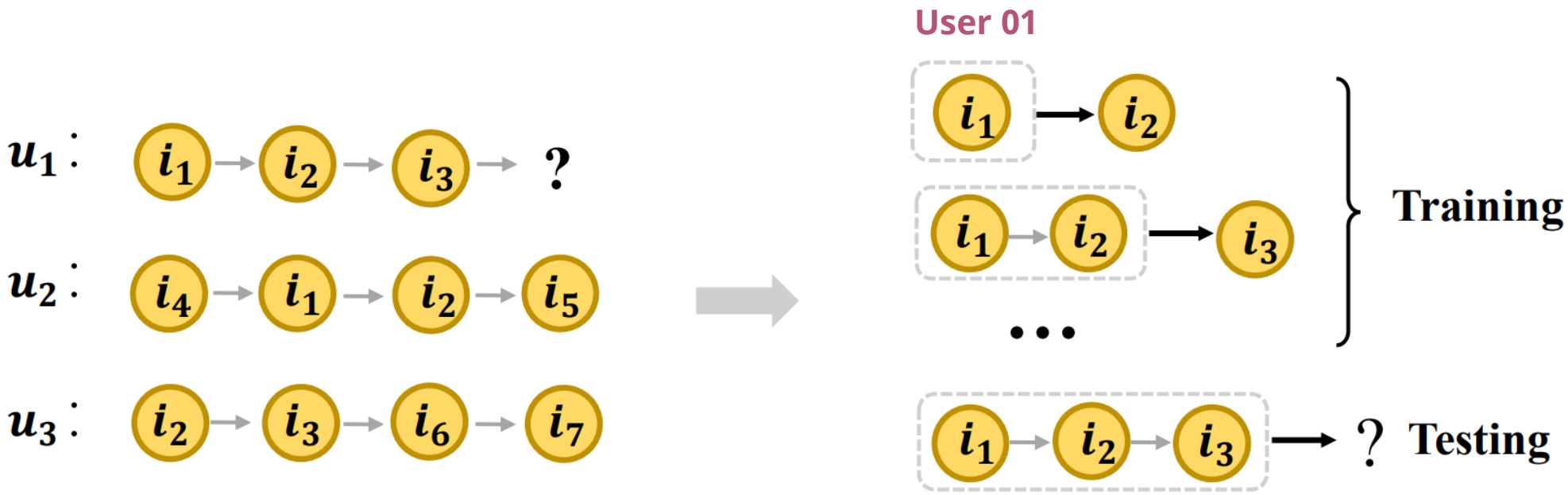
Graph Neural Network Approaches in Sequential Recommendation

Jihwan Lee 2022-05-13

Sequential Recommendation

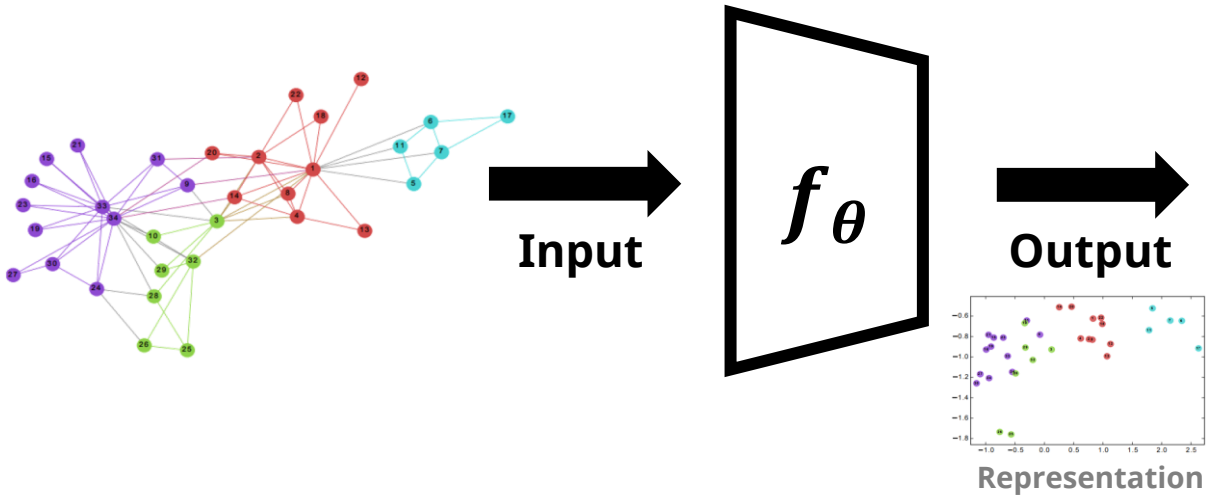


- Sequential Recommendation is to predict the next interaction of user.
- The right figure shows about the training and testing paradigm of most sequential models.

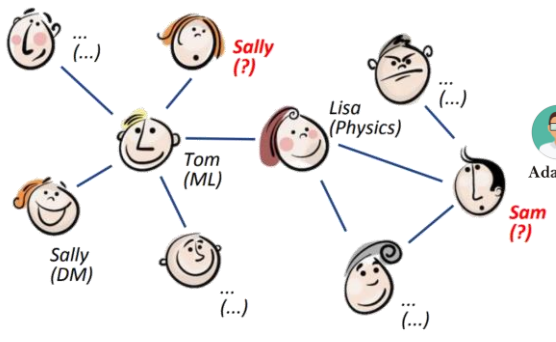
Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014.

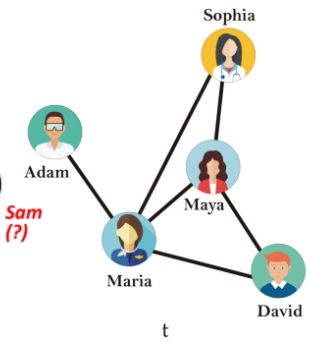
Graph Neural Network

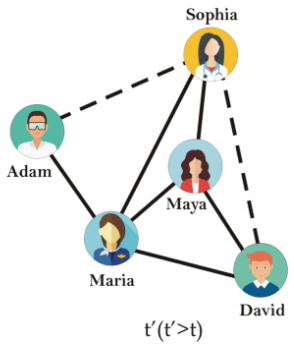
Graph Neural Network Tasks



Node Embedding



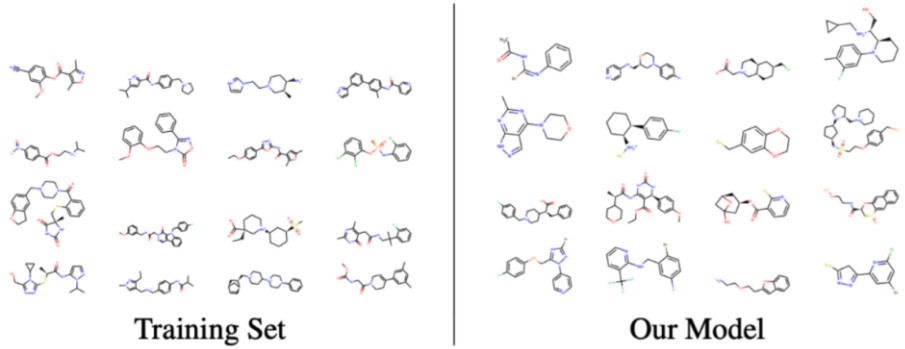




Node Classification

Edge(Link) Prediction

Graph Embedding



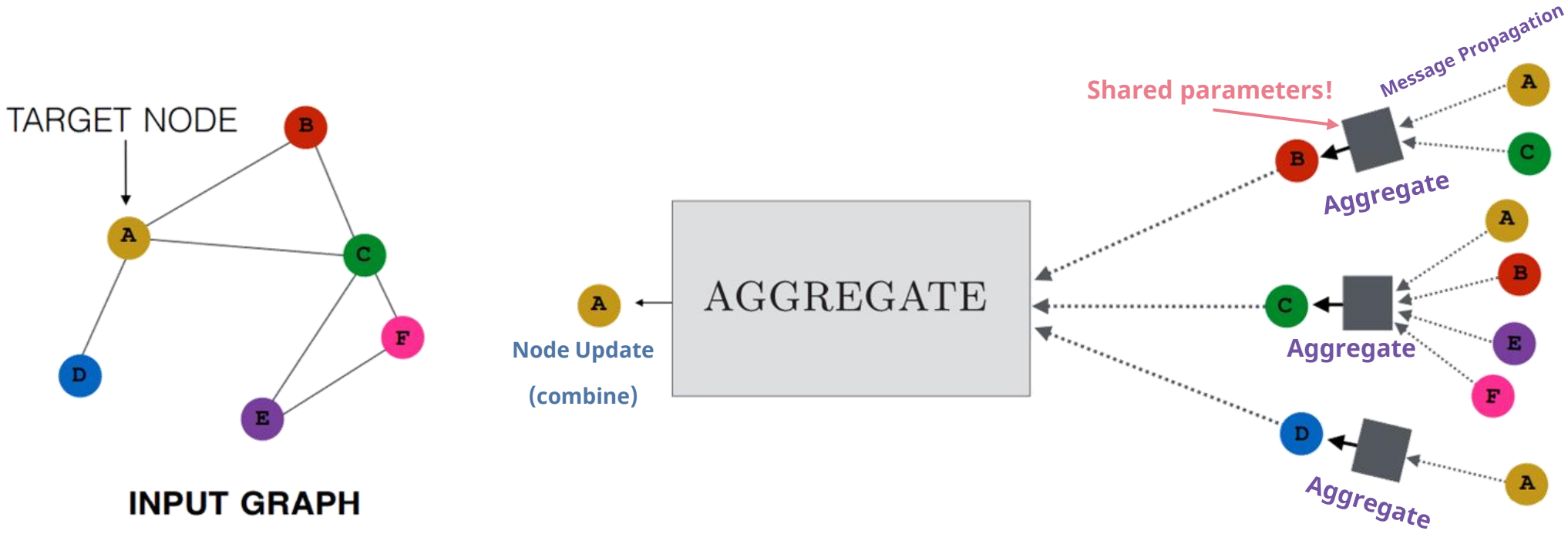
Graph Classification

- GNN's output is the process of converting a graph into a vector or set of vectors. (바꿔????!)
- graph embedding 의 output 벡터는 여러개일수도 하나일수도

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014.

