서버리스 컴퓨팅 기반의 확장 가능한 추천 시스템

Scalable Recommender System based on Serverless Computing

저자 (Authors)
이성재, 최재강, 최운호, 이경용
Sungjae Lee, Jaeghang Choi, Unho Choi, Kyungyon

출처 (Source)
한국정보과학회 학술발표논문집 , 2020.12, 16-18 (3 pages)

발행처 (Publisher)
한국정보과학회
The Korean Institute of Information Scientists and Engineers

URL
http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE10529527

APA Style
이성재, 최재강, 최운호, 이경용 (2020). 서버리스 컴퓨팅 기반의 확장 가능한 추천 시스템. 한국정보과학회 학술발표논문집, 16-18.

서버리스 컴퓨팅 기반의 확장 가능한 추천 시스템

이성재○, 최재강, 최운호, 이경용
국민대학교 소프트웨어융합대학
{odobenus, chl8273, yms04089, leeky}@kookmin.ac.kr

Scalable Recommender System based on Serverless Computing

Sungjae Lee○, Jaeghang Choi, Unho Choi, Kyungyong Lee
College of Computer Science, Kookmin University

요 약

서버리스 컴퓨팅은 클라우드 환경에서 복잡한 서버 관리와 사양에 대한 설정을 줄여, 사용자가 핵심적인 기능 개발에 집중할 수 있도록 구성한 컴퓨팅 환경을 의미한다. 서버리스 컴퓨팅을 구현하는 방식으로 FaaS (Function-as-a-Service) 가 있으며, 함수 단위의 가상화 및 실행을 통해 별도의 설정 없이 높은 확장성을 가진다. 이러한 서버리스 환경에서 기존의 응용을 개발하기 위해서는, 응용의 구조를 느슨하게 결합된 (Loose Coupling) 형태로 새롭게 설계할 필요가 있다. 본 논문에서는 추천 시스템에서 핵심이 되는 협업 필터링 알고리즘을 학습과 추론 단계로 나누어 서버리스 컴퓨팅 시스템에 적용하고, 실험을 통해 추론 단계에서 높은 확장성을 가짐을 보인다.

1. 서 론

사용자의 경험 패턴에 따라 사용자가 관심을 가질만한 새로운 아이템을 제안하는 것을 추천 시스템이라 한다. 기업들은 영화, 음악, 텍스트 등의 다양한 타입의 콘텐츠를 중심으로 개인화된 서비스를 제공하고자 이러한 추천 시스템을 적극적으로 도입하고 있다. 하지만 추천 시스템의 개발 및 운용에 있어서 확장성, 가용성 등을 고려하여 서비스를 제공하는 것은 여전히 많은 기업들에게 어려운 문제로 남아있다.

클라우드 컴퓨팅 기술은 직접 서버의 운용 및 관리가 필요한 온 프레미스(On-premise) 환경에서 벗어나, 필요에 따라 컴퓨팅 자원을 비용 효율적으로 사용할 수 있는 플랫폼을 제공한다. 최근에는 사용자 측면에서 하드웨어 및 소프트웨어 관리를 거의 하지 않고도 다양한 서비스를 구축할 수 있도록 완전 관리형 (Fully-managed) 서비스를 제공하는 추세이다. 그 중에서도 서버리스 컴퓨팅은 손쉽게 핵심 기능을 개발 및 배포할 수 있도록 도와주며, 동시에 수많은 요청을 안정적으로 처리할 수 있도록 설계되어 높은 확장성을 가지는 완전 관리형 서비스이다.

본 논문에서는 클라우드 환경에서 서버리스 컴퓨팅을 활용한 추천 시스템의 설계를 제안하고, 구현된 추천 시스템이 사용자의 서비스 이용 측면에서 일관된 성능을 보일 수 있다는 점을 실험을 통해 확인하고자 한다.

