

```

import torch
from diffusers import DiffusionPipeline

# switch to "mps" for apple devices
pipe = DiffusionPipeline.from_pretrained("zai-org/GLM-Image", dtype=torch.bfloat16, device_map="cuda")

prompt = "Astronaut in a jungle, cold color palette, muted colors, detailed, 8k"
image = pipe(prompt).images[0]

Flax classes are deprecated and will be removed in Diffusers v1.0.0. We recommend migrating to PyTorch classes or pinning you
Flax classes are deprecated and will be removed in Diffusers v1.0.0. We recommend migrating to PyTorch classes or pinning you
model_index.json: 100%
619/619 [00:00<00:00, 63.9kB/s]

-----
AttributeError                               Traceback (most recent call last)
/tmp/ipython-input-1678869869.py in <cell line: 0>()
      3
      4 # switch to "mps" for apple devices
----> 5 pipe = DiffusionPipeline.from_pretrained("zai-org/GLM-Image", dtype=torch.bfloat16, device_map="cuda")
      6
      7 prompt = "Astronaut in a jungle, cold color palette, muted colors, detailed, 8k"

      ▲ 5 frames
/usr/local/lib/python3.12/dist-packages/diffusers/utils/import_utils.py in __getattr__(self, name)
 1007         value = getattr(module, name)
 1008     else:
-> 1009         raise AttributeError(f"module {self.__name__} has no attribute {name}")
 1010
 1011     setattr(self, name, value)

AttributeError: module diffusers has no attribute GlmImagePipeline

```

Next steps: [Explain error](#)

`!pip install diffusers transformers accelerate torch`

```

Requirement already satisfied: diffusers in /usr/local/lib/python3.12/dist-packages (0.36.0)
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.57.3)
Requirement already satisfied: accelerate in /usr/local/lib/python3.12/dist-packages (1.12.0)
Requirement already satisfied: torch in /usr/local/lib/python3.12/dist-packages (2.9.0+cu126)
Requirement already satisfied: importlib_metadata in /usr/local/lib/python3.12/dist-packages (from diffusers) (8.7.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from diffusers) (3.20.2)
Requirement already satisfied: httpx<1.0.0 in /usr/local/lib/python3.12/dist-packages (from diffusers) (0.28.1)
Requirement already satisfied: huggingface-hub<2.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from diffusers) (0.36)
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages (from diffusers) (2.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from diffusers) (2025.11.3)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from diffusers) (2.32.4)
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.12/dist-packages (from diffusers) (0.7.0)
Requirement already satisfied: Pillow in /usr/local/lib/python3.12/dist-packages (from diffusers) (11.3.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.3)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.12/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: typing-extensions>=4.10.0 in /usr/local/lib/python3.12/dist-packages (from torch) (4.15.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch) (1.14.0)
Requirement already satisfied: networkx>=2.5.1 in /usr/local/lib/python3.12/dist-packages (from torch) (3.6.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec>=0.8.5 in /usr/local/lib/python3.12/dist-packages (from torch) (2025.3.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch) (9.10.2.2
Requirement already satisfied: nvidia-cUBLAS-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.4.1
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch) (11.3.0.4)
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch) (10.3.7.
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch) (11.7.1
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch) (12.5.4
Requirement already satisfied: nvidia-cusparseL-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch) (0.7.1)
Requirement already satisfied: nvidia-nccl-cu12==2.27.5 in /usr/local/lib/python3.12/dist-packages (from torch) (2.27.5)
Requirement already satisfied: nvidia-nvshmem-cu12==3.3.20 in /usr/local/lib/python3.12/dist-packages (from torch) (3.3.20)
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.77)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch) (12.6.8

