

## Процесс кодирования Python API для создания изображений искусственного интеллекта

*Беликов Андрей Геннадьевич*

*Приамурский государственный университет имени Шолом-Алейхема*

*Студент*

### **Аннотация**

В данной статье был рассмотрен процесс кодирования Python API для создания изображений искусственного интеллекта.

**Ключевые слова:** Stable Diffusion, ИИ, Python

### **The process of coding the Python API to create Artificial Intelligence images**

*Belikov Andrey Gennadievich*

*Sholom-Aleichem Priamursky State University*

*Student*

### **Abstract**

In this article, the process of coding the Python API for creating artificial intelligence images was considered.

**Keywords:** Stable Diffusion, AI, Python

В данной статье показан процесс кодирования Python API для создания изображений искусственного интеллекта

Цель данной статьи создать кодировку Python API для создания изображений искусственного интеллекта.

Для создания проекта была рассмотрена статья А. М. Мартыненко и С. В. Васильев, в которой они анализируют нейронную сеть «stable diffusion» для генерации фотографий [1]. В статье А. П. Лосева, Д. А. Поленовой, Е. И. Тумановой, описывается возможность оценки качества компрессии изображений при помощи модели нейронной сети stable diffusion [2]. Была рассмотрена статья Г. А. Урванцева, К. Т. Шариповой, А. П. Маринской в которой был произведен анализ перспектив применения мультимодальных нейросетевых технологий в современном медиамаркетинге на примере stable diffusion [3].

Для начала необходимо войти в систему, чтобы использовать Google Collaboratory (Рисунок 1).

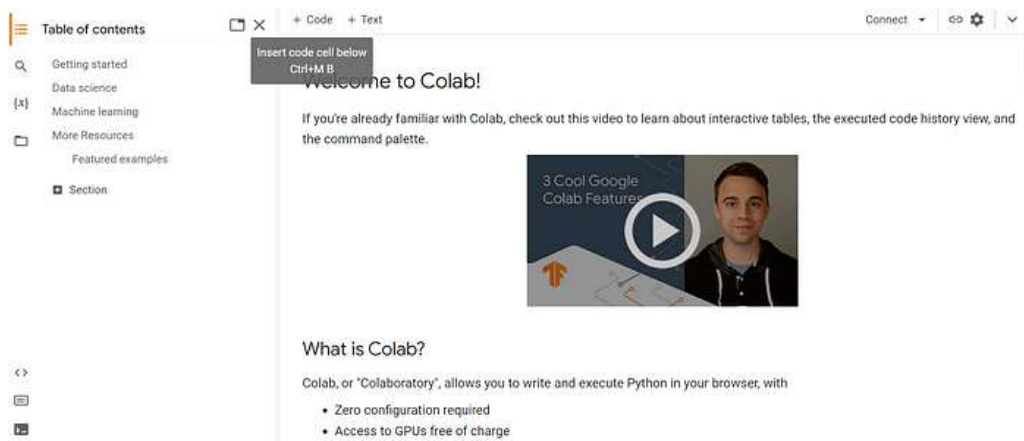


Рисунок 1. Google collaoratory

После первого создания новой записной книжки необходимо выбрать [Runtime] — [Change Runtime Type]. Когда появится диалоговое окно “Notebook Settings”, выберите “GPU” и нажать “Save”, чтобы включить GPU. Прописываем следующие команды, для начала нужно подтвердить активацию графического процессора. (Рисунок 2).

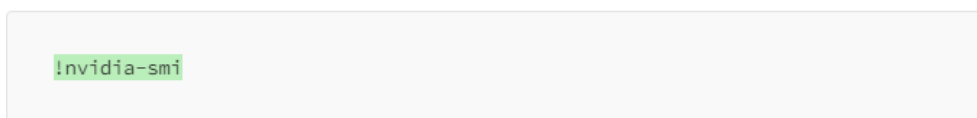


Рисунок 2. Проверка графического процессора

После должно появиться данное окно, которое подтверждает, что графический процесс был включён (Рисунок 3).

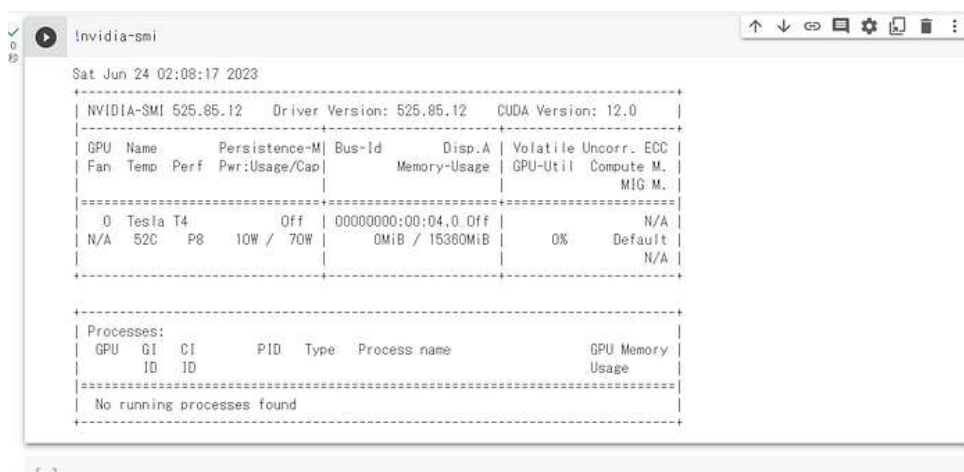


Рисунок 3. Окно включения графического процесса

Далее устанавливаем библиотеку диффузоров (diffusers) с помощью pip.

