KIPS Tr. Comp. and Comm. Sys.
Vol.7, No.4 pp.99~102 pISSN: 2287-5891

https://doi.org/10.3745/KTCCS.2018.7.4.99

Extending Caffe for Machine Learning of Large Neural Networks Distributed on GPUs

Oh Jong-soo† · Lee Dongho††

ABSTRACT

Caffe is a neural net learning software which is widely used in academic researches. The GPU memory capacity is one of the most important aspects of designing neural net architectures. For example, many object detection systems require to use less than 12GB to fit a single GPU. In this paper, we extended Caffe to allow to use more than 12GB GPU memory. To verify the effectiveness of the extended software, we executed some training experiments to determine the learning efficiency of the object detection neural net software using a PC with three GPUs.

Keywords : Parallel Programming, GPU Computing, Machine Learning

대규모 신경회로망 분산 GPU 기계 학습을 위한 Caffe 확장

오 종 수† · 이 동 호††

요 약

Caffe는 학술 연구용으로 널리 사용되는 신경회로망 학습 소프트웨어이다. 신경회로망 구조 결정에서 가장 중요한 요소에 GPU 기억 용량이 포함된다. 예를 들어 많은 객체 검출 소프트웨어는 신경회로망이 12GB 이하의 기억 용량을 사용하게 하여 하나의 GPU에 적합하게 설계되어 있다. 본 논문에서는 큰 신경회로망을 두 개 이상의 GPU에 분산 저장하여 12GB 이상의 기억 용량을 사용할 수 있게 Caffe를 확장하였다. 확장된 소프트웨어를 검증하기 위하여 3개 GPU를 가진 PC에서 최신 객체 검출 소프트웨어의 배치 크기에 따른 학습 효율을 실험하였다.

키워드 : 병렬 처리, GPU 컴퓨팅, 기계 학습

1. 서 론

신경회로망의 크기와 학습 데이터의 양이 증가함에 따라 많은 저자들이 신경회로망 학습의 대규모 분산 컴퓨팅에 관하여 연구하였다[1-3]. 이들 연구는 주로 본질적으로 순차적인 SGD(synchronous stochastic gradient descent, SSGD) 알고리즘을 병렬화하는 방법을 연구하였다. 제안된 방법들은 ASGD(asynchronous SGD) 알고리즘을 사용하여 근사적으로 해답 공간을 탐색하였다. 최근 순차적인 SGD 알고리즘과 동일한 탐색이 가능한 분산 동기 탐색 방법에 대한 관심이 커지고 있다[3]. Caffe[4]는 널리 사용되는 신경회로망 학습 소프트웨어이다. [5]에서는 Caffe를 확장하여 GPU 기능뿐만 아니라 CPU 병렬 처리 기능까지 사용하는 방법을 연구하였다. Caffe를 확장한 Caffe2는 GPU 메모리를 더 효율적으로 사용하고 모바일 환경까지 지원한다[6]. Caffe(Caffe2)는 SSGD로 구현되어 있으며 배치(batch) 크기를 늘여 데이터를 병렬 처리함으로써 학습 속도를 높일 수 있다. Caffe에서는 모든 GPU는 동일한 신경회로망을 저장한다. 본 논문에서는 신경회로망을 수평으로 분할하여 2개 이상의 GPU에 분산 저장하여 하나의 GPU에 저장되지 못하는 큰 신경회로망을 학습할 수 있게 Caffe를 확장하였다.

본 논문은 먼저 Caffe 소프트웨어를 확장하여 신경회로망을 여러 개의 GPU에 분산하여 저장할 수 있는 방법을 소개하고 확장된 Caffe를 최근 소개된 객체 검출 신경회로망 소프트웨어에 적용하여 훈련 진행에 따른 테스트 검출 결과를 보고한다. 실험에서는 많은 객체 검출 소프트웨어 가운데에서 [7]에서 소개된 MSCNN(multi-scale convolutional neural network)을 사용하였다. MSCNN은 비교적 우수한 성능을 가지고 있고 C++만 사용하여 구현되어 있어서 소프트웨어 확장 실험에 편리하였다.

