TensorFlow Serving 서비스를 지원하는 고성능 GPU 기반 컨테이너 클라우드 시스템

장경수, 김중환
㈜케이아이엔엑스
e-mail : {blueface, jhkim}@kinx.net

A Study on High Performance GPU based Container Cloud System supporting TensorFlow Serving Deployment Service

Kyung-Soo Jang, Jung-Hwan Kim
Cloud Technical Group, KINX Inc.

요 약

TensorFlow와 알파고의 등장으로 인공지능의 높은 성능과 다양한 활용 가능성을 보이면서, 폭넓은 산업 분야에서 머신러닝 기술에 대한 수요가 증가하고 있다. 반면, 머신러닝 기술은 GPU 기반 고속 병렬처리 기술과 인프라 기술을 기반으로 하고 있기 때문에, 머신러닝 기반 서비스 개발 및 제공에 어려움을 겪고 있다. 본 논문에서는 이와 같은 문제를 개선하기 위해서 개발한 고성능 GPU 기반 컨테이너 클라우드 시스템을 소개한다. 해당 시스템은 GPU 기반 고속 병렬처리를 지원하고, Kubernetes 클러스터에서 컨테이너를 기반으로 TensorFlow Serving을 손쉽게 배포할 수 있는 기능을 제공한다.

1. 서론

2007년 NVIDIA의 CUDA 플랫폼을 필두로 고속 병렬처리에 대한 연구개발이 대중화되었고, 2015년 구글 브레인팀에서 공개한 TensorFlow와 2016년 알파고의 등장으로 인공지능의 강력함과 다양한 어플리케이션에서의 활용 가능성을 보여줌으로써 폭발적인 주목을 받기 시작했다. 이와 맞물려 자율주행, 광고, 통신, 음성인식, 의료, 금융 등의 다양한 분야에서 고속 병렬처리와 인공지능에 대한 수요가 증가하면서 산업분야의 핵심기술로 자리를 잡아가고 있다 [1, 2, 3].

반면, 인공지능을 기반으로 한 다양한 서비스를 실현하기 위해서는 GPU 기반 고속 병렬처리와 대규모 인프라 기술이 요소기술로 요구되며, 인공지능 개발자와 인프라 엔지니어와의 간극이 커지고 있어 인공지능 서비스 런칭에 많은 어려움을 겪고 있다.

본 논문에서는 이와 같은 문제를 효과적으로 해결하기 위해서 개발한 고성능 GPU를 제공하는 컨테이너 기반 클라우드 시스템을 소개한다. 개발한 시스템은 클라우드 관리 플랫폼 오픈소스 소프트웨어인 OpenStack을 기반으로 개발되었으며, TensorFlow로 학습된 모델만 준비되면 해당 모델에 대한 TensorFlow Serving을 GPU 기반 컨테이너로 손쉽게 런칭할 수 있도록 API 및 웹 인터페이스를 제공한다.

이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 '범부처 Giga KOREA 사업'의 지원을 받아 수행된 연구임 (No.GK17P0100, Giga Media 기반 Tele-Experience 서비스 SW플랫폼 기술 개발)

2. 관련 연구

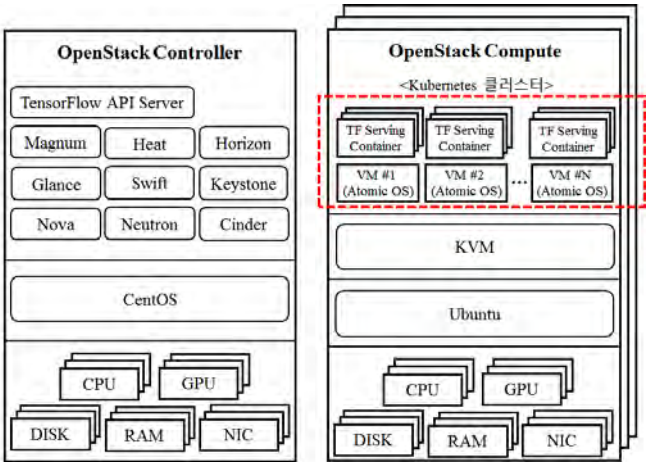
고성능 GPU가 장착된 가상머신을 지원하는 클라우드 서비스로서 대표적으로 Amazon EC2, Google Cloud Platform, Microsoft Azure 등이 있다. 기술과 요금체계는 상이하지만, 대부분 TensorFlow 환경이 구축되어 있고, 고성능 GPU가 할당된 가상머신, TensorFlow와 같은 머신러닝 및 인공지능 플랫폼 또는 라이브러리를 제공한다 [4, 5, 6]. 이와 같은 방식은 TensorFlow 개발환경을 용이하게 구축할 수 있지만, 이후에 발생하는 TensorFlow 기반 서비스의 개발/실행/운영은 여전히 클라우드 사용자의 부담으로 돌아온다는 문제가 있다.

