Temporal Dynamics-Driven Session-Based Recommendation System with Graph Neural Networks

Teja Sree Parasa (tfp5349@psu.edu)

I. Abstract

Session-based recommendation systems face unique challenges due to the dynamic and transient nature of user interactions within a single session. Unlike traditional recommender systems that rely on long-term user profiles, session-based recommendation must operate solely on the immediate context of a user's session. This poses significant hurdles, including the sparsity and volatility of session data, and the absence of historical user context. Furthermore, inferring a user's current implied interest from short but complex interaction patterns within a session is a non-trivial task. For instance, predicting the next item a user may interact with based on their current session interactions requires sophisticated techniques capable of discerning immediate user preferences solely from session data. Additionally, accurately capturing item transitions within session sequences is paramount for effective recommendation systems. Existing methods often struggle to achieve the required precision in generating embeddings for both items and sessions, highlighting the need for innovative approaches to address these challenges.

This project introduces a novel approach, Aggregated SR-GNN (Session-based Recommendation with Graph Neural Networks), inspired by the architecture proposed in the paper "Session-Based Recommendation with Graph Neural Networks" by Wu et al. (2019). Session-based recommendation diverges from traditional methods, focusing solely on user interactions within a session to tailor recommendations. Challenges such as sparsity and volatility of session data, and the absence of long-term user profiles necessitate innovative techniques. Aggregated SR-GNN dynamically updates node representations based on graph interactions, enabling nuanced capture of item transitions within sessions. It aggregates node embeddings based on their connections, offering a holistic representation of session data. Preliminary experimentation showcases Aggregated SR-GNN's efficacy in outperforming state-of-the-art methods, promising more accurate and personalized recommendations in dynamic online environments.

II. Introduction

Session-based recommendation represents a paradigm shift in the field of recommender systems, focusing on delivering item suggestions based solely on a user's interactions within a single session. This approach diverges from traditional methods that rely on long-term user profiles, instead tailoring recommendations to the immediate needs and preferences of users within their current session. The significance of session-based recommendation lies in its ability to address the dynamic nature of user behavior in today's digital landscape. With users engaging with online platforms for various purposes, each session presents a unique context and set of intentions. By providing timely and relevant recommendations based on real-time interactions, session-based recommendation enhances user satisfaction and engagement, ultimately driving improved user experiences across a wide range of domains. However, session-based recommendation also presents challenges, including the sparsity and volatility of session data. Unlike long-term user profiles, session data often comprises short and sparse sequences of interactions, making it difficult to capture meaningful user preferences and patterns. Additionally, the absence of historical context in session data requires recommender systems to adapt quickly to changing user interests within each session, adding complexity to the recommendation process. Despite these challenges, innovative approaches that leverage real-time analysis and contextual understanding of session data hold the key to unlocking the full potential of session-based recommendation, offering personalized and timely recommendations that meet the evolving needs of users in today's dynamic online environments.

Inferring a user's current implied interest from a relatively short but complex interaction pattern within a session poses a challenge in session-based recommendation systems. For instance, consider a scenario where a user sequentially visits five items on an e-commerce website. In this case, the first four visited items serve as training data, and the objective is to accurately predict the last item, as referenced in Figure 1. This task is complicated by the absence of long-term user profiles or auxiliary information, such as user logins, further emphasizing the need for sophisticated techniques capable of discerning immediate user preferences solely from session interactions.

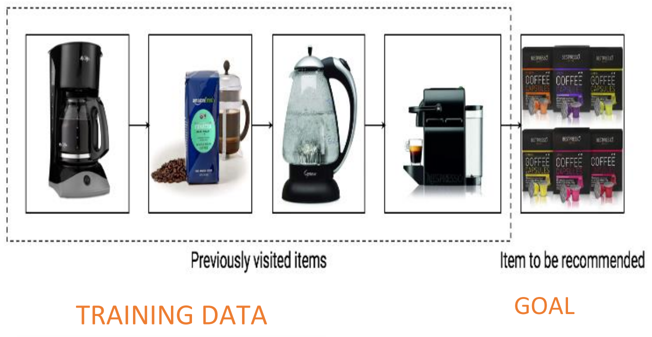


Figure 1: Predicting Next Item in Session

In session-based recommendation, two primary challenges hinder effective recommendation systems. Firstly, capturing the intricate behavioral patterns of users presents a significant hurdle, as understanding these behaviors is crucial for accurate recommendations. Particularly challenging is the task of capturing item transitions within session sequences, which forms the cornerstone of session-based recommendation. Secondly, obtaining accurate embeddings for both items and sessions is paramount for facilitating effective recommendations.

