Session-based Recommendation with Graph Neural Networks

김태진

Prev

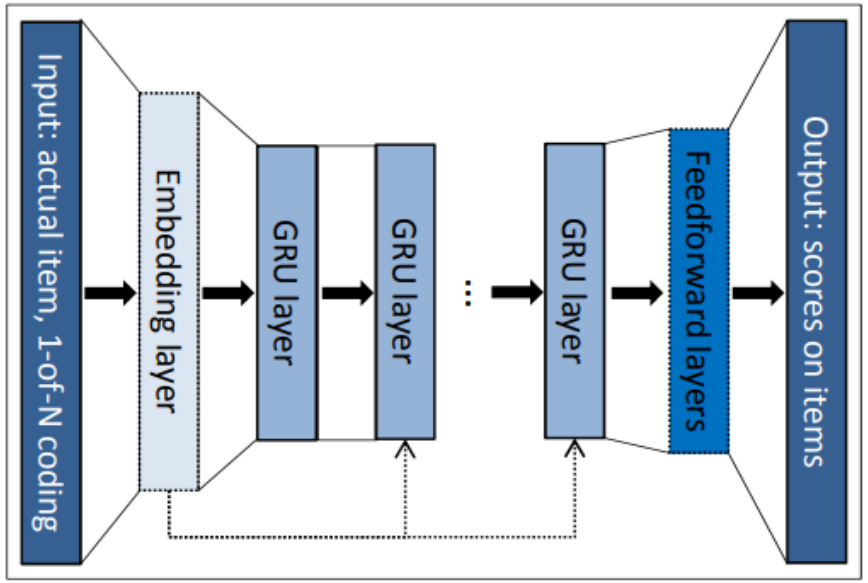
Session-based Recommendation ?

사용자 기반 프로필이 굳이 없더라도,

한 세션 안에서 만들어진 시퀀스 만을 가지고

마지막 클릭 이후, 다음 클릭할 아이템은 과연 무엇일까를 예측!

Prev



Session-Based with RNN

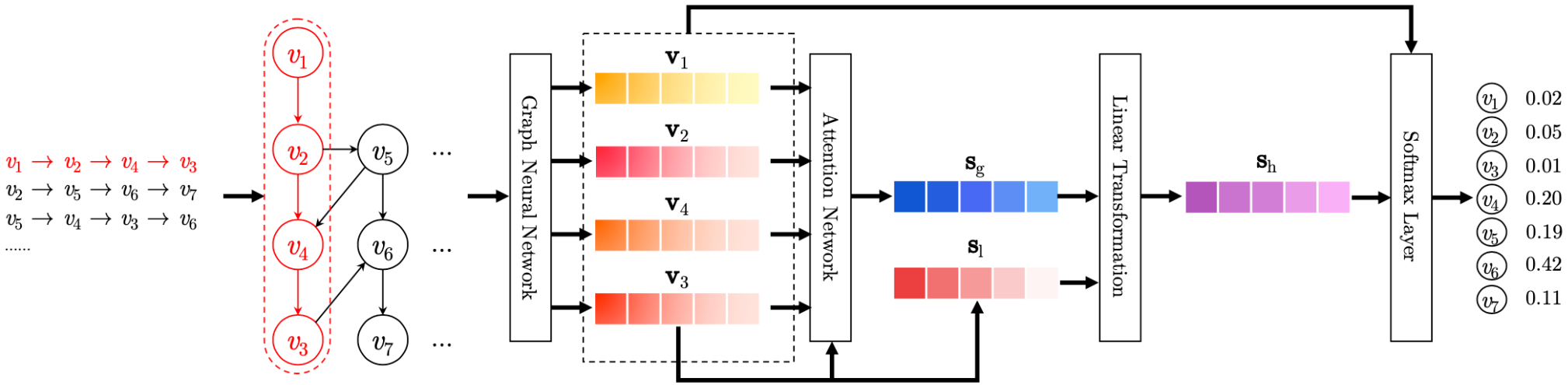
- Using user action sequence
- RNN (GRU) based

Cons..