† 비 회 원 : 경북대학교 전자공학부 박사과정
†† 종신회원 : 경북대학교 전자공학부 교수
Manuscript Received : September 13, 2017
Accepted : January 30, 2018
* Corresponding Author : Lee Dongho(dhlee@ee.knu.ac.kr)

2. Caffe 소프트웨어 확장 구현

Caffe에는 여러 개의 GPU를 이용하여 동일한 신경회로망에 서로 다른 입력 배치를 처리하는 병렬 처리 기능이 포함되어 있다. [3]에서는 이를 SSGD라 분류하였다. Caffe를 확장하여 신경회로망을 여러 GPU에 분산 저장할 수 있으면 기존의 SSGD 분산 처리 방법을 유지하면서 큰 영상을 이용한 학습을 가능하게 하거나 배치 크기를 크게 할 수 있다. Caffe는 오픈 소스 소프트웨어로 많은 연구자들이 연구 목적에 맞게 소스 코드를 변경하여 배치 정규화(batch normalization) [8]와 같이 새로운 학습 능력을 가진 뉴런을 첨가하거나 데이터 입력과 오류 손실 함수를 정의하기 위하여 수행 알고리즘의 목적에 적합한 뉴런들을 고안하여 추가한다. Caffe 확장 코드는 이렇게 변경된 소프트웨어에 용이하게 적용될 수 있어야 한다. 먼저 Caffe의 GPU 사용 방법을 간단하게 설명하고 신경회로망을 여러 GPU에 분산 저장하기 위해 필요한 소스 코드 변경에 대하여 기술한다.

Caffe는 뉴런 계층 구현에 CUDA 기반 GPU 라이브러리인 CUBLAS의 벡터, 행렬 연산 기능을 주로 사용하고 필요한 경우 CUDA 커널 함수(kernel function)를 프로그램해서 사용한다. CUDA에서 하나의 CPU 스레드(thread)가 하나의 GPU를 사용하다가 다른 GPU를 사용하기 위해서는 CUDA의 장치 지정 함수(`cudaSetDevice`)를 사용하여 GPU 장치를 변경한다. 모든 GPU 간의 데이터 복사에는 메모리 복사 함수(`cudaMemcpy`)를 사용한다. CUDA의 통합 메모리(unified memory) 기능은 메모리 주소만으로 서로 다른 GPU 간 데이터 복사가 가능하게 한다. 메모리 복사 함수와 달리 모든 연산 함수와 사용자가 작성하는 커널 함수들은 수행 시에 모든 입출력 데이터가 동일한 GPU에 있어야 한다. Caffe의 경우 특별한 경우를 제외하고는 모두 동기식 함수 호출 방법을 사용하므로 하나의 CPU 스레드로 두 개 이상의 GPU를 동시에 작동하게 할 수 없다. 현재 Caffe는 병렬 처리 시 여러 개의 CPU 스레드가 각각 다른 GPU를 사용한다.

기존 Caffe를 최소한으로 변경하여 여러 GPU에 분산된 신경회로망을 학습하기 위해서 복사 계층(copy layer) 뉴런을 새로 정의하였다. 새로 정의된 복사 계층 뉴런은 전후진 처리(forward/backward processing) 수행 중에 GPU 사이의 데이터 복사를 담당한다. 복사 계층 뉴런을 적절하게 사용하면 필요한 입력 데이터가 다른 GPU에서 생성되었더라도 각각의 뉴런 계층 처리 시 이미 현재 사용 장치로 지정된 GPU로 복사되어 있게 할 수 있다. 이로 인하여, 현재 뉴런 계층을 수행하는데 사용된 GPU 라이브러리 함수나 커널 함수의 모든 입출력 계수가 현재 장치로 지정된 GPU의 메모리에 저장되어 있으므로, 기존의 GPU 한 개만 사용하는 코드를 있는 그대로 사용할 수 있다.

