Self-Attentive Sequential Recommendatiaon (ICDM’ 18)

M2023089 이영준

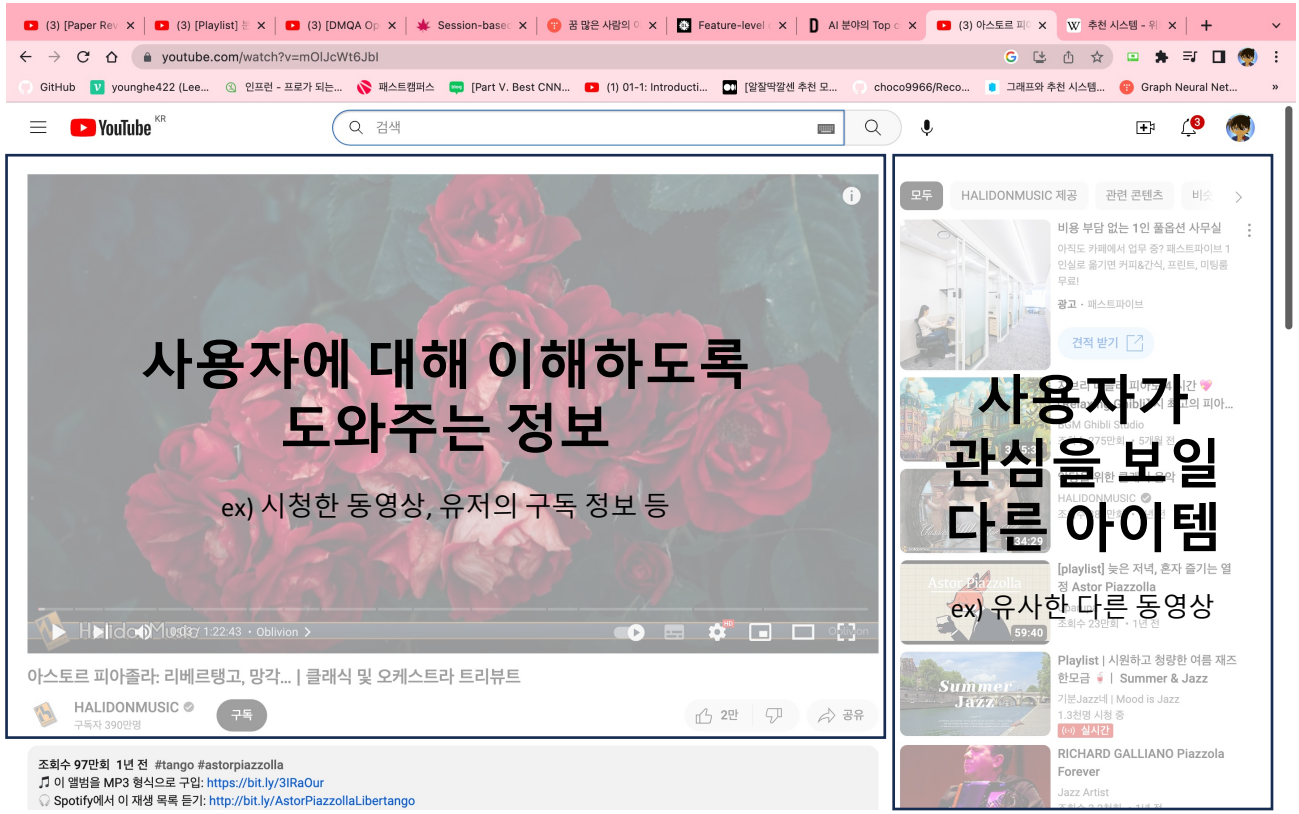
Index

Introduction

1.1. Introduction to Recommendation System

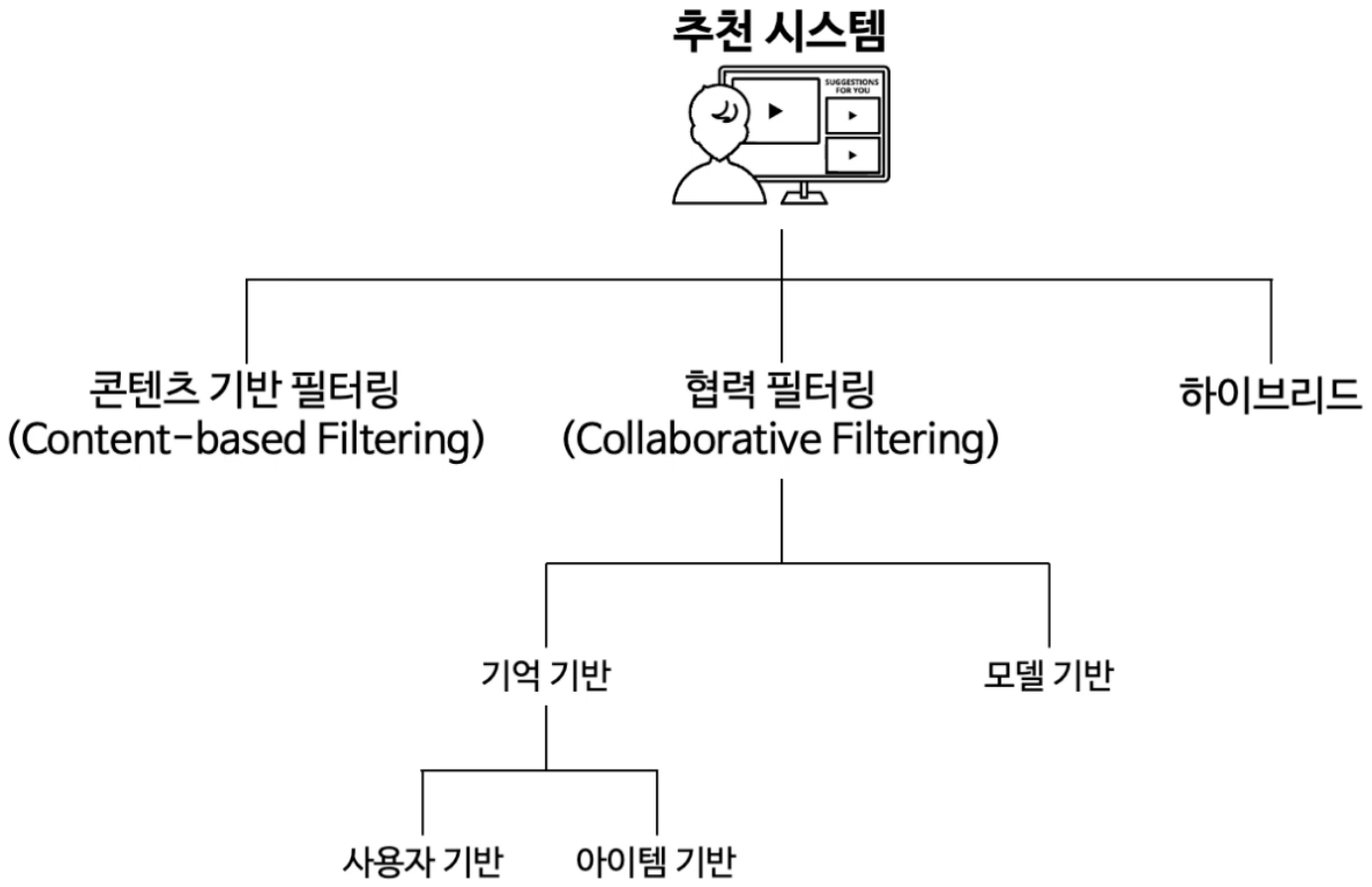
What is Recommendation System?

- 사용자에 대해 이해하도록 도와주는 정보를 통하여
- 사용자가 관심을 보일 다른 아이템을 추천해 주는 것을 의미한다.



1.1. Introduction to Recommendation System

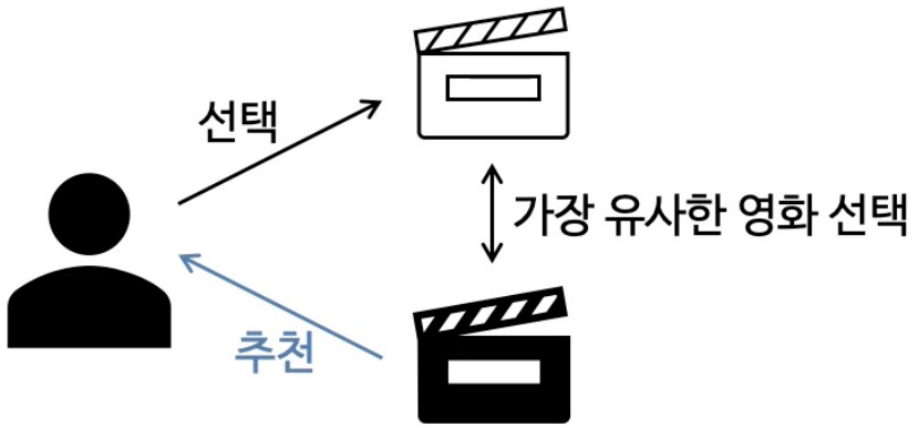
What is Recommendation System?



