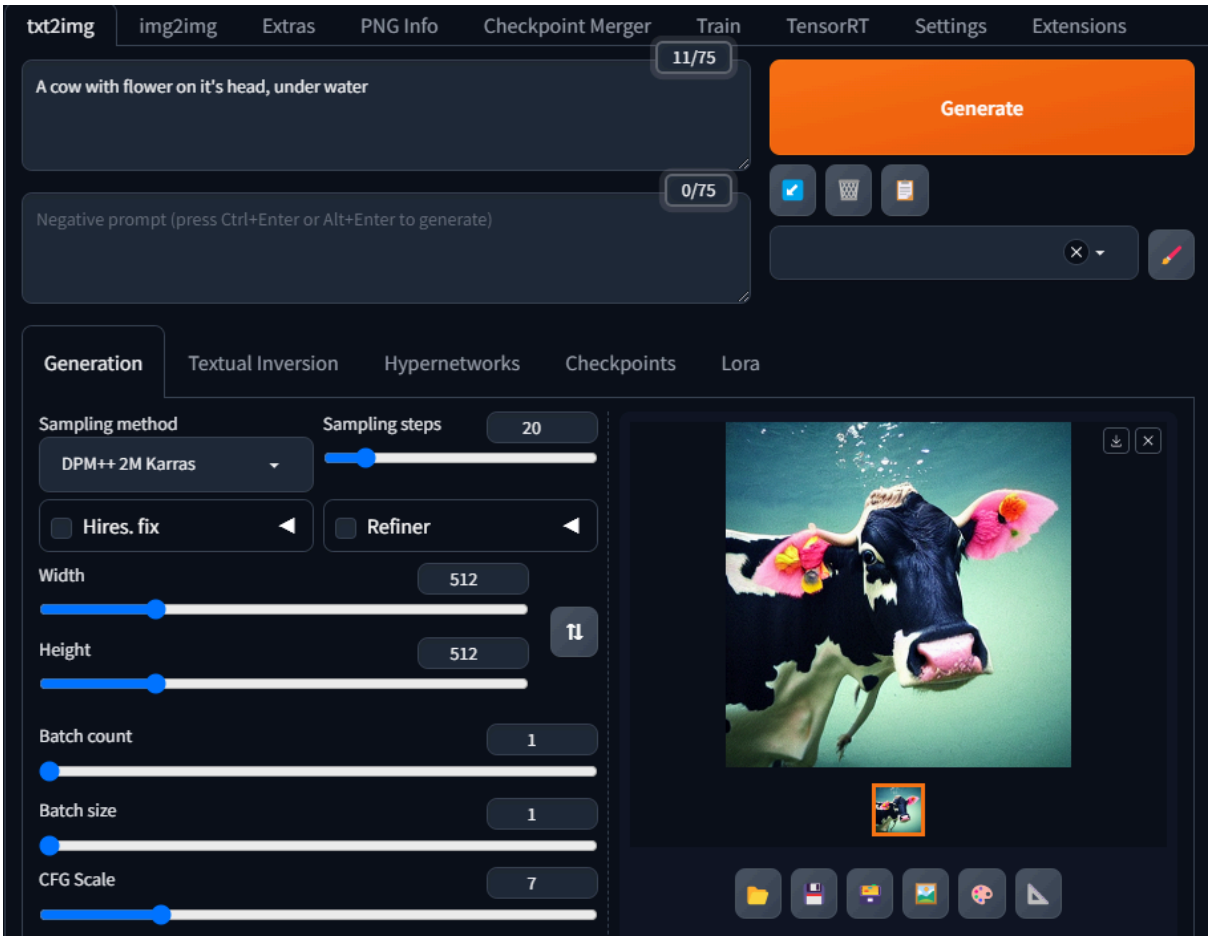
2. Open the Webui from a bowser using port:7860

- Open your browser and access http://<IP_ADDRESS>:7860
- 192.168.50.XX:7860 (XX is your assigned board number e.g 60)