Graph Neural Network

Structure of GNNs



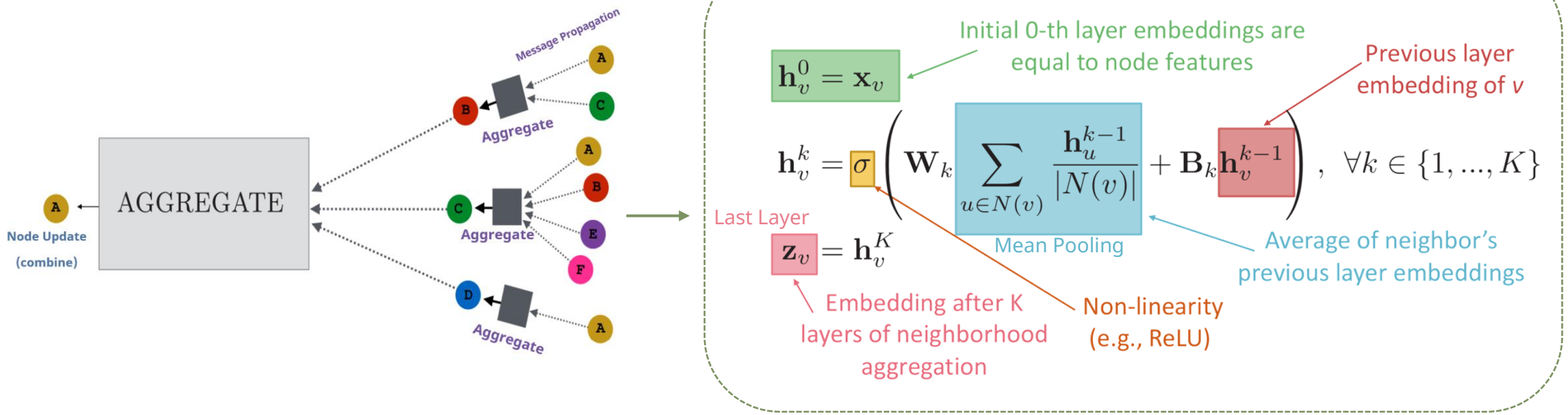
[fig 01] Example of node update process (target node: A, Number of layers:2)

- **In general, GNNs consist of two steps:**
  1. **Neighbor Aggregation**
  2. **Node Update (combine)**

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014.

Graph Neural Network

Single Layer of Graph Encoder



$$\mathbf{h}_v^0 = \mathbf{x}_v$$

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1}\right), \quad \forall k \in \{1, ..., K\}$$

$$\mathbf{z}_v = \mathbf{h}_v^K$$

[fig 02] Basic equations of GCN-model in training

- **Aggregating neighbor nodes and Defining the final node representation are also various.**
- **Node Aggregation Methods: [1] Mean-Pooling, [2] Max-Pooling,...** → **mean({ ● ● }) = mean({ ● ● ● ● })**
- **Final Node Representation: [1] Last hidden layer = Final Node Representation** → $\mathbf{h}_u^* = \mathbf{h}_u^{(L)}$

  [2] Mean,Sum,Weighted-Pooling, Concatenate all layers = Final Node Representation

Mean-pooling: $\mathbf{h}_u^* = \frac{1}{L+1} \sum_{l=0}^{L} \mathbf{h}_u^{(l)},$

Sum-pooling: $\mathbf{h}_u^* = \sum_{l=0}^{L} \mathbf{h}_u^{(l)},$

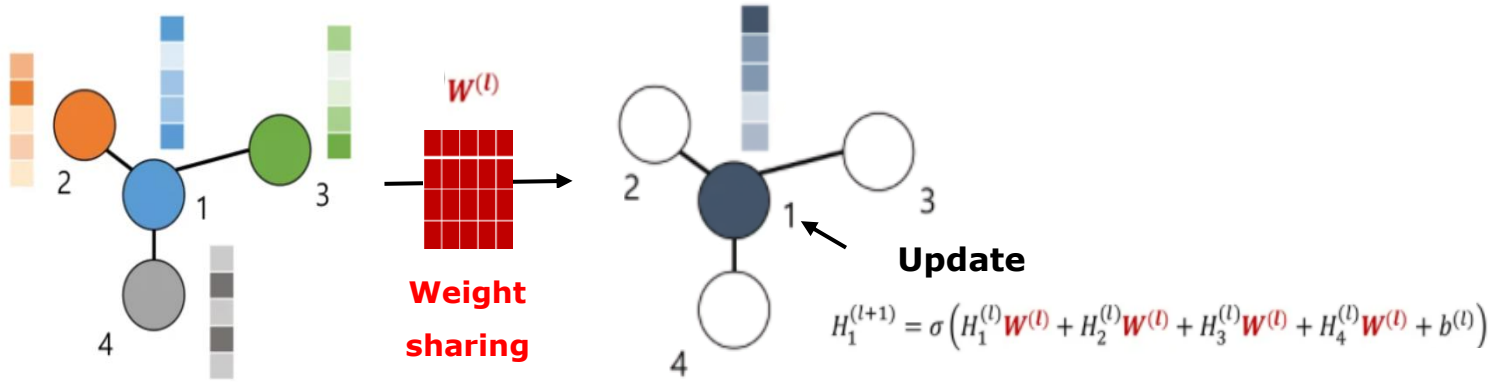
Weighted-pooling: $\mathbf{h}_u^* = \frac{1}{L+1} \sum_{l=0}^{L} \alpha^{(l)} \mathbf{h}_u^{(l)},$

Concatenation: $\mathbf{h}_u^* = \mathbf{h}_u^{(0)} \oplus \mathbf{h}_u^{(1)} \oplus \cdots \oplus \mathbf{h}_u^{(L)},$

Wu, Shiwen, et al. "Graph neural networks in recommender systems: a survey." *ACM Computing Surveys (CSUR)* (2020).

Graph Neural Network Approaches

Graph Convolutional Neural Network

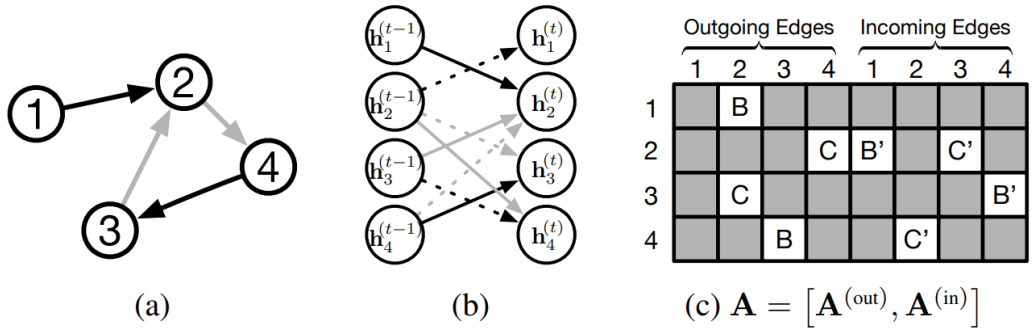


Combine

$$\mathbf{h}_u^L = \frac{1}{|\mathcal{N}_u|}\sum\nolimits_{i\in\mathcal{N}_u} \mathbf{W_1}^{(l-1)}\mathbf{h}_i^{(l-1)},$$

$$\mathbf{h}_i^L = \frac{1}{|\mathcal{N}_i|}\sum\nolimits_{u\in\mathcal{N}_i} \mathbf{W_2}^{(l-1)}\mathbf{h}_u^{(l-1)},$$

Graph Gated Neural Network



Aggregation

$$a_v^{(k-1)} = \sum\nolimits_{u\in N(v)} \sum h_u^{(k-1)}$$

Combine

$$h_v^{(k)} = GRU(h_v^{(k-1)}, a_v^{(k-1)})$$

Li, Yujia, et al. "Gated graph sequence neural networks." *arXiv preprint arXiv:1511.05493* (2015).

Berg, Rianne van den, Thomas N. Kipf, and Max Welling. "Graph convolutional matrix completion." *arXiv preprint arXiv:1706.02263* (2017).

