```
from google.colab import files
files.upload()  # Select your kaggle.json file here
```

```
Dosyaları Seç Seçilen dosya yok    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username":"bozkuya","key":"1db5dafb1badc90297e987b10a46f3bd"}'}
```

```
!mkdir -p ~/.kaggle
!mv kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d denislukovnikov/ffhq256-images-only
```

```
Downloading ffhq256-images-only.zip to /content
100% 6.91G/6.91G [06:23<00:00, 22.6MB/s]
100% 6.91G/6.91G [06:23<00:00, 19.3MB/s]
```

```
!unzip ffhq256-images-only.zip -d data/
```

```
Görüntülenen çıkış son 5000 satıra kısaltıldı.
  inflating: data/ffhq256/65000.png
  inflating: data/ffhq256/65001.png
  inflating: data/ffhq256/65002.png
  inflating: data/ffhq256/65003.png
  inflating: data/ffhq256/65004.png
  inflating: data/ffhq256/65005.png
  inflating: data/ffhq256/65006.png
  inflating: data/ffhq256/65007.png
  inflating: data/ffhq256/65008.png
  inflating: data/ffhq256/65009.png
  inflating: data/ffhq256/65010.png
  inflating: data/ffhq256/65011.png
  inflating: data/ffhq256/65012.png
  inflating: data/ffhq256/65013.png
  inflating: data/ffhq256/65014.png
  inflating: data/ffhq256/65015.png
  inflating: data/ffhq256/65016.png
  inflating: data/ffhq256/65017.png
  inflating: data/ffhq256/65018.png
  inflating: data/ffhq256/65019.png
  inflating: data/ffhq256/65020.png
  inflating: data/ffhq256/65021.png
  inflating: data/ffhq256/65022.png
  inflating: data/ffhq256/65023.png
  inflating: data/ffhq256/65024.png
  inflating: data/ffhq256/65025.png
  inflating: data/ffhq256/65026.png
  inflating: data/ffhq256/65027.png
  inflating: data/ffhq256/65028.png
  inflating: data/ffhq256/65029.png
  inflating: data/ffhq256/65030.png
  inflating: data/ffhq256/65031.png
  inflating: data/ffhq256/65032.png
  inflating: data/ffhq256/65033.png
  inflating: data/ffhq256/65034.png
  inflating: data/ffhq256/65035.png
  inflating: data/ffhq256/65036.png
  inflating: data/ffhq256/65037.png
  inflating: data/ffhq256/65038.png
  inflating: data/ffhq256/65039.png
  inflating: data/ffhq256/65040.png
  inflating: data/ffhq256/65041.png
  inflating: data/ffhq256/65042.png
  inflating: data/ffhq256/65043.png
  inflating: data/ffhq256/65044.png
  inflating: data/ffhq256/65045.png
  inflating: data/ffhq256/65046.png
  inflating: data/ffhq256/65047.png
  inflating: data/ffhq256/65048.png
  inflating: data/ffhq256/65049.png
  inflating: data/ffhq256/65050.png
  inflating: data/ffhq256/65051.png
  inflating: data/ffhq256/65052.png
  inflating: data/ffhq256/65053.png
```

```
       inflating: data/ffhq256/65054.png
       inflating: data/ffhq256/65055.png
       inflating: data/ffhq256/65056.png
```

```
%%capture
!pip install diffusers[training]==0.11.1
```

```
from huggingface_hub import notebook_login

notebook_login()
```

```
Token is valid (permission: write).

Your token has been saved in your configured git credential helpers (store).

Your token has been saved to /root/.cache/huggingface/token

Login successful
```

```
%%capture
!sudo apt -qq install git-lfs
!git config --global credential.helper store
```

```
from dataclasses import dataclass

@dataclass
class TrainingConfig:
    image_size = 32  # the generated image resolution
    train_batch_size = 128
    eval_batch_size = 128  # how many images to sample during evaluation
    num_epochs = 20
    gradient_accumulation_steps = 1
    learning_rate = 1e-4
    lr_warmup_steps = 500
    save_image_epochs = 10
    save_model_epochs = 10
    mixed_precision = 'fp16'  # `no` for float32, `fp16` for automatic mixed precision
    output_dir = 'Yasincan/ffhq'  # the model namy locally and on the HF Hub

    push_to_hub = True  # whether to upload the saved model to the HF Hub
    hub_private_repo = False
    overwrite_output_dir = True  # overwrite the old model when re-running the notebook
    seed = 0

config = TrainingConfig()
```

```
from datasets import load_dataset
dataset = load_dataset("imagefolder", data_dir="/content/data/ffhq256")
```

```
Resolving data files: 100%                                    70000/70000 [00:01<00:00, 112785.32it/s]
```

```
config.dataset_name = "/content/data/ffhq256"
```

```
dataset = load_dataset(config.dataset_name, split="train")
```

```
Resolving data files: 100%                                    70000/70000 [00:00<00:00, 114977.46it/s]
```

```
dataset
```

```
Dataset({
    features: ['image'],
    num_rows: 70000
})
```

```
import matplotlib.pyplot as plt

fig, axs = plt.subplots(1, 4, figsize=(16, 4))
for i, image in enumerate(dataset[:4]["image"]):
    axs[i].imshow(image)
    axs[i].set_axis_off()
fig.show()
```

```
from torchvision import transforms

