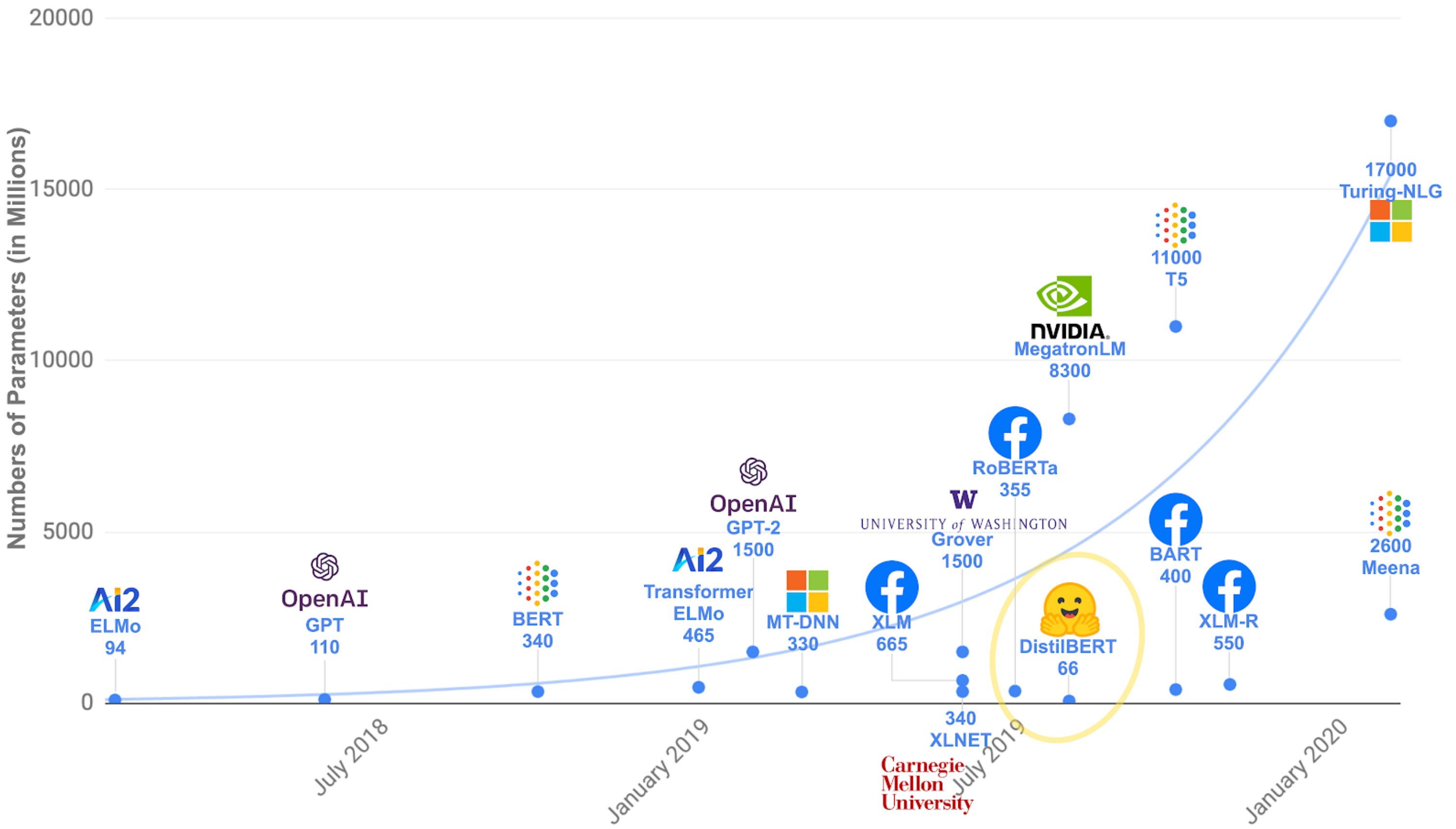
Modern Distributed Data-Parallel Large-Scale Pre-training Strategies For NLP models