Existing methods, often representing sessions as sequential data, struggle to achieve the required precision in embedding generation. Addressing these challenges requires innovative approaches that can effectively capture item transitions within sessions and produce accurate embeddings, thereby enhancing the efficacy of session-based recommendation systems.

To address these challenges, this project introduces a novel approach: Aggregated SR-GNN (Session-based Recommendation with Graph Neural Networks). Inspired by the architecture proposed in the paper "Session-Based Recommendation with Graph Neural Networks" by Wu et al. (2019), our implementation extends this framework to enhance recommendation accuracy. The key novelty of our approach lies in the representation method. Unlike traditional methods that represent session sequences as sequential data, Aggregated SR-GNN dynamically updates node representations based on graph interactions, considering both incoming and outgoing edges. This dynamic aggregation approach enables more nuanced capture of item transitions within sessions, leading to improved recommendation accuracy.

Moreover, Aggregated SR-GNN does not explicitly differentiate between local and global embeddings, as in the original approach. Instead, it aggregates node embeddings based on their connections in the graph, offering a more holistic representation of session data. Furthermore, our approach combines original embeddings with updated embeddings based on reset and update gates, enhancing the robustness of the recommendation system.

Preliminary experimentation on real-world datasets has shown promising results, with Aggregated SR-GNN outperforming the state-of-the-art methods. These findings highlight the efficacy of our approach in addressing the challenges inherent in session-based recommendation, paving the way for more accurate and personalized recommendations tailored to individual user preferences in dynamic online environments.

III. Problem Description

Session-based recommendation systems predict user actions within anonymous sessions, lacking long-term user profiles. Unlike traditional methods, they rely solely on ongoing session interactions, posing a challenge in accurately predicting user preferences without historical context. Users engage in varied activities within each session, necessitating effective capture of immediate interests. This dynamic environment demands personalized recommendations tailored to the current session, enhancing user experience. The project addresses these challenges by developing techniques to improve the accuracy of session-based recommendation systems, enabling them to generate relevant recommendations based solely on the interactions occurring within each session.

IV. Related Work

Conventional Methods

Matrix Factorization: Often used for long-term user preference predictions based on user-item rating matrices. Less suitable for session-based recommendations due to its reliance on extensive user interaction histories, which are typically not available in such settings.

Item-Based Neighborhood Methods: Determine item similarity from co-occurrences in sessions, offering straightforward recommendations based on item affinity. They fall short in dynamic session-based environments as they do not consider the sequence in which items are engaged with, potentially missing out on capturing the evolving user interests.

Sequential Methods

Markov Chains: Offer a probabilistic approach to predicting user actions, looking at immediate past interactions to anticipate future ones. However, this method assumes that the next action is dependent only on the current state, not considering the possibility of longer, more complex sequences influencing the decision, which can be a significant limitation.

Factorizing Personalized Markov Chains (FPMC) [2]: An advanced sequential method that incorporates matrix factorization to leverage user-item interactions, along with Markov chains to maintain the sequential integrity of user sessions. It enhances sequence modelling by capturing not just immediate transitions but also the nuanced preferences exhibited across user sessions.

Our Approach with SR-GNN

SR-GNN: We built upon these methods by introducing a session-based recommendation system powered by Graph Neural Networks. Unlike conventional methods, SR-GNN excels in session-based scenarios by effectively utilising limited user click data and capturing the complex, non-linear interactions between items within sessions.

Advantages Over Baseline Models:

SR-GNN and its aggregated version outperform conventional and sequential methods by providing more nuanced item embeddings and understanding session contexts deeply. Our model demonstrates its advanced capabilities by achieving superior performance in key metrics such as Hit Rate, MRR, and NDCG, indicating not only an understanding of user's session-level intent but also a higher precision in the recommendation ranking.

