```
pip install torch torchvision diffusers transformers p
```

Resources

You are not subscribed. Learn more

You currently have zero compute units available. Resources offered free of charge are not guaranteed. Purchase more units here.

At your current usage level, this runtime may last up to 1 hour 40 minutes.

Manage sessions

Want more memory and disk space? Upgrade to Colab Pro

Python 3 Google Compute Engine backend (GPU)

Showing resources from 9:39 PM to 10:12 PM

System RAM
4.3 / 12.7 GB

GPU RAM
3.9 / 15.0 GB

Disk
43.9 / 112.6 GB

```
Successfully installed nvidia-cublas-cu12-12.4.5
```

Start coding or generate with AI.

```
from diffusers import StableDiffusionPipeline
import torch

# Load Stable Diffusion model
model_id = "runwayml/stable-diffusion-v1-5"
pipe = StableDiffusionPipeline.from_pretrained(model_i
pipe.to("cuda" if torch.cuda.is_available() else "cpu"

# Define prompt
prompt = "A serene sunset over a futuristic city"

# Generate and save 3 images
for i in range(3):
    image = pipe(prompt).images[0]
    image.save(f"generated_image_{i+1}.png")
```

```
The cache for model files in Transformers v4.22.0
     0/0 [00:00<?, ?it/s]
/usr/local/lib/python3.11/dist-packages/huggingfac
The secret `HF_TOKEN` does not exist in your Colab
To authenticate with the Hugging Face Hub, create
You will be able to reuse this secret in all of yo
Please note that authentication is recommended but
  warnings.warn(
model_index.json: 100%  541/541 [00:00<00:00, 37.5kB/s]
Fetching 15 files: 100%  15/15 [00:31<00:00,  2.21s/it]
model.safetensors: 100%  492M/492M [00:10<00:00, 74.5MB/s
model.safetensors: 100%  1.22G/1.22G [00:17<00:00, 93.2MB/
special_tokens_map.json: 100%  472/472 [00:00<00:00, 9.55kB
config.json: 100%  4.72k/4.72k [00:00<00:00, 98.6kB/s]
scheduler_config.json: 100%  308/308 [00:00<00:00, 4.99kB/s]
config.json: 100%  617/617 [00:00<00:00, 6.59kB/s]
merges.txt: 100%  525k/525k [00:00<00:00, 8.55MB/s]
preprocessor_config.json: 100%  342/342 [00:00<00:00, 5.59kB
diffusion_pytorch_model.safetensors: 100%  3.44G/3.44G [00:3
tokenizer_config.json: 100%  806/806 [00:00<00:00, 16.1kB/s]
vocab.json: 100%  1.06M/1.06M [00:00<00:00, 9.62MB/s]
config.json: 100%  743/743 [00:00<00:00, 9.60kB/s]
config.json: 100%  547/547 [00:00<00:00, 10.5kB/s]
diffusion_pytorch_model.safetensors: 100%  335M/335M [00:08
```

```
pip install opencv-python pillow torch torchvision
```

```
Requirement already satisfied: opencv-python in /u
Requirement already satisfied: pillow in /usr/loca
Requirement already satisfied: torch in /usr/local
Requirement already satisfied: torchvision in /usr
Requirement already satisfied: numpy>=1.21.2 in /u
Requirement already satisfied: filelock in /usr/lo
Requirement already satisfied: typing-extensions>=
Requirement already satisfied: networkx in /usr/lo
```

```
Requirement already satisfied: jinja2 in /usr/loca
Requirement already satisfied: fsspec in /usr/loca
Requirement already satisfied: nvidia-cuda-nvrtc-c
Requirement already satisfied: nvidia-cuda-runtime
Requirement already satisfied: nvidia-cuda-cupti-c
Requirement already satisfied: nvidia-cudnn-cu12==
Requirement already satisfied: nvidia-cublas-cu12=
Requirement already satisfied: nvidia-cufft-cu12==
Requirement already satisfied: nvidia-curand-cu12=
Requirement already satisfied: nvidia-cusolver-cu1
Requirement already satisfied: nvidia-cusparse-cu1
Requirement already satisfied: nvidia-nccl-cu12==2
Requirement already satisfied: nvidia-nvtx-cu12==1
Requirement already satisfied: nvidia-nvjitlink-cu
Requirement already satisfied: triton==3.1.0 in /u
Requirement already satisfied: sympy==1.13.1 in /u
Requirement already satisfied: mpmath<1.4,>=1.1.0
Requirement already satisfied: MarkupSafe>=2.0 in
```

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

# Define preprocessing steps
transform = transforms.Compose([
    transforms.Resize((224, 224)),  # Resize to 224x224
    transforms.ToTensor(),          # Convert to tensor
    transforms.Normalize(mean=[0.5], std=[0.5])  # Norma
])

# Load and process images
preprocessed_images = []
for i in range(3):
    img = Image.open(f"generated_image_{i+1}.png")
    img = transform(img)  # Apply transformations
    preprocessed_images.append(img)

# Save processed images for verification
for i, img_tensor in enumerate(preprocessed_images):
    img_pil = transforms.ToPILImage()(img_tensor)
    img_pil.save(f"preprocessed_image_{i+1}.png")
```

