```
!pip install diffusers transformers accelerate
```

```
Requirement already satisfied: diffusers in /usr/local/lib/python3.10/dist-packages (0.31.0)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.47.0)
Requirement already satisfied: accelerate in /usr/local/lib/python3.10/dist-packages (1.2.1)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from diffusers) (8.5.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from diffusers) (3.17.0)
Requirement already satisfied: huggingface-hub>=0.23.2 in /usr/local/lib/python3.10/dist-packages (from diffusers) (0.29.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from diffusers) (1.26.4)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from diffusers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from diffusers) (2.32.3)
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from diffusers) (0.4.5)
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (from diffusers) (11.0.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.21.0)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: torch>=1.10.0 in /usr/local/lib/python3.10/dist-packages (from accelerate) (2.5.1+cu121)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.2->diffusers
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.2-
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (2025.0.1)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (2022.0.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (2.4.1)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (3.1.4)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.10.0->accelerate) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch>=1.10.0->acc
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata->diffusers) (3.21.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->diffusers) (3.4.1
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->diffusers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->diffusers) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->diffusers) (2025.1.31)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.10.0->accelerate)
Requirement already satisfied: intel-openmp>=2024 in /usr/local/lib/python3.10/dist-packages (from mkl->numpy->diffusers) (2024.2.0
Requirement already satisfied: tbb==2022.* in /usr/local/lib/python3.10/dist-packages (from mkl->numpy->diffusers) (2022.0.0)
Requirement already satisfied: tcmlib==1.* in /usr/local/lib/python3.10/dist-packages (from tbb==2022.*->mkl->numpy->diffusers) (1.2
Requirement already satisfied: intel-cmplr-lib-rt in /usr/local/lib/python3.10/dist-packages (from mkl_umath->numpy->diffusers) (202
Requirement already satisfied: intel-cmplr-lib-ur==2024.2.0 in /usr/local/lib/python3.10/dist-packages (from intel-openmp>=2024->mkl
```

```
!pip show torch torchvision
```

```
Name: torch
Version: 2.0.1
Summary: Tensors and Dynamic neural networks in Python with strong GPU acceleration
Home-page: https://pytorch.org/
Author: PyTorch Team
Author-email: packages@pytorch.org
License: BSD-3
Location: /usr/local/lib/python3.10/dist-packages
Requires: filelock, jinja2, networkx, nvidia-cublas-cu11, nvidia-cuda-cupti-cu11, nvidia-cuda-nvrtc-cu11, nvidia-cuda-runtime-cu11,
Required-by: accelerate, easyocr, fastai, kornia, lpips, peft, pytorch-ignite, pytorch-lightning, sentence-transformers, stable-base
---
Name: torchvision
Version: 0.15.2+cu118
Summary: image and video datasets and models for torch deep learning
Home-page: https://github.com/pytorch/vision
Author: PyTorch Core Team
Author-email: soumith@pytorch.org
License: BSD
Location: /usr/local/lib/python3.10/dist-packages
Requires: numpy, pillow, requests, torch
Required-by: easyocr, fastai, lpips, timm
```

```
!pip uninstall torch torchvision -y
!pip cache purge  # 🔥 Clear cache to avoid conflicts

# ✅ Reinstall matching versions
!pip install torch==2.0.1 torchvision==0.15.2 --index-url https://download.pytorch.org/whl/cu118
```

```
Found existing installation: torch 2.0.1
Uninstalling torch-2.0.1:
  Successfully uninstalled torch-2.0.1
Found existing installation: torchvision 0.15.2+cu118
Uninstalling torchvision-0.15.2+cu118:
  Successfully uninstalled torchvision-0.15.2+cu118
Files removed: 190
Looking in indexes: https://download.pytorch.org/whl/cu118
Collecting torch==2.0.1
  Downloading https://download.pytorch.org/whl/cu118/torch-2.0.1%2Bcu118-cp310-cp310-linux_x86_64.whl (2267.3 MB)
```

```
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.9/2.3 GB 199.2 MB/s eta 0:00:02 Keeping Kaggle active...
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.3/2.3 GB 458.3 kB/s eta 0:00:0000:0100:01
Collecting torchvision==0.15.2
  Downloading https://download.pytorch.org/whl/cu118/torchvision-0.15.2%2Bcu118-cp310-cp310-linux_x86_64.whl (6.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.1/6.1 MB 99.9 MB/s eta 0:00:00:00:01
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (3.17.0)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (1.13.1)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (3.1.4)
Requirement already satisfied: triton==2.0.0 in /usr/local/lib/python3.10/dist-packages (from torch==2.0.1) (2.0.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from torchvision==0.15.2) (1.26.4)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from torchvision==0.15.2) (2.32.3)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in /usr/local/lib/python3.10/dist-packages (from torchvision==0.15.2) (11.0.0)
Requirement already satisfied: cmake in /usr/local/lib/python3.10/dist-packages (from triton==2.0.0->torch==2.0.1) (3.31.2)
Requirement already satisfied: lit in /usr/local/lib/python3.10/dist-packages (from triton==2.0.0->torch==2.0.1) (15.0.7)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch==2.0.1) (3.0.2)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (2025.0.1)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (2022.0.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.10/dist-packages (from numpy->torchvision==0.15.2) (2.4.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->torchvision==0.15
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->torchvision==0.15.2) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->torchvision==0.15.2) (2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->torchvision==0.15.2) (2
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch==2.0.1) (1.3.0)
Requirement already satisfied: intel-openmp>=2024 in /usr/local/lib/python3.10/dist-packages (from mkl->numpy->torchvision==0.15.2)
Requirement already satisfied: tbb==2022.* in /usr/local/lib/python3.10/dist-packages (from mkl->numpy->torchvision==0.15.2) (2022.6
Requirement already satisfied: tcmlib==1.* in /usr/local/lib/python3.10/dist-packages (from tbb==2022.*->mkl->numpy->torchvision==0
Requirement already satisfied: intel-cmplr-lib-rt in /usr/local/lib/python3.10/dist-packages (from mkl_umath->numpy->torchvision==0
Requirement already satisfied: intel-cmplr-lib-ur==2024.2.0 in /usr/local/lib/python3.10/dist-packages (from intel-openmp>=2024->mkl
Installing collected packages: torch, torchvision
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
pytorch-lightning 2.5.0.post0 requires torch>=2.1.0, but you have torch 2.0.1+cu118 which is incompatible.
torchaudio 2.5.1+cu121 requires torch==2.5.1, but you have torch 2.0.1+cu118 which is incompatible.
Successfully installed torch-2.0.1+cu118 torchvision-0.15.2+cu118
```

```
import torch
torch.cuda.empty_cache()
```

```
import torch
from diffusers import DiffusionPipeline

# ✅ Model Loading
model = DiffusionPipeline.from_pretrained("CompVis/ldm-celebahq-256")
model.to("cuda")  # Move model to GPU
```

```
The cache for model files in Transformers v4.22.0 has been updated. Migrating your old cache. This is a one-time only operation. You
0it [00:00, ?it/s]
/usr/local/lib/python3.10/dist-packages/torchvision/datapoints/__init__.py:12: UserWarning: The torchvision.datapoints and torchvisi
  warnings.warn(_BETA_TRANSFORMS_WARNING)
/usr/local/lib/python3.10/dist-packages/torchvision/transforms/v2/__init__.py:54: UserWarning: The torchvision.datapoints and torchv
  warnings.warn(_BETA_TRANSFORMS_WARNING)
model_index.json:   0%|          | 0.00/228 [00:00<?, ?B/s]
Fetching 6 files:   0%|          | 0/6 [00:00<?, ?it/s]
config.json:   0%|          | 0.00/712 [00:00<?, ?B/s]
config.json:   0%|          | 0.00/486 [00:00<?, ?B/s]
scheduler_config.json:   0%|          | 0.00/249 [00:00<?, ?B/s]
diffusion_pytorch_model.bin:   0%|          | 0.00/1.10G [00:00<?, ?B/s]
diffusion_pytorch_model.bin:   0%|          | 0.00/221M [00:00<?, ?B/s]
Loading pipeline components...:   0%|          | 0/3 [00:00<?, ?it/s]
An error occurred while trying to fetch /root/.cache/huggingface/hub/models--CompVis--ldm-celebahq-256/snapshots/03978f22272a3c2502c
Defaulting to unsafe serialization. Pass `allow_pickle=False` to raise an error instead.
/usr/local/lib/python3.10/dist-packages/torch/_utils.py:776: UserWarning: TypedStorage is deprecated. It will be removed in the futu
  return self.fget.__get__(instance, owner)()
An error occurred while trying to fetch /root/.cache/huggingface/hub/models--CompVis--ldm-celebahq-256/snapshots/03978f22272a3c2502c
Defaulting to unsafe serialization. Pass `allow_pickle=False` to raise an error instead.
The config attributes {'timestep_values': None, 'timesteps': 1000} were passed to DDIMScheduler, but are not expected and will be ig
LDMPipeline {
  "_class_name": "LDMPipeline",
  "_diffusers_version": "0.31.0",
  "_name_or_path": "CompVis/ldm-celebahq-256",
  "scheduler": [
    "diffusers",
    "DDIMScheduler"
  ],
  "unet": [
    "diffusers",
    "UNet2DModel"
  ],
  "vqvae": [
    "diffusers",
    "VQModel"
  ]
}
```

```
import time
import threading

def prevent_disconnect():
    """
    ✅ Runs a background thread that prints a message every 5-10 minutes
    to prevent Kaggle from disconnecting.
