ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

상품 동시 발생 정보와 유사도 정보를 이용한 협업적 필터링

Collaborative Filtering using Co-Occurrence and Similarity information

나 광 택[1] 이 주 홍[1*]
Kwang Tek Na Ju Hong Lee

요 약

협업적 필터링(CF)은 사용자와 상품간의 관계를 해석하여 특정 사용자에게 상품을 추천 해주는 시스템이다. CF 모델은 컨텐츠 등 다른 추가 정보 없이 평점 데이터만으로 사용자에게 상품을 추천해 줄 수 있다는 장점이 있다. 하지만 사용자는 전체 상품의 극히 일부분만을 소비하고 상품을 소비한 후에도 평점을 부여하지 않는 경우가 매우 많다. 이는 관찰된 평점의 수가 매우 적으며 사용자 평점 행렬이 매우 희박함을 의미한다. 이러한 평점 데이터의 희박성은 CF의 성능을 끌어올리는데 문제를 야기한다. 본 논문에서는 CF 모델 중 하나인 잠재 요인 모델(특히 SVD)의 성능을 끌어올리는데 집중한다. SVD에 상품 유사도 정보와 상품 동시 발생(co occurrence) 정보를 포함시킨 새로운 모델을 제안한다. 평점 데이터로부터 얻어지는 유사도와 동시 발생 정보는 상품 잠재 요인에 대한 잠재 공간상의 표현력을 높여주어 기존방법보다 Recall은 약 16%, Precision과 NDCG는 각각 8%, 7% 상승하였다. 본 논문에서 제안하는 방법이 향후 다른 추천 시스템과 결합하면 기존의 방법보다 더 좋은 성능을 보여줄 것이다.

☞ 주제어 : 협업적 필터링, 추천시스템, 동시 발생, 유사도, SVD, 잠재 요인 모델

ABSTRACT

Collaborative filtering (CF) is a system that interprets the relationship between a user and a product and recommends the product to a specific user. The CF model is advantageous in that it can recommend products to users with only rating data without any additional information such as contents. However, there are many cases where a user does not give a rating even after consuming the product as well as consuming only a small portion of the total product. This means that the number of ratings observed is very small and the user rating matrix is very sparse. The sparsity of this rating data poses a problem in raising CF performance. In this paper, we concentrate on raising the performance of latent factor model (especially SVD). We propose a new model that includes product similarity information and co occurrence information in SVD. The similarity and concurrence information obtained from the rating data increased the expressiveness of the latent space in terms of latent factors. Thus, Recall increased by 16% and Precision and NDCG increased by 8% and 7%, respectively. The proposed method of the paper will show better performance than the existing method when combined with other recommender systems in the future.

☞ keyword : Collaborative filtering, Recommender system, co occurrence, similarity, SVD, latent factor model

1. 서 론

현대의 소비자들은 넘쳐나는 정보 속에서 자신의 의사결정을 해야 한다. 이들 정보는 유용한 정보도 있겠지만, 의미 없는 정보들도 상당하다. 이러한 정보의 과부하는 소비자로 하여금 선택의 혼란을 가중한다. 전자상거래 업체 역시 수 많은 서비스 및 상품들을 제공하고 있으며, 소비자들의 효율적인 의사결정을 돕기 위해 추천 시스템의 중요도는 날로 높아지고 있다[1]. 추천 시스템이란 특정 사용자가 특정 상품에 흥미가 있을지 없을지 판단하거나(prediction problem), 특정 사용자가 관심을 가질 만한 N개의 상품 항목을 찾아내는데(top-N Recommendation problem) 사용되는 기술이다[2]. 여러 추천 시스템 중에서도 협업적 필터링(CF)는 많은 전자상거래 업체에서 대표적으로 사용하는 모델이다. 다른 추천 시스템과 비교해 CF는 콘텐츠 등 다른 추가 정보를 필요로 하지 않는다[3]. 이는 오직 사용자들의 과거 행적만을 이용해 상품을 추천해 준다는 의미이며, 기존의 사용자와 상품 간의 관계를 이용해 새로운 사용자-상품 관계를 예측한다[4]. CF 모델은 크게 이웃 기반과 잠재 요인 모델 기반으로 나뉜다[5]. 이웃 기반 모델은 속도는 빠르지만 추천 품질이 떨어지는 단점이 있으며, 잠재 요인 모델 기반 CF는 추천을 하는 비용이 높은 반면, 추천의 질이 높다는 장점이 있다

[1] Dept of Computer Science and Information Engineering, Inha University, Incheon, 22201, Korea.
* Corresponding author (juhong@inha.ac.kr)
[Received 24 November 2016, Reviewed 30 November 2016(R2 10 March 2017), Accepted 4 April 2017]

[6]. 잠재 요인 모델은 상품과 아이템을 각각 잠재 요인 공간(latent factor space)의 벡터로서 표현한다. 잠재 요인 모델은 최근에 많은 연구들이 진행되었다. Koren[7]은 암시적 피드백을 모델에 통합할 수 있는 SVD++을 제안하였고, Hoffmann[8]은 사용자 상품 관계를 잠재 변수(latent variable)로 설명한 생성 모델을 제안했다. 자연어 처리 분야에서 주목받고 있는 단어 임베딩(word embedding)을 응용한 모델도 등장했다. 이는 사용자는 문서, 상품은 단어로 취급한 동시 발생 행렬을 모델에 추가함으로써 성능 향상을 꾀했다. 하지만 이 모델은 사용자가 명시적으로 자신의 의견을 표현한 평점 데이터를 활용하지 못한다는 단점이 있다.

본 논문에서는 SVD+CS라는 새로운 잠재 요인 모델을 제안한다. SVD+CS는 평점 행렬과 상품-유사도, 상품-동시 발생 행렬을 동시에 분해하여 top-n 추천의 품질을 높였다. 이러한 결과는 잠재 공간에 표현되는 상품 요인이 상품-사용자, 상품-상품, 상품-콘텍스트 상품 관계를 설명할 수 있는 능력을 갖는다는 것을 의미한다.