Graph Neural Network in Sequential Recommendation

Limitation of directed graphs (SR-GNN)

list of the node vectors in session s

$$\mathbf{a}_{s,i}^t = \mathbf{A}_{s,i:} \left[\mathbf{v}_1^{t-1}, \ldots, \mathbf{v}_n^{t-1}\right]^\top \mathbf{H} + \mathbf{b},$$

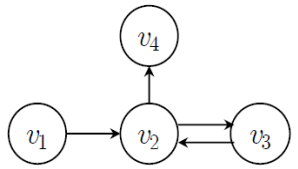
$$\mathbf{z}_{s,i}^t = \sigma\left(\mathbf{W}_z \mathbf{a}_{s,i}^t + \mathbf{U}_z \mathbf{v}_i^{t-1}\right), \quad \text{Update gate}$$

$$\mathbf{r}_{s,i}^t = \sigma\left(\mathbf{W}_r \mathbf{a}_{s,i}^t + \mathbf{U}_r \mathbf{v}_i^{t-1}\right), \quad \text{reset gate}$$

$$\widetilde{\mathbf{v}}_i^t = \tanh\left(\mathbf{W}_o \mathbf{a}_{s,i}^t + \mathbf{U}_o\left(\mathbf{r}_{s,i}^t \odot \mathbf{v}_i^{t-1}\right)\right),$$

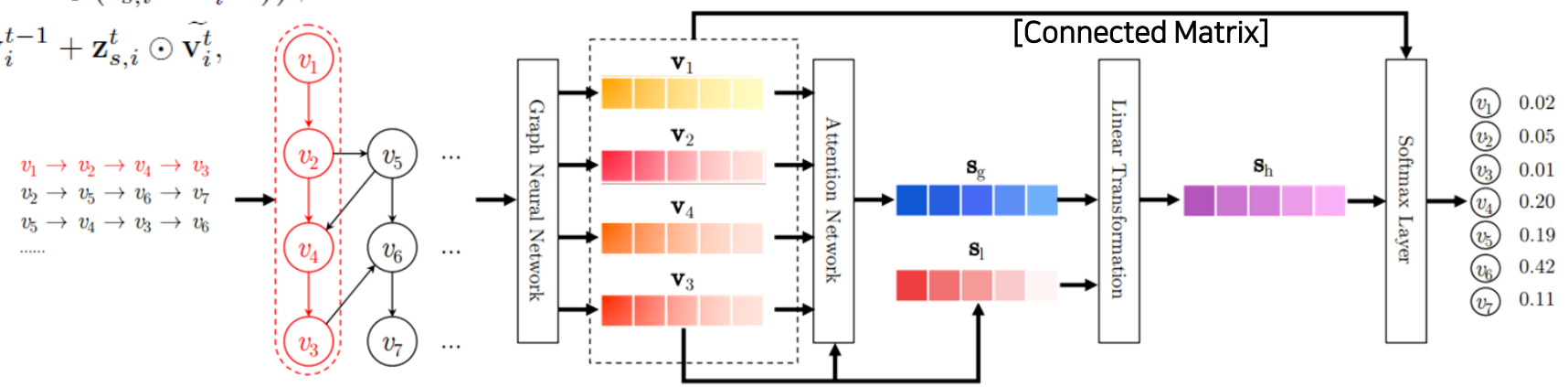
$$\mathbf{v}_i^t = \left(1 - \mathbf{z}_{s,i}^t\right) \odot \mathbf{v}_i^{t-1} + \mathbf{z}_{s,i}^t \odot \widetilde{\mathbf{v}}_i^t,$$

$$s = [v_1, v_2, v_3, v_2, v_4]$$



	Outgoing edges				Incoming edges			
	1	2	3	4	1	2	3	4
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

matrix $\mathbf{A}_s$

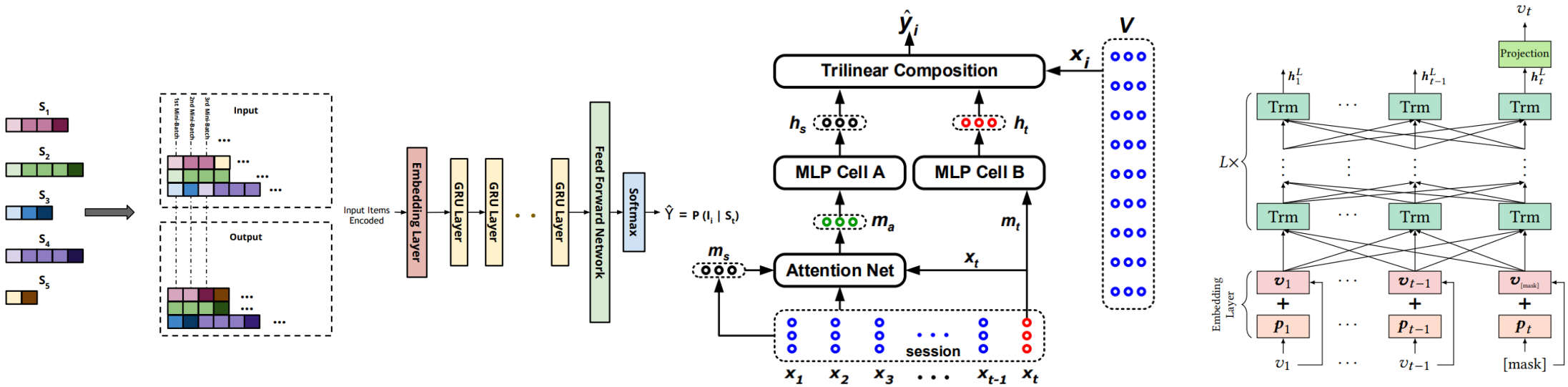


[fig 03] SR-GNN framework

- SR-GNN learns embedding of session graphs by using a GNN to aggregate item embedding.
- It fails to model temporal collaborative signals because there is no time information in the edges.

Wu, Shu, et al. "Session-based recommendation with graph neural networks." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. No. 01. 2019.

Main problems in SR Recommendation



(b) BERT4Rec model architecture.

[fig 04] GRU4Rec,STAMP,BERT4Rec

- SR models only focus on the item transitions within sequences.
- It unable to unify the import temporal collaborative signals within sequential patterns.
- It can’t generalized to unseen timestamps.
- It designed to capture sequential patterns, while ignoring the important temporal information.

Sun, Fei, et al. "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer." *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019.

Main problems in SR Recommendation

1. It is hard to simultaneously encode collaborative signals and sequential patterns.

   ex) only consider the sequential patterns

   u1,u3: i2 → i3 then, u4 at t5 i2 → i3 will be recommended.

   ++) ----- collaborative signal

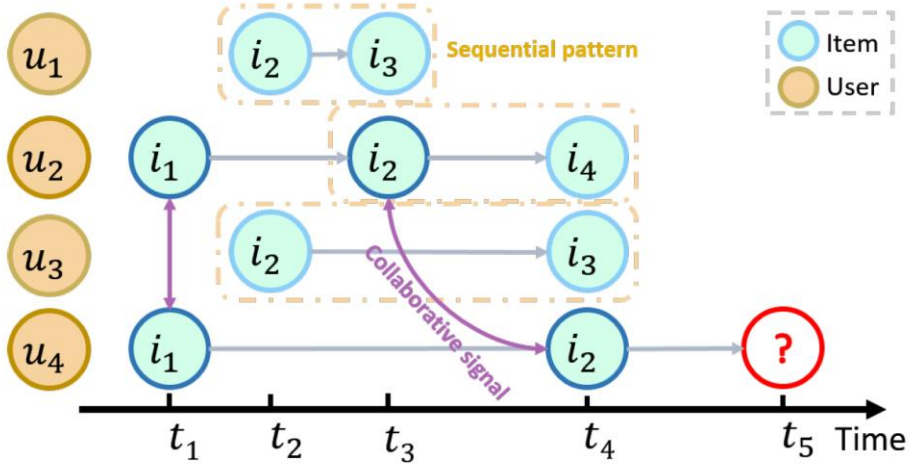
   u2 and u4 pattern same, u4 at t5 t5 i2 → i4 will be recommended.

2. It is hard to express the temporal effects of collaborative signals.

   ex) i1 is interacted with u2,u4 at t1.

   i2 is interacted with u2 and u3 respectively at t3 and t4.

   → **Problem**: It is problematic to ignore the time gap and assume that they are of <u>equal contributions.</u>

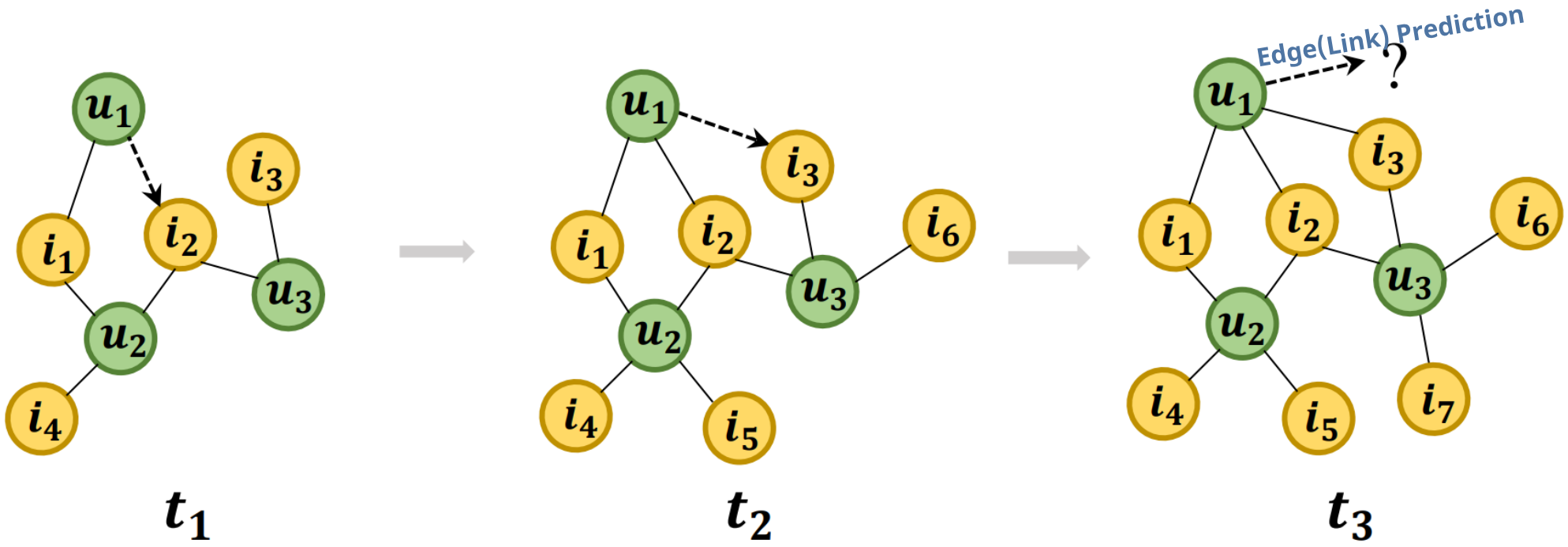


[fig 05] Sequential Recommendation problems

Dynamic Graph Neural Networks for Sequential Recommendaion

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

“Dynamic Graph Neural Network for Sequential Recommendation(DGSR)”

