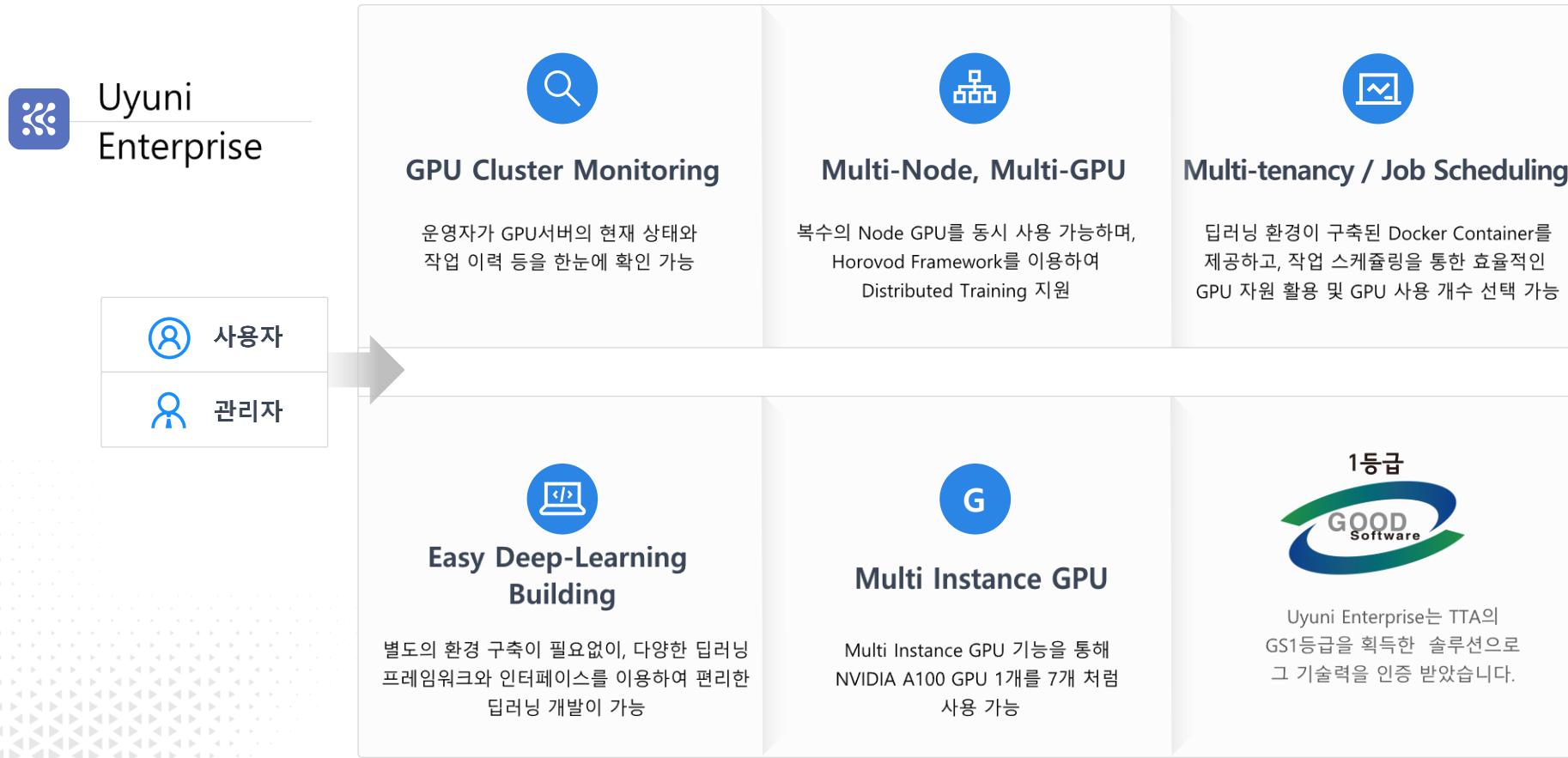