2. 클라우드 환경에서의 추천 시스템

2.1 추천 알고리즘의 학습과 추론

아마존, 넷플릭스, 스포티파이와 같이 개인화된 상품 정보, 미디어 콘텐츠 등을 제공하는 기업에서는 사용자의 과거 행동을 기반으로 미래의 선호를 예측하는 것이 중요하다. 이러한 목표를 이루기 위해서 사용자의 상품 구매 내역이나 콘텐츠 선호도를 분석하고, 추천 알고리즘을 통해 사용자가 아직 경험하지 못한 상품 또는 콘텐츠를 추천하게 된다.

추천 시스템에서 일반적으로 사용되는 알고리즘으로 협업 필터링 (Collaborative Filtering) 이 있으며, 아마존에서는 새로운 사용자를 대상으로 적절한 추천을 수행하기 위해 아이템 기반 협업 필터링 (Item-based Collaborative Filtering) 을 제안하였다[1]. 협업 필터링의 경우에는 크게 메모리 기반 (Memory-based), 모델 기반 (Model-based) 으로 나누어 지며, 본 논문에서 다룰 아이템 기반 협업 필터링의 경우 메모리 기반의 협업 필터링으로 볼 수 있다.

아이템 기반 협업 필터링 알고리즘은 크게 학습과 추론의 두 단계로 나누어 볼 수 있다. 우선 m 명의 사용자와 n 개의 아이템이 존재할 때, 각 사용자가 아이템에 대해 평가한 결과는 $m \times n$ 형태의 매트릭스로 표현할 수 있다. 학습 단계에서는 해당 매트릭스에 대해 아이템 사이의 유사도를 계산하여 $n \times n$ 형태의 매트릭스를 산출한다. 이 과정에서 유사도를 계산하는 방법으로는 유클리디안 거리 (Euclidean Distance), 코사인 유사도 (Cosine Similarity), 피어슨 유사도 (Pearson Correlation) 등을 사용하게 된다. 추론 단계에서는 특정한 사용자 1명이 n 개의 아이템을 평가한 $1 \times n$ 의 벡터가 입력으로 들어온다. 해당 벡터값과 앞에서 계산된 유사도 매트릭스를 내적(Dot Product) 연산을 통해 $1 \times n$ 벡터를 산출해 내고, 그 결과 특정한 사용자가 각각의 아이템에 대해 가지게 될 평가 점수를 나타내게 된다. 최종적으로 해당 $1 \times n$ 형태의 벡터에서 평가하지 않은 아이템 항목을 추출하여 가장 평가 점수가 높을 것으로 예상되는 아이템을 k 개 선정하여 반환하게 된다. 선정된 k 개의 아이템은 해당 사용자가 관심을 가질 가능성이 가장 높은 것으로 판단될 수 있으며, 이를 활용하여 개인 맞춤형 서비스의 제공이 가능하다.

2.2 비관리형 및 완전관리형 추천 시스템

클라우드 컴퓨팅 환경에서 개인화된 추천 시스템을 구축하기 위해서는 비관리형 (Unmanaged) 또는 완전 관리형 (Fully-managed) 서비스를 사용할 필요가 있다. 대표적인 클라우드 공급자인 AWS (Amazon Web Services) 에서 일반적으로 사용되는 비관리형 서비스로는 AWS EC2 와 같은 컴퓨팅 자원이 있다. 하지만 해당 환경에서 추천 시스템을 구축할 경우에는 CPU, Memory, Network 등의 하드웨어 설정과 OS, Runtime 등의 소프트웨어 환경 설정 및 관리가 복잡하며, 요청을 처리하기 위해 지속적인 비용 지출이 필요하다는 점에서 단점을 가지고 있다.

AWS에서 제공하는 완전 관리형 추천 시스템으로는 Amazon Personalize[2] 가 대표적이며, 해당 서비스를 통해 사용자는 하드웨어 및 소프트웨어 설정 없이 손쉽게 추천 시스템을 개발 및 배포할 수 있다. 하지만 지정된 기계학습 모델만 사용할 수 있다는 점과 추천 시스템을 사용하지 않는 상황에서도 기본 비용이 발생한다는 측면에서 여전히 단점을 가지고 있다.