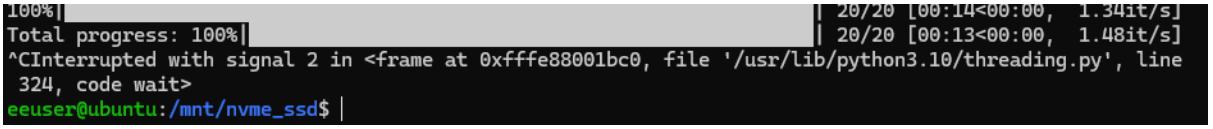
192.168.50.60:7860



3. Generate the images with text prompt
    - Enter the text prompt
    - Press Generate



4. Exit the server
    - Press ctrl + c in the terminal



Part 2: Using multimodal model with text-generation-webui

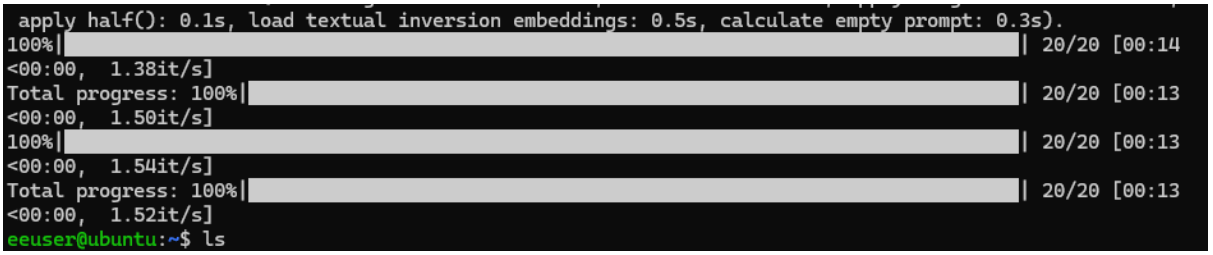
The text-generation-webui supports multimodal models. In this lab would demonstrate how to download a multimodal model and run the WebUI server with image input. LLaVA is a popular multimodal vision/language model that you can run locally on Jetson to answer questions about image prompts and queries. Llava uses the CLIP vision encoder to transform images into the same embedding space as its LLM (which is the same as Llama architecture). Below we cover different methods to run Llava on Jetson, with increasingly optimized performance.

Procedure:

(Reference: https://www.jetson-ai-lab.com/tutorial_llava.html)

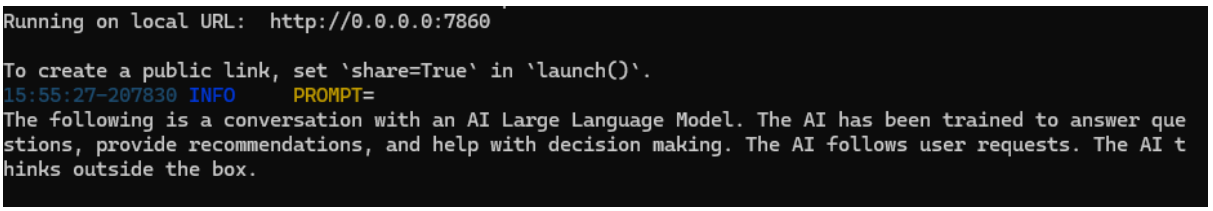
1. Download the llava model

```
>> jetson-containers run --workdir=/opt/text-generation-webui $(autotag
text-generation-webui) \
  python3 download-model.py --output=/data/models/text-generation-webui \
    TheBloke/llava-v1.5-13B-GPTQ
```



