Knowledge Distillation at Recommender Systems

2020.05.
20

임진혁

Knowledge Distillation at Recommender Systems

Paper:

Ranking Distillation: Learning Compact Ranking Models With High Performance for Recommender System

Select because

1) Citation is over 23

2) Given the fundamental limitations of KD we saw in the last review, we expect to see if the KD concept is really practical and worth doing.

3) It is expected to be able to grasp the standards, utilization, use, and experimental methods.(Public Data and SOTA model inside the recommendation system)

   More details:

   - Verification model at Rec
   - Verification data set at Rec
   - Possible to understand the composition of the sample experiment at Rec

해당 주제를 선택한 이유- 복선?

5 Bonus 발표자의 잡다한 생각들

T[4]. VAE나 Gan도 파라미터가 매우 크던데 이에 대해서도 KD가 가능할지 궁금하다.

T[5]. 특정 모델의 결과값을 사용함으로써 해당 모델의 성능을 갖게 되는게
NN이 아닌 다른 방법론을 NN에 이식하는 방법이 될 수 있는지 궁금하다. 예를 들어 DT의
결과로 NN을 학습시켜서 DT와 비슷한 결과를 내게 만드는게 가능할까?

Recommender System에서도 가능할지 궁금하다.
SAR의 결과를 사용해서 NN이 비슷한 성능을 갖게끔 유도하는게 가능할까?

관련 자료를 찾아보니 이미 Ranking Distillation이라는 논문이 존재한다.
정확하게 어떤 방식인지는 추후 읽어야겠다.

T[6]. MNIST에 대해 돌려볼 수 있는 KD 코드를 구하던지 구현해보던지
다음 발표자 발표 때까지 가져오도록 하겠습니다!

28

저번 발표에서 KD 개념 스터디- [Bonus]

Index

왜 추천시스템에서 KD 개념 도입이 필요할까?

추천시스템의 간략한 정의:

The core in such systems is a ranking model for computing the relevance score of each (q,d) pair for future use,
Where q is the query (keywords for webpage retrieval and user profile for recommender systems) and d is a document (a web page or item).

최근 Rec의 연구 동향
-> NN의 도입으로 인한 q<->d 의 interaction을
더 잘 잡아내서 더 좋은 성능 (latent space)

문제발생! SIZE(model parameter)가 너무 커져버려서 latency time 발생!

