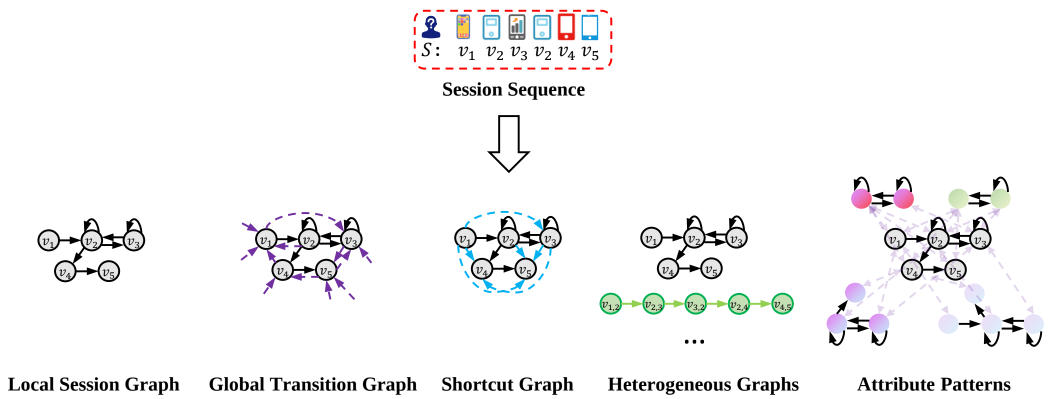

Figure 7: Different granularities of graphs derived from a session, where the violet edges, blue edges, and green edges correspond to the global collaborations, shortcuts, and higher-level heterogeneous connections, respectively.

A Limitations

There are three limitations to our current proposed method and evaluation. First, our method separately processes and retrieves patterns for each attribute type. We do not merge all attributes in a candidate pool because we aim for our method to easily generalize to real recommendation systems with hundreds of attribute types and category hierarchies. The current implementation supports adding a new attribute type to a model as long as its embeddings align with the embeddings of other attributes. Second, we conducted experiments based on "clean" session data. Most E-commerce platforms do not have truly clean data on product attributes, so attribute data, in general, is very sparse and full of invalid values. We performed human-centric attribute regularization to drop products without valid attribute values, which may create a gap compared to a real industrial system. Third, the evaluation does not consider the same products with different identifiers. Therefore, evaluating results (especially MRR) cannot accurately reflect the performance. To better reflect the real performance with error tolerance, a larger K is suggested. The current comparison is still fair for all algorithms, and we address this synonym problem in attribute estimation in § 5.4, where we merge attribute values based on semantics and syntax.

B Transition Graph Density

The graph structure is crucial for neural networks to capture explicit transitions and implicit connections. A local session records the history of a user's clicks or purchases, which is usually sparse. In contrast, the global collaborative graph could be extremely dense because each pair of items may have a potential connection. From this perspective, the density indicates the explicit information provided from session data. On the other hand, different graph topologies and densities also present different focuses and challenges. The sparse local transition graph emphasizes current intents, while the global collaboration indicates broader interests and revenues. Graph neural networks excel at capturing local features, but a large number of neighbors can overshadow important connections with less significant ones. Considering that previous methods have focused on different granularities individually, we summarize them in Figure 7 and compare them in terms of optimization interpretations[1].

- **Local** session graphs correspond to item transitions within a session, where edges are created between two consecutively clicked/purchased items. The density is usually sparse (slightly greater than 1.0), allowing exploration and global collaboration to be learned through model parameters instead of explicit connections. Therefore, generalizing to unseen click patterns becomes challenging.
- **Global** transition graphs record all collaborations. In a real industrial system, the density is usually beyond one hundred or even one thousand. Ideally, any session can benefit from this global collaborative information, including multi-hop connections. However, optimizing graph neural

[1] Graphs are typically considered undirected in practical algorithms.

networks to learn such topologies (due to oversmoothing) and building a large model for real-time inference (due to latency and streaming processing) pose challenges.