V. Datasets

We have primarily chosen two datasets: Yoochoose and Digentica.

A. Yoochoose:

Clicks Dataset	Buys Dataset
Fields: *Session ID*: Unique identifier for the browsing session. *Timestamp*: Exact time when the click occurred. *Item ID*: Unique identifier for the clicked item. *Category:* The category to which the clicked item belongs.	*Fields:* *Session ID*: Unique identifier correlating to the session purchases were made. *Timestamp:* Precise time when the purchase was made. *Item ID*: Unique identifier for the purchased item. *Price*: Sale price of the purchased item. *Quantity*: Number of units of the item purchased.

Clicks Dataset: Captures user-click events on items, offering insights into browsing behavior.

Buys Dataset: Documents user purchase events, revealing conversion from views to sales.

Test Dataset: Consists of user click events like the Clicks Dataset but is used for model evaluation during the challenge. Excludes purchase events, focusing purely on clickstream data for testing purposes. Data was aggregated throughout the year 2014, reflecting a full range of seasonal user behaviors. Originates from an unspecified online European retailer, covering a diverse European market demographic.

B. Diginetica:

The Diginetica dataset provides a comprehensive collection of user interactions with an e-commerce platform, facilitating research and development in recommendation systems and e-commerce analytics.

Contents:

- User interactions: Clicks, views, purchases, and product categories.
- Metadata: Timestamps, user IDs, item IDs, and contextual information.

Data Format:

- Structured format: Typically provided as a CSV file or similar tabular format.
- Each row represents a single user interaction.
- Columns include attributes such as user ID, item ID, interaction type, timestamp, and additional metadata.

The dataset comprises four main categories: Clicks, Views, Purchases, and Product Categories. The Clicks dataset records user interactions by mapping browsing sessions, including Session ID, Timestamp, Item ID, and Category. Similarly, the Views dataset tracks product views with the same fields. The Purchases dataset chronicles completed sales with Session ID, Timestamp, Item ID, Price, and Quantity. Finally, the Product Categories dataset organizes items into categories, linking Item ID with Category Name for more detailed analysis. Together, these datasets provide comprehensive insights into user behavior, browsing habits, and purchasing preferences, facilitating informed decision-making and personalized recommendations.

· Utilizing these datasets, we conducted a series of experiments to validate the predictive accuracy and the capability of our model to recommend relevant items within a session.

· Through iterative testing, we were able to refine the model's algorithms, ensuring they are well-suited to real-world e-commerce applications.

· The experiments were carefully designed to simulate authentic user sessions, enabling our model to learn from actual interaction patterns and to predict user preferences with high precision.

VI. Methodology

Sessions as Graphs The core elements of graphs are nodes and edges. Graph ML models try to capture the representations of nodes and the edges connecting them, as represented in Figure 2. For our prediction task, each session is represented as a graph and each interacted item in the session is represented as a node. After looking into a session graph, we predict what the next item the user clicked was.

The proposed architecture of SR-GNN comprises four fundamental steps:

1. **Session Graph Modeling:**
   - Each session sequence is represented as a directed graph. Edge weight normalization is performed by dividing the edge occurrence by the outdegree of the start node.
2. **Node Representation Learning:** Learn representations for graph nodes, capturing users' interactions with items.

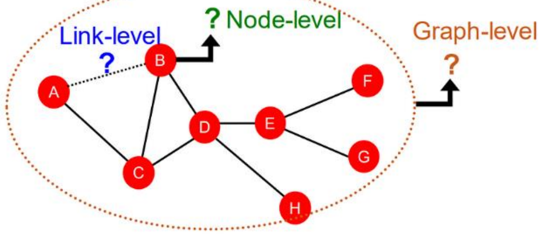


Figure 2: Graph Based Predictions

3. **Session Representation Generating:**
    - Generate session representations by aggregating node representations to capture session-level information.
4. **Making Recommendation:**
    - Provide personalized suggestions to users based on session representations.

These steps collectively enable SR-GNN to effectively capture the dynamics of user-item interactions within sessions and make accurate recommendations tailored to individual user preferences.