본 논문은 다음과 같이 구성된다. 2절에서는 지금까지 제안된 관련 연구들을 소개하고 본 논문의 제안 방법을 이해하는데 필요한 기본적인 용어와 기본적인 방법들을 소개한다. 3절에서는 본 논문에서 제안하는 SVD+CS를 소개하며 4절에서는 실험결과를 소개한다.

2. 관련 연구

Top-N 추천이란 사용자가 관심을 가질 만한 순위가 매겨진 N개의 상품을 추천해 주는 것이다. Top-N 추천은 크게 두 가지 범주로 나뉘는데, 하나는 이웃 기반이며 다른 하나는 잠재 요인 모델 기반이다[5]. 이웃 기반은 다시 사용자 기반과 상품 기반으로 나뉜다. 사용자기반 모델은 특정 사용자가 구매 혹은 평점을 부여한 패턴을 분석하여 비슷한 이웃 사용자를 찾는다. 그리고 이웃 사용자를 기반으로 N개의 상품을 해당 사용자에게 추천해준다. 반대로 상품 기반 모델은 특정 상품의 판매 혹은 받은 평점을 이용해 이웃 상품들을 찾아낸다. 그 후에 이웃 상품을 기반으로 특정 사용자에게 N개의 상품을 추천해준다[6]. 이웃 상품 기반을 이용하는 대표적인 기업이 아마존이다. 아마존은 미리 특정 상품과 유사한 상품들의 리스트를 만들어 놓고 사용자가 어떤 상품을 구매하거나 평점을 매기면 미리 만들어 놓은 유사 상품 리스트에서 상품들을 추천해준다.

잠재 요인 모델은 행렬 분해와 같은 방법을 사용해 평점 행렬을 사용자 요인과 상품 요인으로 분해하여 같은 잠재 요인 공간에 매핑 시킨다. 사용자 요인은 사용자의 취향을 표현할 수 있고, 상품 요인은 상품의 특징을 표현할 수 있다. 하지만 요인에 대한 정확한 해석은 불가능하다. 예측은 두 사용자 요인과 상품 요인의 내적으로 이뤄진다.

일반적으로 사용자-상품 관계 데이터(평점/구매 데이터)의 경우 그 희소 정도가 95% 이상으로 대단히 높다. 많은 결측치는 추천 시스템의 성능에 커다란 영향을 미치므로 추천 시스템은 데이터의 높은 희소성을 극복해야 한다.

초창기 시스템에서는 관찰되지 않은 평점들을 다른 특정 숫자로 치환하여 행렬을 분해하였다. 하지만 이러한 방식은 치환으로 인한 비용이 증가하고 그 결과 또한 좋지 못했다. 이러한 단점을 보완하고자 관찰된 데이터만을 이용하여 모델링 하는 방식이 제안되었다[9, 10, 11].

Koren[7]은 사용자의 평점을 명시적 피드백(explicit feedback)으로 분류하고, 평점을 부여하거나 또는 부여하지 않은 행위 자체를 암시적 피드백(implicit feedback)으로 분류하였다. 그리고 이 암시적 피드백을 모델에 적용시켜 성능을 향상시킨 SVD++을 발표하였다. Hofmann[8]은 PLSA(Probabilistic Latent Semantic Analysis)를 제안했다. 사용자와 상품의 관계를 잠재 변수를 도입하여 설명하였다. X. Liu et al.[3]은 커널을 이용한 행렬 분해를 제안했다. 사용자와 상품의 요인이 선형성을 보이지 않는다면 기존의 행렬 분해 방법으로는 사용자-상품 관계를 해석할 수 없다 판단하였다. 그래서 커널을 이용해 이들을 고차원 공간에 매핑하여 비선형 관계를 해석할 수 있는 커널라이즈드 행렬 분해를 제안하였다. D. Liang et al.[12]은 단어 임베딩을 평점 행렬에 응용하여 상품 임베딩을 생성하는 모델을 제안하였다.

잠재 요인 모델은 데이터의 다양한 면을 잘 설명할 수 있는 능력이 있으며, 여러 연구에서 이웃 기반 모델보다 우수한 성능을 보이는 것이 입증되었다. 하지만 대부분의 전자상거래 시스템들은 이웃 기반 모델에 기반을 두고 있다. 특히 아마존의 CF는 아이템 기반이다. 실제 시스템에서 이러한 정확도가 낮은 모델을 고수하는 이유는 다음과 같다. 첫째, 추천 결과의 이유를 잘 설명할 수 있으며, 사용자 경험의 질을 향상시킨다. 둘째, 사용자의 피드백이 입력되는 즉시 추천이 가능하다[10]. 반면 잠재 요인 모델은 Netflix Prize를 통해 그 중요성이 부각되었다. 특히 우승자의 알고리즘이 SVD를 이용한 행렬 분해였다.

2.1 CF를 위한 행렬 분해(SVD)

일반적으로 행렬분해를 이용한 CF는 평점 행렬을 사용자 행렬과 아이템 행렬로 분해하는 방식을 사용한다.

$$R \cong P^{T}Q$$

$$(R \in \mathbb{R}^{U\times I}, P \in \mathbb{R}^{U\times f}, Q \in \mathbb{R}^{I\times f})$$

사용자 U, 아이템 I에 대해서 평점 행렬 $R \in \mathbb{R}^{U\times I}$이 주어지면 사용자 잠재 요인 $p_u \in \mathbb{R}^f$ $(u = 1, \dots, U)$, 아이템 잠재 요인 $q_i \in \mathbb{R}^f$ $(i = 1, \dots, I)$간의 내적으로 표현된다. 즉 사용자와 아이템을 잠재 요인 공간에 맵핑 하는 것이다. 시스템은 다음과 같은 손실 함수를 최소화시키는 p_u와 q_i 찾는다[7].

$$\min_{p_*,q_*} \sum_{(u,i)\in\mathcal{K}} (r_{ui} - p_u^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \quad (1)$$

r_{ui}는 사용자 u가 상품 i에 부여한 평점이며, $\mathcal{K}$는 관찰된 (u,i)쌍 집합이다. 상품 평점과 같은 데이터는 사용자 각각이 저마다 평점을 부여하는 특성이 있다. 또한 상품의 경우 평점을 부여받는 특성이 있다. 예를 들어 어떤 사용자의 경우 다른 사용자보다 점수를 후하게 줄 수도 있는 반면, 어떤 사용자는 점수를 적게 줄 수도 있다. 이런 특성을 반영하기 위해 $b_{ui} = \mu + b_u + b_i$ 로 표현되는 기준선 예측을 추가한다. μ는 전체 평점의 평균, b_u와 b_i 는 각각 사용자와 아이템의 평균 μ와의 차이이다. 기준선 예측을 식 (1)에 적용하면 다음과 같다.

$$\min_{p_*,q_*} \sum_{(u,i)\in\mathcal{K}} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2\right) \quad (2)$$

식(1)과 식(2)에 나타난 λ항은 벌점항으로 과적합 문제를 방지한다. λ는 설계자가 경험적으로 선택하는 상수이다. 위 문제는 경사 하강법(gradient descent)으로 최적화가 가능하다.