- **Shortcut** graphs aim to avoid constructing global graphs and make session learning more efficient. They were proposed by LESSR [1] to address information loss in graph convolutions. Specifically, they allow latent items to be aware of all previous clicks, resembling shortcuts for multi-hop neighbors in the directed local session graph. However, they lack the extensive exploration capabilities of the global transition graph and suffer from oversmoothing issues due to dense connections.
- **Heterogeneous** graphs strike a balance between shortcuts and local adjacency. Nodes with different numbers of items are categorized into different groups, and transition edges capture varying levels of spatial continuity. From a high-level perspective, this graph is sparser than the local session graph, resulting in faster convergence for optimization. However, the propagation of high-order information introduces additional processing costs and the risk of overfitting.
- **Patterns**, especially attribute patterns, should be the most efficient features for recommendations in a large candidate item pool. Each pattern can be considered a higher-grained heterogeneous graph. However, pattern filtering can significantly eliminate noise influence, not to mention the benefits gained from offline indexing. Besides, the partial match of patterns can provide the intent information from other sessions, making the learning and prediction more reliable and steady.

C Experimental Data

C.1 Public Benchmarks

We choose two public benchmarks for session-based recommendation evaluation: *diginetica* [2] is CIKM Cup 2016 that contains the browser logs and anonymized transactions; *Tmall* [3] comes from a competition in IJCAI-15 which collects anonymous users' shopping logs on the Tmall online website. We acquire attributes from the original data and drop items without attributes or with invalid values. Therefore, the performance of baselines may not be exactly same as the reported numbers in the original papers.

C.2 E-commerce Data Collection

We collect E-commerce data from our log systems in two months. We follow the same procedure to clean and process session data in *beauty*, *books*, and *electronics* domains[4]:

I. We focus on successful purchases so that we only keep sessions ending with "purchase" actions.

II. To make sure previous clicks can reflect the purchase intent, we drop actions 20 minutes ago.

III. We filter out items with missing attributes (i.e., books without publishers, authors, or genre, and electronics without colors and brands).

IV. We adopt the 20-core setting to finalize the item sets, in which items appear on at least 20 different days.

V. Only sessions whose length is no greater than 50 are preserved.

VI. We retrieve item attributes in our attribute databases.

VII. For GNN models that requires the global transition graph from training data, we maintain 12 neighbors based on the co-occurrence, which is consistent with GCE-GNN [21].

[2] `https://competitions.codalab.org/competitions/11161`

[3] `https://tianchi.aliyun.com/dataset/dataDetail?dataId=42`

[4] The sampled data scales and distributions are different in real systems due to out-of-domain items filtering.

Table 6: Statistics of datasets based on timestamps.

	Public		Industrial (E-commerce)		
	diginetica	Tmall	Beauty	Books	Electronics
#User	57,623	7,576	2.6 M	3.2 M	10.2 M
#Item	43,074	39,768	39.2 K	94.8 K	244.7 K
#Click	993,163	438,315	27.2 M	38.8 M	115.6 M
Avg. Len.	4.850	6.649	10.325	11.912	11.249
#Train	630,789	303,181	19.6 M	28.2 M	84.1 M
#Valid	78,708	33,735	2.4 M	3.5 M	10.5 M
#Test	78,907	35,481	2.5 M	3.8 M	10.6 M
#Attribute	category : 995	category : 821 brand : 4,304	category : 359 color : 1,101 brand : 4,359 size : 1,883	category : 18 publisher : 2,751 author : 27,651 genre : 2,634	type : 123 category : 881 color : 2,096 brand : 24,196
#Pattern	category : 1,866	category : 33,582 brand : 2,497	category : 970 color : 4,059 brand : 254 size : 1,091	category : 24 publisher : 4,370 author : 1,399 genre : 12,535	type : 9,289 category : 13,991 color : 146,402 brand : 14,043
Density	Local: 0.886 Global: 11.329 Shortcut: 2.512 Heterogeneous: 0.543 Pattern: 1.023	Local: 1.249 Global: 10.222 Shortcut: 4.983 Heterogeneous: 0.707 Pattern: 1.165	Local: 4.510 Global: 70.504 Shortcut: 29.827 Heterogeneous: 3.412 Pattern: 1.095	Local: 3.554 Global: 99.389 Shortcut: 26.649 Heterogeneous: 2.333 Pattern: 1.085	Local: 2.910 Global: 128.041 Shortcut: 19.865 Heterogeneous: 2.049 Pattern: 1.189

C.3 Data Split

We follow previous settings that split training/validation/testing data based on timestamps. For *diginetica*, we gather the last 8-14 days as validation, the last 7 days as testing, and remaining as training. For *Tmall*, we use the last 101-200 seconds as validation, the last 100 seconds as testing, and remaining as training. For our industrial E-commerce data (i.e., *Beauty*, *Books*, *Electronics*), we select the last 6-10 days as validation, the last 5 days as testing, and remaining as training.