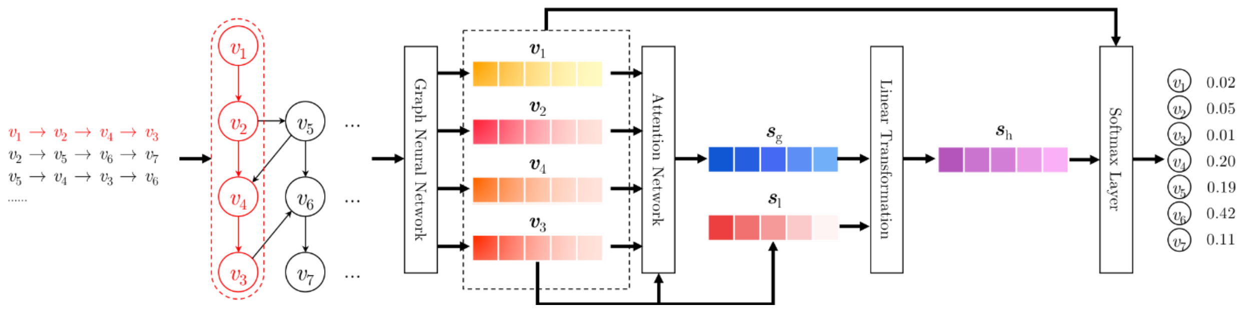


Figure 3: SR-GNN architecture. Original image inspired by the diagram from the SR-GNN paper.

Node Embeddings:

The initial step in constructing the SR-GNN model involves creating node embeddings. This process entails encoding unique item IDs into vector-form embeddings, laying the foundation for subsequent operations. By generating embeddings for each node in the graph, the model establishes a basis for representing the items within a session effectively.

Gated Graph Neural Network (GG-NN) Layer:

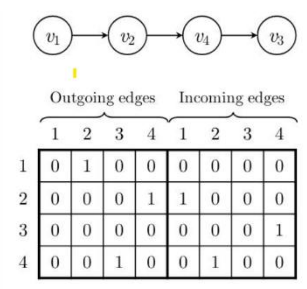
Following the creation of node embeddings, the SR-GNN model, as represented in Figure 3, utilizes Gated Graph Neural Network (GG-NN) layers to refine these initial representations. GG-NNs offer significant advantages over traditional Graph Neural Networks (GNNs). While GNNs propagate representations for each node in the graph, GG-NNs leverage gated recurrent units (GRUs) and unroll recurrence for fixed steps. This architecture enables GG-NNs to achieve better temporal modeling and gradient computation through Backpropagation Through Time (BPTT).

GGNNs excel in capturing temporal dynamics in dynamic graphs, which is crucial for tasks like social network analysis and dynamic recommendation systems. Moreover, the gated mechanisms inherent in GGNNs facilitate efficient information propagation across the graph, leading to enhanced communication and collaboration among nodes. Additionally, GGNNs offer scalability for learning representations in large-scale graphs while remaining versatile across various graph-based tasks and domains, making them indispensable tools in graph analytics and machine learning research.

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

Building an Adjacency Matrix:

An integral part of the SR-GNN construction process is building an adjacency matrix. This matrix allows the model to explicitly learn how nodes interact within a session based on all available session information, as represented in Figure 4. By incorporating session connectivity data into the adjacency matrix, the model gains insights into the relationships between items, enhancing its ability to capture the dynamics of user interactions effectively.



	1	2	3	4	1	2	3	4
1	0	1	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0
3	0	0	0	0	0	0	0	1
4	0	0	1	0	0	1	0	0

Figure 4: Building Adjacency Matrix

Adjacency Matrix into Activations:

The transformation of the adjacency matrix into activations is a critical step in the SR-GNN framework, facilitating the model's understanding of the dynamic relationships within a session. By converting the adjacency matrix into activations, the model gains insights into how nodes interact with each other over time, enabling it to capture the evolving dynamics of user-item interactions effectively.