latency time 발생이 추천시스템에서 큰 문제가 되는 이유

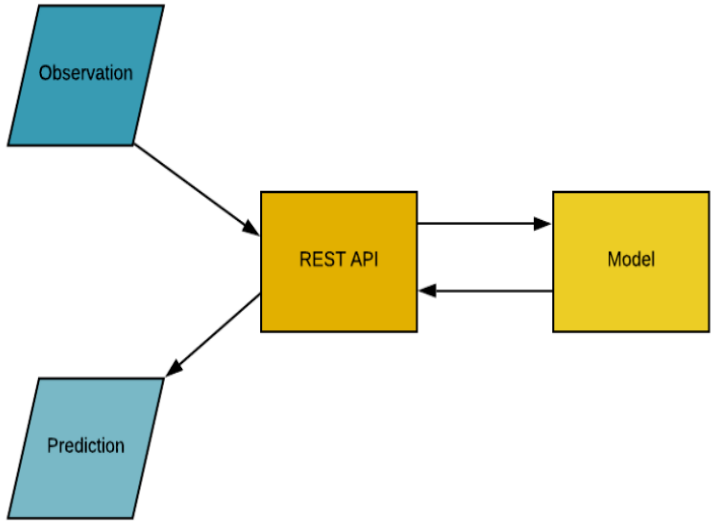


Figure 2. Online Inference

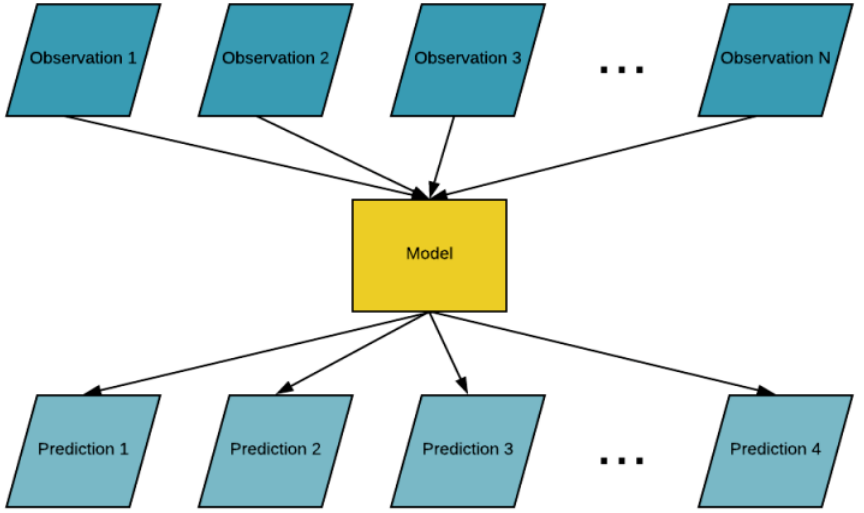


Figure 1. Batch Inference

online inference <-> batch inference

기계학습 모델 요청이 올 때마다 수행

하나의 인풋값마다 아웃풋이 나옴

유저가 원할 때마다 예측값 제공가능

배치 인퍼런스는 보통 배치 단위 (시간,일)로 예측

실시간이 불가능하고 새로운 데이터에 대한 예측이 불가능함

만약 일단위라면 그날 가입한 유저는 다음날부터 결과를 받을 수 있기 때문(넷플릭스)

레이턴시 요구사항을 신경써야함(시스템이 데이터를 받고,추론,검증,네트워크로 보내는걸 100ms안에 다 해야하기 때문)

따라서 최근 rec 연구는 성능과 효율 2가지 측면을 모두 만족시키
기 위한 방법론을 찾아가는 방향으로 발전하고 있음

Discrete hashing techniques

binary coding of model parameters

database-related method

pruning

등 등의 방법이 제안되었지만
한계점:

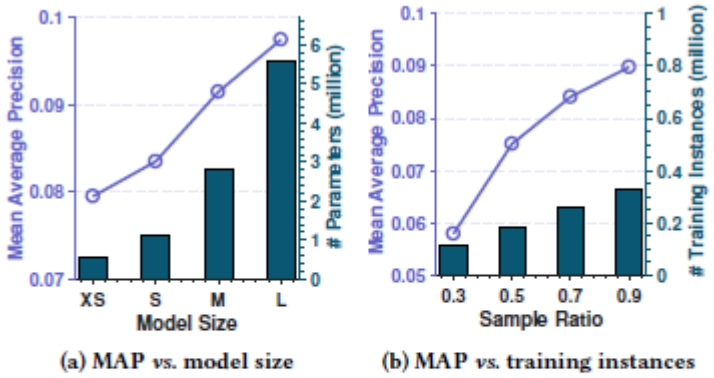
1. constraint에 기반한 효율성 증대 -> 성능 저하
2. 최적화에 기반하여 특정한 모델,데이터에서만 성능이 보장되어 확장성이 떨어짐. (Trade Off)

따라서! 본 연구는
성능 저하를 최대한 줄이면서 확장성이 있는 방법론을 제시해야한다. **:잘맞추면서 빠른 Rec**

해당 논문에서 제기하고 있는
추천시스템이 모두 적절한 능력을 갖춰야 하는 2가지 핵심 측면

Effectiveness how well the ranking model perform

Efficeinct how fast the systems will respond to user queries



(a) MAP *vs.* model size (b) MAP *vs.* training instances

Figure 2: Two ways of boosting mean average precision (MAP) on Gowalla data for recommendation. (a) shows that a larger model size in number of parameters, indicated by the bars, leads to a higher MAP. (b) shows that a larger sample size of training instances leads to a higher MAP.

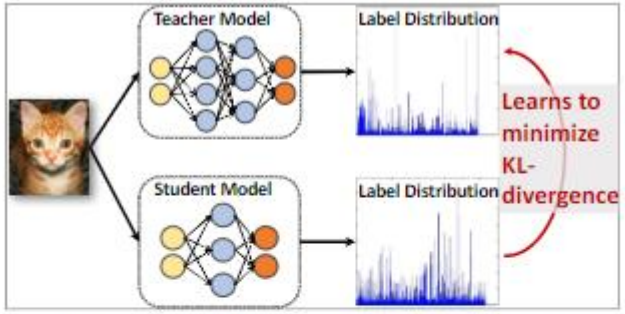
Trade off 관계에 있는 2가지 측면을 모두 충족시킬 수 있는 방법?