```

```
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch) (1.11.1.6)
Requirement already satisfied: triton==3.5.0 in /usr/local/lib/python3.12/dist-packages (from torch) (3.5.0)
Requirement already satisfied: anyio in /usr/local/lib/python3.12/dist-packages (from httpx<1.0.0->diffusers) (4.12.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.12/dist-packages (from httpx<1.0.0->diffusers) (2026.1.4)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.12/dist-packages (from httpx<1.0.0->diffusers) (1.0.9)
Requirement already satisfied: idna in /usr/local/lib/python3.12/dist-packages (from httpx<1.0.0->diffusers) (3.11)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.12/dist-packages (from httpcore==1.*->httpx<1.0.0->diffuse)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<2.0,>=0.
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch) (1.3
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.12/dist-packages (from importlib_metadata->diffusers) (3.
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch) (3.0.3)
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->diffusers)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->diffusers) (2.5.
```

```
import torch
from diffusers import StableDiffusionPipeline

# Load Stable Diffusion model
pipe = StableDiffusionPipeline.from_pretrained(
    "runwayml/stable-diffusion-v1-5",
    torch_dtype=torch.float16
)

# Move model to GPU
pipe = pipe.to("cuda")

# Text prompt
prompt = "create an ice lattee"

# Generate image
image = pipe(prompt).images[0]

# Save output
image.save("astronaut_jungle.png")

# Display image
image
```

Loading pipeline components...: 100%

7/7 [00:21<00:00, 2.47s/it]

100%

50/50 [00:07<00:00, 6.27s/it]



▼ Task

Create a Python list of 10 lung disease names. Using the `StableDiffusionPipeline` model, generate 10 X-ray images for each disease, using an f-string prompt, and save them in separate subfolders for each disease. Confirm that all images are generated and saved correctly.

✓ Define Lung Diseases

Subtask:

Create a Python list containing 10 different lung disease names that will be used as prompts.

Reasoning: I need to create a Python list named `lung_diseases` and populate it with 10 distinct lung disease names as instructed.

```
lung_diseases = [
    'Pneumonia',
    'Bronchitis',
    'Asthma',
    'Emphysema',
    'Tuberculosis',
    'Cystic Fibrosis',
    'Lung Cancer',
    'Pulmonary Fibrosis',
    'Acute Respiratory Distress Syndrome (ARDS)',
    'Chronic Obstructive Pulmonary Disease (COPD)'
]
print(f"Created a list of {len(lung_diseases)} lung diseases.")
print(lung_diseases)
```

```
Created a list of 10 lung diseases.
['Pneumonia', 'Bronchitis', 'Asthma', 'Emphysema', 'Tuberculosis', 'Cystic Fibrosis', 'Lung Cancer', 'Pulmonary Fibrosis', 'A
```

✓ Initialize Image Generation Pipeline

Subtask:

Load the Stable Diffusion model using `DiffusionPipeline.from_pretrained` and move it to the GPU, similar to the working example provided, to prepare for image generation.

Reasoning: I need to load the Stable Diffusion model, move it to the GPU, and prepare it for image generation as per the instructions. This involves importing necessary libraries and configuring the pipeline.

```
import torch
from diffusers import StableDiffusionPipeline

# Load Stable Diffusion model
pipe = StableDiffusionPipeline.from_pretrained(
    "runwayml/stable-diffusion-v1-5",
    torch_dtype=torch.float16
)

# Move model to GPU
pipe = pipe.to("cuda")

print("Stable Diffusion model loaded and moved to GPU.")
```

Loading pipeline components...: 100%

7/7 [00:20<00:00, 2.39s/it]

Stable Diffusion model loaded and moved to GPU.

Start coding or generate with AI.

✓ Generate and Save Images

Subtask:

Iterate through each lung disease. For each disease, create a dedicated subfolder. Inside this subfolder, generate 10 X-ray images using an f-string prompt (e.g., 'generate Xray image of this {disease} disease'). Each image will be saved with a unique name in its respective disease subfolder.