$$\mathbf{z}_{s,i}^{t} = \sigma\left(\mathbf{W}_z \mathbf{a}_{s,i}^{t} + \mathbf{U}_z \mathbf{v}_i^{t-1}\right), \qquad (2)$$

$$\mathbf{r}_{s,i}^{t} = \sigma\left(\mathbf{W}_r \mathbf{a}_{s,i}^{t} + \mathbf{U}_r \mathbf{v}_i^{t-1}\right), \qquad (3)$$

$$\widetilde{\mathbf{v}_i^t} = \tanh\left(\mathbf{W}_o \mathbf{a}_{s,i}^{t} + \mathbf{U}_o\left(\mathbf{r}_{s,i}^{t} \odot \mathbf{v}_i^{t-1}\right)\right), \qquad (4)$$

$$\mathbf{v}_i^t = \left(1 - \mathbf{z}_{s,i}^{t}\right) \odot \mathbf{v}_i^{t-1} + \mathbf{z}_{s,i}^{t} \odot \widetilde{\mathbf{v}_i^t}, \qquad (5)$$

Generating Session Embeddings:

In the SR-GNN framework, session embeddings play a crucial role in capturing the overall user preferences and interests within a session. This process involves representing each session directly by the node embeddings associated with the items interacted with during that session.

$$\boldsymbol{s}_{\mathrm{l}} = \boldsymbol{v}_n \qquad (6)$$

Local Embedding:

The local embedding represents the user's current interest and emphasizes the impact of the last clicked item within the session. By focusing on the most recent interaction, the local embedding provides insight into the immediate preferences of the user, enabling the model to make contextually relevant recommendations based on the latest interactions.

Global Embedding:

In contrast, the global embedding captures the user's overall preferences and interests across the entire session. This is achieved through a soft-attention network, which assigns weights to each node embedding based on its importance in representing the user's global preferences. By considering the entire session history, the global embedding provides a comprehensive understanding of the user's preferences, facilitating the generation of personalized recommendations tailored to the user's long-term interests.

$$\alpha_i = \boldsymbol{q}^{\top} \sigma(\boldsymbol{W}_1 \boldsymbol{v}_n + \boldsymbol{W}_2 \boldsymbol{v}_i + \boldsymbol{c}), \qquad (7)$$

$$\boldsymbol{s}_{\mathrm{g}} = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i. \qquad (8)$$

Hybrid Embedding Creation

To create a comprehensive session representation, the local and global embeddings are combined into a hybrid embedding. Through a straightforward linear transformation, these embeddings are weighted and summed, blending the immediate preferences captured by the local embedding with the broader user preferences represented by the global

$$\boldsymbol{s}_{\mathrm{h}} = \boldsymbol{W}_3 \left[\boldsymbol{s}_{\mathrm{l}}; \boldsymbol{s}_{\mathrm{g}}\right] \qquad (9)$$

embedding. This integrated representation ensures a balanced understanding of the session, facilitating accurate recommendations based on evolving user interests.

Making Recommendations

In the recommendation phase, the model computes a score for each candidate item by performing a dot product between its embedding and the session representation. This score reflects the item's relevance to the user's current session. Subsequently, a softmax function (eq.10) is applied to obtain the output vector, which represents the probability distribution over candidate items. The model's performance is evaluated using the cross-entropy loss function (eq.11), measuring the disparity between the predicted probabilities and the actual outcomes for each session graph.

$$\boxed{\hat{\boldsymbol{y}} = \operatorname{softmax}\left(\boldsymbol{s}_{\mathrm{h}}^{\top} \boldsymbol{v}_i\right) \qquad (10)}$$

$$\boxed{\mathcal{L}(\hat{\boldsymbol{y}}) = -\sum_{i=1}^{m} \boldsymbol{y}_i \log\left(\hat{\boldsymbol{y}}_i\right) + \left(1-\boldsymbol{y}_i\right) \log\left(1-\hat{\boldsymbol{y}}_i\right) \qquad (11)}$$

