공학석사학위논문

강화학습 기반
평점 유도 전략 최적화 메타러닝 모델

Meta-learning Model for Optimizing Rating Elicitation Strategies Based on Reinforcement Learning

2018 년 8 월

서울대학교 대학원

산업공학과

신 동 민

강화학습 기반
평점 유도 전략 최적화 메타러닝 모델

Meta-learning Model for Optimizing Rating Elicitation Strategies Based on Reinforcement Learning

지도교수 조 성 준

이 논문을 공학석사 학위논문으로 제출함

2018 년 8 월

서울대학교 대학원

산업공학과

신 동 민

신동민의 공학석사 학위논문을 인준함

2018 년 6 월

위 원 장 박 종 헌 (인)

부위원장 조 성 준 (인)

위 원 장 우 진 (인)

초록

대다수의 추천 시스템 연구는 기존의 알고리즘을 개선하거나 새로운 알고리즘을 제안하는 방식으로 평점 예측에 대한 성능을 개선시켜왔다. 그러나 추천 시스템이 가지고 있는 데이터의 성김성(sparsity)와 롱테일(long-tail) 분포를 고려하였을 때, 추천 시스템에 학습시키는 데이터풀 자체를 변화시킬 필요가 있다. 이에 본 논문에서는 평점 유도 전문가(Rating Eliciation Expert; REE) 모델을 제안하고자 한다. 이 모델은 1) 추천 시스템의 성능을 효율적으로 향상시킬 수 있고 2) 유저가 입력할 가능성이 더 높은 아이템을 먼저 데이터를 입력받는 일종의 액티브 러닝(active learning) 방식 따른다. REE는 적용된 추천 시스템의 학습 과정을 데이터로 여기고 학습함으로써 데이터 제안 규칙을 발전시켜가는 메타 러닝(meta-learning) 모델이며, 이는 기존에 평점 유도 전략을 인간의 직관으로 제안했던 모델들과는 차이가 있다. 실험 결과, REE는 기존 전략들에 비해 수집되는 데이터의 개수는 적었지만, REE로 수집된 데이터들의 평균 성능 개선 정도가 다른 전략에 비해 뛰어났으며 오히려 적은 수의 데이터로 더 좋은 성능 개선을 이끌어 내었다. 이러한 REE는 추천 시스템 모델과 학습 방식에 종속되지 않아 어떠한 시스템에든 적용 가능할 것이라 기대한다.

주요어: 추천 시스템, 평점 유도 전략, 액티브 러닝, 메타 러닝, 강화 학습

학번: 2016-26635

목차

표 목차

그림 목차

제 1 장 서론

모두 습득할 수 없을 만큼 많은 양의 정보가 쏟아지고, 그러한 정보를 토대로 무수한 선택지 중 선택을 내려야하는 현대 사회에서 추천 시스템(recommender system)은 집중해야할 정보와 선택지를 효과적으로 줄여주는 역할을 수행한다. 추천 시스템은 이미 쇼핑, 영화, 음악 등 일상생활 전반에서 활용되고 있고, 그만큼 오랫동안 연구가 진행되어 왔다. 여태까지의 연구들은 대부분 새로운 추천 시스템 알고리즘을 제시하는 방향으로 이어져 왔는데, 가장 대표적인 모델에는 비슷한 성향을 가진 유저를 찾아 아이템의 선호를 예측하는 협업 필터링(collaborative filtering)이 있으며, 최근에는 이에 딥러닝을 적용하여 평가를 예측하고 선호를 파악하는 모델[7]도 등장하였다. 그러나 추천 시스템은 알고리즘의 개선만큼 데이터의 특성이 성능을 좌우하는 분야 중 하나이다. 추천 시스템에 활용되는 데이터들의 특성을 규정하고 그 특성이 모델이 미치는 영향을 파악하는 연구[1]가 별도로 진행될 정도이다. 이러한 데이터들이 가지는 대표적인 특성으로는 수집된 데이터의 수가 모델이 예측해야할 평가의 개수에 비해 현저히 적은 성김성(sparsity), 그리고 그 데이터들가 소수의 유저(혹은 아이템)에 편중되어 나타나는 롱테일(long-tail) 분포가 있다. 이러한 특성은 모델의 성능에 무시하지 못할 만한 영향을 미치며, 이러한 데이터의 문제적 특성에 잘 대응하는 것만으로도 모델의 개선없이 성능 향상을 기대할 수 있다.

본 논문은 이렇게 데이터의 특성을 다루어 추천 시스템의 성능을 개선하는데 초점을 맞추고, 이를 위해 추천 시스템이 필요로 하는 데이터를 먼저 제안하고 유저로부터 입력을 유도하는 모델을 제안하고자 한다. 성김성과 롱테일 분포의 특성은 근본적으로 한정된 수의 데이터 밖에 얻을 수 없는 문제로 인해 발생하는 것이다. 이는 유저가 추천

시스템을 이용하는 본래의 목적인 추천에 관심이 있는 반면, 그 추천을 위해 자신의 선호를 표시하는 평가 행위는 부수적인 것으로 여기는 자연스러운 행동 패턴때문이다. 따라서 유저는 평가 행위를 가능하면 적게하려고 하며, 추천 시스템에 입력하는 데이터를 유저의 자유 의사에 맡겨 입력 받는 것보다 유저가 조금이라도 더 평가를 입력할 가능성이 높은, 그리고 효율이 좋은 데이터를 제안하는 것이 훨씬 도움이 될 것이다. 이와 관련해 평점 유도 전략을 제안하는 추천 시스템에서의 액티브 러닝(active learning)에 관한 연구[4]가 있었으나, 이러한 연구들은 인간의 직관을 활용한 휴리스틱한 방법에 머무는 한계를 보였다.

본 논문에서 제안하는 평점 유도 전문가(Rating Elicitation Expert; REE) 모델은 추천 시스템의 데이터와 추천 시스템이 학습되는 과정을 데이터로 받아들이고 학습하는 메타 러닝(meta-learning) 모델이다. 따라서 인간의 개입을 최소화하였으며, 학습 과정에서 패턴을 발견함으로써 새로운 데이터가 입력될 때마다 지속적으로 발전할 수 있는 모델이다. 따라서 REE는 인간이 발견하지 못하는 학습 과정에 큰 영향을 미치는 데이터의 특성을 잡아낼 수 있으며, 이를 통해 학습에 효과적인 데이터를 선별해 유저에게 제안하는 것이 가능하다. 더불어 강화 학습을 활용하여 유저가 평점을 매길 가능성이 더 높은 데이터를 제안하는 방향도 고려하여 REE를 학습시킬 수 있고, 절대적으로 추천 시스템이 좀 더 많은 양의 데이터로 학습할 수 있게 한다. 이러한 REE는 추천 시스템에서 활용하는 모델에 종속되지 않기 때문에 어떠한 시스템에서든 유저에게 평점 데이터를 입력하는 모델로 활용할 수 있다.

제 2 장 선행연구

2.1 추천 시스템

2.1.1 협업 필터링(Collaborative Filtering)

협업 필터링은 추천 시스템에서 가장 대표적으로 사용되는 머신러닝 기법 중 하나이다. 시스템을 이용하는 유저가 아이템에 대해 평가한 데이터를 이용하여, 추천을 해줄 유저와 추천하고자하는 아이템의 관계를 통해 파악하고 특정 유저(혹은 아이템)와 비슷한 유저(혹은 아이템)를 파악해 아직 평가되지 않은 아이템(혹은 아직 평가하지 않은 유저)에 대한 평가를 예측하는 방식이다. 여기에는 크게 두 가지 방법이 있는데 메모리 기반(Memory-based)과 모델 기반(Model-based)이 있다.

행렬 분해 (Matrix Factorization)는 모델 기반 협업 필터링의 대표적인 방법이다. 행렬 분해는 유저가 아이템에 대해 평가한 내용을 행렬로 만들고 그 행렬보다 더 작은 계수(rank)를 가지는 두 개의 행렬로 분해하는 것을 말한다. 평가를 통해 만든 행렬을 R, 유저의 수를 n, 아이템의 수를 m, R의 계수보다 작은 k로 설정하면, 행렬 분해는 다음을 만족하는 행렬 U, V를 찾는 것이라고 볼 수 있다.

$$R \approx UV R \in \mathbb{R}^{n \times m}, U \in \mathbb{R}^{n \times k}, V \in \mathbb{R}^{k \times m}$$

이렇게 해서 분해한 행렬 U는 시스템 내 존재하는 모든 유저를 k개의 특성으로 표현하는 것이라 볼 수 있고, 마찬가지로 V는 시스템의 아이템을 k개의 특성으로 표현하는 것이라고 볼 수 있다. 위의 식을 만족하는 U, V를 구하기 위해서는 시스템 내 존재하는 평가 데이터 r만을 가지고 학습한다. 가장 간단한 형태의 행렬 분해는 특정 유저 u의

아이템 v에 대한 평가 r_{uv}, 유저 u의 특성을 $\mathbf{u}_u$, 아이템 v의 특성을 $\mathbf{v}_v$라고 했을 때 다음 목표 함수를 통해 학습할 수 있다.

$$\arg\min_{u,v} \sum (r_{uv} - \mathbf{u}_u * \mathbf{v}_v)^2$$

결국, 위의 목표함수를 만족하는 것은 현재 보유하고 있는 평가 데이터를 가장 잘 예측하는 U와 V를 찾는 것이라고 볼 수 있다. 이렇게 구한 U와 V를 통해 아직 평가하지 않은 아이템이라 할지라도 $\hat{r} = \mathbf{u}_u * \mathbf{v}_v$로 예측을 할 수 있는 것이 행렬 분해이다.