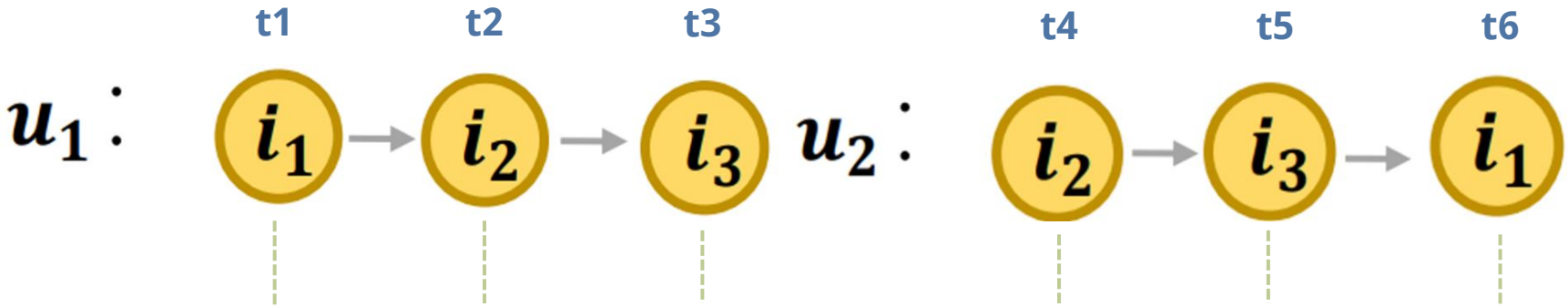


- Dynamic graph neural network connects different user sequence through a dynamic graph structure.
- It can explore the interactive behavior of users and items with time and order information.
- After constructing the dynamic graph, graph can extract user’s preference.
- The task is changed to link prediction between user node and item node.

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

STEP 01) Dynamic Graph Construction



$$\mathcal{G} = \{(u, i, t, o_u^i, o_i^u) | u \in \mathcal{U}, i \in \mathcal{V}\}$$

o_u^i Position of item i in all items that the u has interacted with

o_i^u The order of u in all user nodes that have interacted with item i

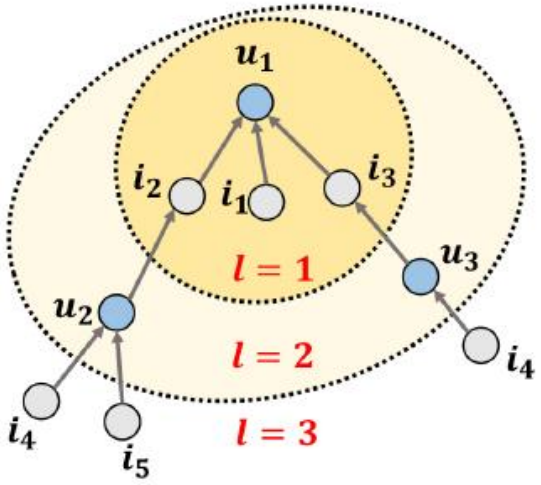
$$(u_1, i_1, t_1, 1, 1)\ (u_1, i_2, t_2, 2, 1)\ (u_1, i_3, t_3, 3, 1)\ (u_2, i_2, t_4, 1, 2)\ (u_2, i_1, t_5, 2, 2)\ (u_2, i_3, t_6, 3, 2)$$

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

STEP 02) Sub-Graph Sampling – definition of multi-hop

“Multi-Hop Neighbors”

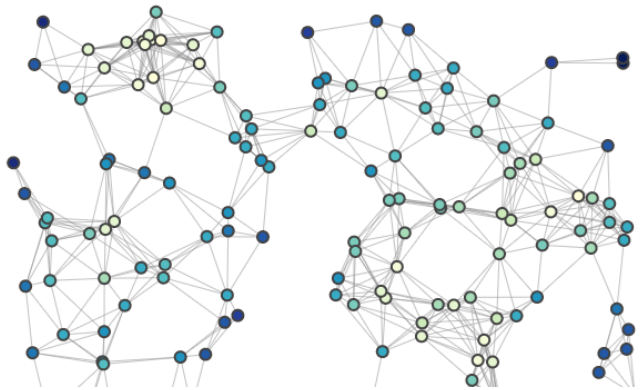


- Since no direct links exist between users or items, and thus the information should be propagated via multi-hop neighbor nodes.
- For instance, to propagate some information from user u1 to a similar user u2 one needs to first propagate it to a bridge item v1 connecting both users,and then to u2 from v1.

Wang, Shoujin, et al. "Graph learning based recommender systems: A review." *arXiv preprint arXiv:2105.06339* (2021).

DGSR

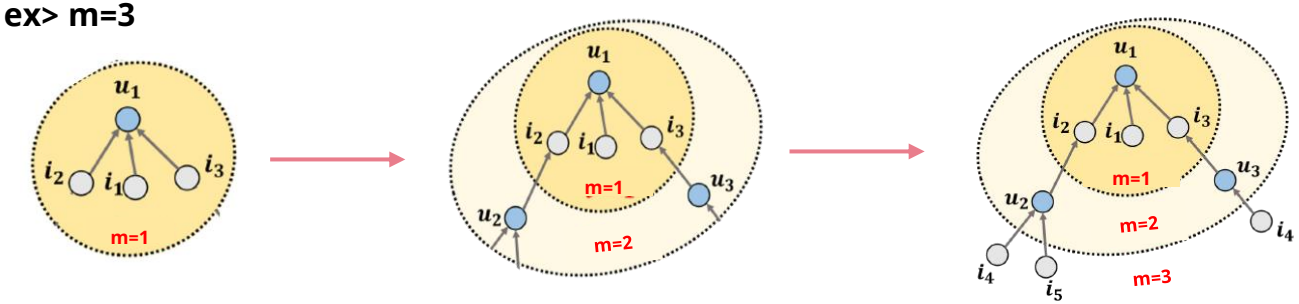
STEP 02) Sub-Graph Sampling



t1,t2,….,tn → Graph is gradually expanding → Increase the computational cost → too much noise to predict target sequence

Sub-graph sampling: $\mathcal{G}_u^m(t_k) \in \mathcal{G}$

ex> m=3



Algorithm 1: Sub-graph Sampling Algorithm

Input : Sequence $S^u = (i_1, i_2, \cdots, i_k)$, timestamp sequence $T^u = (t_1, t_2, \cdots, t_k)$, dynamic graph $\mathcal{G}^{t_k}$, and the order of sub-graph m.

Output: The m-order sub-graph $\mathcal{G}_u^m(t_k)$. **m: hyperparameter**

```
// Initialization
U_m, U_temp ← {u}, I_m, I_temp ← ∅
// Node sampling
for k ∈ [1, .., m] do
    if k is an odd number then
        for u ∈ U_temp do
            I_temp ← I_temp ∪ N_u
        I_temp ← I_temp \ I_m   // Itemp set means that it doesn't include item list for user Utemp
        if I_temp = ∅ then
            Break
        I_m ← I_m ∪ I_temp      // Avoid repeated sampling
    else
        for i ∈ I_temp do
            U_temp ← U_temp ∪ N_i
        U_temp ← U_temp \ U_m
        if U_temp = ∅ then
            Break
        if |U_temp| > n then
            U_temp ← Sampling n nodes from U_temp
        U_m ← U_m ∪ U_temp
// Sub-graph generation
G_u^m(t_k) = (U_m, I_m), U_m, I_m ∈ G^{t_k}
```

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

STEP 03) Dynamic Graph Recommenation Network

* Message Propagation – Dynamic Graph Attention Mechanism

- GCN and GAT are powerful in various graph structure data. But, they fail to capture sequential information of neighbors suitably for each user and item.

- DGSR consider both long-term information and short-term information when prepare the message propagation.

  [long-term information]

  (1) from item to user: general preference

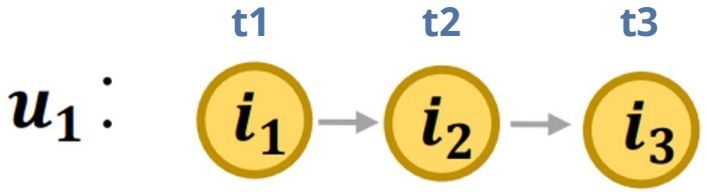
  (2) from user to item: general characters of item

  [short-term information]

  from item to user: his or her latest interest, from user to item: the newest property of item (i.e,player poster during the world cup season)

- DSGR suggests new message propagation method, which name is dynamic graph attention mechanism.

- DSGR introduces new term $r_u^i = |\mathcal{N}_u| - o_u^i$ as the relative-order of item I to the last item in the neighbors of user node.



$$r_u^i = |\mathcal{N}_u| - o_u^i$$

$(u_1, i_1, t_1, 1, 1)$ $(u_1, i_2, t_2, 2, 1)$ $(u_1, i_3, t_3, 3, 1)$

$r_{u_1}^1 = 3 - 1 = 2$

$r_{u_1}^2 = 3 - 2 = 1$

$r_{u_1}^3 = 3 - 3 = 0$

Last item → value ↓

$$\mathbf{p}_{r_u^i}^K \in \mathbb{R}^d$$

Parameter vector as the

relative-order embedding to ecode the order information

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

STEP 03) Dynamic Graph Recommenation Network

* Message Propagation – Dynamic Graph Attention Mechanism

Long-term construction

$$e_{ui} = \frac{1}{\sqrt{d}}\left(\mathbf{W_2}^{(l-1)}\mathbf{h}_u^{(l-1)}\right)^{\mathrm{T}}\left(\mathbf{W_1}^{(l-1)}\mathbf{h}_i^{(l-1)} + \mathbf{p}_{r_u^i}^{K}\right)$$

$$\alpha_{ui} = \mathrm{softmax}(e_{ui})$$

- long-term preference of user

$$\mathbf{h}_u^L = \sum\nolimits_{i\in\mathcal{N}_u} \alpha_{ui}\left(\mathbf{W_1}^{(l-1)}\mathbf{h}_i^{(l-1)} + \mathbf{p}_{r_u^i}^{V}\right)$$

Aggregating the information from its all neighbor (item)

- long-term character of item calculate same steps

$$\mathbf{h}_i^L = \sum\nolimits_{u\in\mathcal{N}_i} \beta_{iu}\left(\mathbf{W_2}^{(l-1)}\mathbf{h}_u^{(l-1)} + \mathbf{p}_{r_i^u}^{V}\right),$$

where

$$\beta_{iu} = \mathrm{softmax}(e_{iu}),$$

$$e_{iu} = \frac{1}{\sqrt{d}}\left(\mathbf{W_1}^{(l-1)}\mathbf{h}_i^{(l-1)}\right)^{\mathrm{T}}\left(\mathbf{W_2}^{(l-1)}\mathbf{h}_u^{(l-1)} + \mathbf{p}_{r_i^u}^{K}\right)$$

Short-term information

It consider attention mechanism to model the expliect effectiveness between last interaction with historical interactions.

