GPU성능파워업!‘LMS’기술테스트결과는?

딥러닝(Deep learning)이든 전통적인 HPC(High Performance Computer)든 GPU(Graphics Processing Unit)를 사용하시는 모든 분들이 골치 아파하시는 것 중 하나가 바로 ‘GPU 메모리’입니다.

GPU에서 무언가를 연산하기 위해서는 해당 로직과 데이터를 CPU 메모리에서 GPU 메모리로 복사(Copy)부터 해야했습니다. 그런데 보통 12GB~24GB 정도인 작은 GPU 메모리 때문에 원하는 데이터를 다 복사해 올 수도 없고 속도도 느려 성능상 만족스럽지 못했죠.

IBM은 이를 보완하기 위해 획기적인 솔루션, LMS(Large Model Support)을 개발했습니다. 그리고 테스트를 통해 월등히 나아진 성능을 확인할 수 있었습니다.

GPU 메모리의 한계 ‘속도•대역폭•용량’

[출처 : IBM] 속도와 대역폭, latency가 충분히 만족스럽지 않다는 것이 문제

GPU에서 연산을 위한 copy(cudaMemcpyHtoD)는, CPU와 GPU를 연결하는 시스템버스인 PCIe를 통해 이루어집니다. 원래 PCIe는 I/O를 연결하기 위한 버스라서 속도와 대역폭, latency가 충분히 만족스럽지 않다는 것이 문제였죠. 그래서 IBM POWER8, POWER9 프로세서에서는 NVLink(CPU와 GPU를 연결할 때 PCIe 대신 사용하는 시스템 버스 기술)를 위한 전용 port를 별도로 CPU칩에 넣어,

CPU와 GPU가 고속의 NVLink로 통신하게 함으로써 그 문제를 해결했습니다.

그러나 문제는 거기에서 그치지 않았습니다.

GPU 메모리는 일반 호스트 서버 메모리인 DDR4보다 훨씬 빠르고 비싼 HBM2를 사용하는데, 가격과 전력, 그리고 물리적 공간 때문에 무한정 큰 사이즈의 메모리를 장착할 수가 없습니다. 그러다 보니 GPU 메모리의 용량이 충분히 크지 않다는 불만들이 나올 수 밖에 없었죠. 이렇게 제한된 GPU 메모리 사이즈로 인한 문제는 deep learning에서 batch_size를 크게 했더니 CUDA에서 out-of-memory 에러가 나는 사소한 불편으로 그지치 않습니다.

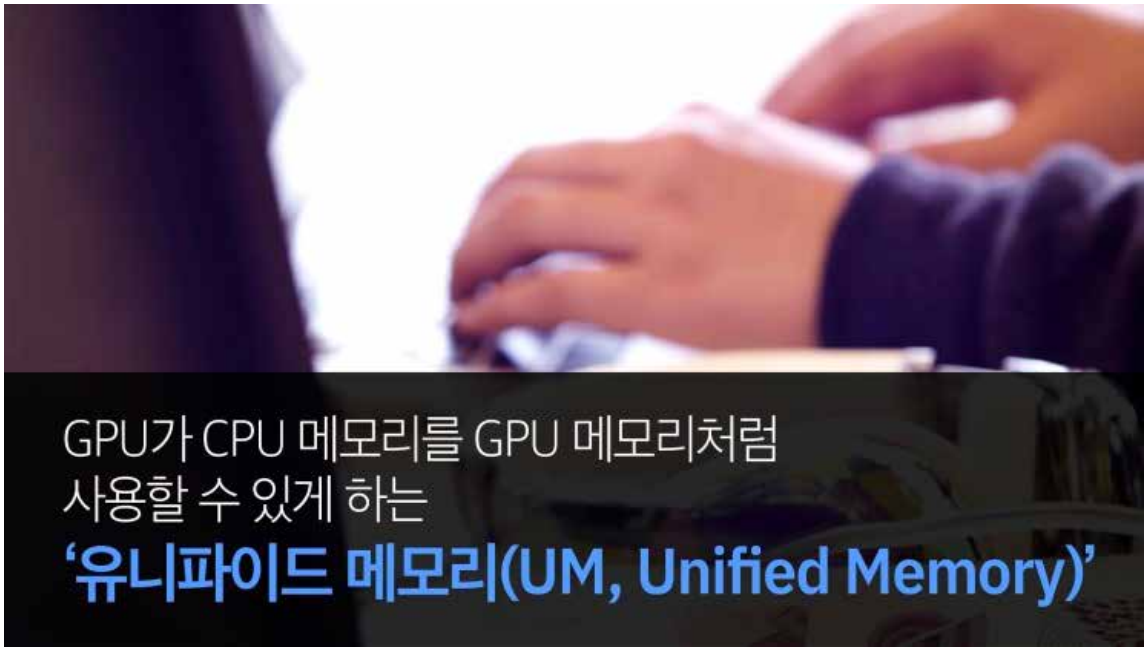
매우 큰 신경망(neural network)을 train 해야하거나

고해상도 이미지 등 크기가 큰 데이터셋을 이용해 train 해야하는 경우

GPU 1장의 메모리 안에 아예 들어가지 않기 때문에 train 자체가 불가능한 경우도 있습니다.