이 방법이 발전하여 가중치 정규화 행렬 분해 (Weighted Regularization Matrix Factorization; WRMF [11]), 비음수 행렬 분해 (Non-negative Matrix Factorization; NMF [9])등의 방법들이 등장하였다. 또한 딥러닝 기법을 활용해 행렬 분해를 시행하는 모델도 최근 들어 각광을 받고 있다. 평가 데이터만을 활용한 모델[7]부터, 유저 혹은 아이템의 컨텐츠를 활용하여 특성 벡터를 학습하는 모델들[15], [10]도 등장했다.

2.1.2 추천 시스템에서 수집되는 평가 데이터의 특성

추천 시스템에서 수집되는 평가 데이터는 다른 태스크에 활용되는 데이터들과는 몇 가지 고유한 특성을 보인다. Adomavicius 등[1]은 MovieLens 데이터를 기반으로 협업 필터링을 진행할 때에 있어 고려해야할 특성들을 연구하였는데, 여기서 제시된 주요한 특성은 성김성(Sparsity), 평가 데이터 개수의 롱테일 분포(long-tail distribution) 등이 있다.

성김성(Sparsity)은 보유한 평가 데이터를 행렬의 형태로 나타내었을 때 해당 행렬에 채워지지 않은 데이터의 비율을 말한다. 일반적으로 성김성은 유저와 아이템의 크기가 클수록 더 빠르게 증가하는데, 본 논문에서 활용한 MovieLens-1m 데이터의 경

우 4%의 밀도(행렬 내 채워진 데이터의 비율)를 보이며, 이보다 더 큰 MovieLens-20m의 경우 MovieLens 1m에 비해 23배 더 많은 유저와 6.8배 더 많은 아이템에 대한 데이터 이며 0.5%로 더 낮은 밀도를 가지고 있어 성김성이 더 커지는 것을 알 수 있다. 성김성은 모델이 학습할 수 있는 데이터의 개수가 예측해야하는 양에 비해 적다는 것을 의미하기 때문에 예측 성능을 떨어뜨리며 동시에 과적합(over-fit)의 가능성도 증가시키는 특성이다.

이에 더불어 이 데이터들이 롱테일 분포를 띠고 있기 때문에 데이터 불균형성(data imbalance)의 문제도 동시에 발생한다. 롱테일 분포는 분포의 형태를 본따 지어진 이름인데, 아이템(혹은 유저) 각각에 대해 유저로 부터 받은 (혹은 아이템이 얻은) 평점 데이터의 개수를 계산하고 이에 대해 분포를 그렸을 때 데이터의 수가 많은 아이템의 수가 점점 줄어드는 형태의 분포를 보이는 것을 말한다(4.1 참고). 이러한 분포가 의미하는 것은 소수의 인기가 있는 아이템(혹은 소수의 평점을 많이 매긴 유저)에 평점 데이터가 편중되어 있다는 것이다. 모든 아이템에 대해 고른 예측 성능을 보이기 위해서는 추천 시스템의 학습 과정에서 모든 아이템이 학습될 기회가 고르게 주어져 아이템의 특성 벡터들이 고르게 학습되어야 한다. 이 때 학습의 기회는 평점 데이터의 수에 비례해서 주어지는데, 롱테일 분포 하에서는 결국 데이터가 편중되어 있으므로 특정 아이템만 과도하게 학습되는 결과를 낳게 된다.

2.2 커리큘럼 러닝(Curriculum learning)

일반적으로 머신러닝은 해결하고자 하는 태스크가 주어지고 그것을 주어진 데이터를 통해 해결하는 형태가 된다. 이 때 태스크에 맞게 데이터의 특성을 파악하고 전처리하는 과정이 기본적으로 진행되기는 하지만, 주된 관심사는 해당 태스크를 제대로 수행하기 위해 어떤 기법을 활용하는가 이다. 예컨데, 추천 시스템과 같이 평점을 잘 예측해야하는 태스크에서는 주어진 데이터를 활용해 행렬 분해를 적용할 것인지, 혹은 딥러닝을 활용할 것에 초점이 맞춰진다.

본 절에서 설명하고자 하는 커리큘럼 러닝은 Bengio 등[3]이 제안하였으며, 머신러닝을 적용하는데 있어 모델 그 자체가 아니라 모델들에 활용되는 데이터를 활용하는 전략에 초점을 맞추고 있다. Bengio 등은 모델의 학습이 진행하는 과정에서 데이터를 통해 커리큘럼을 정의하고, 이를 토대로 학습 과정에 활용하는 데이터셋을 달리해야 한다고 주장했다. 이 때 커리큘럼은 학습하여야할 데이터의 분포 배열을 말하며, 이 배열을 정렬하는 기준이 데이터 분포의 엔트로피가 증가하는 방향이 된다. 보다 수식적으로 표현해보면,

$$H(Q_d) < H(Q_d + \lambda) \tag{2.1}$$

즉 학습에 활용할 데이터의 분포를 배열하는데 있어 다음에 올 데이터의 분포 $Q_{d+\lambda}$에 대한 엔트로피는 현재 데이터 분포 Q_d의 엔트로피보다 커야하며, 엔트로피가 크다는 것은 더 다양한 종류의 데이터를 포함하고 있다는 것을 의미한다. 결국 엔트로피가 큰 데이터 분포 $Q_{d+\lambda}$는 Q_d에 비해 더 어려운 데이터 셋이라고 보는 것이다. 따라서 엔트로피가 증가하는 방향으로 데이터셋을 변화시키면서 분포를 배열하면 모델은 쉬운 데이터셋부터 어려운 데이터셋 순으로 학습을 진행하게 되고 이러한 학습 방법은 더

나은 실험 결과를 가져온다고 주장하였다.Bengio 등은 큰 개념의 틀에서 커리큘럼 러닝의 컨셉을 제안하였으나, 실 엔트로피에 대응되는 제적인 학습 과정에서 측정 가능한 데이터의 난이도를 구체적으로 제시하지는 못했다.

데이터의 난이도 개념을 비교적 구체적으로 정립하여 커리큘럼 러닝을 시도한 연구는 Kumar 등[8]의 자기 주도 러닝 (Self-paced learning)이다. Kumar 등은 모델이 특정 학습 시점에서 해당 데이터를 토대로 예측했을 때의 예측값과 실제값과의 차이를 난이도로 정의하였다. 예측값과 실제값의 차이가 크다는 것은 그만큼 현 시점에서 모델에게 어려운 데이터라는 의미이며 차이가 작다는 것은 더 쉬운 데이터라는 의미를 가지도록 한 것이다. 이렇게 정의된 난이도 척도를 이용하여 커리큘럼 러닝을 실질적으로 적용하기 위해, 현재 학습 시점의 모델에 미리 지정된 난이도 임계치 k를 기준으로 데이터를 사용할지 말지를 결정하는 항을 도입하였다.

Kumar 등 이후로 여러 머신러닝 기법들에 자기 주도 러닝을 적용하는 방법들이 연구되어 왔으며 행렬 분해에 대해서도 이러한 연구[14]가 진행되었다. 이 연구에서는 가장 기본적인 행렬 분해 기법에 자기 주도 러닝을 도입하는 목표함수 식을 제안하였는데, 그 식은 다음과 같다.

$$\arg\min_{U,V,w} \sum_{(u,v)\in\Omega} w_{uv} l(r_{uv}, \mathbf{u}_u \mathbf{v}_v) + \lambda R(U,V) + \sum_{(u,v)\in\Omega} \frac{\gamma^2}{w_{uv} + \gamma k} \tag{2.2}$$

본래의 행렬 분해와 식이 다른 부분은 w_{uv}를 도입하였다는 점이다. 이 w_{uv}가 난이도 임계값 k에 따라 학습을 할지 말지를 결정하는 부분이며, 마지막 항은 w_{uv}에 대한 정규화를 위한 항이다. 특별히 위 논문에서는 w의 값을 0, 1의 이산적인 값으로 보는 것이 아니라 0과 1사이의 연속적인 값으로 보는 soft-weighting을 도입하였다는 점이 특징이라고 할 수 있다.

2.3 메타 러닝(Meta-learning)

그러나 커리큘럼 러닝은 그 자체로서 몇 가지 한계점을 지닌다. 그것은 인간이 난이도에 대한 임계치를 지정해주어야한다는 점이다. 이는 데이터를 기반으로 스스로 학습을 진행하도록 하려는 머신러닝의 목표와는 대치되는 부분이다. 이러한 부분을 해결하기 위해 도입한 것은 메타 러닝(meta-learning)이다. 메타 러닝은 굉장히 광범위한 개념이지만 본 논문에서 주목하고자 하는 방향은 특정한 모델의 학습 과정을 학습하는 모델에 관한 것이다. 즉, 관심이 있는 머신러닝 모델의 학습 정도를 최상으로 방법을 찾는 것으로, 대표적으로 시도되었던 것은 머신러닝 모델에서 인간이 지정해주어야하는 변수들인 하이퍼파라미터(hyperparmeter)를 설정하는 모델을 제안하는 연구[2]이다. 이 연구에서는 실제로 학습을 진행하는 머신러닝 모델을 Optimizee로 두고 Optimizee의 학습 과정을 학습해 하이퍼파라미터 중 하나인 학습 속도(learning rate)를 결정해주는 Optimizer를 도입하였다. RNN 계열의 LSTM으로 구성된 Optimizer는 먼저 그림 2.1에 나타나 있는 과정을 통해 Optimizee의 매 학습 과정마다 학습 계수를 학습한다. 그리고 Opitmizee가 학습을 진행할 때 각 시점마다 학습 계수를 새롭게 정의하는 역할을 한다. 위의 연구와 같이 하이퍼파라미터를 자동으로 설정하는 메타 러닝 모델은 커리큘럼 러닝과 밀접한 연관성이 있다. 자기 주도 러닝의 난이도 임계치 역시 하이퍼파라미터의 일종이며, 따라서 이를 자동으로 설정하는 메타 러닝 모델을 도입할 수도 있기 때문이다.