Для этого необходимо установить библиотеку под названием transformers, которая необходима для использования обученных моделей. Прописываем следующую команду для установки (Рисунок 4).

```
!pip install diffusers==0.12.1 transformers==4.19.2 ftfy accelerate
```

Рисунок 4. Установка библиотеки

При запуске процесса установки окно должно выглядеть следующим образом (Рисунок 5).

```
!pip install diffusers==0.12.1 transformers==4.19.2 ftfy accelerate
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting diffusers==0.12.1
  Downloading diffusers-0.12.1-py3-none-any.whl (604 kB)
----- 604.0/604.0 kB 41.5 MB/s eta 0:00:00
Collecting transformers==4.19.2
  Downloading transformers-4.19.2-py3-none-any.whl (4.2 MB)
----- 4.2/4.2 MB 60.0 MB/s eta 0:00:00
Collecting ftfy
  Downloading ftfy-6.1.1-py3-none-any.whl (53 kB)
----- 53.1/53.1 kB 6.6 MB/s eta 0:00:00
Collecting accelerate
  Downloading accelerate-0.20.3-py3-none-any.whl (227 kB)
----- 227.6/227.6 kB 22.5 MB/s eta 0:00:00
Collecting importlib-metadata (from diffusers==0.12.1)
  Downloading importlib_metadata-6.7.0-py3-none-any.whl (22 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from diffusers==0.12.1) (3.12.2)
Collecting huggingface-hub>=0.10.0 (from diffusers==0.12.1)
  Downloading huggingface_hub-0.15.1-py3-none-any.whl (236 kB)
----- 236.0/236.0 kB 24.1 MB/s eta 0:00:00
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from diffusers==0.12.1) (1.22.4)
Requirement already satisfied: regex<=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from diffusers==0.12.1) (2022.10.31)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from diffusers==0.12.1) (2.27.1)
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (from diffusers==0.12.1) (8.4.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers==4.19.2) (23.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers==4.19.2) (6.0)
Collecting tokenizers==0.11.3 (from transformers==4.19.2)
  Downloading tokenizers-0.11.3-cp310-cp310-manylinux2_12_x86_64.manylinux2010_x86_64.whl (6.6 MB)
----- 6.6/6.6 MB 119.7 MB/s eta 0:00:00
```

Рисунок 5.

Далее вводим следующий код на python (Код 1).

```
import torch
from diffusers import StableDiffusionPipeline

model_id = "CompVis/stable-diffusion-v1-4"

# Specify GPU "CUDA-NVIDIA"
device = "cuda"

# prompt
prompt = "Manhattan, NY in style"

# pipeline
pipe = StableDiffusionPipeline.from_pretrained(model_id,
revision="fp16", torch_dtype=torch.float16)
pipe = pipe.to(device)

# excute
generator = torch.Generator(device).manual_seed(20)
with torch.autocast("cuda"):
image = pipe(prompt, guidance_scale=7.5,
generator=generator).images[0]

# saving
image.save("successful2.png")
```

Код 1. Код программы

Поскольку нужно использовать графический процессор “CUDA-NVIDIA”, мы указываем устройство следующим образом  
device = “cuda”

В переменной с именем `prompt` введите содержимое изображения, которое вы хотите создать

`prompt = "Манхэттен".`

В данном случае хотим создать образ Манхэттена, штат Нью-Йорк, Мы хотим создать изображение Манхэттена, Нью-Йорк, поэтому указываем следующее

`prompt = "Манхэттен"`

Затем имя файла, который будет создан в `image.save("successful2.png")` (Рисунок 6-7).



```
import torch
from diffusers import StableDiffusionPipeline

model_id = "CompVis/stable-diffusion-v1-4"

# Specify GPU "CUDA-NVIDIA"
device = "cuda"

# prompt
prompt = "Manhattan, NY in style"

# pipeline
pipe = StableDiffusionPipeline.from_pretrained(model_id, revision="fp16", torch_dtype=torch.float16)
pipe = pipe.to(device)

# execute
generator = torch.Generator(device).manual_seed(20)
with torch.autocast("cuda"):
    image = pipe(prompt, guidance_scale=7.5, generator=generator).images[0]

# saving
image.save("successful2.png")
```

