도커 컨테이너를 이용한 GPU 프로그램 실행

(Running a GPU Program with Docker Container)

※ GPU 자원을 사용하기 위한 별도의 도커 엔진(nvidia-docker)이 존재한다

2023. 03. 28.

Ref. https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/user-guide.html
https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html
https://github.com/NVIDIA/nvidia-docker
https://www.sysnet.pe.kr/2/0/12816 GPU 사용 환경 설정

내용

- GPU 환경 고려한 이미지 생성 : Dockerfile 활용

- nvidia-docker 실행 환경 구축 및 실행 확인
  - GPU 도커 엔진 설치
  - GPU 자원을 사용 가능하게 도커 컨테이너 실행
  - GPU 환경 확인
  - CUDA 이용 프로그램 동작 확인

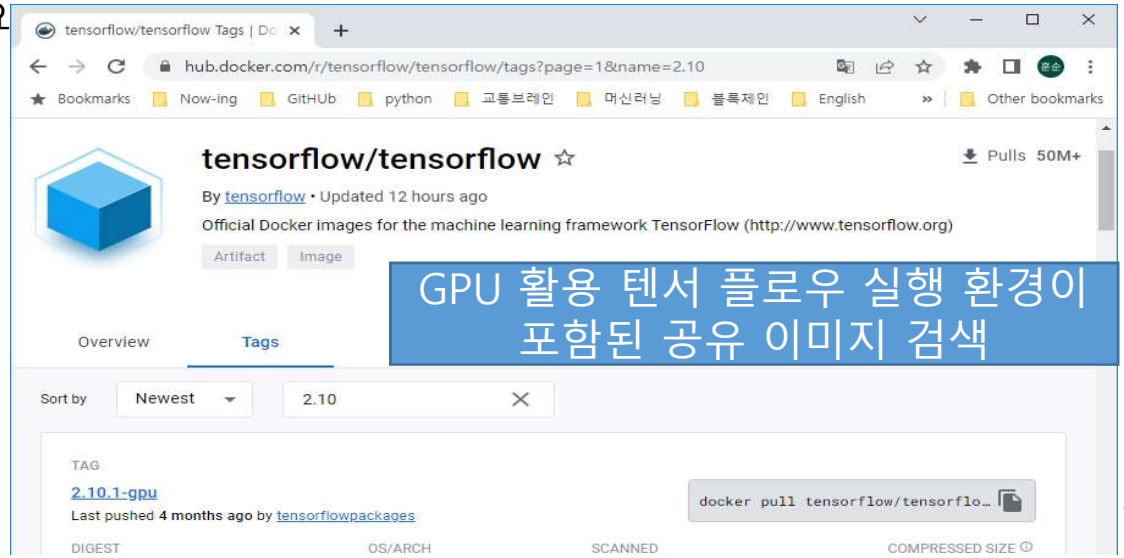
- UNIQ에 적용 / 동작 확인

GPU 환경 고려한 이미지 생성 : Dockerfile 활용

GPU 실행 고려 이미지 자체 제작 VS **GPU 고려된 공유 이미지 활용하여 확장** 선택

- CUDA 실행 환경을 위한 패키지들이 포함되도록 스크립트 작성
  - 기본 OS 이미지에서 시작 : 예, Ubuntu:20.04
  - GPU 환경에서 CUDA 실행을 위한 패키지 추가
    - 예, tensorflow 2.10.0, CUDA_VERSION 11.2.1, CUDNN 8.1.0.77-1, Python 3.8.10
    - Ref. https://www.sysnet.pe.kr/2/0/12816
  - 신호 최적화 실행에 필요한 패키지 추가
- (+) 이미지 (크기) 최적화 가능
- (-) 패키지들 간 버전/호환 고려해야
  - 많은 시행착오(시간, 노력) 필요

- 도커 허브 상의 공유된 이미지 활용하여 확장
  - GPU 환경 고려된 이미지 중 하나 선정 https://hub.docker.com/r/tensorflow/tensorflow/tags
  - 신호 최적화 실행에 필요한 패키지 추가
- (+) 이미지에 포함되어야 하는 패키지들 간 버전 충돌(호환성)로 인한 문제 발생 가능성 감소
- (-) 이미지 크기 증가



