CS492D: Diffusion Models and Their Applications

Assignment 3 Session

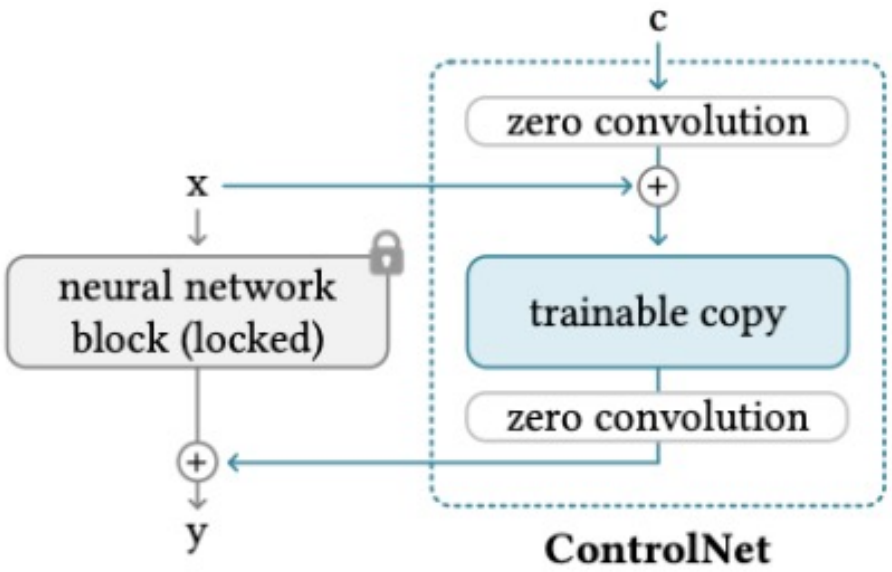
YUSEUNG LEE

Fall 2024
KAIST

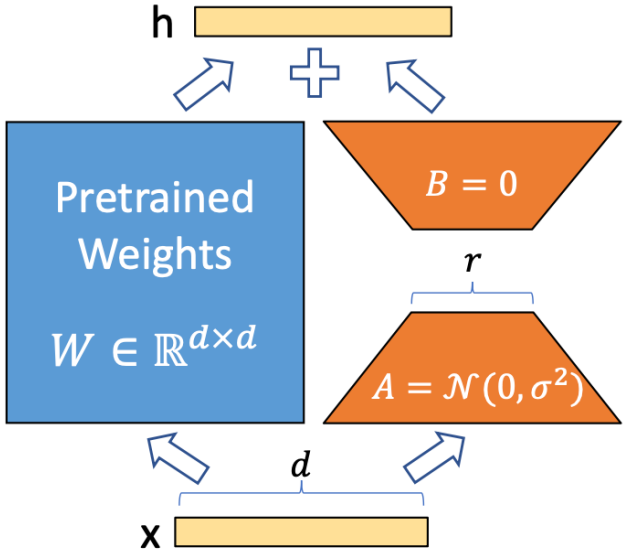
Introduction

Assignment 3 consists of **two** tasks: ControlNet and LoRA.