Caffe는 신경회로망 기술과 학습 제어 파일을 구글의 프로토콜 버퍼(Protocol Buffers)의 프로토텍스트(ProtoText)로 기술한다. 신경회로망을 여러 GPU에 분산하려면 각각의 계층들을 어느 장치에 저장하는가를 기술하여야 한다. 또한 입력 신경회로망 프로토콜 기술을 변경하여 GPU가 변경되는 장소에 복사 계층 뉴런을 삽입하여야 한다. 이 과정을 자동으로 수행하기 위하여 신경회로망 분산 구조(distribution architecture)는 학습 제어 파일(solver ProtoText file)에 저장되며 그 구조는 Fig. 1의 BNF(Backus-Naur form)로 표시된다.

```
<distrib> ::= <train-distrib> <test-distrib>
<train-distrib> ::= dist { <architect-desc>+ }
<test-distrib> ::= dist { <architect-desc>+ }
<architect-desc> ::= arch { <layer-assign>+ }
<layer-assign> ::= layer_assign {
        name: <name_id>, device: <device_num>}
```

Fig. 1. Distribution Architecture Specification

BNF 기술에서 아키텍처 기술(`<architect-desc>`)은 하나의 신경회로망에 해당되며 신경회로망의 각각의 뉴런 계층은 아키텍처에 기술된 계층별 GPU 번호 목록에 따라 정해진 GPU에 할당된다. GPU 할당을 간단하게 기술하기 위해 모든 계층에 대한 GPU 할당을 각각 기술하지 않고 GPU 할당이 바뀌는 시작 계층의 GPU 할당만 기록하게 하였다. 훈련 분산 구조(`<train-distrib>`)는 SSGD 병렬 처리 시 여러 개의 아키텍처를 지원하며 테스트 분산 구조(`<test-distrib>`)는 여러 개의 테스트 신경회로망을 사용할 수 있게 한다. 따라서 컴퓨터 시스템이 많은 수의 GPU를 가진 경우 다수의 훈련 신경회로망 아키텍처를 이용한 병렬 처리로 대규모 신경회로망에 대한 SSGD 학습도 가능하다.

3. 객체 검출 실험

실험에는 2개의 Nvidia Titan X와 1개의 GeForce GTX 980 Ti를 가진 PC를 사용하였다. 객체 검출 신경회로망 소프트웨어는 MSCNN[7]을 사용하였다. MSCNN은 [9]에서 제안한 방식에 기초한 2단계 객체 검출 신경회로망 소프트웨어이다. [7]에서 저자는 원래 영상 크기인 1280×384뿐만 아니라 1920×576, 2560×768 크기로 영상을 확대하여 사용하였다. 실험에서는 1280×384 크기 영상 입력을 사용했다. KITTI 데이터세트에는 훈련 영상(training image) 7481개와 테스트 영상(test image) 7518개가 있다[10]. 테스트 영상에는 검증 데이터(validation data)가 없어 실험에 적합하지 않다. 실험에서는 [7]에서와 같이 검증 데이터가 있는 훈련 영상을 나누어, 3712개를 훈련 영상으로 사용하고 3769개를 테스트 영상으로 사용하였다. KITTI 데이터는 객체의 높이(height), 가림(occlusion), 잘림(truncation) 등에 따라 목표 객체를 E(easy), M(moderate), H(hard)의 난이도로 나누었다. 3769개의 테스트 영상에 포함된 목표 객체는 모두 26766개이며 E, M, H에 따라 각각 2907개, 7876개, 10963개를 목표 객체로 정한다.