GPU 도커 엔진 설치

ubuntu 기준으로 설명
CentOS는 부록 참고

docker 설치

업데이트 및 패키지 설치

```
$ sudo apt update
$ sudo apt-get install -y ca-certificates  curl  software-properties-common  apt-transport-https  gnupg  lsb-release
```

GPG 키와 저장소 추가

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
```

도커 엔진 설치 및 버전 확인

```
$ sudo apt update
$ sudo apt install docker-ce docker-ce-cli containerd.io
$ docker --version
```

nvidia-docker 설치

GPG 키와 저장소 추가

```
$ distribution=$(. /etc/os-release;echo $ID$VERSION_ID)
   && curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
   && curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
```

nvidia-docker 설치

```
$ sudo apt-get update
$ sudo apt-get install -y nvidia-docker2
```


Ref. https://data-science-hi.tistory.com/203

GPU 자원을 사용 가능하게 도커 컨테이너 실행

방법 1) 컨테이너 실행 옵션 활용

--gpus 옵션

```
$ sudo docker run --rm --gpus all ubuntu:20.04 nvidia-smi
$ sudo docker run -it --gpus all tensorflow/tensorflow:2.10.0-gpu  /bin/bash

## 디바이스 지정
$ sudo docker run --rm --gpus "device=1" ubuntu:20.04 nvidia-smi
$ sudo docker run -it --gpus "device=1" tensorflow/tensorflow:2.10.0-gpu  /bin/bash

$ sudo docker run -it --gpus 1 tensorflow/tensorflow:2.10.0-gpu  /bin/bash
```

--runtime 과 -e NVIDIA_VISIBLE_DEVICE 옵션 이용

```
$ sudo docker run --rm --runtime=nvidia -e NVIDIA_VISIBLE_DEVICES=all ubuntu:20.04 nvidia-smi

## 디바이스 지정
$ sudo docker run --rm --runtime=nvidia -e NVIDIA_VISIBLE_DEVICES=0 ubuntu:20.04 nvidia-smi
```

방법 2) 기본으로 nvidia runtime을 사용하게 설정

```
$ sudo systemctl stop docker
$ sudo systemctl stop docker.socket

$ sudo vi /etc/docker/daemon.json
   "default-runtime": "nvidia"     # 추가; 앞선 설정이 있다면 콤마를 붙이고 넣어 준다

$ sudo systemctl start docker
```


Ref. https://dongle94.github.io/docker/docker-nvidia-docker-install/ https://kyumdoctor.co.kr/18

GPU 환경 확인

Host Machine

```
(p3.8) tsoexp@hunsooni-dev:~$ nvidia-smi
Wed Mar 15 13:20:13 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 26%   37C    P8    21W / 250W |    183MiB / 11016MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  Off  | 00000000:4D:00.0 Off |                  N/A |
| 26%   36C    P8     1W / 250W |      5MiB / 11019MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                167MiB |
|    0   N/A  N/A      1954      G   /usr/bin/gnome-shell               14MiB |
|    1   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                 4MiB |
+-----------------------------------------------------------------------------+
(p3.8) tsoexp@hunsooni-dev:~$ nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Sun_Jul_28_19:07:16_PDT_2019
Cuda compilation tools, release 10.1, V10.1.243
(p3.8) tsoexp@hunsooni-dev:~$
```

디스플레이(그래픽) 위한 프로세스 존재

docker

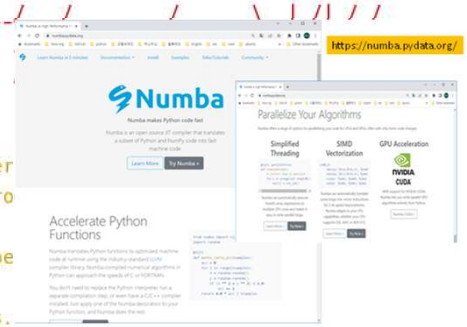
```
(p3.8) tsoexp@hunsooni-dev:~$ sudo docker run -it --gpus all \
>    -v /home/tsoexp/z.docker_test/io:/uniq/optimizer/io optimizer:v2.10.0-gpu \
>    nvidia-smi
Wed Mar 15 12:16:46 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 26%   38C    P8    21W / 250W |    183MiB / 11016MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  Off  | 00000000:4D:00.0 Off |                  N/A |
| 27%   38C    P8     2W / 250W |      5MiB / 11019MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
(p3.8) tsoexp@hunsooni-dev:~$
(p3.8) tsoexp@hunsooni-dev:~$ sudo docker run -it --gpus all \
>     -v /home/tsoexp/z.docker_test/io:/uniq/optimizer/io optimizer:v2.10.0-gpu \
> nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2021 NVIDIA Corporation
Built on Sun_Feb_14_21:12:58_PST_2021
Cuda compilation tools, release 11.2, V11.2.152
Build cuda_11.2.r11.2/compiler.29618528_0
(p3.8) tsoexp@hunsooni-dev:~$
```

격리되었기에 프로세스가 보이지 않음

Skip

CUDA 이용 프로그램 동작 확인

