한국정보전자통신기술학회논문지(jkiiect)'19-12 Vol.12 No.6
논문 19-12-06-591 http://dx.doi.org/10.17661/jkiiect.2019.12.6.591

스트리밍 서버를 이용한 AWS 기반의 딥러닝 플랫폼 구현과 성능 비교 실험

윤필상*, 김도연*, 정구민**

Implementation of AWS-based deep learning platform using streaming server and performance comparison experiment

Pil-Sang Yun*, Do-Yun Kim*, Gu-Min Jeong**

요 약 본 논문에서는 로컬 PC의 성능이 주는 영향이 적은 딥러닝 동작 구조를 구현하였다. 일반적으로, 딥러닝 모델은 많은 연산량을 가지고 있어 처리하는 PC의 성능에 영향을 많이 받는다. 본 논문에서는 이와 같은 제약 사항을 줄이기 위하여 AWS와 스트리밍 서버를 이용하여 딥러닝 동작을 구현하였다. 첫 번째, AWS에서 딥러닝 연산을 하여 로컬 PC의 성능이 떨어지더라도 딥러닝 동작이 정상적으로 작동할 수 있도록 하였다. 하지만 AWS를 통해 연산 시 입력에 대해 출력의 실시간성이 떨어진다. 두 번째, 스트리밍 서버를 이용하여 딥러닝 모델의 실시간성을 증가시킨다. 스트리밍 서버를 사용하지 않았을 경우 한 이미지씩 처리하거나 이미지를 쌓아서 동영상으로 만들어 처리하여야 하기 때문에 실시간성이 떨어진다. 성능 비교 실험을 위한 딥러닝 모델로는 YOLO v3모델을 사용하였고, AWS의 인스턴스들 및 고성능 GPU인 GTX1080을 탑재한 로컬 PC의 성능을 비교하였다. 시뮬레이션 결과 AWS의 인스턴스인 p3 인스턴스를 사용하였을 때 한 이미지 당 테스트 시간이 0.023444초로써 고성능 GPU인 GTX1080을 탑재한 로컬 PC의 한 이미지 당 테스트 시간인 0.027099초와 유사하다는 결과를 얻었다.

Abstract In this paper, we implemented a deep learning operation structure with less influence of local PC performance. In general, the deep learning model has a large amount of computation and is heavily influenced by the performance of the processing PC. In this paper, we implemented deep learning operation using AWS and streaming server to reduce this limitation. First, deep learning operations were performed on AWS so that deep learning operation would work even if the performance of the local PC decreased. However, with AWS, the output is less real-time relative to the input when computed. Second, we use streaming server to increase the real-time of deep learning model. If the streaming server is not used, the real-time performance is poor because the images must be processed one by one or by stacking the images. We used the YOLO v3 model as a deep learning model for performance comparison experiments, and compared the performance of local PCs with instances of AWS and GTX1080, a high-performance GPU. The simulation results show that the test time per image is 0.023444 seconds when using the p3 instance of AWS, which is similar to the test time per image of 0.027099 seconds on a local PC with the high-performance GPU GTX1080.

Key Words : AWS, Cloud Computing Service, Deep Learning, Streaming server, YOLO

This paper is a research conducted by the government (Ministry of Trade, Industry and Energy) in 2019 with support of the core technology development project of the robotics industry (No. 10080615)
*Electronic Engineering, Kookmin University
**Corresponding Author : Electronic Engineering, Kookmin University (gm1004@kookmin.ac.kr)
Received October 07, 2019 Revised October 28, 2019 Accepted November 01, 2019

1. 서론

최근 4차 산업혁명의 등장에 따라 인공지능 기술의 중요도가 올라가면서 에이전트, 자율주행 시스템, 기업용 플랫폼, 산업용 플랫폼 및 개발툴 등 다양한 분야에서 핵심기술들이 확산되며 이들을 조합해 새로운 인공지능을 만들며 인공지능 기술의 발전이 빠르게 이루어지고 있다[1].

다양한 인공지능 기술을 동작하기 위해서는 PC의 성능이 중요하다. 본 논문에서는 이와 같은 제약사항을 줄이기 위하여 딥러닝 AWS라는 클라우드 기반의 딥러닝 플랫폼을 구현하였다.