Hao Bai
Junior, Computer Engineering
National Center For Supercomputing Applications, UIUC
Champaign, Illinois, USA

Author & Supervisor

- AUTHOR | Jack Bai (Junior UG, CompE, UIUC)
  - Take a look at my profile at https://www.jackgethome.com/
  - My paper preprint is on ArXiV: https://arxiv.org/pdf/2206.06356.pdf
- ADVISOR | Volodymyr Kindratenko (Assistant Director at NCSA)
  - Research interests mainly in HPC
  - Take a look at his profile at https://ece.illinois.edu/about/directory/faculty/kindrtnk
  - Has lots of supercomputers, much acknowledge to my prof

Why do we pick an NLP pre-trained model?



Pain...

I’m sure everyone who used PyTorch or TensorFlow experiences this kind of pain.

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 375.66                 Driver Version: 375.66                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 108...  Off  | 0000:02:00.0     Off |                  N/A |
| 30%   53C    P2    63W / 250W |   8779MiB / 11172MiB |      6%      Default |
+-------------------------------+----------------------+----------------------+
|   1  GeForce GTX 108...  Off  | 0000:04:00.0     Off |                  N/A |
| 30%   53C    P2    59W / 250W |   8663MiB / 11172MiB |      6%      Default |
+-------------------------------+----------------------+----------------------+
|   2  GeForce GTX 108...  Off  | 0000:83:00.0     Off |                  N/A |
| 31%   55C    P2    62W / 250W |   8663MiB / 11172MiB |      4%      Default |
+-------------------------------+----------------------+----------------------+
|   3  GeForce GTX 108...  Off  | 0000:84:00.0     Off |                  N/A |
| 28%   50C    P2    61W / 250W |   8989MiB / 11172MiB |      6%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|    0     85736    C   python                                       8777MiB |
|    1     85736    C   python                                       8661MiB |
|    2     85736    C   python                                       8661MiB |
|    3     85736    C   python                                       8987MiB |
+-----------------------------------------------------------------------------+
```

Dream?

How to improve the volatile GPU utility like this?

```
xiongyu@ubuntu:~$ watch nvidia-smi
|    0     47741      C    ./mtcnn_c                                   461M
Every 2.0s: nvidia-smi

Fri Jun 29 11:25:43 2018
+-----------------------------------------------------------------------------
| NVIDIA-SMI 390.48                 Driver Version: 390.48
|-------------------------------+----------------------+----------------------
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. E
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute
|===============================+======================+======================
|   0  Tesla M40           On   | 00000000:03:00.0 Off |
|  0%   57C    P0   217W / 250W |  10806MiB / 11448MiB |     99%      Defau
+-------------------------------+----------------------+----------------------
|   1  Tesla M40           On   | 00000000:04:00.0 Off |
|  0%   58C    P0   200W / 250W |   9992MiB / 11448MiB |     99%      Defau
+-------------------------------+----------------------+----------------------
|   2  Tesla M40           On   | 00000000:84:00.0 Off |
|  0%   56C    P0   222W / 250W |   9955MiB / 11448MiB |     99%      Defau
+-------------------------------+----------------------+----------------------
|   3  Tesla M40           On   | 00000000:85:00.0 Off |
|  0%   57C    P0   203W / 250W |   9960MiB / 11448MiB |     98%      Defau
+-------------------------------+----------------------+----------------------