KD : 모델의 성능을 유지하면서 크기를 압축하는 방법론!

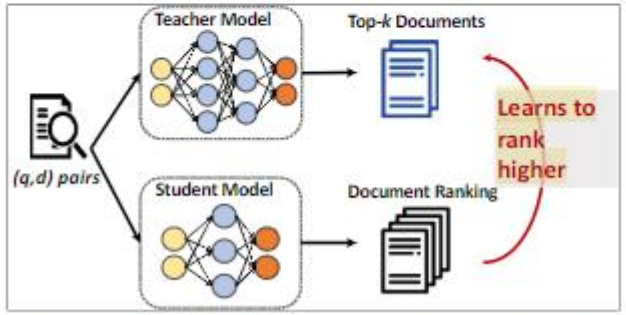
완전 딱 들어맞는다

따라서 Ranking problem에 KD 방법론을 도입하여 해결함으로써 (Online inference 가능)
RD 방법론으로 명명 (Ranking Distillation)

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법



(a) Knowledge Distillation for classification



(b) Ranking Distillation for ranking problems

KD의 기본적인 정의 –
Smaller student model is then trained by minimizing two deviations

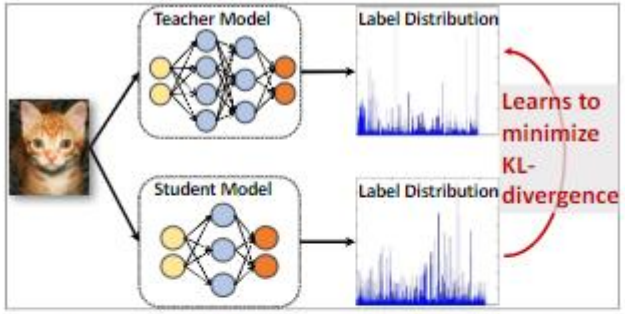
1) the deviation from the training set's ground-truth label distribution
2) the deviation from the label distribution generated by the teacher model

The student model trained with KD has an effectiveness comparable to that of the teacher model and can make more efficient online inference due to its small model size.

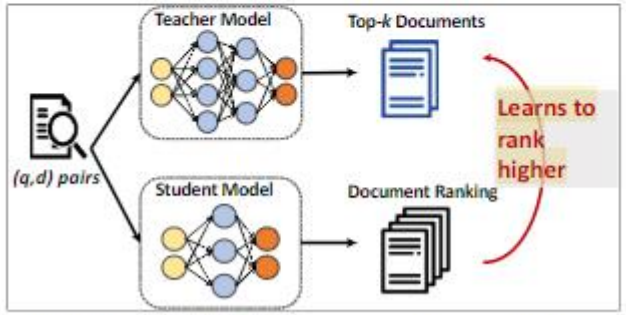
Rec에서 요구되는 2가지 핵심에 부합하는 KD의 특성

"Our objective is achieving the ranking performance of a large model with the online inference efficiency of a small model." **(1.2 Contributions 중)**

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법



(a) Knowledge Distillation for classification



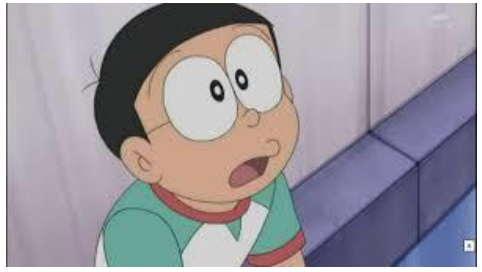
(b) Ranking Distillation for ranking problems

KD의 기본적인 정의 –
Smaller student model is then trained by minimizing two deviations

1) the deviation from the training set's ground-truth label distribution
2) the deviation from the label distribution generated by the teacher model

The student model trained with KD has an effectiveness comparable to that of the teacher model and can make more efficient online inference due to its small model size.