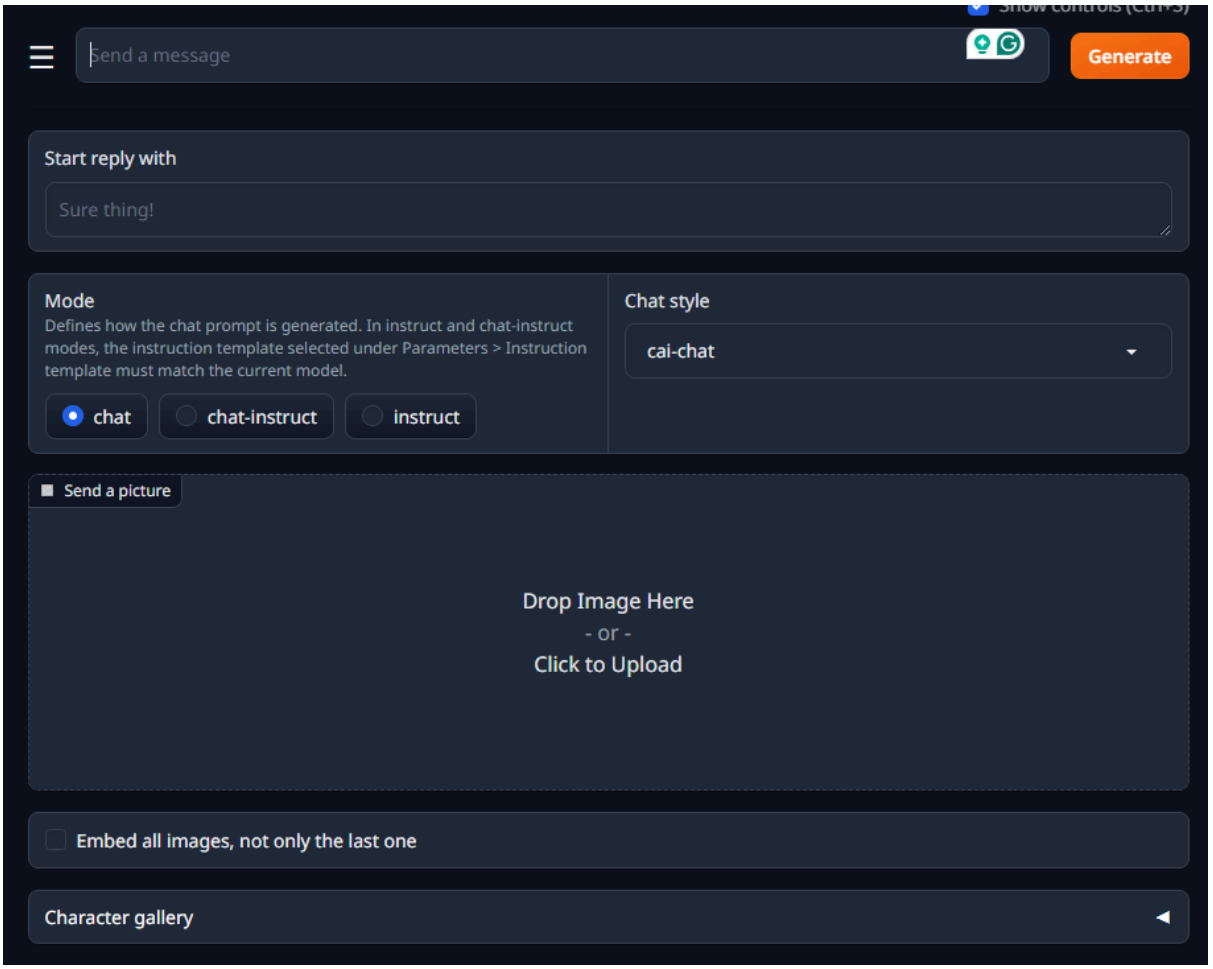
2. Run the text-geneartion-webui server with multimodal input support and llava server

```
>>jetson-containers run --workdir=/opt/text-generation-webui $(autotag
text-generation-webui) \
  python3 server.py --listen \
    --model-dir /data/models/text-generation-webui \
    --model TheBloke_llava-v1.5-13B-GPTQ \
    --multimodal-pipeline llava-v1.5-13b \
    --loader autogptq \
    --disable_exllama \
    --verbose
```