+-----------------------------------------------------------------------------
| Processes:                                                        GPU Memo
|  GPU       PID   Type   Process name                            Usage
|=============================================================================
|    0     35346      C   python2                                      110M
|    0     55323      C   python                                     10673M
|    1     55323      C   python                                      9970M
```

Model Parallel And Data Parallel

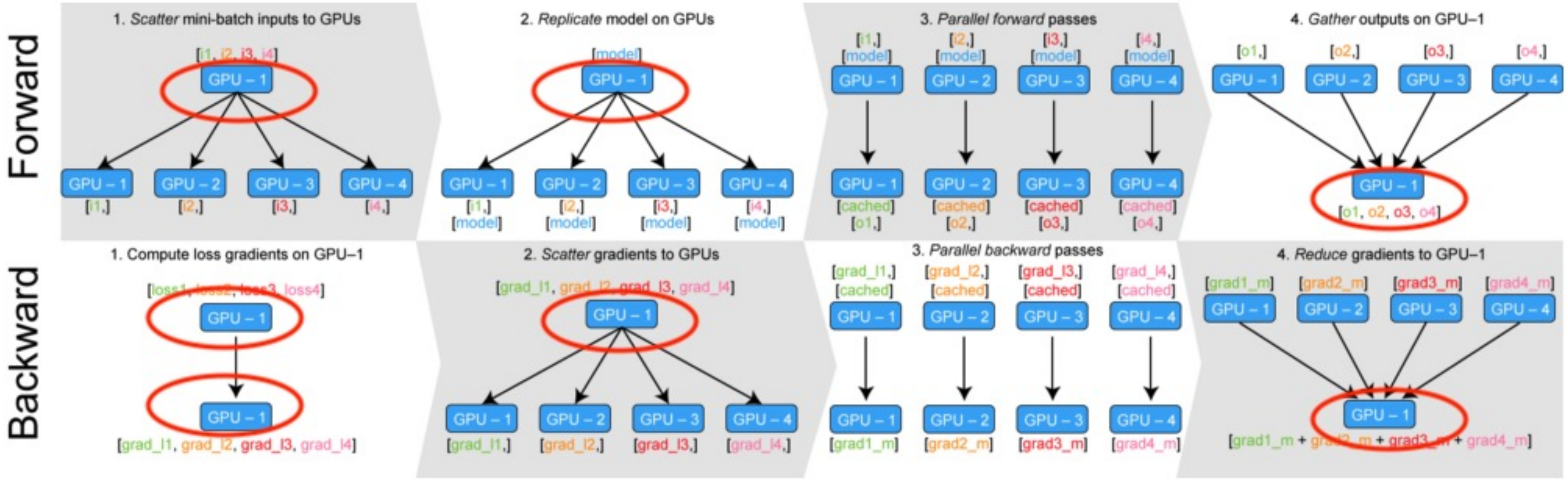
Not exactly a dream.

We have model parallel and data parallel to help.

- Model parallel: parts of model on different GPUs.
- Data parallel: scatter and gather data on different GPUs.

Data Parallel In PyTorch

- Let's take PyTorch for example.
- **Who encountered to use nn.DataParallel library?**
- I'm sure you're worried about the performance. When you're doing a Kaggle, you get anxious about your code.



Forward and Backward passes with torch.nn.DataParallel

Bottleneck

- It's not exacly your fault. We've got similar results.