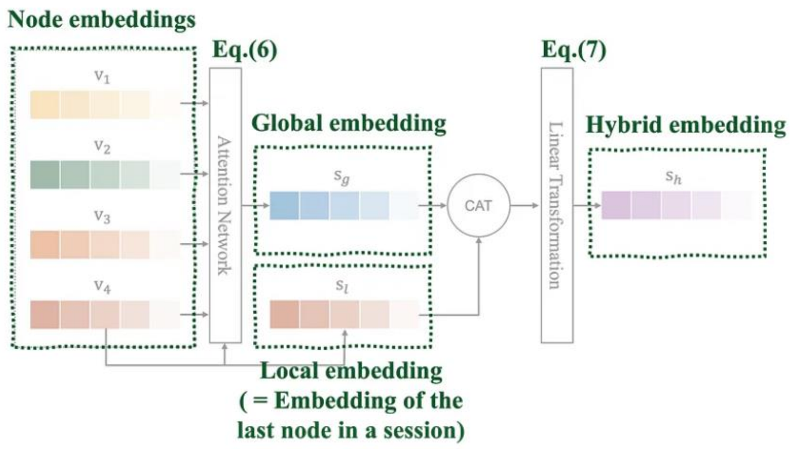


Figure 5: Different components of the architecture

The Novelty of our Approach

Dynamic Node Representation: Our method introduces dynamic updates to node representations, considering interactions between graph nodes, including both incoming and outgoing edges. This contrasts with traditional session embedding generation, where session representations rely solely on static node embeddings.

Unified Embeddings: Unlike conventional techniques, our approach does not explicitly distinguish between local and global embeddings. Instead, it aggregates node embeddings based on their graph connections, offering a unified representation. In contrast, traditional methods employ separate local and global embeddings to capture immediate user interest and overall preferences.

Enhanced Embedding Fusion: Our method combines original embeddings with updated ones, leveraging reset and update gates. In contrast, conventional techniques typically employ a linear transformation to combine local and global embeddings. This refined fusion process contributes to improved recommendation accuracy.

Overall, our novel approach emphasizes dynamic aggregation of node embeddings based on graph interactions, departing from the traditional session embedding generation approach.

The function we have used to alter the existing architecture:

```
def aggregate(self, A, emb_items):
        h_input_in = self.fc_edge_in(torch.matmul(A[:, :, :A.shape[1]], emb_items))
        h_input_out = self.fc_edge_out(torch.matmul(A[:, :, A.shape[1]: 2 * A.shape[1]], emb_items))
        h_inputs = torch.cat([h_input_in, h_input_out], 2)
        r_input, z_input, h_input = self.fc_rzh_input(h_inputs).chunk(chunks=3, dim=2)
        r_old, z_old = self.fc_rz_old(emb_items).chunk(chunks=2, dim=2)
        reset = torch.sigmoid(r_old + r_input)
        update = torch.sigmoid(z_old + z_input)
        h = torch.tanh(h_input + self.fc_h_old(reset * emb_items))
        return (1 - update) * emb_items + update * h
```

VII. Experiment

A. Metrics

Hit Rate (HR)

Definition: Hit Rate measures whether the actual item (ground truth) appears in the top-K recommended items. It's a binary indicator, typically averaged over several instances to provide a proportion of hits in the test set.

Calculation: The function checks if the ground truth item (gt) is in the top k_metric predictions pred. If found, the hit counter is incremented.

Mean Reciprocal Rank (MRR)

Definition: MRR is the average of the reciprocal ranks of the first correct prediction. For each query, you find the rank of the first relevant item and take its reciprocal. This metric emphasizes the importance of the rank of the first relevant answer.

Calculation: If the ground truth item is in the predictions, MRR is updated by adding the reciprocal of the position (1-based index) of the first correct prediction.

Normalized Discounted Cumulative Gain (NDCG)

Definition: NDCG accounts for the position of the hit by penalizing hits at lower ranks. It's particularly useful when the position of the recommendation is relevant (e.g., the top results are more important than those at the bottom).

Calculation: NDCG uses a logarithmic discount for items at lower ranks and compares it to the ideal Discounted Cumulative Gain (DCG) where the relevant item is in the first position. The ndcg value is updated by adding the discounted value of the hit based on its rank.

Layer Configuration

Graph Neural Network Layer: Single-step GNN architecture, sufficient for capturing the immediate dependencies in the data.

Recommendation Cutoff: Top 20 items considered for evaluation metrics, balancing the depth of the recommendation list with practical usability.

B. SR-GNN *VS* Baseline models

SR-GNN vs. FPMC

Sequential Modeling: While FPMC [2] integrates sequential behavior modeling with user preferences effectively, SR-GNN can potentially capture more complex, non-linear transitions due to its graph-based structure.