1.1. Introduction to Recommendation System

What is Recommendation System? - Contents based Filtering

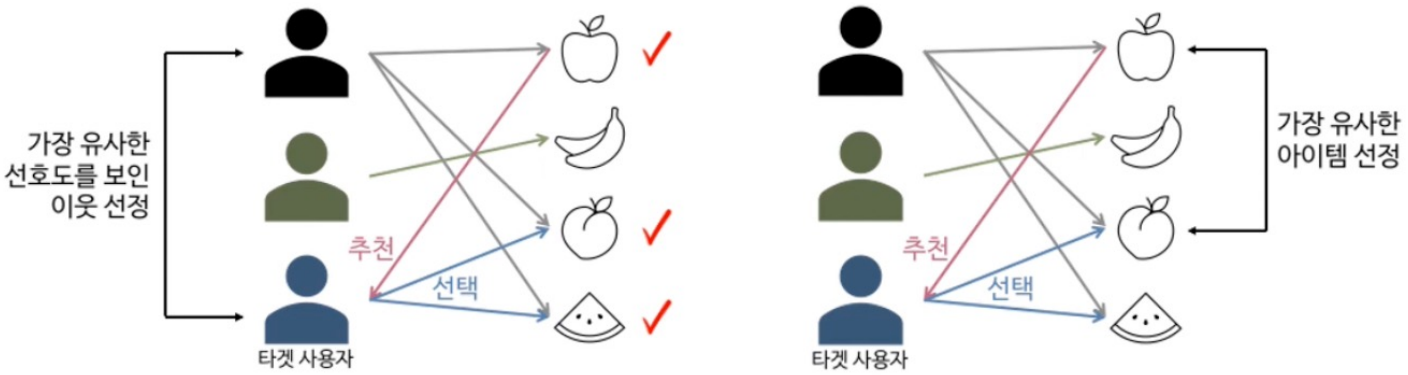
- 콘텐츠 기반으로 분석해서 추천해주는 방식
- 특정 아이템의 메타데이터를 기반으로 선호 아이템 파악
- 선호하는 아이템과 가장 유사한 다른 아이템 추천



1.1. Introduction to Recommendation System

What is Recommendation System? - Collaborative Filtering (Based Memory)

- ‘특정 아이템에 대해 선호도가 유사한 고객들은 다른 아이템에 대해서도 비슷한 선호도를 보일 것이다.’ 라는 가정에서 출발함
- 사용자 혹은 아이템간 유사도를 기반으로 선호도 예측
- 크게 아이템 기반 협업 필터링과 사용자 기반 협업 필터링으로 나뉘어짐



1.1. Introduction to Recommendation System

What is Recommendation System? - Collaborative Filtering (Based Memory)

- 아이템 기반 협력 필터링 :

두 아이템 유사

	User 1	User 2	User 3	User 4	User 5
Item A	5	4	4		5
Item B	4	4	4	5	
Item C	1	1	2	3	

평점분포를 바탕으로, Item A와 B가 유사하다고 판단

따라서 Item B를 높게 평가한 User 4에게 Item A를 추천.

- 사용자 기반 협력 필터링 :

두 사용자 유사

	Item A	Item B	Item C	Item D	Item E
User 1	3	4	4		5
User 2	4	4	4	1	
User 3	1	1	2	5	

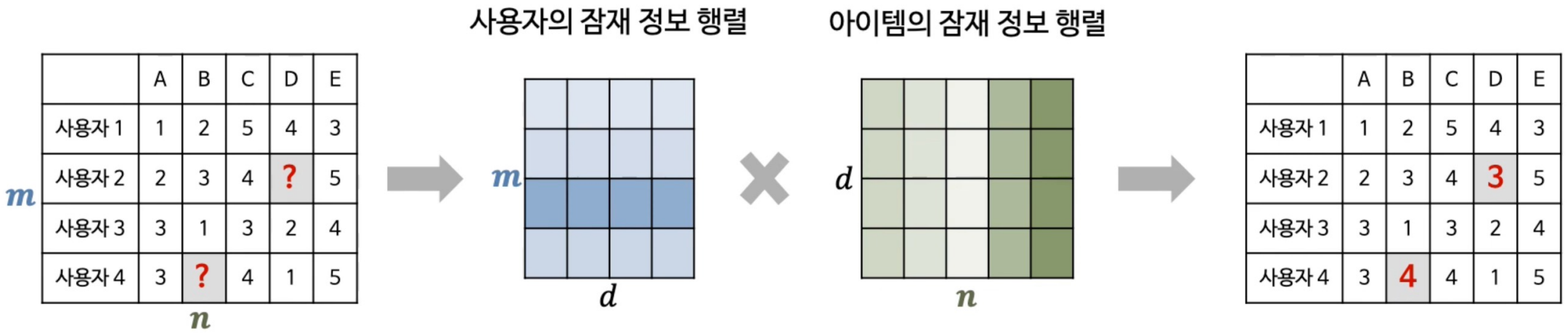
평점분포를 바탕으로, User 1와 2가 비슷한 성향을 가지고 있다고 판단

User 1이 Item E를 높게 평가하였으므로, 비슷한 성향을 가진 User 2에게도 이를 추천

1.1. Introduction to Recommendation System

What is Recommendation System? - Collaborative Filtering (Based Model)

- **Matrix Factorization**
  - '사용자와 아이템 사이에는 사용자의 행동과 평점에 영향을 끼치는 잠재적 특성이 있을 것이다' 라는 가정



	A	B	C	D	E
사용자 1	1	2	5	4	3
사용자 2	2	3	4	?	5
사용자 3	3	1	3	2	4
사용자 4	3	?	4	1	5

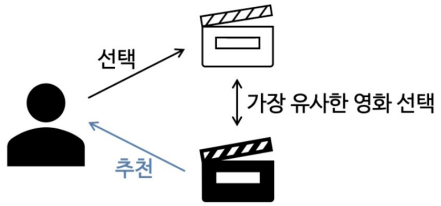
	A	B	C	D	E
사용자 1	1	2	5	4	3
사용자 2	2	3	4	3	5
사용자 3	3	1	3	2	4
사용자 4	3	4	4	1	5

*예시 : 4명의 사용자가 5개의 아이템에 대한 평점을 매긴 경우

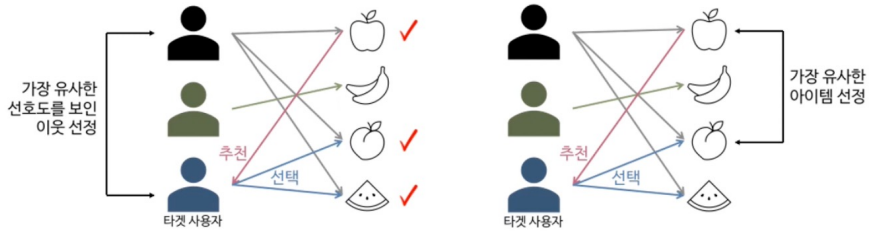
1.2. Introduction to Sequential Recommender