3. Chat with the AI

- Enter the text in “send a message” box
- Send a picture by upload or dragging
- Press Generate



4. Exit the server
    - Press ctrl + c in the terminal

Bonus Part: Stable-diffusion XL

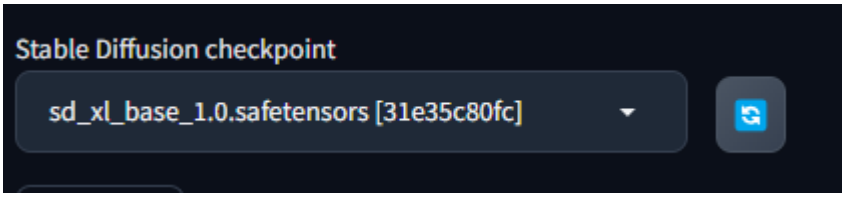
The quality of the images are low using the stable-diffusion-webui, this is because there are only 0.98B parameters in the model, there is a SDXL model with 6.6B parameters that can give a better image quality.

Procedure:

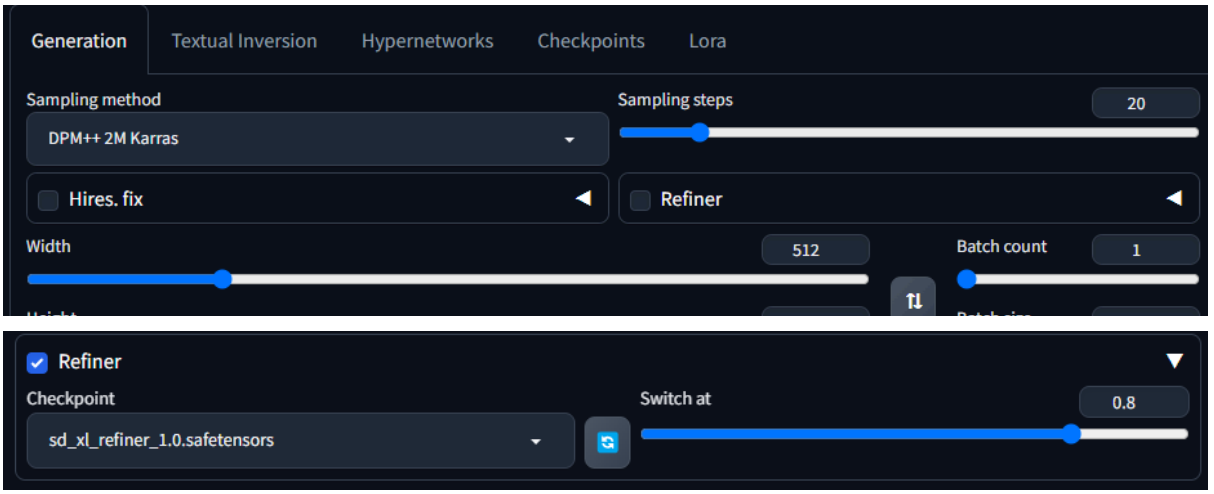
1. Download the model from Hugging Face

```
>>CONTAINERS_DIR=/mnt/nvme_ssd/jetson-containers
MODEL_DIR=$CONTAINERS_DIR/data/models/stable-diffusion/models/Stable-diffusion/
sudo chown -R $USER $MODEL_DIR
wget -P $MODEL_DIR https://huggingface.co/stabilityai/stable-diffusion-xl-base-1.0/resolve/main/sd_xl_base_1.0.safetensors
wget -P $MODEL_DIR https://huggingface.co/stabilityai/stable-diffusion-xl-refiner-1.0/resolve/main/sd_xl_refiner_1.0.safetensors
```

2. Open the WebUI and change the model



3. Choose the installed refiner



4. Increase the resolution of the images

5. Generate the image