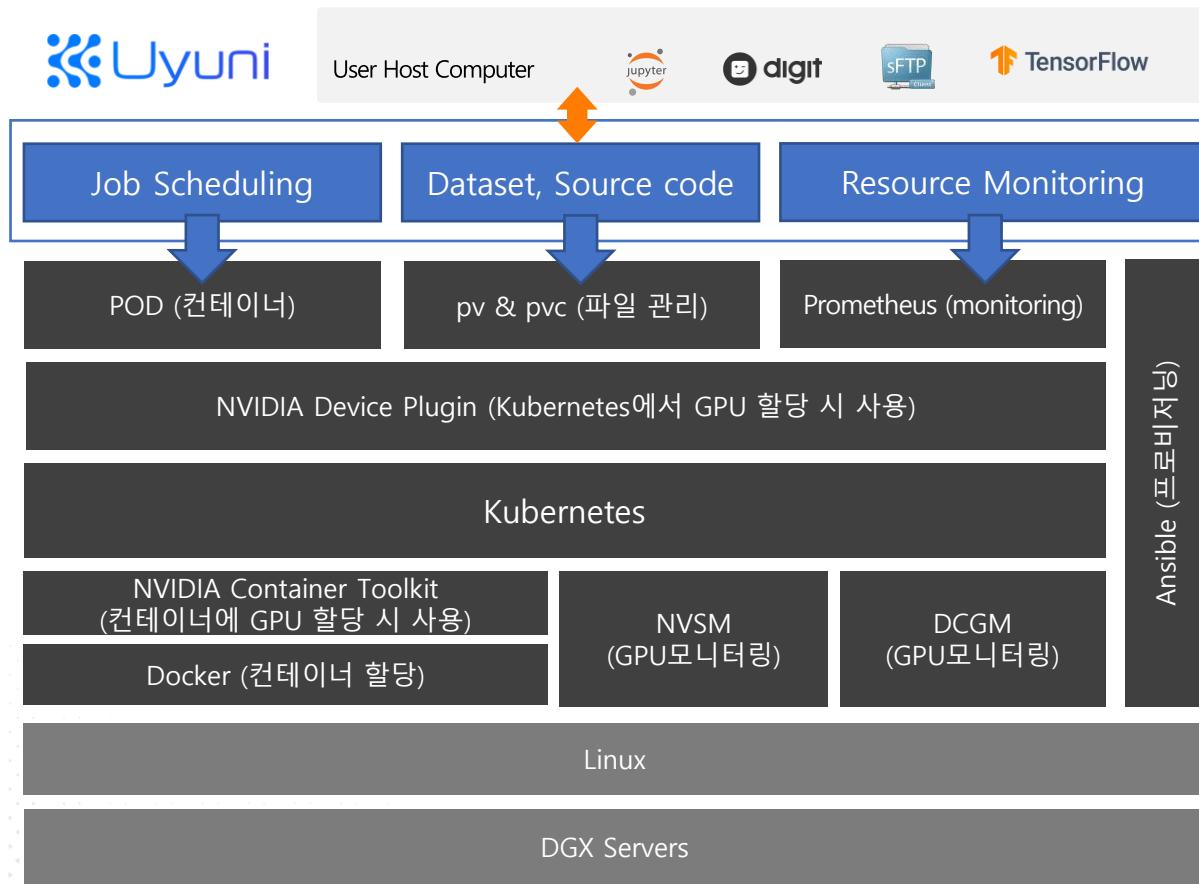