Task 1. ControlNet

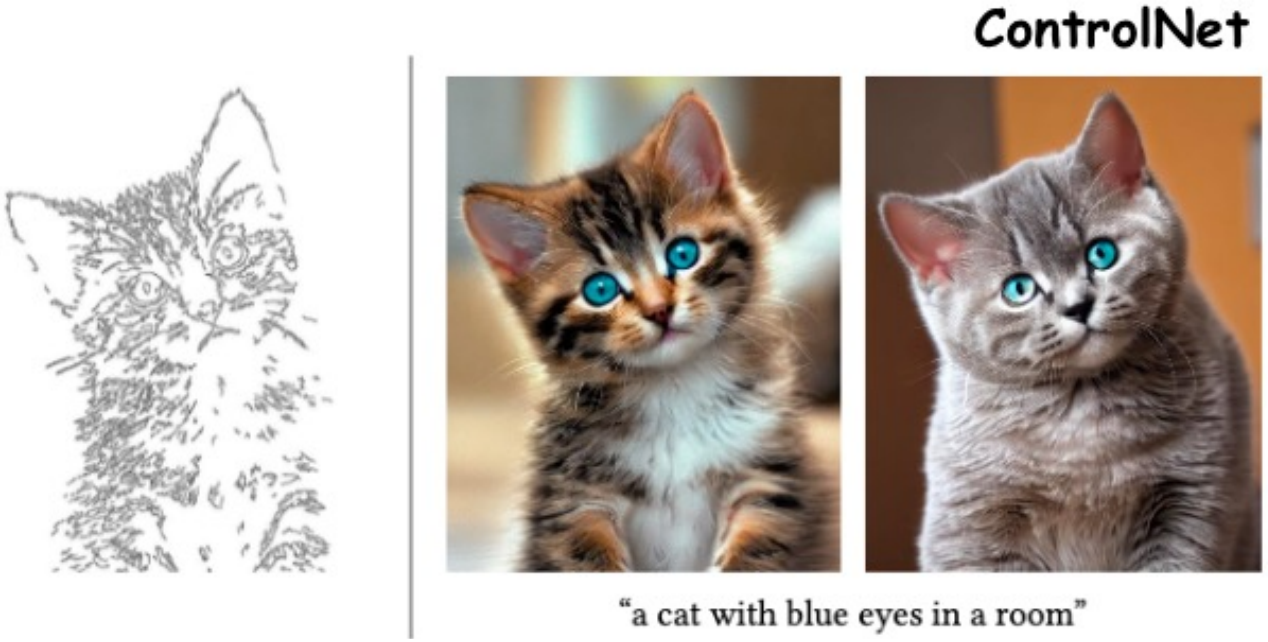


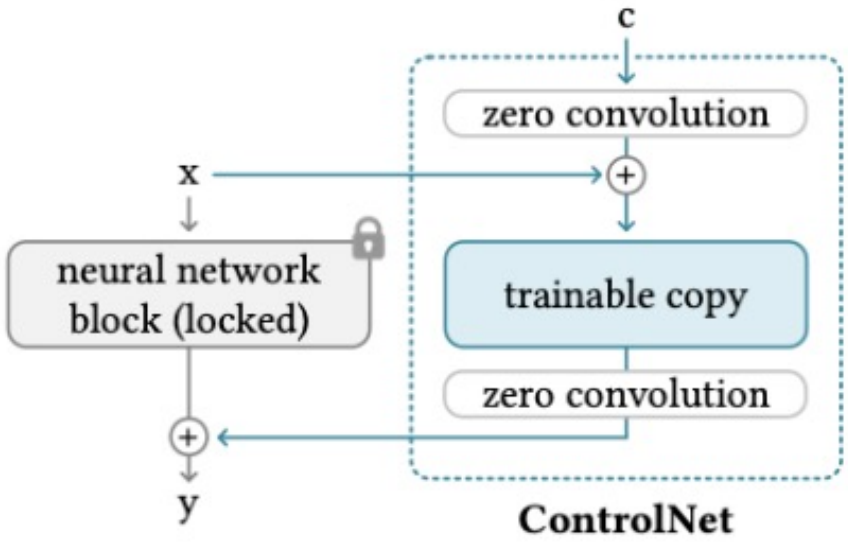
Task 2. LoRA



Introduction

In Task 1, you will implement the key components of **ControlNet**.





Adding Conditional Control to Text-to-Image Diffusion Models, Zhang *et al.*, ICCV 2023

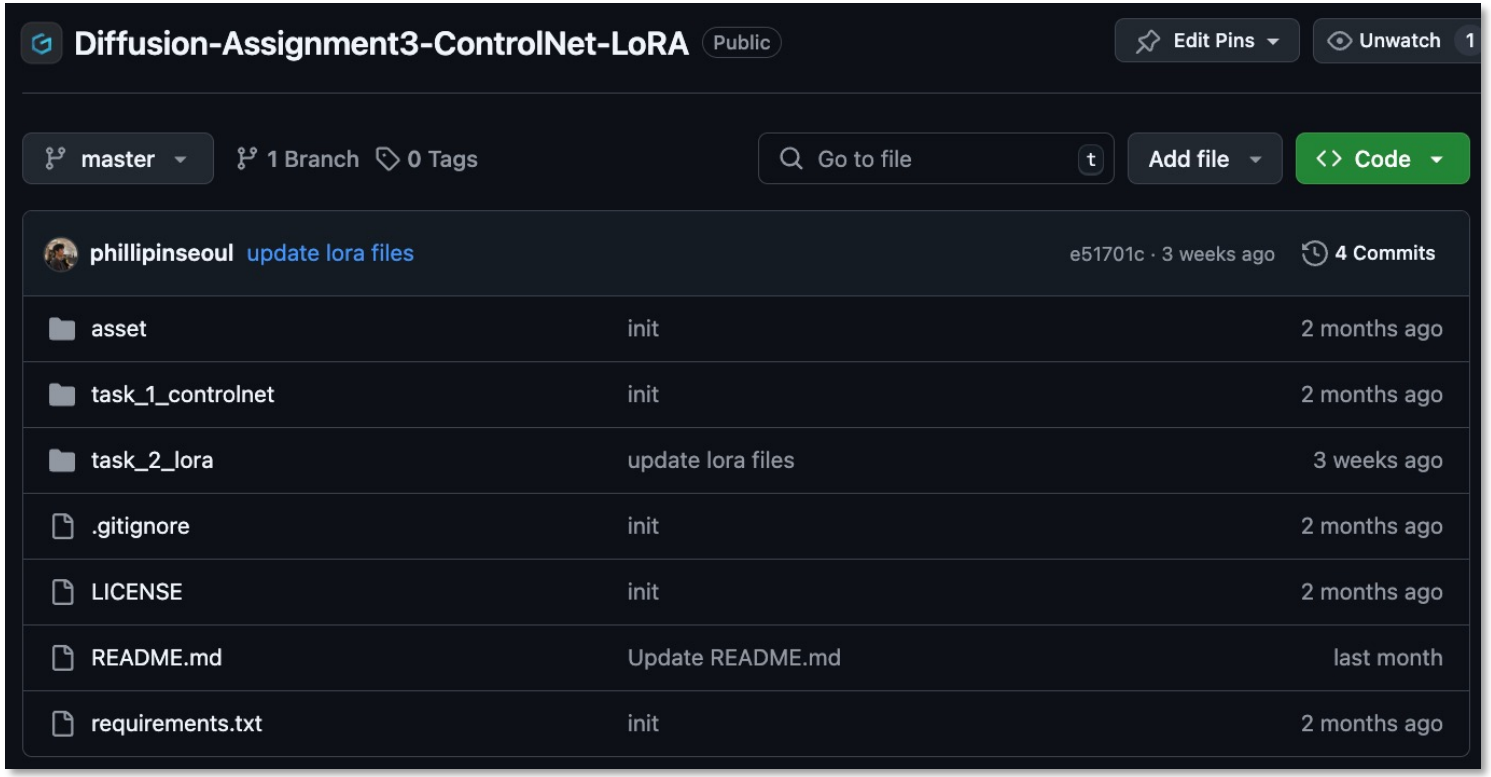
Introduction

In Task 2, you will train custom **LoRA** models with a given library (No implementation).