```
(p3.8) tsoexp@hunsooni-dev:~$ sudo docker run --gpus all -v /tmp:/io \
>  -it tensorflow/tensorflow:2.10.0-gpu /bin/bash

________                               _______________
___  __/__________________________________ ____/__  /________      __
__  /  _  _ \_  __ \_  ___/  __ \_  ___/_  /_  _  / / /  _ \ | /| / /
_  /   /  __/  / / /(__  )/ /_/ /  /   _  / /  __/ / /  __/ |/ |/ /
/_/    \___//_/ /_//____/ \____//_/    /_/  \___/  \___/|__/|__/

WARNING: You are running this container
mounted volumes to be created as the ro

To avoid this, run the container by spe

$ docker run -u $(id -u):$(id -g) args.

root@3843adc38bf1:/# pip install numba
Collecting numba
  Downloading numba-0.56.4-cp38-cp38-manylinux2014_x86_64.manylinux_2_17_x86_64.v
     |████████████████████████████████| 3.5 MB 10.2 MB/s
Requirement already satisfied: importlib-metadata; python_version < "3.9" in /us
ackages (from numba) (4.12.0)
Requirement already satisfied: numpy<1.24,>=1.18 in /usr/local/lib/python3.8/dist
3.2)
Collecting llvmlite<0.40,>=0.39.0dev0
  Downloading llvmlite-0.39.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_6
     |████████████████████████████████| 34.6 MB 2.7 MB/s
Requirement already satisfied: setuptools in /usr/local/lib/python3.8/dist-packag
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.8/dist-package
 python_version < "3.9"->numba) (3.8.1)
Installing collected packages: llvmlite, numba
Successfully installed llvmlite-0.39.1 numba-0.56.4
```



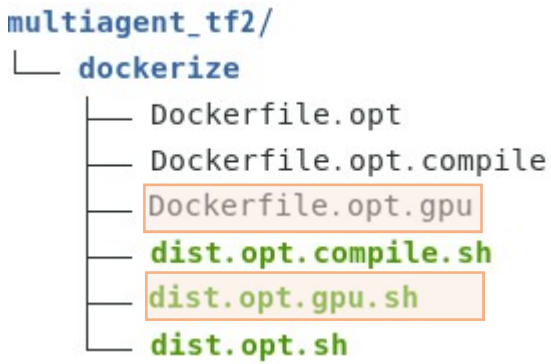