Reasoning: I need to implement the logic for generating and saving X-ray images for each lung disease, following the instructions to create subfolders, construct prompts, and save images with unique filenames.

```
import os

# Create a base directory to store all generated images
base_dir = "generated_xrays"
os.makedirs(base_dir, exist_ok=True)
print(f"Base directory '{base_dir}' created or already exists.")

# Loop through each lung disease to generate images
for disease in lung_diseases:
    # Create a subfolder for each disease
    disease_folder = os.path.join(base_dir, disease.replace(' ', '_'))
    os.makedirs(disease_folder, exist_ok=True)
    print(f"Subfolder '{disease_folder}' created or already exists.")

    # Generate 10 images for each disease
    for i in range(10):
        # Construct the prompt using an f-string
        prompt = f"X-ray image of a human lung with {disease}"

        # Generate the image
        image = pipe(prompt).images[0]

        # Construct a unique filename for the image
        filename = f"{disease.replace(' ', '_')}_xray_{i+1}.png"
        filepath = os.path.join(disease_folder, filename)

        # Save the image
        image.save(filepath)
        # print(f"Saved: {filepath}")

    print(f"Successfully generated and saved 10 X-ray images for '{disease}'.")

print("Image generation and saving complete for all lung diseases.")
```



```
Base directory 'generated_xrays' created or already exists.  
Subfolder 'generated_xrays/Pneumonia' created or already exists.  
100% 50/50 [00:07<00:00, 6.76it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.63it/s]  
100% 50/50 [00:07<00:00, 6.41it/s]  
100% 50/50 [00:07<00:00, 6.41it/s]  
100% 50/50 [00:07<00:00, 6.56it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.77it/s]  
100% 50/50 [00:07<00:00, 6.77it/s]  
Successfully generated and saved 10 X-ray images for 'Pneumonia'.  
Subfolder 'generated_xrays/Bronchitis' created or already exists.  
100% 50/50 [00:07<00:00, 6.73it/s]  
100% 50/50 [00:07<00:00, 6.73it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.58it/s]  
100% 50/50 [00:07<00:00, 6.64it/s]  
100% 50/50 [00:07<00:00, 6.62it/s]  
100% 50/50 [00:07<00:00, 6.65it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.75it/s]  
Successfully generated and saved 10 X-ray images for 'Bronchitis'.  
Subfolder 'generated_xrays/Asthma' created or already exists.  
100% 50/50 [00:07<00:00, 6.76it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.63it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
Potential NSFW content was detected in one or more images. A black image will be returned instead. Try again with a differer  
Successfully generated and saved 10 X-ray images for 'Asthma'.  
Subfolder 'generated_xrays/Emphysema' created or already exists.  
100% 50/50 [00:07<00:00, 6.69it/s]  
Potential NSFW content was detected in one or more images. A black image will be returned instead. Try again with a differer  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.74it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.73it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
Potential NSFW content was detected in one or more images. A black image will be returned instead. Try again with a differer  
100% 50/50 [00:07<00:00, 6.69it/s]
```

```
100%                                         50/50 [00:07<00:00,  6.66it/s]
Successfully generated and saved 10 X-ray images for 'Emphysema'.
Subfolder 'generated_xrays/Tuberculosis' created or already exists.
100%                                         50/50 [00:07<00:00,  6.72it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.64it/s]
100%                                         50/50 [00:07<00:00,  6.63it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.71it/s]
100%                                         50/50 [00:07<00:00,  6.64it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
Successfully generated and saved 10 X-ray images for 'Tuberculosis'.
Subfolder 'generated_xrays/Cystic_Fibrosis' created or already exists.
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.66it/s]
100%                                         50/50 [00:07<00:00,  6.67it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.67it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
Successfully generated and saved 10 X-ray images for 'Cystic Fibrosis'.
Subfolder 'generated_xrays/Lung_Cancer' created or already exists.
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.65it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.73it/s]
100%                                         50/50 [00:07<00:00,  6.66it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.67it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
Successfully generated and saved 10 X-ray images for 'Lung Cancer'.
Subfolder 'generated_xrays/Pulmonary_Fibrosis' created or already exists.
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.67it/s]
100%                                         50/50 [00:07<00:00,  6.65it/s]
100%                                         50/50 [00:07<00:00,  6.66it/s]
100%                                         50/50 [00:07<00:00,  6.67it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.70it/s]
100%                                         50/50 [00:07<00:00,  6.64it/s]
100%                                         50/50 [00:07<00:00,  6.68it/s]
100%                                         50/50 [00:07<00:00,  6.69it/s]
```

