하둡 맵리듀스에서의 Item-Item CF알고리즘적용 설계

채한울 이창현 김다은 송재영
국민대학교 소프트웨어학부 소프트웨어학과 전공
lunahc92@gmail.com, lch01387@gmail.com, ekdms71700@gmail.com

Design of Item-Item CF Algorithm in Hadoop Map-Reduce

Hanul Chae , ChangHyun Lee , DaEun Kim · JaeYoung Song
Computer Science, Kookmin University

요 약

Item-Item CF 알고리즘을 이용하여 추천 시스템을 구현하는 프로젝트이다. 사용자가 review한 정보를 토대로 선호도를 알고 싶은 Item의 선호도를 예측할 수 있다. 사용자가 입력한 유사도는 Cosine Similarity를 사용하였다. 계산된 Similarity를 Item-Item CF 알고리즘에 적용하여 선호도를 구할 수 있다.

1. 서 론

본 과제에서는 국민대학교 소프트웨어학부의 빅데이터 최신기술 강의에서 시작하여 Hadoop 플랫폼에 Item-Item CF 알고리즘을 적용하는 프로젝트를 진행하였다.

본 과제에서는 아마존 쇼핑몰을 이용한 특정 사용자의 과거 review 데이터를 토대로 새롭게 주어진 item에 대한 선호도를 측정하려고 한다. 아마존에서는 실시간으로 수많은 리뷰데이터를 쏟아내며, 이러한 빅데이터를 분석하는 과정에서 종래의 중앙집중형 컴퓨팅은 효율적이지 못함이 알려져 있다. 2004년 구글에서 발표된 map-reduce 방식은 크기가 큰 데이터를 분산처리를 통하여 효율적으로 분석할 수 있다. Item-Item CF(Collaborative Filtering)은 과거에 사용자들로부터 얻은 기호정보에 따라 사용자들의 관심사를 예측할수 있도록 도와주는 알고리즘으로서 본 과제의 목표와 같이 아직 알려지지 않은 특정 아이템의 선호도를 추측하는데 적절한 해결법이다. 본 과제에서는 이러한 Item-Item CF를 map-reduce개념을 적용할수 있도록 고안된 Hadoop에서 활용해보고자 한다. Testset은 아마존 Book market의 리뷰 데이터를 이용하였다.

본 논문의 구성은 다음과 같다. 2장에서는 Item-Item CF를 도출하기 위한 중간과정인 Similarity의 개념을 소개하고 3장에서는 Item-Item CF 연산과정을 설명한다. 4장에서는 Item-Item CF공식을 최적화한다. 5장에서는 알고리즘을 하둡 map-reduce 적용하기까지의 설계을 도식화 하며, 마지막으로 6장에서 실행결과를 기록하며 마무리지으려고 한다.

2. Similarity

$$\frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

그림1. Similarity 측정공식

Similarity(유사도) 측정은 Item-Item CF 연산에 필요한 중간과정이다. 각각의 Item은 수많은 사용자들에게 평가점수를 부여받으며 이것은 사용자 숫자만큼의 차원을 가진 벡터로서 표현될 수 있다. Similarity는 이러한 벡터들이 서로 얼마나 유사한지 나타낸다. 그중에서도 코사인 유사도(cosine similarity)는 내적공간의 두 벡터간 각도의 코사인값을 이용하여 두 벡터간의 유사한 정도를 나타낸다. cos0° 곧 1의 결과값을 가질 때 가장 유사한 벡터임을 직관적으로 파악할 수 있다.

3. Item-Item CF 알고리즘

Item-Item CF 알고리즘은 앞서 측정된 similarity의 값을 통해 Item-Item CF 알고리즘을 적용하여 구하고자 했던 선호도를 알아낼 수 있다. 해당 알고리즘은 다음과 같다.

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

r_{ix} be the vector of user x's ratings on item i
s_{ij}... similarity of items i and j
$N(i;x)$... set items rated by x similar to i

그림2. Item-Item CF 공식

위 공식으로 사용자의 과거 리뷰정보를 참고하여 다른 아이템에 대한 선호도를 측정할수 있다. 선호도를 구하고자 하는 Item과 사용자가 과거 리뷰했던 Item들 간의 similarity값이 필요하며, Similarity값이 높은 상위 몇개의 리뷰정보만 이용하는 방법이 존재하지만 이번 과제에서는 사용자의 과거 모든 리뷰정보를 이용했다.

