MLOps를 위한 클라우드 환경에서의 데이터파이프라인 구축 및 ETL 작업 자동화 구축

조인화
경희대학교 컴퓨터공학과
inhwa1025@khu.ac.kr

Data Pipelines and Automating ETL Operations in Cloud Computing for MLOps

Inhwa Jo

요 약

고성능 GPU 의 등장과 다량의 데이터 학습 등으로 인공지능의 성능이 폭발적으로 높아짐에 따라 다양한 분야에 머신 러닝이 적용되고 있다. 또한 클라우드 컴퓨팅은 몇 년 전 인공지능 등과 함께 큰 주목을 받으며 등장한 이후, 최근 국내외에서 4 차 산업혁명의 핵심기술로 급부상하고 있다. 따라서 본 문서는 머신 러닝 개발 환경을 위한 클라우드 환경에서의 데이터파이프라인 및 ETL 작업 자동화 구축을 제안하고 구현한다.

1. 서 론

머신 러닝이란 인간과 같은 학습 능력을 기계를 통해 구현하는 방법이다. 머신 러닝은 다양한 알고리즘 기법을 적용하는 여러 유형의 머신 러닝 모델로 구성된다. 머신 러닝의 알고리즘은 대규모 데이터 셋에서 패턴과 상관관계를 찾고 분석을 토대로 최적의 의사결정과 예측을 수행하도록 훈련된다. 머신 러닝 애플리케이션은 적용을 통해 개선되며 이용 가능한 데이터가 증가할수록 더욱 정확해 진다.

따라서 머신 러닝에서의 데이터 수집은 기계 학습 모델을 구축하는 데 매우 중요한 역할을 한다. 데이터 수집은 모델의 성능을 결정하는 중요한 요소 중 하나이기 때문이다. 빅데이터의 출현으로 인해 머신 러닝의 잠재력을 최대치로 끌어내는 것에 대한 실현 가능성이 높아지고 있는 가운데, 빅데이터를 처리하기 위해 병렬처리 기법 등을 이용한 접근 방법이 활용되고 있다.

최근 머신 러닝, 인공 지능, 데이터 분석을 위한 클라우드 기반 툴이 증가하고 있다. 클라우드 컴퓨팅은 몇 년 전 빅데이터 등과 함께 큰 주목을 받으며 등장한 이후, 최근 국내외에서 4 차 산업혁명의 핵심기술로 급부상하고 있다. 세계 클라우드 시장은 현재도 성장 중이며, 시장조사기관 마다 규모의 차이는 다소 있으나 공통적으로 높은 성장률을 예측한다.

클라우드 기술을 사용한다면 어느 컴퓨터에서나 중앙 저장소에 로그인할 수 있으며 어디에서나 작업을 할 수 있다. 클라우드에서는 백업과 동기화를 작업으로 처리해 주기 때문에 모든 작업이 간소화된다. 또한 클라우드는 사용한 만큼 비용을 지불하기 때문에 컴퓨팅 자원에 대한 시간과 비용을 절감할 수 있다.

이에 본 연구에서는 클라우드 컴퓨팅 기술을 기반으로 Docker, Kubernetes, Airflow, Hadoop, Spark 등 여러 빅데이터 오픈소스 툴과 컨테이너 기술을 활용하여 MLOps(Machine Learning Operations)를 위한 클라우드 네이티브 데이터 파이프라인 자동화 및 ETL 작업 자동화 구축을 제안하고자 한다.

2. 관련 연구

2.1 NHN Cloud

NHN Cloud는 오픈 스택 기반 클라우드 서비스로 유연한 클라우드 인프라를 제공한다. 인프라에 기반한 플랫폼 중심 클라우드이며 인프라, 콘텐츠, 분석, 게임, 보안, 알림 메시지, 기타 애플리케이션을 운영할 때 필요한 각종 기능을 제공한다. NHN Kubernetes Service, NHN Container Registry 등 컨테이너 관련 기능과 여

