ML in PL CONFERENCE 2022

Samsung Research America

Harnessing the power of graph structures to understand changing user preferences over time and enhance item recommendations

Date

5th November 2022

Organization

Visual Display Intelligence Lab | AI team

SAMSUNG

Agenda:

- Introduction
- Background
  - Types of Recommender Systems
- Why graphs for Recommender systems?
- Session based Recommender Systems
  - SR-GNN approach
- Session aware Recommender Systems
  - A-PGNN approach
- Our methodology
- Experimental setup
- Results
- Conclusion

Introduction:

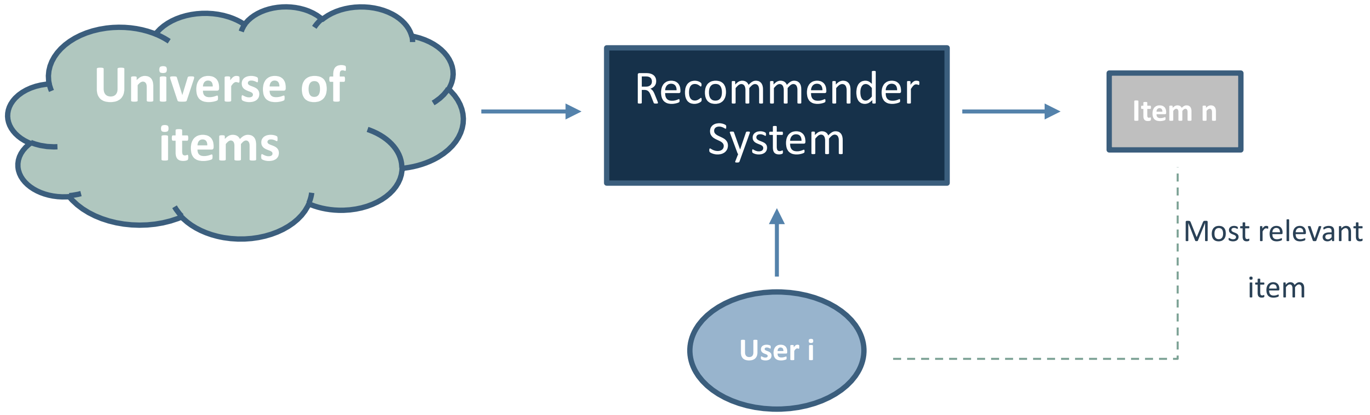
Tomasz Palczewski
Samsung Research America
tomasz.p@samsung.com

Anirudh Rao
Samsung Research America
a.rao2@samsung.com

Background:

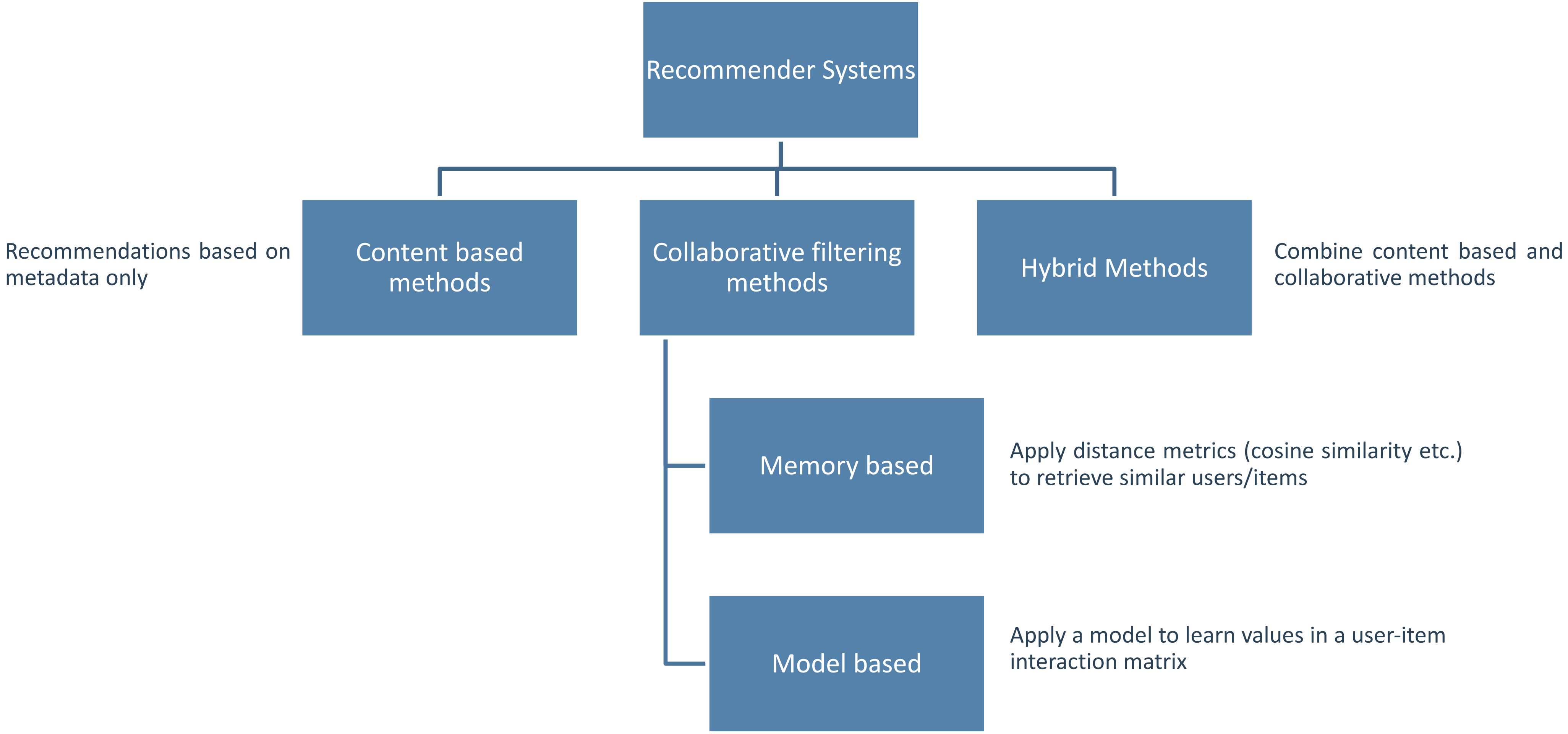
Why use a recommender system?

- Filter information to most relevant
- Avoids information overload
- Enhanced user experience
  - Customer retention
  - Encourages exploration
- Boost revenue



ML in PL CONFERENCE 2022

Types of Recommender Systems:



ML in PL CONFERENCE 2022

Content based Filtering

- Recommends items that are similar to the items the user has already liked in the past
- Similarity of items based on discrete features,
  - Item descriptions
  - Genre
- **Pros:**
  - Needs less information about the users
  - Relies solely on item information
- **Cons:**
  - Recommended items are too similar to past items
  - Less exploration of the item catalogue
  - Item features not always easily accessible

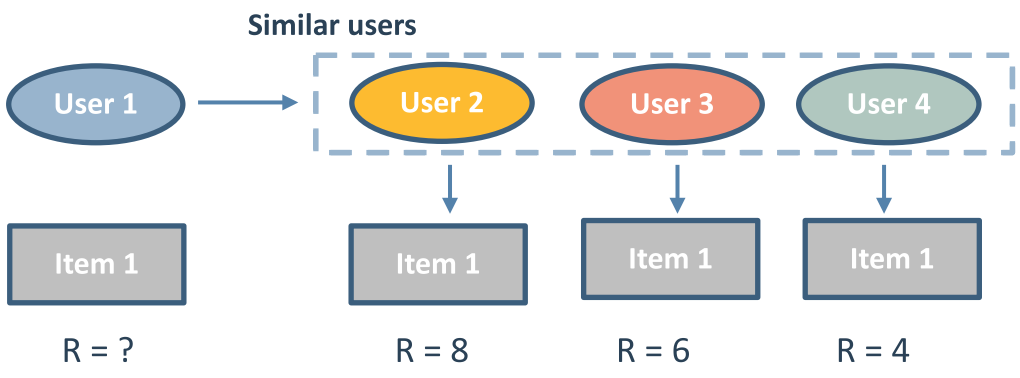
Collaborative filtering

- Recommends items based on the tastes of similar users
- Compares user activity based on:
  - Explicit feedback (ratings, like/dislike etc.)
  - Implicit feedback (purchased item, item view count etc.)
- **Pros:**
  - Needs less explicit information about items
  - More variety in the recommendations
- **Cons:**
  - Cold Start problem
  - Data Sparsity

