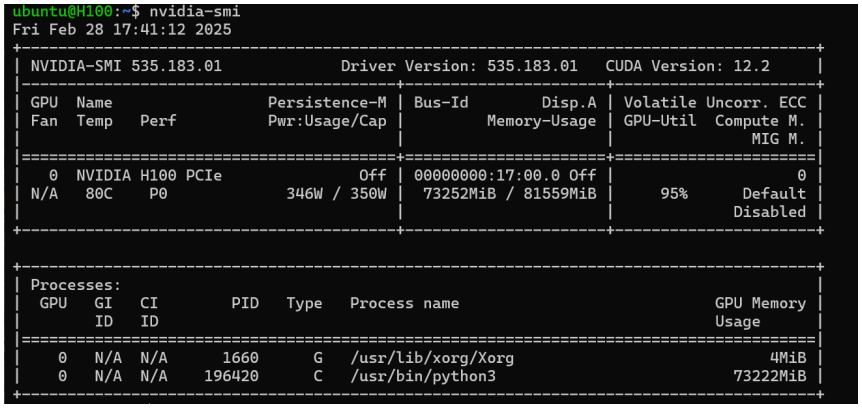
MT [Google/Gemma-2-9b-it (DOCKER)]

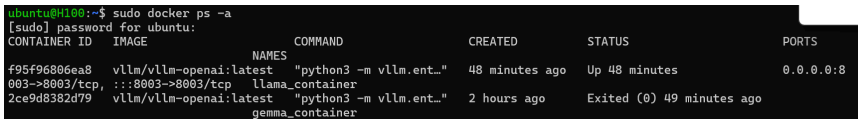
실행 환경

: GPU H100



: cuda: 12.2

:torch:2.5.1



Docker 컨테이너 실행 (`docker run`)

```
# Deploy with docker on Linux:
sudo docker run --runtime nvidia --gpus all --name gemma_container -v ~/.cache/huggingface:/root/.cache/huggingface --env "H
```

FastAPI 서버 실행

Gemma 모델을 사용하는 경우, `/generate` 엔드포인트가 없는 것은 특정 모델 서버나 API 설정에 따라 다를 수 있어.

일반적으로 Gemma 모델을 FastAPI, vLLM, Text Generation Inference (TGI) 등의 서버로 배포하는 경우, 엔드포인트가 다르게 설정될 수도 있어.

Gemma 모델의 API 엔드포인트 확인

1. **FastAPI 기반 서버**

   보통 `/v1/chat/completions`을 사용한다면, OpenAI 호환 API 서버를 띄운 것 같은데, FastAPI 기반이면 `/generate`가 없을 수도 있어.

2. **TGI (Text Generation Inference)**

   Hugging Face의 `text-generation-inference`를 사용했다면 보통 `/generate` 엔드포인트가 있음.

3. **vLLM 서버**

   vLLM으로 띄웠다면 `/generate`가 아니라 `/v1/completions` 또는 `/v1/chat/completions`을 사용해야 함.

```
# 도커 컨테이너 실행
sudo docker start gemma_container

# 도커 컨테이너 종료
sudo docker stop gemma_container
```

```
# 도커 컨테이너 상태 확인
sudo docker ps
```

```
python3 WORK_TEST08.py
```

```
import pymysql
import requests
import re

# LLM FastAPI 서버 URL
LLM_SERVER_URL = "http://localhost:8002/v1/chat/completions"

SIMPLEST = """
적합한 프롬프트
"""


# Corrected function (Gemma 모델 특성)
def call_llm_server(prompt):
    try:
        # OpenAI-style request payload
        data = {
            "model": "google/gemma-2-9b-it",
            "messages": [
                {
                    "role": "user",
                    "content": prompt
                }
            ]
        }

        response = requests.post(LLM_SERVER_URL, json=data)

        if response.status_code == 200:
            result_text = response.json().get("choices", [{}])[0].get("message", {}).get("content", "").strip()
            print(result_text)
            return clean_llm_response(result_text)
        else:
            print(f"LLM 서버 오류: {response.status_code}, {response.text}")
            return "0"
    except Exception as e:
        print(f"LLM 서버 요청 실패: {e}")
        return "0"
```