What is Sequential Recommender

- 사용자의 과거 아이템 선택의 정보가 동일하게 중요하다는 기본 가정에서 출발한 추천 시스템



Contents-based Filtering



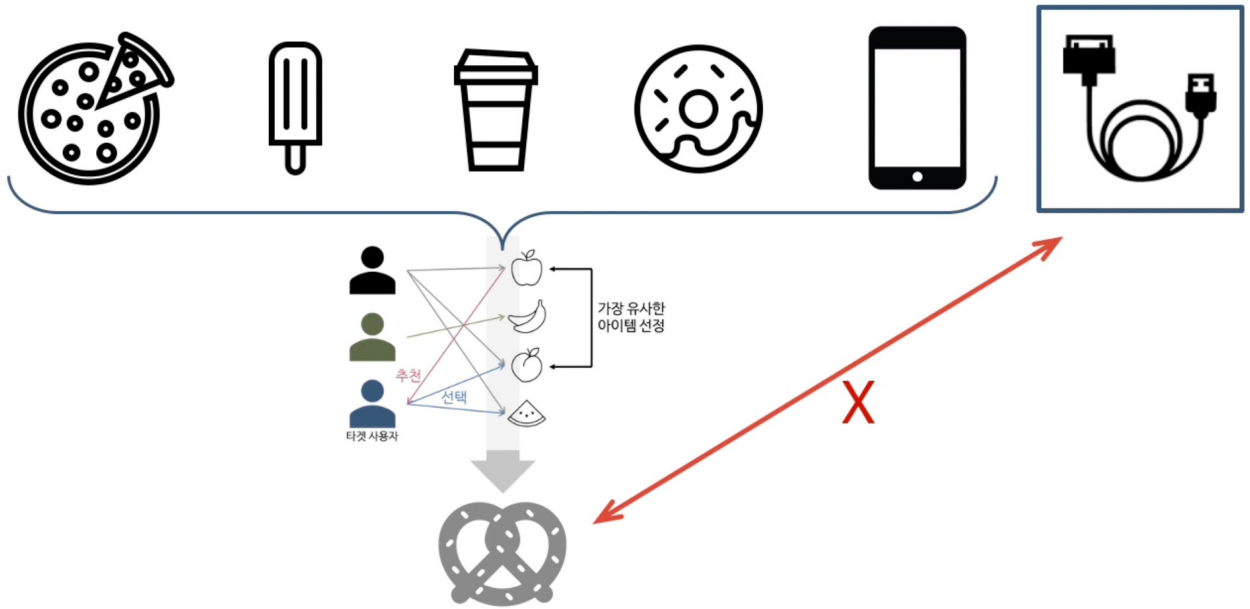
Collaborative Filtering

실제로 사용자가 아이템을 선택할 때에는
과거 구매 정보가 동일하게 중요할까?

1.2. Introduction to Sequential Recommender

What is Sequential Recommender

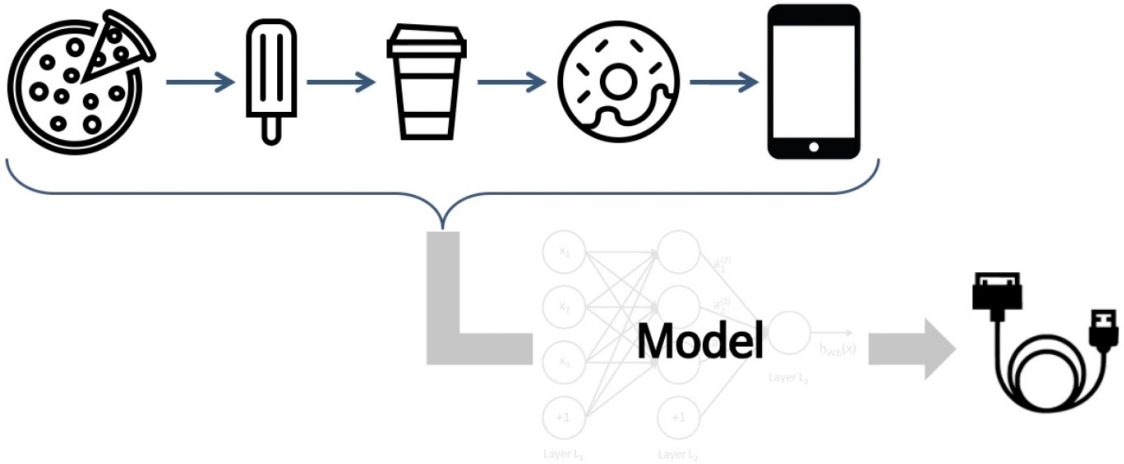
- 주로 선택하던 아이템과 다른 아이템이 등장할 경우 이에 대한 설명 부족
- 협력 필터링 등 보편적으로 사용되는 추천시스템 모델에서는 사용자가 주로 사용하던 아이템 추천



1.2. Introduction to Sequential Recommender

What is Sequential Recommender

- 사용자의 선호도, 관심은 끊임없이 변화하고 발전한다는 아이디어에서 시작
- 조금 더 실제 사용자의 관심사를 잘 반영하는, 변화의 패턴까지도 잡아낼 수 있는 모델을 만들어보자
- 과거 행동들 에서 유의미한 순차적 패턴을 찾고, 최근 아이템에 더 집중하여 다음 아이템을 추천한다



Self-Attentive Sequential Recommendation

2.1. Structure

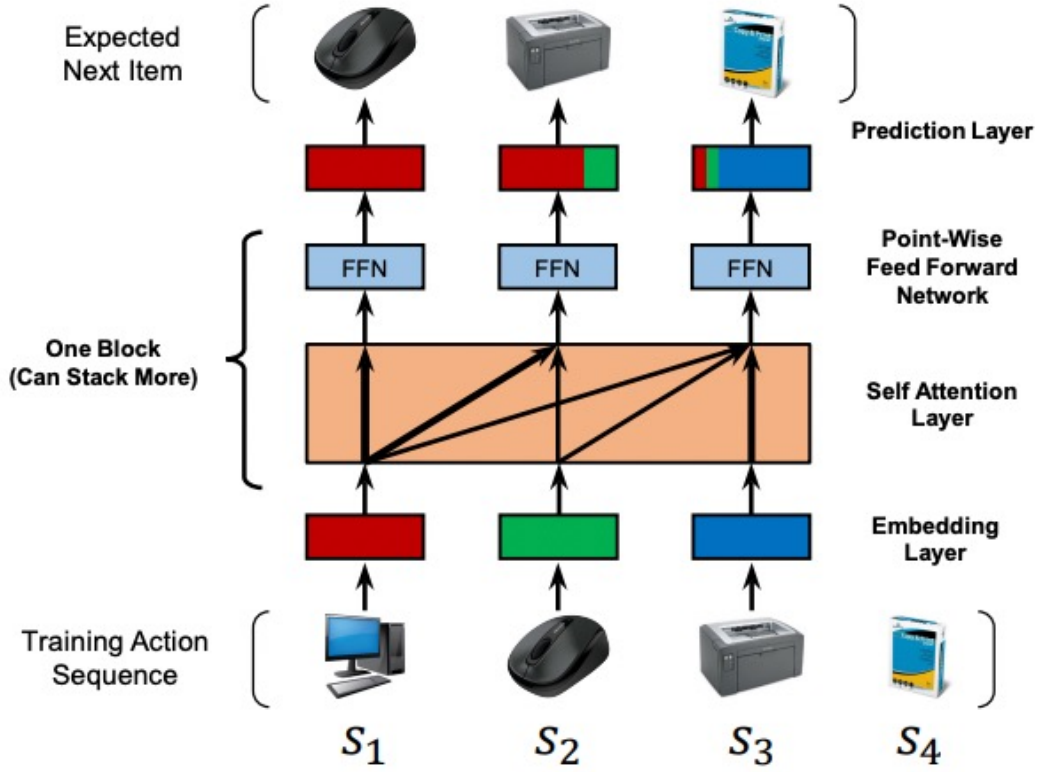
기존 Rnn 기반 모델의 단점 : 순차적 계산

- 매 Step 마다 모델 학습
- Long Sequence 계산 시 시간 소요 증가
- Long Term의 정보를 반영하기 힘듦

2.1. Structure

기존 Rnn 기반 모델의 단점 : 순차적 계산

- 매 Step 마다 모델 학습
- Long Sequence 계산 시 시간 소요 증가
- Long Term의 정보를 반영하기 힘듦



2.2. Embedding Layer

	Input Sequence						
시점(t)	1	2	3	4	···	t−1	t
		Output Seqeunce					

Item dimension : d차원 , random init

길이가 너무 길 경우 : 최근 n개 선택

짧을 경우 : n개 전 0 padding

하나의 유저의 history 를 time stamp 에 따라 순차 정렬함(i.e. 시간을 고려 x , 순서만고려)

Index 를 가지고 있음

User 1	5	2	3	2	10	4

n

User u	pad	pad	3	2	6	10

$User\ u : \mathbb{R}^{n \times d}$

Item Matrix

index	Look up table(vector)
1	D차원
2	
3	
...	
#item num	

2.2. Embedding Layer

User 1	5	2	3	2	…	4
position	1	2	3	4	…	n

$$User\ seq\ E : Item + Position(learnable)$$
$$\mathbb{R}^{n \times d} \qquad \mathbb{R}^{n \times d}$$

