

unified-sdk - TensorRT 셋업 검토 건

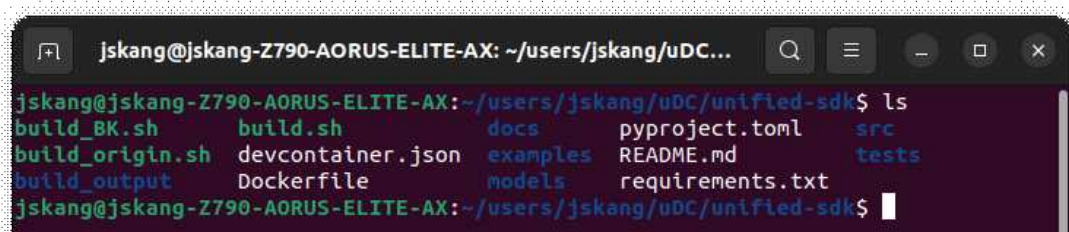
작성일 : 251121

작성자 : 강주성

0. 목적

- 본 문서의 목적은 uDC 과제 개발 수행중인 unified-sdk 개발 중, TensorRT 에 대한 셋업 현황을 체크하기 위한 목적이다.

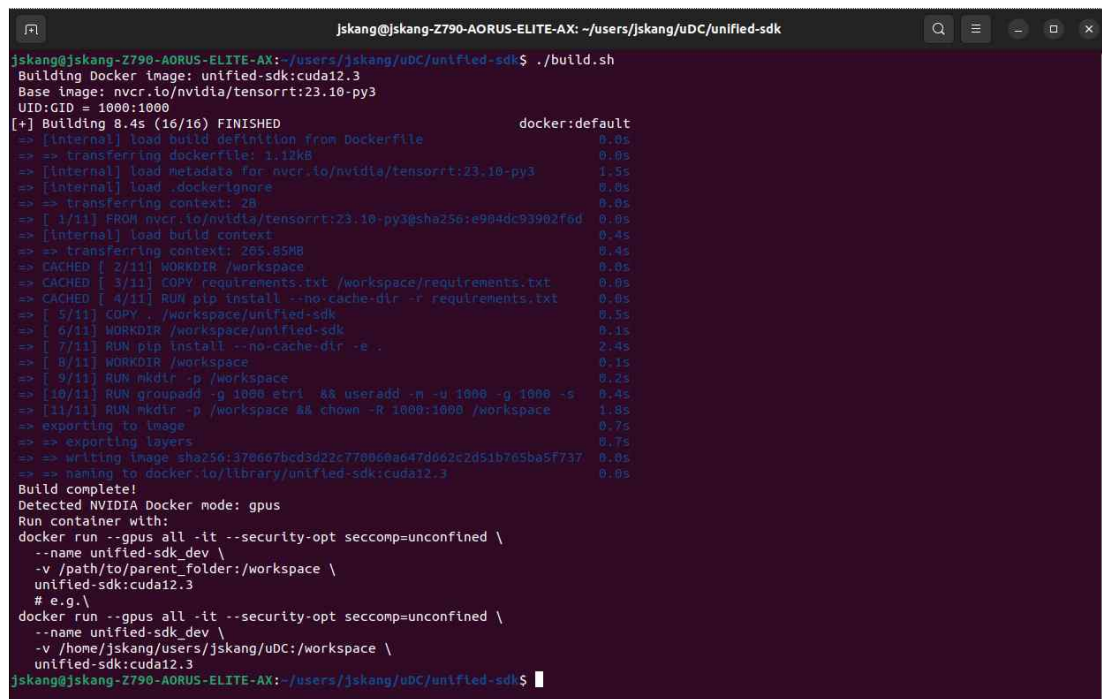
1. unified-sdk 폴더 구조



```
jskang@jskang-Z790-AORUS-ELITE-AX: ~/users/jskang/uDC...  
jskang@jskang-Z790-AORUS-ELITE-AX:~/users/jskang/uDC/unified-sdk$ ls  
build_BK.sh      build.sh          docs              pyproject.toml    src  
build_origin.sh  devcontainer.json examples          README.md         tests  
build_output     Dockerfile        models            requirements.txt
```