또한 TensorFlow와 같은 특정 어플리케이션을 쉽게 구동할 수 있는 서비스와 인터페이스를 제공하기 위해서는 클라우드 사업자들에게 서비스 개발을 위한 비용 손실이 발생하며, 수익률 증감에 영향을 주기 때문에 다양한 클라우드 서비스 개발을 파격적으로 진행하기가 쉽지 않은 상황이다.

3. TensorFlow Serving 서비스를 지원하는 고성능 GPU 기반 컨테이너 클라우드 시스템 구조

본 논문에서는 학습된 TensorFlow 모델 파일만 가지고 있으면 OpenStack 사용자 웹 대시보드를

통해서 고성능 GPU 기반 클라우드 환경에서 해당 모델에 대한 TensorFlow Serving 클러스터를 컨테이너 형태로 배포할 수 있는 클라우드 시스템을 제안한다.



(그림 1) TensorFlow Serving 서비스를 지원하는 컨테이너 클라우드 시스템 구조도

개발된 시스템 구조는 (그림 1)과 같이 클라우드 오픈소스 소프트웨어인 OpenStack을 기반으로 구축되어 있으며, OpenStack Fuel 이미지를 이용해 기본적인 OpenStack Controller 노드 및 Compute 노드에 대해서 프로비저닝 및 디플로이를 진행하여 기본적인 OpenStack 환경을 구성한다 [7]. 시스템 구축 및 개발에 사용된 소프트웨어 상세 정보는 <표 1>과 같다.

<표 1> 시스템 개발에 활용된 소프트웨어 정보

Software	Version	Purpose
Fuel	10.0	OpenStack Deployment
OpenStack	Newton	Cloud Service
Magnum	stable/newton	Container Orchestration Manager
Django	1.8.14	TensorFlow API Server, Web Server
RPM-OSTree	v2017.7	Atomic OS Package Management Tool
Ansible	2.3.0.0 이상	Atomic OS Customizing Tool
Build-Atomic-Host	-	Ansible Playbook to Custom Atomic OS
Kubernetes	1.5.3	Container Orchestration Engine
Atomic OS	Fedora 25	VM OS Image for Container Cluster
TensorFlow Serving	0.5.0	TensorFlow Running Service + Web Interface

일반적인 OpenStack 기반 클라우드 시스템과 다르게 컨테이너 관련 인프라 관리와 오케스트레이션을 제공하기 위해서 OpenStack Magnum을 사용하여 시스템을 구축했다. Magnum은 템플릿을 기반으로 가상머신 또는 베어메탈 머신 환경에서의 컨테이너 클러스터를 생성해 준다. 또한 컨테이너 클러스터를 관리하는 오케스트레이션 툴로 Kubernetes, Swarm, Mesos를 지원하고 있으며, 본 클라우드 시스템에서는 Kubernetes를 이용하여 TensorFlow Serving 서비스를 제공하고 있다 [8].

또한 본 클라우드 시스템에서는 Tesla P100 GPU를 제공하고 있으며, 컨테이너 환경에서 GPU 사용이 가장 용이한 Kubernetes 환경을 집중적으로 개발하였다. Magnum에서 Kubernetes 클러스터의 가상머신의 이미지를 Atomic OS 또는 CoreOS를 사용할 수 있지만, 다음과 같은 이유로 Atomic OS를 선정하여 커스터마이징 하였다 [8]. GPU를 사용하는 컨테이너 생성이 가능하기 위해서는 1.5 버전 이상의 Kubernetes를 사용해야 한다. 따라서 Atomic OS로 컨테이너 클러스터를 생성하였다. 또한 NVIDIA 드라이버 및 CUDA 라이브러리를 Atomic OS에 추가하기 위해서, Atomic OS를 커스터마이징하는 과정을 진행하였다.