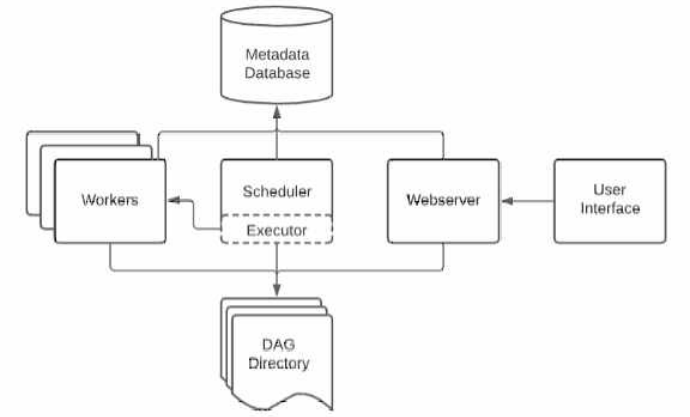
러 AI 서비스 기능 또한 제공하고 있다. 또한 사용한 만큼 요금을 지불하는 pay as you go 클라우드이므로 합리적인 가격으로 클라우드 서비스를 사용할 수 있다. 몇몇 보조 서비스들을 무료로 사용 가능하기도 한다. NHN Cloud는 웹 브라우저 상에서 간단한 클릭만으로 서비스를 사용할 수 있어 누구나 쉽게 사용할 수 있는 클라우드 서비스로 평가받고 있다.

2.2 에어플로우(Airflow)

Airflow는 Airbnb에서 개발한 데이터 파이프라인 자동화 오픈소스 플랫폼이며, 현재는 아파치 재단에서 관리중이다. Airflow는 DAG(Directed Acyclic Graph)라는 개념을 사용하여 작업 간의 의존성을 정의하고, 이를 바탕으로 지정한 일정에 따라 작업을 실행하고 모니터링한다. Airflow는 다양한 데이터 소스로부터 데이터를 추출하고, 변환하며, 적재할 수 있는 풍부한 연결성을 가진다. 또한 Airflow는 Python으로 작성되어 있기 때문에, 개발자들은 Python을 사용하여 각 작업을 편리하게 커스텀 하여 구성할 수 있다.

- Webserver: Airflow UI를 제공하며, DAG 및 작업 실행 정보를 시각화한다.
- Scheduler: 정의된 DAG를 실행하는 역할을 한다.
- Metadata Database: Airflow 설정과 DAG 상태 정보 등을 저장한다.
- Executor: Airflow가 작업을 실행하는 방식을 결정한다. LocalExecutor, CeleryExecutor, Kubernetes Executor 등이 있다.
- Worker: 실제 Task를 실행하는 주체이다.

주기적으로 실행하는 것은 Linux Cron과 유사하지만, Cron은 하나의 애플리케이션만 지정 가능하고 각 태스크를 연결하는 것이 불가능한 반면 Airflow는 복잡한 작업을 쉽게 구성할 수 있고, 각 태스크 별로 연결할 수 있어 전체 파이프라인 구성에 적합하다.



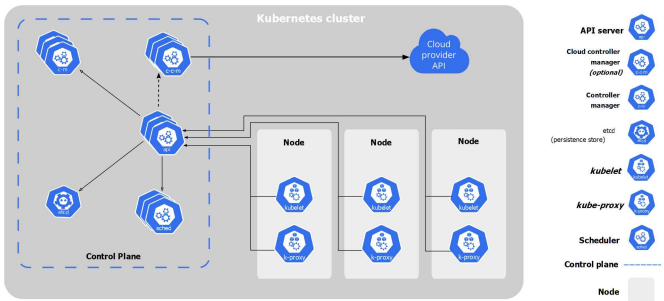
[그림 1] Airflow 구성도
[Fig. 1] Configuration of Airflow

2.3 쿠버네티스(Kubernetes)

Kubernetes는 컨테이너 오케스트레이션 도구이다. 컨테이너화 된 애플리케이션을 관리하고 배포하기 위한 플랫폼으로, 대규모 분산 시스템에서 일어날 수 있는 문제들을 자동으로 처리하고, 애플리케이션을 수평 확장하며, 서비스 디스커버리 및 로드밸런싱 등을 제공한다.