    """
    interval = 60*10  # ✅ 5 minutes interval
    print("Preventing Kaggle auto-disconnect...")

    def keep_active():
        while True:
            print(" Keeping Kaggle active...")
            time.sleep(interval)

    # ✅ Run the keep-alive thread
    thread = threading.Thread(target=keep_active)
    thread.daemon = True  # ✅ Stops the thread when the script stops
    thread.start()

# ✅ Start the script
prevent_disconnect()
```

```
Preventing Kaggle auto-disconnect...
 Keeping Kaggle active...
```

```
import os
import numpy as np
import torch
from torchvision import transforms
from PIL import Image
from diffusers import VQModel

import torch
import h5py
import numpy as np
import os

def preprocess_and_encode_hdf5(hdf5_file, model, batch_size=2000):
    """
    Preprocess and encode images from the HDF5 file into latent representations.
```

```
    Args:
        hdf5_file (str): Path to the HDF5 file.
        model (LDMPipeline): LDM model for encoding.
        batch_size (int): Batch size for encoding.

    Returns:
        np.ndarray: Array of latent representations.
    """
    latents = []

    # ✅ Open HDF5 file
    with h5py.File(hdf5_file, "r") as f:
        images = f['X_jets']  # Only extract the images
        num_images = images.shape[0]

        print(f"Total images: {num_images}")

        # ✅ Process in batches
        for i in range(0, num_images, batch_size):
            batch = images[i:i + batch_size]  # Load batch
            print(f"Processing batch {i // batch_size + 1}/{(num_images // batch_size) + 1} with shape: {batch.shape}")

            batch_latents = []

            for img in batch:
                # ✅ Convert to PyTorch tensor
                img_tensor = torch.from_numpy(img).float()

                # ✅ Normalize from [0, 255] to [-1, 1]
                img_tensor = img_tensor / 255.0
                img_tensor = (img_tensor - 0.5) / 0.5

                # ✅ Ensure shape is (3, 125, 125)
                if img_tensor.shape == (125, 125, 3):
                    img_tensor = img_tensor.permute(2, 0, 1)

                # ✅ Add batch dimension and move to GPU
                img_tensor = img_tensor.unsqueeze(0).to("cuda")

                # ✅ Encode using VQVAE to latent space (CORRECTED!)
                with torch.no_grad():
                    latent = model.vqvae.encode(img_tensor).latents.cpu().numpy()  # 🔥 Use .latents
                    batch_latents.append(latent)

            # ✅ Append batch latents
            latents.extend(batch_latents)

    return np.array(latents)

# ✅ Define paths
hdf5_file = "/kaggle/input/dataset-hdf5/quark-gluon_data-set_n139306.hdf5"
latent_save_dir = "/kaggle/working/ldm_latents"

# ✅ Create save directory
os.makedirs(latent_save_dir, exist_ok=True)

# ✅ Preprocess and encode
latents = preprocess_and_encode_hdf5(hdf5_file, model, batch_size=2000)

# ✅ Save the latent representations
latent_file = os.path.join(latent_save_dir, "latents.npy")
np.save(latent_file, latents)

print(f"✅ Saved latents at {latent_file}")


import os
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import Dataset, DataLoader
from diffusers import UNet2DModel, AutoencoderKL  # Use the individual components
from tqdm import tqdm
from torch.cuda.amp import GradScaler, autocast
import numpy as np

# ✅ Configuration Parameters
BATCH_SIZE = 64
EPOCHS = 20
LEARNING_RATE = 1e-4
SAVE_DIR = "/content/MyDrive/MyDrive/QuarkGluonBatches"
LATENT_FILE = "/content/MyDrive/MyDrive/QuarkGluonBatches/latents.npy"
```

```
USE_MULTI_GPU = True

# ✅ Ensure save directory exists
os.makedirs(SAVE_DIR, exist_ok=True)



### 🚦 **✅ 1️⃣ Memory-Mapped Latent Dataset**

class LatentDataset(Dataset):
    """
    Custom dataset to load latent vectors batch-wise using memory mapping.

    Args:
        latent_file (str): Path to the single .npy latent file.
    """
    def __init__(self, latent_file):
        # ✅ Memory-mapping the entire .npy file
        self.latents = np.load(latent_file, mmap_mode="r")
        self.num_samples = self.latents.shape[0]

    def __len__(self):
        return self.num_samples

    def __getitem__(self, idx):
        latent = torch.tensor(self.latents[idx], dtype=torch.float32)
        return latent


# ✅ Load the dataset and create DataLoader
print("✅ Loading dataset with memory mapping...")
dataset = LatentDataset(LATENT_FILE)
dataloader = DataLoader(dataset, batch_size=BATCH_SIZE, shuffle=True, num_workers=4, pin_memory=True)
print(f"✅ Dataset size: {len(dataset)} latents")


import os
import subprocess
from IPython.display import FileLink, display

def download_file(path, download_file_name):
    os.chdir('/kaggle/working/')
    zip_name = f"/kaggle/working/{download_file_name}.zip"
    command = f"zip {zip_name} {path} -r"
    result = subprocess.run(command, shell=True, capture_output=True, text=True)
    if result.returncode != 0:
        print("Unable to run zip command!")
        print(result.stderr)
        return
    display(FileLink(f'{download_file_name}.zip'))


# download_file('/kaggle/working/ldm_latents/latents.npy', 'out')


import numpy as np

# ✅ Load the latents
latents = np.load("/kaggle/input/ldm-latents/latents.npy")

# ✅ Print the shape and type
print("Latents shape:", latents.shape)
print("Latents dtype:", latents.dtype)
```

```
Latents shape: (139306, 1, 3, 31, 31)
Latents dtype: float32
```

```
# ✅ Upgrade diffusers and PyTorch
!pip install --upgrade diffusers torch
```

```
Requirement already satisfied: diffusers in /usr/local/lib/python3.10/dist-packages (0.31.0)
Collecting diffusers
  Downloading diffusers-0.32.2-py3-none-any.whl.metadata (18 kB)
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (2.0.1+cu118)
Collecting torch
  Downloading torch-2.6.0-cp310-cp310-manylinux1_x86_64.whl.metadata (28 kB)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from diffusers) (8.5.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from diffusers) (3.17.0)
Requirement already satisfied: huggingface-hub>=0.23.2 in /usr/local/lib/python3.10/dist-packages (from diffusers) (0.29.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from diffusers) (1.26.4)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from diffusers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from diffusers) (2.32.3)
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from diffusers) (0.4.5)
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (from diffusers) (11.0.0)
```

```
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.10/dist-packages (from torch) (4.12.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch) (2024.12.0)
Collecting nvidia-cuda-nvrtc-cu12==12.4.127 (from torch)
  Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-runtime-cu12==12.4.127 (from torch)
  Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-cupti-cu12==12.4.127 (from torch)
  Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cudnn-cu12==9.1.0.70 (from torch)
  Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cublas-cu12==12.4.5.8 (from torch)
  Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cufft-cu12==11.2.1.3 (from torch)
  Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu12==10.3.5.147 (from torch)
  Downloading nvidia_curand_cu12-10.3.5.147-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cusolver-cu12==11.6.1.9 (from torch)
  Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparse-cu12==12.3.1.170 (from torch)
  Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-manylinux2014_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparselt-cu12==0.6.2 (from torch)
  Downloading nvidia_cusparselt_cu12-0.6.2-py3-none-manylinux2014_x86_64.whl.metadata (6.8 kB)
Collecting nvidia-nccl-cu12==2.21.5 (from torch)
  Downloading nvidia_nccl_cu12-2.21.5-py3-none-manylinux2014_x86_64.whl.metadata (1.8 kB)
Collecting nvidia-nvtx-cu12==12.4.127 (from torch)
  Downloading nvidia_nvtx_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (1.7 kB)
Collecting nvidia-nvjitlink-cu12==12.4.127 (from torch)
  Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting triton==3.2.0 (from torch)
  Downloading triton-3.2.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.4 kB)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch) (1.3.0)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.2->diffuser
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.2->diffusers)
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.23.2->diffusers)
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata->diffusers) (3.21.6
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch) (3.0.2)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-packages (from numpy->diffusers) (0.1.1)
```

```
!pip install ldm
```

```
Collecting ldm
  Downloading ldm-0.1.3.tar.gz (6.1 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: ldm
  Building wheel for ldm (setup.py) ... done
  Created wheel for ldm: filename=ldm-0.1.3-py3-none-any.whl size=6206 sha256=4042094e9fffbb5c71ed0b8de7be7f3a666ef2241a6247769bab6b
  Stored in directory: /root/.cache/pip/wheels/d7/66/44/8ac06fa0add7124672b8e7413aad60f972f2a29d8ef07678f1
Successfully built ldm
Installing collected packages: ldm
Successfully installed ldm-0.1.3
```

```
import torch
import numpy as np
from torch.utils.data import DataLoader, TensorDataset
from diffusers import AutoencoderKL
from tqdm import tqdm
import gc

# ✅ Config
device = "cuda" if torch.cuda.is_available() else "cpu"
print(f"🔥 Using device: {device}")

# ✅ Load raw images into RAM
raw_images = np.load("/content/MyDrive/MyDrive/QuarkGluonBatches/latents.npy").astype(np.float32)
print("🔥 Raw image shape:", raw_images.shape)

# ✅ Remove unnecessary dimensions
if raw_images.shape[1] == 1:
    raw_images = raw_images.squeeze(1)  # Remove single dimension → (139306, 3, 31, 31)

# ✅ Normalize images if needed
if raw_images.max() > 1.0:
    raw_images /= 255.0

# ✅ Convert to TensorDataset & DataLoader
images = torch.from_numpy(raw_images).float()
dataset = TensorDataset(images)
BATCH_SIZE = 32  # 🔥 Smaller batch size to avoid OOM
```

```
loader = DataLoader(dataset, batch_size=BATCH_SIZE, shuffle=False)

# ✅ Load VAE model once (Lazy loading)
print("🚀 Loading VAE model...")