LoRA: Low-Rank Adaptation of Large Language Models, Hu *et al.*, ICLR 2022

Introduction

The skeleton code and instructions are available at:
https://github.com/KAIST-Visual-AI-Group/Diffusion-Assignment3-ControlNet-LoRA



Important Notes

- All programming assignments are due **two weeks** after the assignment session.
- Late submission will incur **20% penalty** for **each** late day!
- Please carefully check the README of each assignment.
- Missing items in your submission will also incur penalties.

Overview

0. Introduction to 🤗 **Hugging Face** and 🧨 **Diffusers**

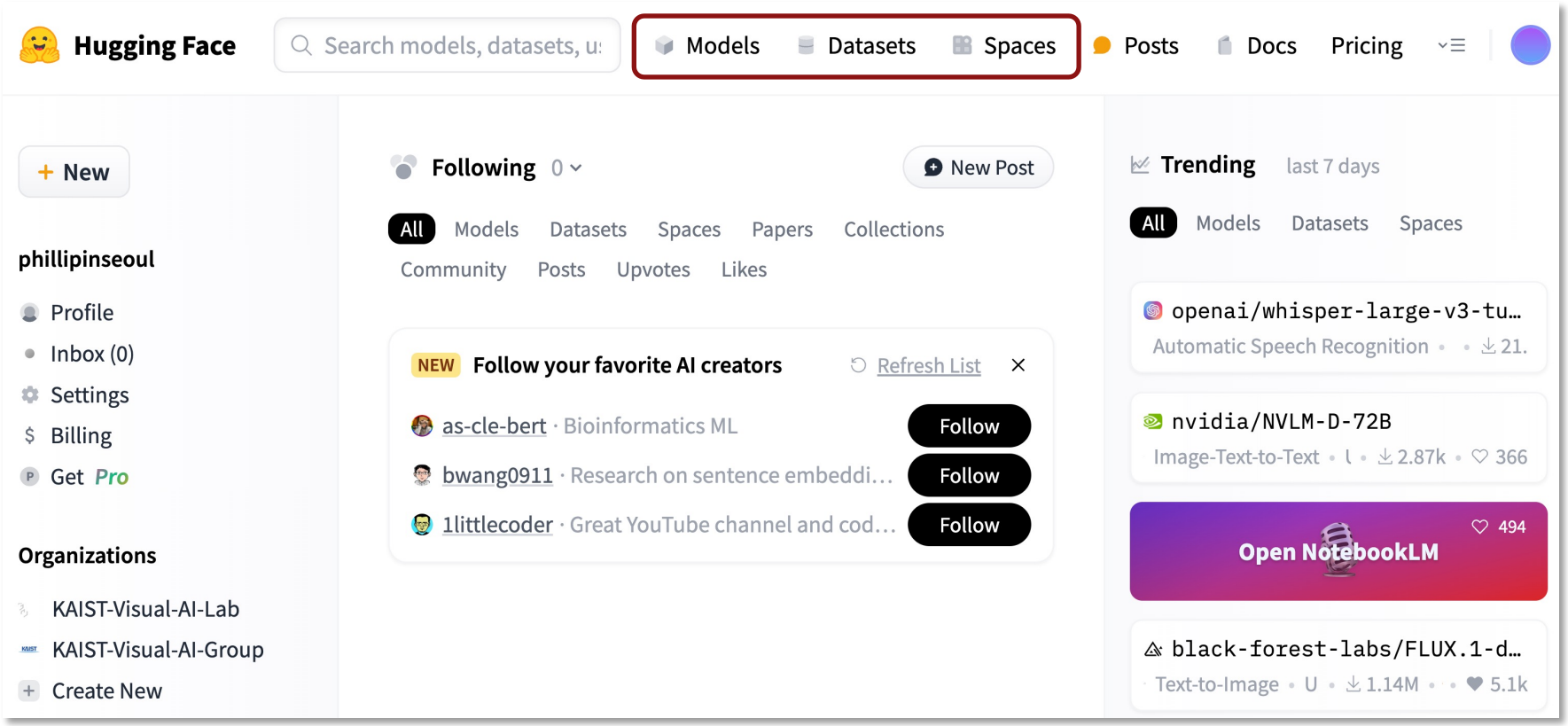
1. [**Task 1**] Implementing **ControlNet**

2. [**Task 2**] Training Custom **LoRA** Models

Introduction to Hugging Face and Diffusers

Hugging Face

An open-source platform that serves as a hub for machine learning applications.



Diffusers Library D🧨ffusers

Python library for state-of-the-art pre-trained diffusion models.

Popular Tasks & Pipelines

Task	Pipeline	🤗 Hub
Unconditional Image Generation	DDPM	google/ddpm-ema-church-256
Text-to-Image	Stable Diffusion Text-to-Image	runwayml/stable-diffusion-v1-5
Text-to-Image	unclip	kakaobrain/karlo-v1-alpha
Text-to-Image	DeepFloyd IF	DeepFloyd/IF-I-XL-v1.0
Text-to-Image	Kandinsky	kandinsky-community/kandinsky-2-2-decoder
Text-guided Image-to-Image	Controlnet	lllyasviel/sd-controlnet-canny
Text-guided Image-to-Image	Instruct Pix2Pix	timbrooks/instruct-pix2pix
Text-guided Image-to-Image	Stable Diffusion Image-to-Image	runwayml/stable-diffusion-v1-5
Text-guided Image Inpainting	Stable Diffusion Inpaint	runwayml/stable-diffusion-inpainting
Image Variation	Stable Diffusion Image Variation	lambdalabs/sd-image-variations-diffusers
Super Resolution	Stable Diffusion Upscale	stabilityai/stable-diffusion-x4-upscaler
Super Resolution	Stable Diffusion Latent Upscale	stabilityai/sd-x2-latent-upscaler

e.g. Loading Stable Diffusion from diffusers

```python
import torch
from diffusers import StableDiffusionPipeline

model_id = "runwayml/stable-diffusion-v1-5"
pipe = StableDiffusionPipeline.from_pretrained(
    model_id,
    torch_dtype=torch.float16
)
pipe = pipe.to("cuda")

prompt = "a photo of an astronaut riding a horse on mars"
image = pipe(prompt).images[0]

image.save("astronaut_rides_horse.png")
```

What to Do: Task 0

You need access to Hugging Face to proceed with Task 1 and 2:

1. Sign into Hugging Face.
2. Obtain your access token at https://huggingface.co/settings/tokens.
3. From your terminal, log into Hugging Face using

```
$ huggingface-cli login
```

   and enter your Access Token.