하지만 근본적으로 자기 주도 러닝에서 임계치를 설정하는 문제 자체를 되짚어볼 필요가 있다. 커리큘럼 러닝을 통해 해결하고 싶은 근본적인 문제는 어떤 머신러닝 모델을 학습할 때 빠르게 모델의 성능을 올려주는 효과적인 데이터셋을 찾는 것이다. 즉 궁극적인 목적은 모델의 성능을 효율적으로 향상시키는 것이라는 점이다. 따라서 자기 주도 러닝에서 도입한 난이도 임계치는 모델의 성능과 연관된 잠재적인 데이터셋의

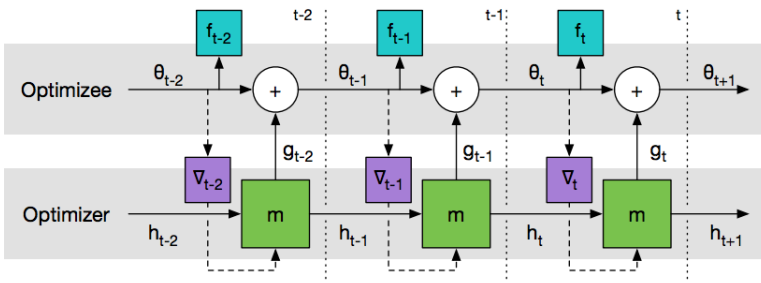


Figure 2.1: [2]에서 제안한 모델의 학습 과정 모식도

특성 중 극히 일부이며 인간이 인위적으로 설정한 척도 중 하나에 지나지 않는 것이다.

이러한 관점에서 보다 직접적으로 모델의 학습 과정과 전반적인 데이터에 관여하는 메타 러닝 모델로 NDF(Neural Data Filtering [6])가 있다. NDF는 머신러닝 모델의 학습 과정을 강화 학습을 통해 현 시점의 머신러닝 모델이 학습하고자 하는 데이터들 각각을 사용할지 말지를 결정하는 메타 러닝 모델이며, 특히 분류 문제를 해결하는데 효과적인 것으로 나타났다. 본 논문에서 목표하고 있는 모델 역시 이러한 NDF의 컨셉과 같이 인간이 모델의 학습 과정에 대한 개입을 최소화하고, 메타 러닝 모델이 머신러닝 모델의 학습 과정을 직접 학습할 수 있는 강화 학습을 도입하는 것이다.

2.4 강화 학습(Reinforcement learning)

본 논문의 메타 러닝 모델이 활용한 학습 프레임워크는 강화 학습이다. 강화 학습의 전반적인 구조는 그림 2.2로 나타낼 수 있다. 모델에 해당하는 행위자(agent), 그리고 행위자의 외부에서 상태와 보상을 주는 환경(environment)이 서로 상호작용하며 학습을 진행한다. 환경에서 상태가 주어지면 행위자는 그 상태를 기반으로 정책(policy)에 따라 행동(action)를 하게 되며, 이 행동의 결과가 환경을 변화시켜 환경은 보상(reward)를 행위자에게 알려준다. 행위자는 이러한 상태-행동-보상의 반복적인 기록을 하나의

에피소드로 받아들이게 되며, 에피소드 내의 보상과 보상 시점을 토대로 행위자가 학습해야할 가치(value)를 결정하여 정책을 결정하는 것이 강화 학습의 컨셉이다.

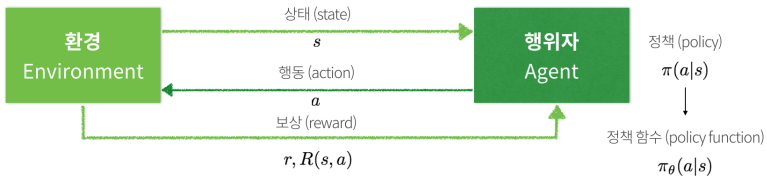


Figure 2.2: 강화 학습의 모식도

위와 같은 형태의 학습 구조는 마르코브 결정 과정(Markov Decision Process; MDP)라고 불리는데, 이는 강화 학습이 마르코브 성질을 만족하도록 가정하고 있기 때문이다. 그림 2.3에 나타난 것과 같이 행위자가 취하는 행동 A_i은 현재 상태 S_i에만 영향을 받고, 그로 인한 보상 R_{i+1}은 행동 A_i과 상태 S_i, 그리고 행동으로 인해 변화된 상태 S_{i+1}에만 의존한다고 본다.

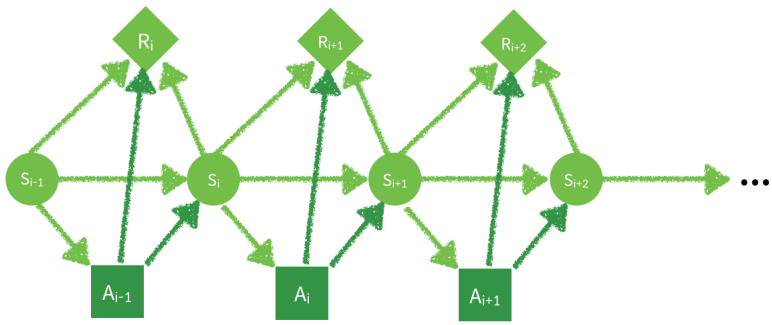


Figure 2.3: 마르코브 결정 과정에 대한 모식도

마르코브 결정 과정 중 많은 부분들을 변수화할 수 있는데, 상태에 따른 행위를 결정하는 정책을 행위자 내 내부 모델로 보는 정책 함수(policy function, $\pi_\theta(a|s)$)가 대표적이다. 정책 함수를 변수화할 때 인공 신경망을 활용하기 때문에 정책 함수를

정책망(policy network)이라고도 한다. 본 논문에서도 정책망을 도입하였으며, 이를 학습시키기 위해 활용한 방법은 REINFORCE[13]이다. 몬테-카를로 정책 그래디언트(Monte-Carlo Policy Gradient)라고 불리는 이 방법은 에피소드에 기록된 각 시간 간격별로 미래의 보상을 고려한 가치(value)로 계산한다.

$$G_t = r_t + \gamma r_{t+1} + ...\gamma^{T-t} r_T \tag{2.3}$$

현재 시점의 가치 G_t는 현 시점의 보상 r_t부터 최종 시점의 보상 r_T까지를 합한 값으로 나타내며, 이 때 할인율γ 만큼을 매 시점마다 보상에 계속 곱해주며 더한다. 따라서 가치는 한 에피스드 내에서 해당 시간 간격에 정책망이 행한 행위가 미래에 얼마만큼의 영향을 끼쳤는지를 고려하는 값이며, 이를 이용하여 미래까지를 고려하여 현재 시점의 행위를 평가하여 정책망을 학습하는 것이라 볼 수 있다. 결국 정책망은 가치를 최대화하는 방향으로 학습시키게 된다.

$$\theta \leftarrow \theta + G_t \sum \frac{\partial \log \pi_\theta(a|s)}{\partial \theta} \tag{2.4}$$

REINFORCE의 학습을 나타내는 수식 2.4를 살펴보면, 비용의 최소화가 아닌 가치의 최대화를 목표로 하기 때문에 일반적인 신경망을 학습시키는 방향과 반대 방향의 그래디언트를 가지는 것을 알 수 있다. 그리고 가치를 학습하기 위해 각 시점마다 계산한 가치 G_t를 곱한 형태로 학습시킨다.

2.5 액티브 러닝(Active learning)

앞서 언급한 커리큘럼 러닝과 메타러닝 모델은 근본적으로 이미 수집된 데이터를 어떤 식으로 학습할 것인가를 결정하는 모델들이다. 이러한 모델들을 주로 사용하는

이유는 이러한 모델을 적용하는 태스크들이 추가로 데이터를 수집하기 어려운 여건에 놓여 있는 예측과 분류를 다루기 때문이다. 그러나 추천 시스템의 경우 기본적으로 유저와의 상호작용을 통해 데이터를 지속적으로 수집할 수 있는 여건이 갖춰져있다. 따라서 아예 데이터를 수집하는 단계에서 부터 유저에게 얻고 싶은 데이터를 제안하는 방식으로 데이터의 질을 향상시킬 방법이 존재한다.

이와 관련된 연구 분야는 액티브 러닝(active learning)이다. 액티브 러닝(active learning)[12]은 실제 태스크를 수행하는 모델을 학습자(learner)로 보고 학습자가 학습에 앞서 전문가(expert) 모델에 데이터의 사용여부를 물어보고 사용하도록 하는, 능동적인 학습 프레임 워크이다. 이렇듯 포괄적인 의미에서의 액티브 러닝은 커리큘럼 러닝과 비슷한 개념으로 사용되지만, 추천 시스템에서의 액티브 러닝은 평점 유도 전략(rating elicitation strategy[4])에 초점을 맞춘다. 이는 추천 시스템이 학습자가 데이터를 수집하기에 앞서 전문가 모델이 미리 현재 상태의 학습자를 진단해 필요한 데이터를 제안할 수 있도록 설계하는 것이 가능하기 때문이다.

