HuggingFace. Text Generation Inference 구축 및 사용법

> 참고 자료
>
> - Text Generation Inference (huggingface.co)
> - huggingface/text-generation-inference: Large Language Model Text Generation Inference (github.com)

이해 요약 및 정리

- 24.01.28 기준 다시 해당 이해를 정리한 내용임.
  - TGI 설치 및 사용을 위해 사용할 수 있는 가장 간단한 방법은 Docker 이용.
  - Docker 자체에 대한 이해는 Docker 란 무엇인가 참고.
- TGI 가 뭐냐? : Text Generation Inference (TGI) is a **toolkit for deploying and serving Large Language Models (LLMs)** = 즉, LLM 을 배포하고 제공하는 데 사용할 수 있는 Toolkit 이다! Inference API, Inference Endpoint 가 통합되어 있어, API Server 를 쉽게 구축하는데 용이하다!
- Docker는 위에서 정리했다시피, Container 를 이용하는 걸 도와주는 Solution이다. 부가적으로, Hub 을 통해 Image 를 내려받고 실행하여, 이미 만들어진 Container 앱을 쉽게 사용할 수 있다.
  - TGI 를 Docker 를 통해 설치하라고 하는 것은 즉, TGI 의 Git URL 기반으로 Image를 실행하여, Container 앱을 구축하는 것과 동일하다고 생각하면 될 듯!

> 아래는 본문 자세한 내용임.

Quict Start

- Text Generation Inference (TGI) is a **toolkit for deploying and serving Large Language Models (LLMs)**.
- Open-source text generation LLMs 인 Llama, Falcon, StarCoder, BLOOM, GPT-NeoX, and T5 을 높은 Performance 로 구동할 수 있도록 해줌.
- 이외 기타 특징의 경우, "Simple launcher to serve most popular LLMs" 외 기타 등등 많은데, 다 좀 전문적인 지식들이라 열거하지는 않음.

Quick Tour

1. Docker Get Docker | Docker Docs 이거 통해서 처음 설치하고.
2. Model 및 명령어 입력하여, Container 배포하면 끝임.
    - Docker 에 아마 Container 화 할 때 뭐 어떻게 해줄지에 관한 게 TGI 기반으로 해서 이미 존재해서, 이를 사용해주기만 하면 되는 것으로 보임.

```shell
model=tiiuae/falcon-7b-instruct
volume=$PWD/data # share a volume with the Docker container to avoid downloading weights every run

docker run --gpus all --shm-size 1g -p 8080:80 -v $volume:/data ghcr.io/huggingface/text-generation-inference:1.4 --model-id $model
```

- NDIVIA Graphic Card 기반 환경일 경우, NVIDIA Container Toolkit 설치와 Quick Tour (huggingface.co) 를 참고하여야 함.
- AMD 도 지원하는데, 이는 Supported Hardware section, AMD documentation 이 둘 참고하여야 함.

3. 이후 URL 기반으로 API Request 해서 접근 가능.

> 생각한 것보다 너무 쉬움!

Installation

- 여기에는 CLI Tool 을 Install 하는, 즉 TGI 를 Source 로 부터 Install 하는 과정을 서술함. Docker 를 이용하는 것이 훨씬 간편하므로, Docker 이용을 굉장히 권고하고 있음.
    - *Q. CLI Tool 이 뭐냐? => A. "Command Line Interface Tool"의 약자로, 텍스트 기반의 사용자 인터페이스를 통해 컴퓨터와 상호작용하는 프로그램을 의미. 이 Section은 CLI 로 제공되는 TGI Tool 이 있고, 이걸 활용하는 방법을 서술함.*
- 다운이나 사용법도 되게 간단하더라! 근데 그렇게 필요하지는 않을 것 같아서 넘어감!