[7]에서는 입력 크기와 디콘볼루션(deconvolution) 사용 여부에 따라 다섯 가지의 신경회로망을 실험하였다. 본 연구에서는 이중 디콘볼루션을 사용한 7s-384-2x 신경회로망 아키텍처의 승용차 검출(car detection) 실험 결과를 기술하였다.

Table 1. Test Coverage Change During Training

Iteration	B=8	B=16	B=32	Iteration	B=8	B=16	B=32	B=16+	B=32+
1K	68.37	77.31	75.02	11K	88.72	89.03	88.85	88.75	88.92
2K	78.29	79.42	78.91	12K	88.71	89.03	88.81	88.90	88.80
3K	82.04	82.04	80.90	13K	88.82	88.95	88.90	88.89	88.85
4K	81.53	86.78	85.36	14K	88.92	89.02	88.86	88.97	88.95
5K	86.63	87.02	86.17	15K	88.85	89.07	88.62	88.92	88.96
6K	86.65	87.61	87.27	16K	89.11	89.27	88.76	88.98	88.96
7K	87.14	88.22	87.60	17K	88.89	89.17	88.71	89.03	89.05
8K	87.10	87.91	88.23	18K	89.02	89.13	88.67	89.04	88.93
9K	87.93	88.22	88.18	19K	88.88	89.13	88.73	88.99	88.95
10K	87.60	88.56	88.61	20K	88.91	89.10	88.83	88.97	89.13

실험에서는 배치 크기 B를 8, 16, 32 세 가지로 정하고 각각의 경우에 객체 검출률이 훈련 진행에 따라 어떻게 변화하는지를 조사하였다. Fig. 2에는 B=16인 경우 실선은 전진 처리(forward processing), 후진 처리(backward processing), 계수 갱신(parameter update) 시 GPU 메모리가 할당되는 과정을 나타내었으며, 점선은 이들을 합하여 각 계층별 총 GPU 메모리 사용량의 증가를 나타내었다. 배치 크기 B=32의 경우는 신경회로망의 크기가 Titan X의 최대 GPU 메모리 사용량인 12GB보다 크므로 3개의 GPU 즉, Titan Xp(6.35GB), 980 Ti(4.49GB), Titan X(8.80GB)에 각각 분산하여 저장하였다. 이들은 엔비디아 신경회로망 라이브러리 cuDNN[11]을 사용한 실험 결과이다.

MSCNN에서는 학습은 두 단계로 이루어져 있다. 첫 번째 단계에서는 신경회로망의 객체 제안(object proposal) 계층들의 계수들을 훈련하고 두 번째 단계에서는 객체의 종류와 세밀한 위치를 결정하도록 첨가된 뉴런 계층들의 계수들을 훈련한다. 실험에서는 두 번째 단계에서 1,000회 훈련할 때마다 훈련된 신경회로망을 출력하여 검증 소프트웨어를 이용하여 M 난이도의 경우 객체 검출률을 Table 1에 기술하였다. MSCNN 객체 검출 방법의 경우 배치 크기 16의 신경회로망의 다소 높은 검출률을 보이는 것을 알 수 있다. 실험적으로 배치 크기를 크게 한 경우와 작게 한 경우를 비교해보면 배치 크기를 크게 하여 학습한 경우 정확도가 나빠지는 것으로 알려져 있다[6, 12]. 16+, 32+ 난은 [12]에서 제안된 방법으로 B=8로 10K회 훈련 후 B=16, 32로 계속 훈련한 결과이다.