Item Matrix

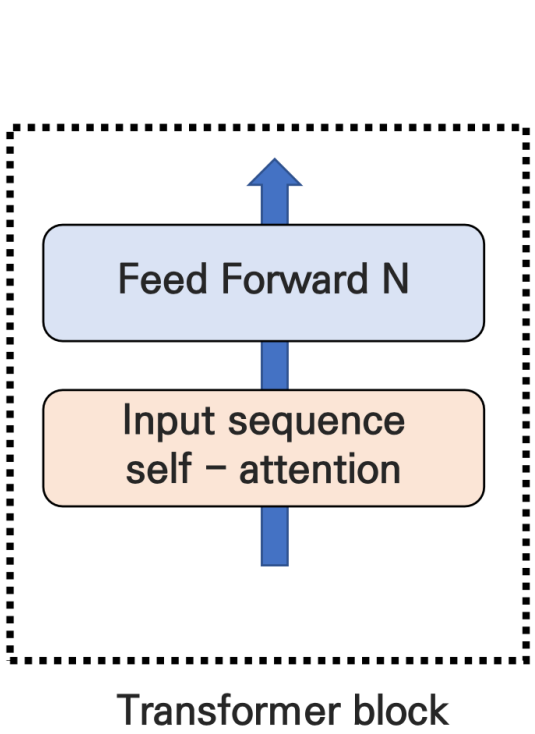
index	Look up table(vector)
1	D차원
2	
3	
…	
#item num	

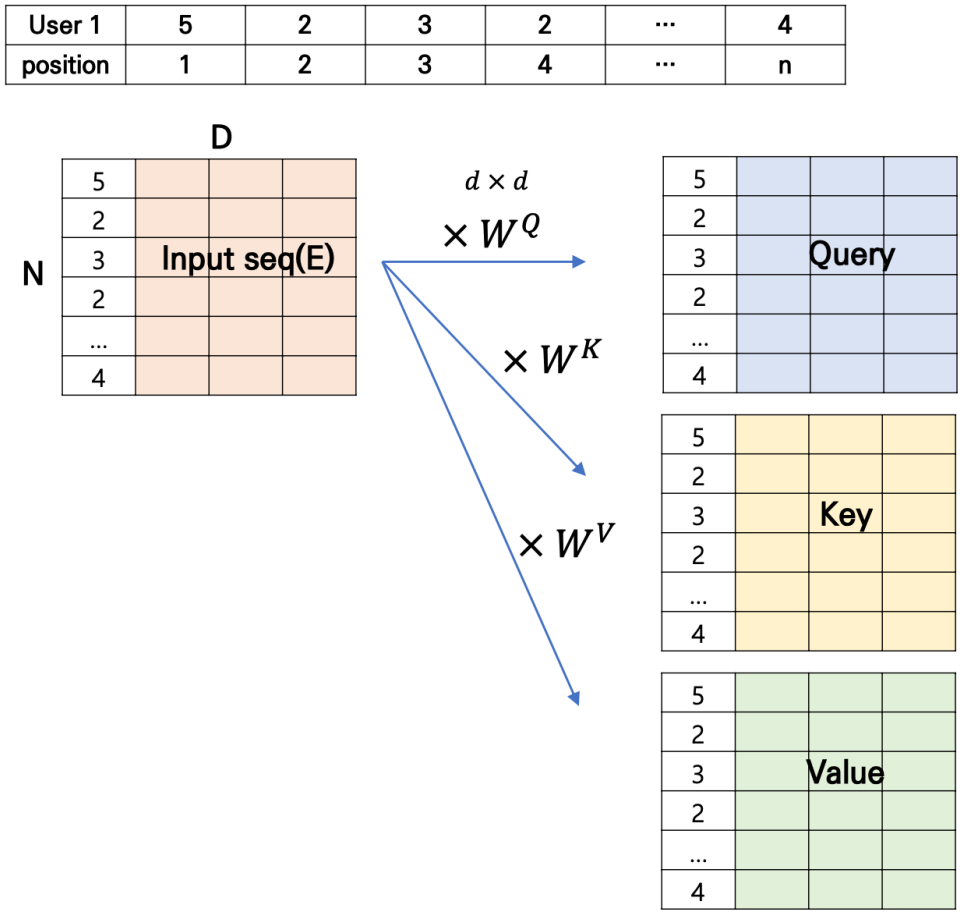
Position Matrix

index	Look up table(vector)
1	D차원
2	
3	
…	
n (max len)	

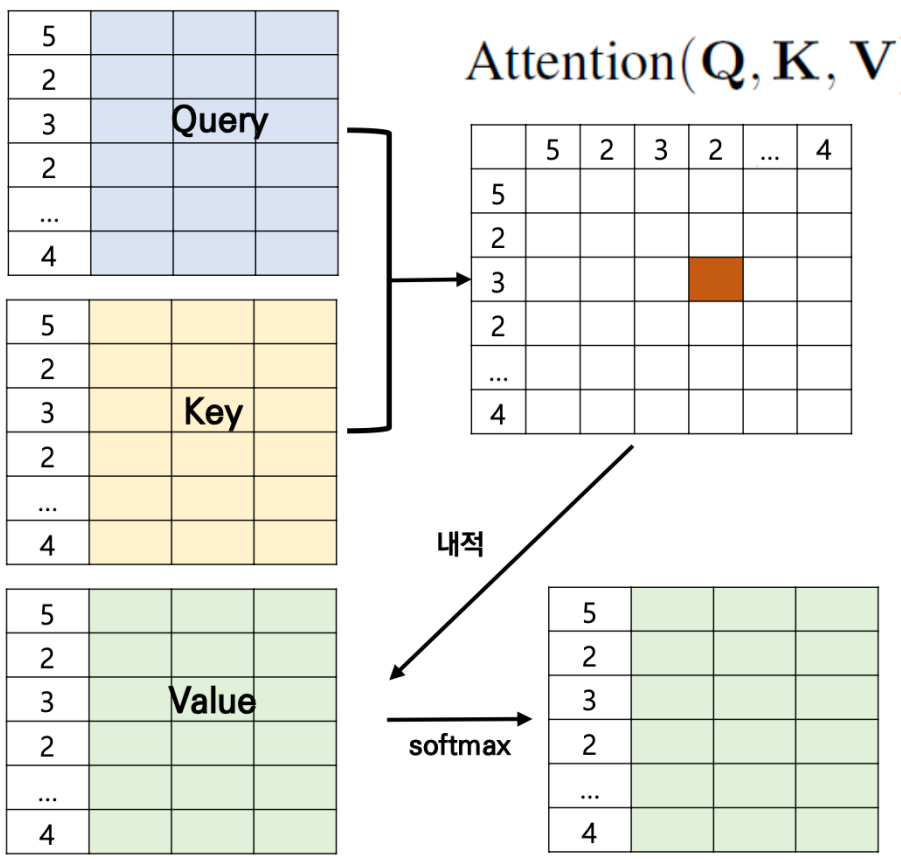
✓ 실험적 결과 : Fixed position embedding 을 사용 해봤으나 성능이 떨어짐

2.3. Self-Attention Layer





2.3. Self-Attention Layer



$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$

Key와 Query의 관계 matrix → 아이템 간 상관관계

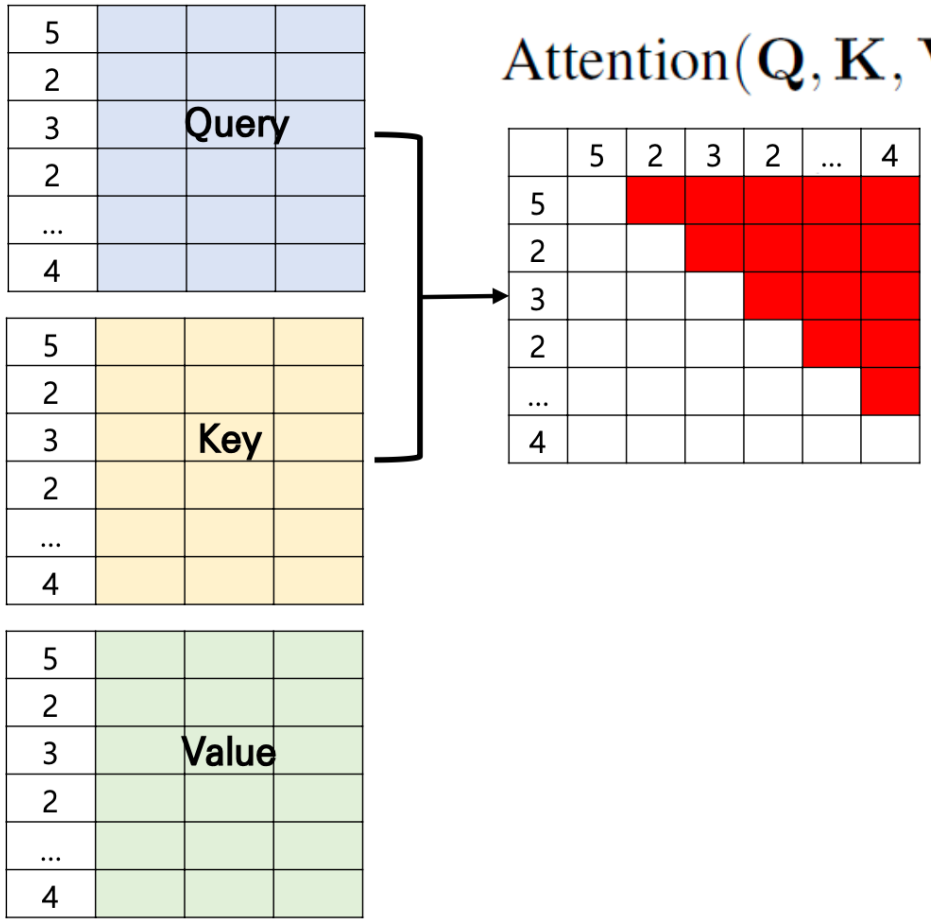
Inner product 시 너무 큰 값이 나오는 것을 방지하기 위해, scale parameter d 사용

