

# GRU4Rec

## SESSION-BASED RECOMMENDATIONS WITH RECURRENT NEURAL NETWORKS

우리 대체 몇조조

20172848 이지평 20172853 장성현 20192761 김정하

# CONTENTS

**01. Abstract**

**02. Introduction**

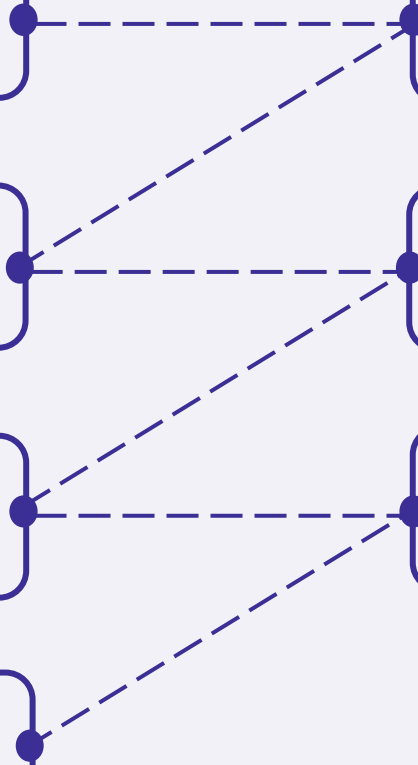
**03. Recommendation with RNNs**

**04. Experiments**

**05. Conclusion**


**06. Future work**

**07. Appendix**




# 01


## Abstract

 **오늘의 쇼핑 제안**

13%




앤커 인체공학 버티컬 무...  
로켓배송  
★★★★★ (1,658)  
추가 할인쿠폰




로이체 피너즈 멀티태스...  
로켓배송  
★★★★★ (709)  
추가 할인쿠폰

32%




제이로드 아이패드 에어5...  
재트배송  
★★★★★ (2,348)

16%




신지모루 태블릿 저반사 ...  
로켓배송  
★★★★★ (2,562)  
추가 할인쿠폰




로이체 디즈니 무소음 무...  
로켓배송  
★★★★★ (78)  
추가 할인쿠폰

광고


지금 이 상품이 필요하신가요? 광고 ① 1/3




셀리온 프리미엄 돈모를 포...  
링 헤어브러쉬, 3호, 1개  
7,160원 로켓배송




CP-1 혼자하는 앞머리 셀프...  
세트, 1세트  
7,500원 로켓배송  
★★★★★ (697)



달리드 글함 노워서 극손상...  
단발질 헤어팩 인 미스트,  
200ml, 2개  
22,800원 로켓배송  
10ml당 520원



벨로티 뿌리부터 집게질 헤어...  
를 2개입, 핑크, 1개  
5,500원 로켓배송  
★★★★★ (981)



유닉스 테이크아웃 시즌4 무...  
선고데기 UCI-A2022, 민트그...  
린  
29,750원 로켓배송

→ 기존의 방법(MF 같은)으로는 정확하지 않음

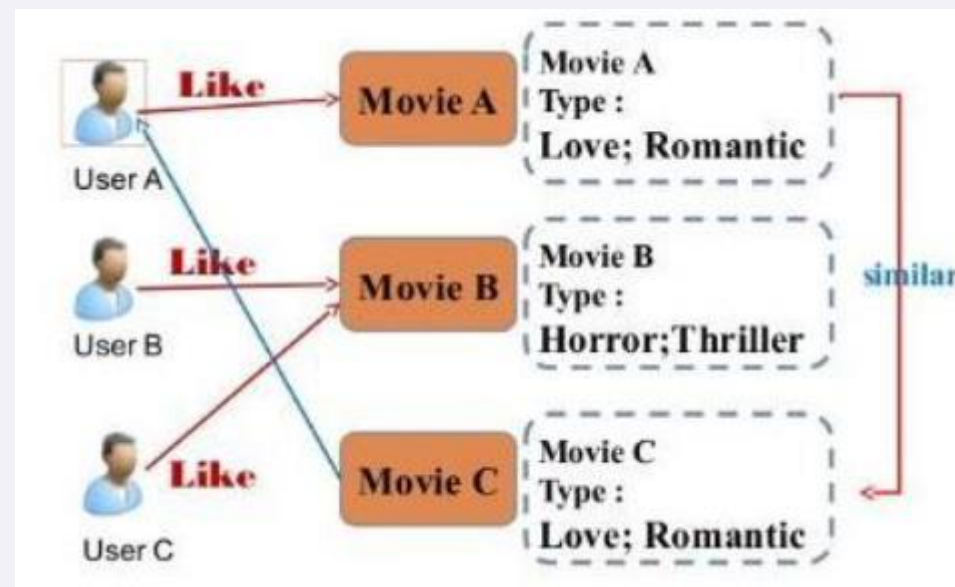
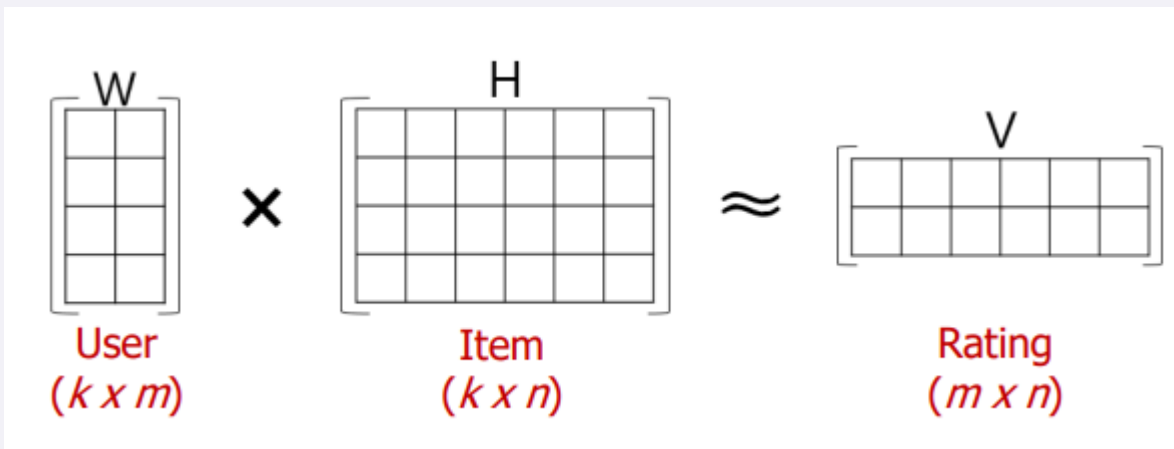
→ RNN 기반의 SBR 제안

# 02

## Introduction

지금까지 추천시스템에서 사용되었던 기본적인 방법들은,

### 1. Factor Models



따라서 RNN을 이용하여

과거정보를 반영한

Session 단위의 추천시스템 생성이 목적!