What to Do: Task 0

4. **To check the access to Hugging Face,** download Stable Diffusion from Hugging Face and generate an image with it:

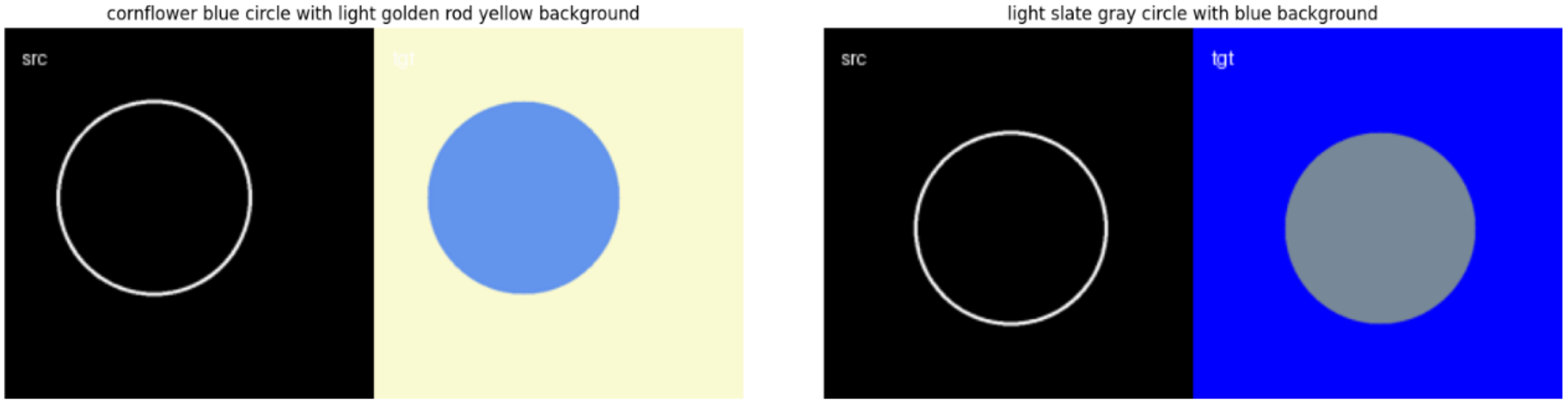
```python
import torch
from diffusers import StableDiffusionPipeline

model_id = "CompVis/stable-diffusion-v1-4"
device = "cuda"


pipe = StableDiffusionPipeline.from_pretrained(model_id, torch_dtype=torch.float16)
pipe = pipe.to(device)

prompt = "a photo of an astronaut riding a horse on mars"
image = pipe(prompt).images[0]

image.save("astronaut_rides_horse.png")
```

[Task 1]
Implementing ControlNet

What to Do: Task 1

Goal: Train ControlNet on Fill50K dataset.