vae = AutoencoderKL.from_pretrained("stabilityai/stable-diffusion-2-1", subfolder="vae").to(device)
vae.eval()

# ✅ Encode the images into latents
latents = []

# ✅ Encoding loop with memory clearing
with torch.no_grad():
    for batch_idx, (batch,) in enumerate(tqdm(loader)):
        batch = batch.to(device)

        # ✅ Resize to (256, 256) if necessary
        if batch.shape[-1] != 256:
            batch = torch.nn.functional.interpolate(batch, size=(256, 256), mode='bilinear', align_corners=False)

        # ✅ Encode batch into latents
        try:
            encoded = vae.encode(batch).latent_dist.sample()
            latents.append(encoded.cpu().numpy())

            # ✅ Clear CUDA memory after each batch
            del batch, encoded
            torch.cuda.empty_cache()
            gc.collect()

        except torch.cuda.OutOfMemoryError:
            print(f"🚫 OOM at batch {batch_idx}. Reducing batch size!")
            BATCH_SIZE = max(BATCH_SIZE // 2, 1)  # Reduce batch size by half
            loader = DataLoader(dataset, batch_size=BATCH_SIZE, shuffle=False)

# ✅ Concatenate all latent batches
latents = np.concatenate(latents, axis=0)

# ✅ Save the latents
output_path = "/kaggle/input/encoded-latents/encoded_latents.npy"
np.save(output_path, latents)
print(f"🔥 Latents saved as '{output_path}'")
print(f"✅ Final latent shape: {latents.shape}")


import os
import gc
import time
import torch
import numpy as np
import torch.nn as nn
import torch.optim as optim
from diffusers import UNet2DConditionModel, AutoencoderKL
from torch.cuda.amp import GradScaler, autocast
from torch.utils.data import DataLoader, TensorDataset
from tqdm import tqdm

# ✅ CUDA Environment Fixes
os.environ["PYTORCH_CUDA_ALLOC_CONF"] = "expandable_segments:True"
os.environ["CUDA_LAUNCH_BLOCKING"] = "1"  # ✅ More accurate error reporting
torch.cuda.memory._set_allocator_settings("max_split_size_mb:128")  # ✅ Prevent fragmentation

# ✅ Memory Cleanup
torch.cuda.empty_cache()
gc.collect()

# ✅ TF32 Stability Fix
torch.backends.cuda.matmul.allow_tf32 = False
torch.backends.cudnn.allow_tf32 = False

# ✅ Config
device = "cuda" if torch.cuda.is_available() else "cpu"
print(f"🚀 Using device: {device}")

# ✅ Memory Management
def free_memory():
    """🔥 Free VRAM & clear cache"""
    torch.cuda.empty_cache()
    gc.collect()

free_memory()

# ✅ Load Latents
print("🔥 Loading latents...")
```

```
latents_path = "/kaggle/input/encoded-latents/encoded_latents.npy"

if not os.path.exists(latents_path):
    print(f"🚫 Latent file not found: {latents_path}")
    exit(1)

latents = np.load(latents_path).astype(np.float32)
print(f"✅ Latents shape: {latents.shape}")

# ✅ Prepare DataLoader with Dynamic Batch Resizing
BATCH_SIZE = 8  # ✅ Static batch size initially
ACCUMULATION_STEPS = 8
SHUFFLE = True

latents_tensor = torch.from_numpy(latents).float().to(device)
dataset = TensorDataset(latents_tensor)
loader = DataLoader(dataset, batch_size=BATCH_SIZE, shuffle=SHUFFLE, num_workers=0)

# ✅ Load VAE model with AMP & gradient checkpointing
print("🚀 Loading VAE model...")
vae = None
try:
    vae = AutoencoderKL.from_pretrained(
        "stabilityai/stable-diffusion-2-1",
        subfolder="vae",
        force_download=True
    ).to(device)

    # ✅ Half precision for memory efficiency
    vae.half()

    # ✅ Enable gradient checkpointing
    vae.enable_gradient_checkpointing()
    vae.eval()

    print("✅ VAE model loaded successfully!")

except Exception as e:
    print(f"❌ VAE model loading failed: {e}")
    exit(1)

# ✅ Load UNet model with OOM Handling
print("🚀 Loading UNet model...")

try:
    free_memory()

    unet = UNet2DConditionModel.from_pretrained(
        "CompVis/stable-diffusion-v1-4",
        subfolder="unet"
    ).to(device)

    # ✅ Half precision for memory efficiency
    unet.half()

    # ✅ Graph Compilation for memory optimization
    unet = torch.compile(unet)

    # ✅ Enable gradient checkpointing
    unet.enable_gradient_checkpointing()

    print("✅ UNet model loaded successfully!")

except torch.cuda.OutOfMemoryError:
    print("❌ CUDA OOM during UNet loading. Exiting training.")
    free_memory()
    exit(1)

except Exception as e:
    print(f"❌ Failed to load UNet: {e}")
    free_memory()
    exit(1)

# ✅ Optimizer, Scaler, and Loss
LEARNING_RATE = 1e-4
EPOCHS = 10
optimizer = optim.AdamW(unet.parameters(), lr=LEARNING_RATE, weight_decay=1e-2)
scaler = GradScaler()
criterion = nn.MSELoss()

# ✅ Dynamic Batch Resizing Function
# ✅ Dynamic Batch Resizing Function
```

```
def dynamic_batch_resize(batch_idx, batch):
    """🔥 Automatically reduce batch size on OOM"""
    global BATCH_SIZE

    try:
        with autocast():
            # ✅ Decode latent batch back to image with AMP
            with torch.no_grad():
                decoded = vae.decode(batch).sample  # (B, 3, H, W)

                # ✅ Fix channel mismatch by adding a 4th channel
                noise_channel = torch.zeros_like(decoded[:, :1, :, :])  # (B, 1, H, W)
                decoded = torch.cat((decoded, noise_channel), dim=1)  # (B, 4, H, W)

            # ✅ Add noise
            noise = torch.randn_like(decoded)

            # ✅ UNet Forward Pass with Timesteps and Hidden States
            timesteps = torch.randint(0, 1000, (noise.shape[0],), device=device)
            encoder_hidden_states = torch.randn_like(noise)

            output = unet(noise, timesteps, encoder_hidden_states).sample

            # ✅ Loss calculation
            loss = criterion(output, decoded) / ACCUMULATION_STEPS
            scaler.scale(loss).backward()

        return batch  # ✅ Successful batch return

    except torch.cuda.OutOfMemoryError:
        print(f"🚫 OOM at batch {batch_idx}. Reducing batch size...")

        # ✅ Reduce batch size progressively
        BATCH_SIZE = max(BATCH_SIZE // 2, 1)
        loader = DataLoader(dataset, batch_size=BATCH_SIZE, shuffle=SHUFFLE, num_workers=0)

        free_memory()  # ✅ Immediate VRAM cleanup
        return None  # 🚫 Skip batch on OOM


# ✅ Training Loop with Frequent Memory Cleanup
SAVE_DIR = "/kaggle/working/checkpoints"
os.makedirs(SAVE_DIR, exist_ok=True)

print("🔥 Starting training...")

for epoch in range(EPOCHS):
    unet.train()

    running_loss = 0.0
    optimizer.zero_grad()

    for batch_idx, (batch,) in enumerate(tqdm(loader)):
        batch = batch.to(device)

        # ✅ Apply dynamic batch resizing on OOM
        batch = dynamic_batch_resize(batch_idx, batch)
        if batch is None:
            continue  # 🚫 Skip batch if OOM occurred

        # ✅ AMP Training
        with autocast():
            # ✅ Decode latent batch back to image with AMP
            with torch.no_grad():
                decoded = vae.decode(batch).sample

            # ✅ Add noise
            noise = torch.randn_like(decoded)

            # ✅ Forward pass through UNet
            timesteps = torch.randint(0, 1000, (noise.shape[0],), device=device)
            encoder_hidden_states = torch.randn_like(noise)

            output = unet(noise, timesteps, encoder_hidden_states).sample

            # ✅ Loss calculation
            loss = criterion(output, decoded) / ACCUMULATION_STEPS

        # ✅ Backpropagation with gradient accumulation
        scaler.scale(loss).backward()

        # ✅ Gradient accumulation step
```

```
        if (batch_idx + 1) % ACCUMULATION_STEPS == 0 or batch_idx == len(loader) - 1:
            scaler.step(optimizer)
            scaler.update()
            optimizer.zero_grad()

        running_loss += loss.item() * ACCUMULATION_STEPS

        # ✅ Memory cleanup after every batch
        free_memory()

        if batch_idx % 10 == 0:
            print(f"Epoch [{epoch + 1}/{EPOCHS}] Batch [{batch_idx}/{len(loader)}] Loss: {loss.item():.6f}")

    avg_loss = running_loss / len(loader)
    print(f"🔥 Epoch [{epoch + 1}/{EPOCHS}] - Average Loss: {avg_loss:.6f}")

    # ✅ Save model checkpoint
    checkpoint_path = os.path.join(SAVE_DIR, f"unet_epoch_{epoch + 1}.pt")
    torch.save(unet.state_dict(), checkpoint_path)
    print(f"✅ Model saved at {checkpoint_path}")

print("🎯 Training completed successfully!")