preprocess = transforms.Compose(
    [
        transforms.Resize((config.image_size, config.image_size)),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize([0.5], [0.5]),
        transforms.Resize((32, 32)),
    ]
)


def transform(examples):
    images = [preprocess(image.convert("RGB")) for image in examples["image"]]
    return {"images": images}

dataset.set_transform(transform)


fig, axs = plt.subplots(1, 4, figsize=(16, 4))
for i, image in enumerate(dataset[:4]["images"]):
    axs[i].imshow(image.permute(1, 2, 0).numpy() / 2 + 0.5)
    axs[i].set_axis_off()
fig.show()
```

```
import torch

train_dataloader = torch.utils.data.DataLoader(dataset, batch_size=config.train_batch_size, shuffle=True)


from diffusers import UNet2DModel


model = UNet2DModel(
    sample_size=32,  # Given the new resolution
    in_channels=3,
    out_channels=3,
    layers_per_block=1,  # Reduce the number of ResNet layers per block
    block_out_channels=(64, 128, 256),  # Reduced number of channels and blocks
    down_block_types=(
        "DownBlock2D",
        "DownBlock2D",
        "DownBlock2D",
```

```
    ),
    up_block_types=(
        "UpBlock2D",
        "UpBlock2D",
        "UpBlock2D",
      ),
)



sample_image = dataset[0]['images'].unsqueeze(0)
print('Input shape:', sample_image.shape)
```

```
Input shape: torch.Size([1, 3, 32, 32])
```

```
print('Output shape:', model(sample_image, timestep=0).sample.shape)
```

```
Output shape: torch.Size([1, 3, 32, 32])
```

```
from diffusers import DDPMScheduler

noise_scheduler = DDPMScheduler(num_train_timesteps=1000)


import torch
from PIL import Image

noise = torch.randn(sample_image.shape)
timesteps = torch.LongTensor([50])
noisy_image = noise_scheduler.add_noise(sample_image, noise, timesteps)

Image.fromarray(((noisy_image.permute(0, 2, 3, 1) + 1.0) * 127.5).type(torch.uint8).numpy()[0])
```

```
import torch.nn.functional as F

noise_pred = model(noisy_image, timesteps).sample
loss = F.mse_loss(noise_pred, noise)


optimizer = torch.optim.AdamW(model.parameters(), lr=config.learning_rate)


from diffusers.optimization import get_cosine_schedule_with_warmup

lr_scheduler = get_cosine_schedule_with_warmup(
    optimizer=optimizer,
    num_warmup_steps=config.lr_warmup_steps,
    num_training_steps=(len(train_dataloader) * config.num_epochs),
)


from diffusers import DDPMPipeline

import math

def make_grid(images, rows, cols):
    w, h = images[0].size
    grid = Image.new('RGB', size=(cols*w, rows*h))
    for i, image in enumerate(images):
        grid.paste(image, box=(i%cols*w, i//cols*h))
    return grid

def evaluate(config, epoch, pipeline):
    # Sample some images from random noise (this is the backward diffusion process).
    # The default pipeline output type is `List[PIL.Image]`
    images = pipeline(
        batch_size = config.eval_batch_size,
        generator=torch.manual_seed(config.seed),
    ).images

    # Make a grid out of the images
    image_grid = make_grid(images, rows=4, cols=4)

    # Save the images
```

```
    test_dir = os.path.join(config.output_dir, "samples")
    os.makedirs(test_dir, exist_ok=True)
    image_grid.save(f"{test_dir}/{epoch:04d}.png")


from accelerate import Accelerator
from huggingface_hub import HfFolder, Repository, whoami

from tqdm.auto import tqdm
from pathlib import Path
import os

def get_full_repo_name(model_id: str, organization: str = None, token: str = None):
    if token is None:
        token = HfFolder.get_token()
    if organization is None:
        username = whoami(token)["name"]
        return f"{username}/{model_id}"
    else:
        return f"{organization}/{model_id}"

def train_loop(config, model, noise_scheduler, optimizer, train_dataloader, lr_scheduler):
    # Initialize accelerator and tensorboard logging
    accelerator = Accelerator(
    mixed_precision=config.mixed_precision,
    gradient_accumulation_steps=config.gradient_accumulation_steps)

    if accelerator.is_main_process:
        if config.push_to_hub:
            repo_name = get_full_repo_name(Path(config.output_dir).name)
            repo = Repository(config.output_dir, clone_from=repo_name)
        elif config.output_dir is not None:
            os.makedirs(config.output_dir, exist_ok=True)
        accelerator.init_trackers("train_example")

    # Prepare everything
    # There is no specific order to remember, you just need to unpack the
    # objects in the same order you gave them to the prepare method.
    model, optimizer, train_dataloader, lr_scheduler = accelerator.prepare(
        model, optimizer, train_dataloader, lr_scheduler
    )

    global_step = 0

    # Now you train the model
    for epoch in range(config.num_epochs):
