도서 추천을 위한 분산 Heterogeneous Restricted Boltzmann Machines

이은진 김소현 이기쁨 이서라 조동섭
이화여자대학교 공과대학 컴퓨터공학과

Distributed Heterogeneous Restricted Boltzmann Machines for Book Recommendation

Eunjin Lee Sohyun Kim Gibbeum Lee Seora Lee Dongsub Cho
Department of Computer Science & Engineering, Ewha Womans University

Abstract - 최근 콘텐츠의 규모가 방대해지면서 원하는 콘텐츠를 더 쉽게 찾을 수 있도록 도와주는 추천 시스템의 중요도가 매우 높아졌다. 따라서 데이터를 군집화하고 각 그룹에 추가적으로 학습된 RBM을 사용해 특화된 추천을 하면서 추천의 성능을 높이는 Heratogeneous RBM(Restricted Boltzmann Machine)을 제안하고자 한다. Heterogeneous RBM의 학습방법과 Heterogeneous RBM(Restricted Boltzmann Machine)의 출력에 기초하여 최종 출력을 생성하는 M-selection, M-Weighted, M-Ensemble의 세 가지 출력 통합 방법을 제안하고 모델의 성능을 검증하기 위해 각기 다른 출력 통합 방법을 사용하는 모델의 비교, 여러 추천 시스템과의 성능 비교, 단일 RBM과의 비교를 진행했다. 이를 통해 M-Ensemble은 다른 두 출력 통합 방법보다 Hit Rate, ARHR, 실행시간 면에서 높은 성능을 보이며, 기존의 추천 시스템보다 개선된 성능을 보이는 것을 알 수 있다.

1. 서 론

오늘날 전 세계에서 이용 가능한 콘텐츠는 매우 방대하며, 콘텐츠를 소비하는 사용자의 수 또한 폭발적으로 증가하고 있다. 이에 따라 사용자가 자신이 원하는 콘텐츠를 찾기 위해서는 많은 시간과 노력이 소요된다. 추천 시스템은 이러한 문제점을 극복하고 실용적인 검색 서비스를 제공하는 데 도움을 줄 수 있다.

본 논문에서는 협업 필터링(Collaborative Filtering)에 사용할 수 있는 강력한 머신러닝 모델인 RBM(Restricted Boltzmann Machine)을 사용하여, 단일 RBM 모델 또는 다른 기본적인 추천 시스템들에 비해 전체 성능을 향상시킬 수 있는 새로운 Heterogeneous RBM 모델 구조를 제안한다. 제안된 모델은 여러 RBM 모델에 훈련 데이터를 분배하고 여러 결과를 출력한다. 이렇게 분산된 RBM의 출력은 제시하는 세 개의 방법 중 한 가지를 통해 통합되어 최종 추천을 만들어낸다. 실험을 통해 각 통합 방법의 성능을 비교하여 제안된 방법 중 어느 방식의 성능이 가장 좋은지 확인하고 다른 추천 시스템과도 비교하여 제시하는 모델이 높은 성능을 가지고 있는 것을 확인한다.

본 논문은 제 3.1장에서 분산 데이터 학습을 이용한 Heterogeneous RBM 프레임워크와 여러 모델의 결과를 취합하는 세 가지 방법을 설명하고, 제 3.2장에서 실험과 경험적 연구를 통해 얻은 결과를 다양한 추천 알고리즘과 비교하여 제안된 모델의 성능에 관하여 기술한다.