- 컨테이너 오케스트레이션: 컨테이너를 배치하고 스케줄링 하며, 컨테이너가 죽었을 때 자동으로 다시 시작하는 등의 기능을 제공한다.
- 서비스 디스커버리: 컨테이너화 된 애플리케이션의 인스턴스들을 자동으로 발견하고 로드밸런싱을 수행하여 애플리케이션의 가용성과 안정성을 높인다.
- 스토리지 오케스트레이션: 컨테이너화 된 애플리케이션에서 사용하는 스토리지 리소스를 관리하고, 스토리지 볼륨을 동적으로 할당하여, 데이터의 안정성과 지속성을 보장한다.
- 자동 스케일링: 컨테이너화 된 애플리케이션의 부하량에 따라 자동으로 인스턴스 수를 늘리거나 줄여서, 애플리케이션의 가용성과 성능을 최적화한다.

Kubernetes는 대규모 분산 시스템에서 사용될 수 있도록 설계되었으며, 클라우드, 온프레미스, 하이브리드 등 다양한 환경에서 사용될 수 있다. 현재 많은 국내외 기업들이 Kubernetes를 클라우드 네이티브 애플리케이션 개발 및 운영에 활용하고 있다.



[그림 2] Kubernetes 구성도
[Fig. 2] Configuration of Kubernetes

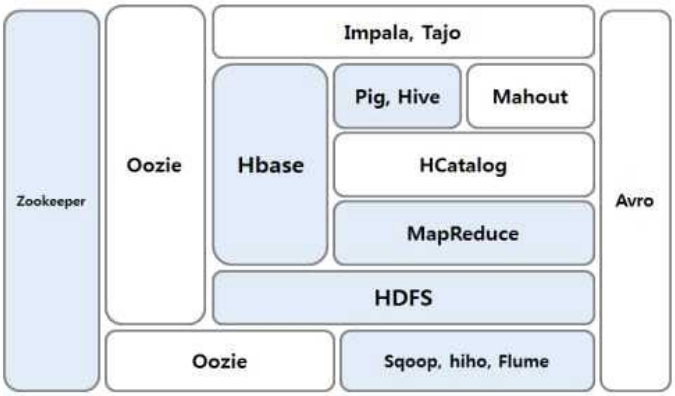
2.4 하둡(Hadoop)

Hadoop은 하나의 컴퓨터를 Scale up 하여 데이터를 처리하는 대신 적당한 성능의 범용 컴퓨터 여러 대를 Scale out한 후, 클러스터화 하여 큰 크기의 데이터를 클러스터에서 병렬로 동시에 처리한다. 이를 통해 처리 속도를 높이는 것을 목적으로 두는 오픈소스 프레임워크로 아래와 같은 모듈로 구성된다.

- Hadoop Common: Hadoop의 다른 모듈을 지원하기 위한 공통 컴포넌트 모듈

- Hadoop MapReduce: 대용량 데이터 처리를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작된 소프트웨어 프레임워크
- Hadoop HDFS: 분산 저장을 처리하기 위한 모듈, 여러 개의 서버를 하나의 서버처럼 묶어서 데이터를 저장
- Hadoop YARN: 병렬 처리를 위한 클러스터 자원 관리 및 스케줄링 담당
- Hadoop Ozone: 하둡을 위한 오브젝트 저장소

Hadoop은 시스템을 중단하지 않고 장비의 추가가 용이하며 일부 장비에 장애가 발생하더라도 전체 시스템 사용성에 영향이 적다는 장점이 있다. 그렇지만 데이터를 DISK 기반으로 처리하기 때문에 데이터 처리 시간 외에도 read/write 연산에 추가 시간이 소요된다. 또한 동일 데이터에 대해서 작업할 때, 매번 read 연산이 필요하다.