Fill50K

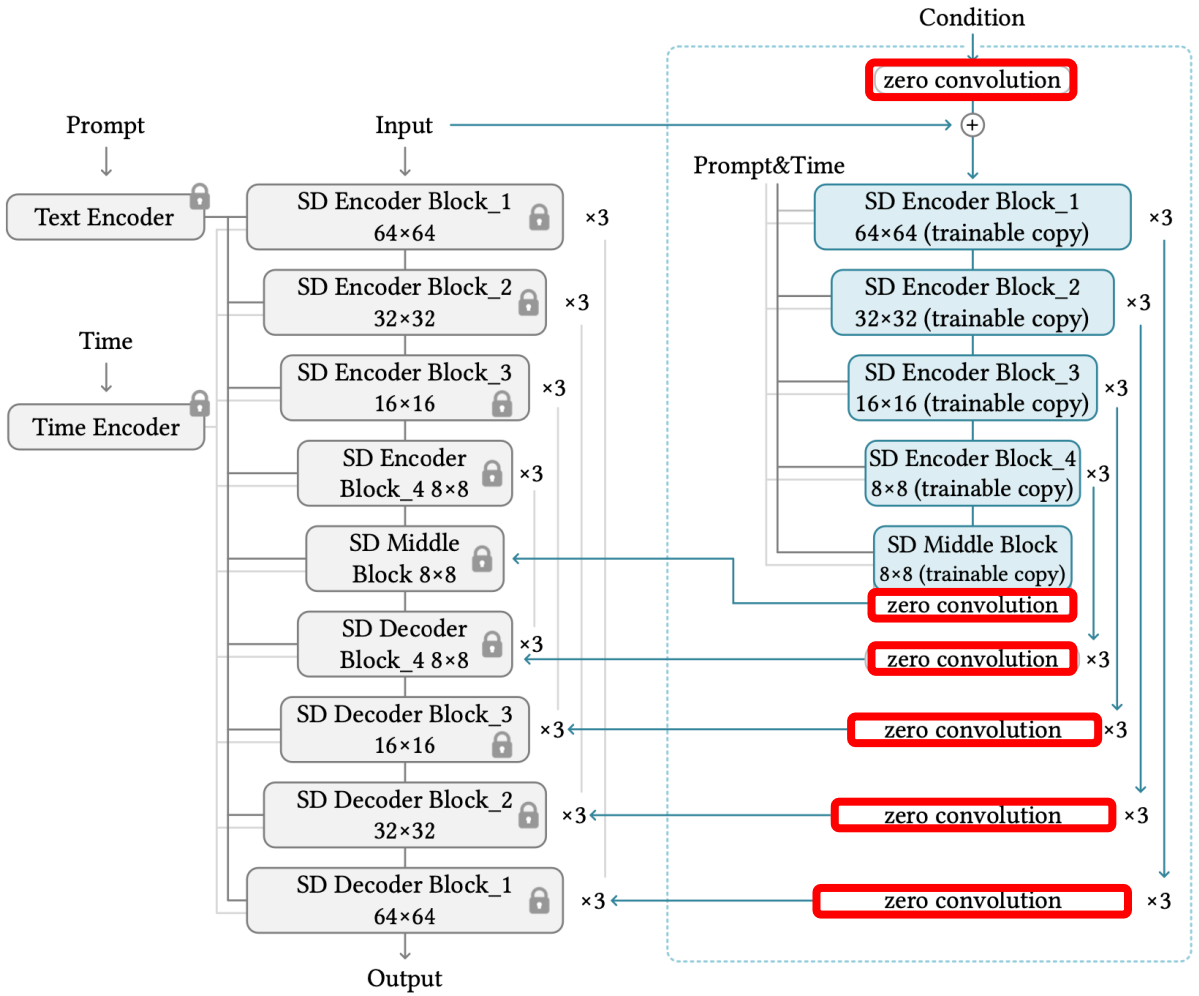


What to Do: Task 1

Implement Zero-Convolution.

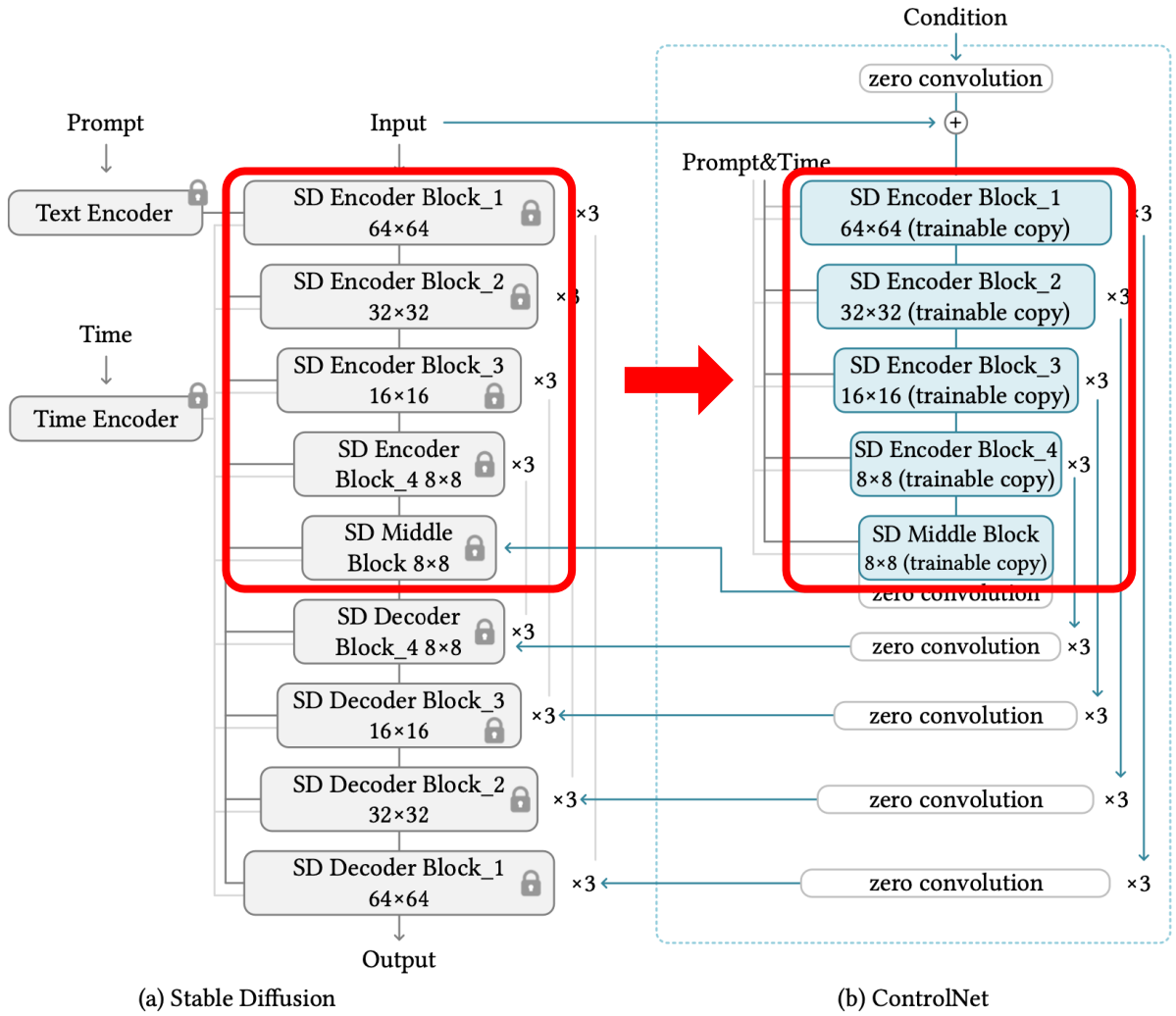
```python
def zero_convolution(
    in_channels, out_channels,
    kernel_size=1, stride=1, padding=0
):
    """
    Return a 'zero-convolution layer'
    (Initialized weight & bias as zeros.)
    """
    ######## TODO (1) ########
    # DO NOT change the code outside this part.
    # Return a zero-convolution layer,
    # with the weight & bias initialized as zeros.
    module = None
    ######## TODO (1) ########

    return module
```