# 02

## Introduction

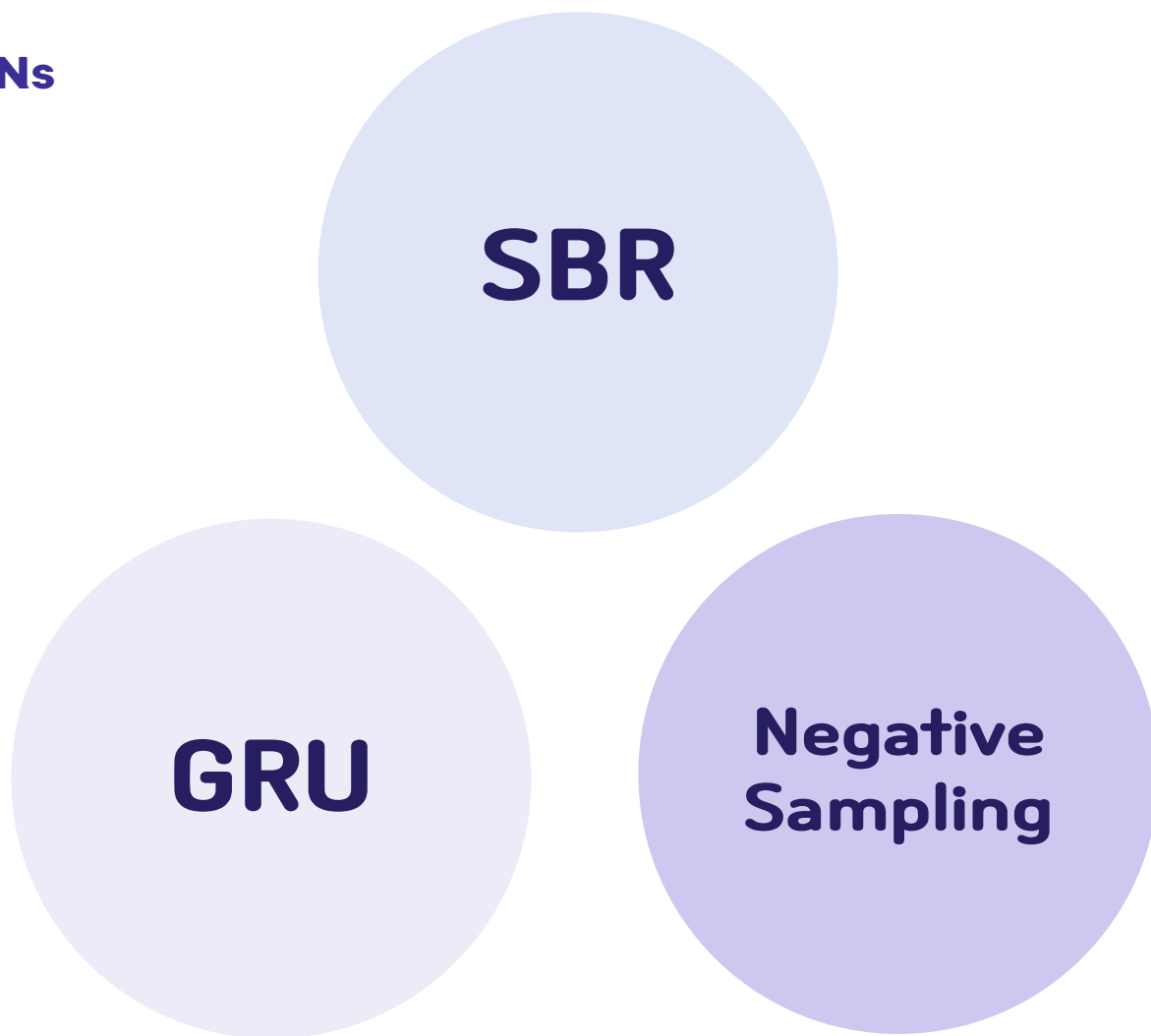
### 논문에서 주의 깊게 보아야 할 점

- 추천 Task를 위해 도입한 **Ranking Loss**는 무엇을 사용하였는지?
- Click-stream 데이터 같은 상당히 큰 데이터에서 훈련 시간과 확장성을 어떻게 고려하였는지?