[그림 3] 하둡에코시스템 구성도
[Fig. 3] Configuration of Hadoop Ecosystem

2.5 스파크(Spark)

Spark는 빅데이터에 주로 사용되는빅데이터에 주로 사용되는 오픈소스 클러스터 컴퓨팅 프레임워크로 Hadoop의 MapReduce와 비슷한 역할을 한다. Spark는 통합 컴퓨팅 엔진이며 클러스터 환경에서 데이터를 병렬로 처리하는 라이브러리 집합이다. RDD(Resilent Distributed Datasets)을 사용하며 인메모리 저장 및 효과적인 장애복구 지원에 기반하여 동작한다. Spark는 반복적인 기계 학습 알고리즘에 대해 Hadoop과 비교해서 10배까지 빠른 성능을 나타낸다. Hadoop의 MapReduce는 작업의 중간 결과를 디스크에 써서 IO로 인해 작업 속도에 제약이 생기는 반면 Spark는 메모리에 중간결과를 저장하여 반복 작업의 처리 효율이 높다. Spark는 SQL, Streaming, 머신러닝, 그래프 연산 등 다양한 컴포넌트를 제공해준다. 또한 다양한 클러스터 매니저를 지원하여 YARN, Mesos, Kubernetes 등 다양한 클러스터에서 작동이 가능하다.

2.4 MLOps의 기존 병렬 처리에 관련된 솔루션 연구

Tensorflow는 그래프 연산을 사용하여 다중 CPU 및 GPU에서 병렬 처리를 지원한다. Tensorflow의 병렬 처리 방법에는 크게 두 가지가 존재한다. 첫 번째는 하나의 모델을 여러 개의 GPU 또는 CPU에 분할하여 병렬 처리하는 Data Parallelism 방법이다.

TensorFlow는 'tf.distribute.Strategy' API를 통해 Data Parallelism를 구현할 수 있다. 두 번째는 모델의 레이어를 여러 개의 GPU 또는 CPU에 분할하여 병렬 처리하는 Model Parallelism 방법이다. TensorFlow에서 Model Parallelism을 위한 고급 기능을 제공하지는 않지만, 사용자가 직접 구현하여 사용할 수 있다.

2.5 기존 연구의 문제점 및 해결 방안

Tensorflow에서 병렬 처리를 사용하는 것은 복잡하므로 사용자가 쉽게 구현하기는 어렵다. 또한 병렬 처리를 위해서는 효과적인 분산 환경이 필요하다. 따라서 많은 컴퓨팅 리소스를 사용할 수 있는 환경이 필요하며, 이를 위해서 비용적인 문제가 커질 수 있다. 또한 TensorFlow에서는 작업 스케줄링, 데이터 전송, 오류 처리 등과 같은 분산 처리에 관련된 문제를 해결하기 어렵다.

따라서 본 연구를 통해 병렬 처리 환경에서도 자동으로 컴퓨팅 리소스를 관리해줄 수 있는 환경을 구현하고자 한다. 자동 스케일링을 구현하여 컨테이너화된 애플리케이션의 부하량에 따라 자동으로 인스턴스 수를 늘리거나 줄여서, 애플리케이션의 가용성과 성능을 최적화 한다.

3. 연구 내용

3.1 시나리오

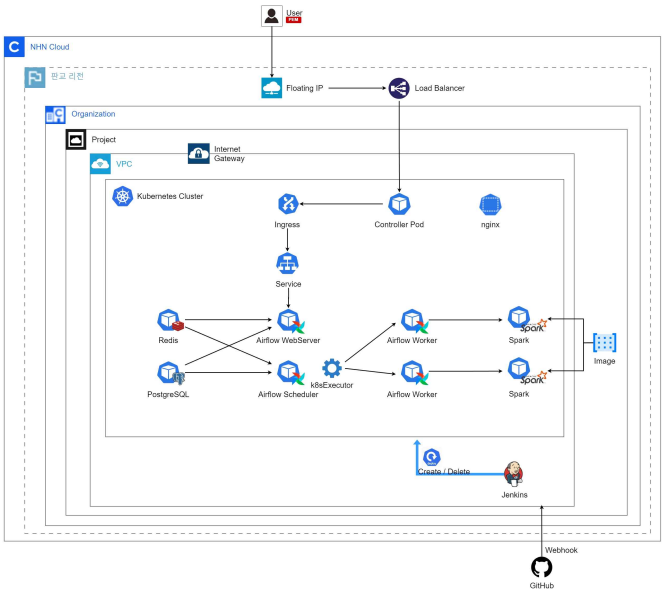
MLOps를 위한 클라우드 환경에서의 데이터파이프라인 구축 및 ETL 작업 자동화 구축을 목표로 한다. 요구사항은 다음과 같다. CI/CD를 통해 사용자의 코드를 자동으로 ETL 처리 서버에 적용한다. Airflow는 해당 코드를 분석하여 Task간 의존성 관계를 파악하고 배치 작업으로 등록한다. 배치 작업 수행 시에 적절하게 Task를 쪼개어 여러 Worker로 분산 처리한다. 이때 Worker는 Kubernetes Pod 형태로 동적으로 생성되며, Task가 완료되면 Pod는 삭제된다. Worker를 동적으로 생성하고 제거하는 과정에서 컴퓨팅 자원 지표를 분석하여 자동으로 필요한 리소스를 할당받거나, 필요하지 않은 리소스를 반납한다.