`task_1_controlnet/diffusion/controlnet.py`



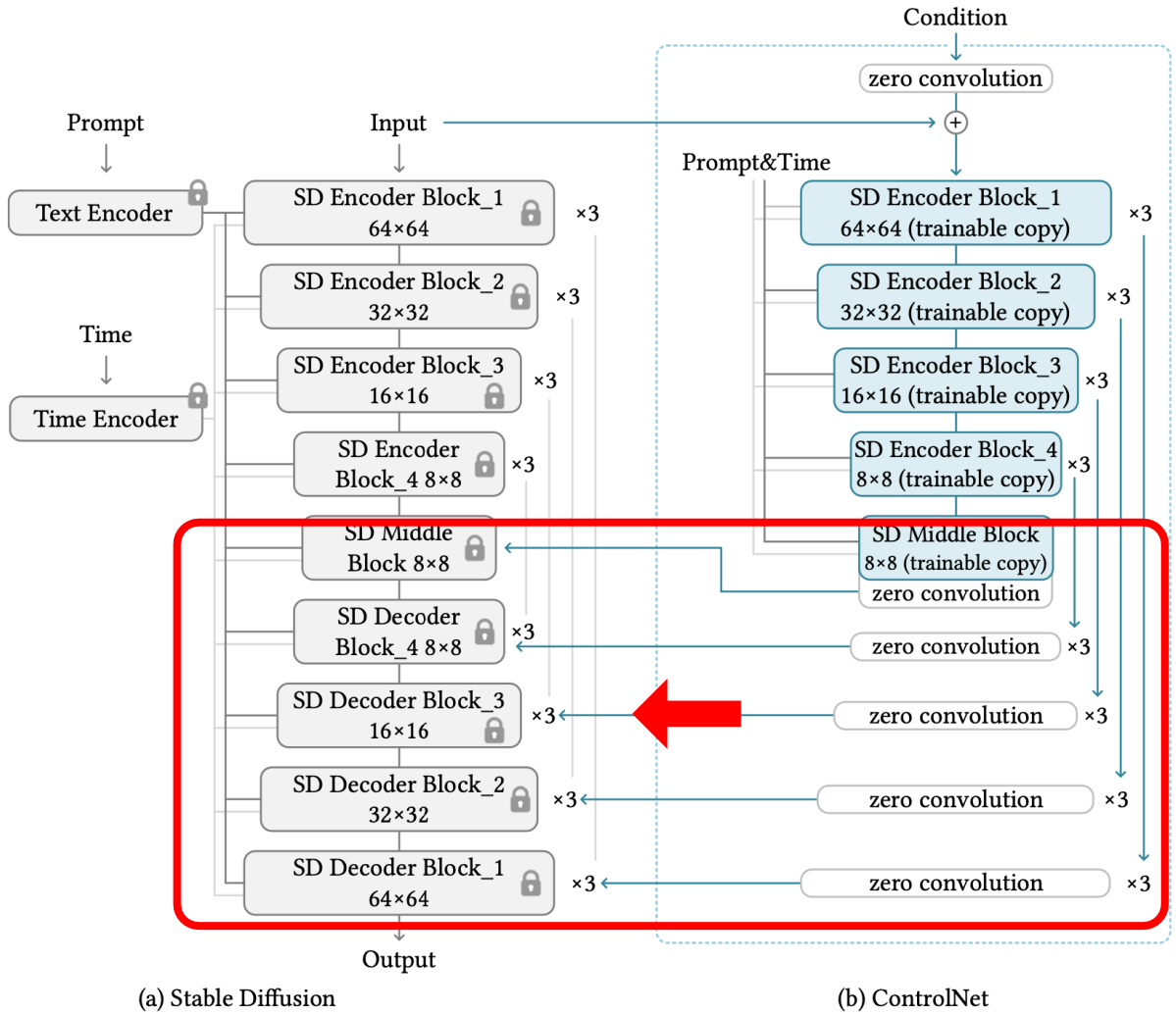
What to Do: Task 1

Initialize ControlNet using a pre-trained U-Net from Stable Diffusion.



What to Do: Task 1

- Apply Zero-Convolution to the residual features of each ControlNet block.
- Integrate outputs from ControlNet blocks into U-Net decoder of Stable Diffusion.

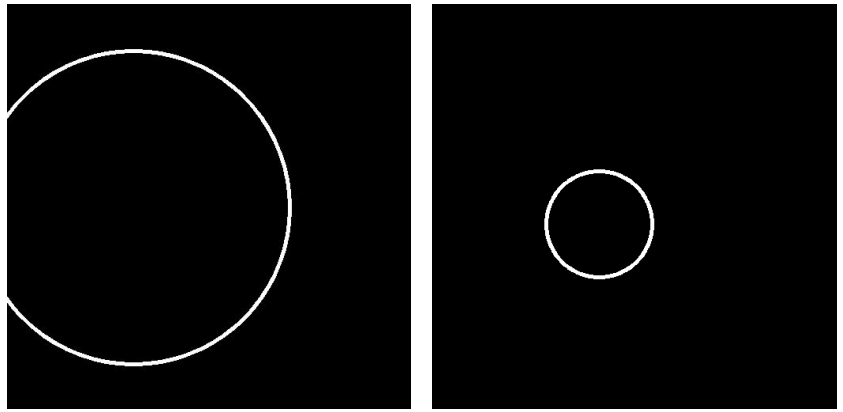


What to Do: Task 1

- Train ControlNet using: `$ sh train.sh`

- Generate images with 5 different conditions (./data/test_conditions) and text prompts from (./data/text_prompts.json).