User Preferences: FPMC's matrix factorization component can grasp user-specific tastes. SR-GNN, with its graph neural network framework, might enrich this by considering more intricate user-item interaction patterns within the graph representation.

SR-GNN vs. Item-KNN

Similarity Computation: Item-KNN [3] bases recommendations on similarity metrics like cosine similarity. In contrast, SR-GNN can implicitly learn item similarities in a high-dimensional space, potentially leading to more nuanced recommendations.

Handling Sparsity: Item-KNN may struggle with sparse data, whereas SR-GNN can still learn from limited interactions by leveraging graph connections.

SR-GNN vs. STAMP

Attention Mechanism: STAMP [4] applies an attention mechanism to capture both the user's general and immediate session interests. SR-GNN might offer a different approach to attention by exploiting node-level relations and dependencies directly within its graph structure.

Temporal Dynamics: While STAMP is proficient in prioritizing recent interactions, SR-GNN can be more flexible in modeling various lengths of user sessions by its graph-based nature, which does not assume sequentially.

C. Experimental settings

Model Hyperparameters

Hidden Layer Size: Set to 100 units, providing sufficient complexity for feature representation.

Learning Rate: Initialized at 0.001 to balance the speed and stability of convergence.

L2 Regularization: Applied at a rate of 0.00001 to mitigate overfitting by penalizing large weights.

Batch Size: Limited to 20 to manage memory efficiently and ensure training stability.

Training Dynamics

Epochs: Capped at 8 to prevent overfitting, as additional epochs do not yield performance improvements.

Step: Adjusted every 5 epochs to methodically decrease the learning rate.

Gamma: Set to 0.1, reducing the learning rate by a factor of ten to fine-tune the model weights.

Optimization Framework:

Loss Function: Cross-Entropy Loss, suitable for classification tasks by comparing predicted probabilities with the actual distribution.

Optimizer: Adam with weight decay to adaptively adjust the learning rate for different parameters and encourage sparsity.

Learning Rate Scheduler: StepLR, which decreases the learning rate at specified intervals, improving convergence.

D. Results

Training Phase

Loss: Achieved a low training loss of 4.67099, demonstrating the model's capability in accurately fitting the training data.

Hit Rate (HR): A high HR of 0.5212 indicates that more than half of the time, the model successfully included the true next item within its top recommendations.

Mean Reciprocal Rank (MRR): An MRR of 0.3373 highlights the model’s effectiveness in ranking the correct items higher on the list.

Normalized Discounted Cumulative Gain (NDCG): With an NDCG score of 0.2831, the model proves its proficiency in ranking items by their relevance.

Validation Phase

Loss: Recorded a validation loss of 5.01881, signifying the model's generalization performance.

Hit Rate (HR): The HR of 0.4343 on the validation set suggests a robust predictive ability on unseen data.

Mean Reciprocal Rank (MRR): The MRR of 0.3201 demonstrates the model's consistent performance in placing the correct recommendation near the top of the list.

Normalized Discounted Cumulative Gain (NDCG): An NDCG of 0.2535 further indicates a good ranking quality in the validation phase.

E. On Comparison with Baseline Models

The Aggregated SR-GNN model exhibits the best performance with an MRR score of 32.01, as referenced in Table 1. This underscores the advanced capability of GNN architectures in capturing the complex patterns within session data, significantly improving the recommendation rankings.

Algorithm	MRR for top 20
FPMC	15.01
Item - KNN	21.81
Stamp	29.67
SR-GNN	30.94
Aggregated SR-GNN	32.01

Table 1: Comparison of our implementation with Baseline Models.

F. Code Base

Our implementation is shared as a repository in GitHub, here: https://github.com/tejasreeparasa/IST_597_Spring2024

VIII. Ablation Study

Node Representation Updates: Evaluate the impact of dynamic node representation updates by comparing performance with static representations. Assess how different update mechanisms affect the ability of the model to capture evolving user preferences within sessions. Investigate whether the inclusion of dynamic updates improves recommendation accuracy or if static representations suffice.
Separate vs. Unified Embeddings: Compare the effectiveness of using separate local and global embeddings versus unified embeddings in the recommendation process. Analyze how the choice between separate and unified embeddings influences the model's ability to capture both short-term and long-term user preferences. Determine if unified embeddings lead to simplifications in the model architecture without sacrificing recommendation quality.
Edge Weight Normalization: Explore the necessity of edge weight normalization in session graph modeling by comparing results with and without this normalization step. Assess how edge weight normalization affects the model's sensitivity to variations in session graph structures and interaction patterns. Determine if edge weight normalization contributes significantly to the stability and robustness of the recommendation system.