- Why? Because it's Single Parameter Server training. Can you get the bottleneck?
- Yes, it's even slower than the baseline with only 1 GPU. Don't use it that way!

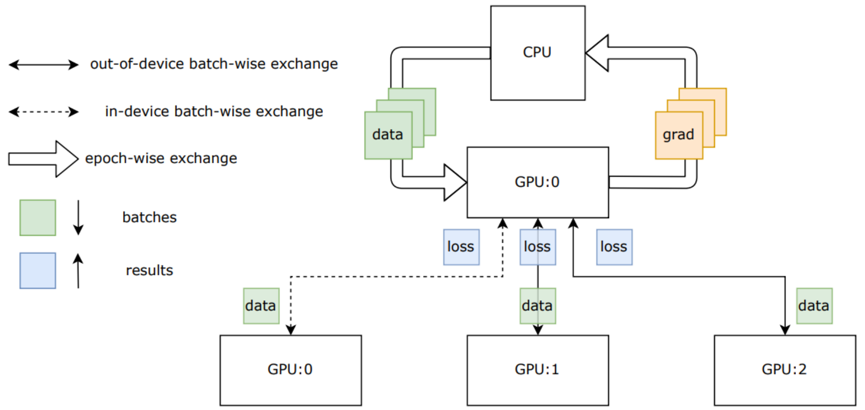


Figure 3: Conceptual architecture of Single Parameter Server parallization approach.

Parallel Type	PrmSrv	GPU0	GPU1	GPU2	GPU3	Total
Baseline	0	85%	0	0	0	0.94
SPS	1	17%	13%	11%	14%	0.55

nn.DistributedDataParallel (DPS)

- Distributed Data Parallel (DPS) helps with the problem.
- No surprise. Think about why it's faster. Does it solve the bottleneck here?
- All-reduce algorithm to synchronize the training process.

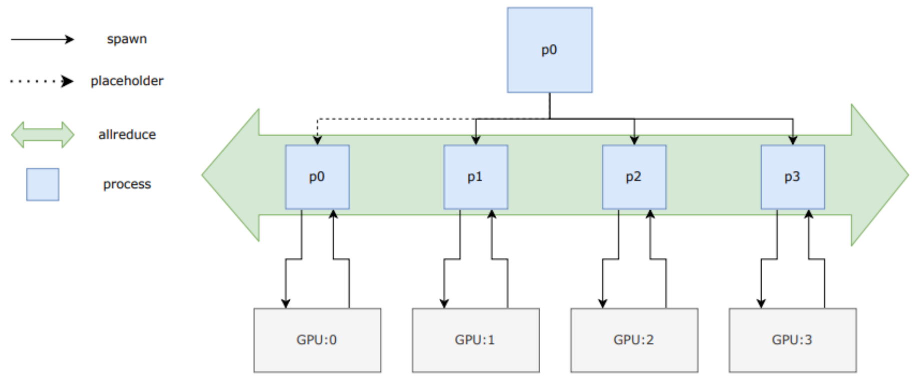


Figure 4: Conceptual architecture of Distributed Parameter Server parallization approach.

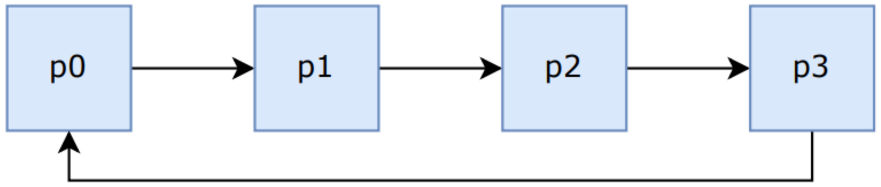
Parallel Type	PrmSrv	GPU0	GPU1	GPU2	GPU3	Total
Baseline	0	85%	0	0	0	0.94
SPS	1	17%	13%	11%	14%	0.55
DPS	4	94%	94%	89%	94%	3.71

Good approches come with price…

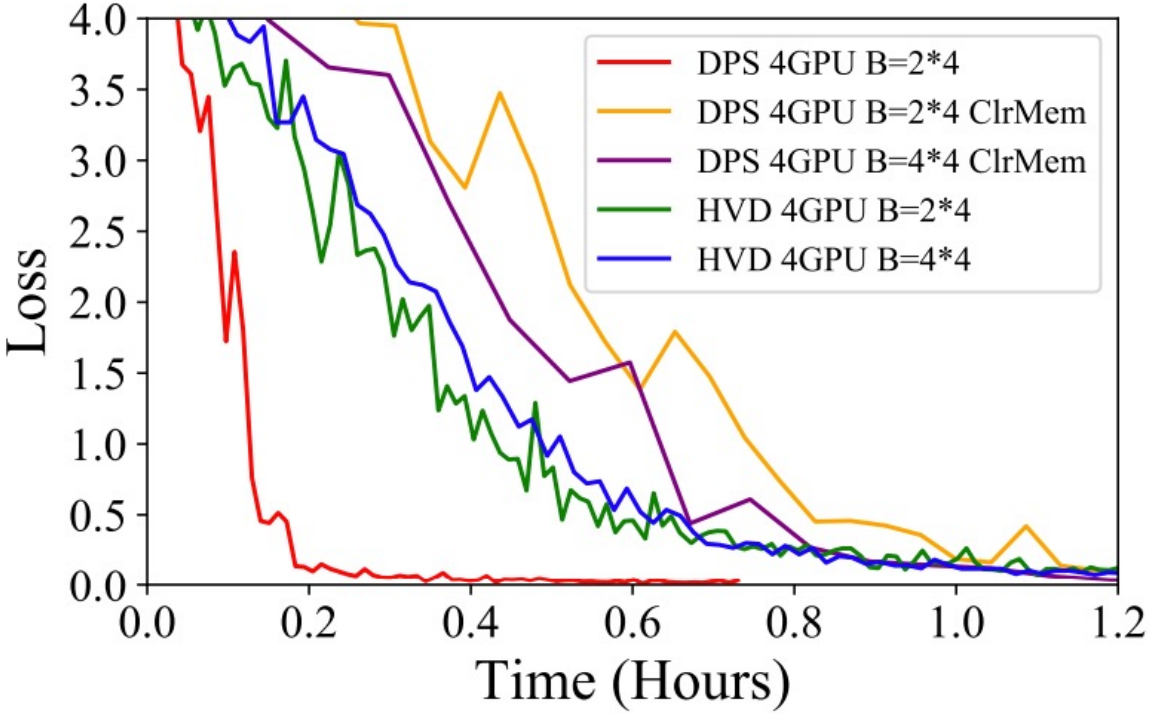
- Good on one node, hard on multiple nodes.
- Because you have to launch the script on every node.
- Any frameworks solving this problem?

Horovod

- Yes! Let's use Horovod, a distributed deep learning framework.