GPU Infra Management Deep Learning Platform Solution Uyuni Enterprise



Uyuni는 단일 GPU 서버에서 대규모 GPU 클러스터 환경까지 모든 환경에서 최적의 딥러닝 모델 개발 환경을 제공하는 Docker / Kubernetes 기반의 딥러닝 플랫폼 솔루션입니다.





제품사양

- Node :** 1대 이상(GPU 1개 이상)
- OS :** ubuntu 16.04 이상
- RAM :** 8GB 이상
- DISK :** 128GB 이상
- GPU :** CUDA 지원 및 NVIDIA Driver 설치 필수

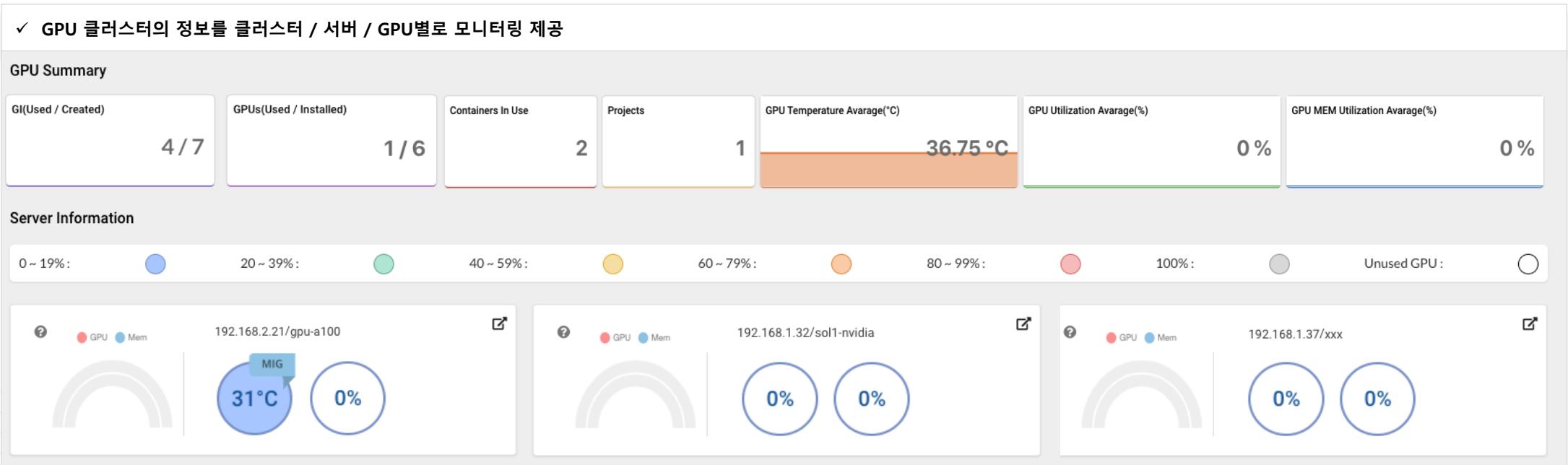
구성 소프트웨어

Linux Ubuntu 16.04	pyCaffe 0.15.14
Nvidia Graphic Driver 410.48	Python 2.7
Docker 18.03	NumPy 1.16.3
Nvidia-Docker 2.03	SciPy 0.17.0
JDK 8.0	Keras 2.2.4
Tensorflow 1.12.0 GPU version	Pyyaml 5.1
CUDA 9.0	HDF5 1.8.11 and h5py 2.6.0
cuDNN v7	DIGITS 6.1.0
GCC 5.4.0	pyCaffe 0.15.14
BLAS via ATLAS 3.10.1, MKL	Python 2.7
Boost >= 1.58	NumPy 1.16.3
Protobuf 3.2.0, glog 0.3.3-1, gflags 2.0-1, hdf5 1.8.11	SciPy 0.17.0

주요 소프트웨어	상세 소프트웨어	내용
Operation System Virtual Machine	Linux Ubuntu 16.04	OS
	Nvidia Graphic Driver 410.48	서비스 운영 환경을 묶어서 배포, 실행하는 경량 컨테이너 기술
	Docker 18.03	GPU 사용을 위한 Nvidia-docker
	Nvidia-Docker 2.03	자바 애플리케이션을 구축하기 위한 핵심 플랫폼 구성요소
Tensorflow	JDK 8.0	
	Tensorflow 1.12.0 GPU version	기계 학습과 딥러닝을 위한 오픈소스 라이브러리
	CUDA 9.0	
	cuDNN v7	
	GCC 5.4.0	
Caffe	BLAS via ATLAS 3.10.1, MKL	
	Boost >= 1.58	
	Protobuf 3.2.0, glog 0.3.3-1, gflags 2.0-1, hdf5 1.9.11	표현식, 속도 모듈화를 고려한 딥러닝 프레임워크
	pvCaffe 0.15.14	
Keras	Keras 2.2.4	
	Pvyaml 5.1	Tensorflow, Theano를 위한 딥러닝 라이브러리
	HDF5 1.8.11 and h5py 2.6.0	
DIGITS	 DIGITS 6.1.0	딥러닝 모델 트레이닝을 위한 웹 앱