        progress_bar = tqdm(total=len(train_dataloader), disable=not accelerator.is_local_main_process)
        progress_bar.set_description(f"Epoch {epoch}")

        for step, batch in enumerate(train_dataloader):
            clean_images = batch['images']
            # Sample noise to add to the images
            noise = torch.randn(clean_images.shape).to(clean_images.device)
            bs = clean_images.shape[0]

            # Sample a random timestep for each image
            timesteps = torch.randint(0, noise_scheduler.num_train_timesteps, (bs,), device=clean_images.device).long()

            # Add noise to the clean images according to the noise magnitude at each timestep
            # (this is the forward diffusion process)
            noisy_images = noise_scheduler.add_noise(clean_images, noise, timesteps)

            with accelerator.accumulate(model):
                # Predict the noise residual
                noise_pred = model(noisy_images, timesteps, return_dict=False)[0]
                loss = F.mse_loss(noise_pred, noise)
                accelerator.backward(loss)

                accelerator.clip_grad_norm_(model.parameters(), 1.0)
                optimizer.step()
                lr_scheduler.step()
                optimizer.zero_grad()

            progress_bar.update(1)
            logs = {"loss": loss.detach().item(), "lr": lr_scheduler.get_last_lr()[0], "step": global_step}
            progress_bar.set_postfix(**logs)
            accelerator.log(logs, step=global_step)
```

```
            global_step += 1

        # After each epoch you optionally sample some demo images with evaluate() and save the model
        if accelerator.is_main_process:
            pipeline = DDPMPipeline(unet=accelerator.unwrap_model(model), scheduler=noise_scheduler)

            if (epoch + 1) % config.save_image_epochs == 0 or epoch == config.num_epochs - 1:
                evaluate(config, epoch, pipeline)

            if (epoch + 1) % config.save_model_epochs == 0 or epoch == config.num_epochs - 1:
                if config.push_to_hub:
                    repo.push_to_hub(commit_message=f"Epoch {epoch}", blocking=True)
                else:
                    pipeline.save_pretrained(config.output_dir)
```

```
!pip install --upgrade accelerate
```

```
Requirement already satisfied: accelerate in /usr/local/lib/python3.10/dist-packages (0.23.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from accelerate) (1.23.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from accelerate) (23.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.10/dist-packages (from accelerate) (6.0.1)
Requirement already satisfied: torch>=1.10.0 in /usr/local/lib/python3.10/dist-packages (from accelerate) (2.0.1+cu118)
Requirement already satisfied: huggingface-hub in /usr/local/lib/python3.10/dist-packages (from accelerate) (0.17.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (3.12.2)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (4.5.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (3.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (3.1.2)
Requirement already satisfied: triton==2.0.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (2.0.0)
Requirement already satisfied: cmake in /usr/local/lib/python3.10/dist-packages (from triton==2.0.0->torch>=1.10.0->accelerate) (3.27.4
Requirement already satisfied: lit in /usr/local/lib/python3.10/dist-packages (from triton==2.0.0->torch>=1.10.0->accelerate) (16.0.6)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from huggingface-hub->accelerate) (2023.6.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from huggingface-hub->accelerate) (2.31.0)
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub->accelerate) (4.66.1)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.10.0->accelerate) (2.1
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub->acc
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub->accelerate) (3.
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub->accelerat
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub->accelerat
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.10.0->accelerate) (1.3.0)
```

```
from accelerate import notebook_launcher
args = (config, model, noise_scheduler, optimizer, train_dataloader, lr_scheduler)
notebook_launcher(train_loop, args, num_processes=1)
```

```
Launching training on one GPU.
/content/Yasincan/ffhq is already a clone of https://huggingface.co/Yasincan/ffhq. Make sure you pull the latest changes with `repo.git
WARNING:huggingface_hub.repository:/content/Yasincan/ffhq is already a clone of https://huggingface.co/Yasincan/ffhq. Make sure you pul
```