```
root@3843adc38bf1:/# cat -n /io/sample2.py
     1  import numpy as np
     2  from numba import cuda
     3
     4  # Set the number of elements in the arrays
     5  n = 10000
     6
     7  # Define the kernel function to add two arrays on the GPU
     8  @cuda.jit
     9  def add_kernel(x, y, out):
    10      idx = cuda.grid(1)
    11      if idx < n:
    12          out[idx] = x[idx] + y[idx]
    13
    14  # Generate two arrays to add
    15  x = np.arange(n)
    16  y = np.ones(n)
    17  print(f"x={x}\ny={y}")
    18
    19  # Allocate memory on the GPU for the arrays
    20  d_x = cuda.to_device(x)
    21  d_y = cuda.to_device(y)
    22  d_out = cuda.device_array(n)
    23
    24  # Define the block size and grid size for the GPU computation
    25  block_size = 32
    26  grid_size = (n + block_size - 1) // block_size
    27
    28  # Launch the kernel on the GPU
    29  add_kernel[grid_size, block_size](d_x, d_y, d_out)
    30
    31  # Copy the result back from the GPU to the CPU
    32  out = d_out.copy_to_host()
    33
    34  # Print the result
    35  print(f"x+y={out}")
root@3843adc38bf1:/# python /io/sample2.py
x=[   0    1    2 ... 9997 9998 9999]
y=[1. 1. 1. ... 1. 1. 1.]
x+y=[1.000e+00 2.000e+00 3.000e+00 ... 9.998e+03 9.999e+03 1.000e+04]
root@3843adc38bf1:/#
```

UNIQ에 적용

- nvidia-docker 엔진 설치

- Docker image 생성
  - Base image를 GPU 사용이 가능하도록 설정된 이미지 활용
    - https://hub.docker.com/r/tensorflow/tensorflow
    - https://hub.docker.com/r/tensorflow/tensorflow/tags 에서 적당한 것 검색
    - tensorflow/tensorflow:2.10.0-gpu
      - ✓ tensorflow 2.10.0
      - ✓ CUDA_VERSION 11.2.1
      - ✓ CUDNN 8.1.0.77-1
      - ✓ Python 3.8.10
  - Dockerfile에 반영
    - Base image 설정

      FROM tensorflow/tensorflow:2.10.0-gpu
    - Python 관련하여 설치했던 것 조정 : 중복/덮어쓰기 방지
      - ✓ python 3.8, Tensorflow 등



Ref. https://github.com/hunsooni/traffic-signal-optimization-for-dist/blob/master/atsc-rl/multiagent_tf2/dockerize/Dockerfile.opt.gpu

동작 확인 : docker container가 실행되는 노드

```
$ sudo docker run -it --gpus 1 -v /home/tsoexp/z.docker_test/io:/uniq/optimizer/io images4uniq/optimizer:v2.0.a-gpu-test ₩
        python run.py --mode train --map doan --target-TL "SA 101" --epoch 1 --io-home io --scenario-file-path io/scenario
```

학습 실행 전

```
tsoexp@hunsooni-dev:~$ nvidia-smi
Thu Mar 16 17:59:45 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 30%   40C    P8    21W / 250W |    183MiB / 11016MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  Off  | 00000000:4D:00.0 Off |                  N/A |
| 27%   39C    P8     1W / 250W |      5MiB / 11019MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                167MiB |
|    0   N/A  N/A      1954      G   /usr/bin/gnome-shell               14MiB |
|    1   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                 4MiB |
+-----------------------------------------------------------------------------+
tsoexp@hunsooni-dev:~$
```

메모리/전기 사용 변화가 보인다

학습 실행 중

```
tsoexp@hunsooni-dev:~$ nvidia-smi
Thu Mar 16 18:06:23 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 35%   49C    P2    72W / 250W |   1190MiB / 11016MiB |      4%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  Off  | 00000000:4D:00.0 Off |                  N/A |
| 27%   39C    P8     1W / 250W |      5MiB / 11019MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                167MiB |
|    0   N/A  N/A      1954      G   /usr/bin/gnome-shell               14MiB |
|    0   N/A  N/A   1276950      C   python                           1005MiB |
|    1   N/A  N/A      1697      G   /usr/lib/xorg/Xorg                 4MiB |
+-----------------------------------------------------------------------------+
tsoexp@hunsooni-dev:~$
```

프로세스 생성이 보인다

동작 확인 : docker container 내부

```
$ sudo docker run -it --gpus 1 -v /home/tsoexp/z.docker_test/io:/uniq/optimizer/io images4uniq/optimizer:v2.0.a-gpu-test /bash/bin
```

```
root@0d177975a88a:/uniq/optimizer# python run.py --mode train --map doan --target-TL "
SA 101" --epoch 1 --io-home io --scenario-file-path io/scenario >& z &
[1] 187735
root@0d177975a88a:/uniq/optimizer# tail -f z
59% done
64% done
69% done
74% done
79% done
84% done
89% done
94% done
[Simulation End]
Elapsed Time: 33 seconds
```

Docker Container 내에서 학습 실행

학습 실행 전