# 03

## Recommendation with RNNs

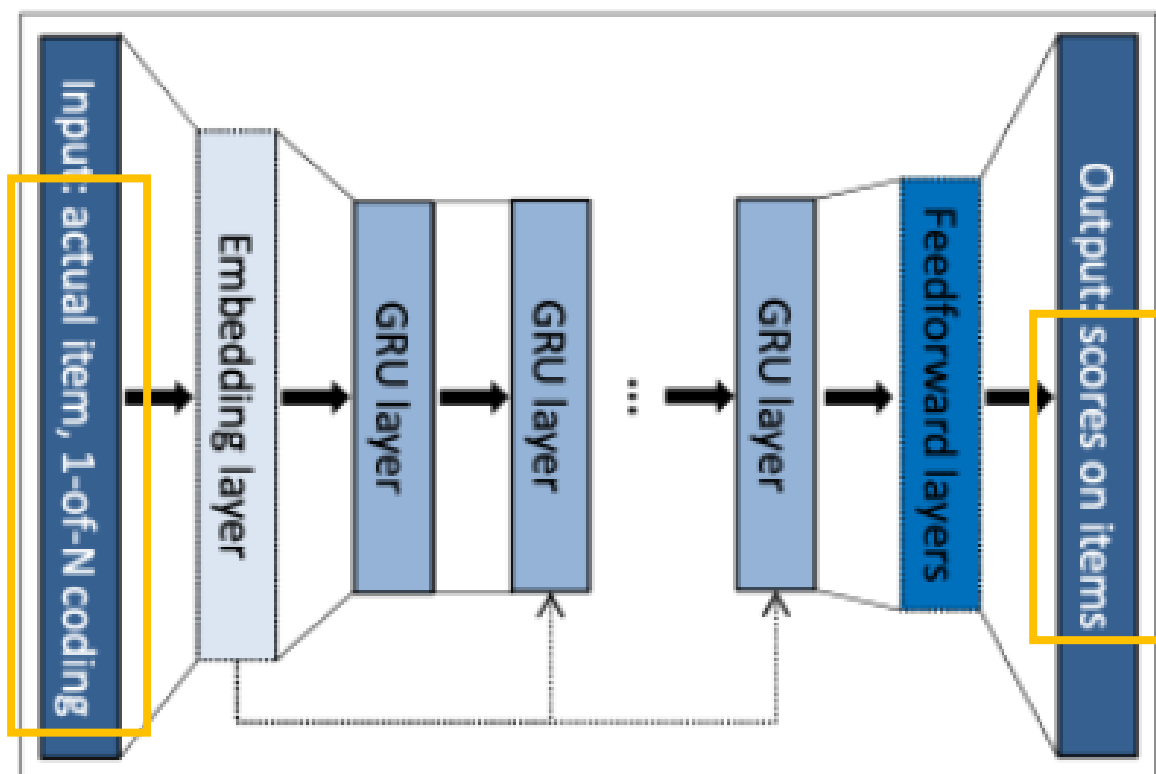
### 3.0 배경지식



# 03

## Recommendation with RNNs

### 3.0.1 SBR

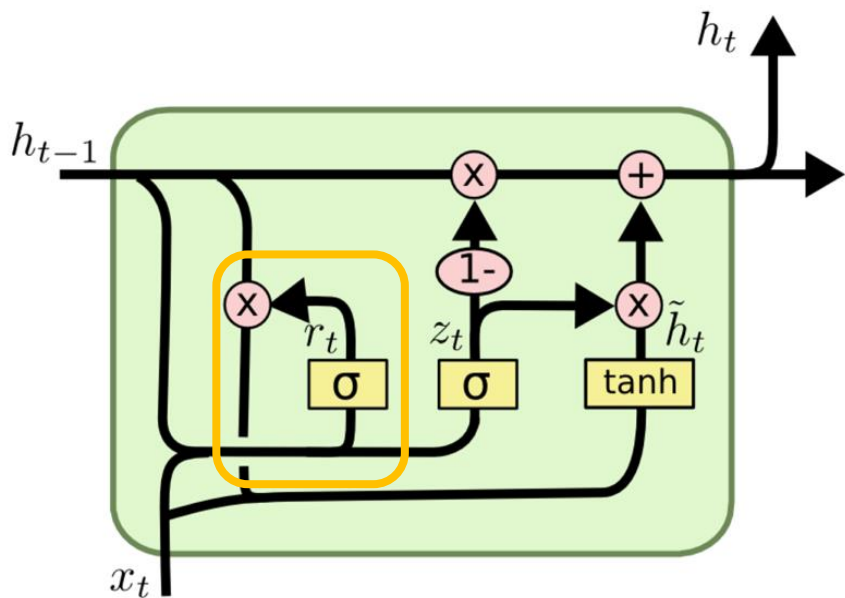




# 03

## Recommendation with RNNs

### 3.0.2 GRU ( Gated Recurrent Units )



#### 1. Reset Gate

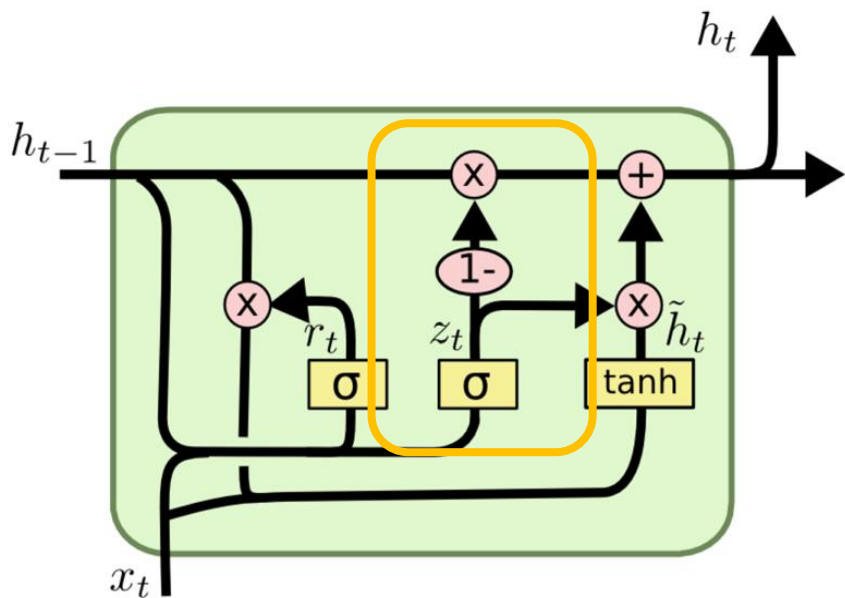
: 과거의 정보를 적당히 reset 시키는 것이 목적

$$r^{(t)} = \sigma \left( W_r h^{(t-1)} + U_r x^{(t)} \right)$$

# 03

## Recommendation with RNNs

### 3.0.2 GRU ( Gated Recurrent Units )



#### 2. Update Gate

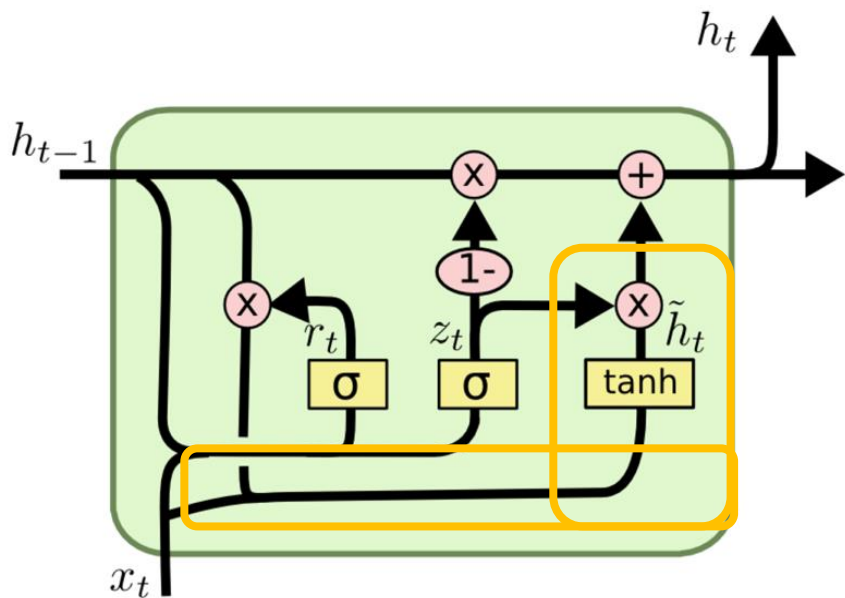
: 과거와 현재의 정보의 최신화 비율을 결정

$$u^{(t)} = \sigma \left( W_u h^{(t-1)} + U_u x^{(t)} \right)$$

# 03

## Recommendation with RNNs

### 3.0.2 GRU ( Gated Recurrent Units )



#### 3. Candidate

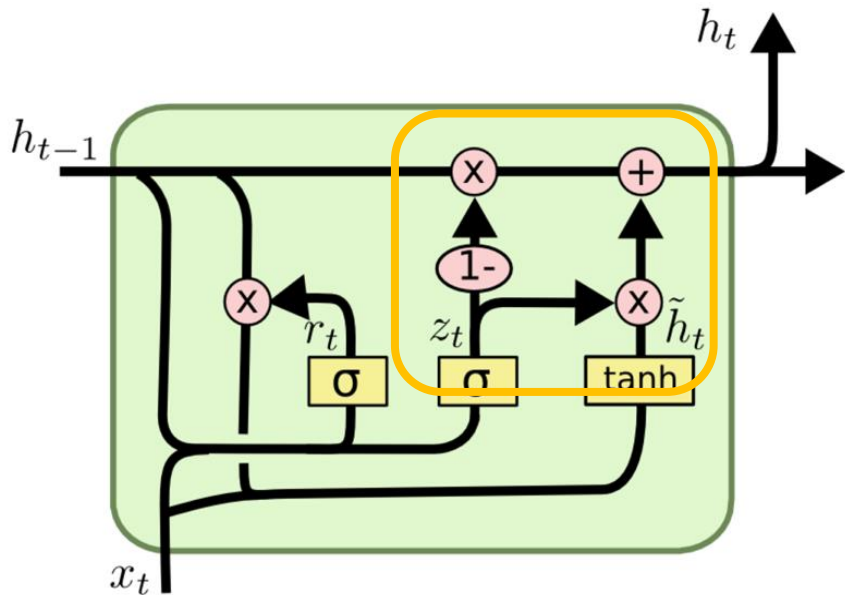