3,2 ETL 처리 서버

현재 데이터 수집 애플리케이션이 웹사이트에서 데이터를 크롤링하고 해당 데이터를 학습 데이터로 가공하기 위해 병렬로 전처리를 진행하고 있는 상태이다. 쿠버네티스 환경에서 리소스 사용량을 관찰하여 자동으로 클라우드 리소스를 할당받을 수 있도록 조정하는

작업을 진행하고 있다. 해당 작업이 끝나면 수집한 데이터를 활용하여 쿠버네티스 환경에서 병렬로 머신러닝을 학습시켜볼 예정이다.

3.3 프로젝트 아키텍처



[그림 4] 프로젝트 아키텍처
[Fig. 4] Project Architecture

MLOps를 위한 클라우드 환경에서의 데이터파이프라인 구축 및 ETL 작업 자동화 구축을 목표로 한다. 요구사항은 다음과 같다. CI/CD를 통해 사용자의 코드를 자동으로 ETL 처리 서버에 적용한다. Airflow는 해당 코드를 분석하여 Task간 의존성 관계를 파악하고 배치 작업으로 등록한다. 배치 작업 수행 시에 적절하게 Task를 쪼개어 여러 Worker로 분산 처리한다. 이때 Worker는 Kubernetes Pod 형태로 동적으로 생성되며, Task가 완료되면 Pod는 삭제된다. Worker를 동적으로 생성하고 제거하는 과정에서 컴퓨팅 자원 지표를 분석하여 자동으로 필요한 리소스를 할당받거나, 필요하지 않은 리소스를 반납한다.

4. 결론 및 향후연구

본 연구에서 제안하는 솔루션은 사용자가 컴퓨팅 자원이나 인스턴스간 네트워크 설정, 배치 작업 자동화 등에 대한 노력을 줄이고 필요한 개발에만 집중할 수 있는 플랫폼을 제공한다. 편리하고 직관적인 웹 UI를 함께 제공함으로써 손쉽고 편리하게 배치 작업을 등록하고 데이터 처리 결과를 확인할 수 있다..

머신 러닝에 대한 도입이 활성화되고 있는 요즘, 빅데이터 수집 및 처리에 대한 수요 또한 급증하고 있기 때문에 이를 필요로 하는 사용자에게 유용한 도구가 될 것이다.

5. 참고 문헌

[1] 박지훈, 「빅데이터 시스템의 데이터 수집 및 저장에 관한 연구」, 『2017년 추계학술발표대회 논문집 제24권 제2호』, 2017.

[2] 인포매티카, 「빅 데이터(Big Data)의 폭발적 증가 - 빅 데이터를 큰 비즈니스 기회로 연결시키는 Informatica 9.1 플랫폼」, 2011.

[3] 임수종, 민옥기, 「빅데이터 활용을 위한 기계학습 기술 동향」, 2012.

[4] hs_seo 저, 빅데이터 - 하둡, 하이브로 시작하기

[5] hs_seo 저, 빅데이터 - 스칼라, 스파크로 시작하기

[6] 신호승, 강성원, 이지현, 「확장형 실시간 데이터 파이프라인 시스템 아키텍처 설계」, 『정보과학회논문지』, 2015.

[7] 류우석, 「스파크를 이용한 머신러닝의 분산 처리 성능 요인」, 『한국전자통신학회 논문지』, 2021.