평점 유도 전략은 크게 두 방향으로 나눠지는데 모든 유저에게 동일하게 적용하는 비개인화(non-personalized) 전략과 개별 유저에게 다른 평점 데이터 입력을 유도하는 개인화(personalized) 전략[5]이다. 먼저 비개인화 전략의 경우 추천 시스템이 보유하고 있는 모든 데이터를 활용하여 전반적인 평점 유도 전략을 수립하는 방법으로, 대표적인 방법으로 인기 기반(popularity based)과 엔트로피 기반(entropy based)가 있다. 인기 기반 방법은 용어 그대로 보유한 데이터 중 가장 많은 데이터를 보유한 아이템부터 제안하는 방법으로, 추천 시스템 내 보편적인 아이템을 잘 추천하는 것에 초점을 맞추고 있는 것이라 볼 수 있다. 엔트로피 기반 방법은 아이템 별로 평점의 분포를 보았을 때 가장 다양한 종류의 평점을 받은 영화부터 입력을 제안하는 방식이다. 엔트로피가 큰 영화일수록 특성이 더욱 불확실하다는 것을 의미하며, 따라서 이를 먼저 제안하여 평점

분포를 변화시켜 엔트로피를 줄이는 것을 목표로 한다. 엔트로피가 줄어든다는 것은 평점 분포가 어느 한 분포에 쏠리게 된다는 것을 의미하고 이는 해당 아이템에 대한 특성이 보다 확실해지게 된다는 것을 의미한다.

개인화 전략은 유저 각각에 대해 서로 다른 아이템에 대한 데이터 입력을 유도하는 전략을 말한다. 대표적인 방법으로 이항 예측(binary prediction) 방법이 있다. 이 방법은 추천 시스템의 평점 데이터의 존재 유무만을 가지고 이항적인 행렬을 만든 뒤 그 행렬에 대해 행렬 분해 등과 같은 추천 시스템의 예측 모델로 학습시킨 모델을 이용한다. 이항적 행렬에 대한 예측 모델에서 유저와 아이템에 대해 예측되는 값은 해당 유저가 해당 아이템에 대해 평가할 확률값이라고 볼 수 있기 때문에 이항 예측 방법은 개개의 유저마다 평가할 확률이 높은 아이템을 제안하는 방식이라고 볼 수 있다.

이와 같은 고전적 평점 유도 전략들은 추천 시스템 데이터에 대해 직관적으로 분석 가능한 규칙들을 기반으로 하기 때문에, 추천 시스템의 예측 성능을 향상 시키려고 하는 목적을 직접적으로 달성하는 데에 한계가 존재한다. 또한 그러한 규칙들을 휴리스틱하게 만들었기 때문에 데이터를 기반으로 규칙 자체를 발전시키는 것이 불가능한 구조이다.

제 3 장 제안하는 모델

3.1 추천 시스템의 구조

본 논문에서 제안하고자 하는 평점 유도 전문가(Rating Elicitation Expert; REE) 모델은 개별적으로 동작하는 것이 아닌, 추천 시스템 전반의 구조를 바꾸게 된다. 그림 3.1와 같은 일반적인 추천시스템에서는 유저에게 정보를 입력 받고 학습 한 후 예측 또는 추천을 제공해준다.

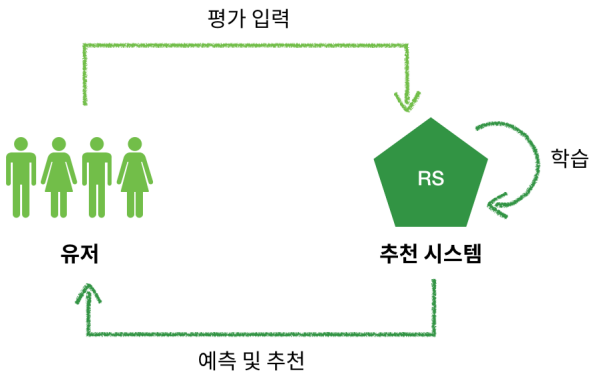


Figure 3.1: 일반적인 추천 시스템의 구조

그러나 REE가 도입된 추천 시스템은 그림 3.2와 같이 유저에게 정보를 입력 받기에 앞서 현재 추천 시스템의 상태를 평점 유도 전문가에게 입력 받고, 이를 기반으로 REE는 각 유저에게 데이터의 입력을 제안한다. 유저는 이에 대해 평가를 입력하며 이 평가가 추천 시스템의 학습과 추천에 활용된다.

이 때, 그림 3.2과 같은 새로운 구조에서 목표하는 바는 1) 추천 시스템의 성능을 효과적으로 향상시킬 수 있는 데이터를 효율적으로 얻고 2) 입력 제안을 통해 유저에게

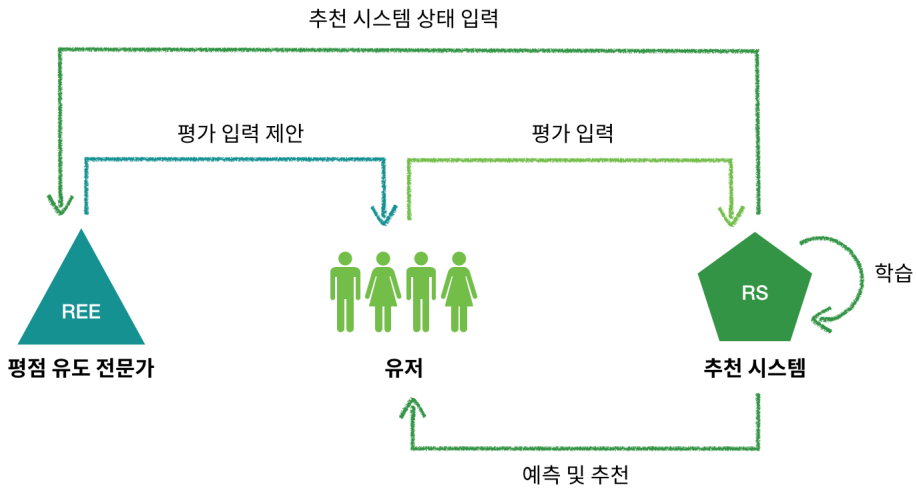


Figure 3.2: REE가 도입된 추천 시스템의 구조

더 많은 평점 데이터를 입력받을 수 있도록 하는 것이다.

이처럼 REE는 다양한 추천 시스템에 도입이 가능하지만 본 연구에서는 보다 구체적인 설명을 위해 다음과 같은 추천 시스템에 REE를 도입하는 경우에 대해 설명하겠다.

1. **협업 필터링(collaborative filtering)** 협업 필터링을 추천 시스템의 평점 예측 모델로 사용한다.

2. **오프라인 러닝(offline-learning)** 추천 시스템의 학습은 모여있는 데이터를 가지고 서비스와 별도로 진행한다.

3. **동기적 제안-응답(synchronous suggestion-response)** REE에 의해 평점을 얻을 데이터가 결정되고 제안되면 유저가 이에 대해 즉각적으로 응답한 것으로 간주한다.

4. **점진적 학습(incremental learning)** 응답이 기록되자마자 마치 온라인 러닝

(online-learning)처럼 현재 상태의 추천 시스템에 응답을 더하여 학습한다.

3.1.1 학습 과정

위와 추천 시스템에서 REE를 도입하여 학습하는 방법은 알고리즘 3.1.1와 같다.

Algorithm 1 Training process of recommender system introduced REE

Input : dataset D, number of epoch e_{max}
Output : trained $\mathcal{RS}$

procedure TRAININGRSINTRODUCEDREE
  split D to training data D_{train} and test data D_{test}
  split D_{train} to trained data($\mathcal{RS}$ already known) D^k_{train} and train needed data D^n_{train}
  train recommender system $\mathcal{RS}$ initially with D^k_{train}
  for from epoch $e = 0$ to $e = e_{max}$ (increment by 1) **do**
    foreach user $u \in U_{D^n_{train}}$ **do**
      for not suggested data $x_u \in X_u = \{i_u | i_u \notin D^k_{train}\}$ **do**
        get state s_{x_u} from $\mathcal{RS}$ and store to S_u
        decide a_{x_u} to suggest($a = 1$) or not($a = 0$) using policy network $\pi_\theta(a_{x_u}|s_{x_u})$ of trained REE and store to A_u ▷ Algorithm 2(알고리즘 3.2.2)
      end for
      get suggestions $SI_u = \{x_u | a_{x_u} = 1\}$, elicited data $d'_u = \{x_u | SI_u \cap D^n_{train\_train}\}$
      train $\mathcal{RS}$ with d'_u
      add d'_u to D^k_{train}, remove d'_u from D^n_{train}
    end for
  end for
end procedure

일반적인 추천 시스템과 달리 학습 과정에서 새롭게 도입되는 부분은 평점을 유저에게 제안하는 상황을 재현하고 이를 학습하는 부분이다. 이를 위해서는 REE가 제안할 아이템을 뽑아내는 과정이 필요하며, 제안된 아이템이 학습 데이터 내에 존재하면 학습하는 방식으로 학습이 진행되어야 한다(동기적 제안-응답). 이러한 학습 방식은 [4]에서 제안한 일반적인 평점 유도 전략을 이용한 학습 구조를 REE에 맞게 조금 변형한 형태이다.