: 현 시점의 정보 후보군을 계산하는 단계

$$\tilde{h}^{(t)} = \tau \left( W h^{(t-1)} * r^{(t)} + U x^{(t)} \right)$$

# 03

## Recommendation with RNNs

### 3.0.2 GRU ( Gated Recurrent Units )



#### 4. 은닉층 계산

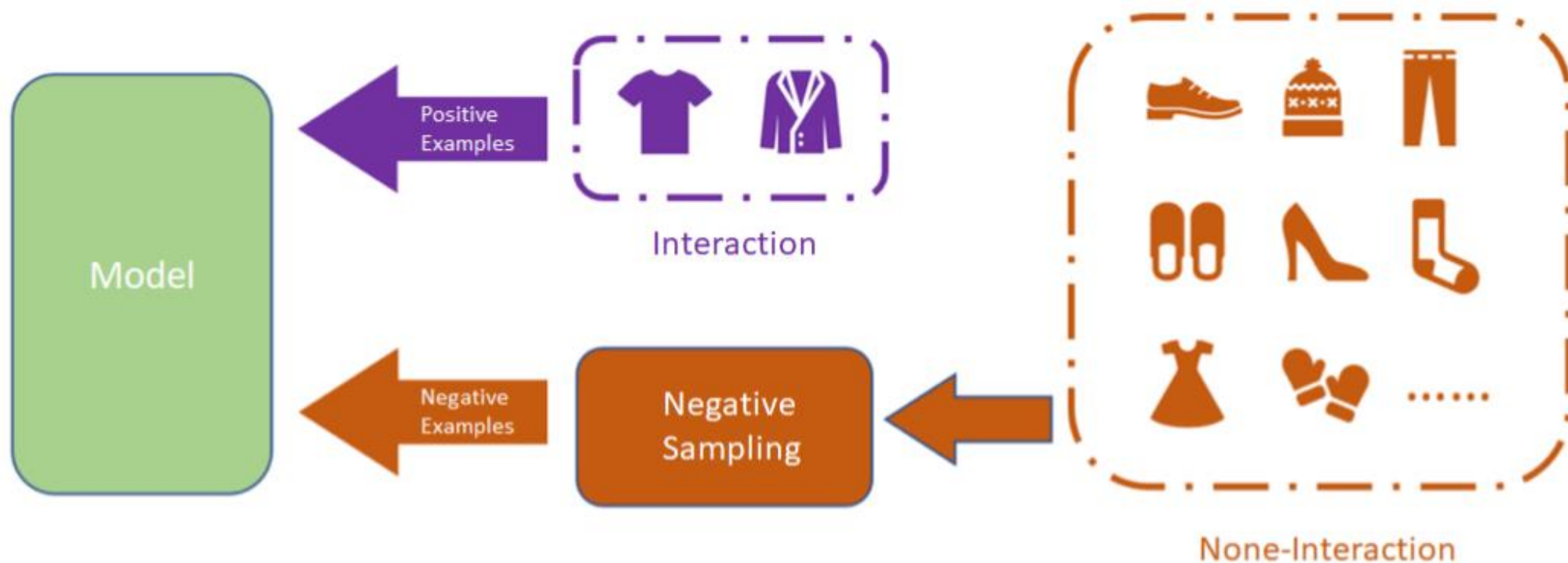
: update gate 결과와 candidate 결과를 결합

$$h^{(t)} = (1 - u^{(t)}) * h^{(t-1)} + u^{(t)} * \tilde{h}^{(t)}$$

# 03

## Recommendation with RNNs

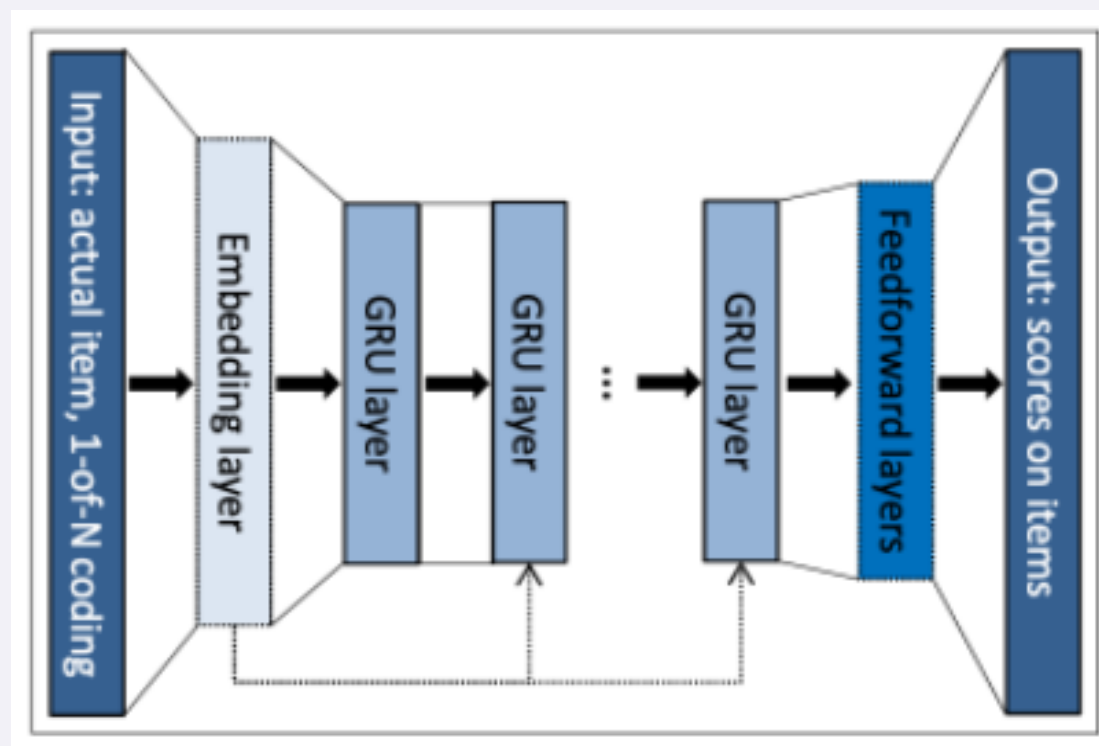
### 3.0.3 Negative Sampling



# 03

## Recommendation with RNNs

### 3.1 Customizing the GRU Model



# 03

## Recommendation with RNNs

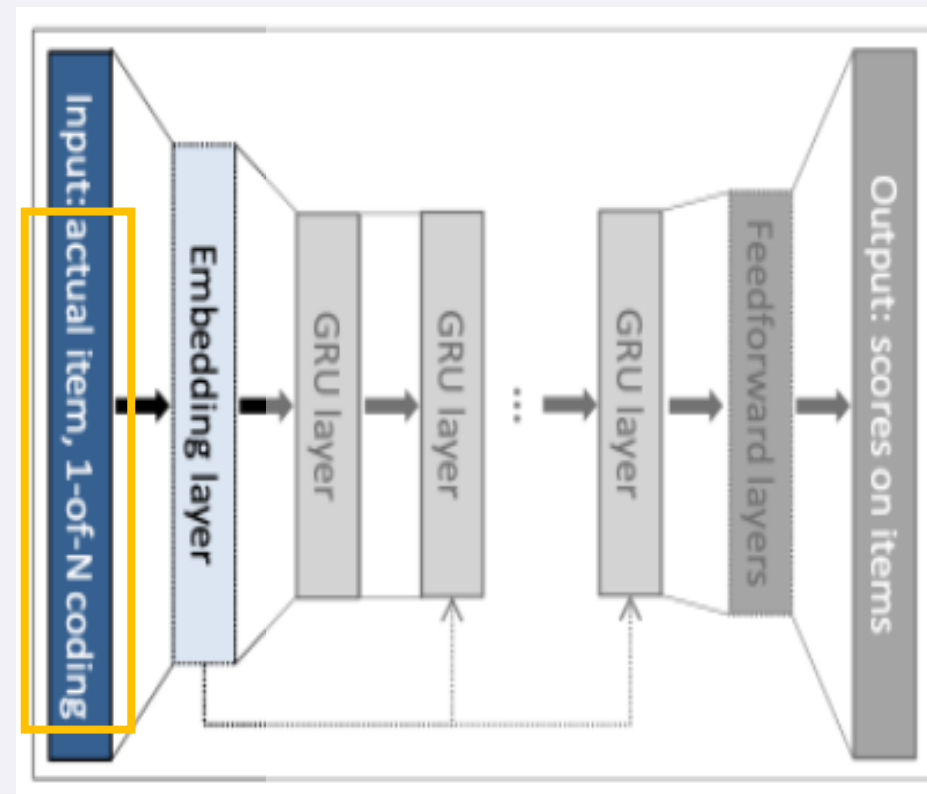
### 3.1 Customizing the GRU Model

#### 1. the item of the actual event

1-of-N 인코딩 ( 원핫인코딩 )

#### 2. the events in the session

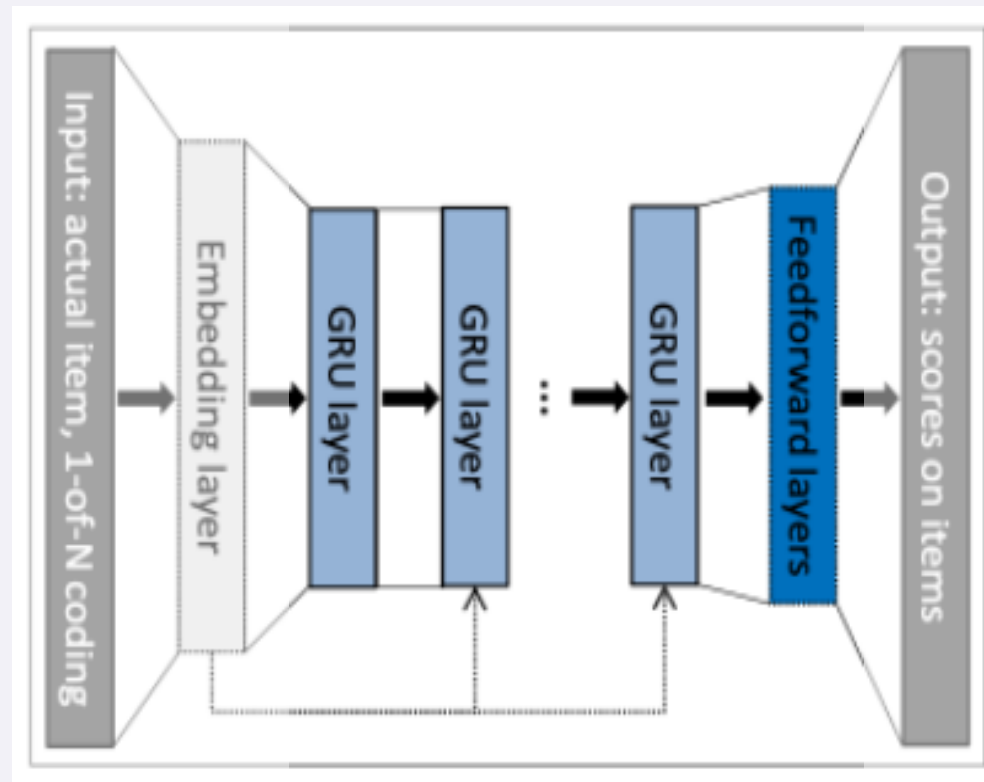
가중합 ( 최근꺼에 더 많은 가중치 )