Rec에서 요구되는 2가지 핵심에 부합하는 KD의 특성



흠 뻔하네~
더 레전드 힐튼 님의 KD 방법론처럼
더 많은 파라미터로 높은 성능을 가지도록 Teacher Recommender을 학습시키고
1) Train data의 groud truth ranking (True Y)
2) Train data에 대한 TR(Teacher rec)의 예측 ranking(Prediction Y')
두 가지를 Student Rec에게 학습시키면 되는 거 아녀? 논문 더 볼 것도 없구마잉~

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

Ranking Distillation Flow

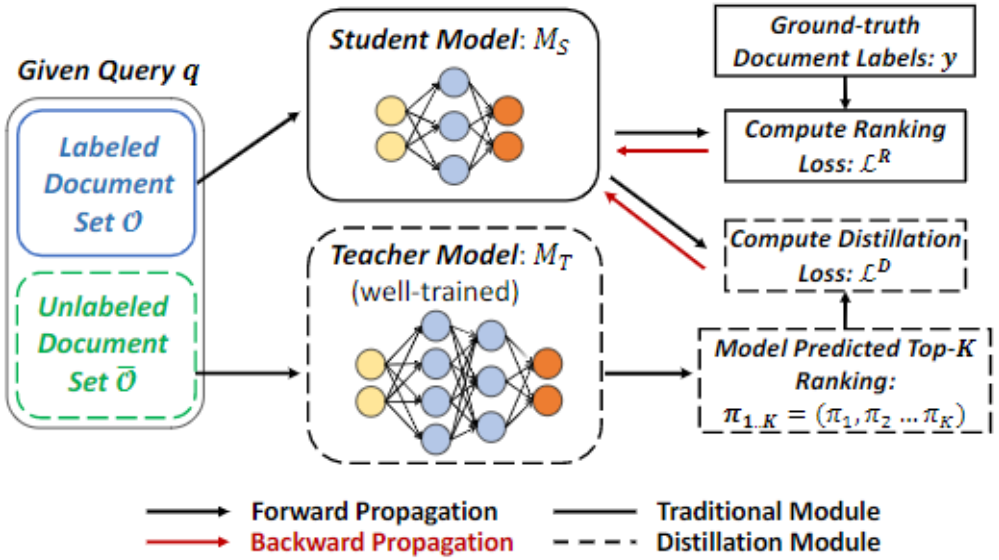


Figure 3: The learning paradigm with ranking distillation. We first train a teacher model and let it predict a top-K ranked list of unlabeled (unobserved) documents for a given query q. The student model is then supervised by both ground-truth ranking from the training data set and teacher model's top-K ranking on unlabeled documents.

KD를 그대로 바로 추천시스템(Ranking Problem)에 적용할 수 없는 이유

1. KD는 애초 "분류 " 문제를 위한 개념
(<-> REC: 상대적인 중요도의 "순서"를 예측)

2. KD는 query(q)에 대한 documents(d)의 distribution을
(해석-> 데이터(X)에 대한 라벨(Y)의 확률 연산)

teacher와 student가 모두 연산하는 것이 필요
(이것은 라벨이 비교적 매우 적기 때문에 (1000개:ImageNet)

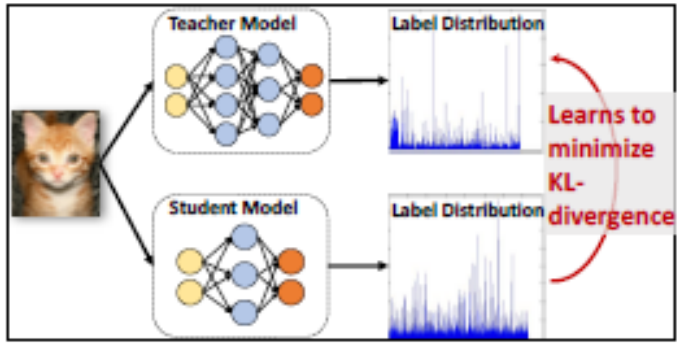
(<->REC: 라벨(documents = Item)이 수도없이 많음
(몇 만 ~ 몇 십 만의 라벨 -> 전부 다 도출 불가능!
또한, 각 q에 대한 가장 높은 rank의 d만 의미가 있음)

3. 시점에 따른 Sequential한 contex을 가진다.
(추천시점 전의 prior information(ex: user behavior)을
기반으로 output 생성)

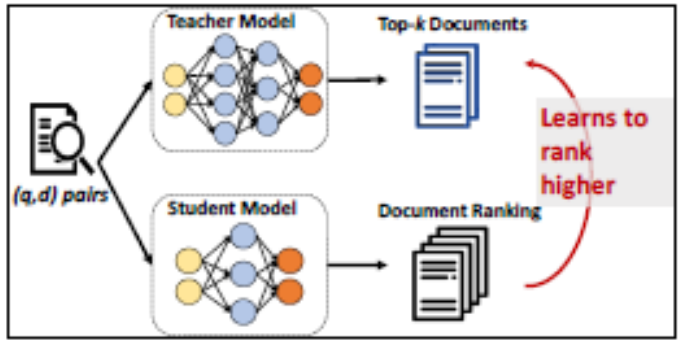
2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

그렇다면 KD와 RD의 실제적인 차이점은 무엇인가?