2.3 서버리스 추천 시스템

서버리스 컴퓨팅은 서버의 하드웨어 및 소프트웨어의 사양을 거의 고려하지 않고 애플리케이션의 핵심 기능에 집중하여 개발할 수 있도록 한다. 또한 사용량에 따라 서비스가 확장 및 축소되며, 이에 맞춰 요금이 부과되기 때문에 유연한 시스템 구조를 가진다고 볼 수 있다.

최근에는 이러한 서버리스 기반의 다양한 애플리케이션을 개발하고 성능을 분석하는 연구가 진행되고 있다 [3]. 아이템 기반 협업 필터링의 경우 매트릭스 곱연산과 벡터-매트릭스 내적 연산이 핵심 기능이며, 각각의 연산이 서로 다른 기능으로 작동할 필요가 있다. 본 논문에서는 AWS Lambda 를 중심으로 Amazon S3, RDS 등의 서비스와 함께 느슨하게 결합된 형태의 추천 시스템을 제안한다. 이 과정에서 추천 알고리즘으로 사용되는 아이템 기반 협업 필터링을 학습 및 추론 과정으로 나누어 사용자 요청의 증감에 따라 유연하게 확장 및 축소가 가능하고, 비용 최적화된 추천 시스템의 구성이 가능함을 보인다.

3. 서버리스 추천 시스템의 구조

3.1 학습을 위한 주기적 유사도 계산

아이템 기반 협업 필터링을 통해 추천 시스템을 작동시키기 위한 첫 단계는 사용자-아이템 데이터베이스를 통해 아이템 유사도 매트릭스를 계산하는 것이다. 본 시스템에서는 사용자-아이템 데이터베이스 구축을 위하여 AWS RDS (Relational Database)를 사용하였다. RDS에서 추출된 매트릭스의 크기는 사용자의 수 m과 아이템의 수 n에 의해 $m \times n$ 크기를 가지게 되므로, 매 요청마다 유사도를 계산하는 것은 서비스에 지장을 미칠 정도의 과도한 컴퓨팅 자원을 소모한다고 볼 수 있다.

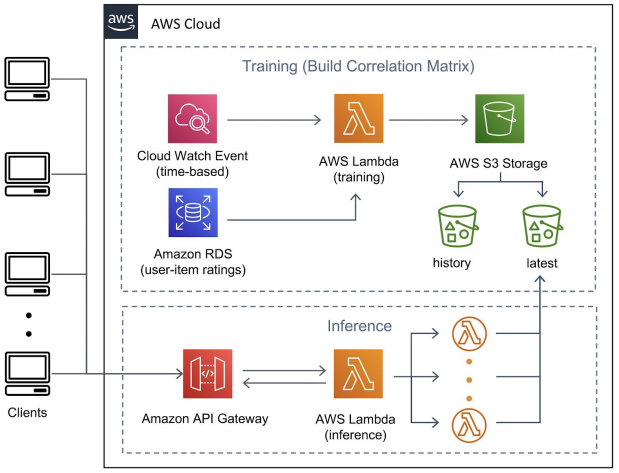

그림 1. 서버리스 추천 시스템의 전체 구조도

이러한 문제를 해결하기 위해 일반적으로 특정 주기마다 아이템 유사도 매트릭스를 계산하도록 시스템을 구성할 필요가 있다. 본 서버리스 추천 시스템에서는 AWS CloudWatch 의 시간 기반 호출을 통해 Lambda 가 RDS에 쿼리를 전송하고, 데이터를 불러와 유사도 계산을 수행하도록 한다. 그 결과 아이템 유사도 매트릭스를 얻을 수 있으며 이를 AWS S3 (Simple Storage Service)에 저장하여 추론 과정에서 빠른 접근이 가능하도록 한다.