# 03

## Recommendation with RNNs

모델 구조의 핵심은 GRU + FFL  
GRU layer를 쌓을수록 성능 향상





# 03

## Recommendation with RNNs

### 3.1.1 Session-Parallel Mini-Batches

$i_{1,1}$   $i_{1,2}$   $i_{1,3}$   $i_{1,4}$

$i_{2,1}$   $i_{2,2}$   $i_{2,3}$

$i_{3,1}$   $i_{3,2}$   $i_{3,3}$   $i_{3,4}$   $i_{3,5}$   $i_{3,6}$

$i_{4,1}$   $i_{4,2}$

$i_{5,1}$   $i_{5,2}$   $i_{5,3}$

하지만 이러한 기법은 추천 Task에는 적합하지 않음

- Session의 길이가 일반적인 문장보다 훨씬 다양
- 목표는 session이 어떻게 변화하는지 포착하는 것이므로 부분마다 자르는 것은 합리적이지 않음

[기존 NLP의 배치방식]

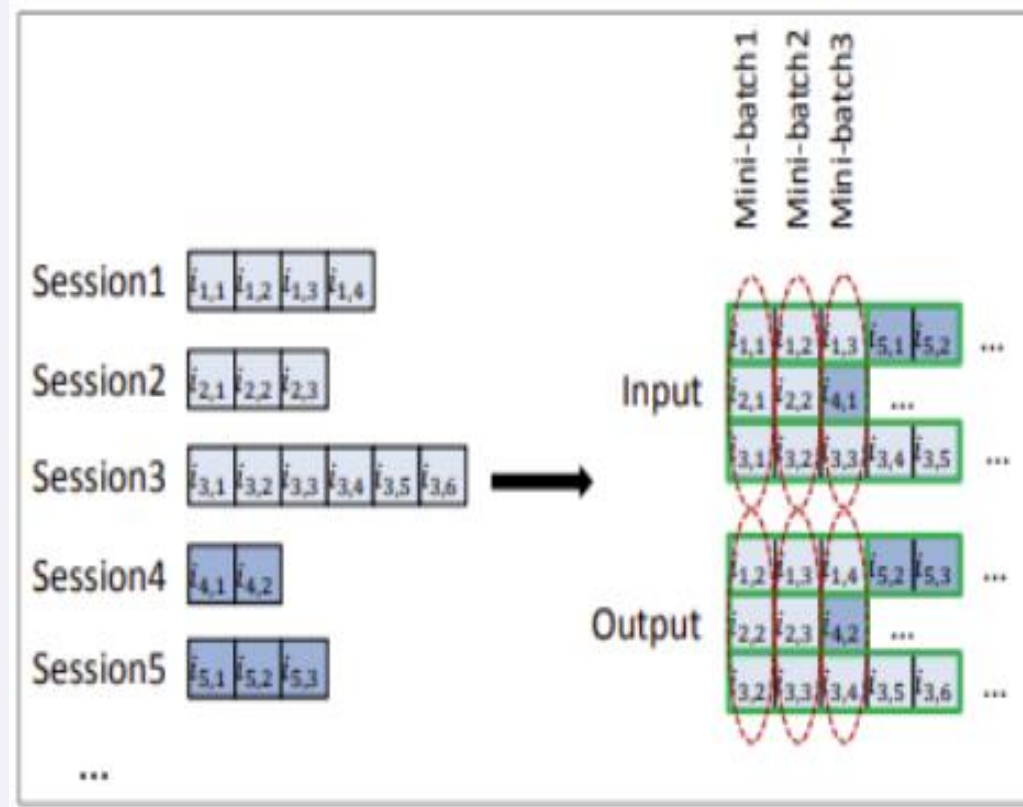
# 03

## Recommendation with RNNs

### 3.1.1 Session-Parallel Mini-Batches

대안으로 session-parallel mini-batches를 사용함

1. 각 session들을 정렬시킴
2. 다음 mini-batch의 형태를 구성하기 위해  
X개의 session에서 첫 event만을 사용
3. 다음 event들로 mini-batch를 구성