이럴 경우 model parallelism을 구현하여 여러 장의 GPU에서 나누어 train 시키거나 이미지를 크로핑(cropping)하여 작은 사이즈로 만들어 train해야 하는데요. 이는 모두 난이도가 높거나 시간이 많이 걸리는 불편한 작업들입니다. 이런 문제를 한번에 쉽게 해결할 방법이 없을까 하는 것은 모든 GPU 사용자들의 공통된 염원이었죠.

‘유니파이드 메모리(UM)’ 개발... 편리하지만 활용도 낮아



[출처 : IBM] 편리하지만 성능은 떨어지는 유니파이드 메모리(UM)

이 문제에 대한 해결책은 NVIDIA에서 꽤 오래 전에 제시된 바 있습니다. NVIDIA의 쿠다(CUDA, GPU를 이용한 병렬처리 프로그래밍 플랫폼)는 버전 6.5부터 ‘유니파이드 메모리(UM, Unified Memory)’를 선보였습니다. GPU가 CPU 메모리를 GPU 메모리처럼 사용할 수 있게 하는 기능이지요. 그러나 GPU가 CPU 메모리에 접근하는 실제 통로는 양방향 32GB/s에 불과한 PCIe이다 보니, UM을 쓰면 편리하기는 해도 성능은 거의 1/10 수준으로 떨어졌습니다.

이는 NVLink가 달린 V100 SXM2를 장착한 서버에서도 마찬가지였습니다.

Intel의 x86 CPU에는 PCIe port만 달려있으므로 GPU에 NVLink가 달려 있다고 해도 GPU끼리만 NVLink로 연결될 뿐, 정작 CPU와 GPU는 PCIe로 연결해야 했거든요. 그래서 결국 아무도 caffe나 tensorflow 등에서 UM을 쓸 생각을 하지 않았습니다.

GPU를 사용하는 이유가 가속화를 위해서인데, 속도가 1/10로 떨어진다면 GPU를 쓸 이유가 없으니까요.

IBM, 새로운 돌파구 ‘LMS’로 GPU 성능 끌어올린다

[출처 : IBM] IBM의 PowerAI에 포함된 tensorflow, caffe-ibm, 그리고 pytorch에서 사용 가능한 LMS

그래서 IBM에서는 획기적인 솔루션을 내놓았습니다.

IBM POWER9 프로세서에는 NVLink port가 탑재되어 있으므로, CPU와 GPU가 NVLink로 직접 연결되며, 그것도 NVLink 3개를 묶어 150GB/s로 연결됩니다. PCIe의 무려 5배에 가까운 대역폭입니다.

이것을 이용하면 GPU 메모리가 부족해져서 GPU가 호스트 서버의 메모리를 사용할 때 발생하는 성능 저하를 최소화할 수 있을 것이라고 봤는데요.

그래서 만들어낸 솔루션이 바로 IBM PowerAI toolkit에 포함된 LMS, ‘Large Model Support’입니다. 현재 LMS는 IBM의 PowerAI에 포함된 tensorflow, caffe-ibm, 그리고 pytorch에서 사용 가능합니다.