Test conditions



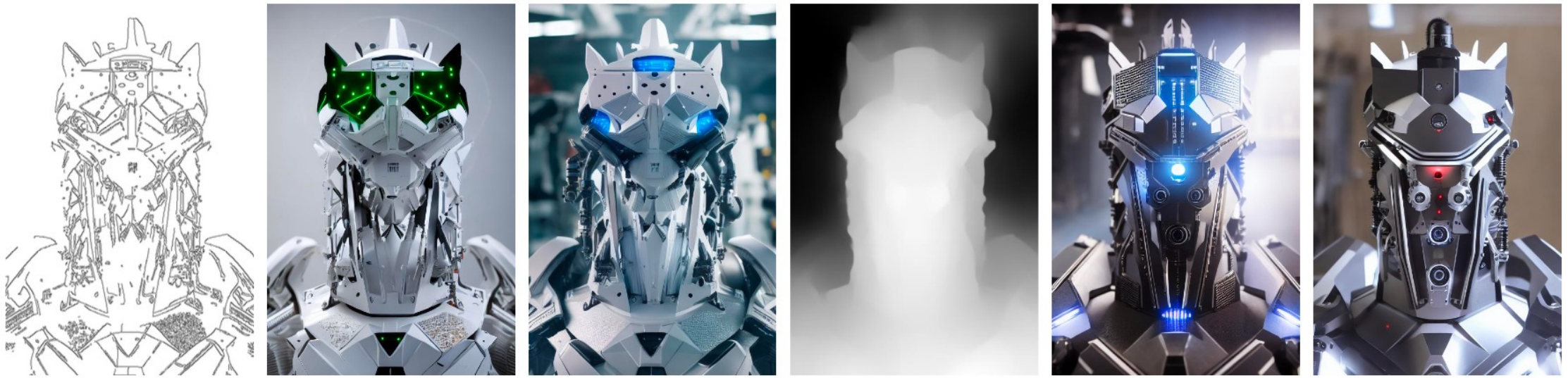
Test prompts

```
{
    "0": "pale golden rod circle with old lace background",
    "1": "sea green circle with a light cyan background",
    "2": "deep sky blue circle with a light yellow background",
    "3": "rosy brown circle with a misty rose background",
    "4": "forest green circle with an antique brown background"
}
```

What to Do: Task 1 (Optional)

Train ControlNet on **a different type of condition** (e.g. depth map, canny edge, sketch, etc.).

You can use either an **open-source dataset** or your own **custom data**.



Canny Edge

Depth (midas)

[Task 2]
Training Custom LoRA Models

What to Do: Task 2

Goal: Train custom LoRA models on three different datasets.

What to Do: Task 2

No implementation required!

The main objective of this task is to:

- **Gain hands-on experience** on customizing diffusion models using LoRA,
- **Explore creative use-cases** of state-of-the-art diffusion models.

What to Do: Task 2

[**Task 2-1**] Train LoRA on a specific **style**.

You can either use an **open-source dataset** and train with

```
$ sh scripts/train_lora.sh
```

Or create a **custom dataset** and train with

```
$ sh scripts/train_lora_custom.sh
```

What to Do: Task 2

[**Task 2-1**] Train LoRA on a specific **style**.

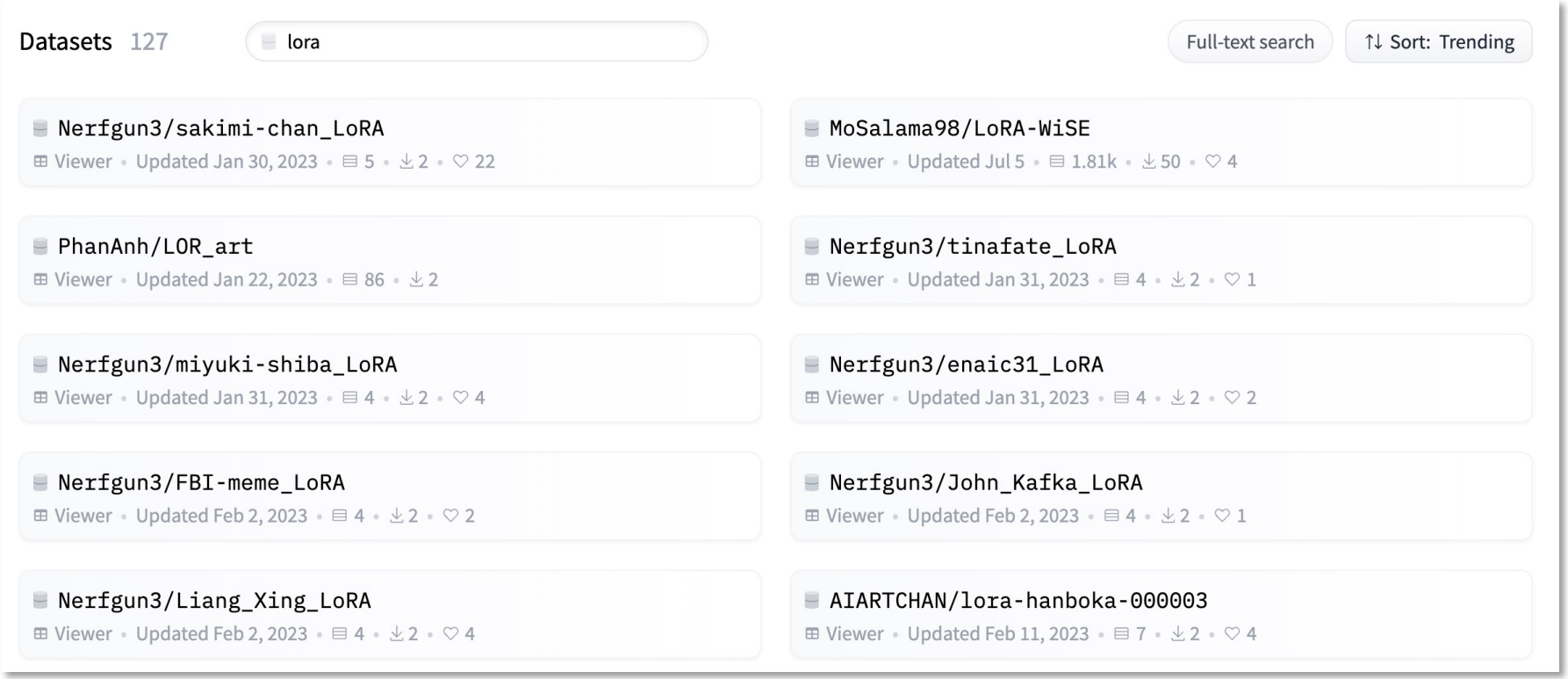
In either case, just simply set the path to the dataset by

```
$ export DATASET_NAME="$PATH_TO_DATASET"
```

