Collaborative Filtering for Implicit Feedback Datasets

발표일	
Keyword	Collaborative Filtering
Youtube	
논문 링크	http://yifanhu.net/PUB/cf.pdf
발표자료	

논문 소개

기존의 추천 시스템은 크게 두 가지의 방식을 사용한다.

1. Content based approach
   사용자와 상품에 각각 external information(=metadata)를 추가하여, 이를 바탕으로 추천하는 방식이다. 예를들어 영화에는 장르, 출연 배우 등의 정보를 기록할 수 있고, 사용자에게는 나이, 성별 등의 정보를 기록할 수있다. 하지만 이러한 방식은 부가 정보를 수집할 수 있을 때에만 제한적으로 사용될 수 있다.
2. Collaborative Filtering
   사용자가 남긴 기록(검색 기록, 별점)을 바탕으로 사용자와 상품의 관계를 분석하는 방식이다. Content based approach에 비해 더 높은 성능을 보이지만, 서비스가 새롭게 시작될 때에는 이러한 정보가 부족하므로 cold start 문제가 발생할 수 있다.

Collaborative Filtering에서 참고하는 데이터의 종류는 두 가지로 나뉠 수 있다.

1. Explicit Feedback
   사용자가 어떤 상품에 남긴 별점, 좋아요/싫어요, 리뷰 등을 말한다. 특히 별점이나 좋아요/싫어요의 경우, 점수의 범위가 정해져 있고, 선호도에 대한 분명한 자료로 활용할 수 있다. 하지만, 모든 사용자가 적극적인 평가에 참여하는 것이 아닌 점, 상황에 따라 이러한 피드백을 수집하지 못하는 경우 이용하기 어렵다는 한계점이 있다.
2. Implicit Feedback
   사용자의 구매 기록, 검색 기록, 마우스 움직임 등을 말한다. 이러한 정보는 제공 동의가 이루어졌을 때 explicit feedback 보다 더 많은 정보를 수집할 수 있다. 또한 explicit feedback을 수집할 수 없는 상황에서도 implicit feedback을 수집할 수 있다는 장점이 있다.

주요 아이디어와 제안한 모델

💡 논문의 주요 아이디어와 논문에서 제안한 모델에 대한 설명

> *논문에서는 explicit feedback에 비해 더 많은 정보가 잠재되어 있는 implicit feedback을 중심으로, 데이터 크기에 따라 linearly scalable 하면서도 높은 성능을 가지는 모델을 제안했다. 그 방법으로 관찰된 데이터를 preference와 confidence로 나누어 수식화하였고, alternating least square로 효율적으로 sparse matrix를 최적화하는 방법을 제안했다. 이 모델은 latent factor model 이면서도 설명 가능한 예측을 수행한다는 장점이 있다.*

*Implicit Feedback*의 특징

논문에서는 아래 네 가지의 *implicit feedback*의 특징을 고려하여 모델을 설계하였다.

1. *Explicit feedback*에는 **낮은 별점**, **싫어요** 등의 불호의 정보가 나타나지만, *implicit feedback*에서는 missing data에 불호 정보가 잠재되어 있다. 따라서 *explicit feedback*에서 조사된 데이터로만 추론을 하는 것과 달리 조사되지 않은 missing data를 포함하여 추론해야 한다.
2. *Implicit feedback*에는 노이즈가 포함되어 있다. 사용자가 어떤 상품을 이용했다는 것이 꼭 상품에 대해 긍정적이라는 것은 아니고, 반대로 어떤 상품을 이용하지 않은 데에는 단순히 그 상품이 싫어서가 아닌 여러 가지 이유가 있을 수 있다는 것이다.
3. *Explicit feedback*에서의 수치는 곧 *preference*를 나타내지만, *Implicit feedback*에서의 수치는 곧 *confidence*를 나타낸다. *Implicit feedback*에서의 수치는 빈도를 나타내기 때문에 이것을 *preference*로 보기는 어렵지만, 높은 빈도는 사용자의 의견을 반영할 가능성이 있다.
4. *Explicit feedback* 기반의 시스템은 MSE와 같은 뚜렷한 지표가 있지만, *Implicit feedback* 기반의 추천 시스템을 평가할 때에는 *explicit feedback* 기반의 시스템에 비해 좀 더 많은 것을 고려해야 한다.

제안한 모델

먼저 사용자 u의 상품 i에 대한 preference p_{ui}는 이진 변수로 다음과 같이 설정했다.

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases}$$

즉, 사용자가 잠깐이라도 어떤 상품을 소비했다면 *선호*한다고 보고, 한 번이라도 소비하지 않으면 *선호하지 않는 것(선호한다고 볼만한 어떤 행동도 하지 않은 것)*으로 보았다. 그리고 이 선호의 강도를 confidence c_{ui}로 표현했다.

$$c_{ui} \ = \ 1 \ + \ \alpha r_{ui}$$

따라서 모든 사용자-상품 쌍에 대해 최소한의 confidence를 가지고 있고, raw observation r_{ui}가 커질수록 선형적으로 증가한다.