ML in PL CONFERENCE 2022

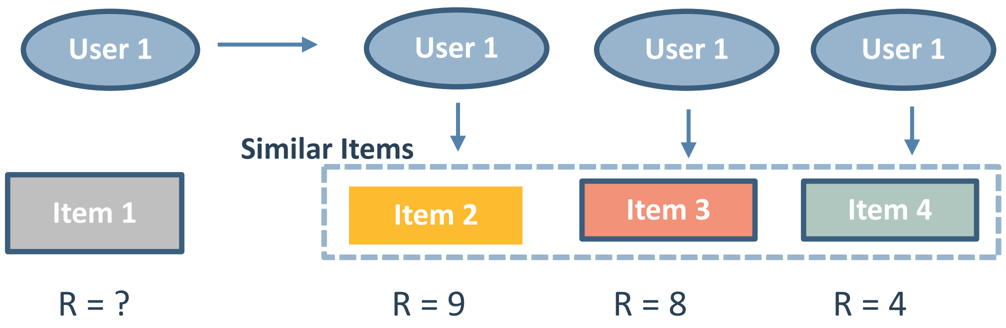
Memory based Collaborative filtering:

User based:



R(User 1, Item 1) = $\frac{8+6+4}{3}$ = 6

Item based:



R(User 1, Item 1) = $\frac{9+8+4}{3}$ = 7

Drawbacks:

- Missing context, since only user-item interactions considered
- Evolving tastes of the users are not captured effectively
- Not scalable since finding similar users/items is a costly operation
- Cold start problem

ML in PL CONFERENCE 2022

Model based Collaborative filtering: Matrix Factorization

Item

User/Item	A	B	C	D
Ted		4		3
Carol	3		2	
Bob		5		2
Alice			4	

User

→

Ted	1.4	.9
Carol	1.2	1
Bob	1.5	.9
Alice	1.2	.8

User

*

A	B	C	D
1.4	1.3	.9	1.2
.8	1.1	2	.8

Item

Drawbacks:

- Missing context
- No Temporal information
- Matrix sparsity
- Cold start problem

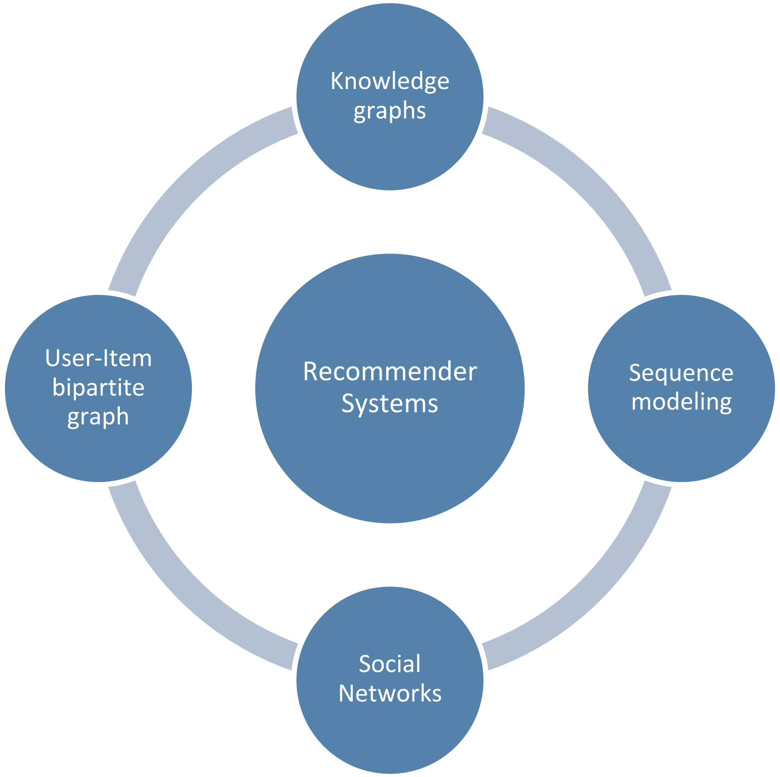
$$R_{n \times m} = P_{n \times f} Q_{f \times m}$$