- Install it on all the nodes, so they share the same backend (MPI).
- Launch on one node, then all the nodes can work together.
- Ring-allreduce: optimized synchronization strategy.



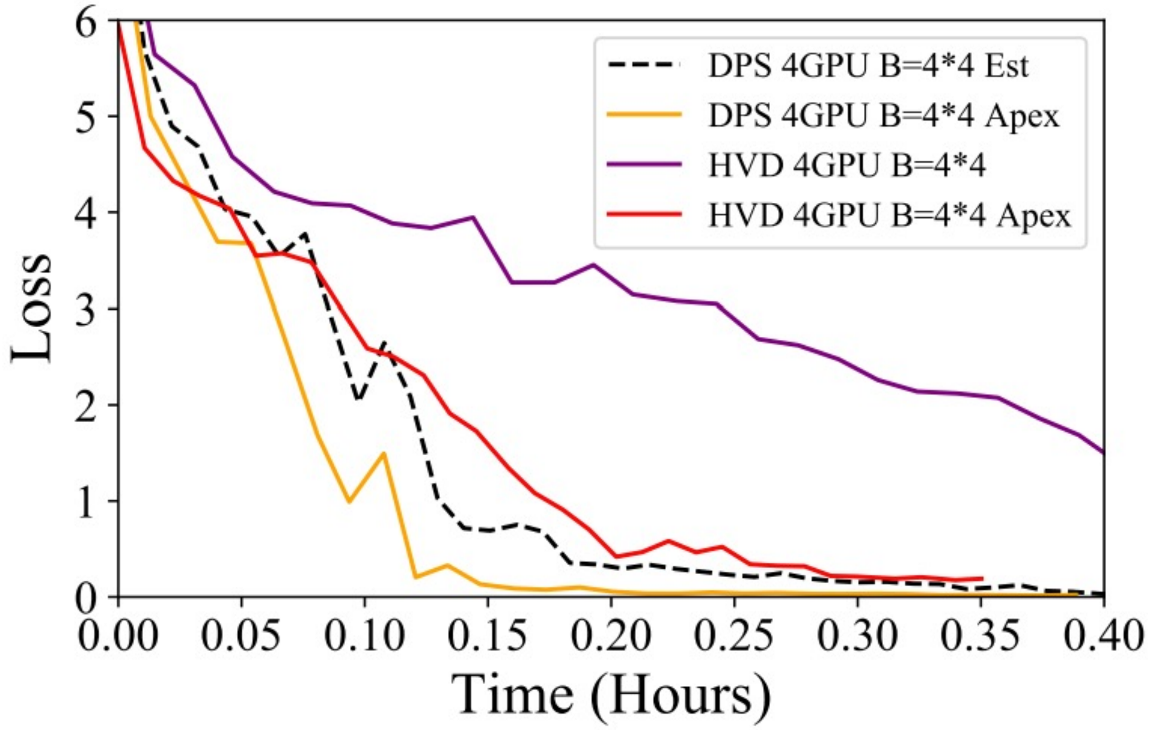
Comparison of DPS and Horovod on one node



(a) Loss curve for DPS (after adding the memory cleaning code) and Horovod, both with respect to a batch size of 8 and 16.

- Look into source code: Horovod is basically a **wrapper** around DPS.
- DPS is faster on one node, but allreduce introduces asymmetricity.
- Horovod is slower on one node, but has built-in cache cleaning techniques and ring-allreduce algorithm eliminates asymmetricity.

Big brother: Apex mixed Precision (fp16+fp32)



(d) Loss curve for DPS and Horovod before and after applying Apex fp16-fp32 mixed training.

- Extreme performance boosting: 2x faster for DPS, 4x faster for Horovod (i.e. nice compatibility with Horovod)
- Half the memory used to store batches, so DPS 4*4 can now run!
- Requires TensorCore to use. Try to pursuade your boss to buy one for you!
- V100+ (V100, A100…)
- Details look at the GPU specs page.

Code? Open-source!

