Variational Collective Graph AutoEncoder for Multi-behavior Recommendation

Yang Liu[1], Qianzhen Rao[1], Weike Pan[1*], Zhong Ming[1,2*]

{2110276162, 2100271001}@email.szu.edu.cn, {panweike, mingz}@szu.edu.cn

[1]College of Computer Science and Software Engineering, Shenzhen University, China

[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, China

Problem Definition

Multi-Behavior Recommendation

- Input: the user-item multi-behavior interaction graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \mathcal{U} \cup \mathcal{I} = \{v\} = \{1, 2, ...V\}$ is the set of nodes in the graph, $\mathcal{E}$ is the set of edges which includes K behavior types.
- Output: the likelihood $\hat{y}_{ui}$ that the user u will interact with the item i under the target behavior b_K.

Challenge

1. The complex *transition relationships* across different types of behaviors. Most existing multi-behavior recommendation methods mainly focus on fusing different behaviors, while ignoring the transition relationships among them.
2. The varying *semantic strength* of different types of behaviors. Different auxiliary behaviors exhibit varying degrees of correlations with the target behavior.

Overall of Our Solution

1. We propose a novel model called VCGAE which leverages the advantages of VAE and GNNs to model users' different types of behaviors.
2. We design a behavior transition network that captures complex transition relationships across different types of behaviors to learn users' personal preferences.
3. We design a behavior contrastive regularization module that extracts different correlations between users' auxiliary behaviors and target behavior to learn the semantic strength of distinct behaviors.

Related Work (1/2)

- Single-Behavior Recommendation
  - Multi-VAE [Liang et al., 2018] is an AE-based model and its key idea is to model the data as a set of latent variables that follow a prior distribution, and then generate new data points by sampling from the learned distribution.
  - LightGCN [He et al., 2020] is a graph neural network model that simplifies the design of GCNs by removing the feature transformation and nonlinear activation operations.
  - VGAE [Kipf and Welling, 2016] is a generative model designed for graph data that leverages the frame of VAE to learn the low-dimensional representation of high-dimensional data.

VGAE is an innovative method that combines the strengths of VAE and GNNs, but it is limited to only modeling single-behavior data. This limitation motivates us to design a new method to solve the problem of multi-behavior recommendation.

Related Work (2/2)

- Multi-Behavior Recommendation
  - VAE++ [Ma et al., 2022] is a VAE-based model that utilizes three types of information, including the purchase behavior, the auxiliary behaviors, and their mixed behaviors. But it fails to capture the high-order interaction information between users and items.
  - GHCF [Chen et al., 2021] is a non-sampling graph neural network model that designs the relation-aware GCN propagation layers to exploit the collaborative high-hop signals. However, it ignores the transition relationships among different behaviors.
  - EHCF [Chen et al., 2020] is a non-sampling method that correlates each behavior in a transition way to capture the users' preferences. However, it doesn't effectively capture the diversity of transition modes across different behaviors.

Inspired by the above methods, our VCGAE inherits the advantages of VAE and GNNs and designs two modules to learn the semantic strength and transition relationships of different behaviors.

VCGAE

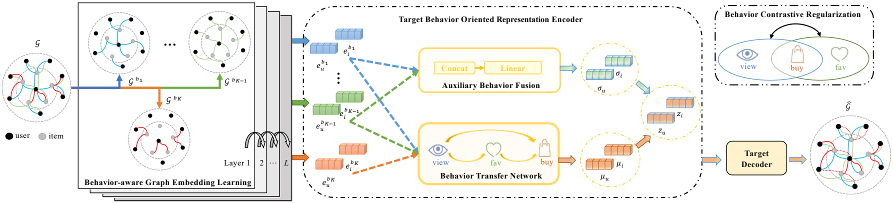
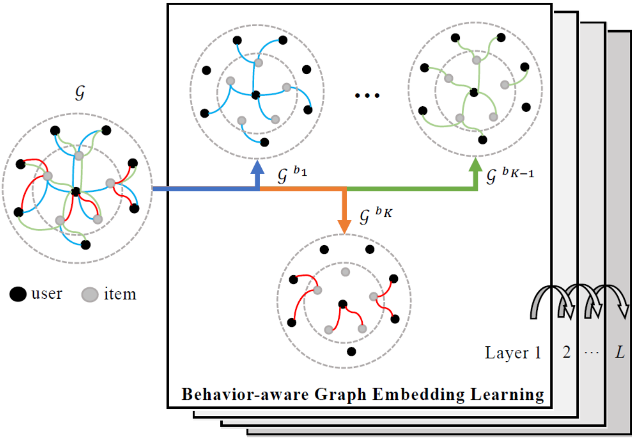