- R is the matrix of ratings
  - n users, m items
- P is a user-factor matrix
- Q is a factor-item matrix

Here, the assumption is that we can factorize the item-user matrix into two separate matrices, one for items and another for users. The model identifies the missing values in the matrix using methods like Gradient descent, SGD etc.

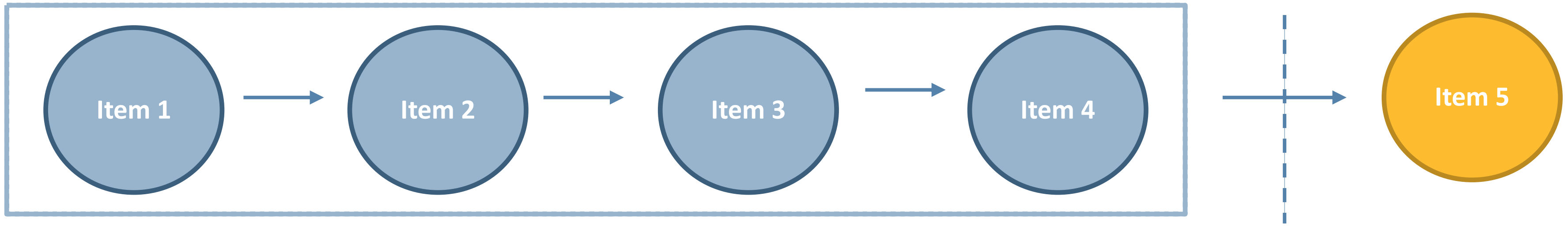
Why graphs for Recommender systems?

- Trivial methods only take user-item interactions
- Abundance of data in recommendation systems
- Graphs can implicitly learn collaborative signals in the data
- Powerful tool to capture multi hop relationships between entities
- Graph structure flexibility :
    - ➢ Direction of the edges, could be directional or non-directional
    - ➢ We can define edge weights considering multiple factors, if needed
    - ➢ The nodes can have their own subgraphs with separate attributes



Session based recommender systems:

- Only look at the current session interactions and recommend the next item in the current session
- Especially useful when user identity is anonymized
    - For example. a user who is just casually browsing items on a website without logging in
- Only try to model the user’s current intent without considering long-term user interest
- Extract item embeddings using the temporal information of user interactions
- From the extracted item embeddings, we learn a more accurate representation of the sessions