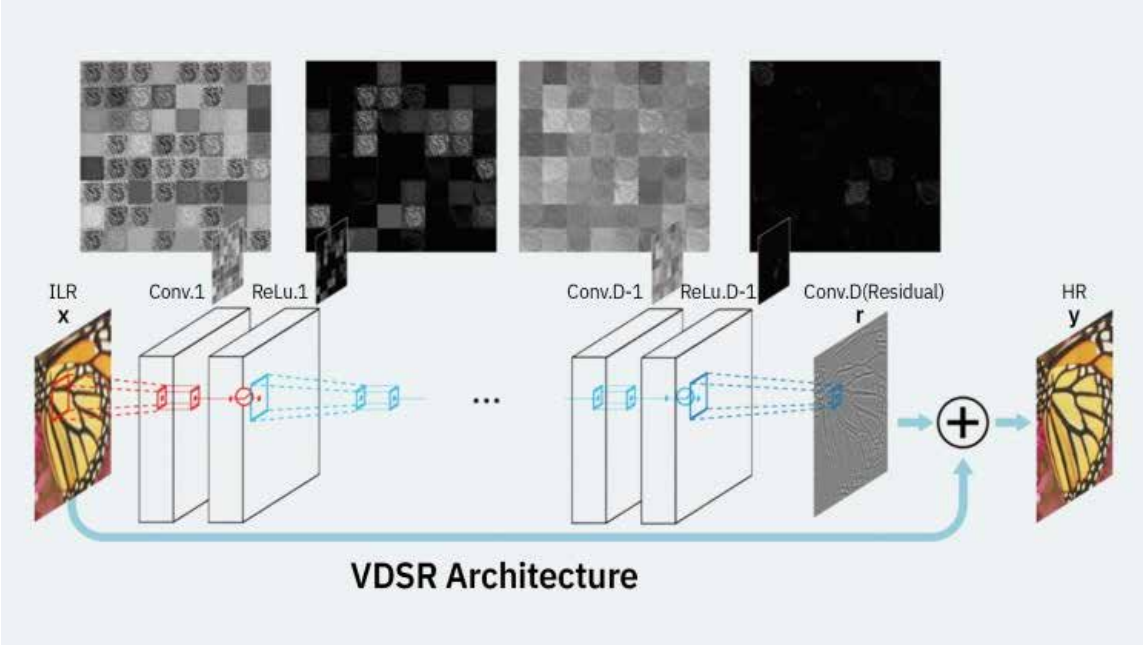
이 LMS 기능이 고해상도 이미지 데이터셋을 이용한 딥러닝 모델에 어떤 혜택을 주는지에 대해서는 2018년 여름 IBM Developer 홈페이지에 올라온 포스팅에 잘 나와 있습니다.

경희대 MLVC Lab, 고해상도 영상 위해 IBM LMS 테스트

위의 포스팅에서 IBM의 딥러닝 엔지니어인 Sam Matzek은 의료용 3D 고해상도 이미지를 그대로 사용하여 기존에는 불가능했던 훨씬 더 높은 정확도의 classification model을 train할 수 있다는 것을 입증해 보였습니다.

그리고 경희대 Machine Learning and Visual Computing Lab (MLVC lab, https://sites.google.com/site/mlvclab/team, 지도교수 배성호)에서 IBM PowerAI의 LMS 기능에 주목하게 되었죠.

MLVC Lab에서는 동영상의 SR(Super Resolution) 처리 및 3차원 물체의 영상 취득 모델링 등을 연구하고 있는데요. 연구 대상의 특성상 고해상도의 영상을 많이 다루다 보니 GPU 메모리 부족을 절감했기 때문입니다.



[출처 : IBM

MLVC Lab은 GPU 메모리 부족 문제를 IBM의 LMS 기술을 이용하여 해결할 수 있는지 평가해 보기 위해, IBM의 AC922 서버를 단기 임대하여 다음과 같이 PoC(Proof of Concept) 테스트를 수행했습니다.

*** 테스트 측정 대상 :**

· LMS 옵션 사용 유무에 따른 VDSR(Very Deep Super Resolution) 모델의 batch_size 증가 가능 여부

· LMS 옵션 사용 유무에 따른 VDSR(Very Deep Super Resolution) 모델의 training 및 validation 속도와 메모리 사용량 비교

*** 테스트 환경 :**

· HW - IBM AC922 서버(POWER9 CPU * 2, RAM 512GB, NVIDIA V100 16GB * 4)

· OS - Ubuntu 16.04 기반의 Python 3.6 docker image를 Redhat 7.5 host OS에서 운용 · CUDA - v10.0

· Framework - PyTorch 1.0 from PowerAI v1.5.4

· SR(Super Resolution) upscale factor - 2로 고정

· Training iteration - 10 epochs

*** Dataset :**

· DF2K - DIV2K + Flickr2K

· Training dataset - 800 + 2060 = 3050 images · Validation dataset - 100 images

· Width pixels : 2040로 고정

· Network input size - 1024x1024

테스트에 사용한 python script의 source code는 GitHub (https://github.com/2KangHo/vdsr_pytorch_lms)에서 자세한 사용법과 함께 보실 수 있습니다. 여기서 제공되는 code에서는 다음과 같이 main.py 수행 시 --lms 옵션을 넣음으로써 간단히 LMS 기능을 on 시킬 수 있습니다.