✓ 운영자가 GPU서버의 상태를 한눈에 확인

- 그래프와 테이블의 형태로 현재 상태와 작업 이력 등을 확인 가능



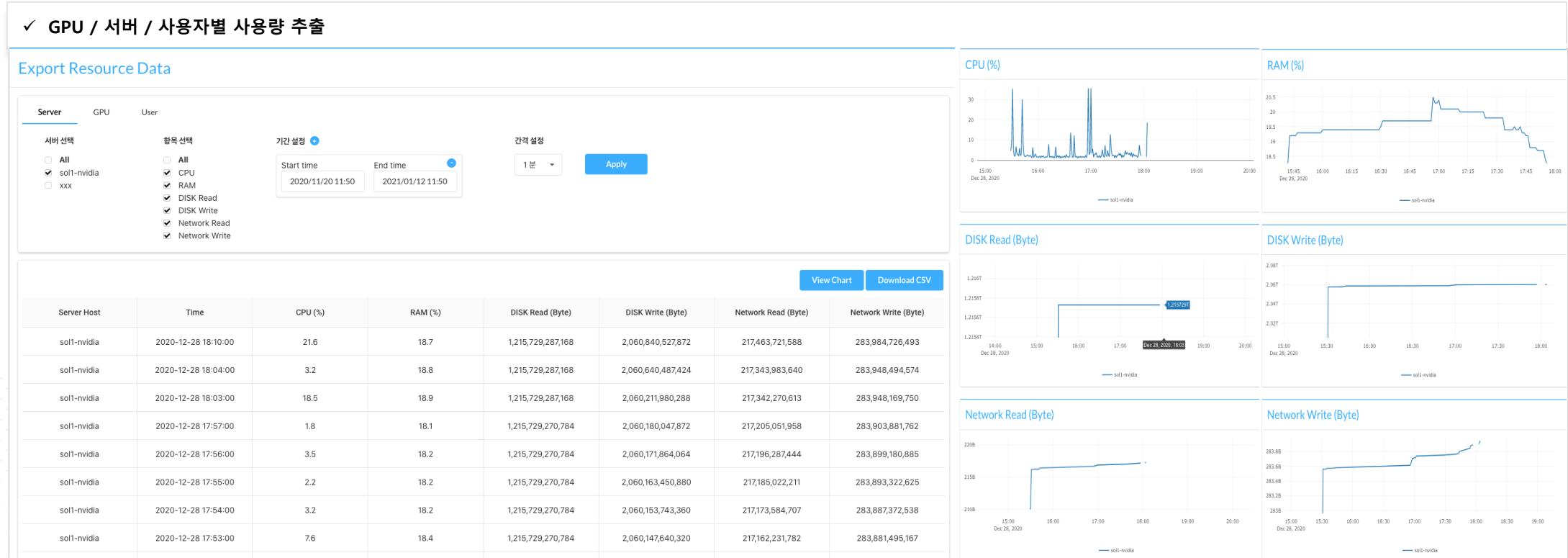
✓ GPU 사용 이력 확인

✓ GPU를 할당 받아 작업을 수행한 이력 확인 가능

Job Logs ?Docker Image RequestDataSet RequestSource Code RequestContainer Request

Project Name ?	User ID	Image	Type	GPU	Request Time	Start Time	End Time	Status ?	Actions
Horovod_test	admin	xiilab/nvidia_e	Multi	2	2021-01-12 11:43:25	2021-01-12 11:43:23	2021-01-13 11:43:23	ACTIVE	ⓘ 📈 ▶ ⚡ ✖
test02	user2	tensorflow/tensorflow	Single	1	2021-01-12 11:44:20	2021-01-12 11:44:19	2021-01-13 11:44:19	DENIED	ⓘ 📈 ▶ ⚡ ✖
test04	user1	tensorflow/tensorflow	Single	1	2021-01-12 11:44:47	2021-01-12 11:44:46	2021-01-13 11:44:46	READY	ⓘ 📈 ▶ ⚡ ✖
test-namespace-04	user2	xiilab/nvidia_e:latest	Multi	2	2021-01-12 10:20:53	2021-01-12 10:19:52	2021-01-13 10:19:52	TERMINATED	ⓘ 📈 ▶ ⚡ ✖