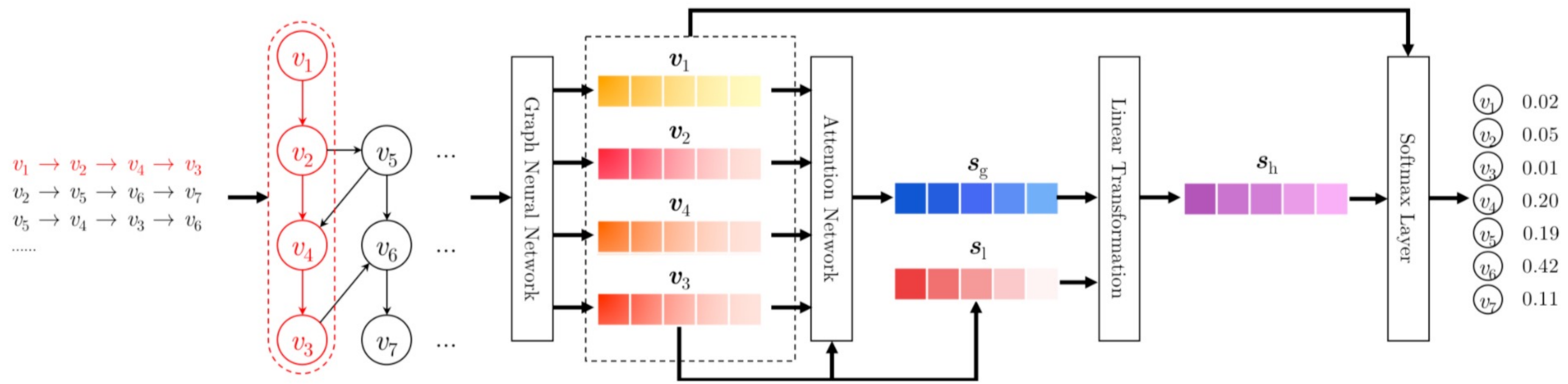


Historical items in the current session

Item to be recommended

The SR-GNN approach [2]:

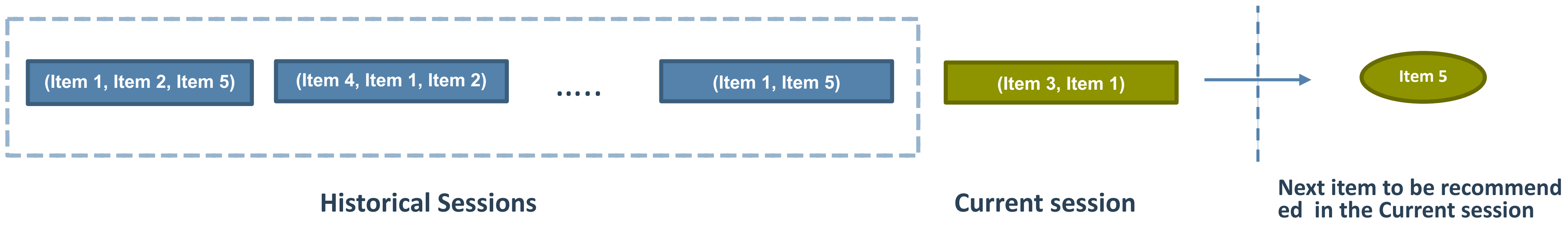
- Represent session data as graphs
- Use item embeddings to represent the sessions
- Use a Graph neural network to extract the item (node) embeddings with temporal information
- Derive session embeddings from the learned item embeddings
- Concatenate the last item's embedding (local embedding) with global session embeddings (global embedding)
- Apply a softmax to get probability for each item using the hybrid embedding



[2] *Session-based Recommendation with Graph Neural Networks:* https://arxiv.org/abs/1811.00855