2.2 피어슨 상관계수(PCC)

PCC는 사용자 기반 또는 상품 기반 CF에서 두 항목 간의 유사도를 측정하는 측도로써 주로 사용된다. 사용자 기반 CF 모델에서는 사용자 a, b가 공통으로 평점을 부여한 상품을 기반으로 두 사용자 간의 유사도를 구한다[13].

$$\mathrm{Sim}(a,b) = \frac{\sum_{i\in I(a)\cap I(b)}(r_{ai} - \bar{r_a})(r_{bi} - \bar{r_b})}{\sqrt{\sum_{i\in I(a)\cap I(b)}(r_{a,i} - \bar{r_a})^2}\sqrt{\sum_{i\in I(a)\cap I(b)}(r_{b,i} - \bar{r_b})^2}} \quad (3)$$

식 (3)은 사용자 a, b간의 유사도를 의미하고 i는 사용자 a, b가 공통으로 평점을 부여한 상품 집합이다. r_{ai}는 사용자 a가 상품 i에 부여한 평점을 의미하고 $\bar{r_a}$ 는 사용자 a의 평균 평점이다.

비슷하게 상품 기반 CF모델에서는 다음과 같이 상품 i, j간의 유사도를 구한다[13].

$$\mathrm{Sim}(i,j) = \frac{\sum_{u\in U(i)\cap U(j)}(r_{ui} - \bar{r_i})(r_{uj} - \bar{r_j})}{\sqrt{\sum_{u\in U(i)\cap U(j)}(r_{u,i} - \bar{r_i})^2}\sqrt{\sum_{u\in U(i)\cap U(j)}(r_{u,j} - \bar{r_j})^2}} \quad (4)$$

식 (4)는 아이템 i와 j간의 유사도를 나타내며, u는 아이템 i와 j에 모두 평점을 부여한 사용자를 나타낸다.

2.3 중요성 가중치

일반적인 PCC는 평점 데이터가 적은 경우에 결과가 좋지 못하다. 실제로는 다른 선호도를 보이는 두 사용자임에도 불구하고 그들이 부여한 평점 데이터가 적은 경우 유사도가 높게 나올 수 있다. 그래서 Hao Ma at el.는 이와 같은 문제를 해결하기 위해 유사도에 가중치를 부여하였다[13].

$$\mathrm{Sim}'(a,b) = \frac{\min(|I_a \cap I_u|, \varepsilon)}{\varepsilon}\mathrm{Sim}(a,b) \quad (5)$$

$$\mathrm{Sim}'(i,j) = \frac{\min(|U_i \cap U_j|, \delta)}{\delta}\mathrm{Sim}(i,j) \quad (6)$$

위 수식들의 의미는 공통된 평점 데이터의 수가 (e.g. $|U_i \cap U_j|$) 설계자가 선택한 상수 (e.g. δ) 보다 적으면 기존 Sim에 가중치(e.g. $\frac{|U_i\cap U_j|}{\delta}$)를 곱해 주어 sim의 크기를 낮춰준다.

2.4 단어 임베딩, 상품 임베딩

단어 임베딩 모델은 자연어 처리 분야에서 주로 쓰이는 모델로 콘텍스트-단어를 분석하여 단어를 공간상의 벡터로 표현한다[14]. Levy와 Goldberg는 이러한 모델이 점별 상호정보량 행렬(Pointwise mutual information matrix)를 분해한 결과와 같음을 보였다[15]. 점별 상호정보량인 Pointwise Mutual Information(PMI)는 다음과 같이 구해진다.

$$\mathrm{PMI}(x,y) = \log\frac{P(x,y)}{P(x)P(y)} \tag{7}$$

단어와 콘텍스트-단어 관계에서 식(7)는 x, y가 각각 단어와 콘텍스트 -단어에 대응된다. 또한 PMI는 경험적으로 다음과 같이 실제 관찰된 데이터의 개수로 구할 수 있다.

$$\mathrm{PMI}(x,y) = \log\frac{\#(x,y)|\mathrm{N}|}{\#(x)\#(y)} \tag{8}$$

식 (8)의 $\#(x,y)$ 는 x, y가 동시에 관찰된 개수이며 $\#(x)$, $\#(y)$는 각각 x와 y의 관찰된 개수이고 N은 전체 데이터의 개수이다. 위와 같이 구해진 PMI는 x와 y가 동시에 나타나는 경우가 없다면 그 값은 $-\infty$가 된다. 이는 PMI 행렬을 정의하기 힘들 뿐만 아니라, 행렬이 조밀해진다는 의미이다. 이런 문제를 해결하기 위해 양수(positive) PMI(PPMI)를 사용한다. 이는 다음과 같이 음수를 0으로 변경한 것이다[15].

$$\mathrm{PPMI}(x,y) = \max(\mathrm{PMI}(x,y), 0) \tag{9}$$

식 (9)는 의미 없는 음수를 0으로 치환함으로써 PPMI 행렬을 희소 행렬로 만들어 준다.

