Handling Information Loss of Graph Neural Networks for Session-based Recommendation

Tianwen Chen
tchenaj@cse.ust.hk
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

Raymond Chi-Wing Wong
raywong@cse.ust.hk
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

MOTIVATION

- *Lossy session encoding problem*

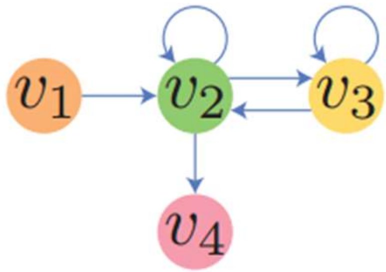

Figure 1: Two different sessions $[v_1, v_2, v_3, v_3, v_2, v_2, v_4]$ **and** $[v_1, v_2, v_2, v_3, v_3, v_2, v_4]$ **are converted to the same graph.**

- *Ineffective long-rang dependency capturing problem*

For GNN model, each layer can capture only 1-hop relation, the optimal number of layers for the GNN models is usually no longer than 3, due to overfitting and over-smoothing problems. Thus, the model can only capture up to 3-hop relation.

S2MG

- *S2MG:Session to EOP (edge-order preserving) Multigraph*

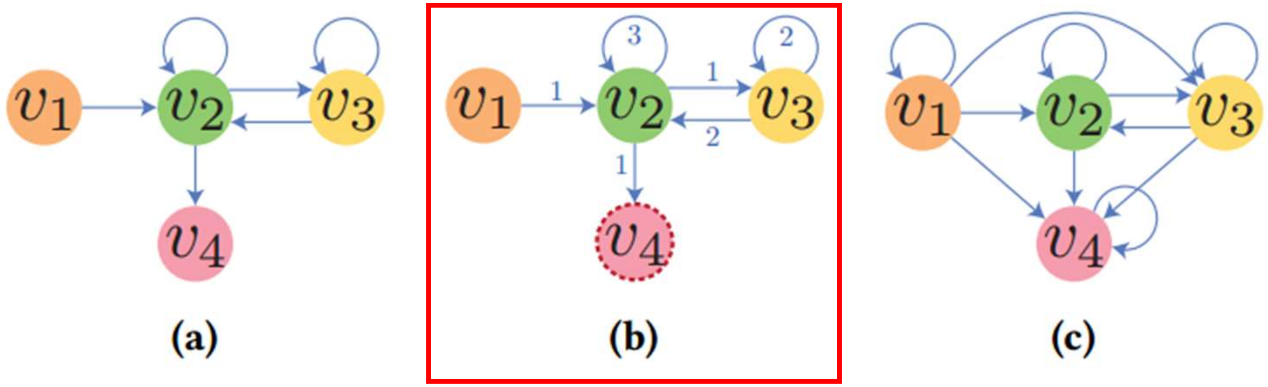

Figure 3: The weighted graph (a), EOP multigraph (b) and shortcut graph (c) of session $[v_1, v_2, v_3, v_3, v_2, v_2, v_4]$ converted by S2WG, S2MG and S2SG, respectively. Note that the weights in (a) are omitted because they are all 1.

For each node v, the edges in $E_{in}(v)$ can be ordered by the time of their occurrences in the session. If there are multiple transitions from u to v, multiple edges from u to v will be created.