Figure: Illustration of our VCGAE. We assume that there are K types of behaviors here, where $b_1, \cdots, b_{K-1}$ are auxiliary behaviors (i.e., view and favorite, respectively) and b_K is the target behavior (i.e., buy).

Modules Overview

- **Behavior-aware Graph Embedding Learning**
- Target Behavior Oriented Representation Encoder
- Target Decoder
- Behavior Contrastive Regularization

Behavior-aware Graph Embedding Learning (1/3)



Behavior-aware Graph Embedding Learning (2/3)

- We aim to learn the representation of the users and items under each behavior.
- For each user u, we refine his or her embedding $e_u^{b_k,(l+1)} \in \mathbb{R}^{1\times d}$ in layer $l+1$ under behavior b_k by aggregating his or her neighboring nodes in layer l:

$$e_u^{b_k,(l+1)} = \sum_{i\in\mathcal{N}_u^{b_k}} \frac{1}{\sqrt{\left|\mathcal{N}_u^{b_k}\right|}\sqrt{\left|\mathcal{N}_i^{b_k}\right|}} e_i^{b_k,(l)} \tag{1}$$

where $\mathcal{N}_u^{b_k}$ denotes the set of items that are interacted by user u under behavior b_k, $\mathcal{N}_i^{b_k}$ denotes the set of users that interact with item i under behavior b_k.

Behavior-aware Graph Embedding Learning (3/3)

- After L layers propagation, we obtain the user u's representation at different layers. Then we adopt the mean operation to derive the user u's final behavior embedding $\overline{e}_u^{b_k} \in \mathbb{R}^{1\times d}$ under behavior b_k:

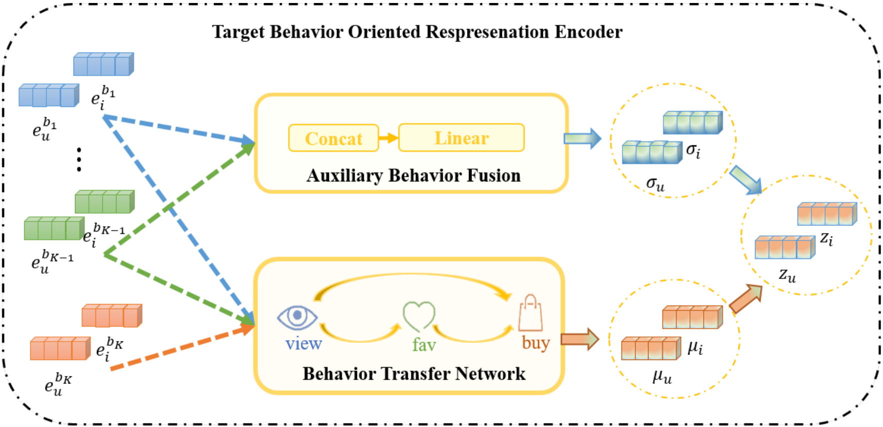
$$\overline{e}_u^{b_k} = \sum_{l=0}^{L} \frac{1}{L+1} e_u^{b_k,(l)} \tag{2}$$

  By doing the same operation for the item side, we can get the smoothed item i's behavior embedding $\overline{e}_i^{b_k} \in \mathbb{R}^{1\times d}$.