TensorFlow API 서버는 TensorFlow 모델 파일을 관리하고, Magnum에 의해서 생성된 컨테이너 클러스터 환경에서 업로드된 모델 파일을 이용한다. TensorFlow Serving 컨테이너를 생성 및 관리하는 기능은 Rest API로 제공한다. 또한, OpenStack 사용자 대시보드(Horizon)에 TensorFlow API와 연동되는 패널을 추가하였다.

따라서 본 논문에서 개발된 시스템에서 사용자는 웹 브라우저를 통해서 TensorFlow Serving을 GPU 기반 컨테이너로 손쉽게 생성할 수 있다.

4. TensorFlow Serving 서비스 배포 절차

본 클라우드 시스템에서 TensorFlow Serving을 배포하기 위해서는 다음과 같은 과정을 통해서 진행된다.

- TensorFlow 모델 학습
- 학습된 모델 파일 준비
- 웹 대시보드를 이용한 모델 파일 업로드
- Kubernetes 클러스터 생성
- TensorFlow Serving 컨테이너 생성

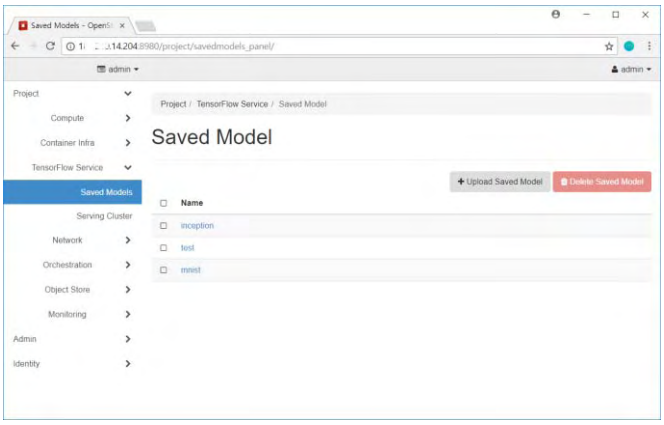
TensorFlow 모델 학습 과정은 요구사항에 부합하는 신경망을 설계 및 개발한 후 방대한 데이터를 이용해 모델을 학습시키게 된다. TensorFlow의 동작과정은 (그림 2)와 같다.

(그림 2) TensorFlow에서의 모델 학습 과정 개요

학습 과정을 통해서 생성된 모델 파일은 TensorFlow Serving에서 신경망 연산을 수행하는데 사용된다 [9]. TensorFlow를 이용한 모델 개발과 학습은 TensorFlow가 설치 가능한 모든 환경에서 가능하지만, 빠른 시간 내에 모델을 학습시키기

위해서 기업용 고성능 서버나 글로벌 클라우드 사업자의 고성능 가상머신을 일반적으로 사용하게 된다. 지속적으로 발생하는 데이터를 이용해서 모델을 개선할 수 있으며, 시간이 지남에 따라 개선된 모델을 파일로 저장할 수 있으며 이를 SavedModel 이라고 한다. 이와 같이 학습된 SavedModel 파일을 준비하는 과정이 필요하다.

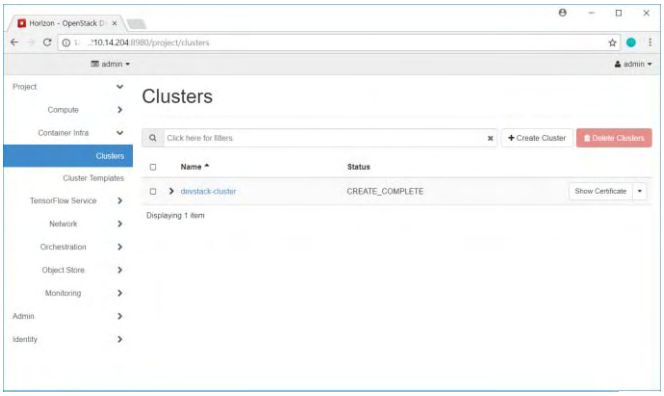
학습된 모델 파일이 준비되면, (그림 3)처럼 개발된 웹 대시보드에 접근하여 SavedModels 패널을 통해서 모델 파일을 업로드 한다.