Last interaction / Historical interactions

$$\mathbf{h}_u^S = \sum\nolimits_{i\in\mathcal{N}_u} \hat{\alpha}_{ui}\mathbf{h}_i^{(l-1)}, \tag{11}$$

$$\mathbf{h}_i^S = \sum\nolimits_{u\in\mathcal{N}_i} \hat{\beta}_{iu}\mathbf{h}_u^{(l-1)}, \tag{12}$$

where attention coefficient $\hat{\alpha}_{i_k}$ and $\hat{\beta}_{u_k}$ can be calculated by,

$$\hat{\alpha}_{ui} = \mathrm{softmax}\left(\frac{1}{\sqrt{d}}\left(\mathbf{W}_3^{(l-1)}\mathbf{h}_{i_{last}}^{(l-1)}\right)^{\mathrm{T}}\left(\mathbf{W}_2^{(l-1)}\mathbf{h}_i^{(l-1)}\right)\right), \tag{13}$$

$$\hat{\beta}_{iu} = \mathrm{softmax}\left(\frac{1}{\sqrt{d}}\left(\mathbf{W}_4^{(l-1)}\mathbf{h}_{u_{last}}^{(l-1)}\right)^{\mathrm{T}}\left(\mathbf{W}_1^{(l-1)}\mathbf{h}_u^{(l-1)}\right)\right), \tag{14}$$

Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

STEP 03) Prediction

* Node Updating (Combine)

$$\mathbf{h}_u^{(l)} = \tanh\left(\mathbf{W_5}^{(l)}\left[\mathbf{h}_u^L \parallel \mathbf{h}_u^S \parallel \mathbf{h}_u^{(l-1)}\right]\right) \leftarrow \begin{aligned}\mathbf{h}_u^L &= \textstyle\sum_{i\in\mathcal{N}_u} \alpha_{ui}\left(\mathbf{W_1}^{(l-1)}\mathbf{h}_i^{(l-1)} + \mathbf{p}^V_{r_i^u}\right)\\ \mathbf{h}_u^S &= \textstyle\sum_{i\in\mathcal{N}_u} \hat{\alpha}_{ui}\mathbf{h}_i^{(l-1)}\end{aligned}$$

$$\mathbf{h}_i^{(l)} = \tanh\left(\mathbf{W_6}^{(l)}\left[\mathbf{h}_i^L \parallel \mathbf{h}_i^S \parallel \mathbf{h}_i^{(l-1)}\right]\right) \leftarrow \begin{aligned}\mathbf{h}_i^L &= \textstyle\sum_{u\in\mathcal{N}_i} \beta_{iu}\left(\mathbf{W_2}^{(l-1)}\mathbf{h}_u^{(l-1)} + \mathbf{p}^V_{r_i^u}\right)\\ \mathbf{h}_i^S &= \textstyle\sum_{u\in\mathcal{N}_i} \hat{\beta}_{iu}\mathbf{h}_u^{(l-1)},\end{aligned}$$

* Recommendation & Optimization

$$\mathbf{h_u} = \mathbf{h}_u^{(0)} \parallel \mathbf{h}_u^{(1)} \cdots \parallel \mathbf{h}_u^{(L)}.$$

Score $\mathbf{s}_{ui} = \mathbf{h_u}^{\mathrm{T}}\mathbf{W_P}\mathbf{e}_i$ candidate item $i \in \mathcal{I}$

$$\mathbf{s}_u = \left(\mathbf{s}_{u1}, \mathbf{s}_{u2}, \cdots, \mathbf{s}_{u|\mathcal{I}|}\right)$$

optimization

$$\hat{\mathbf{y}}_u = \mathrm{softmax}(\mathbf{s}_u).$$

$$Loss = -\sum_{S}\sum_{i=1}^{|\mathcal{I}|} \mathbf{y}_{ui}\log(\hat{\mathbf{y}}_{ui}) + (1-\mathbf{y}_{ui})\log(1-\hat{\mathbf{y}}_{ui}) + \lambda\|\Theta\|_2,$$

Algorithm 2: The DGSR framework (forward propagation)

Input : $S^u = (i_1, i_2, \cdots, i_k)$, timestamp sequence $T^u = (t_1, t_2, \cdots, t_k)$, all sequences of users, and DGRN layer number L.
Output: The next item i_{k+1} of S^u.

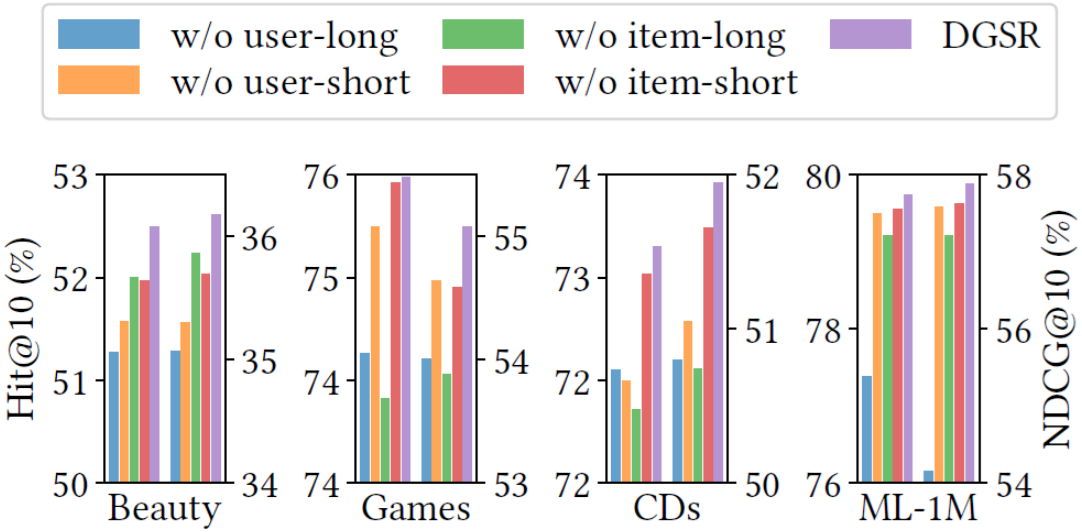
// Dynamic Graph Construction.
Convert all user sequences into a dynamic graph $\mathcal{G}$
// The sub-graph of generation for S^u.
Run the **Algorithm 1** to generate $\mathcal{G}_u^m(t_k)$ from $\mathcal{G}^{t_k}$
// The initialization of node representation.
$\mathbf{h}_u^{(0)} \leftarrow \mathbf{e}_u, \mathbf{h}_i^{(0)} \leftarrow \mathbf{e}_i, \forall u, i \in \mathcal{G}_u^m(t_k)$
// The update of user and item by DGRN.
for $l \in [1 : L]$ **do**
  $\mathbf{h}_u^{(l)}, \mathbf{h}_i^{(l)} \leftarrow \mathrm{DGRN}(\mathbf{h}_u^{(l-1)}, \mathbf{h}_i^{(l-1)}, \mathcal{G}_u^m(t_k))$:
  $\mathbf{h}_u^L, \mathbf{h}_i^L \leftarrow$ Long-term Information Encoding
  $\mathbf{h}_u^S, \mathbf{h}_i^S \leftarrow$ Short-term Information Encoding
  $\mathbf{h}_u^{(l)} \leftarrow \tanh\left(\mathbf{W_5}^{(l)}\left[\mathbf{h}_u^L \parallel \mathbf{h}_u^S \parallel \mathbf{h}_u^{(l-1)}\right]\right)$
  $\mathbf{h}_i^{(l)} \leftarrow \tanh\left(\mathbf{W_6}^{(l)}\left[\mathbf{h}_i^L \parallel \mathbf{h}_i^S \parallel \mathbf{h}_i^{(l-1)}\right]\right)$
// The prediction of next item.
$\mathbf{h_u} = \mathbf{h}_u^{(0)} \parallel \mathbf{h}_u^{(1)}, \cdots, \parallel \mathbf{h}_u^{(L)}$
Next item $\leftarrow \mathrm{argmax}_{i\in\mathcal{V}}(\mathbf{h_u}^{\mathrm{T}}\mathbf{W_P}\mathbf{e}_i)$