Modules Overview

- Behavior-aware Graph Embedding Learning
- **Target Behavior Oriented Representation Encoder**
- Target Decoder
- Behavior Contrastive Regularization

Target Behavior Oriented Representation Encoder (1/6)



Target Behavior Oriented Representation Encoder (2/6)

- Behavior Transition Network
  - In order to explicitly model the sequence relationships among different behaviors, we make the assumption that different behaviors can be arranged in a specific order based on behavior semantic strength: $b_1 \rightarrow b_2 \rightarrow \dots \rightarrow b_K$.
  - For each user u, his or her behavior transition embedding $\check{e}_u^{b_k} \in \mathbb{R}^{1 \times d}$ is obtained by summing his or her own behavior embedding $\overline{e}_u^{b_k}$ with the transition embedding of the other behaviors that occurred before it in the sequence:

$$\check{e}_u^{b_k} = \begin{cases} \overline{e}_u^{b_k} & \text{if } k = 1 \\ \overline{e}_u^{b_k} \oplus \overline{e}_u^{b_1} & \text{if } k = 2 \\ \overline{e}_u^{b_k} \oplus \check{e}_u^{b_{k-1}} \oplus \dots \oplus \check{e}_u^{b_1} & \text{if } k > 2 \end{cases} \tag{3}$$

where $\oplus$ denotes the element-wise addition operation.

Target Behavior Oriented Representation Encoder (3/6)

- Behavior Transition Network
  - When $k = K$, we can obtain the user u's target behavior transition embedding $\check{e}_u^{b_K}$ which takes into account the transition of different auxiliary behaviors to the target behavior.
  - Then we define it as the user u's mean vector of the Gaussian posterior distribution: $\mu_u = \check{e}_u^{b_K}$.
  - By doing the same operation for the item side, we can get the item i's mean vector $\mu_i \in \mathbb{R}^{1\times d}$.

Target Behavior Oriented Representation Encoder (4/6)

- Auxiliary Behavior Fusion
  - In a Gaussian distribution, the variance always fluctuates around the mean.
  - In a recommendation scenario, the mean typically represents a user's purchasing interest, and the variance indicates the range of the user's interest changes which can be captured by auxiliary behaviors' features.
  - The variance $\sigma_u^2 \in \mathbb{R}^{1\times d}$ can be computed as follows:

$$\sigma_u^2 = MLP(\overline{e}_u^{b_1} \| \overline{e}_u^{b_2} \| ... \| \overline{e}_u^{b_{K-1}}) \tag{4}$$

    where $\|$ is the concatenation operation, and MLP is a multi-layer perceptron commonly used in VAE.
  - By doing so, we can get the item i's variance $\sigma_i^2 \in \mathbb{R}^{1\times d}$.

Target Behavior Oriented Representation Encoder (5/6)

- Latent Variable Sampler
  - With the mean μ_v and the variance σ_v^2 of each node, the latent variable $z_v \in \mathbb{R}^{1\times d}$ can be obtained by sampling from a variational distribution q:

$$q(\mathbf{Z} \mid \mathbf{A}) = \prod_{v=1}^{V} q\left(z_v \mid \mathbf{A}\right), \quad \text{with} \quad q\left(z_v \mid \mathbf{A}\right) = \mathcal{N}\left(z_v \mid \mu_v, \operatorname{diag}\left(\sigma_v^2\right)\right) \tag{5}$$

where $V = (|\mathcal{U}| + |\mathcal{I}|)$ is the number of nodes, $\mathbf{Z}$ is the latent variable matrix, $\mathbf{A}$ is the adjacency matrix of the user-item sub-graph $\mathcal{G}^{b_K}$ under the target behavior b_K.

Target Behavior Oriented Representation Encoder (6/6)

- Latent Variable Sampler
  - During training, the stochastic nature of the latent variables poses a challenge for computing gradients of the objective function. To solve this problem, the reparameterization trick has been introduced and widely used [Kingma and Welling, 2013, Rezende et al., 2014].
  - Specifically, we first sample ϵ from a standard normal distribution: $\epsilon \sim \mathcal{N}(0, 1)$, and then we have $z_v = \mu_v + \epsilon \otimes \sigma_v$, where $\otimes$ is the element-wise product operation.