# 03

## Recommendation with RNNs

### 3.1.2 Sampling on the Output

#### Missing event

**사용자가 아이템의 존재를 몰랐기 때문에 상호작용이 없었던 것**

**vs**

**아이템을 알지만 선호하지 않기에 상호작용 하지 않았던 것(낮은 가능성)**

인기가 높은 아이템의 경우 사용자가 알고 있을 가능성이 높음  
허나, Missing event라면 선호하지 않아 상호작용이 없었던 것일 가능성이 높음

**따라서 선호도에 비례하여 아이템을 sampling할 필요가 있음**

# 03

## Recommendation with RNNs

### 3.1.2 Sampling on the Output

각 train 데이터에 대해 sample을 따로 생성하는 대신, **Negative sample**의 역할을 할 수 있도록 mini-batch내의 다른 train데이터들을 negative sample로 설정

→ mini-batch의 다른 train 데이터의 아이템 또한 **선호도**에 비례하기 때문

→ 따로 sampling을 구하지 않기 때문에 계산 시간을 효율적으로 줄임

# 03

## Recommendation with RNNs

### 3.1.3 Ranking Loss

- 학습에 Loss Function을 새롭게 정의
- 순위를 추정하는 방법 총 3가지
  - Pointwise
  - Pairwise
  - Listwise

# 03

## Recommendation with RNNs

### 3.1.3 Ranking Loss

Pairwise 기반 Loss Function 2가지 제안

- **BPR**

긍정적인 항목과 Negative Sampling의 점수를 비교

여기서 긍정적인 항목의 점수를 여러 샘플 항목과 비교

$$L_s = -\frac{1}{N_s} * \sum_{j=1}^{N_s} \log(\sigma(\hat{r}_{s,i} - \hat{r}_{s,j}))$$

03

즉, BPR은

긍정적인 항목과 Negative Sample의  
점수 차가 크길 바라는 것

# 03

## Recommendation with RNNs

### 3.1.3 Ranking Loss

Pairwise 기반 Loss Function 2가지 제안

- **TOP1**

이 논문에서 제안한 Loss Function

$$L_s = \frac{1}{N_s} \cdot \sum_{j=1}^{N_s} \sigma(\hat{r}_{s,j} - \hat{r}_{s,i}) + \sigma(\hat{r}_{s,j}^2)$$



# 03

즉, TOP1은

BPR과 동일한 기능 및

Negative Sample의 점수가 0에 가까워 지길 바라는 것

$$L_{\text{BPR}} = -\sum_{(i,j) \in \mathcal{D}} \log \sigma(\mu_i - \mu_j) + \sum_{(i,j) \in \mathcal{D}^+} \log \sigma(\mu_i - \mu_j)$$

# 04

## Experiments

### 4.0 Experiments

#### - Dataset :

RSC15, VIDEO 두가지를 사용하여 비교

#### - 평가지표:

##### recall@20:

상위 20개 항목 중 원하는 항목을 가진 경우의 비율

상위 N개 항목에 포함된 항목의 **실제 순위를 고려 X**

추천 강조X, 순서가 중요하지 않은 특정 시나리오 모델링 Good!

##### MRR@20(평균 역수 순위):

원하는 항목의 역수 순위의 평균

순위가 200이상이면 역순위는 0으로 설정

**항목의 순위를 고려**

추천 순서가 중요한 경우에 사용

# 04

## Experiments

### 4.1 BASELINES

베이스라인 세트와 비교

- POP
- S-POP
- Item-KNN
- BPR-MF

Table 1: Recall@20 and MRR@20 using the baseline methods

Baseline	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
POP	0.0050	0.0012	0.0499	0.0117
S-POP	0.2672	0.1775	0.1301	0.0863
Item-KNN	0.5065	0.2048	0.5508	0.3381
BPR-MF	0.2574	0.0618	0.0692	0.0374

# 04

## Experiments

### 4.2 PARAMETER & STRUCTURE OPTIMIZATION

100개의 실험으로 하이퍼-파라미터를 진행

Table 2: Best parametrizations for datasets/loss functions

Dataset	Loss	Mini-batch	Dropout	Learning rate	Momentum
RSC15	TOP1	50	0.5	0.01	0
RSC15	BPR	50	0.2	0.05	0.2
RSC15	Cross-entropy	500	0	0.01	0
VIDEO	TOP1	50	0.4	0.05	0
VIDEO	BPR	50	0.3	0.1	0
VIDEO	Cross-entropy	200	0.1	0.05	0.3

# 04

## Experiments

### 4.3 RESULTS

Baseline	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
POP	0.0050	0.0012	0.0499	0.0117
S-POP	0.2672	0.1775	0.1301	0.0863
Item-KNN	0.5065	0.2048	0.5508	0.3381
BPR-MF	0.2574	0.0618	0.0692	0.0374

Table 3: Recall@20 and MRR@20 for different types of a single layer of GRU, compared to the best baseline (item-KNN). Best results per dataset are highlighted.

Loss / #Units	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
TOP1 100	0.5853 (+15.55%)	0.2305 (+12.58%)	0.6141 (+11.50%)	0.3511 (+3.84%)
BPR 100	0.6069 (+19.82%)	0.2407 (+17.54%)	0.5999 (+8.92%)	0.3260 (-3.56%)
Cross-entropy 100	0.6074 (+19.91%)	0.2430 (+18.65%)	0.6372 (+15.69%)	0.3720 (+10.04%)
TOP1 1000	0.6206 (+22.53%)	<b>0.2693 (+31.49%)</b>	<b>0.6624 (+20.27%)</b>	<b>0.3891 (+15.08%)</b>
BPR 1000	<b>0.6322 (+24.82%)</b>	0.2467 (+20.47%)	0.6311 (+14.58%)	0.3136 (-7.23%)
Cross-entropy 1000	0.5777 (+14.06%)	0.2153 (+5.16%)	–	–

# 05

## Conclusion

### 논문에서 주의 깊게 보아야 할 점

- 추천 Task를 위해 도입한 **Ranking Loss**는 무엇을 사용하였는지?

BPR과 TOP1 loss function을 도입함으로써 긍정적인 항목과 부정적인 항목의 점수차가 크게 되도록 학습하였고, top1은 거기에 추가로 부정적인 항목의 점수가 0에 가까워지도록 학습.

- Click-stream 데이터 같은 상당히 큰 데이터에서 훈련 시간과 확장성을 어떻게 고려하였는지?

Negative sampling을 따로 하지 않고 같은 미니배치 내의 다른 데이터를 negative sample로 사용하여 계산 과정을 줄임으로써 훈련시간과 확장성을 고려하였다고 볼 수 있음.