3.2 확장 가능한 실시간 추천

주기적 유사도 계산이 완료되어 아이템간의 상관관계를 표현하는 매트릭스를 사용할 수 있는 시점부터, 사용자에게 실시간 추천 서비스를 제공하는 것이 가능해진다. 사용자는 아이템에 대한 평점 정보를 벡터 형태로 전달하게 되며, 해당 정보는 Amazon API Gateway 를 통해 추론 작업을 수행하는 Lambda (inference) 에 도달한다. Lambda 는 해당 벡터와 앞에서 계산된 아이템 유사도 매트릭스를 내적하기 위해, S3 에 저장된 최신(latest)의 아이템 유사도 매트릭스를 불러온다. 내적의 결과 해당 사용자가 경험하지 못한 아이템에 대해 예상되는 평점 정보를 구할 수 있으며, 정렬 알고리즘을 통해 예상 평점이 가장 높은 k개의 아이템을 사용자에게 전달하게 된다.

4. 실험 및 평가

4.1 실험 환경

서버리스 추천 시스템의 확장성을 검증하기 위한 실험은 AWS 클라우드의 FaaS 인 AWS Lambda 를 활용하여 진행되었다. 추천 시스템에서 학습을 수행하는 training 함수와 추론을 수행하는 inference 함수는 모두 128MB 메모리와 Python 3.7 런타임을 기반으로 생성되었다. inference 함수에 대한 동시 호출은 Amazon API Gateway 를 통해 HTTP 통신으로 이루어지며, 호출 시 마다 핵심 기능을 수행하는 데 소요되는 시간을 측정 및 반환한다.

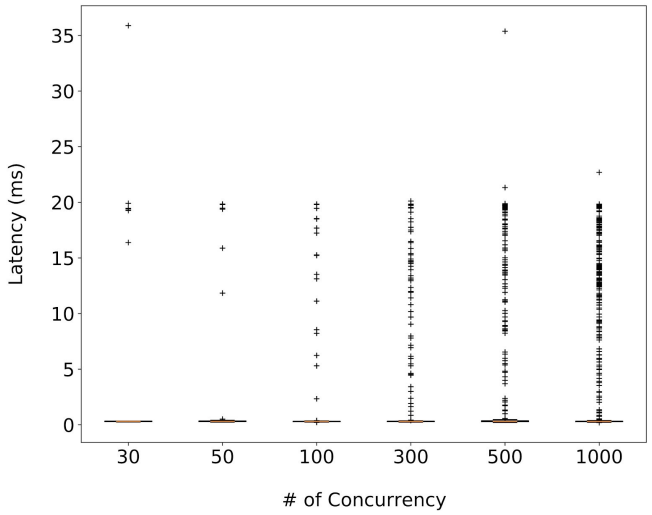

그림 2. 추론 요청의 동시성에 따른 응답 지연시간의 분포

inference 함수는 호출 시 마다 사전에 연산이 완료된 아이템 유사도 매트릭스 파일을 AWS S3 에서 읽어오며, 이를 기반으로 상품 추천을 위한 벡터와 매트릭스의 내적 연산을 수행한다. 실험에 사용된 모든 서비스는 동일한 ap-northeast-2 (seoul) 지역에서 구성되었다.

4.2 실험 방법

서버리스 추천 시스템의 추론 기능에서의 확장성을 테스트하기 위한 방법으로는, 수많은 추론 요청을 동시에 전송하여 각각의 요청에 대한 응답 지연시간을 측정하는 것이 일반적이다 [4]. 동시 요청의 횟수가 증가함에도 측정된 지연시간의 범위가 일정하다면 해당 시스템이 높은 확장성을 가진 안정적인 시스템이라는 것을 확인할 수 있다. 실험의 설계를 위하여 현재 데모 서비스로 운영되고 있는 스낵 추천 시스템[5] 에서의 사용자별 스낵 평점 데이터베이스를 활용하였다. 동시 요청의 횟수는 적은 양의 30회 부터 50, 100, 300, 500, 1000으로 증가시키며 실험을 진행하였다.

