CUDA와 OpenMP를 이용한 신경망 구현

장홍훈 정기철○
숭실대학교 미디어학부
rollco84@ssu.ac.kr, kcjung@ssu.ac.kr

Implementation of Neural Networks using CUDA and OpenMP

Honghoon Jang Keechul Jung○
School of Media, Department of Information Science, Soongsil University

1. 서론

GPU(Graphic Processing Unit)는 하드웨어 구조의 특성상 동일한 연산의 반복 수행에서 CPU보다 더욱 빠른 수행속도를 보인다. 이런 이유 때문에 반복되는 연산이 많은 영상처리나 패턴인식등의 분야에서 GPU를 이용할 경우 알고리즘을 빠르게 수행할 수 있다. 예를 들어 신경망의 경우 계층간의 반복적인 연산이 발생하기 때문에 GPU로 구현하면 빠르고 효율적인 수행이 가능하다[1]. 뿐만 아니라 최근 GPU의 속도, 가격, 프로그래밍 가능성 등의 다양한 면에서 경쟁력을 가짐으로써 영상처리나 패턴인식 분야 등에서 GPU를 이용한 알고리즘 구현이 활발히 이뤄지고 있다[1-4].

위에서 제안된 방법들은 GPU를 이용하여 CPU에서 보다 빠른 수행시간을 보였지만, GPU를 이용하여 일반적인 프로그램을 구현할 때 고려해야 할 두 가지 문제점에 대해 언급하지 않았다.

첫째, GPU에서 알고리즘을 수행하기 위해선 쉐이딩 언어를 이용하여 작성하여야 하며, 쉐이딩 언어로 알고리즘을 구현하기 위해선 그래픽스적인 개념과 GPU의 구조를 잘 이해해야만 한다.

둘째, CPU와 GPU가 서로 다른 메모리를 사용하기 때문에 용량이 많은 데이터를 처리하기 위해선 서로간의 잦은 데이터 교환이 필요하며, 이는 GPU의 효율적 사용을 저해한다.

본 논문에서는 그래픽 하드웨어 GPU와 멀티코어 CPU를 이용하여 보다 빠르고 효율적인 신경망 구현을 제안한다. GPU의 첫 번째 문제인 복잡한 기존의 쉐이딩 언어 대신 C언어 형식으로 구현이 가능하고 일반 수학적인 연산에 더욱 효율적으로 GPU를 사용하는 CUDA를 이용하였으며, 신경망에서 반복적으로 이용되는 행렬의 곱 연산을 CUDA의 메모리 환경에 적합하고 병렬로 효과적으로 처리할 수 있게 구현하였다. 두 번째 문제인 대용량의 데이터 수집을 위해 소모되는 CPU에서의 연산시간을 최소화하기 위해 공유 메모리 환경에서 프로그램을 병렬화할 수 있는 OpenMP를 이용하여 구현하였다.

2. 본론

신경망이라고 불리는 인공신경망(artificial neural network)은 인간 두뇌의 동작 방식을 따라 구현하였으며, 다양한 모양의 신경망이 사용되고 있지만, 대표적으로 다층 퍼셉트론(multilayer perceptron:MLP), learning vector quantization, radial basis function, Hopfied, kohonen등을 주로 많이 이용한다.

본 논문에서 구현할 신경망은 MLP이다. MLP는 1개 이상의 은닉층을 가지며 다수의 출력 노드를 가질 수 있다. 일반적으로 MLP는 인접한 층의 노드들이 완전 연결되어 있고, 층의 개수나 각 층의 노드 수 등에서 변화가 있을 수 있지만, 기본적으로 각 노드는 식 (1)과 같이 연결가중치벡터와 해당 노드의 입력 벡터의 내적연산을 수행한 후, 식 (2)와 같은 활성함수 연산을 수행한다.

$$p_j = \sum w_{ji} x_i + b_j \quad (1)$$

$$r_j = (1 + e^{-p_j})^{-1} \quad (2)$$

식 (1)과 식 (2)에서 첨자 j는 출력 노드를 의미하며, i는 j-번째 노드와 연결되어 있는 하위층의 노드이다. 식 (1)의 w_{ji}는 j-번째 노드와 i-번째 노드를 연결하는 연결가중치, x_i는 입력값, 식 b_j는 j-번째

노드의 바이어스 항(bias term), 식 (2)의 r_j는 j-번째 노드의 최종 출력값을 의미한다. MLP의 첫 번째 은닉층의 노드들부터 이와 같은 연산을 수행하여 차례로 출력층까지 계산한다. 신경망의 각 노드에서의 내적 연산은 입력벡터와 가중치벡터를 축적함으로써 행렬의 곱 연산으로 변환할 수 있다.

