
ML-Sys WS 2017 @ NIPS
Nako Sung, Minkyu Kim, Hyunwoo Jo, Youngil Yang, Jinwoong Kim, Leonard Lausen, Youngkwan Kim, Gayoung Lee, Donghyun Kwak, Jung-Woo Ha, and Sunghun Kim

CLOVA AI Research (CLAIR), NAVER | LINE, Search Solution, NAVER Webtoon, HKUST
What is NSML?

- A machine learning platform that enables you focus on your models
- Two options: on-premise / PaaS
The #1 programmer excuse for legitimately slacking off - 2017 version

THEN

"MY CODE'S COMPILING."

HEY! GET BACK TO WORK!

COMPILING!

OH. CARRY ON.

NOW

"MY MODEL IS TRAINING"

HEY! GET BACK TO WORK!

TRAINING!

OH. CARRY ON.
Wasted Time
Importance of Fast Machines (Multiple Servers and GPUs)

ML Research Challenges: Incidental Tasks

https://www.sportskeeda.com/f1/what-happens-during-f1-pit-stop
ML Research Challenges: Resource Scheduling and Utilization

14 GPUs available but only 7 GPUs can be used in a single machine.
ML Research Challenges: Hyperparameter Tuning
ML Research Challenges: Multiple Experiments

- **DONE** γ=0.1
- **DONE** γ=0.2
- **TRAINING** γ=0.3, K=1
- **TRAINING** γ=1e-2

 Tensor board

 Visdom
ML Research Challenges: Isolated Researchers

Challenges

• Slack
• Incidental Tasks
• Inefficient resource utilization
• Naive hyperparameter tuning
• Painful keeping track of multiple sessions
• Isolated researchers
Requirements of ML Platforms

- Resource Management
  - Better computational resource management
- Data Management
  - Post datasets once and reuse them for multiple models
  - Share datasets with others
- Serverless Configuration
  - No framework / library lock-in
  - Easy and lightweight task submission
Requirements of ML Platforms

- Experiment Management and Visualization
  - Parallel runs with different jobs priorities
  - Automatic visualization and summarization of learning progress
- Leaderboard
  - Leaderboard for each dataset to compare models and hyper parameters
- AutoML
  - Experiment performance prediction based on previously run experiments.
  - Automatic hyper parameter optimization based on the performance predictions.
Limitations of Previous Solutions

- Vendor lock-in (Cloud service)
- Inefficient model experiments
- Inconsistent research environments
- Still hard to keep track of experiments
This work was done for NCSoft and was presented at Nvidia GTC Korea 2015.
My Previous Work in Early 2015

This work was done for NCSoft and was presented at Nvidia GTC Korea 2015.
Every dataset, session and model have uniform resource identifier.

- CIFAR_10
  - CIFAR 10 dataset
- CIFAR_10/researcher_A/24
  - research_A's 24th session for CIFAR_10
- CIFAR_10/researcher_A/24/322
  - Snapshot from epoch 322
Easy One-Liner CLI
Easy One-Liner CLI

Dataset registration

/app/examples/09_NMT$ nsml dataset push NMT_EN_KR ./nmt_en_kr
Easy One-Liner CLI

Dataset registration

```
/app/examples/09_NMT$ nsml dataset push NMT_EN_KR ./nmt_en_kr
```

Train

```
/app/examples/09_NMT$ nsml run -d NMT_EN_KR
Session clair/NMT_EN_KR/1 is running
```
Easy One-Liner CLI

Dataset registration

```
/app/examples/09_NMT$ nsml dataset push NMT_EN_KR ./nmt_en_kr
```

Train

```
/app/examples/09_NMT$ nsml run -d NMT_EN_KR
Session clair/NMT_EN_KR/1 is running
```

Serve

```
/app/examples$ echo Hello | nsml infer clair/NMT_EN_KR/1/12
안녕하세요
```
Parallel Experiments to Kill Slack

Distributed responses

Exp. #1
Exp #2. vari. 1
Exp #2. vari. 2
Exp #3

Time

```
$ /app/examples/02_mnist$ nsml run main.py -- --lr 0.1
Session KR18284/None/12 is running
$ /app/examples/02_mnist$ nsml run main.py -- --lr 0.01
Session KR18284/None/13 is running
$ /app/examples/02_mnist$ nsml run main.py -- --lr 0.001
Session KR18284/None/14 is running
$ /app/examples/02_mnist$ nsml ps
Name                  Created     Args
---------------------- ----------- --------------------------
KR18284/None/14       just now    main.py --lr 0.001
KR18284/None/13       just now    main.py --lr 0.01
KR18284/None/12       seconds ago main.py --lr 0.1
```
Need to Visualize

• Balance your brain to understand without effort

Flexible Analysis

Your code @1

Your code @2

Your code @3

NSML

Visualization tool

```python
nsml.report(
    epoch=epoch+config.iteration,
    epoch_total=config.epochs,
    iter=iter_idx,
    iter_total=total_length,
    batch_size=batch_size,
    train_loss=running_loss / num_runs,
    train_accuracy=running_acc / num_runs,
    scope=locals()
)
```
AL01272634:examples user$ ls
01_hello_nsml    04_freeze         07_vae_gan
02_mnist         05_ladder_networks 08_LiteNet
03_visdom        06_text           09_movie_review
AL01272634:examples user$
AL01272634:examples user$ ls
01_hello_nsml  04_freeze  07_vae_gan
02_mnist  05_ladder_networks  08_LiteNet
03_visdom  06_text  09_movie_review
AL01272634:examples user$
Dynamic Control Flow