2. 초기설정

1) ./build.sh



```
jskang@jskang-Z790-AORUS-ELITE-AX: ~/users/jskang/uDC/unified-sdk  
jskang@jskang-Z790-AORUS-ELITE-AX:~/users/jskang/uDC/unified-sdk$ ./build.sh  
Building Docker image: unified-sdk:cuda12.3  
Base image: nvcr.io/nvidia/tensorrt:23.10-py3  
UID:GID = 1000:1000  
[+] Building 8.4s (16/16) FINISHED                                docker:default  
=> [internal] load build definition from Dockerfile               0.0s  
=> => transferring dockerfile: 1.12kB                             0.0s  
=> [internal] load metadata for nvcr.io/nvidia/tensorrt:23.10-py3 1.5s  
=> [internal] load .dockerignore                                  0.0s  
=> => transferring context: 2B                                       0.0s  
=> [ 1/11] FROM nvcr.io/nvidia/tensorrt:23.10-py3@sha256:a904dc93902f6d 0.0s  
=> [internal] load build context                                  0.4s  
=> => transferring context: 265.85MB                                0.4s  
=> CACHED [ 2/11] WORKDIR /workspace                             0.0s  
=> CACHED [ 3/11] COPY requirements.txt /workspace/requirements.txt 0.0s  
=> CACHED [ 4/11] RUN pip install --no-cache-dir -r requirements.txt 0.0s  
=> [ 5/11] COPY . /workspace/unified-sdk                         0.5s  
=> [ 6/11] WORKDIR /workspace/unified-sdk                        0.1s  
=> [ 7/11] RUN pip install --no-cache-dir -e .                   2.4s  
=> [ 8/11] WORKDIR /workspace                                    0.1s  
=> [ 9/11] RUN mkdir -p /workspace                               0.2s  
=> [10/11] RUN groupadd -g 1000 etri && useradd -m -u 1000 -g 1000 -s 0.4s  
=> [11/11] RUN mkdir -p /workspace && chown -R 1000:1000 /workspace 1.8s  
=> exporting to image                                             0.7s  
=> => exporting layers                                              0.7s  
=> => writing image sha256:370667bcd3d22c770060a047d062c2d51b765ba5f737 0.0s  
=> => naming to docker.io/library/unified-sdk:cuda12.3          0.0s  
Build complete!  
Detected NVIDIA Docker mode: gpus  
Run container with:  
docker run --gpus all -it --security-opt seccomp=unconfined \  
  --name unified-sdk_dev \  
  -v /path/to/parent_folder:/workspace \  
  unified-sdk:cuda12.3  
# e.g.\  
docker run --gpus all -it --security-opt seccomp=unconfined \  
  --name unified-sdk_dev \  
  -v /home/jskang/users/jskang/uDC/workspace \  
  unified-sdk:cuda12.3  
jskang@jskang-Z790-AORUS-ELITE-AX:~/users/jskang/uDC/unified-sdk$
```

2) docker run -gpus / --runtime 실행

```
root@3eabd23e2edd: /workspace
jskang@jskang-Z790-AORUS-ELITE-AX: ~/users/jskang/UDC/unified-sdk$ docker run --gpus all -it --security-opt seccomp=unconfined --name unified-sdk_dev -v /home/jskang/users/jskang/UDC/workspace unified-sdk:cuda12.3

=====
== NVIDIA TensorRT ==
=====

NVIDIA Release 23.10 (build 70756733)
NVIDIA TensorRT Version 8.6.1
Copyright (c) 2016-2023, NVIDIA CORPORATION & AFFILIATES. All rights reserved.

Container image Copyright (c) 2023, NVIDIA CORPORATION & AFFILIATES. All rights reserved.

https://developer.nvidia.com/tensorrt

Various files include modifications (c) NVIDIA CORPORATION & AFFILIATES. All rights reserved.

This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license

To install Python sample dependencies, run /opt/tensorrt/python/python_setup.sh

To install the open-source samples corresponding to this TensorRT release version
run /opt/tensorrt/install_opensource.sh. To build the open source parsers,
plugins, and samples for current top-of-tree on master or a different branch,
run /opt/tensorrt/install_opensource.sh -b <branch>
See https://github.com/NVIDIA/TensorRT for more information.

root@3eabd23e2edd: /workspace#
```

3) 도커 내 GPU 잡히는 지 확인 - nvidia-smi

```
root@3eabd23e2edd: /workspace# nvidia-smi
Mon Nov 24 08:13:54 2025

+-----+
| NVIDIA-SMI 570.195.03              | Driver Version: 570.195.03   | CUDA Version: 12.8 |
+-----+-----+
| GPU Name                               | Persistence-M | Bus-Id  | Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap       |      |      |      |      |
+-----+-----+
| 0  NVIDIA GeForce RTX 4090            | Off       | 00000000:01:00.0  | On     | Default              |
| 0%   32C   P8         26W / 450W      |      | 587MiB / 24564MiB |      | 0%   MIG M.          |
+-----+-----+
| Processes:                               |
| GPU   GI   CI        PID   Type   Process name                      | GPU Memory |
| ID    ID   ID                     |            | Usage                     |
+-----+-----+
root@3eabd23e2edd: /workspace#
```

% 도커 및 GPU작동 확인.