평점 유도 전략을 활용하는 추천 시스템에서는 학습 데이터 셋을 두개로 분리하는 것이 특징이다. 이미 추천 시스템이 알고 있는, 즉 추천 시스템에 입력된 학습된 데이터셋 D^k_{train}과 유저는 알고 있지만 아직 학습되지 않은 학습 필요 데이터셋 D^n_{train}로 나뉜다. 그리고 D^k_{train}를 이용해 추천 시스템을 학습하여 초기값을 설정한다. 이후부터는 평점 유도 모델의 입력값이 되는 유저에 대한 현재 추천 시스템의 상태 S_u를 구하는데, 이는 추천 시스템에는 존재하지만 아직 유저에게 입력 받지 못한 모든 아이템 X_u에 대해 각각 추출한다. 각 아이템에 대해 추출한 상태 S_{x_u}가 평점 유도 전문가 모델에 입력되면 해당 x_u를 제안할지 말지를 결정한 뒤, 제안하고자 하는 아이템 셋 SI_u을 결정한다. 추천 시스템을 학습할 때는 SI_u 중 D^n_{train}에 존재하는 데이터 d'_u만을 현재 시스템에 점진적으로 학습시킨다. 여기서 d'_u는 제안에 대해 유저가 응답한 데이터로 본다. 이러한 과정을 지정된 조건을 만족할 때까지 모든 유저에 대해 반복하는 것이 추천 시스템에 대한 학습 방법이다.

3.2 평점 유도 전문가 모델 (Rating Elicitation Expert; REE)

본 논문에서 제안하는 평점 유도 전문가 모델(Rating Elicitation Expert; REE)은 유저에게 추천 시스템이 입력받길 원하는 아이템을 해당 유저에게 제안하는 모델로, 개인화 평점 유도 전략 모델의 일종이다. 그러나 이 모델은 추천 시스템의 학습 과정을 강화 학습을 통해 학습하는 메타 러닝 모델로, 기존의 휴리스틱한 평점 유도 전략과는 차이가 있다. REE는 해당 유저와 해당 유저가 아직 입력하지 않은 추천 시스템 내 아이템에 대해 다양한 특성을 입력값으로 받으며, 그 결과로 어떤 아이템을 제안할 것인지를 결정하게 된다. REE는 학습 하기에 앞서 기존의 추천 시스템에 수집된 데이터가 어떠한 과정을 거쳐 수집되었는지에 따라 학습 알고리즘이 달라지게 된다. 대부분의 경우에는 기존의 추천 시스템을 사용하고 있기 때문에, 데이터의 입력을 제안하는 부분이

없으며 따라서 이렇게 수집된 데이터는 유저의 자유 의사에 의해 수집된 데이터라고 할 수 있다.

3.2.1 학습을 위한 가정

따라서 이러한 데이터로 REE를 학습하기 위해서는 다음과 같은 가정이 필요하다.

1. **시간 불변 선호 (time invariant preference)** 한 유저에 대해 수집된 데이터는 시간과 무관하게 하나의 선호에 대한 데이터이다. 즉 한 유저는 하나의 선호로 규정된다.

2. **동기적 제안-응답 (synchronous suggestion-response)** REE에 의해 평점을 얻을 데이터가 결정되어 제안되면 유저가 이에 대해 응답하고 추천 시스템이 학습될 때까지 다음 유저에게 제안되지 않는다.

이러한 가정이 필요한 이유는 REE를 학습 시키는 상황에서 유저의 자유 의사에 의한 데이터는 시간적 변화를 고려할 수 없기 때문이다. 시간에 따른 유저의 선호의 변화를 고려하기 위해서는 일정한 주기를 가지고 계속해서 평점 데이터의 입력 혹은 재입력을 요구하면서 응답여부를 기록하는 식의 데이터가 필요하다. 따라서 이러한 제안-응답 과정을 실제로 거치치 않은 자유 의사에 의한 데이터는 REE가 학습 과정 중에 제안-응답을 재현하고자 할 때, 아이템이 평가된 순서를 고려할 수 없으므로 평가된 모든 아이템들을 응답할 수 있는 후보로 올려 놓아야한다.

동기적 제안-응답도 위와 비슷한 맥락에서 필요한 가정이다. 추천 시스템이 운용되는 실제 상황을 고려해보면, 추천 시스템은 시스템 내에 들어오는 유저에게 현재 시점의 데이터를 기반으로 평점 데이터 입력을 제안하게 된다. 이 때, 제안 받는 순서는 시스템에 들어온 순서이지만 응답 순서는 유저마다 서로 다르다. 즉, 제안-응답은 유저에 대해

비동기적(asynchronous)이다. 또한 이와 별개로 추천 시스템이 적용한 모델에 따라 실시간으로 데이터를 입력 받을 때마다 모델을 학습하는 온라인 러닝의 여부도 제각각일 수 있다. 위와 같은 비동기적 상황에서는 데이터가 제안되고 수집된 시점, 그리고 학습된 시점이 중요하다. 평점을 매긴 시간을 고려할 수 없는 자유 의사에 의한 데이터는 결국 이러한 비동기적 상황을 고려한 학습이 불가능하며, 이에 한 유저를 하나의 시간 간격으로 보고 제안-응답-추천시스템 학습하는 가정이 필요하다.

위와 같은 가정은 실제 추천 시스템의 상황과 분명 차이가 있지만 데이터 수집 기간이 지나치게 길지 않은 경우 위와 같은 가정을 적용하더라도 실제 상황을 크게 벗어나지 않으므로 REE 모델 학습을 위해 타당한 가정이라고 할 수 있다.

3.2.2 학습 과정

위의 알고리즘 3.1.1에서 사용한 REE는 알고리즘 3.2.2에 따라 학습된다.

REINFORCE를 활용한 강화 학습을 통한 학습은 일종의 시뮬레이션을 통한 학습과 같다. 이 때 한 시뮬레이션을 하나의 에피소드라고 보며 한 에피소드 마다 추천 시스템을 처음으로 리셋하고 REE와 상호작용을 통해 추천 시스템이 학습하는 과정을 거쳐 REE를 학습하도록 한다. 따라서 매 에피소드마다 기본적인 알고리즘의 틀은 REE를 도입한 추천 시스템의 학습 알고리즘 3.1.1과 동일하다. 단, 매 유저에 대해 REE가 입력 받은 상태 S_u, 아이템의 제안 여부 A_u, 그리고 그에 따른 보상 r_u 를 계산하고 지속적으로 기록한다.

에피소드가 종료되는 조건은 세 가지인데 1) 지정된 최대 시간 간격 t_{max}에 도달하는 경우 2) 학습을 검증하는 학습 검증 데이터 D_{train_val}에 대해 추천 시스템의 성능을 RMSE(Root Mean Squared Error)로 평가해, 설정한 임계값 $rmse_{min}$보다 작아지는 경우 3) 학습 필요 데이터 $D^n_{train_train}$가 더 이상 없는 경우이다.

하나의 에피소드가 종료되면 에피소드를 되감기 하면서 기록된 상태(S_e), 제안 여부

(A_e), 보상(R_e)으로 REE를 학습한다. REE는 시간 간격(한 유저) t마다 기록된 상태 $S_e[t]$와 행동 $A_e[t]$에 대해 그래디언트를 구해서 정책망을 업데이트 하는데, 이 때 해당 시간 간격의 가치 G_t를 그래디언트에 곱해주어 가치를 증가시키는 방향을 강화시킨다.

3.2.3 상태(state)의 정의

S_u로 표현되는 현재 유저에 대한 추천 시스템의 상태는 근본적으로 현재 추천 시스템과 관련된 무엇이든 포함할 수 있다. 본 논문에서는 현재 평점 데이터가 없는 아이템 x_u에 대해 상태 s_{x_u}를 현재 추천 시스템이 나타내는 특성 벡터 $\mathbf{v}_{x_u}$와 현재 유저의 특성 벡터 $\mathbf{u}_u$를 결합한 벡터를 활용하였다. 특성 벡터는 아이템 혹은 유저의 특성을 나타낼 수 있는 방법이며, 동시에 추천 시스템의 현재 학습 상태를 반영하는, 즉 학습이 진행됨에 따라 변화하는 벡터이다. 따라서 추천 시스템의 현재 학습 상태와 데이터의 특성을 모두 잠재적으로 내포하는 데이터로 볼 수 있다.

$$s_{x_u} = (\mathbf{v}_{x_u}, \mathbf{u}_u), \forall x_u \in X_u \tag{3.1}$$

3.2.4 보상(reward)에 대한 설계

강화 학습에 있어서 리워드는 행위자의 정책 방향을 결정하는데 중요한 역할을 한다. 본래 REE를 도입한 추천 시스템을 통해 추구하고자한 것은 데이터의 질을 높이는 것과 좀 더 많은 데이터를 유저가 입력하도록 하는 것이다. 이를 만족하기 위해 리워드 r_u는 다음과 같이 정의된다.

$$r_u = \frac{rmse_{u-1}(D_{train\_val}) - rmse_u(D_{train\_val})}{rmse_{u-1}(D_{train\_val})} \times \frac{|d'_u|}{|SI_u|} \tag{3.2}$$

REE를 학습할 때 한 에피소드 내에서 하나의 시간 간격은 하나의 유저이다. 따라서 유저 별로 제안 후 응답된 데이터로 추천 시스템을 학습한 뒤 보상을 계산하는 과정을 거친다. 수식 3.2.4에서 첫 항은 성능 개선율(performance improvement ratio)로써, 학습-검증데이터 D_{train_val}에 대해 현재 시간 간격과 바로 전 시점 시간 간격의 추천 시스템으로 각각 에러를 계산 한 뒤, 이전에 비해 현재가 얼마나 개선되었는지를 나타낸다. 성능 개선율이 높을수록 현재 시점에 추가된 데이터로 학습된 추천 시스템이 이전에 비해 더 많이 개선된 것이므로 더 좋은 데이터이므로, 이를 보상에 추가하여 성능 개선율을 높일수 있는 데이터를 선택하도록 유도하였다. 한편, 다음 항은 REE에 의해 제안된 아이템의 수 $|SI_u|$에 대해 유저가 얼마만큼을 데이터를 입력했는지($|d'_u|$)를 나타내는 응답율(acquired ratio)로써, 유저가 입력할 가능성이 높은 데이터를 선택하는 방향으로 학습 시키기 위한 보상의 요소이다.