Installing collected packages: torchxrayvision

Successfully installed torchxrayvision-1.4.0

```
import torch
import torchxrayvision as xrv
import torchvision.transforms as transforms
from PIL import Image

# Load pretrained DenseNet model
model = xrv.models.DenseNet(weights="densenet121-res224-all")
model.eval()

# Image preprocessing (important for X-ray)
transform = transforms.Compose([
    transforms.Resize(224),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.5], std=[0.5])
])

# Load chest X-ray image (grayscale)
img = Image.open("chest_xray.png").convert("L")
img = transform(img).unsqueeze(0) # shape: [1, 1, 224, 224]
```

```

with torch.no_grad():
    outputs = model(img)

# Disease labels
labels = xrv.datasets.default_pathologies

# Print predictions
print("Predicted disease probabilities:\n")
for disease, score in zip(labels, outputs[0]):
    print(f"{disease}: {float(score):.3f}")

```

```

from google.colab import drive
drive.mount('/content/drive')

```

Reasoning: The previous image generation encountered 'Potential NSFW content' errors, leading to black images. To address this, I will update the image generation code to include a negative prompt to guide the model away from inappropriate content and improve the quality of the generated X-ray images. I will also make the positive prompt more explicit about the desired output.

```

import os

# Create a base directory to store all generated images
base_dir = "generated_xrays"
os.makedirs(base_dir, exist_ok=True)
print(f"Base directory '{base_dir}' created or already exists.")

# Define a negative prompt to avoid NSFW content and improve image quality
negative_prompt = "nudity, sex, porn, cartoon, painting, illustration, low quality, bad quality, blurry, distorted, unnatural"

# Loop through each lung disease to generate images
for disease in lung_diseases:
    # Create a subfolder for each disease
    disease_folder = os.path.join(base_dir, disease.replace(' ', '_'))
    os.makedirs(disease_folder, exist_ok=True)
    print(f"Subfolder '{disease_folder}' created or already exists.")

    # Generate 10 images for each disease
    for i in range(10):
        # Construct the prompt using an f-string
        prompt = f"Medical X-ray image of a human chest, lung with {disease}, detailed, diagnostic quality, grey scale"

        # Generate the image with a negative prompt
        image = pipe(prompt, negative_prompt=negative_prompt).images[0]

        # Construct a unique filename for the image
        filename = f"{disease.replace(' ', '_')}xray_{i+1}.png"
        filepath = os.path.join(disease_folder, filename)

        # Save the image
        image.save(filepath)
        # print(f"Saved: {filepath}")

    print(f"Successfully generated and saved 10 X-ray images for '{disease}'.")

print("Image generation and saving complete for all lung diseases.")