✓ GPU / 서버 / 사용자별 사용량 추출



✓ 여러 개발자에게 독립적인 개발 환경 제공

- 딥러닝 환경이 구축된 Docker Container을 이용

✓ 효율적인 GPU자원을 활용하여 딥러닝 작업 스케줄링 제공

- GPU 자원 할당을 자동/수동 변경 가능
- GPU 자원 관리 정책을 동적으로 변경 가능

✓ 개발자가 원하는 수의 GPU자원을 할당 받아 딥러닝 수행

- 단일/복수 GPU 할당 지원
- Nvidia Docker을 이용하여 GPU자원을 할당한 Container 생성

✓ 싱글/ 멀티노드 지원

Container Request (Step. 1)

Project Name Used Require Docker Image Port

Project name*

Horovod_test

Multi node ?

✓ 원하는 수의 GPU자원 요청 가능

Container Request (Step. 2)

GPU* ? 2

Use	Host Name	GPU name
<input checked="" type="checkbox"/>	sol1-nvidia	TITAN Xp #0
<input type="checkbox"/>	sol1-nvidia	TITAN Xp #1
<input checked="" type="checkbox"/>	xxx	GeForce GTX 980 Ti #0

✓ 복수의 Node의 GPU를 동시 사용 가능

- Hovrovod Framework를 이용하여 분산학습(Distributed training) 지원

✓ Multi Node & GPU 설정 가능

Container Request (Step. 2)

Project Name: **Used Require**

Start time: 2019-10-18 14:26 End time: 2019-10-19 14:26

GPU: 10

Container Information

Container Name: **uyuni35_master**

Image Name: xillab/nvidia_eib

Container IP: 192.168.2.11

Used GPU

Host name	Used GPU
dgx1	GPU#0 , GPU#1 , GPU#2 , GPU#3 , GPU#4 , GPU#5 , GPU#6 , GPU#7
dgx2	GPU#0 , GPU#1

Port Information

Descript	Container Port	Host Port	Execute
tensorboard	6006	10023	Execute
jupyter	8888	10024	Execute

Password: bTTcg

Interpolated: bash -c 'mpirun -np 2 -H dgx1:8,dgx2:1 -bind-to none -map-by slot -x NCCL_DEBUG=INFO -x NCCL_IB_DISABLE=1'

Container Monitoring Save Container Close

Multi_Node_Test001

uyuni35_master

Container Information

Container Name: **uyuni35_master**

Image Name: xillab/nvidia_eib

Container IP: 192.168.2.11

Used GPU

Host name	Used GPU
dgx1	GPU#0 , GPU#1 , GPU#2 , GPU#3 , GPU#4 , GPU#5 , GPU#6 , GPU#7
dgx2	GPU#0 , GPU#1

Port Information

Descript	Container Port	Host Port	Execute
tensorboard	6006	10023	Execute
jupyter	8888	10024	Execute

Password: bTTcg

Interpolated: bash -c 'mpirun -np 2 -H dgx1:8,dgx2:1 -bind-to none -map-by slot -x NCCL_DEBUG=INFO -x NCCL_IB_DISABLE=1'

Container Monitoring Save Container Close

Multi_Node_Test001

uyuni35_master GPU 3% GPU MEM 2%

uyuni35_master

uyuni35_worker

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0714 - acc: 0.9779

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0714 - acc: 0.9779

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0799 - acc: 0.9766 48/234 [=====>.....] - ETA: 4s - loss: 0.0712 - acc: 0.9779

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0798 - acc: 0.9764

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0740 - acc: 0.9781

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0798 - acc: 0.9764 52/234 [=====>.....] - ETA: 4s - loss: 0.0742 - acc: 0.9781

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0803 - acc: 0.9763 54/234 [=====>.....] - ETA: 4s - loss: 0.0761 - acc: 0.9773