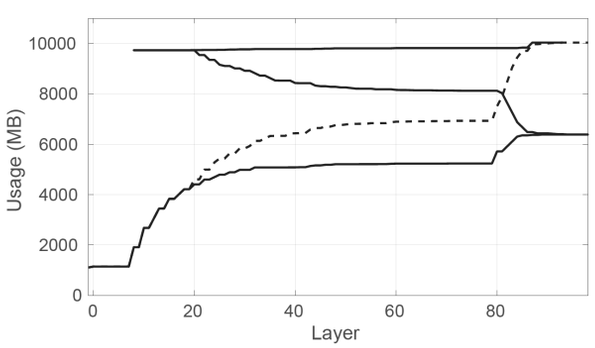

Fig. 2. GPU Memory Usage

4. 결론 및 향후 계획

Caffe는 매우 널리 사용되고 있는 신경회로망 소프트웨어이다. 본 연구에서는 Caffe를 확장하여 다수의 GPU에 신경회로망 데이터 구조를 분산하여 더 큰 신경회로망을 학습할 수 있게 하였다. 최근 배치 정규화가 중요한 신경회로망 학습 기술로 널리 알려지고 있다[8]. 차후 연구로 배치 크기에 따른 배치 정규화의 효율성을 객체 검출 신경회로망 소프트웨어에서 실험할 예정이다[13]. 또한 Caffe는 최근 GP(Gaussian process) 소프트웨어에 포함되어 GP 커널 함수 계수 최적화에도 사용되고 있다[14]. 이 경우에 Caffe는 완전한 배치(full batch) 모드로 사용된다. 이를 위하여 본 연구를 개선하여 배치 크기를 더욱 증가시킬 수 있는 방법을 연구할 예정이다.

References

[1] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large scale distributed deep networks," *Advances in Neural Information Processing Systems 25*, pp.1223-1231, 2012.

[2] F. Niu, B. Retcht, C. Ré, and S. J. Wright. "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent," *Advances in Neural Information Processing Systems 24*, pp.693-701, 2011.

[3] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," *The 4th International Conference on Learning Representations: Workshop Track, arXiv eprint arXiv:1604.00981*, 2016.

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *Proceedings of the 22nd ACM International Conference on Multimedia*, pp.675-678, 2014.

[5] S. Hadjis, F. Abuzaid, C. Zhang, and C. Ré, "Caffe con Troll: Shallow ideas to speed up deep learning," *Proceedings of the 4th Workshop on Data analytics in the Cloud*, pp.2:1-2:4, 2015.

[6] P. Goyal, P. Dollar, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," *arXiv eprint arXiv:1706.02677*, 2017.

[7] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," *Computer Vision - ECCV 2016: Part IV*, Vol.9908 of *Lecture Notes in Computer Science*, pp.354-370, 2016.

[8] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," *arXiv eprint arXiv:1702.03275*, 2017.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems 28*, pp.91-99, 2015.

[10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, Vol.32, Issue 11, pp.1231-1237, 2013.

[11] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cuDNN: Efficient primitives for deep learning," *The NIPS 2014 Deep Learning and Representation Learning Workshop*, *arXiv eprint arXiv:1410.0759*, 2014.

[12] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *The 5th International Conference on Learning Representations: Conference Track*, *arXiv eprint arXiv:1609.04836*, 2017.

[13] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," *Advances in Neural Information Processing Systems 29*, pp.379-387, 2016.

[14] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, Vol.51 of *Proceedings of Machine Learning Research*, pp.370-378, 2016.

오 종 수

https://orcid.org/0000-0001-6708-709X
e-mail : edu@ee.knu.ac.kr
2003년 경북대학교 수학과(학사)
2005년 경북대학교 정보보호학과(석사)
2005년~현 재 경북대학교 전자공학부 박사과정
관심분야 : 병렬 처리, 기계 학습, CAD

이 동 호

https://orcid.org/0000-0002-0106-1337
e-mail : dhlee@ee.knu.ac.kr
1979년 서울대학교 전자공학과(학사)
1981년 한국과학기술원 전산학과(석사)
1992년 Iowa대학교 컴퓨터과학과(박사)
1982년~1992년 ETRI 선임연구원
1992년~1993년 Motorolar Senior CAD Engineer
1993년~현 재 경북대학교 전자공학부 교수
관심분야 : 병렬 처리, 기계 학습, CAD