Supported Models and Hardware

- 지원 모델 : BLOOM / FLAN-T5 / Galactica / GPT-Neox / Llama / OPT / SantaCoder / Starcoder / Falcon 7B / Falcon 40B / MPT / Llama V2 / Code Llama / Mistral / Mixtral / Phi 가 존재!
- 위 모델 내에 존재하지 않는다 하더라도, performance 가 보장되지 않을 뿐, Model 자체를 사용할 수는 있음!
- 지원 하드웨어 : NVIDIA A100, A10G and T4 GPUs with CUDA 12.2+ 가 권장.
    - NVIDIA Container Toolkit을 꼭 다운받아서 사용하는 것이 좋음.
- 다른 NVIDIA GPUs 의 경우, continuous batching 이 여전이 적용되긴 하나, flash attention, paged attention 과 같은 일부 operation 들은 실행되지 않음.
- AMD GPU 중 ROCm-enabled AMD Instinct MI210 and MI250 GPUs, with paged attention, GPTQ quantization, flash attention v2 support 인 것도 지원하긴 함!
    - 단, Loading AWQ checkpoints / Flash layer norm kernel /Kernel for slinding window attention (Mistral) 의 경우, 해당 GPU 에서 지원하지 않음.
- TGI 는 또한 AI hardware accelerators 들인 Habana first-gen Gaudi and Gaudi2, AWS Inferentia2 또한 지원함.

Message API

- Messages API 는 OpenAI Chat Completion API와 호환된다!
- 이는 즉, OpenAI's client libraries 를 이용해 TGI 의 Message 를 보내는 것 또한 가능하다는 것을 의미한다!
- Client 에서 해당 API 를 사용하는 것은 Chat Completion 과 아예 동일하게 사용 가능.
  - **와 이거 확실히 되게 잘 만들어 둔 듯!**
- 메시지 실시간으로 받기 / 비동기로 받기
  - Streaming
  - Synchronous
- Cloud Providers
  - TGI 는 확장 및 강력한 텍스트 생성을 위해 다양한 클라우드 제공업체에 배포 가능.
    - (맹 API Server 를 Local 에서만 돌리던 걸)
  - Amazon SageMaker 는 이 Provider 중 하나로, 좋은 기능을 많이 제공.
- `MESSAGES_API_ENABLED=true` 로 해야된다고 하긴 하는데, 아직 딱히 체감 안 됨!

Tutorials

> Quick Tour 안에서, Docker 를 통해 TGI 를 환경을 구축하는 방법을. / Client 에서 해당 API 를 호출할 때 사용하는 방법을. (OpenAI 의 API 와 동일한 Endpoint 및 구성을 갖기 때문에, 동일한 Library 를 써도 호출됨) 익혔기 때문에 사실 더 할 필요가 없긴 함. 구축, 사용 이미 둘 다 완수되었으므로.
> 하지만, 대략적으로나마 남아있는 내용에 대해 좀 더 파악해보자! (되게 잘 되어 있어서 한번 파악해 보고는 싶네!)

> 하다 보니까, 관련성 없거나, 이런 내용들은 그닥 끌리지가 않아서. 넘어감! (24.01.27)

Consuming TGI

- POST Request 로 호출해야 하는 /generate 와 /generate_stream URL 설명.
- `huggingface-hub` python library 를 통해 Inference Endpoint 에 쉽게 접근 가능.

ChatUI

- ChatUI 는 LLM 제공을 위해 만들어진 open-source interface.
- SERP API 등을 활용하기도 하고, TGI Server 를 자동 소비하거나, 다른 TGI Endpoint 간 전환을 시도할 수도 있음.
  - Hugging Chat 에서 시도 가능.
  - 나만의 Hugging Chat 을 만들기 위해 ChatUI Docker Space 를 기반으로 배포할 수도 있음.
- ChatUI 와 TGI 를 같은 환경에서 제공하기 위해서는 `chat-ui` repository > `.env.local` file > `MODELS` variable 에 나의 Endpoint 를 추가하면 됨. ()

```
{
// rest of the model config here
"endpoints": [{"url": "https://HOST:PORT/generate_stream"}]
}
```

Gradio

- Gradio 는 너의 machine learning model 을 단순한 몇개의 코드로 web application 을 빌드할 수 있도록 해주는 친구임.
- 이 사용 방법에 대해 명시해놨는데, 굳이 정리하지 않아도 상관없을 듯.