C.4 Data Statistics

Table 6 summarizes the statistics of the experimental datasets based on timestamps. The density is calculated based on undirected graphs, which would be doubled during graph convolution in practice. *Local density*, as used in SR-GNN and GC-SAN, corresponds to the average density of local session graphs in E-commerce sessions. On the other hand, *global density*, as used in GCE-GNN, refers to the density of the global collaborative graph obtained by connecting all adjacent items appearing in all sessions. *Shortcut density*, as used in LESSR, is the density resulting from connecting all items in a single session as a complete graph. *Heterogeneous density*, as used in MSGIFSR, refers to the average density of the heterogeneous graphs obtained by regarding the consecutive adjacent two nodes as a fine-grained intent unit. Lastly, *pattern density*, as used in FAPAT, is the density of the acquired frequent and compact patterns. From Table 6, it is evident that leveraging patterns is the most effective way of characterizing user intents because other graph topologies vary with data sources and scales, making it difficult to generalize and provide stable performance. Besides, patterns can be preprocessed as indicies to aid recommendations, making them more practical in industrial scenarios. Moreover, it is easy to update attribute patterns dynamically, whereas other graph structures are more closely coupled with input sessions and are more sensitive to tiny variations.

D Baselines

We compare our method with following baselines:
Sequence-based methods

- **FPMC** [16] learns the representation of session via Markov-chain based methods.
- **GRU4Rec** [6] is the first RNN-based approach that simulates the Markov Decision Process (MDP) but has a better generalization.
- **NARM** [9] is a attention-based RNN model to learn session embeddings.
- **STAMP** [12] adopts attention mechanism between the last item to previous histories to represent users' short-term interests.

- **CSRM** [19] proposes to engage an inner memory encoder and external memory network to capture correlations between neighborhood sessions to enrich the collaborative representations.
- **S3-Rec** [31] is the first pretrained SBR model that predicts items, attributes, and segments during the pretraining stage.
- **M2TRec** [17] is a metadata-aware multi-task Transformer model. In the original paper, the authors ignore item embeddings. For a fair comparison, we also regard the item ids as one of metadata.

Graph-based methods

- **SR-GNN** [22] is the first GNN-based model for the SBR task, which transforms the session data into a direct unweighted graph and learns the representation of the item-transitions graph.
- **GC-SAN** [25] uses gated GNNs to extract local context information and then self-attention to obtain the global representation.
- **S2-DHCN** [24] transforms the session data into hyper-graphs and line-graphs and encodes them via GCNs to enhance the session representations.
- **GCE-GNN** [21] aggregates two levels of item embeddings from session graphs and global graphs with soft attention.
- **LESSR** [1] preserves the edge order and constructs shortcuts to encode sessions for GNNs.
- **MSGIFSR** [5] captures the user intents from multiple granularities to relieve the computational burden of long-dependency. In experiments, we search the best model from the level-1, level-2, and level-3 consecutive intent units.

E Experimental Settings

We fix all embeddings and hidden dimensions as 100, and the batch size is searched among {100, 200, 500} for all methods. We also choose the number of layers/iterations (if applicable) from the validation performance (e.g., MRR@10). A learning scheduler with 10% linear warmup and 90% decay is associated with the Adam optimizer [7]. The initial learning rate is set as 1e-3, and the regularization weight is tuned among {1e-4, 1e-5, 1e-6}. We seek the dropout probability between two modules from {0.0, 0.2, 0.4}, but fix the attention dropout rate as 0.2. The number of attention heads is empirically set as 4. We follow the setting of GCE-GNN that the maximum one-hop neighbor number in GAT is 12. In the interest of fairness, we also set the maximum selected pattern number as 12. Hyper-parameter tuning is time costly on our industrial data so that we use the best combinations obtained from one day transactions. We implement our methods and run experiments with Python and PyTorch over 8 x A100 NVIDIA GPUs.

F Experimental Results

Due to the space limit, we only report some results in the main content. More comprehensive comparisons are shown in Tables 7-11, where standard deviations are enclosed in brackets. The best and second-best results are respectively highlighted in bold and underlined. Methods that use attributes are marked with ‡, and * indicates the p-value < 0.0001 in t-test.

Table 7: Performance evaluation for next-item prediction on *diginetica*.