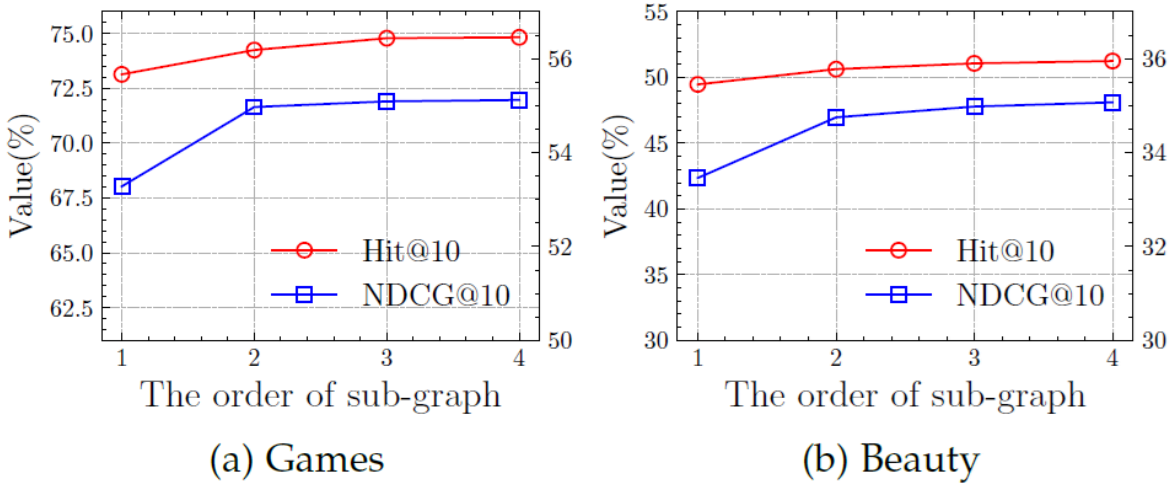
Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

DGSR

Experiments

Datasets	Metric	BPR-MF	FPMC	GRU4Rec+	Caser	SASRec	SR-GNN	HGN	TiSASRec	GCE-GNN	SERec	HyperRec	**DGSR**
Beauty	NDCG@10	21.83	28.91	26.42	25.47	32.19	32.33	32.47	30.45	33.46	33.59	23.26	**35.90**
	Hit@10	37.75	43.10	43.98	42.64	48.54	48.62	48.63	46.87	49.12	49.63	34.71	**52.40**
Games	NDCG@10	28.75	46.80	45.64	45.93	53.60	53.25	49.34	50.19	53.68	53.71	48.96	**55.70**
	Hit@10	37.75	68.02	67.15	68.83	73.98	73.49	71.42	71.85	74.14	74.32	71.24	**75.57**
CDs	NDCG@10	36.26	33.55	44.52	45.85	49.23	48.95	49.34	48.97	49.05	48.34	47.16	**51.22**
	Hit@10	56.27	51.22	67.84	68.65	71.32	69.63	71.42	71.00	70.04	69.44	71.02	**72.43**
ML-1M	NDCG@10	32.87	48.66	53.14	53.29	57.34	54.23	52.57	56.57	56.31	57.01	54.61	**57.89**
	Hit@10	57.81	72.77	73.21	76.81	80.36	76.59	76.57	**80.41**	77.4	78.39	78.24	79.74



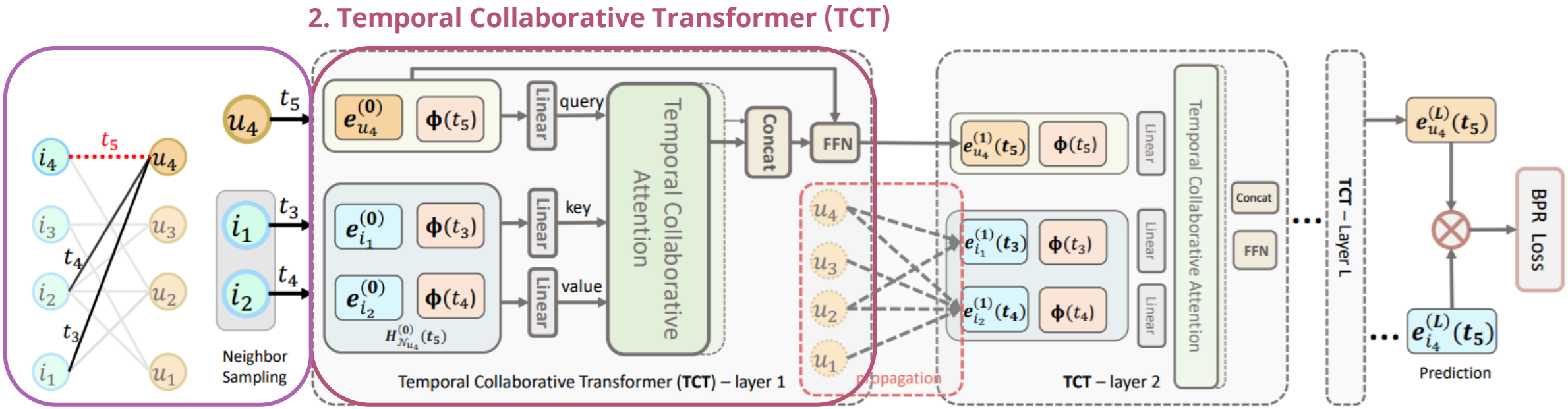


Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

Continuous-Time Sequential Recommendation with Temporal Graph Collaborative Transformer

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

Proposed Model - Temporal Graph Sequential Recommender (TGSRec)



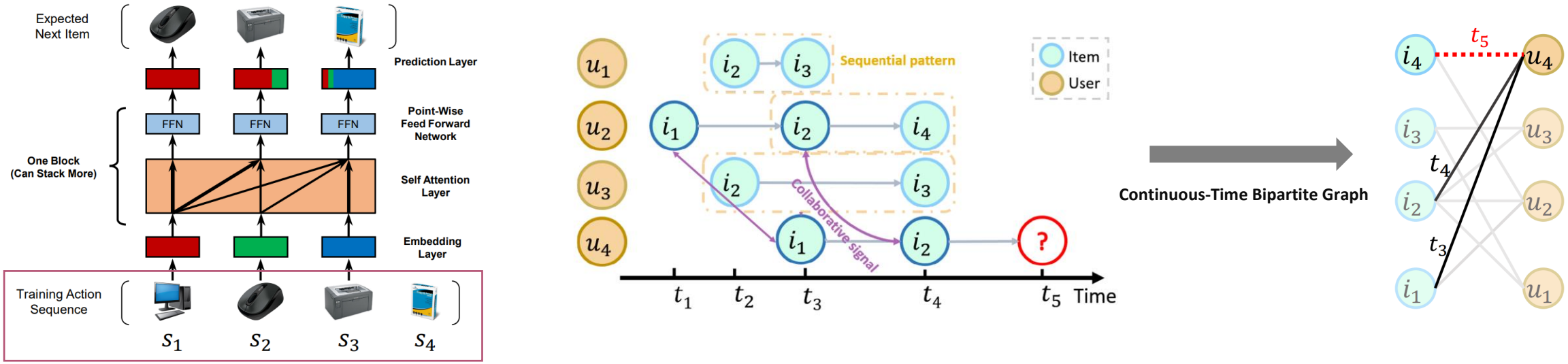
1. CTBG Layer: CTBG layer unifies sequential patterns with temporal collaborative signals.

2. TCT Layer: TCT layer uses temporal information to discriminate impacts of those historical interactions and and makes inferences of temporal node embeddings.

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

01. Embedding Layer



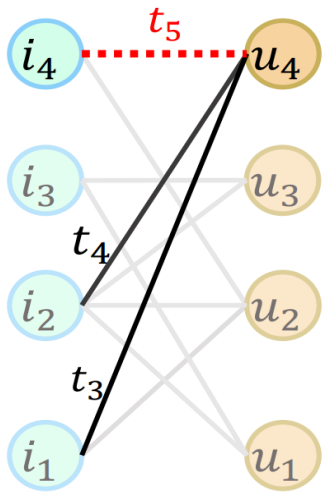
[fig 06] Left: SASRec (transformer based model) Right: TGSRec

- Other works only care about relative order/position in sequence, but TGSRec considers timestamp.
- Dynamic purchase data is changed to continuous-time bipartie graph.
- Vertex: user,item Edge: timestamp

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

01. Embedding Layer



[fig 07] **Continuous-Time Bipartite Graph**

1. **Long-Term user/Item Embeddings (Node Embedding, Temporal Embedding)**

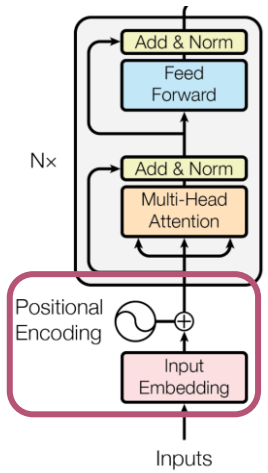
   User and items are necessary for long-term collaborative signals representation.

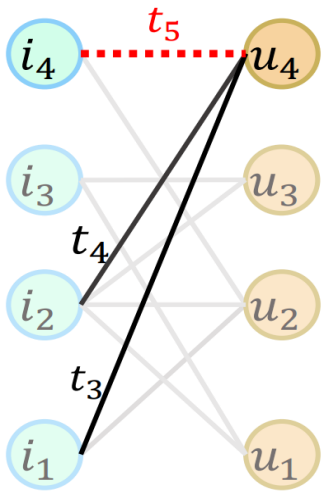
   Embedding tables $\boldsymbol{e}_u(\boldsymbol{e}_i) \in \mathbb{R}^d$ retrieved from $\boldsymbol{E} = [\boldsymbol{E}_{\mathcal{U}}; \boldsymbol{E}_{\mathcal{I}}] \in \mathbb{R}^{d \times |\mathcal{V}|}$ $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

01. Embedding Layer





2. Continuous-Time Embedding (= Timestamp Embedding)

cf) date: '2015-07-23 18:59:09' → timestamp: 1463460958000

- Time encoding function embeds timestamps into vectors.
- Time span plays a vital plays a vital component for expressing the temporal effects and sequential patterns.
- Time span representation: dot product of corresponding time embedding

Example: given a pair of interactions $(u, i, t1), (u, j, t2)$ of the same user(u)

"temporal effect" (Bochner's Theorem)

$$\psi(t_1 - t_2) = \mathcal{K}(t_1, t_2) = \Phi(t_1) \cdot \Phi(t_2)$$

$$\psi(t_1 - t_2) \mapsto \mathbb{R}$$

$$\Phi(t) \mapsto \sqrt{\frac{1}{d_T}} \left[\cos(\omega_1 t), \sin(\omega_1 t), \ldots, \cos(\omega_{d_T} t), \sin(\omega_{d_T} t)\right]^\top, \tag{2}$$

where $\boldsymbol{\omega} = \left[\omega_1, \ldots, \omega_{d_T}\right]^\top$ are learnable and d_T is the dimension.