Algorithm 2 Training Process of Rating Elicitation Expert(REE)

Input : D_{train}, episode number e_n, max time step t_{max}, rmse threshold $rmse_{min}$, discount factor γ

Output : trained REE (trained policy network π_θ)

procedure TRAININGREE

split D_{train} to training data D_{train_train} and validation data D_{train_val}

for from episode $e = 0$ to $e = e_n - 1$ (increment by 1) **do**

reset recommender system model $\mathcal{RS}$, state buffer S_e, action buffer A_e, reward buffer R_e of episode

split D_{train_train} to trained data($\mathcal{RS}$ already known) $D^k_{train\_train}$ and train needed data $D^n_{train\_train}$

train $\mathcal{RS}$ initially with $D^k_{train\_train}$

time step $t \leftarrow 0$

while $t < t_{max}$ AND $rmse(D_{train_val}) > rmse_{min}$ AND $D^n_{train_train} \neq \emptyset$ **do**

select user $u \in U_{D^n_{train}}$

for not suggested data $x_u \in X_u = \{i_u | i_u \notin D^k_{train}\}$ **do**

get state s_{x_u} from $\mathcal{RS}$ and store to S_u

decide a_{x_u} to suggest($a = 1$) or not($a = 0$) using policy network $\pi_\theta(a_{x_u}|s_{x_u})$ and store to A_u

end for

get suggestions $SI_u = \{x_u | a_{x_u} = 1\}$, elicited data $d'_u = \{x_u | SI_u \cap D^n_{train\_train}\}$

train $\mathcal{RS}$ with d'_u

calculate reward $r_u = R(d'_u, SI_u, D_{train\_val})$ and store S_u to S_e, A_u to A_e, r_u to R_e

add d'_u to $D^k_{train\_train}$, remove d'_u from $D^n_{train\_train}$

$t \leftarrow t + 1$

if $t \geq t_{max}$ OR $rmse(D_{train\_val}) \leq rmse_{min}$ OR $D^n_{train\_train} = \emptyset$ **then**

$T \leftarrow t$ and break

end if

end while

for from time step $t = 0$ to $t = T - 1$ (increment by 1) **do**

calculate value $G_t = R_e[t] + \gamma R_e[t+1] + ... + \gamma^{T-t} R_e[T]$

train policy network π_θ using $S_e[t], A_e[t], G_t$

$$\theta \leftarrow \theta + G_t \sum_{\substack{s \in S_e[t] \\ a \in A_e[t]}} \frac{\partial \log \pi_\theta(a|s)}{\partial \theta}$$

end for

end for

end procedure

제 4 장 실험

4.1 실험 문제

4.1.1 데이터

실험에 활용한 데이터는 무비렌즈-1m(MovieLens-1m) 데이터로 6000여 명의 유저가 4000여 개의 영화에 대해 매긴 100만 여개의 평점 데이터이다. 이 데이터는 가장 기본적으로 활용되는 벤치마킹 데이터 중 하나이다. 데이터의 스키마는 표 4.1와 같다.

Table 4.1: 무비렌즈-1m 데이터 스키마

Attribute	**Type**	**Description**
user_id	int	유저에 대한 고유값
item_id	int	아이템(영화)에 대한 고유값
rating	int	영화에 대해 매긴 평점 (1~5점)
timestamp	int (UNIX Epoch time)	평점을 매긴 시점

무비렌즈-1m은 유저가 영화에 대해 매긴 평점과 평점을 매긴 시점이 기록된 데이터이며, 이를 행렬로 바꾸었을 때 밀도는 약 4.1%로 성긴 행렬이다. 또한 각 유저(아이템)에 대해 평점이 매긴(매겨진) 개수를 구하고 이를 분포로 나타내어 보면 그림 4.1와 같은데 유저와 아이템 모두에 대해 롱테일 분포(long-tail)를 보이고 있음을 알 수 있다

이처럼 무비렌즈-1m 데이터는 선행연구에서 언급했던 추천 시스템에 활용되는 데이터들이 가진 특성을 모두 가지고 있는 데이터이며, 따라서 이 데이터를 전반적인 추천 시스템에 대한 실험을 진행하기에 적합한 데이터로 판단하였다.

데이터는 학습 데이터와 검증 데이터 셋으로 나누고, REE를 학습할 때는 학습 데이터를 다시 학습-학습 데이터와 학습-검증 데이터로 나누어 사용하였다. 학습과 검증 데이터 셋을 나눌 때는 콜드 스타트를 고려하지 않기 위해 데이터 내에 존재하는 모든

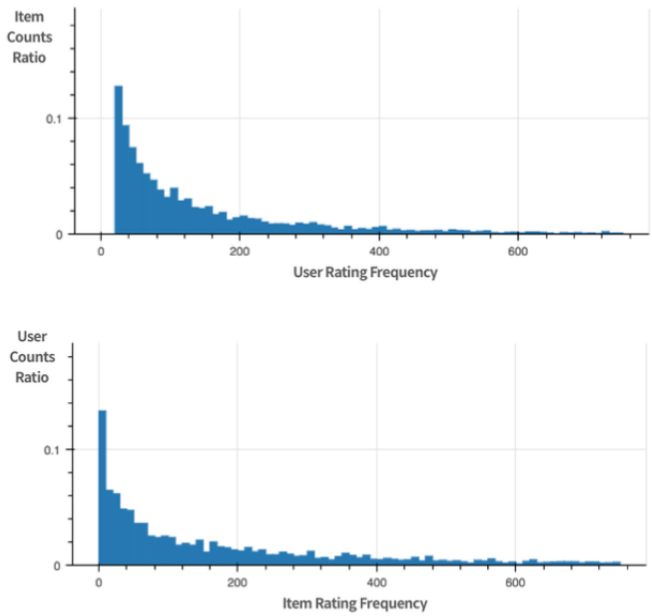


Figure 4.1: 무비렌즈-1m 유저와 아이템의 개수 분포

유저를 기준으로 각각 학습-검증 비율을 맞추어 데이터를 랜덤으로 샘플링하였다. 학습-학습, 학습-검증 데이터도 같은 방식을 적용하였다. 이에 따라 나누어진 데이터셋은 표 4.2와 같다.

Table 4.2: 데이터셋 구분

학습-학습데이터	학습-검증데이터	검증 데이터
867,572	96,397	36,240

4.1.2 추천 시스템 세팅

추천 시스템의 평점 예측에 사용한 모델은 행렬 분해를 활용한 협업 필터링 모델이며, 그 중에서도 비음수 행렬 분해 (Non-negative Matrix Factorizaiton)을 활용하였다. 이 모델은 분해된 행렬의 모든 값이 양수가 되도록 학습하는 것이며 그래디언트 하강

(Gradient Descent) 방법으로 잘 수렴하는 것으로 알려져 있다. 모델에 적용한 각종 파라미터들은 표 4.3와 같다.

Table 4.3: 추천 시스템(비음수 행렬 분해 모델) 세팅

Parameters	**Value**
k	10
Optimizer	Adam
Learning rate	0.03

4.1.3 모델 세팅

REE는 표 4.4와 같이 세팅하였다. REE에 사용한 정책망은 기본적인 3층 인공 신경망이며, 각 층은 15, 10, 5개의 노드로 구성된다. 정책망에 입력된는 상태(state)의 차원은 앞서 세팅한 추천 시스템에서 유저와 아이템의 특성 벡터 길이를 10으로 설정하였기에 이의 두 배인 20으로 설정하였다. 총 200번의 에피소드로 학습을 진행하였으며 최대 시간 간격은 2000으로 설정했으나 대부분 500에서 종료되었다. 가치(value) 계산을 위해 활용하는 할인율은 0.99로 설정하였다.

Table 4.4: REE 모델 세팅

Parameters	**Value**
State dimension	20
Policy Network	DNN(15-10-5)
Episode number	200
Max time step	2000
Discount factor	0.99

한편, REE의 성능을 평가하기 위해서 일반적으로 활용되는 평점 유도 전략들과 비교를 진행하였다. 이에 사용한 전략은 총 3가지로, 인기 기반(PopularityHeuristic), 엔트로피 기반(EntropyHeurisitic), 그리고 이항 예측(BinaryPredictionHeuristic)이다.