```
n_ctx = model_config.n_ctx
if args.bpe_token:
    full_tokenizer = get_encoder(args.encoder_json, args.vocab_bpe)
else:
    full_tokenizer = tokenization_bert.BertTokenizer(vocab_file=args.t
full_tokenizer.max_len = 999999

######
# construct GPU network
torch.distributed.init_process_group(
    backend='nccl'
)
local_rank = torch.distributed.get_rank()
torch.cuda.set_device(local_rank)
device = torch.device("cuda", local_rank)
######
print('using device:', device)
```

- Feel free to copy and paste.
- Easy to follow! '#####' is used to mark the parallel code added to the initial code.
- https://github.com/BiEchi/DistributedTrainingGPT2

Thank you!

```
+------------------------------------------------------+
| NVIDIA-SMI 352.79     Driver Version: 352.79         |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX TIT...  On   | 0000:04:00.0     Off |                  N/A |
| 23%   62C    P2   224W / 250W |  11769MiB / 12287MiB |     98%      Default |
+-------------------------------+----------------------+----------------------+
|   1  GeForce GTX TIT...  On   | 0000:05:00.0     Off |                  N/A |
| 23%   62C    P2   222W / 250W |  11768MiB / 12287MiB |     99%      Default |
+-------------------------------+----------------------+----------------------+
|   2  GeForce GTX TIT...  On   | 0000:06:00.0     Off |                  N/A |
| 22%   51C    P2   227W / 250W |  11768MiB / 12287MiB |     99%      Default |
+-------------------------------+----------------------+----------------------+
|   3  GeForce GTX TIT...  On   | 0000:07:00.0     Off |                  N/A |
| 29%   69C    P2   214W / 250W |  11768MiB / 12287MiB |     98%      Default |
+-------------------------------+----------------------+----------------------+
|   4  GeForce GTX TIT...  On   | 0000:0A:00.0     Off |                  N/A |
| 22%   60C    P2   228W / 250W |  11768MiB / 12287MiB |     99%      Default |
+-------------------------------+----------------------+----------------------+
|   5  GeForce GTX TIT...  On   | 0000:0B:00.0     Off |                  N/A |
| 23%   63C    P2   232W / 250W |  11768MiB / 12287MiB |     99%      Default |
+-------------------------------+----------------------+----------------------+
|   6  GeForce GTX TIT...  On   | 0000:0C:00.0     Off |                  N/A |
| 22%   56C    P2   222W / 250W |  11769MiB / 12287MiB |     98%      Default |
+-------------------------------+----------------------+----------------------+
|   7  GeForce GTX TIT...  On   | 0000:0D:00.0     Off |                  N/A |
| 22%   57C    P2   238W / 250W |  11768MiB / 12287MiB |     99%      Default |
+-------------------------------+----------------------+----------------------+
```