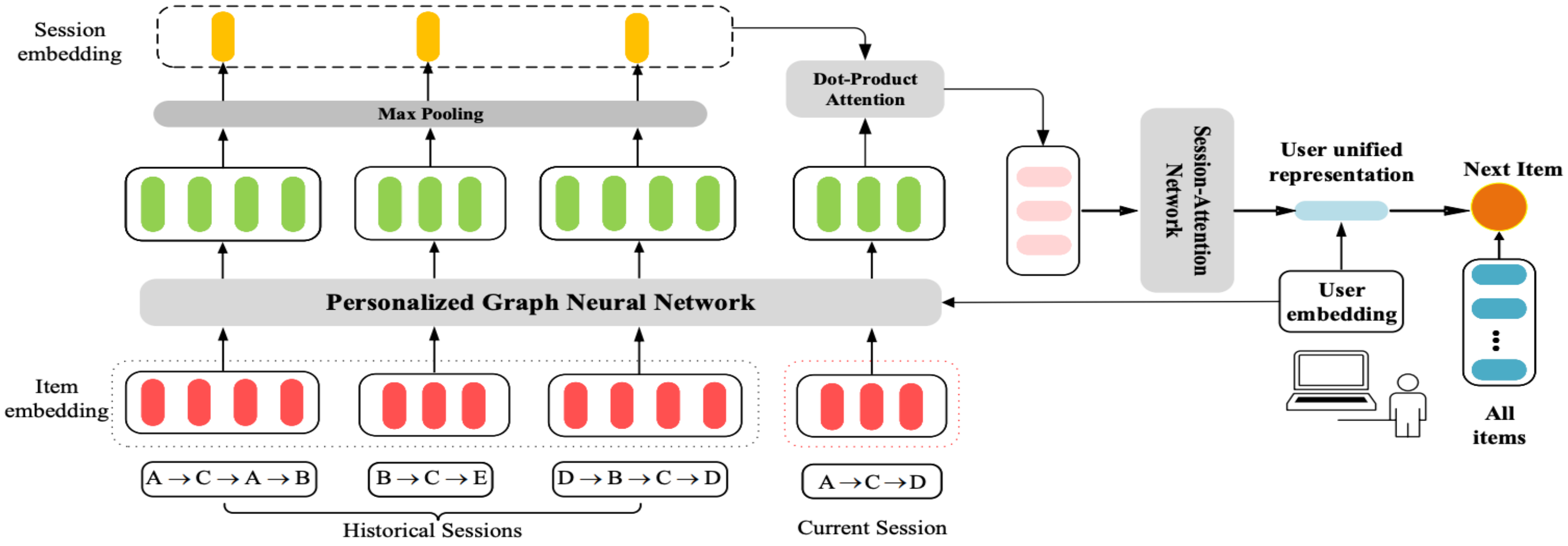
Session aware recommender systems:

- Look at the current session interactions **along with** historical session interactions and recommend the next item
- In this case, we know the user's identity
    - For example. a user who is logged in and is browsing items on a website
- This approach takes long term user preferences into account
- This is a similar approach to session based
- Clear separation of historical and current sessions to explicitly factor in evolving user interest
- We can visualize a session aware recommender scenario as below:



The A-PGNN approach [3]:

- The structure of the model is inspired by SR-GNN
- Again, uses graph structure to represent session data
- The GNN layer is personalized with user embedding as input
- Attention layer to capture the dependency between historical sessions and current session
- Session embeddings are enriched with user embeddings and external information to recommend next item



[3] *Personalized Graph Neural Networks with Attention Mechanism for Session-Aware Recommendation:* https://arxiv.org/abs/1910.08887

Model details:

SR-GNN	A-PGNN
	$\mathbf{a}^{(t)}_{\mathrm{out}_i} = \sum_{v_i \to v_j} \mathbf{A}^{\mathrm{out}}_u[i,j] \left[\mathbf{h}^{(t-1)}_j \parallel \mathbf{e}_u\right] \mathbf{W}_{\mathrm{out}}$
	$\mathbf{a}^{(t)}_{\mathrm{in}_i} = \sum_{v_j \to v_i} \mathbf{A}^{\mathrm{in}}_u[i,j] \left[\mathbf{h}^{(t-1)}_j \parallel \mathbf{e}_u\right] \mathbf{W}_{\mathrm{in}},$
$\mathbf{a}^t_{s,i} = \mathbf{A}_{s,i:} \left[\mathbf{v}^{t-1}_1, \ldots, \mathbf{v}^{t-1}_n\right]^\top \mathbf{H} + \mathbf{b},$	$\mathbf{a}^{(t)}_i = \mathbf{a}^{(t)}_{\mathrm{out}_i} \parallel \mathbf{a}^{(t)}_{\mathrm{in}_i},$
$\mathbf{z}^t_{s,i} = \sigma\left(\mathbf{W}_z \mathbf{a}^t_{s,i} + \mathbf{U}_z \mathbf{v}^{t-1}_i\right),$	$\mathbf{z}^{(t)}_i = \sigma\left(\mathbf{W}_z \mathbf{a}^{(t)}_i + \mathbf{U}_z \mathbf{h}^{(t-1)}_i\right),$
$\mathbf{r}^t_{s,i} = \sigma\left(\mathbf{W}_r \mathbf{a}^t_{s,i} + \mathbf{U}_r \mathbf{v}^{t-1}_i\right),$	$\mathbf{r}^{(t)}_i = \sigma\left(\mathbf{W}_r \mathbf{a}^{(t)}_i + \mathbf{U}_r \mathbf{h}^{(t-1)}_i\right),$
$\widetilde{\mathbf{v}^t_i} = \tanh\left(\mathbf{W}_o \mathbf{a}^t_{s,i} + \mathbf{U}_o \left(\mathbf{r}^t_{s,i} \odot \mathbf{v}^{t-1}_i\right)\right)$	$\widetilde{\mathbf{h}^{(t)}_i} = \tanh\left(\mathbf{W}_o \mathbf{a}^{(t)}_i + \mathbf{U}_o \left(\mathbf{r}^{(t)}_i \odot \mathbf{h}^{(t-1)}_i\right)\right),$
$\mathbf{v}^t_i = \left(1 - \mathbf{z}^t_{s,i}\right) \odot \mathbf{v}^{t-1}_i + \mathbf{z}^t_{s,i} \odot \widetilde{\mathbf{v}^t_i},$	$\mathbf{h}^{(t)}_i = \left(1 - \mathbf{z}^{(t)}_i\right) \odot \mathbf{h}^{(t-1)}_i + \mathbf{z}^{(t)}_i \odot \widetilde{\mathbf{h}^{(t)}_i},$
$\alpha_i = \mathbf{q}^\top \sigma(\mathbf{W}_1 \mathbf{v}_n + \mathbf{W}_2 \mathbf{v}_i + \mathbf{c}),$	$\mathbf{H}_h = \mathrm{Attention}\left(\mathbf{Q}^u, \mathbf{K}^u, \mathbf{V}^u\right) \quad \mathbf{H}^{u'} = \mathbf{H}_h + \mathbf{H}^u$
$\mathbf{s}_g = \sum_{i=1}^{n} \alpha_i \mathbf{v}_i,$	$\mathbf{z}_g = \sum_{i=1}^{m} \alpha_i \mathbf{h}'_i,$
$\mathbf{s}_h = \mathbf{W}_3 \left[\mathbf{s}_l; \mathbf{s}_g\right],$	$\mathbf{z}_u = \mathbf{B}\left[\mathbf{z}_d \parallel \mathbf{e}_u\right],$ where, $\mathbf{z}_d = \mathbf{z}_g \parallel \mathbf{z}_l.$
$\hat{\mathbf{z}}_i = \mathbf{s}_h^\top \mathbf{v}_i$	$\hat{\mathbf{z}}_i = \mathbf{z}_u{}^\top \mathbf{e}_{v_i},$

$$\hat{\mathbf{y}} = \mathrm{softmax}\left(\hat{\mathbf{z}}\right),$$

- A-PGNN adds user embedding to the c urrent representation of node.
- GRU to include information from othe r nodes with hidden states of the previ ous timestamp.
- SR-GNN: soft attention mechanism; Fi nal hybrid embedding from linear tran sformation over the concatenation of l ocal and global embeddings.
- A-PGNN: transformer's attention mec hanism; Final unified representation t hrough a linear transformation.

Our methodology:

Can we combine session based and session aware approaches together?

- Extract session embeddings from both SR-GNN and A-PGNN
- SR-GNN embeddings are without user information
- A-PGNN embeddings are enriched with user information
- Combine embeddings based for overlapping time windows
- Include external information about users and items
- Combine external information with session embeddings
- Dense layers consume this enriched feature set to recommend the final item