**Typical training loop**
- Forward pass
- Backward pass
- Communicate to NSML

```python
nsml.report(
    epoch=epoch+config.iteration,
    epoch_total=epoch+config.epochs,
    iter=iter_idx,
    iter_total=total_length,
    batch_size=batch_size,
    train_loss=running_loss / num_runs,
    train_accuracy=running_acc / num_runs,
    scope=locals()
)
```

**Command queue**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 model</td>
<td>Watch a variable</td>
</tr>
<tr>
<td>2 change_lr(0.2)</td>
<td>Change a hyper parameter on the fly</td>
</tr>
<tr>
<td>3 nsml.save('quick')</td>
<td>Save current snapshot</td>
</tr>
<tr>
<td>4 nsml.load(424)</td>
<td>Load saved snapshot</td>
</tr>
<tr>
<td>5 vis.image(model.generate(2))</td>
<td>Generate an image to visdom</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>[Batch/Total]</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
</tr>
<tr>
<td>4</td>
<td>[1260/60000 (3%)]</td>
</tr>
<tr>
<td>4</td>
<td>[2560/60000 (4%)]</td>
</tr>
<tr>
<td>4</td>
<td>[3200/60000 (5%)]</td>
</tr>
<tr>
<td>4</td>
<td>[3840/60000 (6%)]</td>
</tr>
<tr>
<td>4</td>
<td>[4480/60000 (7%)]</td>
</tr>
<tr>
<td>4</td>
<td>[5120/60000 (9%)]</td>
</tr>
<tr>
<td>4</td>
<td>[5760/60000 (10%)]</td>
</tr>
<tr>
<td>4</td>
<td>[6400/60000 (11%)]</td>
</tr>
<tr>
<td>4</td>
<td>[7040/60000 (12%)]</td>
</tr>
<tr>
<td>4</td>
<td>[7680/60000 (13%)]</td>
</tr>
<tr>
<td>4</td>
<td>[8320/60000 (14%)]</td>
</tr>
<tr>
<td>4</td>
<td>[8960/60000 (15%)]</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>60000 (1%)</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>2560/60000 (4%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>3200/60000 (5%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>3940/60000 (6%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>4460/60000 (7%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>5120/60000 (9%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>5760/60000 (10%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>6400/60000 (11%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>7040/60000 (12%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>7680/60000 (13%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>8320/60000 (14%)</td>
</tr>
<tr>
<td>Train Epoch</td>
<td>8960/60000 (15%)</td>
</tr>
</tbody>
</table>
CLI

• Base of advanced features like save, load, infer, ...

```
/tmp$ nsml run -d mnist_torch
Session KR18284/mnist_torch/12 is running
/tmp$ nsml exec KR18284/mnist_torch/12 model
Net (  
  (conv1): Conv2d(1, 10, kernel_size=(5, 5), stride=(1, 1))  
  (conv2): Conv2d(10, 20, kernel_size=(5, 5), stride=(1, 1))  
  (conv2_drop): Dropout2d (p=0.5)  
  (fc1): Linear (320 -> 50)  
  (fc1_bn): BatchNorm1d(50, eps=1e-05, momentum=0.1, affine=True)  
  (fc2): Linear (50 -> 10)  
  (conv2_bn): BatchNorm2d(20, eps=1e-05, momentum=0.1, affine=True)  
  )
```
Bring Your Own Workspace

- (Almost) Nothing to learn
- Cached (Fast)
Bring Your Own Workspace

- (Almost) Nothing to learn
- Cached (Fast)
No Framework Lock-in

- **tensorflow/tensorflow**
  - By tensorflow • Free Under the Docker Community License
  - Official docker images for deep learning framework TensorFlow (http://www.tensorflow.org)

- **floydhub/pytorch**
  - By floydhub • Free Under the Docker Community License

- **gw000/keras**
  - By gw000 • Free Under the Docker Community License
  - Keras in Docker for reproducible deep learning on CPU or GPU

- **kaixin/cuda-torch**
  - By kaixin • Free Under the Docker Community License
  - Ubuntu Core 14.04 + CUDA + Torch7 (including JITorch)
Interactive Mode

```
/app/examples/02_mnist$ n
```

GPU server 10.0.0.1

```
python your_model.py
```

stdout
Interactive Mode

/app/examples/02_mnist$ n

GPU server 10.0.0.1

python your_model.py

stdout
Pragmatic Research
기대안하고 빠야 할듯....

★ ★ ★ ★ ★ 3
기대안하고 빠야 할듯....
기대안하고 빠야 할듯....

3

기대안하고 빠야 할楪....
Collaboration and Competition
Leaderboard, CI-ML
New Workflow for ML Research

Collaboration and Competition
Leaderboard, CI-ML
Collaborative Research

• Easy to reproduce and extend other’s research.
Collaborative Research

• Easy to reproduce and extend other’s research.
Cohesive and Competitive

Dataset-centric environment
Models are ranked automatically

Standardized and Quantified
Easy to compete
Towards AutoML
Cohesive and Competitive

Dataset-centric environment
Models are ranked automatically

Standardized and Quantified
Easy to compete
Towards AutoML
AutoML

• Quantitive model analysis makes ML workflow as a gym of AutoML
Seamless Connection to Services

SOTA server

```
$ curl -X POST https://service.nsml.navercorp.com/ASR 
  -H "Content-Type: audio/wav" 
  --data-binary "@sample.wav"
```
Seamless Connection to Services

SOTA server

REST API
https://service.nsml.navercorp.com/ASR

```
$ curl -X POST https://service.nsml.navercorp.com/ASR \
-H "Content-Type: audio/wav" \
--data-binary @sample.wav
```
Q1. 2018
Thank you

Several Hundreds of GPUs for this alpha (free)

https://research.clova.ai/nsml-alpha