실제 Value에 attention weight 를 계산하여, context vector 로 활용 목적

5번,2번 아이템을 받았을 때, 3번 아이템을 예측해야 하는 task 에서, 문제 발생

3번과 5번,2번의 관계를 미리 알아 context에 적용되면, 모델이 학습 능력이 저하됨(답을 알려주는 꼴..)

2.3. Self-Attention Layer



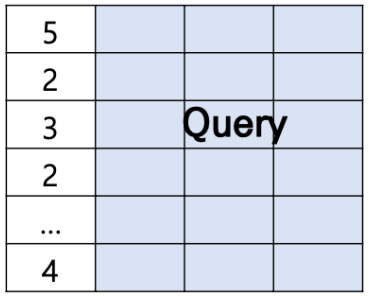
$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$

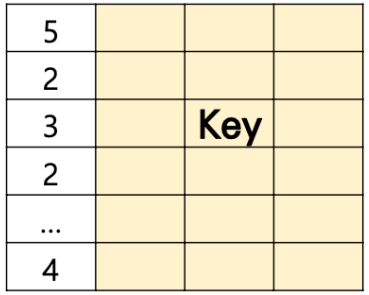
	5	2	3	2	...	4
5						
2						
3						
2						
...						
4						

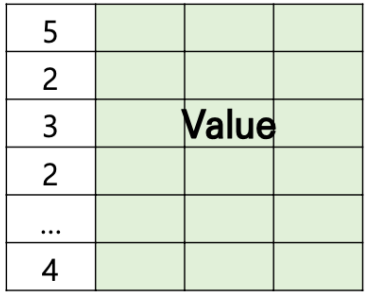
예측 할 다음 item을 해당 시점에서는 모르도록 masking 처리함

즉, 첫번째 예측 시, item 1만 고려
두번째 예측 시, item 1,2 고려 ..

2.3. Self-Attention Layer







$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$

	5	2	3	2	...	4
5						
2						
3						
2						
...						
4						

예측 할 다음 item을 해당 시점에서는 모르도록 masking 처리함

즉, 첫번째 예측 시, item 1만 고려
두번째 예측 시, item 1,2 고려 ..

	5	2	3	2	...	4
5	1	0	0	0	0	0
2	1	1	0	0	0	0
3	1	1	1	0	0	0
2	1	1	1	1	0	0
...	1	1	1	1	1	0
4	1	1	1	1	1	1

×

Value

5	1	2	3
2	4	5	6
3	7	8	9
2	10	11	12
...	13	14	15
4	16	17	18

Context vectors

5	1	2	3
2	1+4	2+5	3+6
3	1+4+7	2+5+8	3+6+9
2			
...			
4			

첫번째 아이템은 첫번째 아이템만 고려된 vector
두번째 아이템은 첫번째 + 두번째 아이템 고려된 vector

2.4. Point-Wise Feed-Forward Network

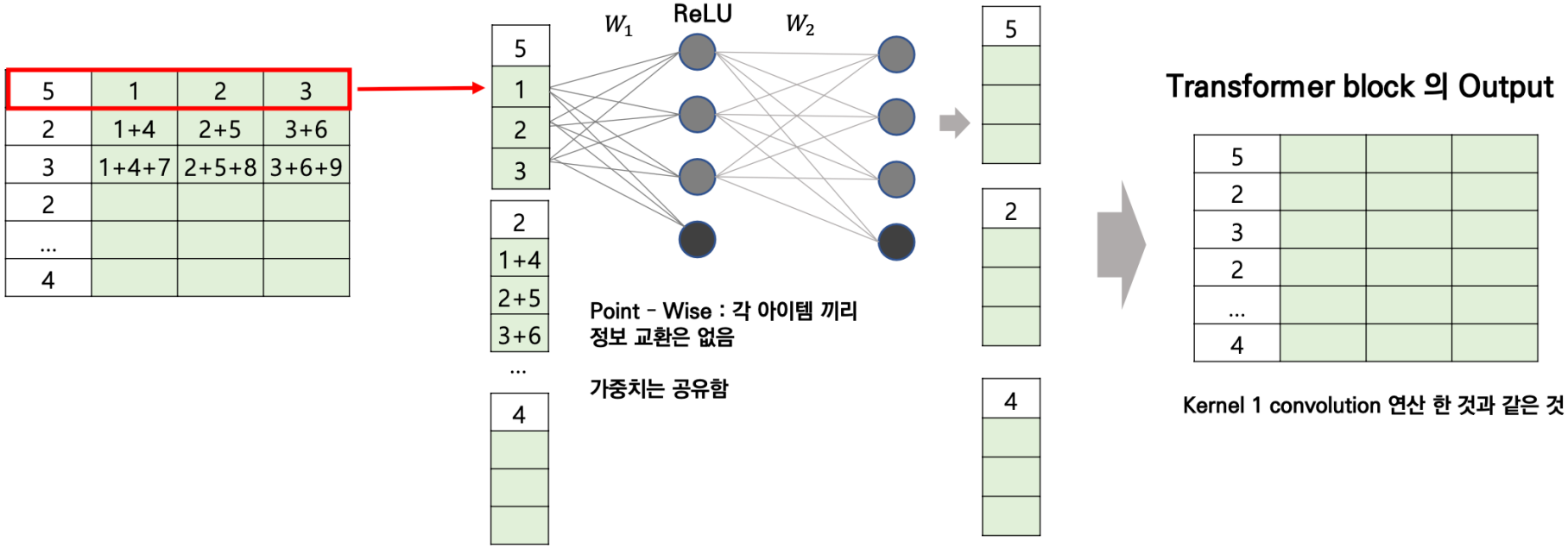
Context vectors

5	1	2	3	S_1
2	1+4	2+5	3+6	S_2
3	1+4+7	2+5+8	3+6+9	
2				
…				
4				S_n

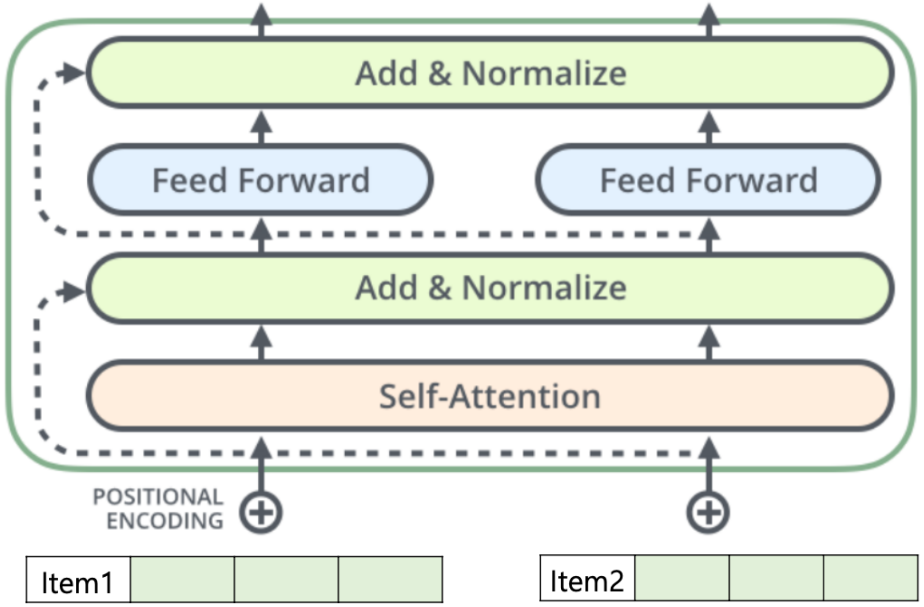
Context vector : linear model

Layer를 쌓는 의미를 부여하기 위해, non-linear activation 사용

$$\mathbf{F}_i = \mathrm{FFN}(\mathbf{S}_i) = \mathrm{ReLU}(\mathbf{S}_i\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{W}^{(2)} + \mathbf{b}^{(2)}$$