4. Item-Item CF 알고리즘 최적화

Item-Item CF 알고리즘에서 간단한 최적화로 map-reduce단계를 줄일 수 있다. Item-Item CF공식에 similarity공식을 대입하면 다음과 같은 모양이 된다.

$$r_{kx} = \frac{\sum_{i \in N} \frac{k \circ i}{\|k\| \cdot \|i\|} \times r_{ix}}{\sum \frac{k \circ i}{\|k\| \cdot \|i\|}}$$

그림3. Item-Item CF공식 최적화

이때 K벡터에 해당하는 item은 고정되어 있으며, 그때문에 벡터K의 크기값은 분모분자간 서로 약분될 수 있다.

5. 맵리듀스 설계 과정

Item-Item CF를 Hadoop에 적용하기 위해서는 여러단계의 맵리듀스를 거쳐야 한다. 이전 단계 job의 출력을 다음단계 job이 읽어들이는 방식이며, 하나의 job이 2개이상의 출력을 읽어들일 때는 2개의 map클래스가 하나의 reduce클래스에 값을 전달한다. 본 과제에서는 총 7개의 job으로 구성된 Hadoop프로그램을 작성했으며 프로그램의 구성도는 다음과 같다.

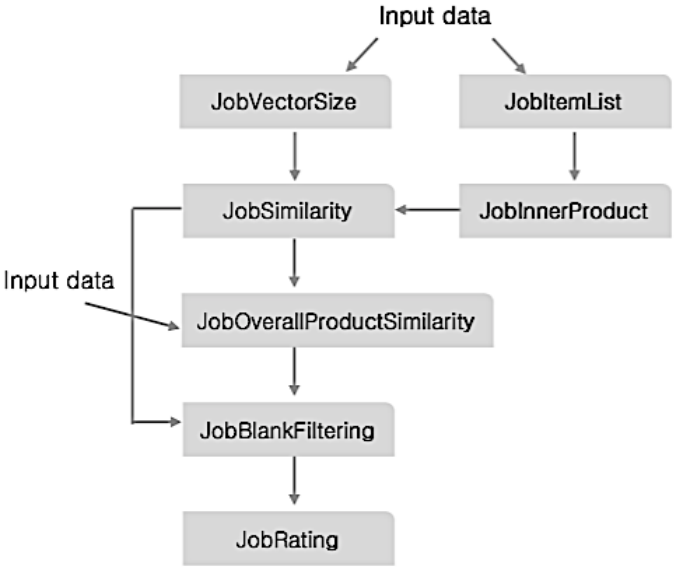

그림4. 본 프로그램 구성도

위 구성도는 특정 사용자와 그 사용자가 평가한적 없는 Item을 입력받았을 때 사용자의 해당 Item에 대한 선호도(Rating)을 연산하는 과정을 보여준다. 각각의 Item벡터의 크기를 나타내는 VectorSize와 각각의 Item의 내적값을 나타내는 InnerProduct는 앞서 설명된 Similarity를 연산하기 위한 준비과정이다. OverallProductSimilarity는 사용자가 평가한 Item-Item CF값의 연산값으로 이용된다. BlankFiltering은 사용자가 이전에 평가하지 않았던 item이 item-item CF결과에 영향을 미치지 않게하기위해 추가된 과정이다. 최종연산은 Rating에서 이루어지며, 결과값으로 최초에 입력된 Item과 해당 Item에 대한 사용자의 추측된 선호도가 씌여진다.

6. 결 론

본 과제의 실행환경은 Hadoop 2.6.00이다. 실행Command는

hadoop jar ItemCF.jar kr.ac.kookmin.cs.bigdata.ItemCF /input/amazon/reviews_Books_5.json /student4/ItemCF A2137L6QD68KVH 0001055178 이다.

출력 결과는 다음과같다.

```
A2137L6QD68KVH  3.422954678155324
```

그림5. 실행 결과

사용자ID와 예상 선호도가($1 \leq R \leq 5$) 출력되었다. Hadoop시스템내에서 동시에 실행되는 별다른 task가 존재하지 않을 때 총 실행시간은 약 7분이었다. 설계 과정의 예상값과 상기 결과가 유사하며 성능또한 만족할만한 결과로 보인다.