```



```
Base directory 'generated_xrays' created or already exists.  
Subfolder 'generated_xrays/Pneumonia' created or already exists.  
100% 50/50 [00:08<00:00, 5.96it/s]  
100% 50/50 [00:08<00:00, 6.17it/s]  
100% 50/50 [00:09<00:00, 6.34it/s]  
100% 50/50 [00:08<00:00, 6.38it/s]  
100% 50/50 [00:07<00:00, 6.59it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.79it/s]  
100% 50/50 [00:08<00:00, 6.75it/s]  
100% 50/50 [00:07<00:00, 6.76it/s]  
Successfully generated and saved 10 X-ray images for 'Pneumonia'.  
Subfolder 'generated_xrays/Bronchitis' created or already exists.  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.55it/s]  
100% 50/50 [00:07<00:00, 6.60it/s]  
100% 50/50 [00:07<00:00, 6.60it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:08<00:00, 4.19it/s]  
100% 50/50 [00:07<00:00, 6.73it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
Successfully generated and saved 10 X-ray images for 'Bronchitis'.  
Subfolder 'generated_xrays/Asthma' created or already exists.  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.60it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:08<00:00, 6.18it/s]  
100% 50/50 [00:07<00:00, 6.43it/s]  
100% 50/50 [00:08<00:00, 6.22it/s]  
100% 50/50 [00:08<00:00, 3.98it/s]  
100% 50/50 [00:10<00:00, 6.71it/s]  
Successfully generated and saved 10 X-ray images for 'Asthma'.  
Subfolder 'generated_xrays/Emphysema' created or already exists.  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.65it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
Successfully generated and saved 10 X-ray images for 'Emphysema'.
```

```
Subfolder 'generated_xrays/Tuberculosis' created or already exists.  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:08<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.65it/s]  
100% 50/50 [00:07<00:00, 6.63it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]
```

Successfully generated and saved 10 X-ray images for 'Tuberculosis'.
Subfolder 'generated_xrays/Cystic_Fibrosis' created or already exists.

```
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:08<00:00, 5.78it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.71it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]
```

Successfully generated and saved 10 X-ray images for 'Cystic Fibrosis'.
Subfolder 'generated_xrays/Lung_Cancer' created or already exists.

```
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.63it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.72it/s]  
100% 50/50 [00:07<00:00, 6.64it/s]
```

Successfully generated and saved 10 X-ray images for 'Lung Cancer'.
Subfolder 'generated_xrays/Pulmonary_Fibrosis' created or already exists.

```
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.65it/s]  
100% 50/50 [00:07<00:00, 6.62it/s]  
100% 50/50 [00:07<00:00, 6.70it/s]  
100% 50/50 [00:07<00:00, 6.65it/s]  
100% 50/50 [00:07<00:00, 6.69it/s]  
100% 50/50 [00:07<00:00, 6.68it/s]  
100% 50/50 [00:07<00:00, 6.67it/s]  
100% 50/50 [00:07<00:00, 6.66it/s]
```

Successfully generated and saved 10 X-ray images for 'Pulmonary Fibrosis'.
Subfolder 'generated_xrays/Acute_Respiratory_Distress_Syndrome_(ARDS)' created or already exists.

Start coding or generate with AI.

100%

50/50 [00:07<00:00, 0.04it/s]

Start coding or generate with AI.

100%

50/50 [00:07<00:00, 6.65it/s]

Confirm that all X-ray images have been successfully generated and saved into their corresponding disease subfolders.

100%

50/50 [00:07<00:00, 6.72it/s]

Subtask:

100%

50/50 [00:07<00:00, 6.69it/s]

Verify the successful generation and saving of all X-ray images into their respective disease subfolders.

100%

50/50 [00:08<00:00, 6.37it/s]

Reasoning: To confirm the successful generation and saving of all X-ray images, I need to iterate through the expected directory structure and check for the existence of each subfolder and image files. Please note that the base directory already exists.

100%

50/50 [00:07<00:00, 6.69it/s]

```
import os

total_images_found = 0
missing_items = []

print("Verifying generated images...")

for disease in lung_diseases:
    # Construct the expected path to the disease subfolder
    disease_folder = os.path.join(base_dir, disease.replace(' ', '_'))

    # Check if the disease subfolder exists
    if not os.path.isdir(disease_folder):
        missing_items.append(f"Missing folder: {disease_folder}")
        continue

    # If the folder exists, check for each of the 10 images
    for i in range(10):
        filename = f"{disease.replace(' ', '_)}_xray_{i+1}.png"
        filepath = os.path.join(disease_folder, filename)

        # Check if the image file exists
        if not os.path.isfile(filepath):
            missing_items.append(f"Missing file: {filepath}")
        else:
            total_images_found += 1

    print(f"Total images found: {total_images_found}")

if missing_items:
    print("\n--- Verification Summary ---")
    print("The following folders/files were not found:")
    for item in missing_items:
        print(f"- {item}")
    print("-----")
    print("Verification failed: Some images or folders are missing.")
else:
    print("Verification successful: All X-ray images and their corresponding folders were found.")
```

Verifying generated images...