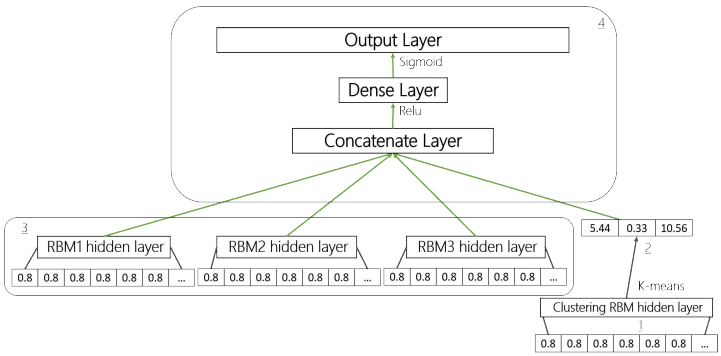
2. 관 련 연 구

추천 시스템은 오래전부터 연구가 꾸준히 진행되어온 분야로 크게 사용자와 아이템 간의 상관관계를 고려하는 협업 필터링(Collaborative Filtering) 방법과 사용자가 선호하는 아이템의 고유 특성을 고려하는 내용기반 추천(Content-Based) 방법으로 나누어진다. 더 자세히 설명하면 협업 필터링 방식은 사용자들의 아이템 평가 데이터 속 비슷한 선호 형태를 보이는 사용자들을 같이 고려해 새롭고 색다른 추천을 하는 방법이고 내용기반 추천은 사용자가 지금까지 선호한 아이템들의 특성을 찾아 가장 비슷한 아이템들을 미래에도 추천해주는 방법이다[1]. 이 연구는 협업 필터링을 사용해 추천을 진행한다.

주로 사용되는 협업 필터링 방식에는 행렬 분해법과 Restricted Boltzmann Machine(RBM)이 있다. 행렬 분해법은 사용자와 아이템 사이의 평가 데이터가 행렬 구조로 나타나 있다면 해당 행렬을 더 작은 차원의 행렬들로 분해해 각 사용자와 아이템에 해당하는 벡터를 얻고 내적을 통해 둘 사이의 관계 값을 얻는 방식이고[2], RBM은 행렬 분해법과 다르게 신경망을 통해 더 작은 차원의 은닉 벡터에서 입력을 재구성하면서 협업 필터링을 수행하는 방식이다[3]. RBM은 신경망을 사용하기 때문에 행렬 분해법보다 더 큰 크기의 데이터세트에도 좋은 성능을 보인다고 알려져 있다. 이 연구는 협업 필터링 방법 중에서도 RBM에 집중해 분산구조를 제안한다.

3. 본 론

3.1 분산 Heterogeneous RBM 모델



<그림 1. 분산 Heterogeneous RBM 모델 구조>

3.1.1 분산 RBM 학습 과정

RBM은 두 개의 신경망 레이어를 통해 입력벡터를 은닉벡터로 보낸 후, 다시 입력벡터를 재구성하면서 협업 필터링을 실행하는 모델이다[1]. 입력으로는 사용자의 평가벡터를 받고, 출력이 입력을 얼마나 정확히 재구성하는지 고려하는 비지도 학습을 통해 모델을 학습한다. 사용자의 평가벡터는 시스템에 아이템이 N개가 있을 때, N 길이의 벡터이고 n번째 노드의 값은 n번째 아이템에 대한 해당 사용자의 평가값을 [0,1]의 범위로 정규화한 값이다. 아이템이 사용자에게 평가되지 않았을 경우 그 값은 0으로 대체한다. RBM을 통해 입력벡터가 재구성되면서 사용자의 입력 벡터 중 평가가 없는 아이템에 대해서도 평가값이 채워져 재구성되기 때문에 이 예측값들을 이용해 추천을 수행할 수 있다. RBM의 은닉벡터는 모든 사용자 데이터에 대해 더 낮은 차원의 표현을 얻고 이를 종합하기 때문에 은닉벡터에서 구성한 출력은 모든 사용자와 아이템과의 관계를 고려한 추천, 즉 협업 필터링이다.

이런 RBM에 데이터를 분산시켜 각 데이터에 특화된 모델들을 얻는 분산 RBM의 학습 과정은 다음과 같다.

step 1. RBM을 사용자가 M, 아이템이 N개인 MXN 크기의 데이터를 사용해 학습시킨다. 결과로는 은닉벡터를 이용해 각 사용자에 대한 낮은 차원의 표현을 얻을 수 있다. 이 모델은 clustRBM이라 칭한다.

step 2. step 1의 은닉벡터를 이용해 사용자를 군집화해 데이터를 K개의 그룹으로 나눈다. 군집 방법은 K-means를 사용하고 K로는 3을 사용한다.

step 3. 미리 학습된 RBM을 step 2에서 군집화한 데이터의 각 그룹에 추가로 학습해 각 그룹에 특화된 K개의 RBM 모델을 얻는다. 이 모델들은 heteroRBM이라 칭한다.