영상처리분야는 이미지 데이터에 대한 반복적인 연산수행을 통해 결과를 얻을 수 있다. 이러한 반복적인 연산은 GPU를 이용하여 효과적인 수행이 가능하기 때문에 신경망기반의 문자추출 프로그램을 통해 실험을 하였다. 입력된 영상의 값과 MxM의 윈도우 마스크와의 연산을 통하여 그레이값을 얻고, 이렇게 분류된 이미지 값은 신경망에 보내어질 데이터로 수집하게 된다. 이 과정에서 멀티코어를 이용한 병렬화 방식을 사용하는 OpenMP를 이용하여 동일한 시간 동안에 더 많은 데이터를 수집할 수 있게 된다. 이렇게 수집된 데이터를 신경망에 전달하게 된다. 신경망에 전달된 데이터는 GPU를 이용하여 수행하게 된다. 우리는 11x11의 윈도우 마스크를 사용하였으며 41개의 은닉층을 사용하여 실험하였다. 이미지의 크기가 작을 경우는 큰 차이를 보이지 않지만 이미지의 크기가 커지면 CPU만을 사용하였을 경우보다 CUDA와 함께 사용한 경우 약 6배의 성능 차이를 보였고, OpenMP와 CUDA를 같이 사용한 경우는 CUDA를 사용한 경우보다 약 4배의 성능향상을 보임을 확인 할 수 있다.

3. 결론

본 논문에선 GPU로 보내어질 대용량의 데이터생성에 이용된 OpenMP와 GPU의 병렬처리를 통하여 보다 빠르고 효율적인 신경망 알고리즘을 제안하였다. GPU사용을 위해 사용된 CUDA는 컴퓨터 그래픽에 대한 상세한 사전지식을 필요로 하지 않는 C언어 스타일의 코드로 작성 할 수 있으며, 그래픽 API함수 없이도 GPU를 이용한 연산이 가능하여 보다 쉽고 효과적으로 GPU연산을 수행할 수 있다. 또한 여러 개의 데이터를 병렬로 처리할 수 있도록 도와주는 OpenMP는 GPU에 전송하기 위한 대용량의 데이터 생성에서 오는 오버헤드를 최소화하였다. 우리는 신경망을 위와 같은 방식으로 구현하였다. 특징 추출에는 CPU의 병렬처리를 이용하여 수행하였고, 주요 신경망의 연산인 내적연산은 GPU를 이용하는 CUDA를 사용하여 수행하였다. 실험결과를 보면 GPU를 사용하는 CUDA만을 사용하는 것보다 CPU의 병렬처리를 동시에 사용하여 신경망을 수행하는 것이 효율적이라는 것을 확인할 수 있다.

참고문헌

[1] K.-S. Kyoung and K. Jung, ¡°GPU Implementation of Neural Network,¡± Pattern Recognition, Vol. 37, Issue 6, pp. 1311-1314, Jun. 2004.
[2] K. Moreland and E. Angel, ¡°The FFT on a GPU,¡± Proceedings of SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware, California, pp. 112-119, 2003.
[3] J. Mairal, R. Keriven, and A. Chariot, ¡°Fast and Efficient Dense Variational Stereo on GPU,¡± Proceedings of International Symposium on 3D Data processing, Visualization, and Transmission, Chapel Hill, pp. 97-104, Jun. 2006.
[4] J. Fung and S. Mann, ¡°OpenVIDIA: Parallel GPU Computer Vision,¡± Proceedings of ACM International Conference on Multimedia, Singapore, pp. 849-852, Nov. 2005.