AWS는 Amazon Web Services의 약어로 아마존에서 제공하는 원격 컴퓨팅 서비스이다[2]. AWS의 서비스는 클라우드 컴퓨팅, 데이터 베이스 등 여러 서비스가 있다. 본 논문에서는 클라우드 컴퓨팅 서비스의 한 종류인 Amazon EC2를 사용하였다. Amazon EC2는 Amazon Elastic Compute Cloud의 약어로 사용자가 가상 컴퓨터를 임대받아 자신의 애플리케이션을 실행할 수 있는 서비스이다. EC2에서는 AMI 인스턴스 유형을 선택할 수 있다. 본 논문에서는 무료 인스턴스와 유료 인스턴스를 사용하여 성능을 비교하였다. 먼저 무료 인스턴스로 Ubuntu Server 16.04 LTS (HVM), SSD Volume Type과 t2.micro 인스턴스 유형을 사용하였고, 유료 인스턴스로 딥러닝과 관련된 프로그램(TensorFlow, PyTorch, CUDA 등)이 설치되어 있는 Deep-Learning AMI (Ubuntu) Version 23.0과 p2, p3 인스턴스를 사용하였다[3].

AWS를 사용할 경우 실시간 성이 떨어진다는 단점이 있다. 따라서 실시간 성을 증가시킬 필요성이 있고 본 논문에서는 이를 스트리밍 서버를 이용하여 해결하였다. 스트리밍 서버는 소켓통신을 통해 로컬 PC로 부터의 입력 데이터를 딥러닝 모델에게 전달하고 딥러닝 모델로 부터의 출력 데이터를 로컬 PC로 전달한다.

2. 관련연구

2.1 Cloud Computing Service

클라우드 컴퓨팅은 인터넷을 통해 서버, 저장소, 애플리케이션 등에 어디서나 접근할 수 있는 모델이다. 이러한 서비스는 초기 구축비용과 유지보수 비용이 저렴하며 사용이 편리하다는 장점이 있다. 또한 사용자가 필요로 하는 양만 사용하고 사용한 만큼의 요금을 지불한다[4].

클라우드 컴퓨팅은 IaaS, PaaS, SaaS, MBaaS라는 4가지 모델이 있다. 그 중 본 논문에서 사용한 Amazon EC2는 IaaS 모델에 속한다.

IaaS란 Infrastructure as a Service의 약어로 물리적 컴퓨팅 자원, 위치, 보안 등과 같은 환경을 제공하는 모델이다.

PaaS란 Platform as a Service의 약어로 응용 프로그램 개발자들에게 개발 환경을 제공하는 모델이다.

SaaS란 Software as a Service의 약어로 공급자가 만든 응용 프로그램에 사용자가 접근하도록 제공하는 모델이다.

MBaaS란 Mobile Backend as a Service의 약어로 백엔드 저장소와 백엔드 애플리케이션에서 사용하는 API를 제공하는 모델이다[5].

2.2 TCP/IP 통신

TCP/IP 통신은 인터넷 프로토콜인 IP와 전송 제어 프로토콜인 TCP로 이루어져 있다. IP는 패킷 전달 여부를 보증하지 않고, 패킷을 보내는 순서와 받는 순서가 다를 수 있다. TCP는 IP 위에서 동작하는 프로토콜로, 데이터의 전달을 보증하고 보낸 순서와 받는 순서가 동일하게 이루어진다[6].

TCP 통신의 동작 시 연결을 위한 3-Way handshake와 4-Way handshake과정이 존재한다. 3-Way handshake는 그림 1과 같이 동작하며 데이터 전송 전 네트워크 연결을 설정하는 과정이다. 4-Way handshake는 그림 2와 같이 동작하며 데이터 전송 후 네트워크 연결을 해제하는 과정이다.

