txttoimage11/

├──API tạo ảnh từ văn bản.ipynb # File code huấn luyện mô hình

└── dataset/ # Thư mục chứa tập dữ liệu

├── naruto\_1.jpg

├── naruto\_1.txt

├── naruto\_2.jpg

├── naruto\_2.txt

└── …

Tôi hiểu bạn muốn xây dựng một API tạo ảnh từ văn bản mô tả, với yêu cầu ảnh chỉ chứa một vật thể duy nhất và không có nền. Bạn muốn sử dụng YOLO để phân tích đặc điểm của vật thể trong ảnh và huấn luyện mô hình Stable Diffusion để tạo ảnh từ văn bản.

Dưới đây là các bước chi tiết để thực hiện điều này:

**1. Tạo nhãn dữ liệu với YOLO:**

* **Cài đặt YOLOv5:**

Bash

!pip install torch torchvision torchaudio

!pip install yolov5

* **Tải mô hình YOLOv5:**

Python

import torch

model = torch.hub.load('ultralytics/yolov5', 'yolov5s') # Hoặc yolov5m, yolov5l, yolov5x

* **Phân tích ảnh và tạo nhãn:**

Python

import os

def analyze\_images(dataset\_path, model):

"""

Phân tích ảnh trong tập dữ liệu và tạo nhãn.

Args:

dataset\_path: Đường dẫn đến thư mục chứa tập dữ liệu.

model: Mô hình YOLOv5.

"""

for filename in os.listdir(dataset\_path):

if filename.endswith(('.jpg', '.jpeg', '.png')):

image\_path = os.path.join(dataset\_path, filename)1

results = model(image\_path)

# Trích xuất đặc điểm đối tượng

labels = []

for \*xyxy, conf, cls in results.xyxy[0]:

label = model.names[int(cls)]

labels.append(label)

# Lưu nhãn vào file txt cùng tên với ảnh

label\_path = os.path.splitext(image\_path)[0] + ".txt"

with open(label\_path, "w") as f:

f.write(", ".join(labels))

# Gọi hàm phân tích ảnh

analyze\_images(dataset\_path, model)

**2. Huấn luyện mô hình Stable Diffusion:**

* **Chuẩn bị dữ liệu:**
  + Tách vật thể khỏi nền và lưu ảnh với nền trong suốt (như đã đề cập ở câu trả lời trước).
  + Kết hợp nhãn từ YOLO vào văn bản mô tả. Ví dụ: nếu nhãn là "person, hat, red hair", văn bản mô tả sẽ là "a person wearing a hat with red hair, no background".
* **Huấn luyện mô hình:**

Python

import torch

from diffusers import StableDiffusionPipeline, DPMSolverMultistepScheduler

from datasets import load\_dataset

from transformers import TrainingArguments, Trainer

# Tải tập dữ liệu

dataset = load\_dataset("imagefolder", data\_dir=dataset\_path)

# Khởi tạo pipeline Stable Diffusion

pipe = StableDiffusionPipeline.from\_pretrained("stabilityai/stable-diffusion-2", torch\_dtype=torch.float16)

pipe.scheduler = DPMSolverMultistepScheduler.from\_config(pipe.scheduler.config)

pipe2 = pipe.to("cuda")

# Xử lý tập dữ liệu (bao gồm đọc nhãn từ file txt)

def preprocess\_function(examples):

images = [preprocess(image.convert("RGB")) for image in examples["image"]]

examples["pixel\_values"] = pipe.feature\_extractor(images=images, return\_tensors="pt").pixel\_values

# Đọc nhãn từ file txt

labels = []

for image\_path in examples["image"]:

label\_path = os.path.splitext(str(image\_path))[0] + ".txt"

with open(label\_path, "r") as f:

labels.append(f.read())

examples["labels"] = labels # Gán nhãn cho ảnh

return examples

dataset = dataset.map(preprocess\_function, batched=True)

# Tham số huấn luyện

training\_args = TrainingArguments(

output\_dir="path/to/save/model",

per\_device\_train\_batch\_size=4,

per\_device\_eval\_batch\_size=4,

learning\_rate=5e-6,

num\_train\_epochs=10,

save\_strategy="epoch",

push\_to\_hub=False,

)

# Huấn luyện mô hình

trainer = Trainer(

model=pipe,

args=training\_args,

train\_dataset=dataset["train"],

)

trainer.train()

# Lưu mô hình

pipe.save\_pretrained("path/to/save/model")

**3. Xử lý mô tả nhập vào:**

* **Phân tích mô tả:** Sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) để phân tích mô tả đầu vào và trích xuất các đặc điểm quan trọng.
* **Tạo prompt:** Kết hợp mô tả đầu vào với các từ khóa bổ sung (như "no background", "solo", "centered") để tạo prompt cho mô hình Stable Diffusion.

**4. Tạo API:**

* **Sử dụng Flask hoặc FastAPI:** Xây dựng API bằng Flask hoặc FastAPI để nhận đầu vào là văn bản mô tả và trả về ảnh được tạo ra bởi mô hình Stable Diffusion.

**Ví dụ API với Flask:**

Python

from flask import Flask, request, jsonify

from diffusers import StableDiffusionPipeline

app = Flask(\_\_name\_\_)

# Tải mô hình Stable Diffusion đã huấn luyện

pipe = StableDiffusionPipeline.from\_pretrained("path/to/save/model")

pipe = pipe.to("cuda")

@app.route('/generate', methods=['POST'])

def generate():

description = request.form['description']

# Xử lý mô tả và tạo prompt (ví dụ: thêm "no background")

prompt = f"{description}, no background"

image = pipe(prompt).images[0]

# Chuyển đổi image sang base64 hoặc lưu vào file

# ...

return jsonify({'image': image\_base64}) # Trả về ảnh

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

**Lưu ý:**

* Đây là một hướng dẫn chung. Bạn cần điều chỉnh code và các bước thực hiện cho phù hợp với tập dữ liệu và yêu cầu cụ thể của bạn.
* Việc huấn luyện mô hình AI yêu cầu tài nguyên tính toán lớn (GPU).
* Bạn cần có kiến thức về huấn luyện mô hình AI, xử lý ảnh, và NLP để thực hiện đầy đủ các bước này.

Chúc bạn thành công!