이후 분산 RBM의 출력들을 통합해서 추천을 진행한다.

3.1.2 출력 통합 방법

각 heteroRBM은 입력을 재구성하며 계산된 다양한 추천 결과를 출력한다. 따라서, 본 논문에서는 heteroRBM의 출력에 기초하여 최종 출력을 생성하는 3가지 출력 통합 방법을 제안한다.

첫 번째 방법은 clustRBM의 은닉벡터에 위에서 학습시킨 K-means를 적용해 군집그룹의 레이블을 얻고 해당 군집그룹에 추가로 학습되었던 heteroRBM의 출력만을 이용하는 방법(M-Selection)이다. 두 번째 방법은 입력에 대한 clustRBM의 은닉벡터와 K-means의 centroid와의 거리를 각 heteroRBM모델 출력의 가중치로 보아 각 heteroRBM의 출력에 가중치가 적용된 평균을 계산하는 방법(M-Weighted)이다. 마지막 방법은 딥러닝을 이용한 앙상블 모델링(M-Ensemble)이다. 앙상블 모델링이란 여러 개의 모델을 학습시켜 그 모델들의 예측 결과들을 이용해 하나의 모델보다 더 나은 값을 예측하는 방법을 말한다[2]. 이 모델에서는 심층 신경망 연결을 이용해 분산 모델의 은닉벡터를 취합한 입력에 clustRBM의 은닉벡터와 K-means centroid 사이의 거리 벡터를 추가해 더 정확한 추천 벡터를 얻는다. 출력으로는 사용자의 아이템 평가 입력벡터에 사용자가 다음으로 좋아한 아이템의 평가가 추가된 벡터를 주어 교차 엔트로피 손실함수를 사용해 학습을 진행한다.

3.2 구현 및 결과 분석

3.2.1 구현 환경

구현에 사용된 RBM 모델은 빠른 속도를 장점으로 하는 python의 머신러닝 라이브러리인 PyTorch를 사용하여 구현되었다. 출력 통합을 위한 앙상블 모델은 Python의 신경망 라이브러리인 Keras로 구현되었다. RBM 모델의 K-Means의 구현에는 python 기계학습 라이브러리인 Scikit-learn에서 제공하는 모듈이 사용되었다. 실험을 위한 Ensemble 모델은 Adam Optimizer를 사용하여 최적화하였다[3].

구현을 위한 데이터세트는 온라인 서점 사이트 알라딘(aladin.co.kr)에서 추출하여 사용한다. 데이터 추출에는 python 웹 크롤링 프레임워크인 Scrapy를 사용해 책의 이름, 작가, 출판사 등의 기본적인 책 정보와 사용자의 도서 평가 정보를 추출하였다. 추출한 데이터에 대하여 후기가 2개 미만인 사용자 및 도서 정보를 제외하고, 작가, 장르, 출판사 정보를 추가 항목으로 고려하고, id를 재할당하는 데이터 전처리 과정을 진행하였다.

