Runpod Worker

- *RUNPOD 는 뭐하는 사이트?* (LLM 개발 환경 구축 (GPT, Llama, Gemini, HyperClovaX) 의 내용 참조.) : GPU 기반 클라우드 컴퓨팅 서비스 제공 사이트.
  - Pods Overview : RUNPOD 가 제공하는 PODs 는 running container instances 들임. 바로 내가 RUN 해준 Container 를 Hosting 해줄 수 있다는 점이, 이 구조가 가지는 강점.
    - 궁금한 거 더 있으면, 해당 Documentation 확인하면 될 듯!
  - *클라우드 컴퓨팅?* : 원격으로 컴퓨팅 자원을 제공, 혹은 원격 컴퓨팅 자원을 이용하는 서비스(사람들이 더 쉽게 쓸 수 있도록).

Runpod Serverless GPU Cmoputing

- **Model 을 이용한 빠른 배포** 등을 강점으로서 가짐. "Runpod Serverless" 군 자체는, 사용자가 Server 에 대해 신경을 쓰지 않더라도, 설정이나 딸깍딸깍 몇 번 하면, AI Model 을 Server 로서 사용할 수 있는 Endpoint 를 제공하는 서비스임.
- 근데 왜 Serverless 라 하는지 모르겠음. auto-generated Server 라고 하는 게 낫지 않나?

Quick Deploy

- RunPod 내에 이미 존재하는 Preset 기반으로 바로 구축할 수 있도록 되어 있는 것!

RunPod Worker

- Cloud 에서 AI Model 을 다뤄주는 작업자가 따로 존재. 사용자 등은 이를 사용하거나 시키기만 하면 됨!
  - RunPod 가 기본 Infra 관리. 서버 설정 및 유지할 필요 없음.
  - Workload 등에 따라 자동으로 크기 조절.
  - RunPod SDK 가 존재해, 여러 언어에 대해 개발 및 사용 가능.
  - 업로드 RunPod 는 Endpoint 를 지원하므로 쉽게 통합 가능.
- RunPod Worker 의 주요 Flow 는 다음과 같이 정리할 수 있다.

“KakaoTalk_20240131_032514174.jpg” could not be found.

- Developer : 새로운 Worker 를 만들어 확장할 수 있다.
  1. RunPod | Endpoints | Workers (github.com) 에서 시작하면 좋다.
  2. Worker 가 할 수 있는 일을 Handler Function 으로 (그냥 바로 반환 할지, asynchronous 할지) 정의 한다.

        - 이때 Handler Function 은 일반적으로 하나만 정의한다. 또한 정의한 handler Function 은 serverless 를 start 할 때 연결된다. (Asynchronous Handler | RunPod Documentation)
    3. 들어오는 Request 의 인수에 따라 처리하는 기능들을 Develop한다.
    4. DockerFile 을 구성 (위 template 에 포함) 및, 이를 Image 로 Deploy 한다.
- User : Image 를 받아와 새로운 Endpoint 를 만들고, Worker 를 이용하는 사람
    1. RunPod 사이트에서 Docker Image 를 기반으로 새로운 Worker 를 만든다. (이때 사용 GPU 및 기타 등등에 관한 내용을 포괄한다.)
    2. 만들어 준 Worker의 Endpoint 를 잘 파악하여 사용한다.

> 전반적인 이해는 챙겼다. 1. 내 입장에선 Developer 관련 내용 살필 시간가 없으므로, 저렇게만 짚고 넘어가자. 2. Endpoint 는 조금 더 탐구 후 더 작성한다.

- 좀 놀라운 거 :
    - 각 Serverless 당 요청할 때 호출하는 Endpoint - API 는 하나만 존재한다
        - RunPod | Endpoints | Workers (github.com)
        - 이유 : 내부 구조에서, Request를 처리하는 API - MVC 에서의 Controller 에 해당하는 Handler 가 하나와만 연결될 수 있도록 만들어 놨다. 이러면, 처리 URL 이 하나일 수밖에 없다.
        - vLLM Worker 를 보면, 들어오는 parameter 마다 처리를 달리 하도록 함으로써. Input 자체는 통일한 반면, 내부 처리가 다르게 일어날 수 있도록 했다.
            - worker-vllm/src/engine.py at main · runpod-workers/worker-vllm (github.com)
    - 왜 Serverless 라고 했는지 알겠네. Server 는 일반적으로 여러 Controller 와 Service 가 있어서 들어오는 친구들을 다 처리. 이에 반해 이 친구는 목적에 따라 Worker(아마 개발 입장에서는 들어오는 건 하나라도, 여러 일이나 업무를 계속 수행하니까 Worker. 요청 창구는 동일하나, 달라지는 요청에 대해 업무를 계속 수행.)=Endpoint(사용자 입장에서는 Controller 가 하나밖에 없는 Server 이므로 Endpoint) 자체를 여러 곳으로 분산함. 즉, 분산 Controller(=Endpoint) 라 보고, 파편화 되어 있으므로, Serverless 라 한 듯!
        - 또한 제공되는 Computer 자원의 수를 Worker 라고 하는 듯!

> 여기서 Endpoint는 Serverless User의 Entrypoint (진입점) 임! - 나는 Handler Function 에 따라 여러 API 가 생겨날 것이라고 생각했는데, 생겨나는 것은 Endpoint 하나임. - 그럼 여러 개의 Handler Function 은 어디에 기여하는 것인가?

RunPod Endpoint

Overview

- 다양한 Endpoint 제공. 비동기 / 동기 작업 모두 존재.

Get Started

- Endpoint Page 에서 먼저, Requests 버튼 > Run 을 눌러, 성공적으로 Response 가 되는지부터 확인하자.
  - 이게 잘 받아져야, 현재 해당 Worker 가 잘 동작하고 있다는 뜻이다.
- 이후 cURL 을 이용해서 요청을 보내는 예시를 살펴보자.

```
curl --request POST \
--url https://api.runpod.ai/v2/${YOUR_ENDPOINT}/runsync
--header "accept: application/json" \
--header "authorization: ${YOUR_API_KEY}" \
--header "content-type: application/json" \
--data '
{
    "input": {
        "prompt": "A coffee cup.",
        "height": 512,
        "width": 512,
        "num_outputs": 1,
        "num_inference_steps": 50,
        "guidance_scale": 7.5,
        "scheduler": "KLMS"
    }
}
'
```

- header 의 경우, `${YOUR_API_KEY}` 로서 표시된 곳은 RunPod 의 해당 Endpoint 에서 생성한 API Key 를 입력하여야 한다.
  - WRW6BRW6402G3YFTPQQ6LP2622GAG8G9RETIWZOO
- data 의 경우, 웬만하면, input 이라는 큰 json object 내에 들어가도록 구성되어 있다. (worker-vllm/src/handler.py at main · runpod-workers/worker-vllm (github.com) : 내가 사용하려는 vLLM Worker 또한 동일한 것으로 보인다.)

    - input 말고 바깥에 들어갈 수 있는 parameter 는 webhook, execution policy, s3-compatible storage 가 있다. 이는 Send a request | RunPod Documentation 확인하면 된다.
- 특이한 게, json object 를 dict, json Array 를 List 라고 표현하고 있다. = runpod-workers/worker-vllm: The RunPod worker template for serving our large language model endpoints. Powered by vLLM. (github.com)

References

- References | RunPod Documentation
- Llama 2 - Build Your Own Text Generation API with Llama 2 - on RunPod, Step-by-Step (youtube.com)
- 이 두 내용 살필 것.