- Just simple sequence data
- Long term dependency

Overview

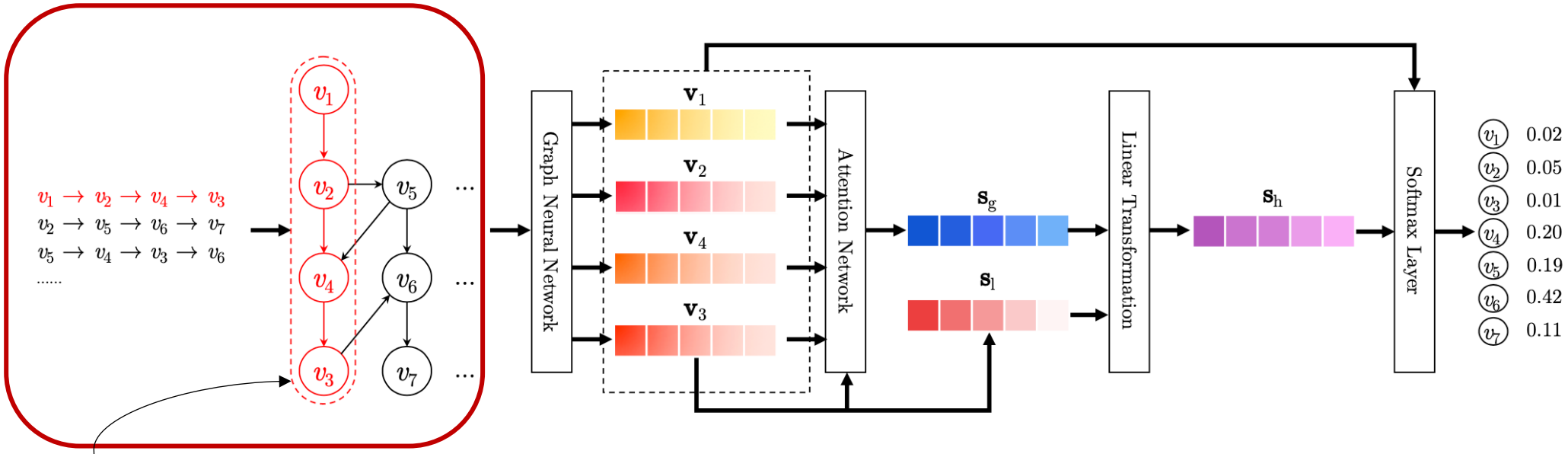
Session-Based with GNN



Capture complex transition of items in session

Predict next click item in that session

Step 1) Session Graph



Anonymous session sequence

$$s = [v_{s,1}, v_{s,2}, \dots v_{s,n}], \qquad v_{s,i} \in V$$

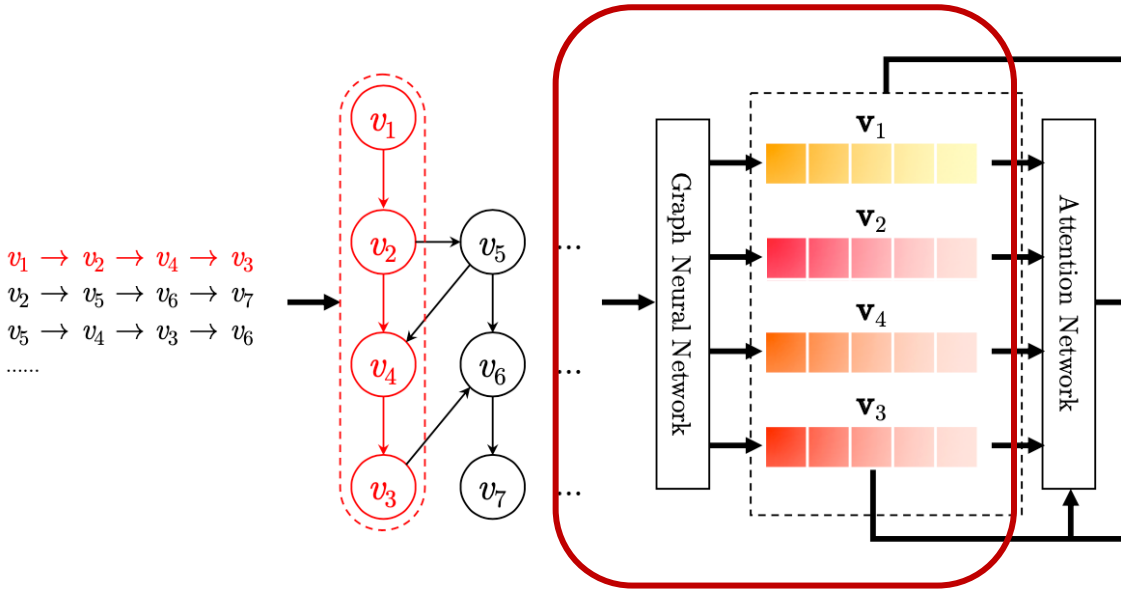
Item vector

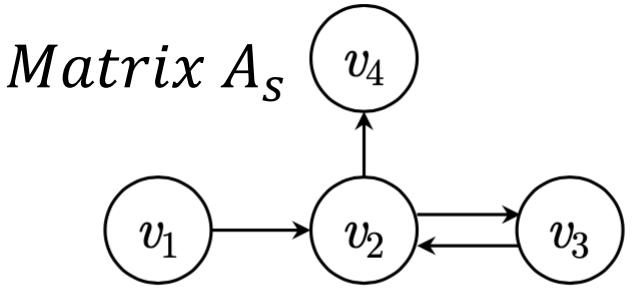
$$V = \{v_1, v_2 \dots, v_m\}$$

Ex) $v_1 \rightarrow v_2 \rightarrow v_4 \rightarrow v_3$

한 세션에서 이 순서대로 아이템을 클릭

Step 2) Item embedding



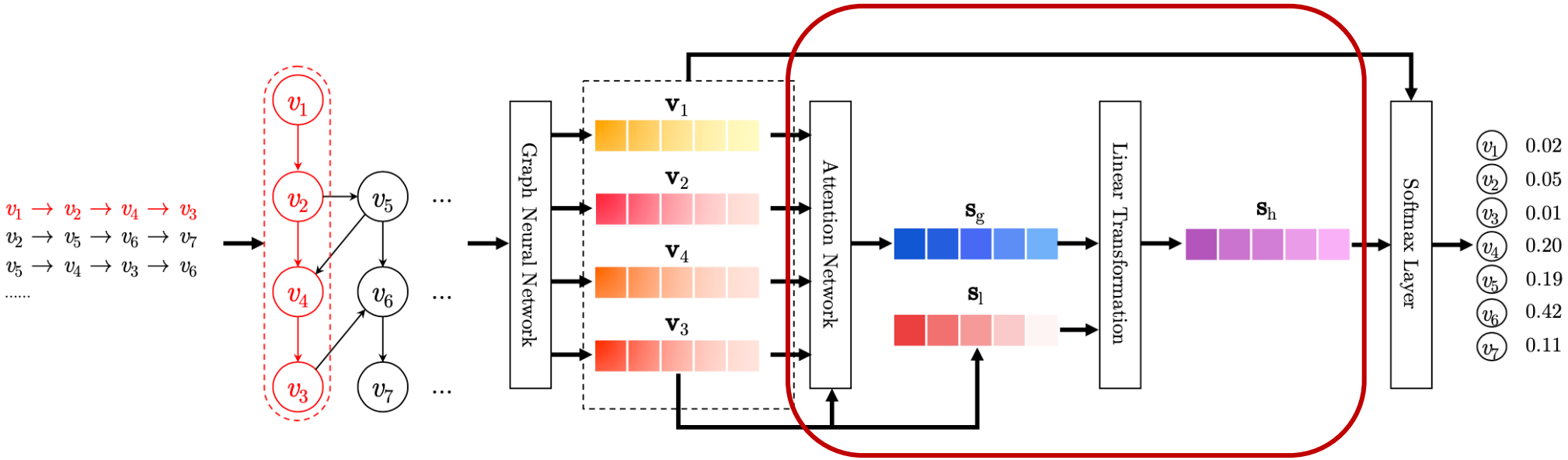


$$
\begin{aligned}
\mathbf{a}_{s,i}^{t} &= \mathbf{A}_{s,i:}\left[\mathbf{v}_1^{t-1}, \ldots, \mathbf{v}_n^{t-1}\right]^{\top}\mathbf{H} + \mathbf{b}, \\