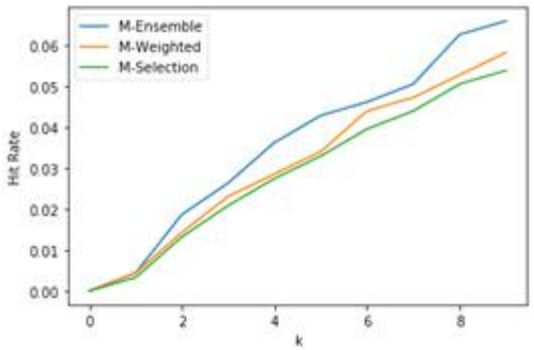
대게 추천 시스템의 성능 평가 지표로는 사용자가 마지막으로 평가한 책의 데이터를 제외하고 훈련한 후 추천 결과에 그 책이 포함되는가를 확인하는 방법을 사용한다[4]. 그리하여 학습데이터는 M-Selection과 M-Weighted에 대해선 20%의 사용자의 마지막으로 평가한 아이템을 제외한 모든 평가 데이터를 사용하고 M-Ensemble에서는 위 학습데이터에서 마지막 아이템 평가를 빼놓은 데이터를 RBM 학습에 사용, 위 학습데이터를 앙상블 모델의 학습데이터로 사용한다. 모든 모델에서 시험데이터는 20%의 사용자의 마지막으로 평가한 아이템 데이터이다. 성능의 평가는 Hit Rate 함수와 ARHR(Average Reciprocal Hit Ranking) 함수를 사용한다. Hit Rate는 추천 알고리즘의 출력이 사용자가 다음 좋아할 책을 Top N 위안에 가지고 있었는지 계산해 추천 알고리즘의 성능을 테스트하는 방법이며, ARHR은 해당 책이 얼마나 높게 평가되었는지 고려하는 성능 분석 알고리즘이다[5].

3.2.2 출력 통합 방법의 비교

위에서 제시하는 분산 RBM의 결과를 통합하는 3가지 방법, ①하나의 RBM 출력을 선택하는 **M-Selection**, ②가중 집적을 사용하는 **M-Weighted**, 그리고 ③딥러닝 앙상블 네트워크를 이용하는 **M-Ensemble** 중 가장 성능이 좋은 방법을 도출하기 위한 실험을 진행했다.

<표 1. 각 출력 통합 방법의 성능>

	M-Ensemble	M-Weighted	M-Selection
HR@10	**0.0748899**	0.0627753	0.0594714
HR@25	**0.123348**	0.11674	0.120044
ARHR	**0.00434233**	0.00354124	0.0037859
Time	**0.523004**	1.004	1.03001



<그림 2. 각 출력 통합 방법의 Hit Rate 비교>

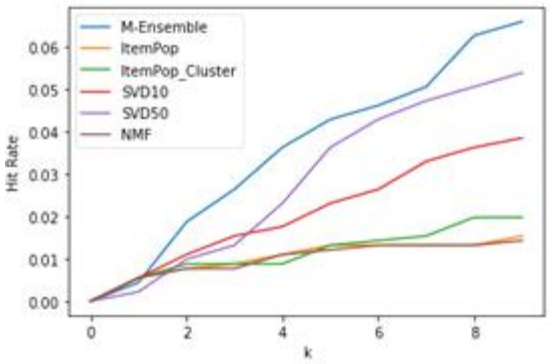
실험 결과 M-Ensemble 모델이 다른 두 개의 모델보다 Hit Rate, ARHR, 실행시간 면에서 높은 성능을 보였다.

3.2.3 성능 비교

<표 2>와 <그림 3>은 여러 추천 시스템들과 Multi-RBM 앙상블 모델의 성능 비교 결과를 보여준다. 실험을 위한 비교 대상으로는 ①제일 인기가 많은 아이템부터 추천하는 Itempop, ②각 군집그룹에서 인기가 많은 아이템을 추천하는 Itempop-Cluster, ③행렬 분해법을 이용해 협업 필터링을 수행하는 SVD와 ④NMF를 사용했다. 각 항목의 왼쪽칸의 값은 N으로 10과 25를 갖는 Hit Rate와 ARHR를 나타내고, 오른쪽 칸의 값은 각 추천 시스템들에 비해 M-Ensemble 모델이 얼마나 더 좋은 성능을 보였는가(%)를 나타낸다. 결과를 보면 HR, ARHR, 시간 모든 방면에서 M-Ensemble의 성능이 좋은 것으로 나타난다. 특히 가장 작은 성능 향상을 보인 SVD와 M-Ensemble의 HR@25의 비교에서도 M-Ensemble이 13.13%나 높게 나오는 것으로 보아 제시하는 모델이 기본적인 추천시스템과 비해 좋은 추천을 해주는 것으로 결론지을 수 있다.