애초에 RD는 knowledge transfer와 learning to rank의 혼합된 개념!



(a) Knowledge Distillation for classification



(b) Ranking Distillation for ranking problems

Ground truth와 Teacher 모델의 Output
2가지 sources의 Information을 통해

[작은 용량]으로 [비슷한 성능]을 갖도록 하는 [압축]은 동일.

하지만

KD는 label이 있는 데이터에 또 다른 label 정보를 추가로 준다면
RD는 아예 additional data 자체를 추가로 준다는 차이점(label + data)

(이 부분에 대한 좀 더 명확한 설명 필요 –
오럴 발표도 그렇고 전체적으로 저자가 대충 말을 붙이는게 아닌가 싶다.
명확하게 단어의 뜻만으로 정확한 의도가 캐치가 안된다)

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

RD의 3가지 주요 Issue

1. Formulation
2. representation of teacher’s supervision
3. Balance between the trust on (the origin data and extra data)

[수식에 대한 이해]를 바탕으로 다음 3가지 Issue를 이해하고 해결 할 수 있음.

결과적으로
RD를 통해 오히려 Teacher Rec보다 Student Rec가 더 작은 용량으로 더 좋은 성능을 보이기도 하였음

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

Notation (수식 정확하게 타이핑 하는 것 찾아야함)

Given a set of queries Q={q1,· · · ,q|Q| }

A set of documentsD={d1,· · · ,d|D| }

The degree of relevance for a query-document pair (q,d)

a relevance score y(q)d is labeled by human (or statistical results) as ground-truth

Ranking model M(q,d; θ) = yˆ(q)d (모델 파라미터)

$$\theta^* = \arg\min_{\theta} \sum_{q \in Q} \mathcal{L}^R(\mathbf{y}^{(q)}, \hat{\mathbf{y}}^{(q)}).$$

$$\mathcal{L}^R(\mathbf{y}, \hat{\mathbf{y}}) = -\Big(\sum_{d \in \mathbf{y}_{d+}} \log(P(rel = 1|\hat{y}_d)) + \sum_{d \in \mathbf{y}_{d-}} \log(1 - P(rel = 1|\hat{y}_d)) \Big),$$

$$\mathcal{L}^R(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{d_i, d_j \in C} \log(P(d_i \succ d_j | \hat{y}_i, \hat{y}_j)),$$