# 05

## Conclusion

세션기반 추천작업은 실질적으로 중요한 영역이지만 잘 연구되지 않았음

- 추천시스템에 반복 신경망(GRU)를 적용함
- 세션 병렬 미니배치, 미니배치 기반 출력 sampling
- 순위 손실함수를 도입

# 06

## Future work

### 6.1 기존 Session-Based Recommendation model 의 단점

Recurrent Neural Networks (RNN) – GRU

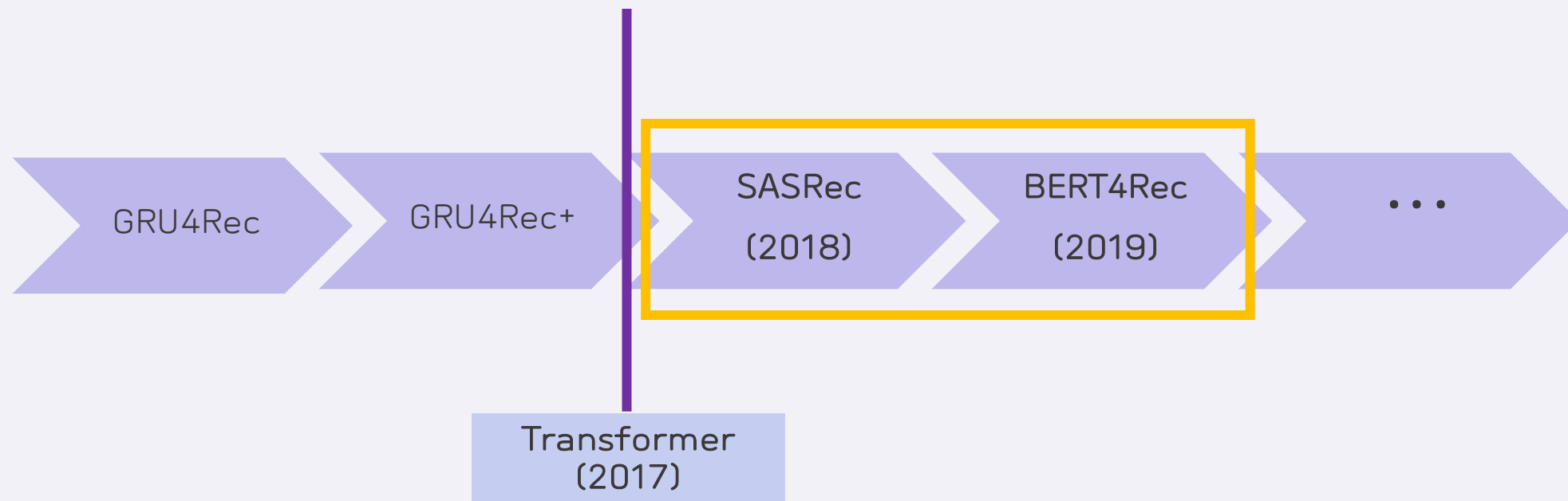
- 희소한 데이터에서는 좋은 성능을 보여주기 어려움
- Long term의 정보를 반영하기 힘들
- 계산 속도가 다소 느림



# 06

## Future work

### 6.2 Future work



# 06

## Future work

### 6.2 Future work

	SASRec	BERT4Rec
Attention	Single-head Attention	Multi-head Attention
Mask	마지막 1개 Mask 사용	여러 개의 Mask 사용
FC Layer 활성화 함수	ReLu 함수 사용	GeLU 사용
공통점	Residual Connection , Layer Normalization , Dropout 사용	

# THANK YOU.

GRU4Rec

SESSION-BASED RECOMMENDATIONS WITH RECURRENT NEURAL NETWORKS

우리 대체 몇조조

20172848 이지평 20172853 장성현 20192761 김정하

# 07

## Appendix

### 2.1 Session-Based Recommendation

기존 방법들의 경우,

사용자의 마지막 클릭 아이템을 기준으로 유사한 아이템을 찾기 때문에  
과거 클릭 정보를 무시하게 되는 문제점이 있음

이러한 문제점을 해결하기 위한 2가지 방법

# 07

## Appendix

### 2.1 Session-Based Recommendation

#### 1) MDP(Markov Decision Process): 순차적 확률적 의사결정 문제의 모델

- MDP 설명

ex) 벽돌 깨기 게임

Environment(환경): 벽돌 깨기

State(상태): 바의 위치, 공의 방향과 위치, 모든 벽돌의 존재유무 등

Agent: 조종자

Action(행동): 바를 왼쪽 또는 오른쪽으로 옮기는 것과 같은 행동

Reward(보상): 행동들의 결과로 얻는 점수

Action들은 Environment에 변화를 일으키고, State값이 변하게 되어, Agent가 또 다른 Action을 선택하도록 하고, 이를 Policy(정책)이라 합니다.

여기서 Environment는 때로 무작위적으로 생성되는데, 예를 들어 공을 놓쳐버린 후에 새로운 공이 발사되는 방향 등은 랜덤하게 설정됩니다.

상태와 행동 그리고 보상들을 통해 마르코프 의사결정 Process가 결정됩니다.

이러한 Process중 하나의 Episode(예를 들면, 게임의 한 턴)가 유한한 상태와 행동, 그리고 보상의 시퀀스를 형성하게 됩니다.

# 07

## Appendix

### 2.1 Session-Based Recommendation

#### 1) MDP(Markov Decision Process): 순차적 확률적 의사결정 문제의 모델

MDP방법을 이용하여 아이템 간 전이확률을 기반으로 간단하게 다음 추천, Action이 무엇인지 계산할 수 있다.

하지만 빠르게 아이템의 수가 많아지기 때문에, 즉 유저가 선택할 수 있는 폭이 넓어지기 때문에 MDP기반의 접근방법만으로는 **state space**를 점점 관리할 수 없게 됨

# 07

## Appendix

### 2.1 Session-Based Recommendation

#### 2) GFF(General Factorization Framework)

아이템에 대해서 두 종류의 latent vector를 사용한다.

하나는 **아이템 그 자체**와 다른 하나는 **session으로서의 아이템**을 사용한다.

따라서 어떤 session은 **session으로서의 아이템들의 평균**으로 표현될 수 있다.

하지만 session간에 순서를 고려하지 않는다.

# 07

## Appendix

### 2.2 Deep Learning In Recommendation

Restricted Boltzmann Machines(RBM) :

사용자 항목 상호 작용을 모델링하고 추천을 수행하는데 사용

Collaborative Filtering 모델들에서 유저와 아이템의 Interaction을 바탕으로

우수한 성능을 나타냄

최근 Deep Model들은 음악이나 이미지같이 구조화되지 않은 콘텐츠에서 feature를 추출하기 위해 사용되어졌으며, 전통적인 CF들과 함께 사용됨