```

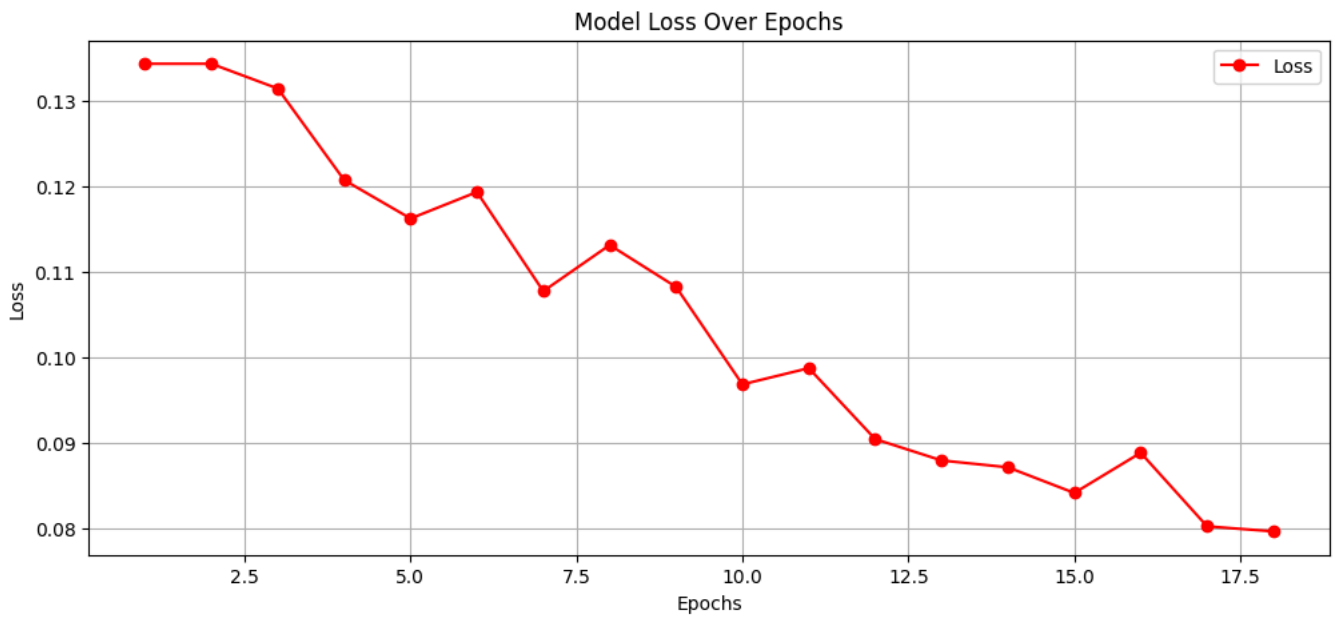
```
🚀 Using device: cuda
🔥 Loading latents...
✅ Latents shape: (139306, 4, 32, 32)
🚀 Loading VAE model...
config.json:   0%|          | 0.00/611 [00:00<?, ?B/s]
diffusion_pytorch_model.safetensors:   0%|          | 0.00/335M [00:00<?, ?B/s]
✅ VAE model loaded successfully!
🚀 Loading UNet model...
✅ UNet model loaded successfully!
🔥 Starting training...

🔥 Epoch [1/18] started...
  1%|          | 1/120 [00:00<00:34,  3.43it/s]Epoch [1/18] Batch [0/120] Loss: 0.057051
 10%|▉         | 12/120 [00:03<00:28,  3.74it/s]Epoch [1/18] Batch [10/120] Loss: 0.123224
 18%|▉▉        | 22/120 [00:05<00:26,  3.76it/s]Epoch [1/18] Batch [20/120] Loss: 0.100639
 26%|▉▉        | 31/120 [00:08<00:24,  3.70it/s]Epoch [1/18] Batch [30/120] Loss: 0.074734
 35%|▉▉▉       | 42/120 [00:11<00:20,  3.77it/s]Epoch [1/18] Batch [40/120] Loss: 0.125025
 43%|▉▉▉▉      | 52/120 [00:14<00:18,  3.75it/s]Epoch [1/18] Batch [50/120] Loss: 0.062071
 52%|▉▉▉▉▉     | 62/120 [00:16<00:15,  3.71it/s]Epoch [1/18] Batch [60/120] Loss: 0.128939
 59%|▉▉▉▉▉▉    | 71/120 [00:19<00:12,  3.80it/s]Epoch [1/18] Batch [70/120] Loss: 0.138267
 68%|▉▉▉▉▉▉▉   | 82/120 [00:21<00:10,  3.78it/s]Epoch [1/18] Batch [80/120] Loss: 0.120313
 76%|▉▉▉▉▉▉▉▉  | 91/120 [00:24<00:07,  3.79it/s]Epoch [1/18] Batch [90/120] Loss: 0.104826
 84%|▉▉▉▉▉▉▉▉▉ | 101/120 [00:27<00:05,  3.46it/s]Epoch [1/18] Batch [100/120] Loss: 0.094069
 93%|▉▉▉▉▉▉▉▉▉▉| 112/120 [00:30<00:02,  3.76it/s]Epoch [1/18] Batch [110/120] Loss: 0.085210
100%|▉▉▉▉▉▉▉▉▉▉| 120/120 [00:32<00:00,  3.73it/s]
🔥 Epoch [1/18] - Average Loss: 0.101093
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_1.pt

🔥 Epoch [2/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [2/18] Batch [0/120] Loss: 0.135708
  8%|▉         | 10/120 [00:02<00:29,  3.74it/s]Epoch [2/18] Batch [10/120] Loss: 0.101904
 18%|▉▉        | 22/120 [00:05<00:25,  3.77it/s]Epoch [2/18] Batch [20/120] Loss: 0.106672
 26%|▉▉        | 31/120 [00:08<00:23,  3.78it/s]Epoch [2/18] Batch [30/120] Loss: 0.132408
 34%|▉▉▉       | 41/120 [00:10<00:21,  3.74it/s]Epoch [2/18] Batch [40/120] Loss: 0.137665
 42%|▉▉▉▉      | 51/120 [00:13<00:18,  3.79it/s]Epoch [2/18] Batch [50/120] Loss: 0.122505
 50%|▉▉▉▉▉     | 60/120 [00:15<00:16,  3.72it/s]Epoch [2/18] Batch [60/120] Loss: 0.142915
 59%|▉▉▉▉▉▉    | 71/120 [00:18<00:13,  3.77it/s]Epoch [2/18] Batch [70/120] Loss: 0.146239
 68%|▉▉▉▉▉▉▉   | 81/120 [00:21<00:10,  3.73it/s]Epoch [2/18] Batch [80/120] Loss: 0.135629
 77%|▉▉▉▉▉▉▉▉  | 92/120 [00:24<00:07,  3.79it/s]Epoch [2/18] Batch [90/120] Loss: 0.111098
 85%|▉▉▉▉▉▉▉▉▉ | 102/120 [00:27<00:04,  3.60it/s]Epoch [2/18] Batch [100/120] Loss: 0.127445
 10%|▉         | 12/120 [00:03<00:28,  3.75it/s]]Epoch [3/18] Batch [10/120] Loss: 0.052347
 18%|▉▉        | 22/120 [00:05<00:26,  3.74it/s]Epoch [3/18] Batch [20/120] Loss: 0.118043
 43%|▉▉▉▉      | 52/120 [00:13<00:18,  3.71it/s]Epoch [3/18] Batch [50/120] Loss: 0.133140
 52%|▉▉▉▉▉     | 62/120 [00:16<00:15,  3.76it/s]Epoch [3/18] Batch [60/120] Loss: 0.140996
 59%|▉▉▉▉▉▉    | 71/120 [00:19<00:13,  3.76it/s]Epoch [3/18] Batch [70/120] Loss: 0.107884
 68%|▉▉▉▉▉▉▉   | 82/120 [00:21<00:09,  3.81it/s]Epoch [3/18] Batch [80/120] Loss: 0.106401
 75%|▉▉▉▉▉▉▉   | 90/120 [00:24<00:07,  3.78it/s]Epoch [3/18] Batch [90/120] Loss: 0.068747
 84%|▉▉▉▉▉▉▉▉▉ | 101/120 [00:26<00:05,  3.64it/s]Epoch [3/18] Batch [100/120] Loss: 0.124007
 93%|▉▉▉▉▉▉▉▉▉▉| 112/120 [00:29<00:02,  3.75it/s]Epoch [3/18] Batch [110/120] Loss: 0.079624
100%|▉▉▉▉▉▉▉▉▉▉| 120/120 [00:32<00:00,  3.74it/s]
🔥 Epoch [3/18] - Average Loss: 0.101321
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_3.pt

🔥 Epoch [4/18] started...