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

02. Temporal Collaborative Transformer (TCT Layer)

- **Step 01) Information Construction**

1. Query input information

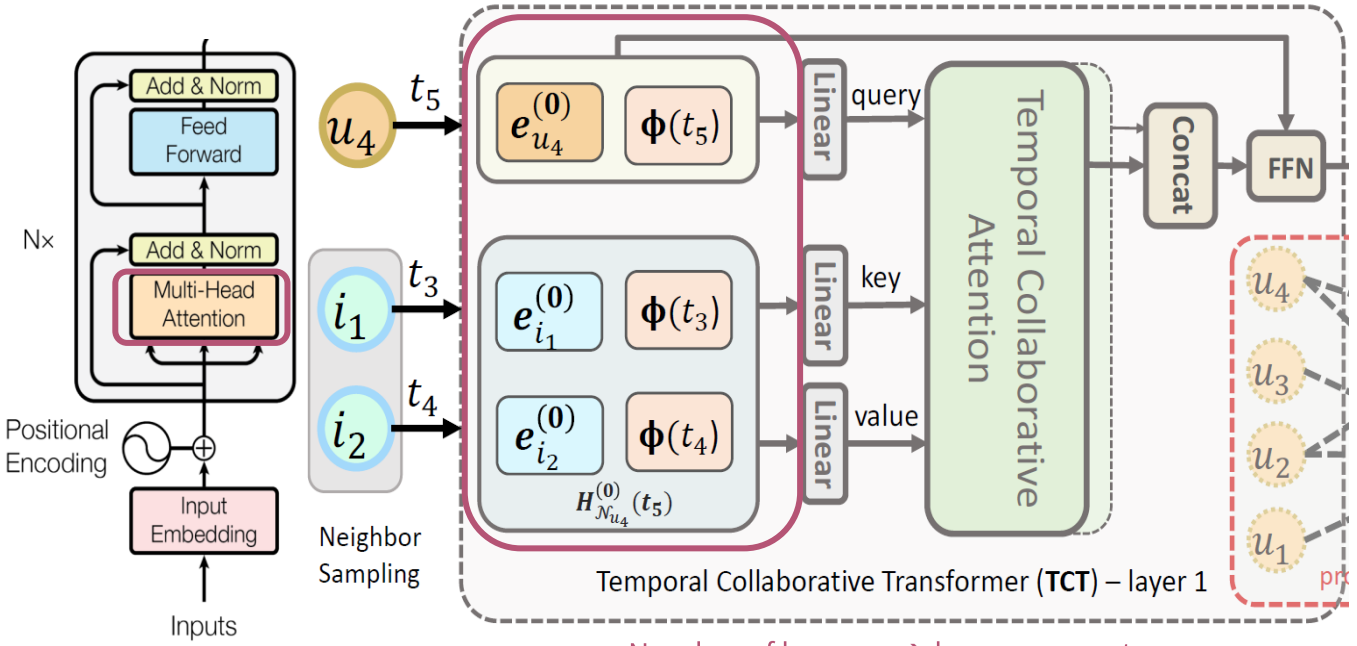
$$\boldsymbol{h}_u^{(l-1)}(t) = \boldsymbol{e}_u^{(l-1)}(t) \| \Phi(t)$$

The combination of long term node embedding and time embedding.

$\boldsymbol{h}_u^{(l-1)}(t) \in \mathbb{R}^{d+d_T}$ Output: Information for user u at t

$\boldsymbol{e}_u^{(l-1)}(t) \in \mathbb{R}^{d}$ Input 01: Temporal embedding of u

$\Phi(t) \in \mathbb{R}^{d_T}$ Input 02: Time vector of t



Number of layers⋯ → hyperparameter

2. Randomly sample S different interaction of u before time t (t: target time)

→ Then samples will be used as Key, Value.

$$\mathcal{N}_u(t) = \{(i, t_s) | (u, i, t_s) \in \mathcal{E}_t \text{ and } t_s < t\} \longrightarrow (i, t_s)$$

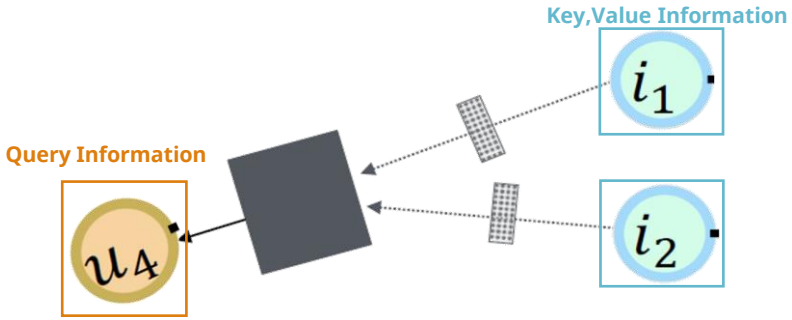
$$\boldsymbol{h}_i^{(l-1)}(t_s) = \boldsymbol{e}_i^{(l-1)}(t_s) \| \Phi(t_s)$$

$\boldsymbol{h}_i(t_s)$ Output: Information for item i at ts

$\boldsymbol{e}_i(t_s)$ Input 01: Temporal embedding of i at ts

$\Phi(t_s)$ Input 02: Time vector of ts

Information Construction step = Ready to propagation

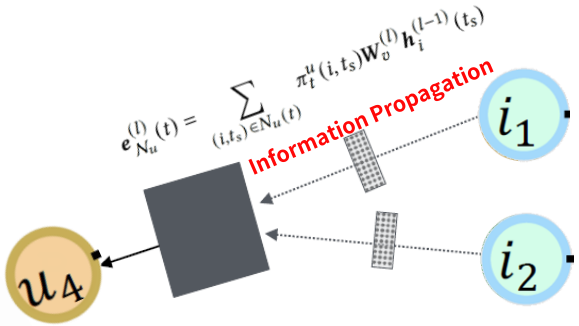


Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

02. Temporal Collaborative Transformer (TCT Layer)

- **Step 02) Information Propagation = Collaborative Signal**



$$e^{(l)}_{\mathcal{N}_u}(t) = \sum_{(i,t_s)\in N_u(t)} \pi^u_t(i,t_s) W^{(l)}_v h^{(l-1)}_i(t_s)$$

Attention $$\pi^u_t(i,t_s) = \frac{1}{\sqrt{d+d_T}} \left(W^{(l)}_k h^{(l-1)}_i(t_s)\right)^\top W^{(l)}_q h^{(l-1)}_u(t)$$

It considers both neighboring interactions and the importance of historical interaction

Normalized $$\pi^u_t(i,t_s) = \frac{\exp\left(\pi^u_t(i,t_s)\right)}{\sum_{(i',t'_s)\in N_u(t)} \exp\left(\pi^u_t(i',t'_s)\right)}$$

[Advantages for temporal collaborative attention term]

1. It considers both neighboring interactions(user) and the temporal information(timestamp) on edges. → It contribute to the importance of historical interaction.
2. Why is this formula better for the temporary collaborative signal than the self-attention method?

→ Because self-attention is to model only with item-item correlation.

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

02. Temporal Collaborative Transformer (TCT Layer)

- **Step 02) Information Propagation = Collaborative Signal**

The weight term formula can be expressed differently.

$$\pi_t^u(i,t_s) = \frac{1}{\sqrt{d+d_T}}\left(\boldsymbol{W}_k^{(l)}\boldsymbol{h}_i^{(l-1)}(t_s)\right)^\top \boldsymbol{W}_q^{(l)}\boldsymbol{h}_u^{(l-1)}(t)$$

$$\boldsymbol{h}_u^{(l-1)}(t) = \boldsymbol{e}_u^{(l-1)}(t)\|\Phi(t)$$

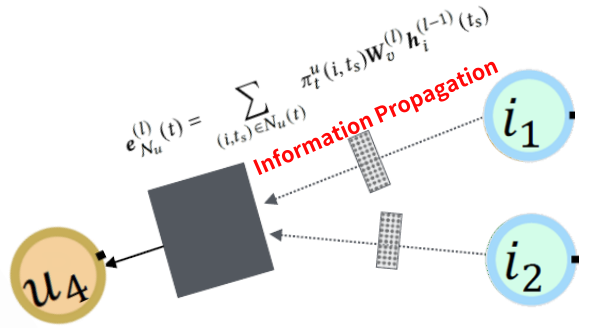
$$\boldsymbol{h}_i^{(l-1)}(t_s) = \boldsymbol{e}_i^{(l-1)}(t_s)\|\Phi(t_s)$$

$$\boldsymbol{e}_u^{(l-1)}(t)\cdot\boldsymbol{e}_i^{(l-1)}(t_s) + \Phi(t)\cdot\Phi(t_s),$$

User item collaborative signal Temporal effect

- **Step 03)Aggregation**

$$\boldsymbol{e}_u^{(l)}(t) = \mathrm{FFN}\left(\boldsymbol{e}_{\mathcal{N}_u}^{(l)}(t)\|\boldsymbol{h}_u^{(l-1)}(t)\right)$$



$$\boldsymbol{e}_{\mathcal{N}_u}^{(l)}(t) = \sum_{(i,t_s)\in N_u(t)} \pi_t^u(i,t_s)\boldsymbol{W}_v^{(l)}\boldsymbol{h}_i^{(l-1)}(t_s)$$

CF) It also can be expressed differently. → It can safely apply the multi-head attention.