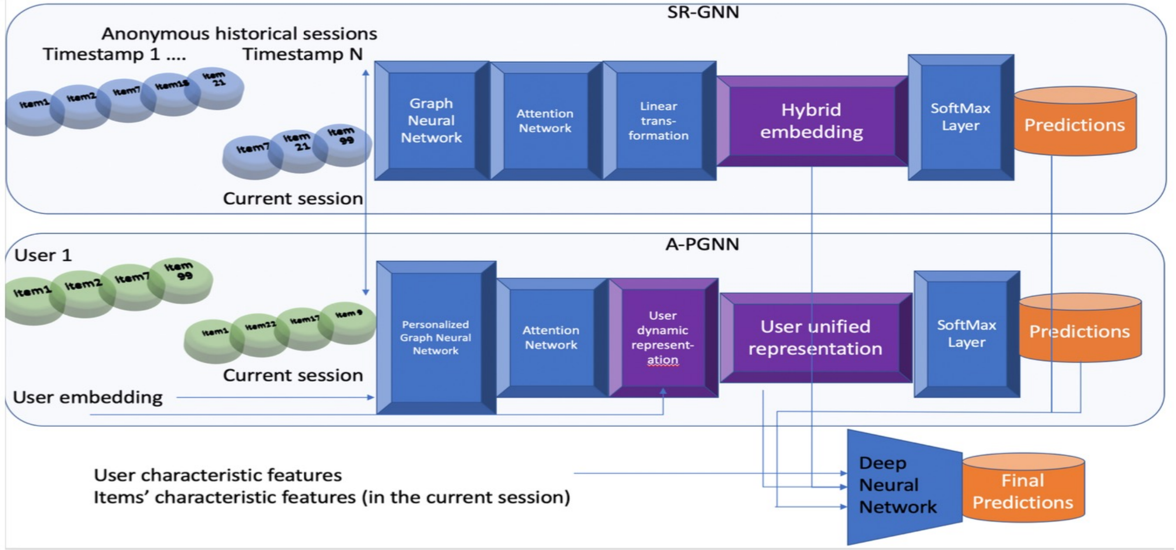


Figure 4: SRA-NN-Rec model architecture

Experimental setup:

- Experiments on two publicly available datasets: Xing and Reddit
- Only sessions with at least 5 interactions are considered
- The interactions within each session are sorted by timestamp
- Once we have the sorted interactions, we use a predefined window to create sub sessions
- We use an 80:10:10 split to divide the transformed sessionized data into train, dev and test sets for model tuning

Attribute	Xing	Reddit
# of users	11,479	18,271
# of items	59,121	27,452
Window length	30 mins	60 mins
Total sessions	91,683	1,135,488

Results:

- Metric used to evaluate model performance: **Recall@K**

R = (# of top k recommendations that are relevant)/(# of all relevant items)

- K values used: **5, 10, 20**
- The values for the SRA-NN-Rec model are the average values after 5 runs

Table 1: Recall@K on Xing dataset

Model	Recall@5	Recall@10	Recall@20
Pop	0.21	0.26	0.58
Item-KNN	8.79	11.85	14.67
ALS	8.48	9.68	9.68
NCF-MF	9.25	10.67	11.33
NGCF	10.11	12.56	14.79
FPMC	1.70	2.42	3.27
SKNN	14.36	19.42	24.12
VSKNN	14.46	**19.60**	24.25
GRU4Rec	10.35	13.15	15.30
SR-GNN	3.38	16.71	19.25
H-RNN	10.74	14.36	17.64
HierTCN	13.57	16.55	19.93
A-PGNN	14.38	17.06	19.98
SRA-NN-Rec	**14.66**	19.07	**24.76**

Table 2: Recall@K on Reddit dataset

Model	Recall@5	Recall@10	Recall@20
Pop	13.22	19.46	26.47
Item-KNN	21.71	30.32	38.85
ALS	10.36	13.18	15.00
NCF-MF	9.99	13.17	15.25
NGCF	27.55	35.01	39.43
FPMC	29.91	34.31	44.32
SKNN	34.29	42.17	49.68
VSKNN	34.25	42.17	49.67
GRU4Rec	33.72	41.73	50.04
SR-GNN	34.96	42.38	50.33
H-RNN	44.76	53.44	61.80
HierTCN	47.15	55.37	63.96
A-PGNN	**49.19**	59.43	68.00
SRA-NN-Rec	48.99	**61.37**	**69.77**

Conclusion:

- Temporal information is critical to recommend relevant items to users
- Simple collaborative/content-based filtering approaches have several limitations
- Graph structures are powerful way to represent sequential data
- We explore a Graph based approach to model a recommender system
- Experiments on two publicly available datasets showcases the similar performance of our model

Thank you!