  1%|          | 1/120 [00:00<00:30,  3.88it/s]Epoch [4/18] Batch [0/120] Loss: 0.077215
  9%|▉         | 11/120 [00:02<00:28,  3.79it/s]Epoch [4/18] Batch [10/120] Loss: 0.058204
 18%|▉▉        | 21/120 [00:05<00:25,  3.82it/s]Epoch [4/18] Batch [20/120] Loss: 0.088260
 26%|▉▉        | 31/120 [00:08<00:23,  3.76it/s]Epoch [4/18] Batch [30/120] Loss: 0.117397
 34%|▉▉▉       | 41/120 [00:10<00:21,  3.75it/s]Epoch [4/18] Batch [40/120] Loss: 0.095689
 42%|▉▉▉▉      | 50/120 [00:13<00:18,  3.79it/s]Epoch [4/18] Batch [50/120] Loss: 0.136348
 51%|▉▉▉▉▉     | 61/120 [00:16<00:15,  3.75it/s]Epoch [4/18] Batch [60/120] Loss: 0.125359
 60%|▉▉▉▉▉▉    | 72/120 [00:19<00:12,  3.79it/s]Epoch [4/18] Batch [70/120] Loss: 0.085457
 68%|▉▉▉▉▉▉▉   | 81/120 [00:21<00:10,  3.79it/s]Epoch [4/18] Batch [80/120] Loss: 0.054168
 77%|▉▉▉▉▉▉▉▉  | 92/120 [00:24<00:07,  3.85it/s]Epoch [4/18] Batch [90/120] Loss: 0.109769
 83%|▉▉▉▉▉▉▉▉▉ | 100/120 [00:26<00:05,  3.55it/s]Epoch [4/18] Batch [100/120] Loss: 0.103079
 92%|▉▉▉▉▉▉▉▉▉▉| 111/120 [00:29<00:02,  3.74it/s]Epoch [4/18] Batch [110/120] Loss: 0.115048
100%|▉▉▉▉▉▉▉▉▉▉| 120/120 [00:31<00:00,  3.77it/s]
🔥 Epoch [4/18] - Average Loss: 0.097986
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_4.pt

🔥 Epoch [5/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [5/18] Batch [0/120] Loss: 0.088131
  8%|▉         | 10/120 [00:02<00:29,  3.79it/s]Epoch [5/18] Batch [10/120] Loss: 0.070260
 18%|▉▉        | 22/120 [00:05<00:26,  3.75it/s]Epoch [5/18] Batch [20/120] Loss: 0.077452
 26%|▉▉        | 31/120 [00:08<00:23,  3.76it/s]Epoch [5/18] Batch [30/120] Loss: 0.141070
 34%|▉▉▉       | 41/120 [00:10<00:20,  3.77it/s]Epoch [5/18] Batch [40/120] Loss: 0.072982
 43%|▉▉▉▉      | 52/120 [00:13<00:17,  3.79it/s]Epoch [5/18] Batch [50/120] Loss: 0.143497
 50%|▉▉▉▉▉     | 60/120 [00:15<00:15,  3.78it/s]Epoch [5/18] Batch [60/120] Loss: 0.111434
 60%|▉▉▉▉▉▉    | 72/120 [00:19<00:12,  3.81it/s]Epoch [5/18] Batch [70/120] Loss: 0.108340
 68%|▉▉▉▉▉▉▉   | 82/120 [00:21<00:09,  3.83it/s]Epoch [5/18] Batch [80/120] Loss: 0.073861
 77%|▉▉▉▉▉▉▉▉  | 92/120 [00:24<00:07,  3.79it/s]Epoch [5/18] Batch [90/120] Loss: 0.054848
 85%|▉▉▉▉▉▉▉▉▉ | 102/120 [00:27<00:04,  3.65it/s]Epoch [5/18] Batch [100/120] Loss: 0.114020
 92%|▉▉▉▉▉▉▉▉▉▉| 110/120 [00:29<00:02,  3.68it/s]Epoch [5/18] Batch [110/120] Loss: 0.112671
100%|▉▉▉▉▉▉▉▉▉▉| 120/120 [00:31<00:00,  3.77it/s]
🔥 Epoch [5/18] - Average Loss: 0.099065
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_5.pt

🔥 Epoch [6/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [6/18] Batch [0/120] Loss: 0.132481
  9%|▉         | 11/120 [00:02<00:28,  3.79it/s]Epoch [6/18] Batch [10/120] Loss: 0.129229
```

```
 18%|██         | 21/120 [00:05<00:26,  3.78it/s]Epoch [6/18] Batch [20/120] Loss: 0.069010
 27%|└└└        | 32/120 [00:08<00:23,  3.77it/s]Epoch [6/18] Batch [30/120] Loss: 0.147190
 35%|└└└└       | 42/120 [00:11<00:20,  3.83it/s]Epoch [6/18] Batch [40/120] Loss: 0.062238
 42%|└└└└       | 51/120 [00:13<00:18,  3.76it/s]Epoch [6/18] Batch [50/120] Loss: 0.080658
 51%|└└└└└      | 61/120 [00:16<00:15,  3.84it/s]Epoch [6/18] Batch [60/120] Loss: 0.137552
 60%|└└└└└└     | 72/120 [00:18<00:12,  3.84it/s]Epoch [6/18] Batch [70/120] Loss: 0.111662
 68%|└└└└└└└    | 82/120 [00:21<00:09,  3.86it/s]Epoch [6/18] Batch [80/120] Loss: 0.139193
 77%|└└└└└└└└   | 92/120 [00:24<00:07,  3.76it/s]Epoch [6/18] Batch [90/120] Loss: 0.149253
 84%|└└└└└└└└└  | 101/120 [00:26<00:05,  3.58it/s]Epoch [6/18] Batch [100/120] Loss: 0.133550
 93%|└└└└└└└└└└ | 112/120 [00:29<00:02,  3.82it/s]Epoch [6/18] Batch [110/120] Loss: 0.071591
100%|└└└└└└└└└└└| 120/120 [00:31<00:00,  3.80it/s]
🔥 Epoch [6/18] - Average Loss: 0.103612
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_6.pt

🔥 Epoch [7/18] started...
  1%|           | 1/120 [00:00<00:30,  3.88it/s]Epoch [7/18] Batch [0/120] Loss: 0.084771
  9%|└           | 11/120 [00:02<00:28,  3.76it/s]Epoch [7/18] Batch [10/120] Loss: 0.149050
 18%|└└          | 21/120 [00:05<00:26,  3.75it/s]Epoch [7/18] Batch [20/120] Loss: 0.066353
 27%|└└└         | 32/120 [00:08<00:23,  3.82it/s]Epoch [7/18] Batch [30/120] Loss: 0.084383
 34%|└└└└        | 41/120 [00:10<00:20,  3.81it/s]Epoch [7/18] Batch [40/120] Loss: 0.095977
 43%|└└└└        | 52/120 [00:13<00:17,  3.80it/s]Epoch [7/18] Batch [50/120] Loss: 0.089509
 52%|└└└└└       | 62/120 [00:16<00:15,  3.84it/s]Epoch [7/18] Batch [60/120] Loss: 0.072127
 59%|└└└└└└      | 71/120 [00:18<00:12,  3.85it/s]Epoch [7/18] Batch [70/120] Loss: 0.112095
 67%|└└└└└└└     | 80/120 [00:21<00:10,  3.74it/s]Epoch [7/18] Batch [80/120] Loss: 0.111773
 76%|└└└└└└└└    | 91/120 [00:23<00:07,  3.81it/s]Epoch [7/18] Batch [90/120] Loss: 0.135004
 84%|└└└└└└└└└   | 101/120 [00:26<00:05,  3.53it/s]Epoch [7/18] Batch [100/120] Loss: 0.125429
 92%|└└└└└└└└└└  | 110/120 [00:28<00:02,  3.81it/s]Epoch [7/18] Batch [110/120] Loss: 0.146340
100%|└└└└└└└└└└└| 120/120 [00:31<00:00,  3.79it/s]
🔥 Epoch [7/18] - Average Loss: 0.096814
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_7.pt

🔥 Epoch [8/18] started...
  0%|           | 0/120 [00:00<?, ?it/s]Epoch [8/18] Batch [0/120] Loss: 0.129721
  9%|└           | 11/120 [00:02<00:28,  3.84it/s]Epoch [8/18] Batch [10/120] Loss: 0.149134
 18%|└└          | 21/120 [00:05<00:26,  3.78it/s]Epoch [8/18] Batch [20/120] Loss: 0.137421
 27%|└└└         | 32/120 [00:08<00:22,  3.83it/s]Epoch [8/18] Batch [30/120] Loss: 0.093179
 34%|└└└└        | 41/120 [00:10<00:20,  3.84it/s]Epoch [8/18] Batch [40/120] Loss: 0.144918
 42%|└└└└        | 51/120 [00:13<00:17,  3.85it/s]Epoch [8/18] Batch [50/120] Loss: 0.060075
 50%|└└└└└       | 60/120 [00:15<00:15,  3.80it/s]Epoch [8/18] Batch [60/120] Loss: 0.066872
 60%|└└└└└└      | 72/120 [00:18<00:12,  3.87it/s]Epoch [8/18] Batch [70/120] Loss: 0.093185
 68%|└└└└└└└     | 81/120 [00:21<00:10,  3.70it/s]Epoch [8/18] Batch [80/120] Loss: 0.115249
 77%|└└└└└└└└    | 92/120 [00:24<00:07,  3.84it/s]Epoch [8/18] Batch [90/120] Loss: 0.070285
 84%|└└└└└└└└└   | 101/120 [00:26<00:05,  3.66it/s]Epoch [8/18] Batch [100/120] Loss: 0.131735
 93%|└└└└└└└└└└  | 112/120 [00:29<00:02,  3.71it/s]Epoch [8/18] Batch [110/120] Loss: 0.129138
100%|└└└└└└└└└└└| 120/120 [00:31<00:00,  3.80it/s]
🔥 Epoch [8/18] - Average Loss: 0.098052
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_8.pt

🔥 Epoch [9/18] started...