2.5. Residual Connections & Dropout



Block 수 : 2개

$$\mathrm{LayerNorm}(\mathbf{x}) = \boldsymbol{\alpha} \odot \frac{\mathbf{x} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \boldsymbol{\beta}$$

5			
2			
3			
2			
...			
4			

Layer - normalization

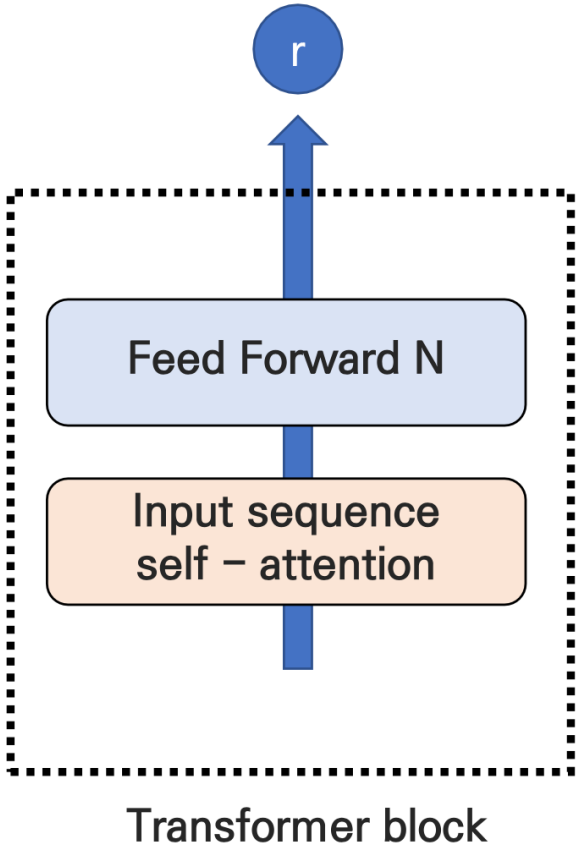
하나의 sample(item) 기준 normalize 진행

learning parameter α, β 가 학습됨
(attention all you need 참조)

Dropout : Sequence Item index 를 통해, Item matrix 에서 Look up 해올 때, Random한 비율로 아이템을 가지고 오지 않음(zero)

이는 아이템의 순서에 대한 robust 함을 상승 시키고, 여러 모델을 앙상블 하는 효과와 같다고 함

2.6. Prediction Layer



$n \times d$

5	1번째 아이템 정보
2	1,2번째 정보
3	1,2,3번째 정보
2	
...	
4	

d

5	1번째 아이템 정보

첫번째 아이템 vector → 두번째 올 아이템 예측

두번째 아이템 vector → 세번째 올 아이템 예측

$$\mathbf{N} \in \mathbb{R}^{|\mathcal{I}| \times d}$$

$$r_{i,t} = \mathbf{F}_t^{(b)} \mathbf{N}_i^T$$

Item Matrix

index	Look up table(vector)
1	D차원
2	
3	
...	
#item num	

GT

2	Look up vector
9	Look up vector
7	Look up vector
15	Look up vector
17	Look up vector

Negtive samples
해당 유저가 산 적이 없는 아이템들 중 랜덤 하게 100개 선택

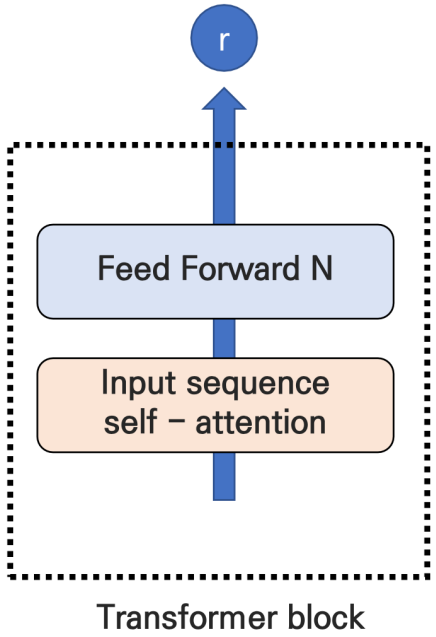
2.7. Network Training

$\mathcal{S}^u$ historical interaction sequence for a user u: $(\mathcal{S}^u_1, \mathcal{S}^u_2, ..., \mathcal{S}^u_{|\mathcal{S}^u|})$

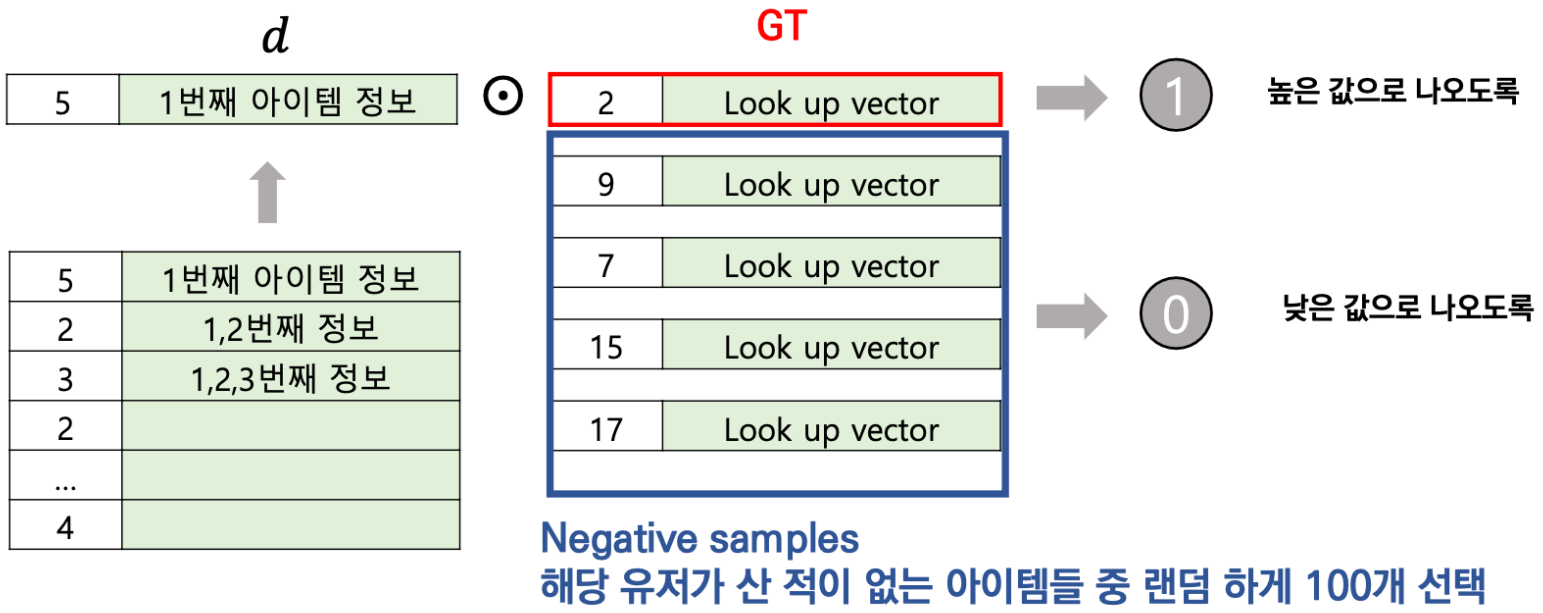
$$o_t = \begin{cases} \texttt{<pad>} & \text{if } s_t \text{ is a padding item} \\ s_{t+1} & 1 \le t < n \\ \mathcal{S}^u_{|\mathcal{S}^u|} & t = n \end{cases},$$

$$r_{i,t} = \mathbf{F}_t^{(b)} \mathbf{M}_i^T.$$

$$-\sum_{\mathcal{S}^u \in \mathcal{S}} \sum_{t \in [1,2,...,n]} \left[\log(\sigma(r_{o_t,t})) + \sum_{j \notin \mathcal{S}^u} \log(1 - \sigma(r_{j,t})) \right]$$