IX. Discussion

In the discussion, we acknowledge that the novelty of our approach contributed to achieving a better Mean Reciprocal Rank (MRR) score, indicating improved recommendation performance. However, it's important to note that our evaluation focused primarily on MRR and may not have covered all relevant evaluation metrics. This presents an area for improvement, as future studies could include a more comprehensive set of metrics to provide a more nuanced assessment of the model's performance.

Regarding implementation, we experimented with a PyTorch implementation of the model. While this allowed us to leverage the capabilities of PyTorch for efficient computation and training, we maintained a close adherence to the original architecture proposed in the research paper. As a result, our implementation may not have explored significant deviations or modifications from the original approach. Exploring more extensive modifications or adaptations could be an avenue for future research, potentially leading to further enhancements in recommendation accuracy or efficiency.

X. Conclusion

In conclusion, session-based recommendation systems offer a promising approach to delivering personalized item suggestions based on users' interactions within single sessions, addressing the dynamic nature of online user behavior. Challenges such as data sparsity and the need for accurate embeddings have driven the development of innovative methods like Aggregated SR-GNN. Our implementation, inspired by Wu et al.'s architecture, dynamically updates node representations based on graph interactions, yielding improved recommendation accuracy. While our approach showed promising results in preliminary experimentation, future studies could explore a broader range of evaluation metrics and further modifications to enhance recommendation performance in dynamic online environments.

XI. Future Work

In future endeavors, we aim to extend the capabilities of session-based recommendation systems. Exploring the integration of heterogeneous graphs promises richer representations of user interactions, accommodating diverse node and edge types for more nuanced recommendations. Additionally, incorporating User Intent Classification holds potential for better understanding and predicting user preferences. Further advancements could involve the incorporation of Global Graphs to capture broader item interactions across sessions, enhancing recommendation accuracy. Dual-Intent Modeling presents an avenue for refining recommendations, particularly for new items, by leveraging attention mechanisms and historical data distributions. These avenues pave the way for continued refinement and innovation in session-based recommendation systems, ultimately leading to more personalized and effective user experiences.

XII. References

[1] Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., & Tan, T. (2019). Session-Based Recommendation with Graph Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01), 346–353. Link

[2] Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web* (pp. 811–820). Link

[3] Xia, X., Yin, H., Yu, J., Wang, Q., Cui, L., & Zhang, X. (2022). Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. arXiv preprint

[4] Liu, Q., Zeng, Y., Mokhosi, R., & Zhang, H. (2018). STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. Link

[5] Liu, Q.; Zeng, Y.; Mokhosi, R.; and Zhang, H. (2018). Stamp: Short-term attention/memory priority model for session-based recommendation. In *KDD*, 1831–1839.

[6] Mao, J.; Xu, W.; Yang, Y.; Wang, J.; Huang, Z.; and Yuille, A. (2015). Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*.

[7] Marino, K.; Salakhutdinov, R.; and Gupta, A. (2017). The more you know: Using knowledge graphs for image classification. In *CVPR*, 20–28.

[8] Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. (2016a). Session-based recommendations with recurrent neural networks. In *ICLR*.

[9] Hidasi, B.; Quadrana, M.; Karatzoglou, A.; and Tikk, D. (2016b). Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys*, 241–248.
[10] Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. (2015). Line: Large-scale information network embedding. In *WWW*, 1067–1077.
[11] Cho, K.; Van Merri¨enboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *EMNLP*, 1724–1734.
[12] Duvenaud, D.; Maclaurin, D.; AguileraIparraguirre, J.; G´ omez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*.
[13] Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. (2009). The graph neural network model. *TNN*, 20(1):61–80.
[14] Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, 3776–3784.