  0%|           | 0/120 [00:00<?, ?it/s]Epoch [9/18] Batch [0/120] Loss: 0.116089
 10%|└           | 12/120 [00:03<00:28,  3.85it/s]Epoch [9/18] Batch [10/120] Loss: 0.050122
 18%|└└          | 22/120 [00:05<00:25,  3.84it/s]Epoch [9/18] Batch [20/120] Loss: 0.127078
 27%|└└└         | 32/120 [00:08<00:23,  3.79it/s]Epoch [9/18] Batch [30/120] Loss: 0.067236
 35%|└└└└        | 42/120 [00:11<00:20,  3.80it/s]Epoch [9/18] Batch [40/120] Loss: 0.053817
 43%|└└└└        | 52/120 [00:13<00:17,  3.82it/s]Epoch [9/18] Batch [50/120] Loss: 0.088420
 50%|└└└└└       | 60/120 [00:15<00:15,  3.84it/s]Epoch [9/18] Batch [60/120] Loss: 0.095163
 60%|└└└└└└      | 72/120 [00:18<00:12,  3.83it/s]Epoch [9/18] Batch [70/120] Loss: 0.134012
 68%|└└└└└└└     | 82/120 [00:21<00:09,  3.82it/s]Epoch [9/18] Batch [80/120] Loss: 0.091482
 77%|└└└└└└└└    | 92/120 [00:24<00:07,  3.84it/s]Epoch [9/18] Batch [90/120] Loss: 0.131734
 85%|└└└└└└└└└   | 102/120 [00:26<00:04,  3.68it/s]Epoch [9/18] Batch [100/120] Loss: 0.050386
 93%|└└└└└└└└└└  | 112/120 [00:29<00:02,  3.84it/s]Epoch [9/18] Batch [110/120] Loss: 0.131303
100%|└└└└└└└└└└└| 120/120 [00:31<00:00,  3.81it/s]
🔥 Epoch [9/18] - Average Loss: 0.101738
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_9.pt

🔥 Epoch [10/18] started...
  1%|           | 1/120 [00:00<00:30,  3.90it/s]Epoch [10/18] Batch [0/120] Loss: 0.087802
 10%|└           | 12/120 [00:03<00:27,  3.86it/s]Epoch [10/18] Batch [10/120] Loss: 0.055338
 18%|└└          | 21/120 [00:05<00:25,  3.83it/s]Epoch [10/18] Batch [20/120] Loss: 0.061860
 27%|└└└         | 32/120 [00:08<00:23,  3.77it/s]Epoch [10/18] Batch [30/120] Loss: 0.135492
 35%|└└└└        | 42/120 [00:11<00:20,  3.81it/s]Epoch [10/18] Batch [40/120] Loss: 0.093798
 43%|└└└└        | 52/120 [00:13<00:17,  3.83it/s]Epoch [10/18] Batch [50/120] Loss: 0.083405
 51%|└└└└└       | 61/120 [00:15<00:15,  3.88it/s]Epoch [10/18] Batch [60/120] Loss: 0.054370
 59%|└└└└└└      | 71/120 [00:18<00:12,  3.78it/s]Epoch [10/18] Batch [70/120] Loss: 0.100350
 68%|└└└└└└└     | 82/120 [00:21<00:09,  3.83it/s]Epoch [10/18] Batch [80/120] Loss: 0.115565
 76%|└└└└└└└└    | 91/120 [00:23<00:07,  3.85it/s]Epoch [10/18] Batch [90/120] Loss: 0.065333
 85%|└└└└└└└└└   | 102/120 [00:26<00:04,  3.76it/s]Epoch [10/18] Batch [100/120] Loss: 0.148383
 93%|└└└└└└└└└└  | 112/120 [00:29<00:02,  3.81it/s]Epoch [10/18] Batch [110/120] Loss: 0.105601
100%|└└└└└└└└└└└| 120/120 [00:31<00:00,  3.81it/s]
🔥 Epoch [10/18] - Average Loss: 0.102397
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_10.pt

🔥 Epoch [11/18] started...
  1%|           | 1/120 [00:00<00:31,  3.78it/s]Epoch [11/18] Batch [0/120] Loss: 0.070927
  9%|└           | 11/120 [00:02<00:28,  3.84it/s]Epoch [11/18] Batch [10/120] Loss: 0.099525
 18%|└└          | 22/120 [00:05<00:25,  3.84it/s]Epoch [11/18] Batch [20/120] Loss: 0.115245
 26%|└└└         | 31/120 [00:08<00:23,  3.82it/s]Epoch [11/18] Batch [30/120] Loss: 0.064922
 35%|└└└└        | 42/120 [00:11<00:20,  3.84it/s]Epoch [11/18] Batch [40/120] Loss: 0.066739
 42%|└└└└        | 51/120 [00:13<00:18,  3.81it/s]Epoch [11/18] Batch [50/120] Loss: 0.077645
 50%|└└└└└       | 60/120 [00:15<00:15,  3.77it/s]Epoch [11/18] Batch [60/120] Loss: 0.103107
 59%|└└└└└└      | 71/120 [00:18<00:12,  3.80it/s]Epoch [11/18] Batch [70/120] Loss: 0.060720
```

```
 68%|██████▊   | 81/120 [00:21<00:10,  3.82it/s]Epoch [11/18] Batch [80/120] Loss: 0.126032
 77%|███████▋  | 92/120 [00:24<00:07,  3.87it/s]Epoch [11/18] Batch [90/120] Loss: 0.120720
 84%|████████▍ | 101/120 [00:26<00:04,  3.87it/s]Epoch [11/18] Batch [100/120] Loss: 0.116880
 92%|█████████▏| 110/120 [00:28<00:02,  3.78it/s]Epoch [11/18] Batch [110/120] Loss: 0.100098
100%|██████████| 120/120 [00:31<00:00,  3.81it/s]
🔥 Epoch [11/18] - Average Loss: 0.101586
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_11.pt

🔥 Epoch [12/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [12/18] Batch [0/120] Loss: 0.093287
  9%|▉         | 11/120 [00:02<00:28,  3.83it/s]Epoch [12/18] Batch [10/120] Loss: 0.140290
 18%|█▊        | 22/120 [00:05<00:25,  3.80it/s]Epoch [12/18] Batch [20/120] Loss: 0.125995
 27%|██▋       | 32/120 [00:08<00:23,  3.79it/s]Epoch [12/18] Batch [30/120] Loss: 0.113982
 34%|███▍      | 41/120 [00:10<00:20,  3.83it/s]Epoch [12/18] Batch [40/120] Loss: 0.069661
 43%|████▎     | 52/120 [00:13<00:18,  3.76it/s]Epoch [12/18] Batch [50/120] Loss: 0.123562
 52%|█████▏    | 62/120 [00:16<00:15,  3.80it/s]Epoch [12/18] Batch [60/120] Loss: 0.083680
 57%|█████▋    | 68/120 [00:17<00:13,  3.85it/s] Keeping Kaggle active...
 60%|██████    | 72/120 [00:18<00:12,  3.78it/s]Epoch [12/18] Batch [70/120] Loss: 0.114531
 67%|██████▋   | 80/120 [00:21<00:10,  3.82it/s]Epoch [12/18] Batch [80/120] Loss: 0.148731
 77%|███████▋  | 92/120 [00:24<00:07,  3.76it/s]Epoch [12/18] Batch [90/120] Loss: 0.116640
 85%|████████▌ | 102/120 [00:26<00:04,  3.82it/s]Epoch [12/18] Batch [100/120] Loss: 0.125652
 92%|█████████▎| 111/120 [00:29<00:02,  3.72it/s]Epoch [12/18] Batch [110/120] Loss: 0.086764
100%|██████████| 120/120 [00:31<00:00,  3.79it/s]
🔥 Epoch [12/18] - Average Loss: 0.097510
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_12.pt

🔥 Epoch [13/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [13/18] Batch [0/120] Loss: 0.058665
  9%|▉         | 11/120 [00:02<00:28,  3.82it/s]Epoch [13/18] Batch [10/120] Loss: 0.086178
 18%|█▊        | 21/120 [00:05<00:26,  3.78it/s]Epoch [13/18] Batch [20/120] Loss: 0.055099
 26%|██▌       | 31/120 [00:08<00:23,  3.83it/s]Epoch [13/18] Batch [30/120] Loss: 0.125803
 35%|███▌      | 42/120 [00:11<00:20,  3.81it/s]Epoch [13/18] Batch [40/120] Loss: 0.051428
 42%|████▎     | 51/120 [00:13<00:18,  3.83it/s]Epoch [13/18] Batch [50/120] Loss: 0.147917
 52%|█████▏    | 62/120 [00:16<00:15,  3.86it/s]Epoch [13/18] Batch [60/120] Loss: 0.100568
 60%|██████    | 72/120 [00:18<00:12,  3.79it/s]Epoch [13/18] Batch [70/120] Loss: 0.090877
 68%|██████▊   | 82/120 [00:21<00:10,  3.73it/s]Epoch [13/18] Batch [80/120] Loss: 0.112474
 76%|███████▌  | 91/120 [00:23<00:07,  3.80it/s]Epoch [13/18] Batch [90/120] Loss: 0.068883
 84%|████████▍ | 101/120 [00:26<00:05,  3.76it/s]Epoch [13/18] Batch [100/120] Loss: 0.068966
 93%|█████████▎| 112/120 [00:29<00:02,  3.80it/s]Epoch [13/18] Batch [110/120] Loss: 0.145712
100%|██████████| 120/120 [00:31<00:00,  3.78it/s]
🔥 Epoch [13/18] - Average Loss: 0.094496
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_13.pt

🔥 Epoch [14/18] started...