<표 2. 기존 추천 시스템과 M-Ensemble 모델 모델의 성능 비교>

	M-Ensemble	ItemPop		ItemPop-Cluster		SVD		NMF	
HR@10	**0.0748899**	0.0187225	300(%)	0.0209251	257.9	0.0561674	33.33	0.0143172	423.08
HR@25	**0.123348**	0.0352423	250	0.0429515	187.18	0.109031	13.13	0.034141	261.29
ARHR	**0.00434233**	0.000732536	492.78	0.00118582	266.19	0.00322322	34.72	0.000867051	400.82
Time	**0.605998**	0.0810008		0.865004		7.176		11.43	



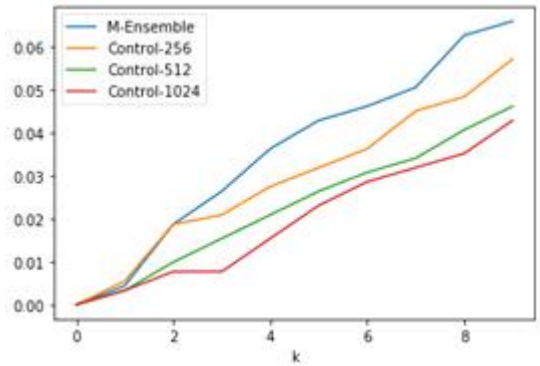
<그림 3. 기존 추천 시스템과 M-Ensemble 모델의 Hit Rate 비교>

3.2.4 분산 RBM의 효과

분산 RBM을 사용하는 것이 단일 RBM만을 사용하는 것보다 효과가 있는지 비교하기 위해 M-Ensemble 모델과 여러 은닉 노드 수를 갖는 단일 RBM과의 추천 성능을 비교해 보았다. 결과적으로 분산 RBM의 수행 속도는 느리지만 여느 단일 RBM보다 추천 성능이 높다는 것을 볼 수 있었다.

<표 3. M-Ensemble 모델과 단일 RBM의 추천 성능 비교>

	M-Ensemble	Control-256		Control-512		Control-1024	
HR@10	**0.0748899**	0.0627753	19.3(%)	0.0550661	36	0.0495595	51.11
HR@25	**0.123348**	0.117841	4.67	0.11674	5.66	0.112335	9.8
ARHR	**0.00434233**	0.00337833	28.53	0.00401065	8.27	0.00424348	2.33
Time	0.487963	**0.229002**		0.321		0.489998	



<그림 4. M-Ensemble 모델과 단일 RBM모델의 Hit Rate 비교>

4. 결 론

본 논문에서는 협업 필터링 추천 시스템에 자주 사용되는 RBM 모델을 사용해 분산 RBM 모델을 구축하여 추천 시스템의 성능을 개선하였다. 자체 수집한 데이터를 사용해 해당 모델에 실험을 실시했을 때 기본 추천 시스템 모델들에 비해 우수한 성능을 보여주는 것을 볼 수 있었다.

향후에는 여러 도메인의 추천에 대해 실험을 진행함으로써, 이 연구에서 제시하는 다중 RBM 모델이 책 추천뿐만 아닌 다른 도메인의 추천 문제에서도 적용이 되는지 등의 연구가 가능할 것이다.

[참 고 문 헌]

[1] Bobadilla, Jesús, et al. "Recommender systems survey." Knowledge-based systems 46 (2013): 109-132.

[2] Luo, Xin, et al. "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems." IEEE Transactions on Industrial Informatics 10.2 (2014): 1273-1284.

[3] Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted Boltzmann machines for collaborative filtering." Proceedings of the 24th international conference on Machine learning. ACM, 2007.

[4] Qiu, Xueheng, et al. "Ensemble deep learning for regression and time series forecasting." 2014 IEEE symposium on computational intelligence in ensemble learning (CIEL). IEEE, 2014.

[5] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[6] He, Xiangnan, et al. "Neural collaborative filtering." Proceedings of the 26th international conference on world wide web. International World Wide Web Conferences Steering Committee, 2017.

[7] Deshpande, Mukund, and George Karypis. "Item-based top-n recommendation algorithms." ACM Transactions on Information Systems (TOIS) 22.1 (2004): 143-177.