<주요 Q&A>

GPU Profiling을 통한 Deep Learning 학습 최적화

Q1. cpu thread에 보이는 CUDA block은 커널을 GPU로 전송하는 latency를 의미하는 건가요?

네, main thread에서 cuda api를 호출한 결과를 의미하구요, 실제 GPU에서 실행되는 영역은 CUDA HW쪽에서 확인할 수 있습니다. async하게 동작하므로 해당 부분을 확인하는 것이 중요합니다.

Q2. 프로파일링을 위한 데이터 수집 시 모든 데이터를 참조하는 것인지 아님 목적에 부합된 데이터만 사용하는지요?

필요한 데이터를 수집해서 확인하게 됩니다. 여러 가지 데이터를 활용할 수 있습니다만 주로 특정 코드 블록의 수행시간을 염두에 둡니다.

Q3. GPU 프로파일링시 학습할 때 오히려 프로파일링 코드가 학습 속도를 저하시키지는 않나요?

맞습니다. 프로파일링에 의한 오버헤드는 피할 수 없는 부분이라 이를 최소화 하며 분석하는 것이 중요합니다. nvidia Nsight Systems는 해당 오버헤드를 최소화 해 놓은 형태로 제공하고 있어 사용할 만 합니다. 물론 실제 학습 시에는 프로파일링을 활용하지 않습니다.

Q4. tf에 대한 분석도 보여주시는 예시와 동일한가요?

텐서플로우는 약간 다르게 사용하셔야 하는데요, 그래프가 있다는 점과 실제 실행이 c영역에서 일어난다는 특징 덕분에 같은 nvtx를 사용하지만 다른 인터페이스를 활용해야 합니다. 텐서플로우에서도 nvtx를 사용할 수 있도록 인터페이스를 제공하고 있어 해당 인터페이스를 활용하면 됩니다.

Q5. deep neural net library 종류에 관계없이 같은 방식으로 사용 가능한가요? (pytorch, tf 등등)

nsight systems는 프레임웍에 상관없이 활용할 수 있습니다. nvtx에 대한 인터페이스는 프레임웍별로 특징이 있어 해당 프레임웍에서 제공하는 인터페이스를 활용하시면 됩니다.

Q6. 라이브러리를 임포트 하지 말고 차라리 프래임워크 단에서 프로파이일링 하면 더 효율적인지 않을까요? 그렇게 못하는 제약사항이 있는 것인가요?

어떤 함수 콜들을 수집할 것이냐의 문제인데요 cProfile과 같이 모든 함수 콜을 모두 수집하게 되면 오버헤드가 많이 발생하게 되어 timeline이 왜곡되는 현상이 있을 수 있습니다. 그래서 정보를 최소화해서 남기는 방향으로 활용하고 있습니다.

Q7. GPU profiling을 통해서 deep learning 학습을 최적화 하기 위해서 사전에 준비하고 구비해야 할 사항들에 대해서 질문 드립니다.

target에 gpu profiler를, host에 visualizer를 각각 설치해야 합니다. Ngc docker image에는 gpu profiler가 설치 되어 있으므로 이를 활용하시면 됩니다.

Q8. gpu커널 정보 말곤 다른 정보는 확인 어려운가요?

저희는 monkey patch를 통해 data pipeline같은 정보도 확인하고 있습니다. 분산처리 시 nccl, mpi와 관련된 정보도 얻을 수 있습니다.

Q9. 대부분 io가 async하게 실행이 되던데, nsight 내에서 특정한 텐서의 전송에 관여한 gpu kernel의 op와 cpu 연산을 구분 지어서 확인 가능한가요?

보통 prefetch를 통해서 data를 async하게 처리하므로 해당 정보를 알기가 쉽지는 않습니다. nvtx라는 라이브러리를 통해 원하는 메시지를 남길 수 있으므로 필요한 정보 (예를 들면 파일 이름과 같은)를 남기실 수도 있습니다.

Q10. 프로파일 후에 딥러닝 프레임웍 자체에 부하가 집중되면 프레임웍 자체를 다시 코딩 하나요?

그렇게 할 수도 있습니다. Kernel등을 fusion하는 예가 있을 수 있겠네요. GPU utilization이 꼭 100이 아니더라도 빠르게 학습을 진행하는 것이 좋을 수도 있습니다, 병목이 크지 않다면요.

Q11. 일반적인 CUDA API를 호출 했을 때, Delay Time은 어느 정도 되나요?

이건 일반적이지는 않습니다. api를 호출하면 queue에 해당 operation이 쌓이게 되고, 차례차례 수행되게 됩니다. 프레임웍에서 제공되는 sync함수를 통해 sync를 맞추는 경우에 따라서도 달라지게 됩니다.

Q12. 이미 학습된 사항을 재 학습할 필요가 있을까요?

지금 수행하고 있는 실험은 convergence까지 학습을 하는 것이 아니라 약 10초정도만 학습을 수행하고 GPU를 더 잘 활용할 수 있도록 개선하는 작업을 하고 있습니다. 코드 수정이 모두 완료된 후에는 최종적으로 convergence를 위한 학습을 수행하게 됩니다.

Q13. gpu training time이랑 gpu processing time이 nsight를 통해서 측정이 되는 것인가요? 아니면 별도의 툴을 통해서 측정을 하신 것인가요?

Nsight Systems라는 프로파일링 tool에서 제공하는 결과입니다.

Q14. 딥러닝을 할 때 프로파일링이 필 수 요건인가요?

꼭 그렇지는 않습니다. GPU를 조금 더 잘 활용하기 위해서 할 수 있는 방법에 대한 소개 정도로 봐 주셔도 될 것 같습니다.

Q15. 프로파일링이 gpu만 가능한 것인지, 아님 cpu도 가능한 것인가요?

cpu도 가능합니다. 처음에 보여 드렸던 cProfile이 python에서 활용할 수 있는 가장 간단한 형태의 cpu profiler이고, 이를 활용하실 수 있습니다.

Q16. per-iteration gpu processing time 등을 측정할 때 참고할만한 자료 등을 안내해주실 수 있으실까요?

시간 측정을 위해서는 사실 third party library의 소스코드를 살펴보고 필요한 부분에 monkey-patch를 하는 편이고요, 성능 최적화를 위한 자료는 발표 자료 속에 레퍼런스로 들어있는 pytorch performance guide, cudnn performance guide정도를 시작으로 하시면 좋을 듯 합니다.

Q17. GPU Profiling이 병목현상 완화에도 도움을 줄 수 있을까요?

네, 병목현상 확인을 위한 용도로 주로 활용됩니다.

Q18. 그럼 cpu 와 gpu의 프로파일링시는 cpu mem의 사용률을 동시에 고려해야 할 것 같은데 그것이 효과적인지 아님 gpu난 유틸라이젠이션하는 것이 deep learning 시 더 효과적인지요?

두 가지 정보를 모두 참고하시는 것이 좋습니다.

Q19. stream으로 memcpy를 async하게 하면 kernel이랑 memcpy kernel 사이에 dependency는 어떻게 처리하나요?

pytorch의 경우에는 wait_stream같은 함수를 제공해 어떤 stream이 다른 stream의 작업을 기다리도록 만들 수 있습니다.