  1%|          | 1/120 [00:00<00:30,  3.85it/s]Epoch [14/18] Batch [0/120] Loss: 0.121470
  9%|▉         | 11/120 [00:02<00:28,  3.79it/s]Epoch [14/18] Batch [10/120] Loss: 0.097145
 18%|█▊        | 21/120 [00:05<00:26,  3.73it/s]Epoch [14/18] Batch [20/120] Loss: 0.129734
 27%|██▋       | 32/120 [00:08<00:22,  3.85it/s]Epoch [14/18] Batch [30/120] Loss: 0.086221
 33%|███▎      | 40/120 [00:10<00:21,  3.77it/s]Epoch [14/18] Batch [40/120] Loss: 0.099285
 42%|████▎     | 51/120 [00:13<00:18,  3.77it/s]Epoch [14/18] Batch [50/120] Loss: 0.113286
 52%|█████▏    | 62/120 [00:16<00:15,  3.81it/s]Epoch [14/18] Batch [60/120] Loss: 0.084221
 60%|██████    | 72/120 [00:18<00:12,  3.80it/s]Epoch [14/18] Batch [70/120] Loss: 0.122193
 68%|██████▊   | 81/120 [00:21<00:10,  3.79it/s]Epoch [14/18] Batch [80/120] Loss: 0.148123
 77%|███████▋  | 92/120 [00:24<00:07,  3.88it/s]Epoch [14/18] Batch [90/120] Loss: 0.064087
 85%|████████▌ | 102/120 [00:26<00:04,  3.76it/s]Epoch [14/18] Batch [100/120] Loss: 0.082818
 93%|█████████▎| 112/120 [00:29<00:02,  3.78it/s]Epoch [14/18] Batch [110/120] Loss: 0.076075
100%|██████████| 120/120 [00:31<00:00,  3.78it/s]
🔥 Epoch [14/18] - Average Loss: 0.104918
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_14.pt

🔥 Epoch [15/18] started...
  1%|          | 1/120 [00:00<00:31,  3.74it/s]Epoch [15/18] Batch [0/120] Loss: 0.135729
  9%|▉         | 11/120 [00:02<00:28,  3.85it/s]Epoch [15/18] Batch [10/120] Loss: 0.070518
 18%|█▊        | 22/120 [00:05<00:25,  3.82it/s]Epoch [15/18] Batch [20/120] Loss: 0.102261
 26%|██▌       | 31/120 [00:08<00:23,  3.74it/s]Epoch [15/18] Batch [30/120] Loss: 0.121652
 35%|███▌      | 42/120 [00:11<00:20,  3.74it/s]Epoch [15/18] Batch [40/120] Loss: 0.138741
 43%|████▎     | 52/120 [00:13<00:18,  3.77it/s]Epoch [15/18] Batch [50/120] Loss: 0.101584
 52%|█████▏    | 62/120 [00:16<00:15,  3.80it/s]Epoch [15/18] Batch [60/120] Loss: 0.065157
 58%|█████▊    | 70/120 [00:18<00:13,  3.75it/s]Epoch [15/18] Batch [70/120] Loss: 0.062211
 67%|██████▋   | 80/120 [00:21<00:10,  3.71it/s]Epoch [15/18] Batch [80/120] Loss: 0.141489
 77%|███████▋  | 92/120 [00:24<00:07,  3.80it/s]Epoch [15/18] Batch [90/120] Loss: 0.066562
 85%|████████▌ | 102/120 [00:26<00:04,  3.80it/s]Epoch [15/18] Batch [100/120] Loss: 0.147642
 93%|█████████▎| 112/120 [00:29<00:02,  3.78it/s]Epoch [15/18] Batch [110/120] Loss: 0.067240
100%|██████████| 120/120 [00:31<00:00,  3.78it/s]
🔥 Epoch [15/18] - Average Loss: 0.101224
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_15.pt

🔥 Epoch [16/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [16/18] Batch [0/120] Loss: 0.141538
  9%|▉         | 11/120 [00:02<00:28,  3.78it/s]Epoch [16/18] Batch [10/120] Loss: 0.061888
 18%|█▊        | 22/120 [00:05<00:26,  3.76it/s]Epoch [16/18] Batch [20/120] Loss: 0.142316
 27%|██▋       | 32/120 [00:08<00:23,  3.72it/s]Epoch [16/18] Batch [30/120] Loss: 0.065524
 34%|███▍      | 41/120 [00:10<00:20,  3.82it/s]Epoch [16/18] Batch [40/120] Loss: 0.054846
 43%|████▎     | 52/120 [00:13<00:18,  3.77it/s]Epoch [16/18] Batch [50/120] Loss: 0.116026
 52%|█████▏    | 62/120 [00:16<00:15,  3.65it/s]Epoch [16/18] Batch [60/120] Loss: 0.087543
 59%|█████▉    | 71/120 [00:18<00:13,  3.76it/s]Epoch [16/18] Batch [70/120] Loss: 0.132308
 67%|██████▋   | 80/120 [00:21<00:10,  3.69it/s]Epoch [16/18] Batch [80/120] Loss: 0.148210
 77%|███████▋  | 92/120 [00:24<00:07,  3.81it/s]Epoch [16/18] Batch [90/120] Loss: 0.141109
 85%|████████▌ | 102/120 [00:27<00:04,  3.76it/s]Epoch [16/18] Batch [100/120] Loss: 0.136500
 92%|█████████▏| 110/120 [00:29<00:02,  3.63it/s]Epoch [16/18] Batch [110/120] Loss: 0.077484
```

```
100%|██████████| 120/120 [00:31<00:00,  3.76it/s]
🔥 Epoch [16/18] - Average Loss: 0.100728
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_16.pt

🔥 Epoch [17/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [17/18] Batch [0/120] Loss: 0.137404
  9%|▉          | 11/120 [00:02<00:28,  3.82it/s]Epoch [17/18] Batch [10/120] Loss: 0.075536
 18%|▉          | 21/120 [00:05<00:26,  3.76it/s]Epoch [17/18] Batch [20/120] Loss: 0.137714
 26%|▉          | 31/120 [00:08<00:23,  3.75it/s]Epoch [17/18] Batch [30/120] Loss: 0.083231
 34%|▉          | 41/120 [00:10<00:21,  3.76it/s]Epoch [17/18] Batch [40/120] Loss: 0.113270
 42%|▉          | 51/120 [00:13<00:18,  3.82it/s]Epoch [17/18] Batch [50/120] Loss: 0.081391
 50%|▉          | 60/120 [00:15<00:15,  3.79it/s]Epoch [17/18] Batch [60/120] Loss: 0.116370
 60%|▉          | 72/120 [00:19<00:12,  3.77it/s]Epoch [17/18] Batch [70/120] Loss: 0.136960
 68%|▉          | 81/120 [00:21<00:10,  3.79it/s]Epoch [17/18] Batch [80/120] Loss: 0.079089
 76%|▉          | 91/120 [00:24<00:07,  3.82it/s]Epoch [17/18] Batch [90/120] Loss: 0.122292
 85%|▉          | 102/120 [00:27<00:04,  3.71it/s]Epoch [17/18] Batch [100/120] Loss: 0.132554
 93%|▉          | 112/120 [00:29<00:02,  3.76it/s]Epoch [17/18] Batch [110/120] Loss: 0.070142
100%|██████████| 120/120 [00:31<00:00,  3.76it/s]
🔥 Epoch [17/18] - Average Loss: 0.098900
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_17.pt

🔥 Epoch [18/18] started...
  0%|          | 0/120 [00:00<?, ?it/s]Epoch [18/18] Batch [0/120] Loss: 0.132282
 10%|▉          | 12/120 [00:03<00:28,  3.76it/s]Epoch [18/18] Batch [10/120] Loss: 0.134403
 18%|▉          | 22/120 [00:05<00:25,  3.78it/s]Epoch [18/18] Batch [20/120] Loss: 0.093459
 26%|▉          | 31/120 [00:08<00:23,  3.74it/s]Epoch [18/18] Batch [30/120] Loss: 0.113691
 35%|▉          | 42/120 [00:11<00:20,  3.82it/s]Epoch [18/18] Batch [40/120] Loss: 0.070989
 42%|▉          | 51/120 [00:13<00:18,  3.75it/s]Epoch [18/18] Batch [50/120] Loss: 0.115354
 52%|▉          | 62/120 [00:16<00:15,  3.79it/s]Epoch [18/18] Batch [60/120] Loss: 0.126488
 60%|▉          | 72/120 [00:19<00:12,  3.80it/s]Epoch [18/18] Batch [70/120] Loss: 0.053077
 68%|▉          | 81/120 [00:21<00:10,  3.81it/s]Epoch [18/18] Batch [80/120] Loss: 0.055663
 77%|▉          | 92/120 [00:24<00:07,  3.80it/s]Epoch [18/18] Batch [90/120] Loss: 0.068961
 84%|▉          | 101/120 [00:26<00:04,  3.80it/s]Epoch [18/18] Batch [100/120] Loss: 0.139920
 93%|▉          | 112/120 [00:29<00:02,  3.77it/s]Epoch [18/18] Batch [110/120] Loss: 0.079045
100%|██████████| 120/120 [00:31<00:00,  3.77it/s]🔥 Epoch [18/18] - Average Loss: 0.101883
✅ Model saved at /kaggle/working/checkpoints/unet_epoch_18.pt

🎯 Training completed successfully!