상품 임베딩은 단어 임베딩을 사용자-상품 관계 데이터에 적용한 것이다[12]. 사용자를 문서로 바라보고 상품을 단어로 바라본다면, 각 문서(사용자)에서 함께 나타나는(구매 또는 평점을 매긴) 단어(상품)들의 빈도수를 이용해 PPMI 행렬을 구성한다. 평점 데이터에서는 관찰된 평점을 대상으로 빈도수를 계산한다. 이렇게 사용자-상품 관계 데이터를 이용해 PPMI 행렬을 구성하면, 이는 곧 동시 발생 정보를 갖는 동시 발생 행렬이 된다.

2.5 공통 요소 모델

D. Ling et al. 은 앞서 설명한 상품 임베딩 모델을 CF에 적용한 공통 요소 모델을 제안하였다[12]. 이는 식 (8)을 이용하여 동시 발생 행렬을 만들었고 다음과 같은 손실 함수를 정의하여 모델을 구축하였다.

$$\begin{aligned}\mathrm{L} = &\sum_{u,i} \overbrace{c_{ui}(y_{ui} - \theta_u^T\beta_i)^2}^{\text{행렬 분해}} \\ &+ \sum_{m_{ij}\neq 0} \overbrace{\left(m_{ij} - \beta_i^T\gamma_j - w_i - c_j\right)^2}^{\text{상품 임베딩}} \\ &+\lambda_\theta \sum_u \|\theta_u\|_2^2 + \lambda_\beta \sum_i \|\beta_i\|_2^2 + \lambda_\gamma \sum_j \|\gamma_j\|_2^2\end{aligned} \tag{10}$$

식 (10)은 [4]에서 제안한 방법과 유사하게 최적화된다[12]. y_{ui}는 $r_{ui} \geq 4$인 경우 1이고, $r_{ui} < 4$인 경우 0으로 정의된다. 즉 y_{ui}높은 평점을 부여한 상품에 대한 구매 정보이다. c_{ui}는 스케일 모수로서 y_{ui}의 각기 다른 신뢰도를 판단하는 역할을 한다. θ_u^T는 사용자 요인이고 상품 요인인 β_i는 행렬 분해항과 상품 임베딩항에서 동시에 공유한다. 위 모델에서 행렬 분해항과 상품 임베딩항에서 나타나는 β_i는 항상 같은 상품의 잠재 요인을 의미한다. γ_j는 콘텍스트-상품을 나타낸다. w_i와 c_j는 편차이다.

3. 제안 모델

3.1 동 기

2.5절에서 본 바와 같이 D. Liang 등은 상품 동시 발생 정보를 모델에 포함 시킴으로써 정확도를 향상시켰다[12]. 그러나 [12]에서 제안한 방법은 다음과 같은 단점이 있다.

첫 번째는 데이터에서 3점 이하의 평점을 모두 제거했다는 것이다. 앞서 살펴본 바와 같이 평점 데이터는 그 희소 정도가 대단히 높다. 이러한 수많은 결측치는 바로 시스템의 성능에 악영향을 끼친다. 이런 특성을 갖는 평점 데이터에서 특정 평점을 0으로 변환하는 것은 더 많은 데이터의 손실을 야기하고 시스템의 성능을 떨어뜨린다.

두 번째는 남겨진 4점 이상의 평점을 모두 1로 바꾼 이

진행렬을 사용했다는 것이다. 이는 높은 평점을 받은 상품만을 고려하여 추천을 진행하겠다는 의미지만, 사용자가 직접 자신의 의견을 반영한 평점 정보가 완전히 사라지는 문제가 발생한다.

세 번째는 상품 동시 발생 정보의 도움을 받아 추천을 하지만 [12]에서 사용한 상품 동시 발생 행렬은 3점 이하의 평점이 제거된 이진 행렬에서 만들어졌다. 3.4절에서 살펴본 바와 같이 상품 동시 발생 행렬은 상품이 등장하는 빈도수를 기반으로 계산된다. 동시 발생 정보를 구하는 식 (9)를 보면 평점에 대한 정보를 전혀 사용하지 않는다는 것을 알 수 있다. 따라서 이미 제거된 평점 정보에 대한 보완 방법이 전혀 없고 이로 인해 정확한 사용자-상품 관계를 해석하는 것은 매우 힘들다.

본 논문에서는 이러한 문제를 개선하기 위해 손실이 없는 평점 데이터를 사용하고, 상호보완적인 상품 동시 발생 정보와 상품 PCC 유사도 정보를 함께 이용하는 모델을 제안한다. 이 모델의 장점은 다음과 같다.

첫째 변환된 이진 행렬이 아닌 평점 행렬을 그대로 사용한다. 사용자의 의견이 반영된 평점 데이터는 그 수치가 높고 낮음에 따라 중요도가 달라지지 않는다. 특히 희소 정도가 매우 높은 평점 행렬의 특성상 모든 데이터를 활용해야만 좋은 성능을 보일 수 있다.

둘째, 상품 동시 발생 행렬과 함께 PCC를 이용한 상품 유사도 행렬도 모델에 포함시킨다. PCC는 평점을 기반으로 상품과 상품 사이의 관계를 수치화 시킬 수 있다. 반면 상품 동시 발생 정보는 상품과 콘텍스트 상품의 관계를 이용해 성능을 향상시킨다.

동시 발생 정보를 이용하면 '이 상품을 구매하면 저 상품도 구매한다.' 라는 정보를 얻을 수 있고, PCC를 이용하면 '이 상품을 좋아하면 저 상품도 좋아한다. 이 상품을 싫어하면 저 상품도 싫어한다.' 등의 상품에 대한 선호도 정보를 얻을 수 있다.

3.2 SVD+CS

사용자-아이템 평점 행렬(R)은 사용자-아이템 간의 관계를 표현하고, 동시 발생 행렬(C)은 아이템과 콘텍스트 아이템 간의 관계를 표현한다. 그리고 유사도 행렬(S)은 유사도 공간상에서 아이템-아이템 간의 관계를 표현한다.

$$\hat{R} = P^T Q, \quad \hat{C} = Q^T W, \quad \hat{S} = Q^T Q$$

여기서 P는 사용자 요인 행렬이다. Q는 상품 요인 행렬이며, W는 콘텍스트 상품 요인 행렬이다. 위 세 개의 수식에서 Q는 모두 동일한 행렬을 나타낸다.