```
root@0d177975a88a:/uniq/optimizer# nvidia-smi
Thu Mar 16 18:30:53 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 34%   45C    P8    20W / 250W |    183MiB / 11016MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
root@0d177975a88a:/uniq/optimizer#
```

학습 실행 중

```
root@0d177975a88a:/uniq/optimizer# nvidia-smi
Thu Mar 16 18:32:32 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.80       Driver Version: 460.80       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:01:00.0 Off |                  N/A |
| 33%   48C    P2    72W / 250W |    934MiB / 11016MiB |      4%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
root@0d177975a88a:/uniq/optimizer#
```

메모리/전기 사용 변화가 보인다

프로세스 생성은 보이지 않는다

Backup slides

- CentOS에서 nvidia-docker 엔진 설치
- 도커 이용 방법

CentOS에서 nvidia-docker 엔진 설치(1/2)

기존 설치된 것 삭제

```
sudo yum remove docker  docker-client  docker-client-latest  docker-common docker-latest ₩
                docker-latest-logrotate  docker-logrotate  docker-engine
```

Docker 설치

```
# Docker CE 저장소 설정 & 확인
sudo yum-config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo
sudo yum repolist –v

# containerd.io 패키지 설치
sudo yum install -y https://download.docker.com/linux/centos/7/x86_64/stable/Packages/containerd.io-1.4.3-3.1.el7.x86_64.rpm

# docker-ce 패키지 설치
sudo yum install docker-ce -y

# docker 서비스 실행 & 확인
sudo systemctl --now enable docker
sudo docker run --rm hello-world
```

NVIDIA Container Toolkit 설치

```
# 저장소와 GPG key 설정
distribution=$(. /etc/os-release;echo $ID$VERSION_ID)  && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/libnvidia-containe

# nvidia-container-toolkit 설치
sudo yum clean expire-cache
sudo yum install -y nvidia-container-toolkit

# NVIDIA Container Runtime을 인식하도록 Docker Daemon 설정
sudo nvidia-ctk runtime configure --runtime=docker

# 데몬 재실행 및 확인
sudo systemctl restart docker

sudo docker run --rm --runtime=nvidia --gpus all nvidia/cuda:11.6.2-base-ubuntu20.04 nvidia-smi
sudo docker run --rm  --gpus all  tensorflow/tensorflow:2.10.0-gpu nvidia-smi
```

Ref. https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html

CentOS에서 nvidia-docker 엔진 설치(2/2)

- CentOS에서 리눅스 도커 엔진 설치 에러

```
$ yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
 File "/usr/bin/yum-config-manager", line 135
 except yum.Errors.RepoError, e:
```

yum-config-manager에서 바라보는
python이 2.x버전이 아니라 3.x인것이 문제

$ **vi** /usr/bin/yum-config-manager

```
#!/usr/bin/python -tt

import os, os.path
import sys
import re
import yum
```

```
[tsoexp@uniq1 bin]$ ls -la python
lrwxrwxrwx 1 root root 7  7월   7  2022 python -> python3
[tsoexp@uniq1 bin]$
```


```
#!/usr/bin/python2 -tt

import os, os.path
import sys
import re
import yum
```

Ref. https://hyen4110.tistory.com/m/127

도커 이용 방법

- 이미지 저장소에서 이미지 다운로드
  - 저장소 : https://hub.docker.com/
  - 이미지 이름 : images4uniq/optimizer:v2.x.x-gpu

  - $ sudo docker pull images4uniq/optimizer:v2.x.x-gpu

- 시나리오(데이터) 준비
  - /home/tsoexp/io/scenario 아래 doan 시나리오 복사

- 도커 이미지 실행
  - $ sudo docker run --rm --gpus all tensorflow/tensorflow:2.10.0-gpu nvidia-smi
  - $ sudo docker run --gpus "device=1" -it tensorflow/tensorflow:2.10.0-gpu /bin/bash
  - $ sudo docker run --gpus "device=1" -v /home/tsoexp/io:/uniq/optimizer/io ₩
    images4uniq/optimizer:v2.x.x-gpu ₩
    python run.py --mode train –map doan --epoch 1 --io-home io ₩
    --scenario-file-path io/scenario