Рисунок 6. Настройка программы



```
# saving
image.save("successful3.png")

Downloading (...)p16/model_index.json: 100% ██████████ 543/543 [00:00<00:00, 25.3kB/s]
Fetching 16 files: 100% ██████████ 16/16 [00:14<00:00, 1.05s/d]
Downloading (...)pytorch_model.bin: 100% ██████████ 608M/608M [00:06<00:00, 144MB/s]
Downloading (...)pytorch_model.bin: 100% ██████████ 246M/246M [00:03<00:00, 67.5MB/s]
Downloading (...)processor_config.json: 100% ██████████ 342/342 [00:00<00:00, 6.76kB/s]
Downloading (...)scheduler_config.json: 100% ██████████ 307/307 [00:00<00:00, 3.82kB/s]
Downloading (...)tokenizer/merges.txt: ██████████ 525k/? [00:00<00:00, 674kB/s]
Downloading (...)_encoder/config.json: 100% ██████████ 572/572 [00:00<00:00, 5.04kB/s]
Downloading (...)_info-checkpoint.json: 100% ██████████ 209/209 [00:00<00:00, 1.48kB/s]
Downloading (...)_checker/config.json: ██████████ 4.63k/? [00:00<00:00, 48.8kB/s]
Downloading (...)tokenizer_config.json: 100% ██████████ 788/788 [00:00<00:00, 14.9kB/s]
Downloading (...)_597/UNET/config.json: 100% ██████████ 772/772 [00:00<00:00, 16.6kB/s]
Downloading (...)tokenizer/vocab.json: ██████████ 1.06M/? [00:00<00:00, 2.57MB/s]
Downloading (...)_on_pytorch_model.bin: 100% ██████████ 1.72G/1.72G [00:13<00:00, 29.8MB/s]
Downloading (...)special_tokens_map.json: 100% ██████████ 472/472 [00:00<00:00, 10.6kB/s]
Downloading (...)_on_pytorch_model.bin: 100% ██████████ 167M/167M [00:02<00:00, 89.8MB/s]
```

Рисунок 7. Настройка программы

Нажимаем на символ файла в левом углу (Рисунок 8).



```
from diffusers import StableDiffusionPipeline

model_id = "CompVis/stable-diffusion-v1-4"

# Specify GPU "CUDA-NVIDIA"
device = "cuda"

# prompt
prompt = "Manhattan, NY in style"

# pipeline
pipe = StableDiffusionPipeline.from_pretrained(model_id, revision="fp16", torch_dtype=torch.float16)
pipe = pipe.to(device)

# execute
generator = torch.Generator(device).manual_seed(20)
with torch.autocast("cuda"):
    image = pipe(prompt, guidance_scale=7.5, generator=generator).images[0]

# saving
image.save("successful2.png")

Fetching 16 files: 100% |#####| 16/16 [00:00<00:00, 434.22it/s]
100% |#####| 50/50 [00:12<00:00, 4.04it/s]
```

Рисунок 8. Настройка программы

Файл png должен быть создан следующим образом (Рисунок 9).



```
from diffusers import StableDiffusionPipeline

model_id = "CompVis/stable-diffusion-v1-4"

# Specify GPU "CUDA-NVIDIA"
device = "cuda"

# prompt
prompt = "Manhattan, NY in style"

# pipeline
pipe = StableDiffusionPipeline.from_pretrained(model_id, revision="fp16", torch_dtype=torch.float16)
pipe = pipe.to(device)

# execute
generator = torch.Generator(device).manual_seed(20)
with torch.autocast("cuda"):
    image = pipe(prompt, guidance_scale=7.5, generator=generator).images[0]

# saving
image.save("successful2.png")

Fetching 16 files: 100% |#####| 16/16 [00:00<00:00, 434.22it/s]
100% |#####| 50/50 [00:12<00:00, 4.04it/s]
```

Рисунок 9. Расположение изображения

Необходимо нажать на файл, чтобы увидеть изображение (Рисунок 10).



Рисунок 10. Вывод изображения

В данной статье показан процесс кодирования Python API для создания изображений искусственного интеллекта

## Библиографический список

1. Мартыненко А. М., Васильев С. В. Анализ нейронных сетей «stable diffusion» для генерации фотографий, по преобразованию текста в изображение / В сборнике: Донецкие чтения 2022: образование, наука,

- инновации, культура и вызовы современности. Материалы VII Международной научной конференции, посвящённой 85-летию Донецкого национального университета. Под общей редакцией С.В. Беспаловой. Донецк, 2022. С. 265-267.
2. Лосев А. П., Поленова Д. А., Туманова Е. И. Оценка качества компрессии изображений при помощи модели нейронной сети *stable diffusion* / В сборнике: Подготовка профессиональных кадров в магистратуре для цифровой экономики (ПКМ-2022). Сборник лучших докладов Всероссийской научно-технической и научно-методической конференции магистрантов и их руководителей. Сост. Н.Н. Иванов. Санкт-Петербург, 2023. С. 92-96.
  3. Урванцев Г. А., Шарипова К. Т., Маринская А. П. Анализ перспектив применения мультимодальных нейросетевых технологий в современном медиамаркетинге на примере *stable diffusion* / В сборнике: Вестник факультета социальных цифровых технологий Санкт-Петербургского государственного университета телекоммуникаций им. проф. м. а. Бонч-бруевича. сборник научно-теоретических статей. Санкт-Петербург, 2022. С. 134-139.