Model	diginetica					
	Hits@10	NDCG@10	MRR@10	Hits@20	NDCG@20	MRR@20
FPMC	31.57(0.04)*	17.40(0.01)*	13.08(0.02)*	43.19(0.05)*	20.33(0.03)*	13.88(0.03)*
GRU4Rec	<u>36.77</u>(0.14)*	20.71(0.05)*	15.80(0.03)*	<u>49.68</u>(0.06)*	<u>23.97</u>(0.03)*	16.70(0.03)*
NARM	35.98(0.10)*	20.18(0.06)*	15.36(0.06)*	48.89(0.12)*	23.44(0.06)*	16.26(0.06)*
STAMP	33.59(0.15)*	18.89(0.18)*	14.41(0.19)*	45.87(0.15)*	22.00(0.18)*	15.26(0.19)*
CSRM	33.97(0.08)*	19.43(0.03)*	14.98(0.03)*	45.83(0.02)*	22.42(0.02)*	15.80(0.02)*
S3-Rec‡	33.48(0.13)*	18.58(0.09)*	14.04(0.10)*	45.97(0.08)*	21.74(0.09)*	14.90(0.10)*
M2TRec‡	29.67(0.43)*	16.30(0.24)*	12.23(0.18)*	41.23(0.63)*	19.22(0.29)*	13.02(0.20)*
SR-GNN	35.21(0.02)*	19.68(0.04)*	14.94(0.04)*	47.99(0.04)*	22.90(0.04)*	15.82(0.04)*
GC-SAN	35.25(0.09)*	19.72(0.04)*	14.97(0.03)*	47.87(0.09)*	22.90(0.04)*	15.85(0.03)*
S2-DHCN	30.76(0.07)*	17.04(0.14)*	12.86(0.16)*	42.39(0.07)*	19.98(0.13)*	13.66(0.16)*
GCE-GNN	36.32(0.09)*	<u>20.77</u>(0.07)*	<u>16.02</u>(0.07)*	48.67(1.12)*	23.89(0.23)*	<u>16.87</u>(0.03)*
LESSR	33.68(0.05)*	18.71(0.03)*	14.14(0.03)*	46.23(0.11)*	21.88(0.05)*	15.01(0.03)*
MSGIFSR	34.74(0.09)*	19.43(0.06)*	14.76(0.07)*	46.23(0.11)*	21.88(0.05)*	15.01(0.03)*
FAPAT‡	**37.42**(0.10)	**21.31**(0.03)	**16.39**(0.04)	**50.41**(0.15)	**24.59**(0.06)	**17.29**(0.04)
Improv.	*3.03%*	*2.60%*	*2.31%*	*1.46%*	*2.59%*	*2.49%*

Table 8: Performance evaluation for next-item prediction on *Tmall*.

Model	Tmall					
	Hits@10	NDCG@10	MRR@10	Hits@20	NDCG@20	MRR@20
FPMC	13.71(0.16)*	9.02(0.02)*	7.56(0.03)*	16.44(0.23)*	9.71(0.04)*	7.74(0.02)
GRU4Rec	18.82(0.17)*	12.28(0.11)*	10.25(0.09)*	22.68(0.21)*	13.25(0.12)*	10.51(0.10)*
NARM	22.74(0.20)*	15.46(0.12)*	13.19(0.10)*	26.73(0.26)*	16.47(0.13)*	13.47(0.10)*
STAMP	24.32(0.31)*	16.55(0.29)*	14.12(0.29)*	28.40(0.35)*	17.58(0.30)*	14.41(0.29)*
CSRM	25.13(0.19)*	18.56(0.18)*	16.48(0.18)*	27.94(0.15)*	19.27(0.17)*	16.68(0.18)*
S3-Rec‡	18.24(0.11)*	12.30(0.07)*	10.46(0.06)*	22.31(0.17)*	13.32(0.08)*	10.74(0.06)*
M2TRec‡	11.42(0.21)*	7.56(0.06)*	6.36(0.11)*	13.75(0.35)*	8.15(0.04)*	6.52(0.10)*
SR-GNN	18.21(0.51)*	12.11(0.32)*	10.20(0.28)*	21.34(0.49)*	12.91(0.31)*	10.42(0.28)*
GC-SAN	19.29(0.14)*	12.80(0.07)*	10.78(0.13)*	23.18(0.23)*	13.78(0.04)*	11.05(0.12)*
S2-DHCN	22.00(0.36)*	13.36(0.21)*	10.68(0.17)*	27.23(0.33)*	14.69(0.20)*	11.05(0.17)*
GCE-GNN	<u>28.33</u>(0.13)*	<u>20.01</u>(0.12)*	<u>17.32</u>(0.13)*	<u>30.24</u>(0.16)*	<u>20.50</u>(0.13)*	<u>17.45</u>(0.13)*
LESSR	20.99(0.26)*	14.64(0.18)*	12.13(0.19)*	25.92(0.23)*	13.96(0.22)*	10.50(0.23)*
MSGIFSR	23.18(0.19)*	15.19(0.11)*	12.69(0.10)*	27.78(0.25)*	16.35(0.11)*	13.01(0.09)*
FAPAT‡	**32.45**(0.21)	**22.02**(0.15)	**18.72**(0.13)	**36.18**(0.21)	**22.97**(0.14)	**18.99**(0.13)
Improv.	*14.19%*	*10.04%*	*8.08%*	*19.64%*	*12.05%*	*8.83%*