S2SG

- *S2MG:Session to Shortcut Graph*

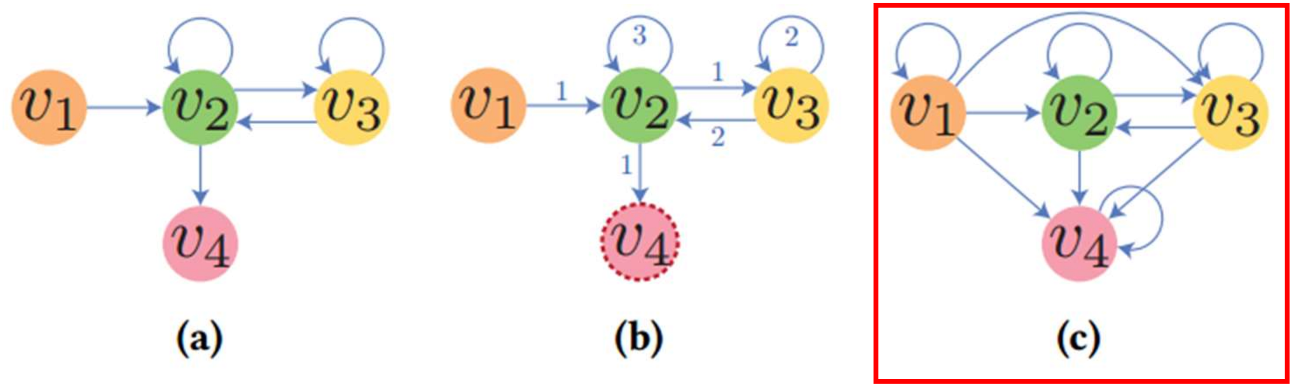

Figure 3: The weighted graph (a), EOP multigraph (b) and shortcut graph (c) of session $[v_1, v_2, v_3, v_3, v_2, v_2, v_4]$ converted by S2WG, S2MG and S2SG, respectively. Note that the weights in (a) are omitted because they are all 1.

For each order pair of nodes (u, v), an edge from u to v is created if and only if there exists a pair (s_{i,t_1}, s_{i,t_2}), such that $s_{i,t_1} = u, s_{i,t_2} = v$ and $t_1 < t_2$. Besides, self-loops are also added to the graph for information aggregation, which is a common practice in GAT models.

LESSR

- *LESSR: Lossless Edge-order preserving aggregation and Shortcut graph attention for Session-based Recommendation*

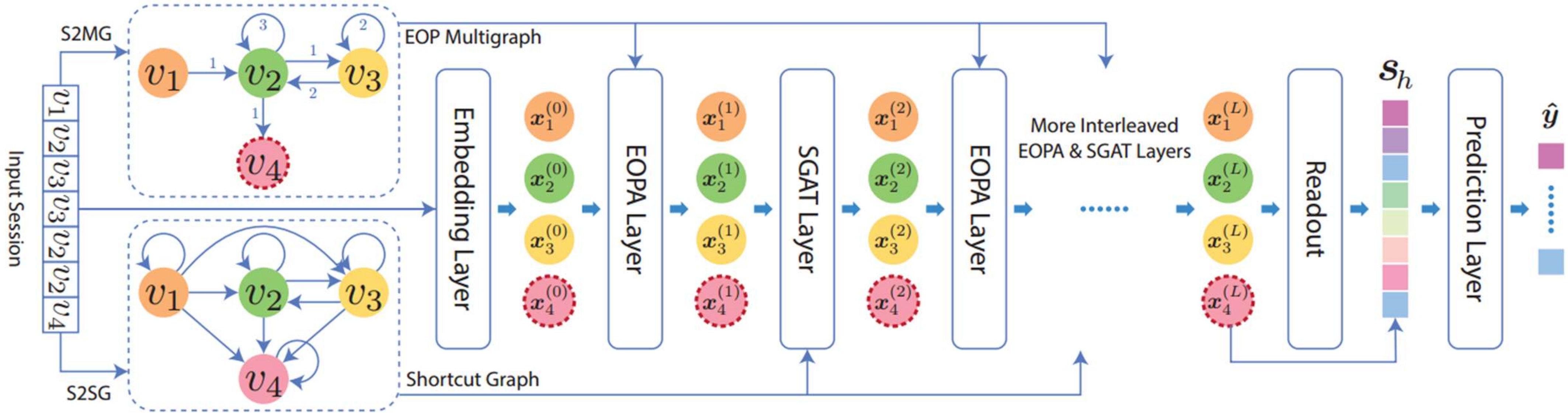

Figure 2: The overview of the proposed model LESSR. Given a session, an edge-order preserving (EOP) multigraph and a shortcut graph is computed. The initial node representations $x_i^{(0)}$ are the item embeddings. The graphs and the node representations are passed as input to multiple interleaved EOPA and SGAT layers. Each layer outputs the new node representations. The readout layer computes a graph-level representation, which is combined with the recent interests to form the session embedding s_h. Finally, the prediction layer computes the probability distribution of the next item $\hat{y}$.

LESSR-EOPA (Edge-Order-Preserving Aggregation) Layer

- *Information update*

$$x_i^{(l+1)} = \boldsymbol{W}_{upd}^{(l)}(x_i^{(l)}||h_{d_i}^{(l)})$$

- *Message aggregation*

$$h_k^{(l)} = GRU^{(l)}(\boldsymbol{W}_{msg}^{(l)}x_{jk}^{(l)}, h_{k-1}^{(l)})$$

where x_i is the embedding of node i, $\{h_k^{(l)}: 0 \leq k \leq d_i\}$ are the hidden states of GRU, and d_i is the integer attributes of the edges for node i. $\boldsymbol{W}_{msg} \in \mathbb{R}^{d\times d}$, $\boldsymbol{W}_{upd} \in \mathbb{R}^{d\times 2d}$ are learnable parameters.