실험 변인 통제를 위해 실험에 사용한 추천 시스템 모델은 평점 유도 전략 모델에 사용되기 전에 이미 학습 데이터 내 초기 학습 데이터셋으로 세팅된 48,198개의 데이

터를 충분히 학습하여 초기값으로 설정하였다. 또한 모든 모델이 한 번에 한 유저에게 제안하고 응답을 받은 뒤 그 데이터를 하나의 배치로 현재 상태에서 한 번 더 학습하는 점진적 학습을 진행한다.

또한 모든 모델은 한 유저에게 20개의 데이터를 제안하며, 이 숫자는 유저가 한 번의 제안에 응답할 수 있을 것이라고 판단되는 실제적인 개수로 고려한 것이다.

4.2 실험 결과

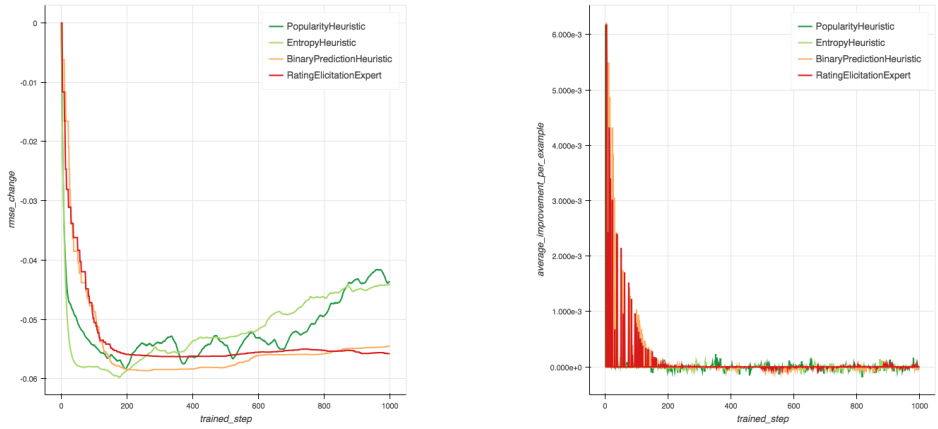
실험 결과는 세 가지의 측면에서 살펴보았다. 먼저 제안하고 수집된 데이터가 얼마나 학습에 도움이 되는지를 알아보기 위한 질적 측면과, 한 번에 제안한 데이터에서 얼마나 많은 데이터가 수집되는지를 알아보기 위한 양적 측면, 그리고 REE가 데이터를 선별하는 규칙이 일반적인 평점 유도 전략과 어떻게 다른지 규칙의 측면을 살펴보았다.

4.2.1 수집된 데이터의 질적 측면

REE가 수집한 데이터가 다른 모델들에 비해 질적으로 어떤 특성을 나타내는지를 분석하기 위해서는 REE가 수집한 데이터가 얼마나 성능 개선에 도움이 되었는지를 살펴보면 된다.

각 학습 시점에 따른 검증 데이터의 에러 감소량을 나타낸 그림 4.2a을 살펴보면, 학습 초반 REE는 인기 기반에 비해서는 월등히 낮고, 엔트로피 기반, 이항 예측 전략과 비슷한 수준의 감소를 보이다가 후반에 가서는 다른 모델들 보다 훨씬 감소량이 큰 경향을 보여준다. 이는 REE를 도입하는 것이 학습의 전 과정에서 다른 전략들에 비해 더 좋은 성능을 보인다는 것을 의미하며 따라서 데이터의 질적 측면이 좋은 것이라 볼 수 있다.

또한 그림 4.2b에서도 REE의 질적 측면이 드러나는데, 이 그림은 매 시점 응답된 데이터를 가지고 학습을 진행한 후, 검증 데이터의 성능이 개선되고 나면 그 개선된 양을 데이터의 수로 나누어 데이터 개수 당 평균적인 개선량으로 나타낸다. 여기서 REE는 학습 초반 이항 예측보다 조금 더 높은 평균 성능 개선량을 보인다. 또한 학습 후반에는 많은 데이터들이 성능이 떨어지는 경향을 보이는데, REE는 이에 비해 성능 감소율이 적은 편으로 나타난다.



(a) 학습 시점에 따른
검증 데이터의 에러 변화량

(b) 학습 시점에 따른
데이터당 평균 성능 개선량

Figure 4.2: 수집된 데이터의 질적 측면에 대한 분석을 나타내는 그래프

4.2.2 수집된 데이터의 양적 측면

학습 시점에 대한 평균적인 응답율은 표 4.5와 같다.

Table 4.5: 모델에 따른 평균 응답율

Model	Acquired ratio
PopularityHeuristic	39.09%
EntropyHeuristic	7.31%
BinaryPredictionHeuristic	1.40%
RatingElictationExpert	1.59%

응답율(acquired ratio)은 한 유저에게 제안된 20개의 아이템 중 유저가 실제로 응답한, 학습 필요 데이터에 존재하는 데이터의 개수에 대한 비율로 나타낸다. 인기 기반 전략의 경우 평균 40%로 다른 전략에 비해 월등히 높은 응답율을 기록했으며 이는 사용한 데이터가 어떠한 제안 없이 유저에 의해서만 쌓인 데이터라는 것을 생각했을 때 납득할만한 수치이다. 이에 반해 엔트로피 기반 전략은 평균 7%를 보였다.

반면 개인화 전략을 활용하는 이항 예측과 REE의 경우에는 평균 1% 대의 응답율을 보였다. REE의 경우에 학습 할 때 보상에 응답율을 고려하는 항이 들어간 것을 감안했

을 때, 예상보다 낮은 수치를 보인 것이라고 할 수 있다. 낮은 응답율은 실제로 모델의 학습에 필요한 데이터가 쌓이고 있지 않다는 것을 방증하는 것이기도 하며 REE가 질적 측면에서 보여준 성과를 본다면 제대로된 응답율 비교를 위해서는 실제로 유저에게 제안하고 그에 대한 응답을 보는 것이 타당하다고 판단된다.

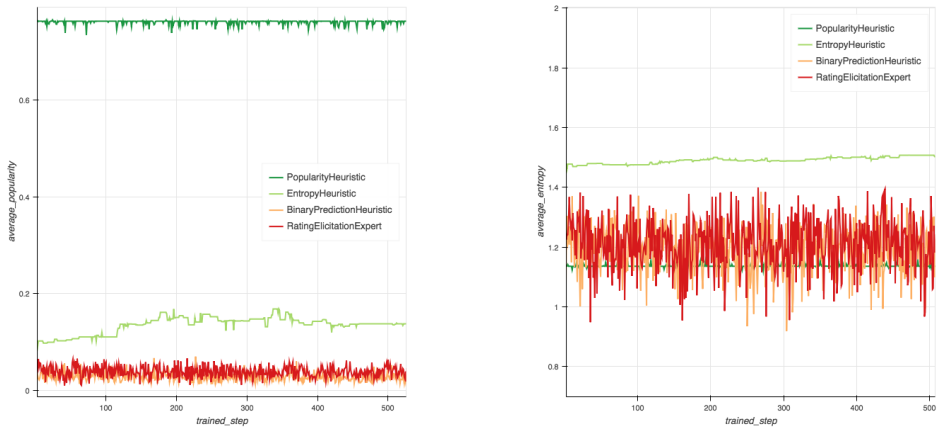
그럼에도 불구하고 여기서 알 수 있는 사실은 무조건 많은 데이터를 사용하는 것이 성능 개선에 큰 영향을 주지 않는다는 점이다. 인기 기반의 경우 REE에 비해 몇 십 배의 데이터를 확보할 수 있지만 그림 4.2에 나타난 것처럼 그 데이터들이 성능 개선에 큰 영향을 미치지 못한다는 것을 보여준다. 따라서 REE와 같이 성능에 개선을 줄 수 있는 데이터를 잘 선별한다면 그 응답되는 개수가 적더라도 큰 영향력을 미칠 수 있다는 사실을 알 수 있다.

4.2.3 데이터 제안 규칙

REE가 데이터를 선별하는 규칙을 살펴보기 위해서, 모든 모델들과 함께 각 시점별로 제안한 데이터가 당시에 추천 시스템 내에 존재하는 데이터들에 대비해서 어떠한 특성을 보이는지 살펴보았다.

이를 위해 제안된 데이터들의 평균 인기와 평균 엔트로피를 계산하였는데, 평균 엔트로피의 경우 엔트로피 기반 전략에서 사용하는 것과 동일한 식으로 계산하였으며 평균 인기의 경우 해당 시점에 평점 데이터가 가장 많은 아이템의 평점 수에 대비해서 제안된 아이템들의 평점 개수가 얼마나 존재하는지를 비율로 나타내어 계산하였다. 결과는 그림 4.3와 같다.

먼저 평균 인기를 살펴보면, REE는 3가지 전략 모델보다 훨씬 더 낮은 인기를 가진 아이템을 선택하는 것을 볼 수 있다. 이는 REE가 대다수에게 평가를 많이 받은 데이터보다 매우 적은 수의 평가를 받은 데이터를 선택하는 경향을 나타낸다고 볼 수 있다. 또한 평균 엔트로피의 경우에는 엔트로피 기반보다는 낮고, 인기 기반보다는 조금 높



(a) 평균 인기 (average popularity) (b) 평균 엔트로피 (average entropy)

Figure 4.3: 학습 시점에 따른 제안 아이템들의 특성

은 데이터들을 선택하는데, 이는 지나치게 평점이 다양한 아이템은 선택하지 않으려는 경향을 보인다는 것을 의미한다.