제안된 논문의 목표는 사용자 u의 상품 i에 대한 preference $p_{ui} = x_u^T y_i$를 계산하는 user-factor x_u와 item-factor y_i를 찾는 것이다. x_u와 y_i는 cost function을 L_2 *Regularization*이 포함된 *Least Square Method*로 구해진다.

$$\min_{x*,\,y*} \sum_{u,\,i} c_{ui}(p_{ui} - x_u^T y_i)^2 \ + \ \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

이 cost function은 (사용자의 수 m × 상품의 수 n)개의 변수가 포함되므로 Stochastic Gradient Descent와 같은 방법을 쓰기에는 연산량이 너무 크다(일반적인 데이터 셋에서는 $m \cdot n$이 수 십억에 이른다). 그래서 논문에서는 이를 업데이트하는 다른 최적화 기법을 제안한다.

Alternating-least-squares 최적화

Cost function에서 user-factor 또는 item-factor 중 하나를 고정하고 나머지 factor에 대해 최적화하면, 고정된 factor에 대한 cost function의 global minimum을 구할 수 있다. 그리고 고정되는 요소를 바꾸어가면서 반복 수행하면 각 step마다 cost function의 값을 강하 시킬 수 있다. 이 방법으로 user-factor, item-factor를 고정하여 각각 미분하게 되면 다음의 solution을 얻는다.

$$x_u = \left(Y^T C^u Y + \lambda I\right)^{-1} Y^T C^u p(u) \tag{1}$$

$$y_i = \left(X^T C^i X + \lambda I\right)^{-1} X^T C^i p(i) \tag{2}$$

이를 계산하는 데에는 각각 $O\left(f^2 \mathcal{N} + f^3 m\right) \sim O(m)$, $O\left(f^2 \mathcal{N} + f^3 n\right) \sim O(n)$ 시간이 든다. 따라서 cost function을 최적화 하는 과정이 linearly scaleable함을 알 수 있다.

상술한 방식으로 모델을 설계함으로, 논문에서 정의한 *implicit feedback*의 특성(#1~#3)을 잘 반영하였다. 먼저 raw observation r_{ui}를 이용하여 preference p_{ui}, confidence c_{ui}를 정의함으로 #2, #3의 특징을 설명하였고, linearly scalable한 최적화 방법을 도입함으로 #1에서의 missing data를 함께 처리할 수 있게 되었다.

모델의 설명가능성

Latent factor approach의 경우 사용자의 과거 행동이 factor 안에 잠재되어 있기 때문에, 과거 행동과 추천 결과의 직접적인 관계를 설명하기 어려웠다. 하지만 논문에서 제안한 모델은 (2)의 식을 활용하여 예측을 설명할 수 있다. 사용자 *u*의 상품 *i*에 대한 예측 preference $\hat{p}_{ui} = y_i^T x_u$는 다음과 같이 표현된다.

$$\begin{aligned}\text{Denote that } f \times f \text{ matrix } \left(Y^T C^u Y + \lambda I\right)^{-1} \text{ as } W^u, \text{ then} \\ \hat{p}_{ui} &= y_i^T x_u \\ &= y_i^T \left(Y^T C^u Y + \lambda I\right)^{-1} Y^T C^u p(u)(3) \\ &= y_i^T W^u Y^T C^u p(u)\end{aligned}$$

이 때 사용자의 관점에서 전체 상품 i와 사용자가 소비한 상품 j의 weighted similarity가 $s_{ij} = y_i W^u y_j$일 때, 예측 preference $\hat{p}_{ui}$는,

$$\hat{p}_{ui} = \sum_{j:r_{ui}>0} s_{ij}^u c_{uj} \tag{4}$$

으로 표현할 수 있다. 즉, 상품 간의 유사도와 사용자의 과거 행동으로 예측 preference를 설명할 수 있다.

기존 추천 모델과의 비교

기존 추천 모델(논문 3. Previous Work에 소개된 모델에 한정) 대비하여 어떤 상황에서 어떤 강점을 갖는지

> *Collaborative Filtering Approach에서 주로 사용하는 모델은 크게 Neighborhood Model, Latent Factor Model이 있다. 논문의 모델은, neighborhood model보다 우수한 추천 성능을 보이고, 기존의 latent factor model이 explicit feedback에서만 활용된 것과 달리 implicit feedback을 효율적으로 활용하고 있다.*

Neighborhood Models

이전의 Neighborhood 기반의 모델은 *explicit feedback*, 그중에서도 *상품에 대한 평가의 유사도*를 비교하여 비슷한 상품을 추천해 주는 *item-oriented approach*가 주로 사용되었다. 해당 연구에서는 사용자들이 상품에 매기는 평가 경향(어떤 사람은 1~5점에서 다양하게 점수를 매기지만, 어떤 사람은 큰 불만이 없으면 5점을 주는 경우)들을 조정하면서 실제 적용되었다. 하지만 이러한 방법은 implicit feedback에서는 적용하기 힘들다. 사용자들의 행동 패턴은 더욱 정규화하기 까다롭고, 따라서 유사도를 분석하기 힘들기 때문이다.