4.3 실험 결과

제안되는 서버리스 추천 시스템에서의 동시성 실험 결과는 각각의 동시 요청 횟수에 따라 [그림 2]와 같이 일정한 범위 내에서 지연시간이 분포됨을 확인할 수 있다. boxplot 의 중심점이 0에 가까운 값을 가지는 것은 대부분 요청에 대한 응답이 0.3ms 내외로 이루어 졌기 때문이며, 그 외에는 1~20ms 사이에서 지연된 응답이 발생하였다. 이러한 지연된 응답이 발생하는 것은 서버리스 컴퓨팅 기술이 가지는 자체적인 문제점으로, 요청을 처리하는 microVM이 작동하기 위한 초기 지연 시간이 발생하기 때문이다. 이러한 cold-start 문제로 인해 지연 시간이 증가하는 문제가 있지만, 30회의 동시 요청에 비해 1,000회의 동시 요청에서도 응답 지연 시간의 발생 범위가 1~20ms 사이라는 점에서 뛰어난 확장성을 가지는 시스템이라고 말할 수 있다.

5. 결론 및 향후 계획

이번 연구에서는 서버리스 환경에서 확장 가능한 추천 시스템의 설계를 제안하고, 해당 시스템이 추론 기능에서 가지는 확장성을 실험을 통해 보였다.

향후 다음과 같은 측면에서 연구의 발전이 필요하다고 생각된다. 첫째, 현재의 실험 시나리오에서는 단일 크기의 데이터 베이스를 사용하여 작은 크기의 사용자-아이템 매트릭스 환경에서만 테스트가 진행되었다. 대부분의 평가 데이터가 희소 행렬의 형태를 띄고 있다는 점에서 다양한 크기와 희소성을 고려한 데이터에서 실험을 진행할 필요가 있다. 나아가 추천 시스템을 테스트하기 위한 데이터셋으로 movielens[6] 등의 일반적인 평점 데이터셋을 도입하는 것 또한 의미있을 것이다. 둘째, 본 연구에서는 사용자의 추론 요청에 대해서만 확장 가능하도록 추천 시스템을 구성하였으나, 내부적으로 수행되는 학습 과정에서도 확장 가능한 시스템을 구현할 필요가 있다. 요구되는 매트릭스의 규모에 따라 AWS EMR 클러스터를 활용한 대규모 매트릭스 곱연산으로 모듈을 대체할 수도 있으며, 또는 여러 개의 서버리스 함수가 동시 작동하여 해당 연산을 분산 처리하는 방법을 적용할 수도 있다. 마지막으로 실험에서도 확인한 바와 같이 cold-start 로 인한 지연시간의 증가를 해결하기 위한 방안을 마련할 필요가 있다. 현재 서버리스 컴퓨팅 환경에서 제공되는 provisioning 기능 등을 활용하여 추가 비용에 따른 성능 향상이 어느 정도인지 비교 분석하고자 한다.

6. 사사

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학지원사업 (2016-0-00021), ICT 연구과제 (2017-0-00396) 및 한국연구재단 이공분야 기초연구 사업 (NRF2016R1C1B2015135)의 지원을 받아 수행됨.

참고문헌

[1] Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, *7*(1), 76-80.

[2] AWS Personalize, https://aws.amazon.com/personalize

[3] Kim, J., & Lee, K. FunctionBench: A suite of workloads for serverless cloud function service. In *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)* (pp. 502-504). IEEE.

[4] Park, J., Kim, H., & Lee, K. Evaluating Concurrent Executions of Multiple Function-as-a-Service Runtimes with MicroVM. In *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)* (Accepted).

[5] snackpot (snack recommendation), http://snackpot.kr

[6] Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, *5*(4), 1-19.