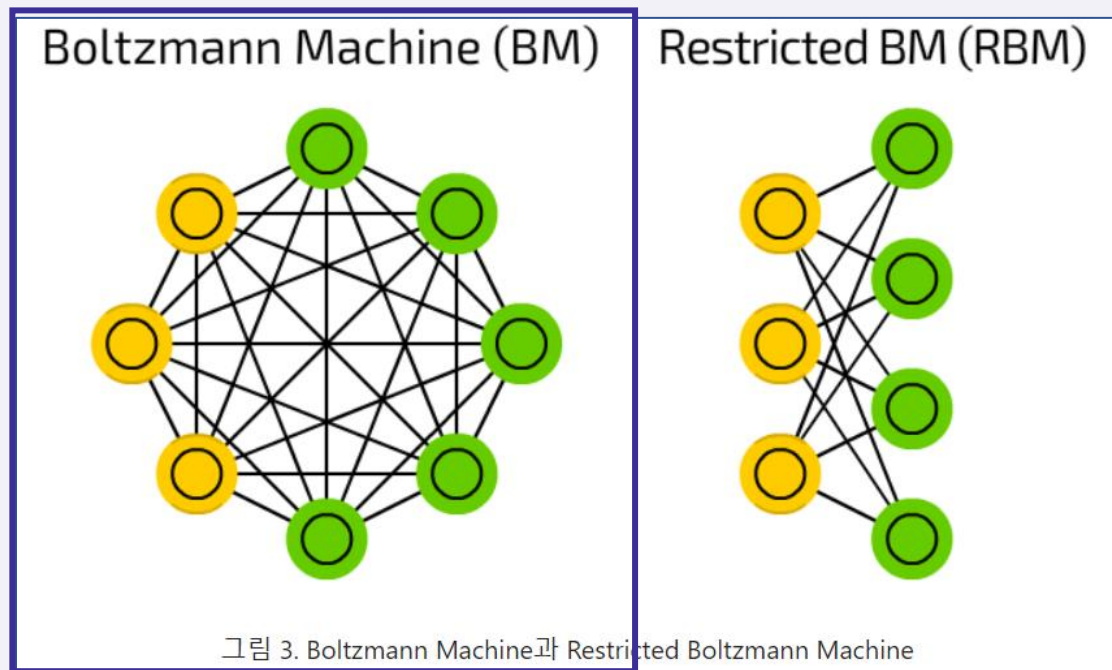
# 07

## Appendix

### 2.2 Deep Learning In Recommendation

#### Boltzmann Machine

- 확률분포를 학습하기 위해 만들어짐
- “우리가 보고 있는 것들 외에도 보이지 않는 요소들까지 잘 포함시켜 학습할 수 있다면 확률분포를 좀 더 정확하게 알 수 있지 않을까?” 라고 가정
- 계산이 복잡하다는 단점



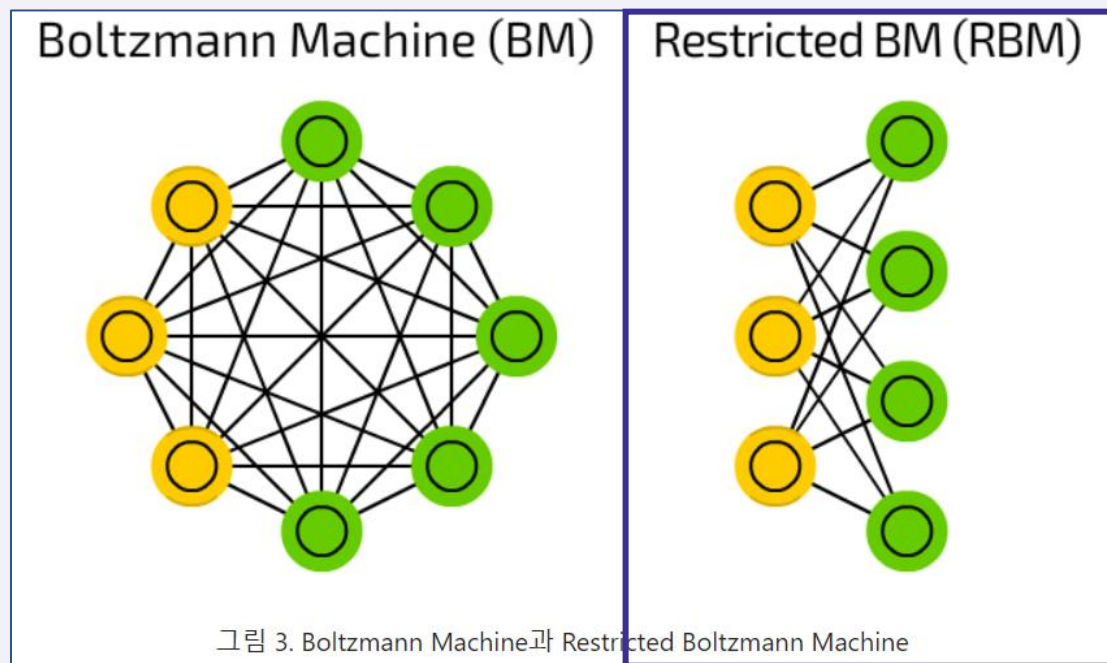
# 07

## Appendix

### 2.2 Deep Learning In Recommendation

#### Restricted Boltzmann Machine

- visible unit과 hidden unit에는 내부적인 연결이 없고, visible unit과 hidden unit간의 연결만이 남아있는 형태
- Boltzmann Machine의 계산이 너무 복잡해 지니까 이를 편하게 하기 위해 덜 엄격한 모델을 구성한 것



# 07

## Appendix

### 4.0 Data set

#### VIDEO

Youtube와 같은 OTT 비디오 서비스 플랫폼에서 수집된다. 일정 시 간 이상 동영상 시청한 이벤트를 수집했다.

특정 지역만 2개월 미만의 짧은 기간 동안 이 수집의 대상이 되었다. 이 시간 동안 화면 왼쪽에서 각 비디오 후에 항목 간 권장 사항이 제공되었다.

이들은 다양한 알고리즘을 선택하여 제공되었으며 사용자의 행동에 영향을 미쳤다. 전처리 단계는 봇에 의해 생성되었을 가능성이 있는 매우 긴 세션을

필터링하는 기능이 추가 된 다른 데이터 세트의 단계와 유사하다. 교육 데이터는 앞서 언급한 기간의 마지막 날을 제 외한 모든 날로 구성되며 33만 개의 동영상에

대해 최대 300만 세션의 1300만 개의 시청 이 벤트가 있다. 테스트 세트에는 수집 기간의 마지막 날의 세션이 포함되어 있으며 최대 37,000 개의 세션과

최대 180,000개의 시청 이벤트가 있다. 이 데이터 세트를 VIDEO라고 한다.

# 07

## Appendix

### 4.0 Data set

#### RSC15

첫 번째 데이터 세트는 RecSys Challenge 2015의 데이터 세트이다. 이 데이터 세트에는 구매 이벤트로 끝나는 e커머스 사이트의 클릭 스트림이 포함되어 있다. 우리는 챌린지의 훈련 세트로 작업하고 클릭 이벤트만 유지한다. 길이가 1인 세션을 필터링한다.

네트워크는 37,483 개 항목에 대한 31,637,239회의 클릭에 대한 7,966,257개의 세션을 포함하는 최대 6개월의 데이터에 대해 훈련된다. 테스트를 위해 다음 날의 세션을 사용한다. 각 세션은 훈련 또는 테스트 세트에 할당되며 세션 중간에 데이터를 분할하지 않는다.

협업 필터링 방법의 특성 때문에 클릭한 항목이 트레인 세트에 없는 테스트 세트에서 클릭을 필터링한다.

길이가 1인 세션 도 테스트 세트에서 제거된다. 전처리 후 테스트 세트에 대한 71,222개 이벤트의 15,324개 세션이 남는다.

이 데이터 세트를 RSC15라고 한다.