API Documentation

- OpenAPI documentation of the `text-generation-inference` REST API 에서 나와 있는 API 정보를 기반으로 작성하면 된다. 실제로, TGI 의 API 의 경우, 해당 API 와 동일하다.
- Swagger UI 도 제공한다 : here

Preparing Model for Serving

- TGI 를 이용함으로써 모델이 얻을 수 있는 이득을 몇 가지 소개.

1. Quantization

- 모델을 경량화 하는 방법 중 하나를 Quantization 으로 알고 있음. (매개변수의 수를 줄이고, 전체 처리 시간을 줄일 수 있음)
- 일부 모델은 bits-and-bytes, GPT-Q, AWQ quantization을 이용하는데, 이들이 적용된 TGI 를 사용할 수 있음.

2. RoPE Scaling

- RoPE scaling 을 CLI 등에서 어떻게 하는지에 대해서만 서술되어 있고, 이것이 무엇을 의미하는지에 대해해서는 나와 있지 않다.

3. SateTensors

- tensor parallelism 을 위해 필요한, 빠르고 안전하게 지속하가능한 형식(format)이다.
- TGI 는 safetensors 모델을 지원한다.

Serving Private & Granted Models

- Model 이 Private 로 설정되어 있거나, 인가가 되어야 하는 경우, Hugging Face Hub tokens page 에서 Access Token 을 발급 받아 아래와 같이 Docker 에서 사용해주면 된다!

SHELL

```
model=meta-llama/Llama-2-7b-chat-hf
volume=$PWD/data
token=<your READ token>

docker run --gpus all \
    --shm-size 1g \
    -e HUGGING_FACE_HUB_TOKEN=$token \
    -p 8080:80 \
    -v $volume:/data ghcr.io/huggingface/text-generation-
inference:1.4 \
    --model-id $model
```

> Docker 도 결국 Linux 명령어 기반 컨테이너화 및 배포 등을 도와주는 친구니까, Shell Script 로 표현하는 게 낫겠지?

- CLI 를 따로 이용하는 경우, Serving Private & Gated Models (huggingface.co) : 해당 링크에서 직접 확인할 수 있도록 할 것.

Using TGI CLI

- 생략

All TGI CLI options

- 생략

Non-core Model Serving

- TGI 는 많은 Architecture 를 지원하지만, 지원하지 않는 모델의 경우, 해당 모델의 `transformers` implementation 을 사용하도록 대체된다. 이에 따라, TGI 가 가지고 있는 강력한 기능들인 tensor-parallel sharding, flash attention 등은 사용할 수 없지만, 여전히 많은 이점을 가지고 있다.
- support 가 아닌 model 또한 다음과 같은 command 를 이용하여 docker 로 container 화 할 수 있다~

SHELL

```
# Full Supported
docker run --gpus all --shm-size 1g -p 8080:80 -v $volume:/data
ghcr.io/huggingface/text-generation-inference:latest --model-id
gpt2

# Local
docker run --gpus all --shm-size 1g -p 8080:80 -v $volume:/data
ghcr.io/huggingface/text-generation-inference:latest --model-id
<CUSTOM_MODEL_ID> --trust-remote-code

# Hub 에서 Available
docker run --gpus all --shm-size 1g -p 8080:80 -v $volume:/data
ghcr.io/huggingface/text-generation-inference:latest --model-id
<CUSTOM_MODEL_ID> --trust-remote-code
```

Conceptual Guides

> 해당 내용은 위에서 언급한 개념들에 대한 상세한 설명이 담겨있음. 아예 모르거나 / 대략적으로 알거나 나뉨. 이건 지금 살펴봐도 따로 효과가 크지 않을 것 같아서. 이후에 원할 때 추가적으로 살필 수 있도록 할 것.

Streaming

- Streaming (huggingface.co)

Quantiztion

- Quantization (huggingface.co)

Tensor Parallelism

- Tensor Parallelism (huggingface.co)

PagedAttention

- Safetensors (huggingface.co)

Flash Attention

- Flash Attention (huggingface.co)