Moreover, the computation is implemented by packing the information of all sampled interactions. To be more specific, we stack the information (Eq. (4)) of all sampled interactions as a matrix $\boldsymbol{H}_{\mathcal{N}_u}^{(l-1)}(t) \in \mathbb{R}^{(d+d_T)\times S}$, as illustrated[2] in Figure 3. We denote $\boldsymbol{K}_u^{(l-1)}(t) = \boldsymbol{W}_k^{(l)}\boldsymbol{H}_{\mathcal{N}_u}^{(l-1)}(t)$, $\boldsymbol{V}_u^{(l-1)}(t) = \boldsymbol{W}_v^{(l)}\boldsymbol{H}_{\mathcal{N}_u}^{(l-1)}(t)$ and $\boldsymbol{q}_u^{(l-1)}(t) = \boldsymbol{W}_q^{(l)}\boldsymbol{h}_u^{(l-1)}(t)$, which are respectively the key, value and query input for the temporal collaborative attention module. We illustrate this in Figure 3 as green blocks. For simplicity and without ambiguity, we ignore the superscripts and time t and combine Eq. (6) and Eq. (8). Then, we can rewrite the Eq. (5) as:

$$\boldsymbol{e}_{\mathcal{N}_u} = \boldsymbol{V}_u\cdot\mathrm{Softmax}\left(\frac{\boldsymbol{K}_u^\top\boldsymbol{q}_u}{\sqrt{d+d_T}}\right), \quad (9)$$

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec

03. Model Prediction and Optimization

- Prediction Layer - *Input:* → final TCT layer embedding $e_u^{(L)}(t) e_i^{(L)}(t)$ ***Output*:** calculate the scores

$$r(u, i, t) = \boldsymbol{e}_u^{(L)}(t) \cdot \boldsymbol{e}_i^{(L)}(t),$$

- Optimization

BPR Loss $\mathcal{L}_{bpr} = \sum_{(u,i,j,t) \in O_T} -\log \sigma \left(r(u, i, t) - r(u, j, t) \right) + \lambda ||\Theta||_2^2$

$O_T = \{(u, i, j, t) | (u, i, t) \in \mathcal{E}_T, j \in \mathcal{I} \setminus \mathcal{I}_u(t)\}$, where the positive interaction (u, i, t) comes from the edge set $\mathcal{E}_T$ of CTBG, the negative item j is sampled from unobserved items $\mathcal{I} \setminus \mathcal{I}_u(t)$ of user u at timestamp t;

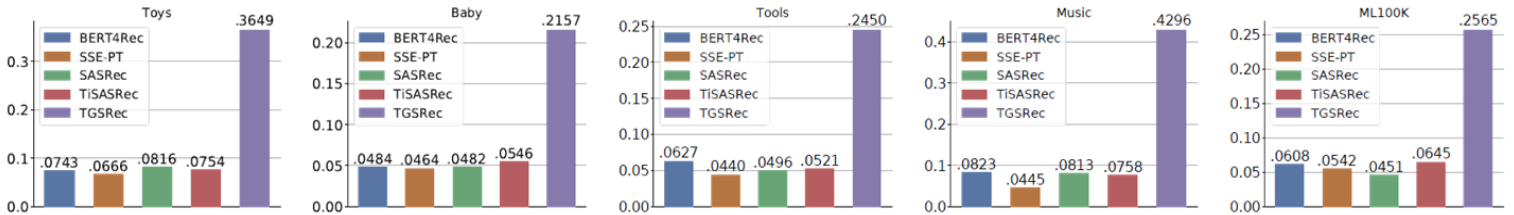
Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

TGSRec
Experiments

Table 2: Overall Performance w.r.t. Recall@{10,20} and MRR.

Transformer based model

Datasets	Metric	BPR	LightGCN	SR-GNN	GRU4Rec	Caser	SSE-PT	BERT4Rec	SASRec	TiSASRec	CDTNE	TGSRec	Improv.
Toys	Recall@10	0.0021	0.0016	0.0020	0.0274	0.0138	0.1213	0.1273	0.1452	0.1361	0.0016	**0.3650**	0.2198
	Recall@20	0.0036	0.0026	0.0033	0.0288	0.0238	0.1719	0.1865	0.2044	0.1931	0.0045	**0.3714**	0.1670
	MRR	0.0024	0.0018	0.0018	0.0277	0.0082	0.0595	0.0643	0.0732	0.0671	0.0025	**0.3661**	0.2929
Baby	Recall@10	0.0028	0.0036	0.0030	0.0036	0.0077	0.0911	0.0884	0.0975	0.1040	0.0218	**0.2235**	0.1195
	Recall@20	0.0039	0.0045	0.0062	0.0048	0.0193	0.1418	0.1634	0.1610	0.1662	0.0292	**0.2295**	0.0663
	MRR	0.0019	0.0024	0.0024	0.0028	0.0071	0.0434	0.0511	0.0455	0.0521	0.0157	**0.2147**	0.1626
Tools	Recall@10	0.0023	0.0021	0.0051	0.0048	0.0077	0.0775	0.1296	0.0913	0.0946	0.0186	**0.2457**	0.1161
	Recall@20	0.0036	0.0035	0.0092	0.0059	0.0161	0.1155	0.1784	0.1337	0.1356	0.0271	**0.2559**	0.0775
	MRR	0.0026	0.0023	0.0028	0.0051	0.0068	0.0419	0.0628	0.0460	0.0480	0.0203	**0.2468**	0.1840
Music	Recall@10	0.0122	0.0142	0.0051	0.0549	0.0183	0.0915	0.1352	0.1372	0.1372	0.0071	**0.5935**	0.4563
	Recall@20	0.0152	0.0183	0.0092	0.0589	0.0346	0.1494	0.2093	0.2094	0.1951	0.0163	**0.5986**	0.3892
	MRR	0.0057	0.0064	0.0028	0.0540	0.0106	0.0423	0.0824	0.0768	0.0681	0.0037	**0.3820**	0.2996
ML100k	Recall@10	0.0461	0.0565	0.0045	0.0996	0.0246	0.1079	0.1116	0.09450	0.1332	0.0350	**0.3118**	0.1786
	Recall@20	0.0766	0.0960	0.0060	0.1168	0.0417	0.1801	0.1786	0.1808	0.2232	0.0536	**0.3252**	0.1020
	MRR	0.0213	0.0252	0.0012	0.0938	0.0147	0.0519	0.0600	0.0448	0.0605	0.0162	**0.2416**	0.1478



(a) NDCG@10 in Toys (b) NDCG@10 in Baby (c) NDCG@10 in Tools (d) NDCG@10 in Music (e) NDCG@10 in ML100K

Figure 4: NDCG@10 Performance in all Datasets. We ignore other methods because of their low values.

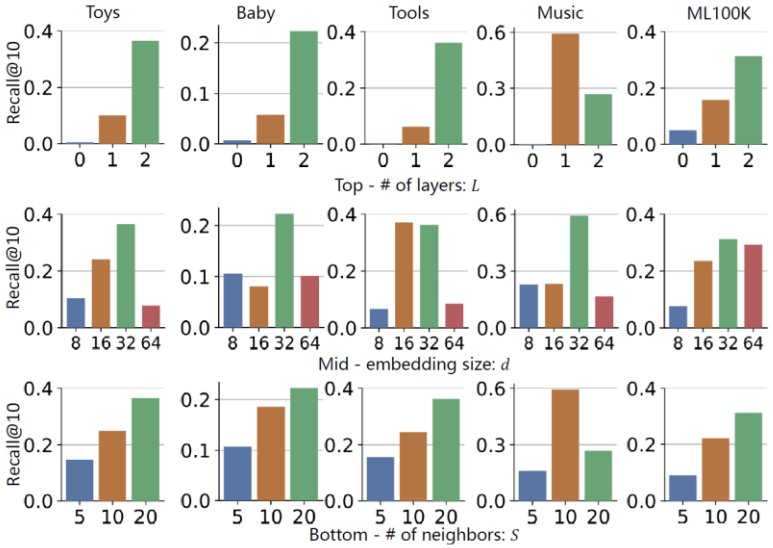


Figure 5: Recall@10 w.r.t. L, d and S on 5 datasets.

Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.

Conclusion

- **TGSRec**

  Idea 01: It apply a temporal kernel to map continuous timestamps on edges to vectors.

  Idea 02: It introduce TCT layer, which can infer temporal embedding of nodes.

  → It samples neighbors and learns attention weight to aggregate both node embedding and time vector.

  - Encode sequential pattern and collaborative signals as well as reveal the temporal effects.

  - TGSRec significantly outperforms existing transformer-based sequential recommendation models.

  - It solve the temporal information with sequential recommendation.

- **DGSR**

  - DGSR realizes the explicit encoding of the dynamic collaborative information among different user sequences.

References

[1] Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014.

[2] Wu, Shiwen, et al. "Graph neural networks in recommender systems: a survey." *ACM Computing Surveys (CSUR)* (2020).

[3] Wang, Shoujin, et al. "Graph learning based recommender systems: A review." *arXiv preprint arXiv:2105.06339* (2021).

[4] Berg, Rianne van den, Thomas N. Kipf, and Max Welling. "Graph convolutional matrix completion." *arXiv preprint arXiv:1706.02263* (2017).

[5] Li, Yujia, et al. "Gated graph sequence neural networks." *arXiv preprint arXiv:1511.05493* (2015).

[6] Sun, Fei, et al. "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer." *Proceedings of the 28th ACM international conference on information and knowledge management*. 2019.

[7] Zhang, Mengqi, et al. "Dynamic graph neural networks for sequential recommendation." *IEEE Transactions on Knowledge and Data Engineering* (2022).

[8] Fan, Ziwei, et al. "Continuous-time sequential recommendation with temporal graph collaborative transformer." *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021.