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0772 - acc: 0.9771 56/234 [=====>.....] - ETA: 4s - loss: 0.0817 - acc: 0.9757

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0771 - acc: 0.9770 58/234 [=====>.....] - ETA: 4s - loss: 0.0820 - acc: 0.9758

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0820 - acc: 0.9757 60/234 [=====>.....] - ETA: 4s - loss: 0.0766 - acc: 0.9771

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0815 - acc: 0.9759 62/234 [=====>.....] - ETA: 4s - loss: 0.0767 - acc: 0.9773

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0761 - acc: 0.9774 64/234 [=====>.....] - ETA: 4s - loss: 0.0800 - acc: 0.9766

2019-10-18 14:41:10.0 0.49/234 [=====>.....] - ETA: 4s - loss: 0.0761 - acc: 0.9774 64/234 [=====>.....] - ETA: 4s - loss: 0.0800 - acc: 0.9766

Close

⌚ 개발자가 별도의 개발 환경을 구축하지 않고 딥러닝 수행

- 개발환경이 구축된 Docker Image를 이용하여 Container 생성

✓ 자주 사용하는 딥러닝 프레임워크와 툴이 설치된 Docker Image를 이용하여 Container 생성

horovod_test

Container Information

Container Name	uyuni157_master
Image Name	xiilab/nvidia_e:190717
Container IP	192.168.1.32

Used GPU

	Host name	Used GPU
soil-nvidia		GPU#0 , GPU#1
soil-nvidia-2		GPU#0 , GPU#1

Port Information

	Descript	Container Port	Host Port	Execute
8888	jupyter	8888	10005	
	ssh	22	10004	
	digits	5000	10006	

Password

Interpolated

```
bash -c 'mpirun -verbose -np 4 -H 192.168.1.32:2,192.168.1.34:2 -mca btl_tcp_if_exclude docker0,lo,docker_gwbridge -mca plm_rsh_args "-p $HOROVOD_PORT" python3 /horovod_examples/keras_mnist_advanced.py'
```

⌚ 다양한 딥러닝 프레임워크를 이용하여 딥러닝 수행

- 딥러닝 프레임워크가 설치된 Docker Images 동적 추가 가능 : Docker Hub, NGC
- 최신 딥러닝 프레임워크를 지원하는 Docker Image 및 Sample Project 업데이트

✓ 자주 사용하는 딥러닝 프레임워크와 툴이 설치된 Docker Image를 이용하여 Container 생성

Docker Image Information

Add Image

Image Repository

- Local Directory
- Manual (selected)

* 표시는 필수 입력입니다.

Image Name	Image Tag	Image Type	Image Repository	User ID	Authority	Status	Option	
xiilab/nvidia_e	latest	Multi	manual	admin	public	COMPLETE		
tensorflow/tensorflow	2.4.0-gpu-jupyter-u					COMPLETE		
xiilab/horovod	0.21.0-tf-2.4.0-torch17.1-met1.8.0-py3.6-cuda11.0					COMPLETE		

Image Name: xiilab/nvidia_e

Image Tag: latest

Execute Option

Ports

Port	Descript	Execute	Add	
8888	jupyter	jupyter notebook	<input checked="" type="checkbox"/>	
22	ssh		<input checked="" type="checkbox"/>	
5000	digit	/root/DIGITS-6.1.0/digits-devserver	<input checked="" type="checkbox"/>	

✓ 사용 편의성 향상

- 컨테이너의 상태 및 웹터미널 다중 팝업을 지원

✓ 컨테이너의 상태 및 웹터미널 다중 팝업을 지원

☑ 다양한 인터페이스를 이용하여 편리하게 딥러닝 수행 및 검증

- 웹터미널, TensorBoard, Jupyter notebook, DIGITS등 지원

✓ Argument 입력

Container Request (Step. 6)



Job Creation

Container Manual Mode Create a container without running the source code.