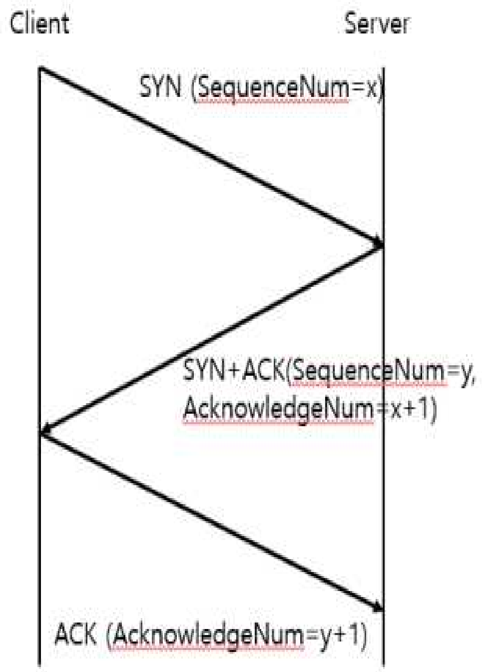

그림 1. TCP 통신 3-Way handshake[7]
Fig 1. TCP connection 3-Way handshake[7]

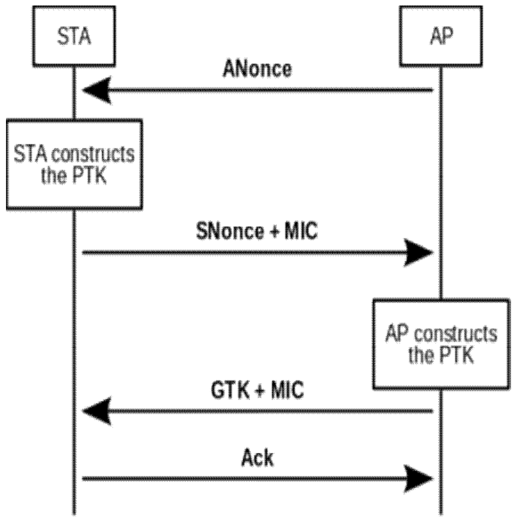

그림 2. TCP 통신 4-Way handshake[8]
Fig 2. TCP connection 4-Way handshake[8]

3. 스트리밍 서버를 이용한 AWS 기반의 딥러닝 플랫폼 제안

본 논문에서는 AWS EC2에 딥러닝 모델을 구축하고 이에 대한 실시간 성을 높이기 위하여 스트리밍 서버를 사용하는 방법을 제안한다. 우선 AWS EC2에 객체인식 딥러닝 모델인 YOLO v3를 구축하고 스트리밍 서버를 이용하여 AWS EC2에 입력 데이터를 전송하고 그에 대한 출력 데이터를 스트리밍 서버로 전송 받는다. 스트리밍 서버는 TCP/IP 기반의 소켓통신을 이용해 구현하였다. 이를 위해 TCP 통신의 연결을 위한 3-Way handshake와 4-Way handshake를 사용하였다. 전체적은 플랫폼 모델의 구조는 그림 3과 같다.

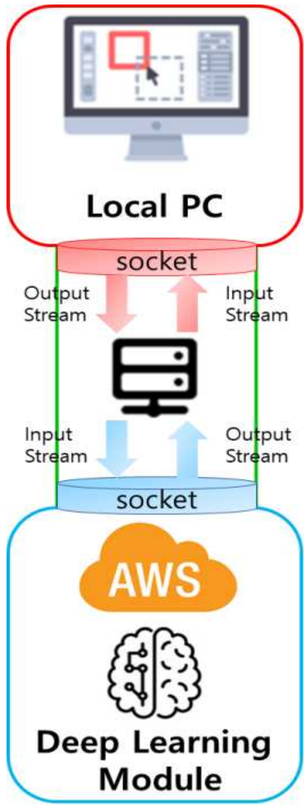

그림 3. 스트리밍 서버를 이용한 AWS 기반의 딥러닝 플랫폼
Fig 3. AWS-based deep learning platform using streaming server

3.1 AWS 기반의 딥러닝 모델 동작

로컬 PC 기반의 딥러닝 모델을 AWS 기반의 딥러닝 모델로 변경하기 위해 우선 인공지능 모듈을 AWS에 설치하였다. AWS에 딥러닝 모델을 설치하기 위한 과정으로 우선 사용할 인스턴스를 선택하고, OS에 대한 가상 이미지를 선택하고 인스턴스를 생성한다. 또한 인스턴스를 사용하기 위하여 key pair를 생성한다. 이렇게 생성한 후에 터미널 프로그램을 사용하여 인스턴스에 접속하고 딥러닝 모델을 설치한다.