본 논문에서는 다음과 같이 여러 행렬을 조합한 손실함수를 제안한다.

$$\begin{aligned} L = & \sum_{(u,i)\in\mathcal{K}^{(R)}} \overbrace{\left(r_{ui} - \mu - b_u^{(r)} - b_i^{(r)} - p_u^T q_i\right)}^{\text{평점 행렬 분해}}{}^2 \\ & + \lambda_1\left(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2\right) \\ & + \sum_{(g,h)\in\mathcal{K}^{(C)}} \overbrace{\left(c_{gh} - b_g^{(c)} - b_h^{(c)} - q_g^T w_h\right)}^{\text{동시 발생 행렬 분해}}{}^2 \\ & + \lambda_2\left(\|q_g\|^2 + \|w_h\|^2\right) \\ & + \sum_{(y,z)\in\mathcal{K}^{(S)}} \overbrace{\left(s_{yz} - b_y^{(s)} - b_z^{(s)} - q_y^T q_z\right)}^{\text{유사도 행렬 분해}}{}^2 \\ & + \lambda_3\left(\|q_y\|^2 + \|q_z\|^2\right) \end{aligned} \tag{11}$$

식 (11)에서 $\mathcal{K}^{(*)}$는 해당 행렬에서 관찰된 데이터 쌍의 집합을 나타낸다. 평점 행렬의 요소인 r_{ui}는 사용자 u가 상품 i에 부여한 평점을 의미한다. 동시 발생 행렬의 요소인 c_{gh}는 상품 g와 상품 h가 함께 등장하는 정도를 의미한다. 유사도 행렬의 요소인 s_{yz}는 상품 y와 상품 z의 유사도를 의미한다. 동시 발생 행렬은 PPMI 계산식인 식 (8)과 (9)를 이용해 만든다. 유사도 행렬은 2.3절에서 식 (6)을 변형한 다음 수식을 이용해 만든다.

$$\text{sim``(i.j)} = \begin{cases} sim`(i.j) & sim`(i.j) > \alpha \\ 0 & sim`(i.j) \le \alpha \end{cases} \tag{11}$$

$sim`(i.j)$는 식 (8)를 나타낸다. 위 수식은 유사도가 α 보다 작으면 0으로 치환하여 유사한 경향을 보이지 않는 상품 쌍의 값을 제거한다.

2.5절의 식 (10)에서 행렬 분해항과 상품 임베딩항에서 나타나는 β_i는 항상 같은 상품의 잠재 요인을 의미한다. 하지만 본 논문에서 제안하는 식 (11)은 q_i, q_g, q_y, q_z 각각이 같은 행렬 Q를 공유하면서 서로 다른 상품 요인으로 작용한다. R 행렬의 편차는 벌점 항을 갖고 있지만 C 행렬과 S 행렬의 편차는 [12]에서와 마찬가지로 벌점을 부

과하지 않았다. $\lambda_1, \lambda_2, \lambda_3$는 벌점 계수로 경험적으로 결정된다. 여기서 q_i, q_g, q_y, q_z로 인해 업데이트 되는 Q는 사용자-상품 관계뿐만 아니라 상품-상품 동시 발생 정보, 상품-상품 유사도까지 설명할 수 있어야 한다.

동시 발생 행렬 C와 유사도 행렬 S를 하나씩만 적용한 모델도 생각해 볼 수 있다. SVD+C와 SVD+S는 SVD+CS에서 S와 C가 각각 빠진 것이다. SVD+C는 동시 발생 정보를 추가하여 사용자-상품 관계를 분석하고, SVD+S는 유사도 정보를 추가하여 관계를 분석한다.

3.3 최적화 알고리즘

모델의 파라미터 $\{p_u,\ q_i,\ b_u^{(r)},\ b_i^{(r)},\ q_g,\ w_h,\ b_g^{(c)},\ b_h^{(c)},\ q_y,\ q_z,\ b_y^{(s)},\ b_z^{(s)}\}$에 대해서 gradient를 구한다. 예를 들어 p_u와 q_i의 경우는 다음과 같다.

$$\frac{\partial L}{\partial p_u} = -2\left(r_{ui} - \mu - b_u^{(r)} - b_i^{(r)} - p_u^T q_i\right) q_i + 2\lambda_1 p_u$$

$$\frac{\partial L}{\partial q_i} = -2\left(r_{ui} - \mu - b_u^{(r)} - b_i^{(r)} - p_u^T q_i\right) p_u + 2\lambda_1 q_i$$

이와 같이 다른 파라미터에 대해서도 동일한 방법으로 gradient를 구한다.

식 (11)에서의 R 행렬과 C, S 행렬은 그 크기가 다르고 관찰된 데이터의 위치 또한 다르다. 평점 데이터 r_{ui}을 위한 행렬 분해항은 크기 l인 미니-배치 경사 하강법을 이용하고 S와 C를 위한 항은 무작위 추출된 데이터에 대해서 각각 크기 n, m인 미니-배치 경사 하강법을 이용한다.

앞서 설명한 gradient의 반대 방향으로 업데이트를 진행한다.

SVD+CS Algorithm

Initialize factor dimension: f
Initialize regularization : $\lambda_1, \lambda_2, \lambda_3$
Initialize each bias : $b_u^{(r)}, b_i^{(r)}, b_g^{(c)}, b_h^{(c)}, b_y^{(s)}, b_z^{(s)}$
Initialize factor matrix : $\mathrm{P} \in \mathbb{R}^{U \times f}$, $\mathrm{Q} \in \mathbb{R}^{I \times f}$, $\mathrm{W} \in \mathbb{R}^{I \times f}$
Initialize other variable : $l,\ n,\ m,\ \gamma$
While not converge do
  $\mathcal{K}^{(r)} \leftarrow$ size l sampling in $\mathcal{K}^{(R)}$
  Rand_$\mathcal{K}^{(C)} \leftarrow$ size n random sampling in $\mathcal{K}^{(C)}$
  Rand_$\mathcal{K}^{(S)} \leftarrow$ size m random sampling in $\mathcal{K}^{(S)}$