Table 9: Performance evaluation for next-item prediction on *Beauty*.

Model	Beauty					
	Hits@10	NDCG@10	MRR@10	Hits@20	NDCG@20	MRR@20
FPMC	72.00	57.20	52.42	75.91	58.19	52.70
GRU4Rec	73.95	58.19	53.13	78.54	59.36	53.45
NARM	88.09	70.44	64.68	91.50	71.31	64.93
STAMP	80.08	63.76	58.47	83.84	64.72	58.73
CSRM	89.74	75.28	70.56	92.61	76.01	70.77
S3-Rec‡	89.64	<u>75.56</u>	<u>70.99</u>	92.53	<u>76.30</u>	<u>71.19</u>
M2TRec‡	80.13	65.97	61.65	83.66	66.87	61.65
SR-GNN	88.69	70.42	64.44	91.74	71.20	64.65
GC-SAN	86.67	70.80	64.71	88.98	72.50	65.97
S2-DHCN	7.25	5.38	4.80	8.87	5.79	4.91
GCE-GNN	89.34	73.15	67.80	91.29	73.65	67.94
LESSR	89.95	71.29	65.18	<u>92.98</u>	72.06	65.40
MSGIFSR	<u>90.18</u>	73.62	65.18	92.50	74.21	65.65
FAPAT‡	**92.72**	**76.29**	**71.09**	**94.10**	**76.87**	**71.24**
Improv.	*2.82%*	*0.97%*	*0.14%*	*1.20%*	*0.75%*	*0.07%*

Table 10: Performance evaluation for next-item prediction on *Books*.

Model	Books					
	Hits@10	NDCG@10	MRR@10	Hits@20	NDCG@20	MRR@20
FPMC	36.51	24.32	20.49	41.90	25.69	20.87
GRU4Rec	47.21	31.86	27.02	53.55	33.47	27.46
NARM	76.09	54.22	47.13	80.83	55.43	47.36
STAMP	61.49	42.13	35.95	67.46	43.65	36.37
CSRM	<u>78.69</u>	56.70	49.54	<u>82.88</u>	57.77	49.83
S3-Rec‡	75.00	<u>58.54</u>	<u>53.23</u>	79.45	<u>59.67</u>	<u>53.55</u>
M2TRec‡	32.56	22.58	24.98	35.39	25.70	22.78
SR-GNN	66.55	47.55	41.32	69.77	48.37	41.55
GC-SAN	72.56	54.92	49.25	75.73	56.05	50.14
S2-DHCN	4.69	3.42	3.03	5.60	3.65	3.09
GCE-GNN	77.61	57.60	51.00	80.03	58.22	51.17
LESSR	73.72	53.86	47.36	82.31	54.77	47.61
MSGIFSR	72.93	52.23	45.66	76.33	53.09	45.66
FAPAT‡	**81.62**	**61.08**	**54.39**	**85.12**	**61.97**	**54.64**
Improv.	*3.72%*	*4.34%*	*2.18%*	*2.70%*	*3.85%*	*2.04%*

Table 11: Performance evaluation for next-item prediction on *Electronics*.