3.2 스트리밍 서버를 이용한 실시간 딥러닝 동작

AWS로 구현한 딥러닝 모델의 실시간 성을 증가시키기 위하여 스트리밍 서버를 사용하였다. 스트리밍서버가 로컬 PC의 데이터 값을 AWS의 딥러닝 모델로 실시간으로 전송하여 AWS에서 인공지능 연산을 수행한다. 그리고 결과를 다시 스트리밍 서버를 통해 로컬 PC로 전달받는다. 이와같은 구조를 구현하면 로컬 PC와 AWS의 연산량이 줄어들어 AWS의 부하가 줄어든다. 그로 인해 성능이 낮은 인스턴스를 사용하더라도 비슷한 성능을 나타낼 수 있다.

4. 스트리밍 서버를 이용한 AWS 기반의 딥러닝 플랫폼 성능 비교 실험

4.1 실험 개요

위에서 제안한 AWS 기반의 딥러닝 동작의 성능 확인을 위해 AWS 인스턴스들의 성능을 비교하였다. 성능 비교를 위해 YOLO라는 실시간 개체 검출 프로그램을 사용하였다. 본 논문에서는 YOLO v3를 사용하였다. YOLO v3는 106단의 레이어를 사용하는 개체 검출 인공지능 프로그램이다[9]. 106단의 레이어를 사용하기 때문에 이 프로그램을 사용하기 위해서는 고성능의 PC사양을 요구한다. 이번 실험에서는 동일한 조건에서 성능을 테스트하기 위해 YOLO에서 기본적으로 제공하는 이미지인 그림 4를 사용하였다.

그림 4. YOLO 기본 제공 이미지
Fig 4. YOLO built-in image

4.2 GPU가 없는 인스턴스의 성능 실험

GPU가 없는 경우에는 YOLO v3을 사용할 수 없기 때문에 대신 15단의 레이어를 사용하는 YOLO v3-tiny를 사용하였다. GPU가 없는 인스턴스인 t2.micro 인스턴스에서 YOLO v3-tiny를 실행하였을 때 이미지당 약 1.76초가 걸렸다. GPU가 없기 때문에 YOLO v3보다 성능이 낮은 모델인 YOLO v3-yiny를 사용했음에도 낮은 성능을 나타내었다.

4.3 p2 인스턴스 간 성능 비교 실험

p2 인스턴스는 인텔 제온 E5-2686 v4(Broadwell) 프로세서와 NVIDIA K80 GPU를 사용한다. p2 인스턴스는 여러 유형이 존재하고 이들 사이에는 GPU 개수, GPU 메모리, 가상 CPU 개수, 메모리, 네트워크 성능에 차이가 있다. p2 인스턴스간의 성능 차이를 알아보기 위해 각 인스턴스를 생성하고 YOLO v3을 실행해 보았다. 이 실험은 표 1과 같은 결과가 나왔다.

표 1. p2 인스턴스들의 이미지 테스트 시간
Table 1. Image test time for p2 instances

Instance	Image test time(sec)
p2.xlarge	0.227133
p2.8xlarge	0.247865
p2.16xlarge	0.219650

이미지 테스트 결과 p2 인스턴스간에는 성능 차이가 나지 않는 것을 알 수 있었다. 하지만 표 2와 같이 비용측면에서 차이가 나기 때문에 제약 사항이 없

다면 저렴한 인스턴스를 사용하는 것이 효율적이다.

표 2. p2 인스턴스들의 시간당 가격[10]
Table 2. Price per hour of p2 instance[10]

Instance	Charge($/hrs)
p2.xlarge	1.465
p2.8xlarge	11.72
p2.16xlarge	23.44

4.4 p2, p3 인스턴스 및 GTX1080 탑재 PC 성능 비교 실험

p3 인스턴스는 인텔 제온 E5-2686 v4(Broadwell) 프로세서와 NVIDIA Tesla V100 GPU를 사용한다. 같은 인스턴스 간 성능 비교실험은 4.3에서 이미 실험을 하였기 때문에 이번에는 p2와 p3 및 GTX1080이 탑재된 PC간의 성능을 비교해보았다. 이 실험은 표 3과 같은 결과가 나왔다.

