Modelos com visão e Geradores de Imagens

Visão

A API Groq oferece inferência rápida e baixa latência para modelos multimodais com recursos de visão para entender e interpretar dados visuais de imagens. Ao analisar o conteúdo de uma imagem, os modelos multimodais podem gerar texto legível por humanos para fornecer insights sobre dados visuais fornecidos.

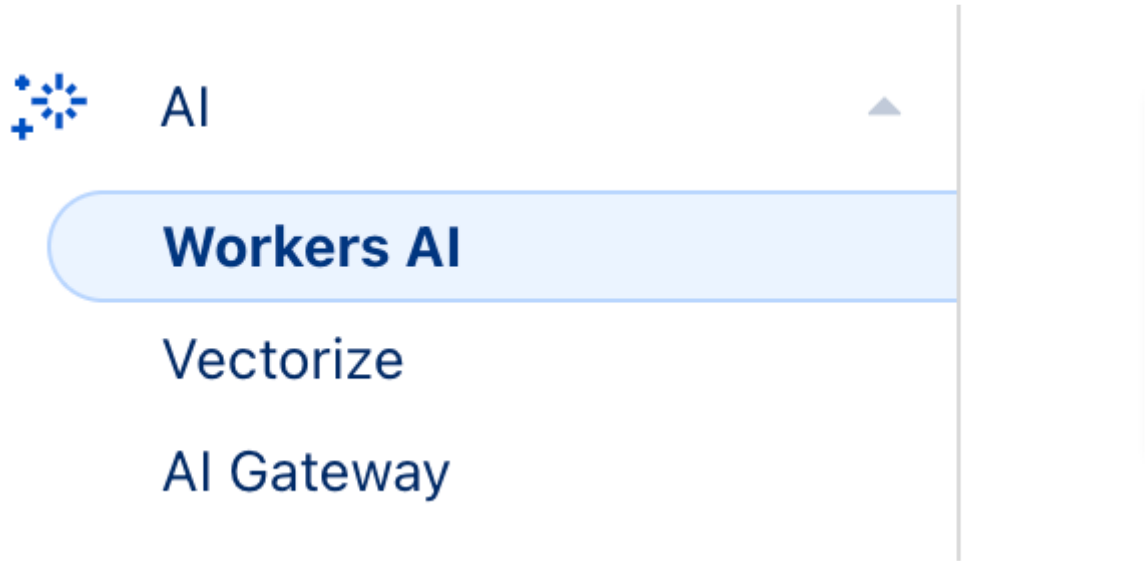
```
from groq import Groq

client = Groq()
completion = client.chat.completions.create(
    model="llama-3.2-11b-vision-preview",
    messages=[
        {
            "role": "user",
            "content": [
                {
                    "type": "text",
                    "text": "What's in this image?"
                },
                {
                    "type": "image_url",
                    "image_url": {
                        "url":
"https://upload.wikimedia.org/wikipedia/commons/f/f2/LPU-v1-die.jpg"
                    }
                }
            ]
        }
    ],
    temperature=1,
    max_completion_tokens=1024,
    top_p=1,
    stream=False,
    stop=None,
)

print(completion.choices[0].message)
```

Modelos:

`llama-3.2-90b-vision-preview`
`llama-3.2-11b-vision-preview`

Criar conta no CloudFlare

https://cloudflare.com/



Obter Token API

Para usar a API do Workers AI, crie um token personalizado com as permissões *de leitura* corretas.

Obter ID da conta

Use o ID de conta para fazer chamadas de API para a API REST do Workers AI.

Exemplo: ID da conta

```
a5f6d7af17a29dfd61e72f5a41603f4f180
```

Faça sua requisição

Substitua seu token da API do Workers AI no exemplo abaixo para fazer sua primeira chamada da API do Worker AI.

```
import requests

API_BASE_URL =
"https://api.cloudflare.com/client/v4/accounts/<ID_ACCOUNT>/ai/run/"

headers = {"Authorization": "Bearer {API_TOKEN}"}

def run(model, inputs):
    input = { "messages": inputs }
    response = requests.post(f"{API_BASE_URL}{model}", headers=headers,
json=input)
    return response.json()
```

```
inputs = [
    { "role": "system", "content": "You are a friendly assistan that helps
write stories" },
    { "role": "user", "content": "Write a short story about a llama that
goes on a journey to find an orange cloud "}
];

output = run("@cf/meta/llama-3-8b-instruct", inputs)

print(output)
```

Gerando Imagens

```
import requests
import base64

# URL do Worker AI da Cloudflare
API_URL =
"https://api.cloudflare.com/client/v4/accounts/<ID_ACCOUNT>/ai/run/@cf/bla
ck-forest-labs/flux-1-schnell"

# Substitua pela sua chave de API (se necessário)
HEADERS = {
    "Authorization": "Bearer bDhnAD4kPlz0sSbIX6Z8F9Y7ehcmZ9ef-xTtIPph",
    "Content-Type": "application/json"
}

# Prompt para gerar a imagem
DATA = {
    "prompt": "A futuristic superbike at sunset, with neon lights
reflecting on the horizon.",
    "width": 1024,
    "height": 1024,
    "num_inference_steps": 30
}

# Faz a requisição para gerar a imagem
response = requests.post(API_URL, json=DATA, headers=HEADERS)

# Verifica se a requisição foi bem-sucedida
if response.status_code == 200:
    result = response.json()

    image_base64 = result["result"]["image"]

    if image_base64:
        # Converte Base64 para imagem
        try:
```

```
            with open("output.png", "wb") as img_file:
                img_file.write(base64.b64decode(image_base64))
            print("Imagem salva como output.png")
        except Exception as e:
            print(f"Erro ao decodificar a imagem: {e}")
else:
    print(f"Erro: {response.status_code}, {response.text}")
```

Atividade prática:

Criar uma interface para a geração de imagens ou Transcrição (Karaokê) usando os modelos da CloudFlare