  $\mathrm{p_u} \leftarrow \mathrm{p_u} + \gamma \sum_{(u,i)\in\mathcal{K}^{(r)}}^{l} \left(e^{(r)}\mathrm{q_i} - \lambda_1 \mathrm{p_u}\right)$
  $\mathrm{q_i} \leftarrow \mathrm{q_i} + \gamma \sum_{(u,i)\in\mathcal{K}^{(r)}}^{l} (e^{(r)}\mathrm{p_u} - \lambda_1 \mathrm{q_i})$
  $\mathrm{b_u^{(r)}} \leftarrow \mathrm{b_u^{(r)}} + \gamma \sum_{(u,i)\in\mathcal{K}^{(r)}}^{l} (e^{(r)} - \lambda_1 \mathrm{b_u^{(r)}})$
  $\mathrm{b_i^{(r)}} \leftarrow \mathrm{b_i^{(r)}} + \gamma \sum_{(u,i)\in\mathcal{K}^{(r)}}^{l} (e^{(r)} - \lambda_1 \mathrm{b_i^{(r)}})$

  $\mathrm{q_g} \leftarrow \mathrm{q_g} + \gamma \sum_{(g,h)\in \mathrm{Rand}\_\mathcal{K}^{(C)}}^{n} (e^{(c)}\mathrm{w_h} - \lambda_2 \mathrm{q_g})$
  $\mathrm{w_h} \leftarrow \mathrm{w_h} + \gamma \sum_{(g,h)\in \mathrm{Rand}\_\mathcal{K}^{(C)}}^{n} (e^{(c)}\mathrm{q_g} - \lambda_2 \mathrm{w_h})$
  $\mathrm{b_g^{(c)}} \leftarrow \mathrm{b_g^{(c)}} + \gamma \sum_{(g,h)\in \mathrm{Rand}\_\mathcal{K}^{(C)}}^{n} (e^{(c)} - \lambda_2 \mathrm{b_g^{(c)}})$
  $\mathrm{b_h^{(c)}} \leftarrow \mathrm{b_h^{(c)}} + \gamma \sum_{(g,h)\in \mathrm{Rand}\_\mathcal{K}^{(C)}}^{n} (e^{(c)} - \lambda_2 \mathrm{b_h^{(c)}})$

  $\mathrm{q_y} \leftarrow \mathrm{q_y} + \gamma \sum_{(y,z)\in \mathrm{Rand}\ \mathcal{K}^{(S)}}^{m} (e^{(s)}\mathrm{q_z} - \lambda_3 \mathrm{q_y})$
  $\mathrm{q_z} \leftarrow \mathrm{q_z} + \gamma \sum_{(y,z)\in \mathrm{Rand}\_\mathcal{K}^{(S)}}^{m} (e^{(s)}\mathrm{q_y} - \lambda_3 \mathrm{q_z})$
  $\mathrm{b_y^{(s)}} \leftarrow \mathrm{b_y^{(s)}} + \gamma \sum_{(y,z)\in \mathrm{Rand}\_\mathcal{K}^{(S)}}^{m} (e^{(s)} - \lambda_3 \mathrm{b_y^{(s)}})$
  $\mathrm{b_z^{(s)}} \leftarrow \mathrm{b_z^{(s)}} + \gamma \sum_{(y,z)\in \mathrm{Rand}\_\mathcal{K}^{(S)}}^{m} (e^{(s)} - \lambda_3 \mathrm{b_z^{(s)}})$
End While

여기서 $e^{(*)}$는 각 행렬에 대한 예측 오차이다. 예를 들어 $e^{(r)} \stackrel{\text{def}}{=} \mathrm{r_{ui}} - \mu - \mathrm{b_u^{(r)}} - \mathrm{b_i^{(r)}} - \mathrm{p_u^T q_i}$이다.

q_i는 사용자-상품 관계를 사용자-상품 공간상에 표현된 벡터를 표현하고, q_g는 상품-콘텍스트 상품 관계를 상품-콘텍스트 상품 공간상에 표현된 벡터를 표현한다. 비슷하게 q_y, q_z는 유사도 공간상에 표현되는 상품 벡터를 표현한다. 이처럼 행렬 Q를 공유하며 동시에 여러 요인 공간에서 행렬 Q를 업데이트 함으로써 행렬 Q는 상품의 여러 관계(사용자, 콘텍스트, 유사도)를 동시에 해석할 수 있는 능력을 갖게 된다.

4. 실 험

4.1 데이터

실험에 사용된 데이터는 MovieLens 100k[16] 이다. 이는 추천 시스템의 성능을 평가하기 위해 자주 사용되는 데이터이다. 1997년 9월부터 1998년 4월까지 7개월간 수집된 데이터이며, 20개 미만의 평점을 부여한 사용자는 제거된 상태이다. 사용자 수는 943명이고 영화 수는 1682개이다. 평점은 1~5점 사이이며, 0은 아직 관찰되지 않은 데이터 이다. 관찰된 총 평점의 개수는 10만개이며, 희소정도가 6.3%로 관찰되지 않은 데이터가 훨씬 많다.

Test set은 사용자 별로 10%의 평점을 무작위로 선택하여 구성된다. Test set을 무작위로 선택하는 과정에서 Train set의 특정 영화 평점이 모두 Test set으로 넘어갈 수 있다. 이런 경우 해당 영화를 dataset에서 제거하였다. 최종적으로 사용자수는 943명, 영화 수는 1671편 이다.

4.2 평가측도

평가 측도로는 순위(ranking) 품질을 평가하는 측도인 Recall, Precision, NDCG를 사용 하였다. 이들 모두 각 사

용자 별로 추천결과의 순위와 실제 순위를 비교하는 방식으로 진행된다. 각 사용자 별로 Recall과 Precision은 다음과 같이 구해진다.

$$\text{Recall@k(u)} = \frac{|rel_u \cap rec_u|}{|rec_u|} \tag{13}$$

$$\text{Precision@k(u)} = \frac{|rel_u \cap rec_u|}{|rel_u|} \tag{14}$$

여기서 k는 추천된 영화 수 이며, rec_u는 사용자 u에게 추천된 영화이다. 또한 rec_u는 사용자 u와 관련된 영화로 추천결과의 실측 정보(ground truth)이다. 다음으로 DCG는 아래와 같이 구해진다.

$$\text{DCG@k(u)} = \sum_{i=1}^{K} \frac{2^{\mathbb{I}\{u(i)=1\}} - 1}{\log(i+1)} \tag{15}$$

여기서 $\mathbb{I}(\cdot)$은 지시함수(indicator function)이며, $u(i)$는 사용자 u가 영화 i에 평점을 부여한 경우 1이된다. NDCG는 DCG를 정규화 시킨 것으로 0과 1사이에의 값을 갖는다.