2. TensorRT 예제 코드 실행

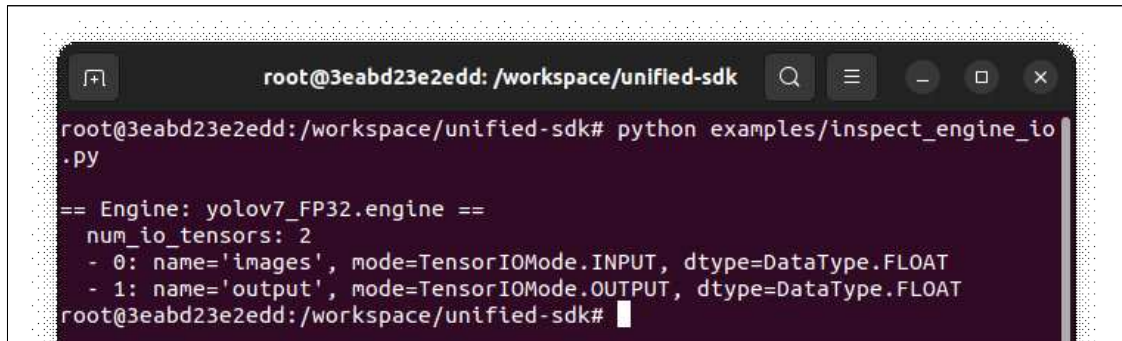
1) run_tensorrt_build.py

- Yolov7 기준, Onnx to TensorRT Compile 변환 과정

```
root@3eabd23e2edd: /workspace/unified-sdk
[11/24/2025-08:19:36] [TRT] [W] onnx2trt_utils.cpp:374: Your ONNX model has
been generated with INT64 weights, while TensorRT does not natively support
INT64. Attempting to cast down to INT32.
[11/24/2025-08:19:36] [TRT] [I] Successfully created engine.
[11/24/2025-08:19:36] [TRT] [I] build_output/yolov7_FP32.engine
root@3eabd23e2edd: /workspace/unified-sdk#
```

2) inspect_engine_io.py

- 생성된 *.engine 파일에 대해, Layer Info.를 확인하는 과정 (Yolov7 기준)

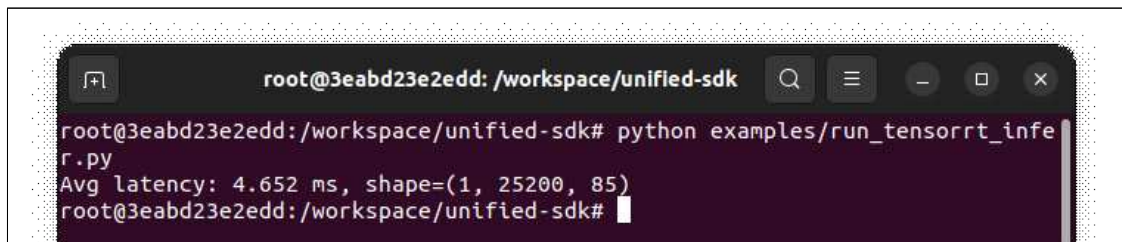
A terminal window with a dark background and light text. The title bar shows 'root@3eabd23e2edd: /workspace/unified-sdk'. The command 'python examples/inspect_engine_io.py' has been executed. The output shows the engine name 'yolov7_FP32.engine' and the number of I/O tensors (2). It then lists the details for each tensor: 'images' (INPUT, FLOAT) and 'output' (OUTPUT, FLOAT).

```
root@3eabd23e2edd: /workspace/unified-sdk# python examples/inspect_engine_io.py

== Engine: yolov7_FP32.engine ==
num_io_tensors: 2
- 0: name='images', mode=TensorIOMode.INPUT, dtype=DataType.FLOAT
- 1: name='output', mode=TensorIOMode.OUTPUT, dtype=DataType.FLOAT
root@3eabd23e2edd: /workspace/unified-sdk#
```

3) run_tensorrt_infer.py

- 생성된 *.engine 파일 기준, model inference time 확인하는 코드

A terminal window with a dark background and light text. The title bar shows 'root@3eabd23e2edd: /workspace/unified-sdk'. The command 'python examples/run_tensorrt_infer.py' has been executed. The output shows the average latency as 4.652 ms for a shape of (1, 25200, 85).

```
root@3eabd23e2edd: /workspace/unified-sdk# python examples/run_tensorrt_infer.py
Avg latency: 4.652 ms, shape=(1, 25200, 85)
root@3eabd23e2edd: /workspace/unified-sdk#
```

% TensorRT 샘플 코드 동작 확인.

100. Issues Reports

- CONFIDENTIAL