또한 item-oriented 모델에 implicit feedback을 적용하였을 때, 사용자의 preference와 confidence를 구별하기 힘들다는 단점 또한 존재한다.

Latent Factor Models

이전의 Latent Factor 기반의 모델 또한 *explicit feedback*을 바탕으로 설계되었다. 이러한 모델들은 observed data에 한정되어 user-factors 벡터 x_u와 item-factors 벡터 y_i를 추정하여 평가를 예측한다. 하지만 기존의 latent factor Model은 모델이 예측한 결과를 설명하기가 힘들다. 이에 더해 implicit feedback에서 관찰되지 않은 missing data 또한 함께 활용해야 한다는 측면에서, 기존 모델들이 채택한 stocastic gradient descent 방식의 최적화 방식을 적용하기에는 데이터의 크기가 크다.

위 두 가지의 기존 모델을 종합적으로 평가했을 때, 기존의 연구들은 더 높은 잠재력을 가진 implicit feedback에 적용하기 힘든 부분이 있다. 논문에서 제안한 모델은 *implicit feedback을 scalable하게 처리하는 latent factor model*을 제시하였고, 해당 방식의 단점인 *설명 가능성이 낮다는 점*에서도 대안을 제시하였다.

실험 방법

💡 제안한 모델의 우수함을 어떤 실험을 통해 어떻게 증명하였는지

> *논문의 실험에서는 먼저 추천 시스템이 특정한 상황에 편향되는 것을 방지하기 위한 전처리 작업을 수행하였고, 시스템 평가에 적절한 평가 지표를 제안했다. 이를 통해 인기순 추천 방식과 neighborhood model에 비해 우수한 성능을 보임을 밝혔고, 설명 가능성에 대해서도 검증하였다.*

실험 조건과 데이터 전처리

실험에서 사용된 학습 데이터는 약 30만 개의 셋톱박스를 대상으로 4주 동안의 시청 기록을 수집한 데이터이고, 평가 데이터는 한 주간의 시청 기록이다. 4주 동안 약 17,000개의 서로 다른 TV 프로그램이 방송되었다. 총 약 51억 개의 r_{ui}값 중 0이 아닌 데이터는 3200만 개다.

사용자들은 보통 지난주에 봤던 TV 프로그램을 그다음 주에도 다시 시청하는 경향이 있기 때문에, 평가 데이터에서는 정확한 평가를 위해 이러한 데이터들은 제거하였다. 또한 어떤 프로그램을 절반 미만으로 시청한 경우, 즉 $r_{ui} < 0.5$인 데이터들은 0으로 처리하였다. 전처리를 실시한 뒤 평가 데이터의 0이 아닌 값은 약 200만 개였다.

또한 같은 프로그램을 반복적으로 시청하는 현상 등에 페널티를 주고자 confidence를 계산할 때 r_{ui}에 log-scale을 적용하였고, 어떤 프로그램을 시청하고 채널을 바꾸지 않은 채 연속해서 다음 프로그램을 시청하는 경우 실제로 시청을 하지 않는다고 보고 이 역시 페널티를 주었다.

평가 지표는 사용자들이 추천 리스트에서 고른 프로그램이 리스트에서 상위 몇 퍼센트에 있을지에 대한 기댓값 $\overline{rank}_{ui}$으로 하였다.

비교 모델로는 프로그램 인기도에 따른 추천 리스트, item-oriented 기반의 neighborhood approach로 산출한 추천 리스트를 사용하였다.

실험 결과

실험 결과, 대조군인 두 모델에 비해 논문에서 제안한 모델이 더 좋은 성능을 보였다. 특히, latent factor를 늘릴수록 지표가 더 좋아짐을 보였다. 이는 대조군의 두 모델의 경우 성능을 더 높일만한 요소가 없지만, 제안한 모델은 컴퓨팅 성능에 비례하여 성능 개선을 기대하볼 수 있다는 점에서 우수하다.

$rank_{ui}$의 누적 분포 함수로 두 모델과 비교하였을 때에도 우수한 결과를 보였다. 특히, 제안한 모델의 추천 결과와 함께 이전에 시청했던 프로그램도 함께 표시하였을 때 더 좋은 성능을 보였다.

논문에서 제안한 raw observation r_{ui}를 preference p_{ui}와 confidence c_{ui}의 두 가지 측면으로 분해하는 방법에 대해서도 검증하였다. 그 방법으로 최적화의 target으로 r_{ui}와 p_{ui}를 사용하여 실험한 결과 full model에 뒤지는 성능을 보였다.

마지막으로 제안한 모델의 설명 가능성에 대해서도 검증하였다. 사용자가 가장 많이 시청한 상위 5개의 TV 프로그램을 바탕으로 추천한 결과는 그와 유사한 장르였다. 이는 추천 성능 면에서도 neighborhood approach 보다 우수하면서도 설명 가능한 결과를 표시한다는 점에서 의의가 있고, latent factor model 연구에서 처음으로 설명 가능성에 대해 검증했다는 점에서도 의의가 있다.