Source Code Repository Horovod

Source Code Name

Argument

<input type="text" value="-X"/>	<input type="text" value="NCCL_DEBUG=INFO"/>	<input type="button" value=""/>
<input type="text" value="-X"/>	<input type="text" value="LD_LIBRARY_PATH"/>	<input type="button" value=""/>
<input type="text" value="-X"/>	<input type="text" value="PATH"/>	<input type="button" value=""/>
<input type="text" value="-mca plm_rsh_args"/>	<input type="text" value="-p \$HOROVOD_PORT"/>	<input type="button" value=""/>

✓ WEB Code Editor

Code Editor

13

gitmodules Dockerfile.cpu setup.py LICENSE docs buildkit horovod horovod.ex Dockerfile.test.cpu Dockerfile.gpu Dockerfile.compose.test.yml Jenkinsfile.ppc64le horovod.ids Dockerfile.test.gpu git github clang-format NOTICE README.md examples readthedocs.yml build-docker-images.sh gitignore CODE_OF_CONDUCT.md CONTRIBUTING.md bin MANIFEST.in

keras_mnist_advanced.py x setup.py x

```
1829 except Exception:
1830     msg = "INFO: Cannot detect if MKLDNN is enabled in MXNet. Please \
1831         set MXNET_USE_MKLDDNN=1 if MKLDNN is enabled in your MXNet build."
1832     if 'linux' not in sys.platform:
1833         # MKLDNN is only enabled by default in MXNet Linux build. Return
1834         # False by default for non-linux build but still allow users to
1835         # enable it by using MXNET_USE_MKLDDNN env variable.
1836     print(msg)
1837     return os.environ.get('MXNET_USE_MKLDDNN', '0') == '1'
1838 else:
1839     try:
1840         import mxnet as mx
1841         mx_libs = mx.libinfo.find_lib_path()
1842         for mx_lib in mx_libs:
1843             output = subprocess.check_output(['readelf', '-d', mx_lib])
1844             if 'mklldn' in str(output):
1845                 return True
1846             return False
1847     except Exception:
1848         print(msg)
1849     return os.environ.get('MXNET_USE_MKLDDNN', '0') == '1'
1850
1851 def is_mx_cuda():
1852     try:
1853         from mxnet import runtime
1854         features = runtime.Features()
1855         return features.is_enabled('CUDA')
1856     except Exception:
1857         if 'linux' in sys.platform:
1858             try:
1859                 import mxnet as mx
1860                 mx_libs = mx.libinfo.find_lib_path()
1861                 for mx_lib in mx_libs:
1862                     output = subprocess.check_output(['readelf', '-d', mx_lib])
1863                     if 'cuda' in str(output):
1864                         return True
1865             return False
1866     except Exception:
```

✓ MIG 기능을 통해 NVIDIA A100 GPU 1 개를 7 개의 GPU 처럼 사용 가능

- 새로운 MIG(Multi-Instance-GPU) 기능을 통해 NVIDIA A100 GPU를 최대 7 개의 개별 GPU 인스턴스로 분할하여 사용할 수 있습니다.
- Uyuni에서는 7 개로 분할된 GPU의 가상화를 지원합니다.
- 컨테이너 생성 시 Profile 선택 가능합니다.

✓ NVIDIA A100 GPU를 최대 7 개의 개별 GPU 인스턴스로 분할

Create GPU Instance

GPU to create instance
 A100-SXM4-40GB #6 A100-SXM4-40GB #7

A100-SXM4-40GB #6

8 memory, 7 compute	
4 memory, 4 compute	4 memory, 3 compute
4 memory, 3 compute	4 memory, 3 compute
2 memory, 2 compute	2 memory, 2 compute
1 memory, 1 compute	1 memory, 1 compute

Create

✓ 컨테이너 생성 시 선택한 GPU Instance Profile를 바탕으로 GPU 인스턴스를 할당 받음

Container Request (Step. 2)

Project Name: 2 Used Require Docker Image: 3 Port: 4 Dataset: 5 Job Creation: 6 Review: 7

Start time: 2020-11-12 17:20 End time: 2020-11-13 17:20

GPU*: 1

Profile*: 1g.5gb

공급사 **XIIIab** [씨이랩]

판매사 **ITmaya**™

T H A N K Y O U

아이티마야 MLOps.

담당자. 임 중 춘 팀장

Mail. chun@itmaya.co.kr

Mobile. 010-2279-544