Transformer block



Negative samples
해당 유저가 산 적이 없는 아이템들 중 랜덤 하게 100개 선택

Experiments

3.1. Datasets & Metrics

Datasets

Dataset	#users	#items	avg. actions /user	avg. actions /item	#actions
Amazon Beauty	52,024	57,289	7.6	6.9	0.4M
Amazon Games	31,013	23,715	9.3	12.1	0.3M
Steam	334,730	13,047	11.0	282.5	3.7M
MovieLens-1M	6,040	3,416	163.5	289.1	1.0M

Metrics

Hit@K : 사용자가 실제로 선호한 아이템 중 모델이 맞게 추천한 아이템의 수

NDCG@K : 모델이 추천한 아이템의 순위 점수/실제로 추천된 아이템의 순위 점수

-> 모델이 K개를 추천하고 점수를 통해 순위를 정했을 때, 이것이 실제로 추천된 이상적인 아이템의 순위에 얼마나 부합하는가

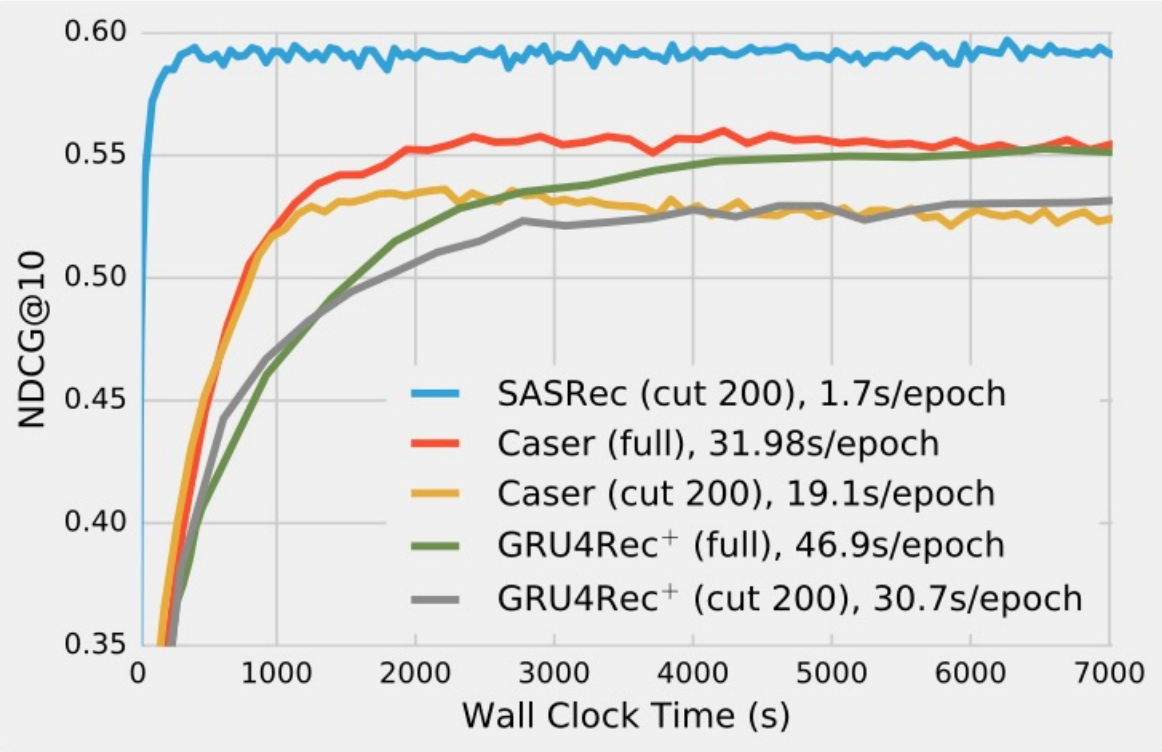
3.2. Experiments

- Does SASRec outperform state-of-the-art models including CNN/RNN based methods?

Dataset	Metric	(a) PopRec	(b) BPR	(c) FMC	(d) FPMC	(e) TransRec	(f) GRU4Rec	(g) GRU4Rec$^+$	(h) Caser	(i) SASRec	Improvement vs. (a)-(e)	Improvement vs. (f)-(h)
Beauty	Hit@10	0.4003	0.3775	0.3771	0.4310	0.4607	0.2125	0.3949	0.4264	**0.4854**	5.4%	13.8%
	NDCG@10	0.2277	0.2183	0.2477	0.2891	0.3020	0.1203	0.2556	0.2547	**0.3219**	6.6%	25.9%
Games	Hit@10	0.4724	0.4853	0.6358	0.6802	0.6838	0.2938	0.6599	0.5282	**0.7410**	8.5%	12.3%
	NDCG@10	0.2779	0.2875	0.4456	0.4680	0.4557	0.1837	0.4759	0.3214	**0.5360**	14.5%	12.6%
Steam	Hit@10	0.7172	0.7061	0.7731	0.7710	0.7624	0.4190	0.8018	0.7874	**0.8729**	13.2%	8.9%
	NDCG@10	0.4535	0.4436	0.5193	0.5011	0.4852	0.2691	0.5595	0.5381	**0.6306**	21.4%	12.7%
ML-1M	Hit@10	0.4329	0.5781	0.6986	0.7599	0.6413	0.5581	0.7501	0.7886	**0.8245**	8.5%	4.6%
	NDCG@10	0.2377	0.3287	0.4676	0.5176	0.3969	0.3381	0.5513	0.5538	**0.5905**	14.1%	6.6%

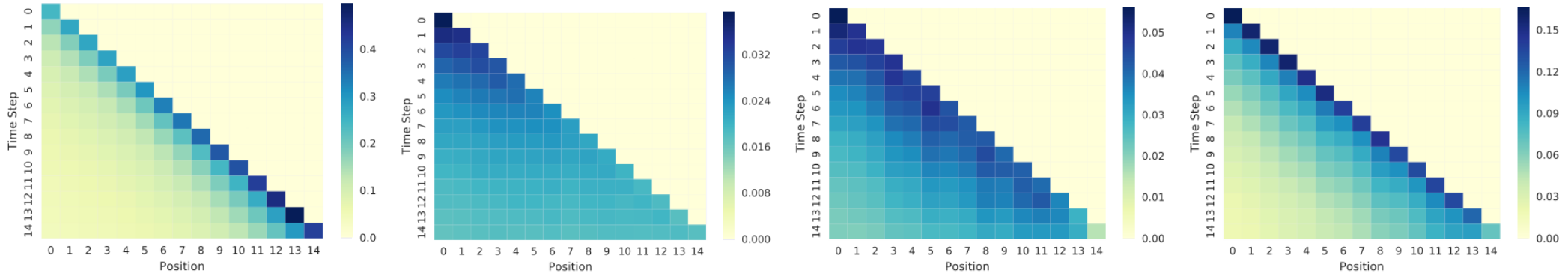
3.2. Experiments

- What is the training efficiency and scalability (regarding n) of SASRec?



3.2. Experiments

- Are the attention weights able to learn meaningful patterns related to positions or items' attributes?