Epoch 0: 100% 547/547 [1:10:12<00:00, 1.31it/s, loss=0.0525, lr=0.0001, step=546]

Epoch 1: 100% 547/547 [1:01:44<00:00, 1.53it/s, loss=0.038, lr=9.92e-5, step=1093]

Epoch 2: 100% 547/547 [53:47<00:00, 1.49it/s, loss=0.0449, lr=9.71e-5, step=1640]

Epoch 3: 100% 547/547 [45:56<00:00, 1.47it/s, loss=0.0261, lr=9.37e-5, step=2187]

Epoch 4: 100% 547/547 [38:05<00:00, 1.57it/s, loss=0.057, lr=8.91e-5, step=2734]

Epoch 5: 100% 547/547 [30:19<00:00, 1.56it/s, loss=0.037, lr=8.35e-5, step=3281]

Epoch 6: 100% 547/547 [22:29<00:00, 1.58it/s, loss=0.0333, lr=7.69e-5, step=3828]

Epoch 7: 100% 547/547 [14:43<00:00, 1.56it/s, loss=0.034, lr=6.97e-5, step=4375]

Epoch 8: 100% 547/547 [42:43<00:00, 1.46it/s, loss=0.0409, lr=6.19e-5, step=4922]

Epoch 9: 100% 547/547 [34:52<00:00, 1.57it/s, loss=0.0348, lr=5.38e-5, step=5469]

100% 1000/1000 [01:02<00:00, 15.72it/s]

```
Adding files tracked by Git LFS: ['samples/0009.png']. This may take a bit of time if the files are large.
WARNING:huggingface_hub.repository:Adding files tracked by Git LFS: ['samples/0009.png']. This may take a bit of time if the files are
```

Upload file samples/0009.png: 100% 35.6k/35.6k [00:02<00:00, 1.80kB/s]

```
To https://huggingface.co/Yasincan/ffhq
   e59f063..cd3dded  main -> main

WARNING:huggingface_hub.repository:To https://huggingface.co/Yasincan/ffhq
```

```
import glob

sample_images = sorted(glob.glob(f"{config.output_dir}/samples/*.png"))
Image.open(sample_images[-1])
```

```
!wget -q https://github.com/ShivamShrirao/diffusers/raw/main/examples/dreambooth/train_dreambooth.py
!wget -q https://github.com/ShivamShrirao/diffusers/raw/main/scripts/convert_diffusers_to_original_stable_diffusion.py
%pip install -qq git+https://github.com/ShivamShrirao/diffusers
%pip install -q -U --pre triton
%pip install -q accelerate transformers ftfy bitsandbytes==0.35.0 gradio natsort safetensors xformers
```

```
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
  Building wheel for diffusers (pyproject.toml) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 89.2/89.2 MB 10.3 MB/s eta 0:00:00
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source
torch 2.0.1+cu118 requires triton==2.0.0; platform_system == "Linux" and platform_machine == "x86_64", but you have triton 2.1.0 which
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.6/7.6 MB 42.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 53.1/53.1 kB 6.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.5/62.5 MB 12.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20.2/20.2 MB 78.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 73.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 167.0/167.0 MB 7.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 105.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.2/66.2 kB 8.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 298.2/298.2 kB 30.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.7/75.7 kB 9.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 138.7/138.7 kB 17.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.7/45.7 kB 5.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 59.5/59.5 kB 7.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 129.9/129.9 kB 17.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 63.3/63.3 MB 13.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 7.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.0/67.0 kB 8.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 76.0/76.0 kB 6.7 MB/s eta 0:00:00
  Building wheel for ffmpy (setup.py) ... done
```