```
!apt update
!apt install -y julia
```

```
Get:1 https://cloud.r-project.org/bin/linux/ubuntu
Get:2 https://developer.download.nvidia.com/comput
Get:3 http://security.ubuntu.com/ubuntu jammy-secu
Hit:4 http://archive.ubuntu.com/ubuntu jammy InRel
Hit:5 https://ppa.launchpadcontent.net/deadsnakes/
```

```
Get:6 http://archive.ubuntu.com/ubuntu jammy-updat
Get:7 https://r2u.stat.illinois.edu/ubuntu jammy I
Hit:8 https://ppa.launchpadcontent.net/graphics-dr
Hit:9 https://ppa.launchpadcontent.net/ubuntugis/p
Get:10 https://r2u.stat.illinois.edu/ubuntu jammy/
Get:11 https://developer.download.nvidia.com/compu
Get:12 http://archive.ubuntu.com/ubuntu jammy-back
Get:13 http://security.ubuntu.com/ubuntu jammy-sec
Get:14 http://archive.ubuntu.com/ubuntu jammy-upda
Get:15 https://r2u.stat.illinois.edu/ubuntu jammy/
Get:16 http://security.ubuntu.com/ubuntu jammy-sec
Get:17 http://security.ubuntu.com/ubuntu jammy-sec
Get:18 http://archive.ubuntu.com/ubuntu jammy-upda
Get:19 http://archive.ubuntu.com/ubuntu jammy-upda
Fetched 29.2 MB in 8s (3,727 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
30 packages can be upgraded. Run 'apt list --upgra
W: Skipping acquire of configured file 'main/sourc
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package julia is not available, but is referred to
This may mean that the package is missing, has bee
is only available from another source

E: Package 'julia' has no installation candidate
```

```
!julia --version
```

```
julia version 1.10.8
```

```
!julia -e 'using Pkg; Pkg.add("IJulia")'
```

```
   Updating registry at `~/.julia/registries/Gene
  Resolving package versions...
 No Changes to `~/.julia/environments/v1.10/Proje
 No Changes to `~/.julia/environments/v1.10/Manif
```

```
!julia -e 'println("Hello, Julia in Colab!")'
```

```
Hello, Julia in Colab!
```

```
!jupyter kernelspec list
```

```
0.00s - Debugger warning: It seems that frozen mod
0.00s - make the debugger miss breakpoints. Please
```

```
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_D
Available kernels:
  ir         /usr/local/share/jupyter/kernels/ir
  julia      /usr/local/share/jupyter/kernels/juli
  python3    /usr/local/share/jupyter/kernels/pyth
```

```
!julia -e 'using Pkg; Pkg.add("IJulia"); Pkg.build("IJul
```

```
   Resolving package versions...
  No Changes to `~/.julia/environments/v1.10/Proje
  No Changes to `~/.julia/environments/v1.10/Manif
    Building Conda → `~/.julia/scratchspaces/44cf
    Building IJulia → `~/.julia/scratchspaces/44cf
```

```
!julia -e 'using Pkg; Pkg.add(["Flux", "Images", "ImageT
```

```
[17] macro expansion
   @ ./loading.jl:1860 [inlined]
 [18] macro expansion
   @ ./lock.jl:267 [inlined]
 [19] __require(into::Module, mod::Symbol)
   @ Base ./loading.jl:1823
 [20] #invoke_in_world#3
   @ ./essentials.jl:926 [inlined]
 [21] invoke_in_world
   @ ./essentials.jl:923 [inlined]
 [22] require(into::Module, mod::Symbol)
   @ Base ./loading.jl:1816
 [23] include
   @ ./Base.jl:495 [inlined]
 [24] include_package_for_output(pkg::Base.PkgId
   @ Base ./loading.jl:2292
 [25] top-level scope
   @ stdin:4
in expression starting at /root/.julia/packages/
in expression starting at stdin:4
```

```
!julia -e 'using Flux; println("Julia is working!")'
```

```
Julia is working!
```

```
!julia -e 'using Flux; println("Flux is installed and
```

```
Flux is installed and ready!
```

```
%%writefile model.jl
using Flux
using Images, ImageTransformations, FileIO

using Flux
using Images, ImageTransformations, FileIO

# Load and preprocess image
function load_image(image_path)
    img = load(image_path)                      # Load
    img = imresize(img, (224, 224))             # Resi
    img = Float32.(channelview(img))            # Conv
    img = permutedims(img, (2, 3, 1))           # Chan
    return reshape(img, 224, 224, 3, 1)         # Add
end

# Define a minimal Flux CNN model
model = Chain(
    Conv((3, 3), 3=>16, relu, pad=1),
    MaxPool((2, 2)),
    Conv((3, 3), 16=>32, relu, pad=1),
    MaxPool((2, 2)),
    Flux.flatten,                       # Fix for undef
```

```
    Dense(32 * 56 * 56, 64, relu),
    Dense(64, 10),                      # Remove softma
    softmax                             # Apply softmax
)

# Load a preprocessed image
image_path = "preprocessed_image_1.png"
input_image = load_image(image_path)

# Forward pass
output = model(input_image)

# Get predicted class
predicted_class = argmax(output)  # Get index of highe
println("Predicted Class: ", predicted_class)
```

```
Writing model.jl
```

```
!julia model.jl
```

```
Predicted Class: CartesianIndex(3, 1)
```

Change runtime type