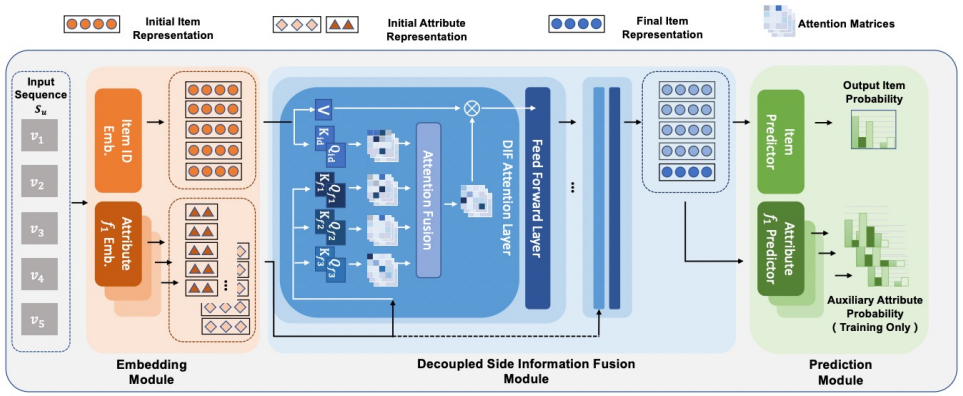
(a) *Beauty*, Layer 1 (b) *ML-1M*, Layer 1, w/o PE (c) *ML-1M*, Layer 1 (d) *ML-1M*, Layer 2

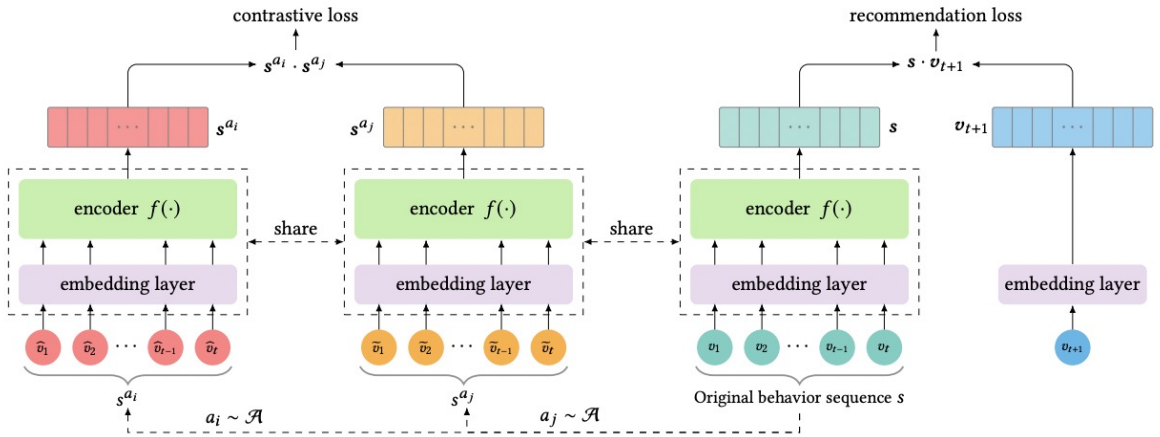
Figure 4: Visualizations of average attention weights on positions at different time steps. For comparison, the heatmap of a first-order Markov chain based model would be a diagonal matrix.

Summary

4.1. Conclusion

- 새로운 self-attention 기반 sequential model인 SASRec을 제안합니다.
- 다양한 실험을 통해 기존 모델과 비교했을 때 더 나은 속도와 성능을 보이며 당시 기준 SOTA 달성
- 앞으로 나올 다양한 self attention 기반 Sequential Recommender 모델의 기본적인 형태가 됨





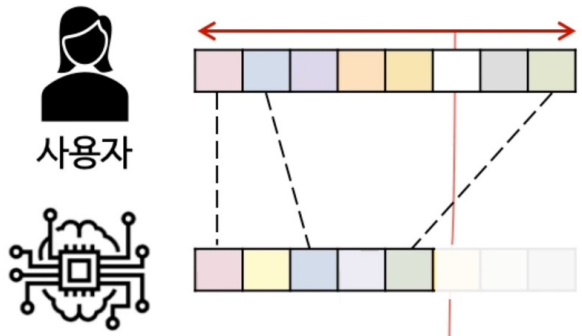
Appendix

3.1. Datasets & Metrics

Metrics

Hit @N (Hit Ratio)

- 추천의 정확도를 나타내는 평가지표
- 사용자의 실제값 중 모델이 맞게 추천한 아이템의 수



모델 : 결과값 중 중요도 기준 상위 N개만 활용
(ex. N = 5)

→ HR @5 = 3/8 = **0.375**

NDCG @N (Normalized Discounted Cumulative Gain)

- 추천된 순위에 가중치를 주어 계산하는 평가지표
- 가장 이상적인 순위와 실제 추천된 순위에 대한 비교 진행

$$DCG = \sum_{i=1}^{n} \frac{relevance_i}{\log_2(i+1)}, \qquad NDCG = \frac{DCG}{iDCG}$$

DCG : Discounted Cumulative Gain, **iDCG** : Ideal Discounted Cumulative Gain

순위	정답 여부	평점 (Relevance)
1	O	1
2	X	1
3	O	1
4	X	1
5	O	1

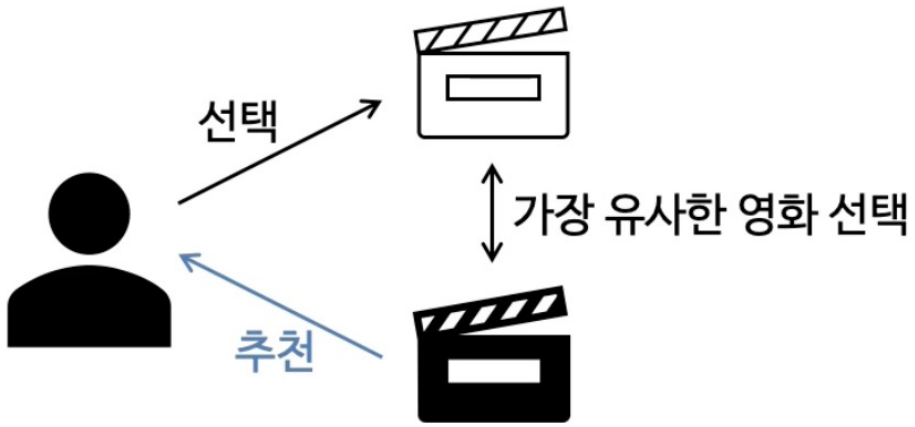
$$DCG = \frac{1}{\log_2(1+1)} + \frac{1}{\log_2(3+1)} + \frac{1}{\log_2(5+1)} = 1.8869$$

$$iDCG = \frac{1}{\log_2(1+1)} + \frac{1}{\log_2(2+1)} + \frac{1}{\log_2(3+1)} = 2.1309$$

→ NDCG @5 = 1.8869/2.1309 = **0.8855**

1.1. Introduction to Recommendation System

What is Recommendation System? - Contents based Filtering



장점

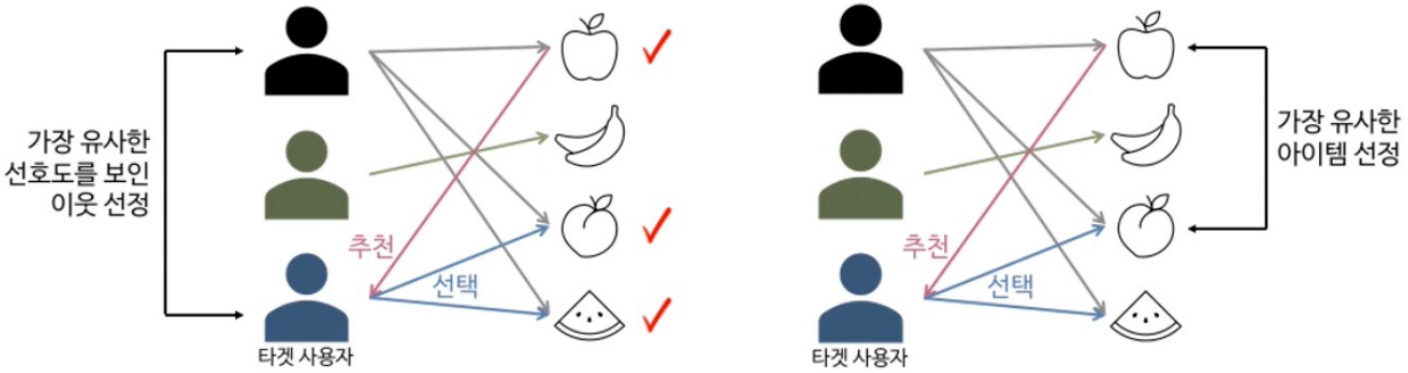
- 사용자가 평점을 매기지 않은 새로운 아이템이 들어와도 추천이 가능함

단점

- 과도한 특수화 (Over Specialization)
  - 이전에 구매 및 선택했던 아이템과 비슷한 제품만 추천하는 경향

1.1. Introduction to Recommendation System

What is Recommendation System? - Collaborative Filtering



- Cold Start : 특정 사용자에 대한 충분한 데이터(평점 등)가 부족한 경우 선호도 예측 불가능
- First Rater : 새로운 아이템이 등장하였을 때, 평점 점수가 부족할 경우 추천 불가능
- Grey Sheep Problem : 일관성이 없는 의견을 가진 사용자들의 데이터들은 추천에 혼란을 줌
- Shibling Attack : 악의적으로 평가 점수를 긍정/부정으로 입력할 경우 추천에 방해가 됨