\mathbf{z}_{s,i}^{t} &= \sigma\left(\mathbf{W}_z\mathbf{a}_{s,i}^{t} + \mathbf{U}_z\mathbf{v}_i^{t-1}\right), \\
\mathbf{r}_{s,i}^{t} &= \sigma\left(\mathbf{W}_r\mathbf{a}_{s,i}^{t} + \mathbf{U}_r\mathbf{v}_i^{t-1}\right), \\
\widetilde{\mathbf{v}}_i^{t} &= \tanh\left(\mathbf{W}_o\mathbf{a}_{s,i}^{t} + \mathbf{U}_o\left(\mathbf{r}_{s,i}^{t} \odot \mathbf{v}_i^{t-1}\right)\right), \\
\mathbf{v}_i^{t} &= \left(1 - \mathbf{z}_{s,i}^{t}\right) \odot \mathbf{v}_i^{t-1} + \mathbf{z}_{s,i}^{t} \odot \widetilde{\mathbf{v}}_i^{t},
\end{aligned}
$$

	Outgoing edges				Incoming edges			
	1	2	3	4	1	2	3	4
$A_{s,i:}$ 1	0	1	0	0	0	0	0	0
2	0	0	1/2	1/2	1/2	0	1/2	0
3	0	1	0	0	0	1	0	0
4	0	0	0	0	0	1	0	0