Model	Electronics					
	Hits@10	NDCG@10	MRR@10	Hits@20	NDCG@20	MRR@20
FPMC	37.87	26.91	23.42	42.07	27.97	23.71
GRU4Rec	58.46	40.69	35.02	64.42	42.21	35.44
NARM	61.10	41.20	32.05	77.36	44.56	33.75
STAMP	59.30	42.04	36.53	67.94	45.07	36.97
CSRM	62.28	44.35	38.59	67.47	45.67	38.96
S3-Rec‡	74.36	<u>56.03</u>	**50.16**	79.63	<u>57.37</u>	**50.53**
M2TRec‡	57.32	44.84	40.85	61.70	45.95	41.15
SR-GNN	<u>74.86</u>	54.30	47.66	<u>79.66</u>	55.52	48.00
GC-SAN	72.76	53.37	45.98	77.34	46.34	49.91
S2-DHCN	4.18	2.65	2.18	5.08	2.88	2.24
GCE-GNN	72.93	53.74	47.59	78.49	55.15	47.98
LESSR	72.91	50.46	43.26	78.78	51.96	43.67
MSGIFSR	73.56	53.83	47.77	77.45	54.73	48.02
FAPAT‡	**78.36**	**56.81**	<u>49.80</u>	**82.81**	**57.94**	<u>50.12</u>
Improv.	*4.68%*	*1.39%*	*-0.07%*	*3.95%*	*0.99%*	*-0.81%*

References

[1] Tianwen Chen and Raymond Chi-Wing Wong. Handling information loss of graph neural networks for session-based recommendation. In *SIGKDD*, pages 1172–1180, 2020.

[2] Yen-Liang Chen, Mi-Hao Kuo, Shin-yi Wu, and Kwei Tang. Discovering recency, frequency, and monetary (RFM) sequential patterns from customers' purchasing data. *Electronic Commerce Research and Applications*, 8(5):241–251, 2009.

[3] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. A (sub)graph isomorphism algorithm for matching large graphs. *TPAMI*, 26(10):1367–1372, 2004.

[4] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, pages 2978–2988, 2019.

[5] Jiayan Guo, Yaming Yang, Xiangchen Song, Yuan Zhang, Yujing Wang, Jing Bai, and Yan Zhang. Learning multi-granularity consecutive user intent unit for session-based recommendation. In *WSDM*, pages 343–352, 2022.

[6] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[8] Nils M. Kriege, Fredrik D. Johansson, and Christopher Morris. A survey on graph kernels. *Appl. Netw. Sci.*, 5(1):6, 2020.

[9] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *CIKM*, pages 1419–1428, 2017.

[10] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *WWW*, pages 2320–2329, 2022.

[11] Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. Interest-aware message-passing GCN for recommendation. In *WWW*, pages 1296–1305, 2021.

[12] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. STAMP: short-term attention/memory priority model for session-based recommendation. In *SIGKDD*, pages 1831–1839, 2018.

[13] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. Rethinking the item order in session-based recommendation with graph neural networks. In *CIKM*, pages 579–588, 2019.

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67, 2020.

[15] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[16] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820, 2010.

[17] Walid Shalaby, Sejoon Oh, Amir Afsharinejad, Srijan Kumar, and Xiquan Cui. M2trec: Metadata-aware multi-task transformer for large-scale and cold-start free session-based recommendations. In *RecSys*, pages 573–578, 2022.

[18] Nino Shervashidze, S. V. N. Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten M. Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, volume 5, pages 488–495, 2009.

[19] Meirui Wang, Pengjie Ren, Lei Mei, Zhumin Chen, Jun Ma, and Maarten de Rijke. A collaborative session-based recommendation approach with parallel memory modules. In *SIGIR*, pages 345–354, 2019.

[20] Shoujin Wang, Qi Zhang, Liang Hu, Xiuzhen Zhang, Yan Wang, and Charu Aggarwal. Sequential/session-based recommendations: Challenges, approaches, applications and opportunities. In *SIGIR*, pages 3425–3428, 2022.

[21] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xianling Mao, and Minghui Qiu. Global context enhanced graph neural networks for session-based recommendation. In *SIGIR*, pages 169–178, 2020.

[22] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *AAAI*, pages 346–353, 2019.

[23] Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *ICLR*, 2022.

[24] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *AAAI*, pages 4503–4511, 2021.

[25] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, pages 3940–3946, 2019.

[26] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, pages 721–724, 2002.

[27] Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. Graphformers: Gnn-nested transformers for representation learning on textual graph. In *NeurIPS*, pages 28798–28810, 2021.

[28] Ghim-Eng Yap, Xiaoli Li, and Philip S. Yu. Effective next-items recommendation via personalized sequential pattern mining. In *DASFAA*, volume 7239, pages 48–64, 2012.

[29] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jaeboum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *WSDM*, 2023.

[30] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. In *ICLR*, 2020.

[31] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, pages 1893–1902, 2020.