Loss에 대한 접근

1. point-wise (값 자체)
2. pairwise (값들의 순서)
3. list-wise

$$= \sum_{i=1}^{n} \big(y_i \log(\pi) + (1 - y_i) \log(1 - \pi) \big)$$

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

Ranking Distillation Flow

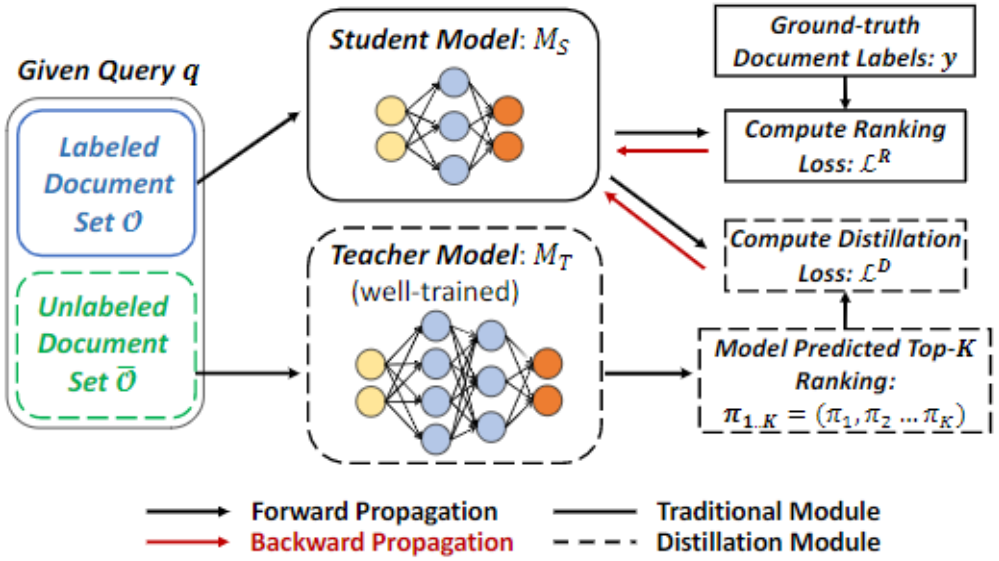


Figure 3: The learning paradigm with ranking distillation. We first train a teacher model and let it predict a top-K ranked list of unlabeled (unobserved) documents for a given query q. The student model is then supervised by both ground-truth ranking from the training data set and teacher model's top-K ranking on unlabeled documents.

$$\mathcal{L}(\theta_S) = (1-\alpha)\mathcal{L}^R(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \alpha\mathcal{L}^D(\boldsymbol{\pi}_{1..K}, \hat{\boldsymbol{y}}).$$

relevance scores of the teacher model M_T for

unlabeled documents
$O' = \{d : y_d = \emptyset\}$

top-K unlabeled document ranking
$\pi 1..K = (\pi 1, \ldots, \pi K)$

$\pi r \in D$
r -th document in this ranking

Teacher Rec : π1. .k
$\hat{y}$ is the student model's predicted scores

α is the hyperparameter
used for balancing these two losses

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

Distillation Loss Formalization

$$\mathcal{L}^D(\pi_{1..K}, \hat{\mathbf{y}}) = -\sum_{r=1}^{K} w_r \cdot \log(P(rel = 1|\hat{y}_{\pi_r}))$$

$$= -\sum_{r=1}^{K} w_r \cdot \log(\sigma(\hat{y}_{\pi_r})),$$

Algorithm 1 Estimate Student's Ranking for π_r

Require: Student Model $M_S(q, d; \theta_S)$, unlabeled document set $\bar{O}$
for a given query q and the hyper-parameter ϵ
$\hat{y}_{\pi_r} \leftarrow M_S(q, \pi_r; \theta_S)$
Initialize $n = 0$
for $t = 1, 2, ...\epsilon$ **do**
  Sample a document d from $\bar{O}$ without replacement
  $\hat{y}_d \leftarrow M_S(q, d; \theta_S)$
  if $\hat{y}_d > \hat{y}_{\pi_r}$ **then**
    $n \leftarrow n + 1$
  end if
end for
$\hat{r}_{\pi_r} \leftarrow \lfloor \frac{n \times (|\bar{O}|-1)}{\epsilon} \rfloor + 1$
return $\hat{r}_{\pi_r}$

$$\mathcal{L}^D(\pi_{1..K}, \hat{\mathbf{y}}) = -\sum_{r=1}^{K} w_r \cdot \log(P(rel = 1|\hat{y}_{\pi_r}))$$

$$= -\sum_{r=1}^{K} w_r \cdot \log(\sigma(\hat{y}_{\pi_r})),$$

Weighting by Position Importance

$$w_r^a \propto e^{-r/\lambda} \quad \text{and} \quad \lambda \in \mathbb{R}^+,$$