(그림 3) 모델 파일을 업로드 하기 위한 웹 대시보드

대시 보드에서는 동일한 모델에 대해서 서로 다른 버전의 모델 파일을 업로드 하는 것이 가능하고, TensorFlow Serving에서 신경망 연산에 사용할 모델의 버전은 TensorFlow Serving에서 설정된 정책에 의해서 선택된다.

그 다음 단계로, TensorFlow Serving 컨테이너를 생성할 Kubernetes 클러스터를 생성한다. (그림 4)와 같은 웹 대시보드에 접근하여 템플릿 기반으로 Kubernetes 클러스터를 생성할 수 있다.



(그림 4) Kubernetes 클러스터를 생성하기 위한 웹 대시보드

Kubernetes 클러스터 생성이 완료되면, TensorFlow Serving 컨테이너를 생성하기 위해 (그림 4)의 Serving Cluster 패널에 접근한다. 이 과정에서 TensorFlow Serving 컨테이너를 실행할 Kubernetes 클러스터, TensorFlow Serving 컨테이너에서 사용할 신경망 모델명, GPU 사용 유무를 지정하게 되고, 이 과정을 마치면 TensorFlow Serving을 실행하는 컨테이너가 생성된다.

이와 같은 과정을 통해서 학습된 모델 파일과 웹 인터페이스 조작만으로 간편하게 TensorFlow Serving을 배포할 수 있게 된다.

5. 결론 및 향후 연구방향

본 논문에서는 고성능 GPU를 제공하고 컨테이너를 기반으로 TensorFlow Serving 배포 서비스를 제공하는 클라우드 시스템을 구축 및 개발하였다.

개발된 클라우드 시스템을 통해서 TensorFlow Serving에 대한 빠른 배포뿐만 아니라, 배포 시간 단축으로 인한 비용 절감 효과도 얻을 수 있다. 또한 Tesla P100과 같은 고성능 GPU를 지원하기 때문에 빠른 응답시간을 제공함으로써 사용자 서비스 품질을 향상시킬 수 있을 것이라 기대된다.

향후 고비용의 GPU 리소스를 효율적으로 사용하기 위한 GPU 스케줄링 기법, TensorFlow Serving 컨테이너에 대한 로드밸런서 지원 방안, GPU 리소스 사용에 대한 과금 방법 등에 대해서 연구개발 및 보완하여 완성도를 높일 예정이다

참고문헌

[1] Wikipedia, “CUDA.” Accessed September 5, 2017. https://en.wikipedia.org/wiki/CUDA.

[2] Zak Stone, Product Manager, “Celebrating TensorFlow’s First Year.” Accessed September 5, 2017. https://research.googleblog.com/2016/11/celebrating-tensorflows-first-year.html.

[3] Demis Hassabis, “What we learned in Seoul with AlphaGo.” Accessed September 5, 2017. https://www.blog.google/topics/machine-learning/what-we-learned-in-seoul-with-alphago/.

[4] Amazon,” The AWS Deep Learning AMI Details.”, Accessed September 5, 2017. https://aws.amazon.com/ko/amazon-ai/amis/details/.

[5] Google, “CLOUD MACHINE LEARNING ENGINE.” Accessed September 5, 2017. https://cloud.google.com/ml-engine/.

[6] Microsoft, “Azure GPU Tensorflow Step-by-Step Setup.” Accessed September 5, 2017. https://blogs.msdn.microsoft.com/uk_faculty_connection/2017/03/27/azure-gpu-tensorflow-step-by-step-setup/.

[7] OpenStack, “Fuel.” Accessed September 6, 2017. https://wiki.openstack.org/wiki/Fuel.

[8] OpenStack, “Magnum User Guide.” Accessed September 6, 2017. https://docs.openstack.org/magnum/latest/user/index.html#overview.

[9] TensorFlow™, “TensorFlow Serving.” Accessed September 6, 2017. https://www.tensorflow.org/serving/.