한편, 학습이 진행되면서 평균 인기와 평균 엔트로피의 변화 경향이 뚜렷하게 나타나지 않는데 이는 학습 과정에 따라 데이터를 선택하는 규칙이 변화하지는 않는다는 것을 의미한다. 이는 크게 두 가지로 해석될 수 있는데 실제로 REE가 데이터를 선택하는 기준은 학습 과정과 무관하게 데이터의 특성에 크게 의존한다는 것이고, 또 다른 하나는 평균 인기와 평균 엔트로피는 설명할 수 없는 기준에 따라서 학습 과정을 고려한 데이터 제안 규칙이 있다는 것이다. 후자의 경우 REE 학습 알고리즘의 특성상 데이터와 학습 과정을 데이터로 넣어 잠재적인 특성을 포착하도록 했다는 점에서 이해될 수 있으며, 따라서 제안 규칙을 이해하기 위해 인기 혹은 엔트로피와 같은 일반적인 척도로만으로는 설명하기 어렵다는 것을 의미하기도 한다.

제 5 장 결론

5.1 결론

본 논문에서는 추천 시스템 모델의 성능 향상을 위해 유저에게 데이터 입력을 제안하는 평점 유도 전문가 모델(REE)를 제안하였다. 기존에 추천 시스템 모델의 알고리즘을 발전시키는 관점에서 벗어나 수집하는 데이터 분포를 성능 향상에 도움되는 방향으로 변화시키는 관점의 모델이며, 이 때 모델은 미리 설정한 규칙에 따라 행동하는 것이 아니라 적용하고자하는 추천 시스템 모델의 학습 과정을 직접 학습하여 규칙을 학습하는 메타 러닝(meta-learning) 모델이다.

기존에 존재하던 평점 유도 전략들과 REE를 비교한 결과, 제안한 데이터에 대한 유저의 응답율은 상대적으로 다른 전략들에 비해 낮게 나타났지만, 수집된 데이터가 추천 시스템의 성능에 미치는 영향은 크게 나타났다. 즉, 적은 양의 데이터를 수집하더라도 추천 시스템의 성능에 결정적인 영향을 미치는 데이터를 선택하는 경향을 보여주었다. 이 데이터들은 추천 시스템 내에 평점 개수가 매우 적으면서 평점 분포가 어느 정도 퍼져있는 것이며 REE는 지속적으로 이러한 데이터들을 선택하는 경향을 보여준다.

이러한 성능을 지닌 REE는 추천 시스템의 모델에 상관 없이 적용할 수 있는 것이 가장 큰 장점이다. 또한 본 논문에서 제안한 학습 알고리즘은 유저가 자유 의사로 제공한 데이터에 대해서 적용될 수 있는 방법이기도 하기 때문에, 현재 서비스되고 있는 추천 시스템에 수집된 데이터를 가지고도 REE를 학습할 수 있다. 학습 알고리즘을 변형할 수도 있기 때문에 추천 시스템의 알고리즘 뿐 아니라 그 알고리즘이 가지고 있던 학습 알고리즘에 대한 종속성도 없는 모델이다.

추후 추천 시스템에 REE를 도입하고, 거기서 수집된 데이터로 REE를 추가 학습한다면 더욱 성능이 좋은 모델로 발전시킬 수 있을 것이라고 생각된다.

5.2 향후 연구

향후 본 논문에 대해 추가로 연구되어야 할 부분은 다음과 같다.

다양한 추천 시스템 학습에 대한 고려 본 논문에서는 점진적인 학습, 즉 온라인 러닝의 컨셉을 적용하였으나 오프라인 러닝을 진행하는 추천 시스템이 적지 않기 때문에 매 시점마다 오프라인 러닝을 진행하는 경우에 대해서도 연구가 필요하다.

콜드 스타트 문제 본 논문에서 제안한 학습 알고리즘은 새로운 유저 혹은 새로운 아이템이 시스템 내에 실시간으로 들어오는 경우에는 적용하지 못한다. 이를 해결하기 위해서는 REE에 입력되는 상태를 다르게 정의하는 것이 반드시 필요하다. 가장 쉬운 방법으로는 유저와 아이템의 특성을 내용(content)으로부터 뽑아내어, 신규 유저(혹은 아이템)를 추천 시스템의 학습 과정에 대한 종속성을 제거하는 방법이 있을 수 있을 것이다.

실제 서비스 중 테스트 기본적으로 본 논문에서 평가를 진행한 데이터는 자유 의사에 의해 수집된 데이터이다. 특히 제안-응답의 경우 이러한 데이터의 구성에 매우 취약하다. REE의 보상이 응답율에 기반하고 있음을 고려했을 때, 이러한 제안-응답에 대한 제대로된 평가를 위해서는 실제로 REE를 도입한 추천 시스템에서 유저를 대상으로 검증하는 것이 필요하다.

비동기적 제안-응답 상황 본 논문에서 가정한 동기적 제안-응답 상황은 실제 추천 시스템과는 큰 차이를 보인다. 이를 비동기적으로 보기 위해서는 데이터가 제안되고 그에 따른 응답으로 기록됨으로써 제안하고 응답된 시점이 분명하게 드러날 수 있어야 한다. 이는 앞서 실제 서비스 중 테스트와 마찬가지로 실제 제안-응답에 의해 데이터가

기록되어야 시도할 수 있을 것으로 생각된다.

제안 규칙을 설명할 새로운 척도의 도입 앞서 절에서 설명했던 REE의 제안 규칙을 제대로 분석하기 위해서는 기존의 평점 유도 전략에서 사용된 인기 혹은 엔트로피 이외에 새로운 척도가 필요할 것이라 생각된다. NDF에서는 데이터의 에러값, 즉 현재 모델이 데이터를 어려워하는 정도를 고려하기도 했다. 그러나 근본적으로 REE와 같은 모델이 학습한 규칙을 인간이 이해할 수 있을 수준으로 표현하는 것이 가능한지에 대해 연구가 먼저 진행되어야할 것이라 생각되며, 그 전에는 가능한한 많은 종류의 척도들을 토대로 다방면에서 분석하는 것이 필요해보인다.

참고 문헌

[1] G. Adomavicius and J. Zhang, *Impact of data characteristics on recommender systems performance*, ACM Transactions on Management Information Systems (TMIS), 3 (2012), p. 3.

[2] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, *Learning to learn by gradient descent by gradient descent*, in Advances in Neural Information Processing Systems, 2016, pp. 3981–3989.

[3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, *Curriculum learning*, in Proceedings of the 26th annual international conference on machine learning, ACM, 2009, pp. 41–48.

[4] M. Elahi, V. Repsys, and F. Ricci, *Rating elicitation strategies for collaborative filtering*, in International Conference on Electronic Commerce and Web Technologies, Springer, 2011, pp. 160–171.

[5] M. Elahi, F. Ricci, and N. Rubens, *A survey of active learning in collaborative filtering recommender systems*, Computer Science Review, 20 (2016), pp. 29–50.

[6] Y. Fan, F. Tian, T. Qin, J. Bian, and T.-Y. Liu, *Learning what data to learn*, arXiv preprint arXiv:1702.08635, (2017).

[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, *Neural collaborative filtering*, in Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.

[8] M. P. Kumar, B. Packer, and D. Koller, *Self-paced learning for latent variable models*, in Advances in Neural Information Processing Systems, 2010, pp. 1189–1197.

[9] D. D. Lee and H. S. Seung, *Algorithms for non-negative matrix factorization*, in Advances in neural information processing systems, 2001, pp. 556–562.

[10] Q. Liu, S. Wu, and L. Wang, *Deepstyle: Learning user preferences for visual recommendation*, in Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 841–844.

[11] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, *One-class collaborative filtering*, in Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 502–511.

[12] B. Settles, *Active learning literature survey*, Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[13] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, *Policy gradient methods for reinforcement learning with function approximation*, in Advances in neural information processing systems, 2000, pp. 1057–1063.

[14] Q. Zhao, D. Meng, L. Jiang, Q. Xie, Z. Xu, and A. G. Hauptmann, *Self-paced learning for matrix factorization.*, in AAAI, 2015, pp. 3196–3202.

[15] L. Zheng, V. Noroozi, and P. S. Yu, *Joint deep modeling of users and items using reviews for recommendation*, in Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, 2017, pp. 425–434.

Abstract

Meta-learning Model for Optimizing Rating Elicitation Strategies Based on Reinforcement Learning

Dongmin Shin
Department of Industrial Engineering
The Graduate School
Seoul National University

Most of the recommender system studies have improved the prediction performance by developing existing algorithms or suggesting new algorithms. However, when considering sparsity and long-tail distribution which are common characteristics of data for recommender system, it is necessary to select the data to be trained. In this thesis, we propose a Rating Elicitation Expert (REE) model. This model follows a kind of active learning method that can suggest items 1) to improve the performance of the system and 2) to increase possibility to offer rating data of users. REE is a meta learning model that changes the item suggestion rule by learning the learning process of the recommender system based on the rating histories, which is different from the conventional models that proposed the elicitation strategy from human intuition. Experimental results show that although the number of

data collected from REE is smaller than that of conventional strategies, the average performance improvement per datum is better than that of other strategies, even data selected by REE leads to best performance above all strategies. Also, REE has the advantage that it is applicable to any system since it is not dependent on the recommendation system model and learning method.

Keywords: Recommender system, Rating elicitation strategy, Active learning, Meta-learning, Reinforcement learning

Student Number: 2016-26635