4.3 비교대상

본 논문에서 제안하는 방법은 기존 잠재 요인 모델(SVD)에 유사도 정보와 동시 발생 정보를 결합한 방법이다. 즉 유사도 정보와 동시 발생 정보에서 표현되는 상품 벡터를 이용해 사용자-상품관계 해석을 돕고자 하는 것이다. 따라서 본 논문에서 제안하는 모델의 기본이 되는 SVD를 비교 모델로 설정하였다. 또한 추가된 각 행렬의 영향을 알아 보기 위해 SVD+S와 SVD+C도 함께 성능을 비교하였다. SVD는 3.1절에서 설명한 내용과 같으며, SVD+S와 SVD+C은 SVD+CS에서 각각 동시 발생 정보(C)와 유사도 정보(S)가 빠진 것이다.

4.4 실험환경

Co-occurrence 행렬과 유사도 행렬은 python 2.7에서 구현되었고, SVD, SVD+S, SVD+C, SVD+CS 세가지 모델은 모두 Tensorflow를 이용하여 구현하였다. 모든 실험은 인텔 i7 6700, 32GB RAM 환경에서 수행하였다. 모델에서 설계자가 직접 설정해야 하는 파라미터는 총 8개로 실험적으로 표 1과 같이 설정하였다.

(표 1) 파라미터 설정
(Table 1) Parameter setting

파라미터		설정 값
잠재 요인 차원	f	60
Learning rete	γ	0.003
벌점 계수	λ_1	0.05
	λ_2	0.5
	λ_3	0.5
평점 임의 추출 크기	l	1000
S 임의 추출 크기	n	700
C 임의 추출 크기	m	1000
유사도 threshold	α	0.3

α는 유사도 한계 값으로 유사도 행렬에서 0.3 이하는 0으로 설정 하기 위해 사용한다. $\hat{C} = Q^T W$, $\hat{S} = Q^T Q$ 에서 Q가 특정 행렬로 빠르게 수렴해 들어가는 것을 막기 위해 S 행렬과 C 행렬의 임의 추출 크기를 각각 n=700, m=1000으로 하였다.

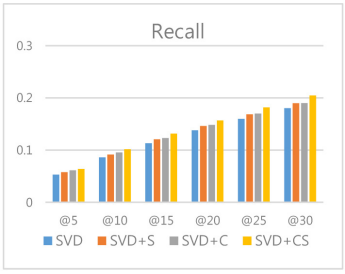
4.5 성능 평가

표 2는 총 10번의 실험으로 나온 10개 값에 대한 평균이다. @k는 4.2절에서 설명한 바와 같이 추천한 영화 수이다. 예를 들어 @5는 영화 5편을 추천한 뒤 정확도를 계산한 것이다.

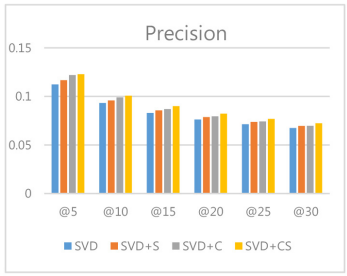
표 2와 그림 1,2,3에서 보이는 바와 같이 4가지 모델 중 SVD+CS 모델이 가장 좋은 성능을 보인다. SVD+S와 SVD+C를 살펴보자. SVD+S의 경우 유사도 행렬을 이용해 상품 요인의 표현에 도움을 줌으로서 SVD 모델 보다 좋은 성능을 보였다. 하지만 SVD+C 모델 보다는 성능이 떨어지는 모습을 보이는데, 다음과 같은 이유에서 발생한다. 먼저 PCC를 이용한 유사도는 평점 데이터와 같이 굉장히 희소 정도가 높은 데이터에서는 정확한 유사도를 구하기 힘들다는 것이다. 이는 전혀 다른 성향의 상품들이 높은 유사도를 가질 수 있고, 반대로 비슷한 성향을 갖는 상품이 낮은 유사도를 가질 수 있다는 의미이다. 식(6)과 (12)를 이용하여 이 문제를 어느 정도 감소 시켰지만 유사도를 기반으로 하는 CF 모델에서 여전히 해결해야 할 과제로 남아있다. 반면에 co-occurrence 행렬은 상품이 함께 나타나는 정도를 수치화 시킨 것으로 유사도 행렬에 비해 상대적으로 오차가 적어 SVD+S 보다 SVD+C가 좋은 성능을 보인다. 기본 비교대상 모델인 SVD와는 평균적으로 Recall은 약 16%, Precision과 NDCG는 각각 8%, 7% 상승하였다

(표 2) Recall, Precision, NDCG
(Table 2) Recall, Precision, NDCG

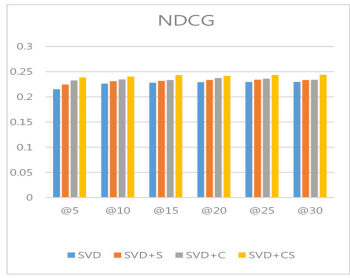
Recall						
	@5	@10	@15	@20	@25	@30
SVD	0.053	0.086	0.113	0.138	0.16	0.18
SVD+S	0.058	0.092	0.121	0.146	0.169	0.19
SVD+C	0.061	0.096	0.123	0.148	0.17	0.19
SVD+CS	0.064	0.102	0.132	0.157	0.182	0.205
Precision						
	@5	@10	@15	@20	@25	@30
SVD	0.112	0.093	0.083	0.076	0.071	0.068
SVD+S	0.117	0.096	0.086	0.079	0.074	0.07
SVD+C	0.122	0.099	0.087	0.08	0.074	0.07
SVD+CS	0.123	0.101	0.09	0.082	0.077	0.072
NDCG						
	@5	@10	@15	@20	@25	@30
SVD	0.215	0.226	0.228	0.229	0.23	0.23
SVD+S	0.225	0.231	0.232	0.234	0.234	0.234
SVD+C	0.233	0.235	0.233	0.237	0.236	0.234
SVD+CS	0.239	0.24	0.243	0.242	0.244	0.244