Learning Item Embedding on session graphs

Step 2) Session embedding



Soft attention mechanism

$$\alpha_i = \mathbf{q}^\top \sigma(\mathbf{W}_1 \mathbf{v}_n + \mathbf{W}_2 \mathbf{v}_i + \mathbf{c}),$$

$$\mathbf{s}_g = \sum_{i=1}^{n} \alpha_i \mathbf{v}_i,$$

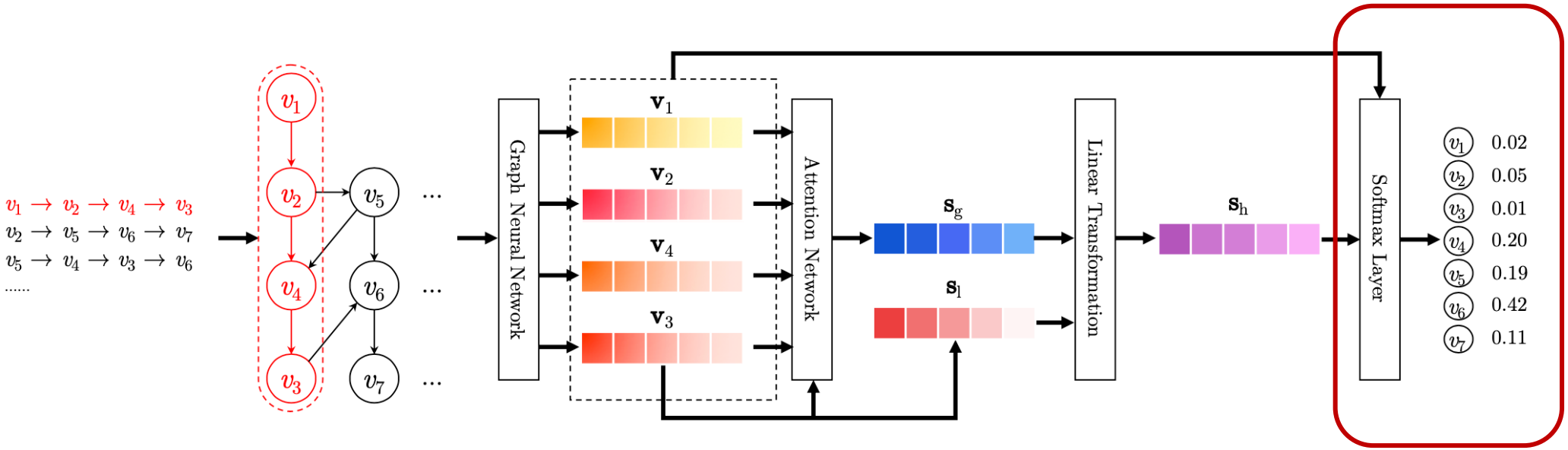
Hybrid embeddings

$$\mathbf{s}_l = \mathbf{v}_n \quad \text{Last click item}$$

$$\mathbf{s}_h = \mathbf{W}_3 [\mathbf{s}_l; \mathbf{s}_g] \quad \text{Concat}$$

$$\mathbf{W}_3 \in \mathbb{R}^{d \times 2d}$$

Step 3) Make result and training



Get score for each item vector

$$\hat{\mathbf{z}}_i = \mathbf{s}_h^\top \mathbf{v}_i.$$

$$\hat{\mathbf{y}} = \mathrm{softmax}\left(\hat{\mathbf{z}}\right), \quad \hat{\mathbf{z}} \in \mathbb{R}^m$$