save_to_gdrive: ☐

```
save_to_gdrive = False #@param {type:"boolean"}
if save_to_gdrive:
    from google.colab import drive
    drive.mount('/content/drive')

#@markdown Name/Path of the initial model.
MODEL_NAME = "Yasincan/ffhq" #@param {type:"string"}

#@markdown Enter the directory name to save model at.
OUTPUT_DIR = "lebron_diffusion_weights" #@param {type:"string"}
if save_to_gdrive:
    OUTPUT_DIR = "/content/drive/MyDrive/" + OUTPUT_DIR
else:
    OUTPUT_DIR = "/content/" + OUTPUT_DIR

print(f"[*] Weights will be saved at {OUTPUT_DIR}")

!mkdir -p $OUTPUT_DIR

concepts_list = [
    {
        "instance_prompt":      "photo of Lebron James",
        "class_prompt":         "photo of a person",
        "instance_data_dir":    "/content/data/lebron",
        "class_data_dir":       "/content/data/ffhq256"
    }
]

# `class_data_dir` contains regularization images
import json
import os
for c in concepts_list:
    os.makedirs(c["instance_data_dir"], exist_ok=True)

with open("concepts_list.json", "w") as f:
    json.dump(concepts_list, f, indent=4)

#@markdown Upload your images by running this cell.
import os
from google.colab import files
import shutil

for c in concepts_list:
    print(f"Uploading instance images for `{c['instance_prompt']}`")
    uploaded = files.upload()
    for filename in uploaded.keys():
        dst_path = os.path.join(c['instance_data_dir'], filename)
        shutil.move(filename, dst_path)
```

Name/Path of the initial model.

MODEL_NAME: " Yasincan/ffhq "

Enter the directory name to save model at.

OUTPUT_DIR: " lebron_diffusion_weights "

Upload your images by running this cell.

```
[*] Weights will be saved at /content/lebron_diffusion_weights
Uploading instance images for `photo of Lebron James`
```