Total images found: 100

Verification successful: All X-ray images and their corresponding folders were found.

Summary:

Data Analysis Key Findings

- A list of 10 distinct lung disease names was successfully created and used as prompts for image generation.
- The `StableDiffusionPipeline` model (`runwayml/stable-diffusion-v1-5`) was successfully loaded and moved to the GPU, ready for image generation.
- During the initial image generation attempt, some diseases (`Asthma`, `Emphysema`, `Acute Respiratory Distress Syndrome (ARDS)`) triggered "Potential NSFW content" errors, leading to the generation of blank or black images.

- This issue was resolved by introducing a `negative_prompt` (e.g., "nudity, sex, porn, cartoon, painting, illustration, low quality, bad quality, blurry, distorted, unnatural") and refining the positive prompt to be more specific (e.g., "Medical X-ray image of a human chest, lung with {disease}, detailed, diagnostic quality, grey scale").
- Following the prompt refinement, 10 X-ray images were successfully generated and saved for each of the 10 lung diseases, totaling 100 images. Each disease received its own subfolder within the `generated_xrays` directory.
- A final verification confirmed that all 100 generated X-ray images and their corresponding disease subfolders were present and correctly saved.

Insights or Next Steps

- The quality and relevance of generated images from diffusion models are highly dependent on careful prompt engineering, including the use of both positive and negative prompts to guide the generation process effectively.
- The generated dataset of synthetic X-ray images could be used for various purposes, such as training or fine-tuning image classification models for lung disease diagnosis, or for educational and research purposes, provided the synthetic nature of the data is acknowledged.

Start coding or [generate](#) with AI.

Task

Create a fine-tuned DenseNet model for chest X-ray classification of the generated lung disease images, evaluate its performance, and summarize the entire process.

Load DenseNet Chest X-ray Classification Model

Subtask:

Import a pre-trained DenseNet model suitable for chest X-ray classification using `torchxrayvision` and prepare it for fine-tuning with 10 output classes.

Reasoning: I need to import the necessary libraries, load the pre-trained DenseNet model, and then modify its final classification layer to prepare it for fine-tuning with 10 output classes, as per the instructions.

```
import torch
import torchxrayvision as xrv
import torchvision.transforms as transforms
from PIL import Image
import torch.nn as nn # Import nn module

# Load pretrained DenseNet model
model = xrv.models.DenseNet(weights="densenet121-res224-all")
print(f"Original model classifier: {model.classifier}")

# Modify the final classification layer for 10 output classes
# Get the number of input features to the original classifier layer
num_ftrs = model.classifier.in_features

# Replace the classifier with a new linear layer for 10 classes
model.classifier = nn.Linear(num_ftrs, 10)

print(f"Modified model classifier for 10 classes: {model.classifier}")
print("DenseNet model loaded and prepared for fine-tuning with 10 output classes.")

Downloading weights...
If this fails you can run `wget https://github.com/mlmed/torchxrayvision/releases/download/v1/nih-pc-chex-mimic\_ch-google-ope
[...]
Original model classifier: Linear(in_features=1024, out_features=18, bias=True)
Modified model classifier for 10 classes: Linear(in_features=1024, out_features=10, bias=True)
DenseNet model loaded and prepared for fine-tuning with 10 output classes.
```

Reasoning: The DenseNet model has been loaded and prepared for fine-tuning. Now, I need to create a custom dataset and dataloader to handle the generated X-ray images. This involves defining image transformations, iterating through the subfolders to get image paths