(그림 1) Recall
(Figure 1) Recall



(그림 2) Precision
(Figure 2) Precision



(그림 3) NDCG
(Figure 3) NDCG

S와 C가 모델에 결합된 SVD+CS는 가장 좋은 성능을 보였다. 이는 상품 요인을 다른 모델 보다 잠재 요인 공간에 잘 표현했기 대문이다. 동시 발생 행렬은 평점에 대한 정보를 놓친 반면 함께 등장하는 상품들의 정보를 갖고 있다. 유사도 행렬은 평점에 대한 정보가 유사도에 반영되었다. 이러한 두 행렬이 모델에 포함됨으로써 다른 방법들 보다 좋은 성능을 보였다.

위 모델을 과적합을 방지하는 정규화 모델로 설명할 수도 있다. 즉, 능형 회귀와 비슷하게 동시 발생 행렬 분해(상품 임베딩)항과 유사도 행렬 분해 항이 평점 행렬 분해 항의 정규화 또는 벌점 항으로 작용한다. 또한 이를 MAP(maximum a posterior)로 해석한다면 동시 발생 정보와 유사도 정보는 사전확률로 작용할 것이다. 이는 모델의 과적합을 방지하여 일반화 능력이 향상되었음을 의미한다.

참 고 문 헌

[1] W. Zhang, H. sun, X. Liu, and X. Guo, “Temporal qos-aware web service recommendation via non-negative tensor factorization”, In WWW, pp. 585~596, 2014. http://dl.acm.org/citation.cfm?id=2568001

[2] M. Deshpande, and G. Karypis. “Item-based top-n recommendation. ACM Transactions on Information Systems”, vol.22, No.1, pp.143~177, 2004. http://dl.acm.org/citation.cfm?id=963776

[3] X. Liu, C. Aggarwal, Y. Li, X. Kong, X. Sun, “Kernelized Matrix Factorization for Collaborative Filtering”, In Proc. SIAM Conference on Data Mining, pp 399~416, 2016. http://epubs.siam.org/doi/abs/10.1137/1.9781611974348.43

[4] Y. Hu, Y. Koren and C. Volinsky, ″Collaborative Filtering for Implicit Feedback Datasets″, 2008 Eighth IEEE International Conference on Data Mining, Pisa, pp. 263-272, 2008, http://ieeexplore.ieee.org/abstract/document/4781121/

[5] Y. Koren, “Factor in the Neighbors: Scalable and Knowledge Discovery from Data”, Vol. 4, No. 1, Article 1, Publication date: January 2010. http://dl.acm.org/citation.cfm?id=1644874

[6] X. Ning and G. Karypis, ″SLIM: Sparse Linear Methods for Top-N Recommender Systems,″ *2011 IEEE 11th International Conference on Data Mining*, pp. 497-506, 2011. http://ieeexplore.ieee.org/abstract/document/6137254/

[7] Y.Koren, R. Bell, and C. Volinsky. “Matrix factorization techniques for recommender systems”, IEEE Computer, 42, pp. 30~37, 2009. http://ieeexplore.ieee.org/abstract/document/5197422/

[8] T. Hofmann, “Latent Semantic Models for Collaborative Filtering”, *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 89 – 115, 2004. http://dl.acm.org/citation.cfm?id=963774

[9] S.Funk, “Netflix Update: Try This at Home”, Dec, 2006: http://sifter.org/~simon/journal/20061211.html

[10] Y. Koren, “Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model”, In Proc. of 14th SIGKDD Int’l Conf. Knowledge Discovery and Data Mining, pp.426~434, 2008. http://dl.acm.org/citation.cfm?id=1401944

[11] P. Cremonesi, Y. Koren, and R. Turrin, “Performance of Recommender Algorithms on Top-N Recommendation Tasks”, In Proc. of 4th ACM Conference on Recommender Systems, pp. 39~46, 2010. http://dl.acm.org/citation.cfm?id=1864721

[12] D. Liang, J. Altosaar, L. Charlin, and D. Beli, “Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-occurrence”, In proc. of the 10th ACM Conference on Recommender Systems, pp 59-66, 2016. http://dl.acm.org/citation.cfm?id=2959182

[13] Hao Ma, Irwin King, and Michael R. Lyu, “Effective missing Data Prediction for Collaborative Filtering”, In Proc. of 30th ACM SIGIR’07, pp 39~46, 2007. http://dl.acm.org/citation.cfm?id=1277751

[14] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean “Distributed Representations of Words and Phrases and their Compositionality”, In Advances in Neural Information Processing Systems, NIPS, pp 3111~3119. 2013. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality

[15] O. Levy, Y. Goldberg, “Neural Word Embedding as Implicit Matrix Factorization”, In Advances in Neural Information Processing Systems, NIPS, pp 2177 – 2185, 2014. http://dl.acm.org/citation.cfm?id=2969070

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”, In proc, of ACM CSCW, pp 175~186, 1994. http://dl.acm.org/citation.cfm?id=192905

◐ 저 자 소 개 ◑

나 광 택

2013년 인하대학교 토목공학과 졸업(학사)
2015년~현재 인하대학교 대학원 컴퓨터공학과 재학(석사)
관심분야 : 머신러닝, 데이터 마이닝, 시계열 분석, 추천시스템
E-mail : kwangteakna@gmail.com

이 주 홍

1983년 서울대학교 전자계산기공학 졸업(학사)
1985년 서울대학교 컴퓨터공학 졸업(석사)
2001년 한국과학기술원 컴퓨터공학 졸업(박사)
2001년~현재 인하대학교 공과대학 컴퓨터공학과 교수
관심분야 : 머신러닝, 데이터 마이닝, 시계열 분석
E-mail : juhong@inha.ac.kr