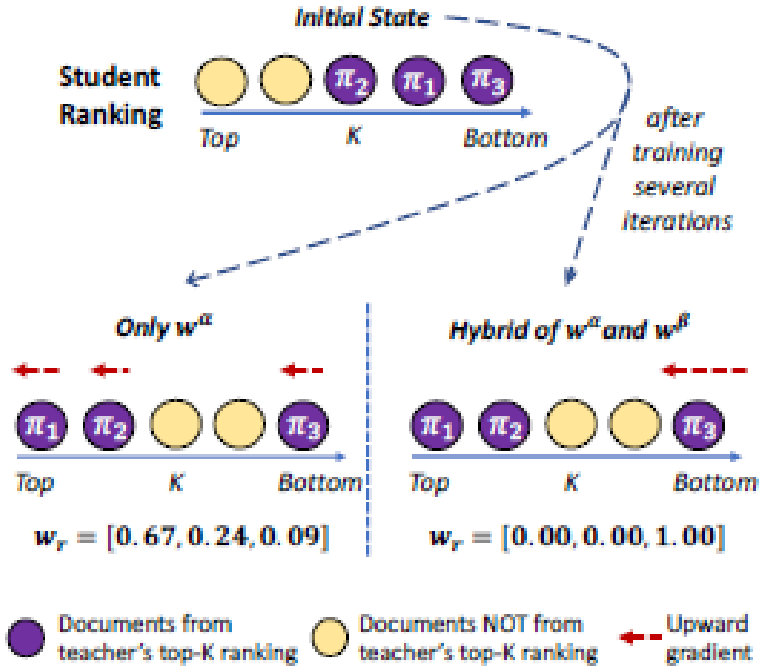
Weighting by Ranking Discrepancy

$$w_r^b = tanh(\max(\mu \cdot (\hat{r}_{\pi_r} - r), 0)),$$

$$w_r = (w_r^a \cdot w_r^b)/(\sum_{i=1}^{K} w_i^a \cdot w_i^b).$$

2 Proposed Approach 논문에서의 추천시스템에 KD 도입 방법

Distillation Loss Formalization



Hybrid Weighting Scheme.

$$w_r = (w_r^a \cdot w_r^b)/(\sum_{i=1}^{K} w_i^a \cdot w_i^b).$$

$$\mathcal{L}^D(\pi_{1..K}, \hat{\mathbf{y}}) = -\sum_{r=1}^{K} w_r \cdot \log(P(rel = 1|\hat{y}_{\pi_r})) = -\sum_{r=1}^{K} w_r \cdot \log(\sigma(\hat{y}_{\pi_r})),$$

$$\mathcal{L}(\theta_S) = (1-\alpha)\mathcal{L}^R(\mathbf{y}, \hat{\mathbf{y}}) + \alpha\mathcal{L}^D(\pi_{1..K}, \hat{\mathbf{y}}).$$

3 Experiments

데이터

Table 1: Statistics of the data sets

Datasets	#users	#items	avg. actions per user	$(u, S^{(u,t)})$ pairs	Sparsity
Gowalla	13.1k	14.0k	40.74	367.6k	99.71%
Foursquare	10.1k	23.4k	30.16	198.9k	99.87%

Evaluation Metrics. Precision@n (Prec@n), nDCG@n, and Mean Average Precision (MAP

Datasets	Model	Time (CPU)	Time (GPU)	#Params	Ratio
Gowalla	*Fossil-T*	9.32s	3.72s	1.48M	100%
	Fossil-RD	4.99s	2.11s	0.64M	43.2%
	Caser-T	38.58s	4.52s	5.58M	100%
	Caser-RD	18.63s	2.99s	2.79M	50.0%
Foursquare	*Fossil-T*	6.35s	2.47s	1.01M	100%
	Fossil-RD	3.86s	2.01s	0.54M	53.5%
	Caser-T	23.89s	2.95s	4.06M	100%
	Caser-RD	11.65s	1.96s	1.64M	40.4%