표 3. p2, p3, GTX1080 PC의 이미지 테스트 시간
Table 3. Image test time for p2, p3, GTX1080 PC

Instance	Image test time(sec)
p2.xlarge	0.227133
p3.2xlarge	0.023444
GTX1080 PC	0.027099

실험결과 p3 인스턴스는 p2 인스턴스의 수행시간보다 약 10배 빠른 결과가 나왔고 고성능 GPU인 GTX1080을 탑재한 PC의 수행시간과 비슷하게 나왔다. 이 결과를 통해 고성능이 필요한 딥러닝 모델일 경우에도 본 논문에서 제안한 모델로 구현이 가능하다는 것을 보였다.

5. 결론

본 논문에서는 AWS 기반의 딥러닝 동작 구조를 구현하였다. 딥러닝 동작을 AWS와 같은 클라우드 기반으로 동작시킬 경우 딥러닝 모델이 사용자의 PC 성능에 영향을 적게 받는다는 장점이 있다. 또한 스트리밍 서버를 이용하여 실시간 성이 증가되었다. 그리고 인스턴스 및 고성능 GPU를 탑재한 PC와의 성능 비교 실험을 통해 클라우드 상에서도 좋은 성능을 낼 수 있다는 것을 확인할 수 있었다. 이 실험을 통하여 AWS에서 로컬 PC를 대신해 인공지능 연산을 수행할 수 있으며, 원하는 성능에 따라 인스턴스를 선택하면 효율적임을 확인하였다.

향후 스트리밍서버에서 단순한 데이터의 전달이 아닌 서버 안에서 처리할 수 있는 연산을 하게 된다면 AWS의 부하가 줄어듦에 따라 조금 더 저렴한 인스턴스를 사용하여 효율적인 모델을 구현할 수 있다고 예상되며, 이에 대한 실험을 진행 중에 있다.

본 논문에서 구현한 AWS 기반의 딥러닝 동작 구조는 향후 다양한 사용자가 사용하며 PC의 성능이 중요한 플랫폼에서 적용 가능할 것으로 생각된다.

REFERENCES

[1] Junho Na, “The future of AI and employment”, 14-17(4 pages), FUTURE HORIZON, 2016.5
[2] Amazon Web Services https://en.wikipedia.org/wiki/Amazon_Web_Services
[3] Amazon EC2 Instance Types http//aws.amazon.com/ec2/instance-types/
[4] Brian Hayes, “Cloud computing” Communication of the ACM, 7, 9-11, 2008 DOI: 10.1145/1364782.1364786
[5] Cloud Computing, https://en.wikipedia.org/wiki/Cloud_computing
[6] Internet protocol suite https://en.wikipedia.org/wiki/Internet_protocol_suite
[7] 3-Way handshake https://en.wikipedia.org/wiki/Handshaking
[8] 4-Wa handshake https://en.wikipedia.org/wiki/IEEE_802.11i-2004#Four-way_handshake
[9] Joseph Redmon, Ali Farhadi, “YOLO v3: An Incremental Improvement.”, Apr, 2018, ArXiv:1804.02767v1
[10] Amazon EC2 pricing https://aws.amazon.com/ec2/pricing/

저자약력

윤 필 상(Pil-Sang Yun) [일반회원]

- 현재 국민대학교 전자공학과 석사 과정

〈관심분야〉 차량용 마이컴, 차량용 소프트웨어, 인공지능, 딥러닝, 자율주행

김 도 연(Do-Yun Kim) [일반회원]

- 현재 국민대학교 전자공학과 석사 과정

〈관심분야〉 차량용 소프트웨어, 인공지능, 딥러닝, 영상처리, 패턴인식

정 구 민(Gu-Min Jeong) [일반회원]

- 1995년 서울대학교, 제어계측공학과 학사
- 1997년 서울대학교, 제어계측공학과 석사
- 2001년 서울대학교, 전기컴퓨터공학부 박사
- 2001년~2004년 ㈜ 네오엠텔 기반 기술팀, 팀장(co-founder)
- 2005년~현재 국민대학교 전자공학부 교수
- 2013년~현재 ㈜ 유비벨록스 사외이사
- 2015년~현재 국가기술표준원 자동차전기전자 및 통신 전문위원회 위원장
- 2019년~현재 ㈜ 휴맥스 사외이사

〈관심분야〉 차량용 마이컴, 차량용 소프트웨어, 커넥티드카, 자율주행