```

```
# ✅ Plotting
plt.figure(figsize=(12, 5))
plt.plot(epochs, loss, marker='o', color='red', label='Loss')

# ✅ Labels and Title
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.title('Model Loss Over Epochs')
plt.legend()
plt.grid(True)

# ✅ Show Plot
plt.show()
```




```
import h5py
import numpy as np
import os
from PIL import Image

# ✅ HDF5 File Path
file_path = '/kaggle/input/dataset-hdf5/quark-gluon_data-set_n139306.hdf5'
output_dir = '/kaggle/working/extracted_images'  # ✅ Output directory for saved images
os.makedirs(output_dir, exist_ok=True)

# ✅ Extract and Save Individual Images
def extract_images(file_path, output_dir, num_images=10):
    """🔥 Extract and save images from X_jets."""

    with h5py.File(file_path, 'r') as f:
        # ✅ Load the image dataset
        X_jets = f['X_jets'][:]

        print(f"✅ Extracting {num_images} images...")

        for i in range(min(num_images, X_jets.shape[0])):
            # ✅ Select image (shape: 125x125x3)
            image = X_jets[i]

            # ✅ Normalize the image to [0, 255] and convert to uint8
            img_normalized = ((image - image.min()) / (image.max() - image.min()) * 255).astype(np.uint8)

            # ✅ Convert to PIL image
            img_pil = Image.fromarray(img_normalized)

            # ✅ Save the image
            img_path = os.path.join(output_dir, f"image_{i + 1}.png")
            img_pil.save(img_path)

            print(f"🔥 Saved {img_path}")

        print("✅ Image extraction completed!")

# ✅ Extract 10 sample images
extract_images(file_path, output_dir, num_images=10)
```

```
✅ Extracting 10 images...
🔥 Saved /kaggle/working/extracted_images/image_1.png
🔥 Saved /kaggle/working/extracted_images/image_2.png
🔥 Saved /kaggle/working/extracted_images/image_3.png
🔥 Saved /kaggle/working/extracted_images/image_4.png
🔥 Saved /kaggle/working/extracted_images/image_5.png
🔥 Saved /kaggle/working/extracted_images/image_6.png
🔥 Saved /kaggle/working/extracted_images/image_7.png
🔥 Saved /kaggle/working/extracted_images/image_8.png
🔥 Saved /kaggle/working/extracted_images/image_9.png
🔥 Saved /kaggle/working/extracted_images/image_10.png
✅ Image extraction completed!
```

```
import os
import torch
from PIL import Image
from torchvision import transforms
from torch.autograd import no_grad

# ✅ Configurations
device = "cuda" if torch.cuda.is_available() else "cpu"
model_path = "/kaggle/working/checkpoints/ldm_best_model.pt"
input_dir = "/kaggle/input/images"
output_dir = "/kaggle/working/generated_images"

# ✅ Ensure output directory exists
os.makedirs(output_dir, exist_ok=True)

# ✅ Load LDM Model
print("🚀 Loading LDM model...")
model = torch.load(model_path, map_location=device)
model.eval()
model.to(device)
print("✅ Model loaded successfully!")

# ✅ Image Transformation Pipeline
transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
    transforms.Normalize((0.5,), (0.5,))
])

# ✅ Load and process images
image_files = sorted([f for f in os.listdir(input_dir) if f.endswith(".png")])
print(f"🔥 Found {len(image_files)} images.")

# ✅ Generate images
with no_grad():
    for idx, image_file in enumerate(image_files, start=1):
        img_path = os.path.join(input_dir, image_file)
        img = Image.open(img_path).convert("RGB")
        img_tensor = transform(img).unsqueeze(0).to(device)

        # ✅ Generate output
        output = model(img_tensor)
        output_img = (output.squeeze(0).cpu().detach().clamp(-1, 1) + 1) / 2  # Denormalize
        output_img = transforms.ToPILImage()(output_img)

        # ✅ Save output image
        output_path = os.path.join(output_dir, f"gen_{idx}.jpg")
        output_img.save(output_path)

        print(f"✅ Saved: {output_path}")

print("🎯 All images generated successfully!")
```

```
🚀 Loading LDM model...
✅ Model loaded successfully!
🔥 Found 10 images.
✅ Saved: /kaggle/working/generated_images/gen_1.jpg
✅ Saved: /kaggle/working/generated_images/gen_2.jpg
✅ Saved: /kaggle/working/generated_images/gen_3.jpg
✅ Saved: /kaggle/working/generated_images/gen_4.jpg
✅ Saved: /kaggle/working/generated_images/gen_5.jpg
✅ Saved: /kaggle/working/generated_images/gen_6.jpg
✅ Saved: /kaggle/working/generated_images/gen_7.jpg
✅ Saved: /kaggle/working/generated_images/gen_8.jpg
✅ Saved: /kaggle/working/generated_images/gen_9.jpg
✅ Saved: /kaggle/working/generated_images/gen_10.jpg
🎯 All images generated successfully!
```

```
!pip install lpips scikit-image tqdm
```

```
Collecting lpips
  Downloading lpips-0.1.4-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: scikit-image in /usr/local/lib/python3.10/dist-packages (0.25.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (4.67.1)
Requirement already satisfied: torch>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from lpips) (2.6.0)
Requirement already satisfied: torchvision>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from lpips) (0.15.2+cu118)
Requirement already satisfied: numpy>=1.14.3 in /usr/local/lib/python3.10/dist-packages (from lpips) (1.26.4)
Requirement already satisfied: scipy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from lpips) (1.13.1)
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (3.4.2)
Requirement already satisfied: pillow>=10.1 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (11.0.0)
Requirement already satisfied: imageio!=2.35.0,>=2.33 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (2.36.1)
Requirement already satisfied: tifffile>=2022.8.12 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (2024.12.12)
Requirement already satisfied: packaging>=21 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (24.2)
Requirement already satisfied: lazy-loader>=0.4 in /usr/local/lib/python3.10/dist-packages (from scikit-image) (0.4)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (2025.0.1)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (2022.0.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.10/dist-packages (from numpy>=1.14.3->lpips) (2.4.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (3.17.0)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (4
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (2024.12.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.4.127 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lp
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0-
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lp
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips)
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips)
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpip
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpip
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lp
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (2
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpi
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch>=0.4.0->lpips) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch>=0.4.0->l
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from torchvision>=0.2.1->lpips) (2.32.3)
Collecting torch>=0.4.0 (from lpips)
  Downloading torch-2.0.1-cp310-cp310-manylinux1_x86_64.whl.metadata (24 kB)
Collecting nvidia-cuda-nvrtc-cu11==11.7.99 (from torch>=0.4.0->lpips)
  Downloading nvidia_cuda_nvrtc_cu11-11.7.99-2-py3-none-manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cuda-runtime-cu11==11.7.99 (from torch>=0.4.0->lpips)
  Downloading nvidia_cuda_runtime_cu11-11.7.99-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cuda-cupti-cu11==11.7.101 (from torch>=0.4.0->lpips)
  Downloading nvidia_cuda_cupti_cu11-11.7.101-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cudnn-cu11==8.5.0.96 (from torch>=0.4.0->lpips)
  Downloading nvidia_cudnn_cu11-8.5.0.96-2-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cublas-cu11==11.10.3.66 (from torch>=0.4.0->lpips)
  Downloading nvidia_cublas_cu11-11.10.3.66-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cufft-cu11==10.9.0.58 (from torch>=0.4.0->lpips)
  Downloading nvidia_cufft_cu11-10.9.0.58-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu11==10.2.10.91 (from torch>=0.4.0->lpips)
  Downloading nvidia_curand_cu11-10.2.10.91-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
```

```
import os
from PIL import Image
import matplotlib.pyplot as plt

# ✅ Configurations
og_dir = "/kaggle/input/images"  # Original images folder
recon_dir = "/kaggle/input/generated-images"  # Reconstructed images folder

# ✅ Load Image Pairs
og_images = sorted([f for f in os.listdir(og_dir) if f.endswith(('.png', '.jpg'))])
recon_images = sorted([f for f in os.listdir(recon_dir) if f.endswith(('.png', '.jpg'))])

if len(og_images) != len(recon_images):
    print("⚠️ Image count mismatch between original and reconstructed folders!")
    exit(1)

# ✅ Display Images Side by Side
print(f"🔥 Found {len(og_images)} image pairs for display.")

fig, axes = plt.subplots(len(og_images), 2, figsize=(12, 4 * len(og_images)))

for idx, (og_img, recon_img) in enumerate(zip(og_images, recon_images)):
    og_path = os.path.join(og_dir, og_img)
    recon_path = os.path.join(recon_dir, recon_img)

    # ✅ Load Images
    og = Image.open(og_path)
    recon = Image.open(recon_path)
```