in the given script file.

What to Do: Task 2

TIP: You can find many open-source LoRA training datasets on Hugging Face.



https://huggingface.co/datasets?modality=modality:image&size_categories=or:%28size_categories:n%3C1K,size_categories:1K%3Cn%3C10K%29&sort=trending&search=lora

What to Do: Task 2

[**Task 2-2**] Train LoRA on a specific **identity** using DreamBooth + LoRA.

Run `$ sh scripts/train_dreambooth_lora.sh` for training.

Generated with GPT4

What to Do: Task 2

Inference Code: `lora_inference.ipynb`

Load Stable Diffusion

In [ ]:

```
pipe = StableDiffusionPipeline.from_pretrained(
    "CompVis/stable-diffusion-v1-4",
    torch_dtype=torch.float16
)
print("[INFO] Successfully loaded Stable Diffusion!")
```

Load LoRA weights

Change to your LoRA weights!

In [22]:

```
# lora_path = "./runs/sd-naruto-model-lora"
lora_path = "./runs/artistic_custom"
# lora_path = "./runs/dreambooth_cat"
# lora_path = None  # if not using LoRA

if lora_path is not None:
    pipe.load_lora_weights(lora_path)
    print("[INFO] Successfully loaded LoRA weights!")

pipe = pipe.to(device)
```

```
[INFO] Successfully loaded LoRA weights!
```

Inference

Change prompt and seed for diverse outputs!

In [27]:

```
prompt = "a man with sunglasses"

seed = 10

seed_everything(seed)

image = pipe(
    prompt,
    num_inference_steps=30,
    guidance_scale=7.5
).images[0]

image
```

```
100%|██████████| 30/30 [00:01<00:00, 22.53it/s]
```

What to Submit

Submit a zip file named {NAME}_{STUDENT_ID}.zip that includes:

- Code for **Task 1**
  - Include your code inside `task_1_controlnet/` directory.
- LoRA Checkpoints for **Task 2**
  - `pytorch_lora_weights.safetensors`
- PDF Report (Max. 2 pages)

What to Submit

The PDF report should include:

[**Task 1**]

- 5 different condition inputs, corresponding text prompts, and the generated images.
- A brief analysis of the results for each condition.
- **(Optional Task 1-1)** 5 different condition inputs, corresponding text prompts, and the generated images.
- **(Optional Task 1-1)** A brief explanation about the training dataset and the training results.

What to Submit

The PDF report should include:

[Task 2]

- (Task 2-1) Decription on the dataset used, including its source.
- (Task 2-1) Visualization of training images and and generated image with the corresponding text prompts.
- (Task 2-2) Decription on the dataset used, including its source.
- (Task 2-2) Visualization of training images and and generated image with the corresponding text prompts.

What to Submit

The zip file should look like:

```
.
├── 2024XXXX.pdf          <-- report (max. 2 pages)
├── task_1_controlnet     <-- code for Task 1
├── lora_1                <-- checkpoints for Task 2
│   └── pytorch_lora_weights.safetensors
├── lora_2
│   └── pytorch_lora_weights.safetensors
└── lora_3
    └── pytorch_lora_weights.safetensors
```

Grading

The scores for each task are detailed as follows:

Task 1 (10pt):

- [0pt] Either the code or the report is not submitted.
- [5pt] Generated images do not align with the input conditions.
- [10pt] Generated images accurately align with the input conditions.

Grading

The scores for each task are detailed as follows:

Task 2 (10pt):

- [0pt] The report is not submitted.
- [5pt] Outputs of either one of the LoRAs do not align with the training data.
- [10pt] Outputs of both LoRAs accurately align with the training data.

! Disclaimer !

Ethical usage of AI models is important:

1. You can freely refer to any online source code when working on Assignment 3. BUT we prohibit simply copying & pasting existing code. The aim for the assignment is to get hands-on experience on how the code works for diffusion model applications.

2. You MUST NOT use personalization techniques for **unethical purposes**, such as generating content that includes nudity, violence of specific identities. It is your responsibility to ensure that these methods are is applied ethically.

! Disclaimer !

Ethical usage of AI models is important:

3. When using open-source datasets, use MUST give credit to the original providers. If you are using such datasets for training ControlNet or LoRA, remember to clearly cite the original source of data.

Demo

Thank You