LESSR-SGAT (Shortcut Graph Attention) Layer

- *Information update*

$$x_i^{(l+1)} = \sum_{(j,i)\in E_{in(i)}} \alpha_{ij}^{(l)} \boldsymbol{W}_{val}^{(l)} x_j^{(l)}$$

$$\alpha_{ij}^{(l)} = softmax\left(e_{ij}^{(l)}\right)$$

$$e_{ij}^{(l)} = \left(\boldsymbol{p}^{(l)}\right)^T \sigma(\boldsymbol{W}_{key}^{(l)} x_i^{(l)} + \boldsymbol{W}_{qry}^{(l)} x_j^{(l)} + \boldsymbol{b}^{(l)})$$

where $E_{in(i)}$ is the set of incoming edges of node i in shortcut graph. $\boldsymbol{p}^{(l)}, \boldsymbol{b}^{(l)} \in \mathbb{R}^d$ and $\boldsymbol{W}_{key}^{(l)}, \boldsymbol{W}_{qry}^{(l)}, \boldsymbol{W}_{val}^{(l)} \in \mathbb{R}^{d\times d}$ are learnable parameters.

LESSR-Session Embedding

- *Graph-level representation*

$$\boldsymbol{h}_G = \sum_{i \in V} \beta_i x_i^{(L)}$$

$$\beta_i = softmax(\epsilon_i)$$

$$\epsilon_i = \boldsymbol{q}^T \sigma(\boldsymbol{W_1} x_i^{(L)} + \boldsymbol{W_2} x_{last}^{(L)} + \boldsymbol{r})$$

where $x_{last}^{(L)}$ is the final layer (L) node representation of the last item in the session. $\boldsymbol{q}, \boldsymbol{r} \in \mathbb{R}^d$ and $\boldsymbol{W_1}, \boldsymbol{W_2} \in \mathbb{R}^{d \times d}$ are learnable parameters.

- *Local-level representation*

$$\boldsymbol{s_l} = x_{last}^{(L)}$$

- *Session Embedding*

$$\boldsymbol{s_h} = \boldsymbol{W_h}(\boldsymbol{s_g} || \boldsymbol{s_l})$$

where $\boldsymbol{s_g} = \boldsymbol{h}_G$, and $\boldsymbol{W_h} \in \mathbb{R}^{d \times 2d}$ is learnable parameters.

LESSR-Prediction Layer

- *Predicted probability of the next item*

$$z_i = s_h^T v_i$$
$$\hat{y}_i = softmax(z_i)$$

where $\hat{y}_i$ is the predicted probability of the next item i.

- *Loss function*

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\boldsymbol{y}^{\boldsymbol{T}} log \hat{\boldsymbol{y}}$$

EXPERIMENTS

Table 2: Experimental results (%) on three datasets

Method	Diginetica		Gowalla		Last.fm	
	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20
Item-KNN	39.51	11.22	38.60	16.66	14.90	4.04
FPMC	28.50	7.67	29.91	11.45	12.86	3.78
NextItNet	45.41	15.19	45.15	21.26	20.12	7.08
NARM	49.80	16.57	50.07	23.92	21.83	7.59
FGNN	50.03	17.01	50.06	24.12	22.20	8.02
SR-GNN	50.81	17.31	50.32	24.25	22.33	8.23
GC-SAN	50.90	17.63	50.68	24.67	22.64	8.42
LESSR	**51.71**	**18.15**	**51.34**	**25.49**	**23.37**	**9.01**
Improv.	1.59%	2.95%	1.30%	3.32%	3.22%	7.01%

EXPERIMENTS

Table 3: The performance of different message passing layers

Layer(s)	Diginetica		Gowalla		Last.fm	
	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20
WGAT	49.71	16.46	50.03	24.02	21.89	7.89
GGNN	49.85	16.59	50.24	24.23	22.08	8.02
EOPA (rand)	49.81	16.56	50.18	24.11	22.05	8.06
EOPA	**50.30**	**16.93**	**50.86**	**24.89**	**22.31**	**8.36**
GGNN+SAN	50.06	16.72	50.37	24.44	22.22	8.18
GGNN+EOPA	**50.28**	**16.91**	**50.76**	**24.96**	**22.41**	**8.40**