Loss function (Cross Entropy)

$$\mathcal{L}(\hat{\mathbf{y}}) = -\sum_{i=1}^{m} \mathbf{y}_i \log\left(\hat{\mathbf{y}}_i\right) + \left(1 - \mathbf{y}_i\right) \log\left(1 - \hat{\mathbf{y}}_i\right),$$

Experiments - Datasets

Yoochoose
(Recsys Challenge 2015)

전자상거래 유저의
클릭 기록과 구매기록

Diginetica
(CIKM Cup 2016)

Personalized E-Commerce
Search Challenge

http://2015.recsyschallenge.com/challege.html

https://competitions.codalab.org/competitions/11161

Experiments - Baseline

- POP : 가장 인기 상품
- S-POP : 현재 세션의 가장 인기 상품
- Item-KNN : 같이 노출된 정도에 따라 아이템에 유사도 부여
- BRR-MF : 주로 사용되는 MF 알고리즘
- FPMC : Sequential prediction based on markov chain
- GRU4REC : Session based rec with RNN (지난 발표 논문)

딥러닝 베이스

- NARM : RNNs with attention for user's purpose and behavior
- STAMP : Captures users' general interests of current session

Result

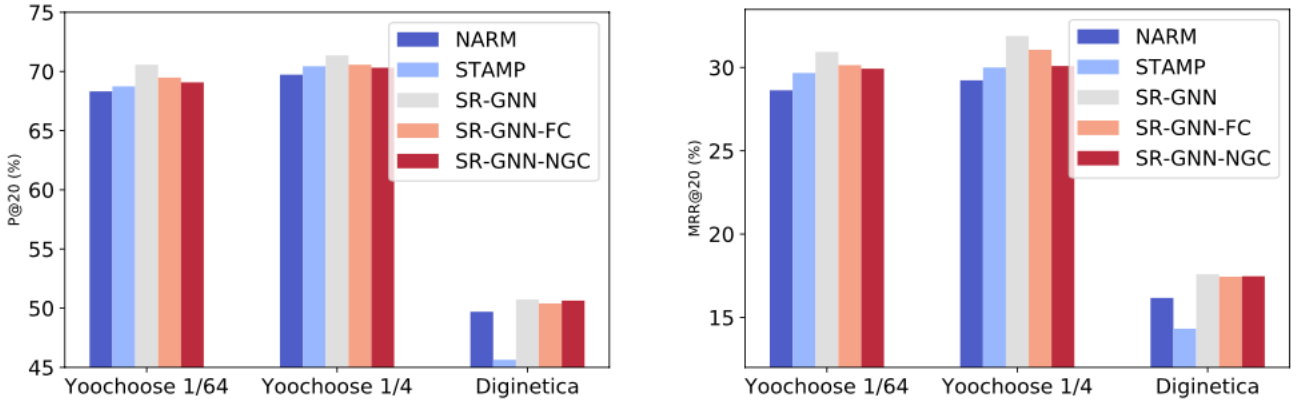
Method	Yoochoose 1/64		Yoochoose 1/4		Diginetica	
	P@20	MRR@20	P@20	MRR@20	P@20	MRR@20
POP	6.71	1.65	1.33	0.30	0.89	0.20
S-POP	30.44	18.35	27.08	17.75	21.06	13.68
Item-KNN	51.60	21.81	52.31	21.70	35.75	11.57
BPR-MF	31.31	12.08	3.40	1.57	5.24	1.98
FPMC	45.62	15.01	–	–	26.53	6.95
GRU4REC	60.64	22.89	59.53	22.60	29.45	8.33
NARM	68.32	28.63	69.73	29.23	49.70	16.17
STAMP	68.74	29.67	70.44	30.00	45.64	14.32
SR-GNN	**70.57**	**30.94**	**71.36**	**31.89**	**50.73**	**17.59**