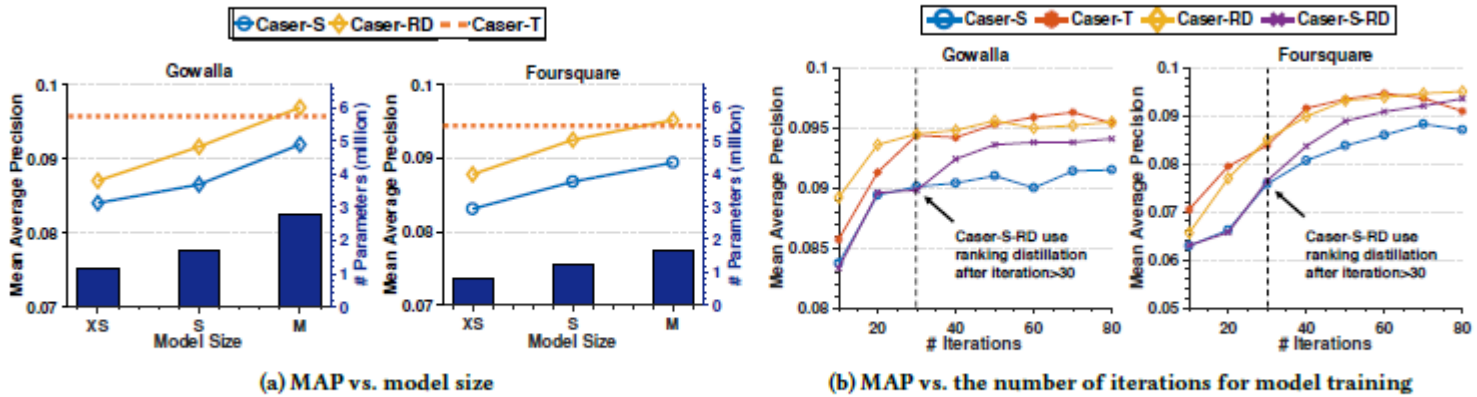
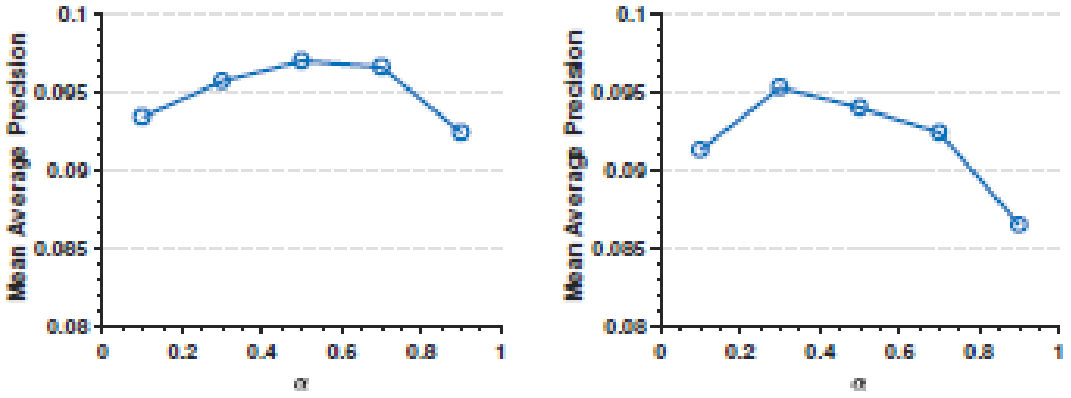


Figure 5: Mean average precision vs. (a) model size and (b) the choice of distillation loss.

3 Experiments



(a) Gowalla (b) Foursquare

Figure 6: MAP vs. balancing parameter α

Gowalla

Model	Prec@3	Prec@5	Prec@10	nDCG@3	nDCG@5	nDCG@10	MAP
Fossil-T	0.1299	0.1062	0.0791	0.1429	0.1270	0.1140	0.0866
Fossil-RD	0.1355	0.1096	0.0808	0.1490	0.1314	0.1172	0.0874
Fossil-S	0.1217	0.0995	0.0739	0.1335	0.1185	0.1060	0.0792
Caser-T	0.1408	0.1149	0.0856	0.1546	0.1376	0.1251	0.0958
Caser-RD	0.1458	0.1183	0.0878	0.1603	0.1423	0.1283	0.0969
Caser-S	0.1333	0.1094	0.0818	0.1456	0.1304	0.1188	0.0919
POP	0.0341	0.0362	0.0281	0.0517	0.0386	0.0344	0.0229
ItemCF	0.0686	0.0610	0.0503	0.0717	0.0675	0.0640	0.0622
BPR	0.1204	0.0983	0.0726	0.1301	0.1155	0.1037	0.0767

Foursquare

Model	Prec@3	Prec@5	Prec@10	nDCG@3	nDCG@5	nDCG@10	MAP
Fossil-T	0.0859	0.0630	0.0420	0.1182	0.1085	0.1011	0.0891
Fossil-RD	0.0877	0.0648	0.0430	0.1203	0.1102	0.1023	0.0901
Fossil-S	0.0766	0.0556	0.0355	0.1079	0.0985	0.0911	0.0780
Caser-T	0.0860	0.0650	0.0438	0.1182	0.1105	0.1041	0.0941
Caser-RD	0.0923	0.0671	0.0444	0.1261	0.1155	0.1076	0.0952
Caser-S	0.0830	0.0621	0.0413	0.1134	0.1051	0.0986	0.0874
POP	0.0702	0.0477	0.0304	0.0845	0.0760	0.0706	0.0636
ItemCF	0.0248	0.0221	0.0187	0.0282	0.0270	0.0260	0.0304
BPR	0.0744	0.0543	0.0348	0.0949	0.0871	0.0807	0.0719

Q&A