다만 이 테스트에서는 LMS의 성능 개선을 위해 적용할 수 있는 여러가지 튜닝 기법들은 전혀 적용하지 않았습니다.

```
$ python main.py --dataset DF2K --cuda --gpuids 0 1 --upscale_factor 2 --crop_size 256 --batch_size 128 --test_batch_size 32 --epochs 100 --lms
```

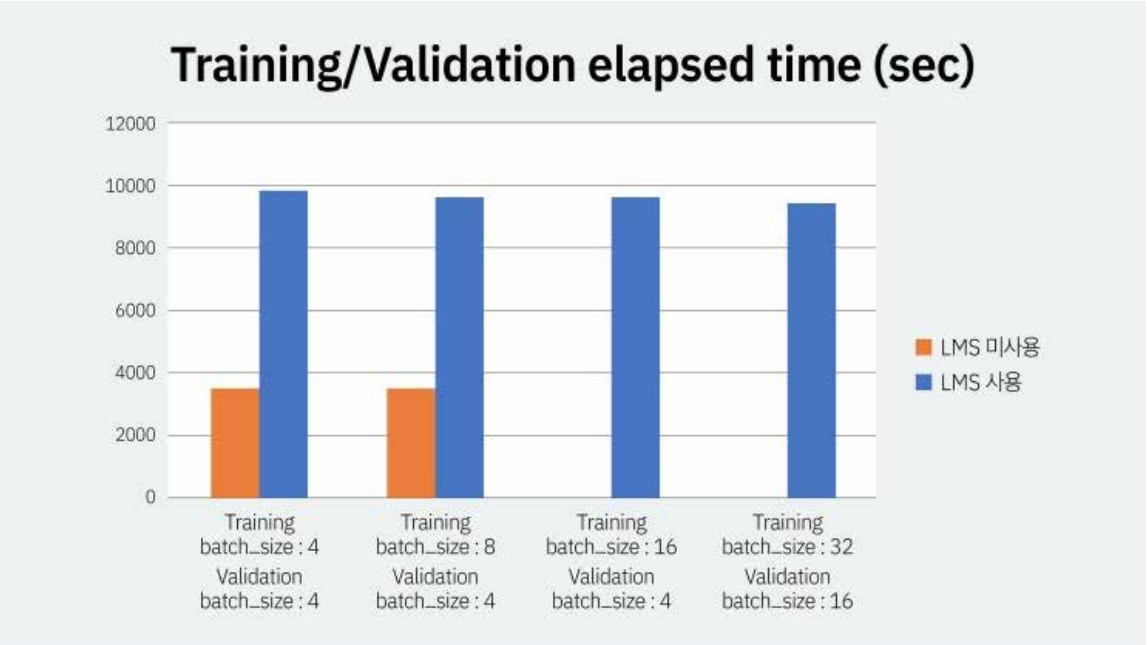
LMS 테스트 결과는? 눈에 띄게 향상된 GPU 성능

위에서 테스트 한 결과는 다음과 같습니다.

LMS 테스트 결과

측정 항목	LMS 사용 여부	Training batch_size : 4 Validation batch_size : 4	Training batch_size : 8 Validation batch_size : 4	Training batch_size : 16 Validation batch_size : 4	Training batch_size : 32 Validation batch_size : 16
수행시간 (sec)	LMS 미사용	3581.36	3529.63	Error 발생	Error 발생
	LMS 사용	**9908.85**	**9701.99**	**9619.66**	**9554.93**
GPU 메모리 사용량 (GB)	LMS 미사용	10.63	11.72	Error 발생	Error 발생
	LMS 사용	**1.73**	**11.72**	**3.97**	**9.77**
Host 메모리 사용량 (GB)	LMS 미사용	89.3	133.03	Error 발생	Error 발생
	LMS 사용	**109.03**	**172.03**	**231.03**	**363.03**

[출처 : IBM] LMS 테스트 결과



[출처 : IBM] Training/Validation elapsed time (sec)

*** 테스트 결과 요약 :**

1 | 고해상도 이미지 데이터셋의 batch_size를 일정 수준 이상으로 늘리는 것이 기존에는 불가능했으나, LMS를 사용할 경우 기존의 4배 이상으로 늘리는 것이 가능하다는 것을 확인했습니다.

2 | GPU 메모리만 이용할 경우에 비해 LMS를 사용할 경우 GPU 메모리(HBM2)보다 더 느린 host 메모리(DDR4)를 사용하므로 training+validation 속도가 약 2.7배 정도 느려지는 것은 피할 수 없으나, batch_size를 더 크게 할 수 있으므로 속도 저감 현상은 어느 정도 보완이 될 수 있음을 확인했습니다.

3 | LMS를 사용할 경우 GPU 메모리 사용량이 크게 줄어들고 대신 host 메모리의 사용량이 기존의 2~3배 증가하는 것을 확인했습니다.

[출처 : IBM]

결론적으로, IBM LMS 기능은 향후 고해상도 영상 처리에 활용 가능성이 크다는 것을 확신할 수 있었습니다.
Host 메모리는 GPU 메모리보다 가격도 싸고 무엇보다 최대 2TB까지 장착이 가능하므로,

LMS 기술은 향후 확장성에 있어서도 전망이 밝을 것으로 예상합니다.

▼ **LMS 기술에 대해 더 궁금한 점이 있다면, 문의 주세요!**

IBM 윤민경 실장 (02-3781-7900 / mkyun@kr.ibm.com)