Dosyaları Seç 11 dosya

- **1.png**(image/png) - 378576 bytes, last modified: 22.09.2023 - 100% done
- **2.png**(image/png) - 474353 bytes, last modified: 22.09.2023 - 100% done
- **3.png**(image/png) - 582050 bytes, last modified: 22.09.2023 - 100% done
- **4.png**(image/png) - 356833 bytes, last modified: 22.09.2023 - 100% done
- **5.png**(image/png) - 481500 bytes, last modified: 22.09.2023 - 100% done
- **6.png**(image/png) - 156260 bytes, last modified: 22.09.2023 - 100% done
- **7.png**(image/png) - 367601 bytes, last modified: 22.09.2023 - 100% done
- **8.png**(image/png) - 2407753 bytes, last modified: 22.09.2023 - 100% done
- **9.png**(image/png) - 838536 bytes, last modified: 22.09.2023 - 100% done
- **10.png**(image/png) - 26679 bytes, last modified: 22.09.2023 - 100% done
- **11.png**(image/png) - 22717 bytes, last modified: 22.09.2023 - 100% done

```
Saving 1.png to 1.png
Saving 2.png to 2.png
Saving 3.png to 3.png
Saving 4.png to 4.png
Saving 5.png to 5.png
Saving 6.png to 6.png
Saving 7.png to 7.png
Saving 8.png to 8.png
Saving 9.png to 9.png
Saving 10.png to 10.png
Saving 11.png to 11.png
```

```
!python3 train_dreambooth.py \
  --pretrained_model_name_or_path=$MODEL_NAME \
  --pretrained_vae_name_or_path="Yasincan/ffhq" \
```

```
  --output_dir=$OUTPUT_DIR \
  --revision="fp16" \
  --with_prior_preservation --prior_loss_weight=1.0 \
  --seed=1337 \
  --resolution=32 \
  --train_batch_size=1 \
  --train_text_encoder \
  --mixed_precision="fp16" \
  --use_8bit_adam \
  --gradient_accumulation_steps=1 \
  --learning_rate=1e-6 \
  --lr_scheduler="constant" \
  --lr_warmup_steps=0 \
  --num_class_images=50 \
  --sample_batch_size=4 \
  --max_train_steps=800 \
  --save_interval=10000 \
  --save_sample_prompt="photo of Lebron James" \
  --concepts_list="concepts_list.json"
```

```
===================================BUG REPORT===================================
Welcome to bitsandbytes. For bug reports, please submit your error trace to: https://github.com/TimDettmers/bitsandbytes/issues
For effortless bug reporting copy-paste your error into this form: https://docs.google.com/forms/d/e/1FAIpQLScPB8emS3Thkp66nvqwmjTEgxp8
================================================================================
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:105: UserWarning: /usr/lib64-nvidia did not contain libcudart.
  warn(
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:27: UserWarning: WARNING: The following directories listed in
  warn(
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:27: UserWarning: WARNING: The following directories listed in
  warn(
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:27: UserWarning: WARNING: The following directories listed in
  warn(
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:27: UserWarning: WARNING: The following directories listed in
  warn(
/usr/local/lib/python3.10/dist-packages/bitsandbytes/cuda_setup/paths.py:27: UserWarning: WARNING: The following directories listed in
  warn(
CUDA_SETUP: WARNING! libcudart.so not found in any environmental path. Searching /usr/local/cuda/lib64...
CUDA SETUP: CUDA runtime path found: /usr/local/cuda/lib64/libcudart.so
CUDA SETUP: Highest compute capability among GPUs detected: 7.5
CUDA SETUP: Detected CUDA version 118
CUDA SETUP: Loading binary /usr/local/lib/python3.10/dist-packages/bitsandbytes/libbitsandbytes_cuda118.so...
2023-09-22 14:16:41.892883: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
Traceback (most recent call last):
  File "/content/train_dreambooth.py", line 869, in <module>
    main(args)
  File "/content/train_dreambooth.py", line 524, in main
    tokenizer = CLIPTokenizer.from_pretrained(
  File "/usr/local/lib/python3.10/dist-packages/transformers/tokenization_utils_base.py", line 1838, in from_pretrained
    raise EnvironmentError(
OSError: Can't load tokenizer for 'Yasincan/ffhq'. If you were trying to load it from 'https://huggingface.co/models', make sure you do
```

```
from natsort import natsorted
from glob import glob
import os
WEIGHTS_DIR = "content/stable_diffusion_weights/zwx"
print(f"[*] WEIGHTS_DIR={WEIGHTS_DIR}")
```

```
[*] WEIGHTS_DIR=content/stable_diffusion_weights/zwx
```

```
print(os.listdir(weights_folder))
```

```
[]
```

```
import os
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

weights_folder = OUTPUT_DIR
folders = sorted([f for f in os.listdir(weights_folder) if f != "0"], key=lambda x: int(x))

row = len(folders)
print(os.listdir(weights_folder))

col = len(os.listdir(os.path.join(weights_folder, folders[0], "samples")))

scale = 4
```

```
fig, axes = plt.subplots(row, col, figsize=(col*scale, row*scale), gridspec_kw={'hspace': 0, 'wspace': 0})

for i, folder in enumerate(folders):
    folder_path = os.path.join(weights_folder, folder)
    image_folder = os.path.join(folder_path, "samples")
    images = [f for f in os.listdir(image_folder)]
    for j, image in enumerate(images):
        if row == 1:
            currAxes = axes[j]
        else:
            currAxes = axes[i, j]
        if i == 0:
            currAxes.set_title(f"Image {j}")
        if j == 0:
            currAxes.text(-0.1, 0.5, folder, rotation=0, va='center', ha='center', transform=currAxes.transAxes)
        image_path = os.path.join(image_folder, image)
        img = mpimg.imread(image_path)
        currAxes.imshow(img, cmap='gray')
        currAxes.axis('off')

plt.tight_layout()
plt.savefig('grid.png', dpi=72)
```

```
[]
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-83-895ebe53ca61> in <cell line: 11>()
      9 print(os.listdir(weights_folder))
     10
---> 11 col = len(os.listdir(os.path.join(weights_folder, folders[0], "samples")))
     12
     13 scale = 4

IndexError: list index out of range
```

ARAMA YIĞINI TAŞMASI