More

Comparison with variants of Connection schemes

- SR-GNN : 각 시퀀스 별로 그래프를 구성 후 학습
- SR-GNN-NGC : 모든 시퀀스를 합쳐서 하나의 그래프로 생성
- SR-GNN-FC : 시퀀스를 시작과 끝만 남기고 중간 과정 생략

Ex) A - B - C => A - C



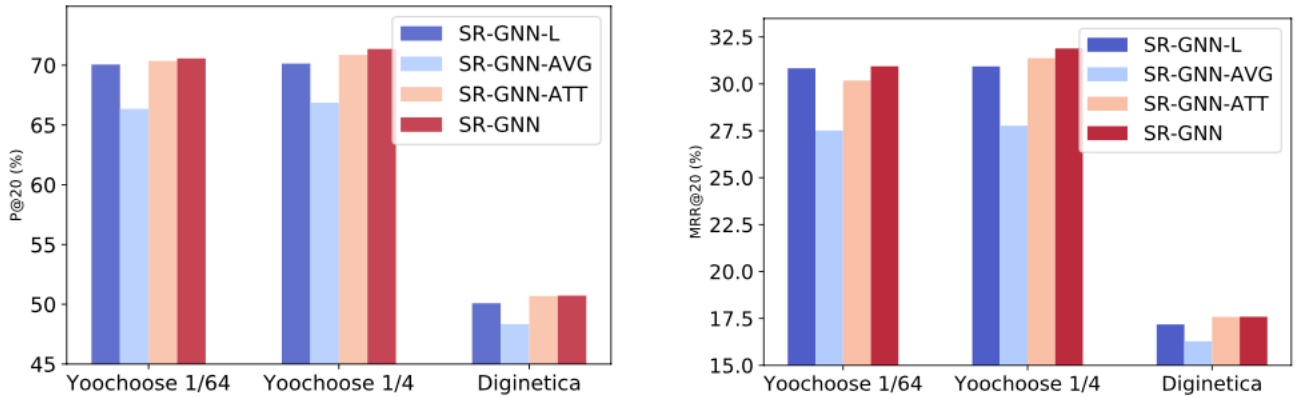
(a) P@20 (b) MRR@20

- NGC의 경우 Global Grpah는 subsequence의 특징에 집중하지 못하는 듯함
- FC의 경우 시작과 결과만 중요한 것이 아니라 중간에 거쳐간 아이템의 존재가 중요하다는 것을 반증

More

Comparison with different session embeddings

- SR-GNN-L : Local Embedding only (s_1)
- SR-GNN-AVG : Global Embedding with AVG Pooling
- SR-GNN-ATT : Global Embedding with Attention



(a) P@20 (b) MRR@20

- ATT의 성능이 나쁘지 않음, 세션별 중요한 Feature를 고르는 것이 필요하다.
- 단순히 마지막 클릭의 임베딩 만으로도 어느 정도 성능을 낼 수 있음.

Long-term preference와 current interest의 정보가 중요

More

Analysis on Session Sequence Lengths

시퀀스의 길이에 따른 성능변화 분석

Table 3: The performance of different methods with different session lengths evaluated in terms of P@20

Method	Yoochoose 1/64		Diginetica	
	Short	Long	Short	Long
NARM	**71.44**	60.79	**51.22**	45.75
STAMP	70.69	64.73	47.26	40.39
SR-GNN-L	70.11	69.73	49.04	50.97
SR-GNN-ATT	70.31	70.64	50.35	51.05
SR-GNN	70.47	**70.70**	50.49	**51.27**

- RNN 계열의 모델에서 Long-term Dependency 문제가 보임
- 그에 비해 SR-GNN은 아이템 간의 관계를 잘 Embedding 할 수 있어 길이가 긴 세션에서도 성능 감소가 없음

Impressions

- Long-term Dependency를 해결하려고 노력한 것은 좋은데
  GNN based가 실제 서비스로 반영할 만한 퍼포먼스를 낼 수 있을지?
  + 아이템이 겁나게 많다면?

- 앞서 소개된 딥러닝 논문 짬뽕에 GNN 소금치기

- GNN으로 만들어진 Embedding 벡터를 여러 조건에 검증해본 것은 좋은 접근

끝

모두들 고생했어요~~