Modules Overview

- Behavior-aware Graph Embedding Learning
- Target Behavior Oriented Representation Encoder
- **Target Decoder**
- Behavior Contrastive Regularization

Target Decoder

- After obtaining the latent variable z_u and z_i, the goal of the target decoder is to reconstruct the target adjacency matrix $\mathbf{A}$:

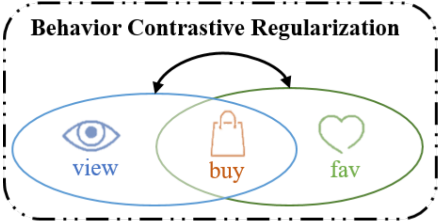
$$p(\widehat{\mathbf{A}} \mid \mathbf{Z}) = \prod_{u=1}^{N} \prod_{i=1}^{M} p\left(\widehat{A}_{ui} \mid z_u, z_i\right), \qquad \text{with} \quad p\left(\widehat{A}_{ui} \mid z_u, z_i\right) = \hat{y}_{ui} = z_u z_i^\top \tag{6}$$

where $\widehat{\mathbf{A}}$ is the reconstructed adjacency matrix and $\widehat{A}_{ui}$ is the probability of a connection between user u and item i.

Modules Overview

- Behavior-aware Graph Embedding Learning
- Target Behavior Oriented Representation Encoder
- Target Decoder
- **Behavior Contrastive Regularization**

Behavior Contrastive Regularization (1/2)



- Different behaviors have different semantic features and strengths.
- There are different semantic correlations between different auxiliary behaviors and target behaviors.

Behavior Contrastive Regularization (2/2)

- These semantic correlations between different auxiliary behaviors and target behavior can be expressed as a similarity in mathematics. In particular, we use cosine similarity to calculate the similarity between two behaviors:

$$sim(x, y) = cos(x, y) = \frac{xy^{\intercal}}{\|x\|\|y\|} \tag{7}$$

- If behavior y has a higher similarity with x compared to z in practical scenarios, we define (x, y) as a positive pair and (x, z) as a negative pair. Then we aim to increase the distance between positive and negative pairs:

$$BCR(x, y, z) = (1 - \sigma(sim(x, y) - sim(x, z)))^2 \tag{8}$$

where σ is the sigmoid function.

Objective Function

- Following VGAE [Kipf and Welling, 2016], the loss function can be represented as:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{REC} + \mathcal{L}_{KL} \\ &= \mathbb{E}_{q(\mathbf{Z}|\mathbf{A})}[\log p(\mathbf{A} \mid \mathbf{Z})] - \mathrm{KL}[q(\mathbf{Z} \mid \mathbf{A}) \| p(\mathbf{Z})] \end{aligned} \tag{9}$$

where the first term is the reconstruction term, and the second term is the KL divergence term. The reconstruction term measures the difference between the predicted and actual adjacency matrices, while the KL divergence term regularizes the latent space by ensuring that the learned distributions match a prior Gaussian distribution.

Objective Function

- Here, we utilize the Bayesian personalized ranking (BPR) loss as the reconstruction term in our model, which is better suited for recommendation systems:

$$\mathcal{L}_{BPR} = -\sum_{u=1}^{N} \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \ln \sigma \left(z_u z_i^{\top} - z_u z_j^{\top} \right) \tag{10}$$

- The overall loss function of our VCGAE is as follows:

$$\mathcal{L} = \mathcal{L}_{BPR} + \beta \mathcal{L}_{KL} + \lambda \mathcal{L}_{BCR} \tag{11}$$

where β and λ are the coefficients that control the weight of the KL regularization and the behavior contrastive regularization, respectively.

Research Questions

- **RQ1.** How does our VCGAE perform compared with the state-of-the-art recommendation baselines?
- **RQ2.** How do the different components in our VCGAE affect the recommendation performance?
- **RQ3.** How does our VCGAE perform when handling different numbers of behaviors?
- **RQ4.** How do the hyperparameters affect the performance of our VCGAE?

Datasets(1/3)

- We use three widely used datasets, i.e., Jdata 2019 (JD), Tmall, and UserBehavior (UB), to evaluate the performance of our VCGAE.
- **JD.** This is an open dataset collected from JD, which is one of the largest e-commerce platforms in China. There are three types of behaviors in this dataset, including view, tag-as-favorite, and purchase.
- **Tmall.** This is an open dataset collected from Tmall. This dataset contains three types of behaviors, namely view, tag-as-favorite, and purchase.
- **UB.** This is an open dataset about user behavior data from Taobao. There are four types of behaviors, including view, tag-as-favorite, add-to-cart, and purchase.

Datasets(2/3)

- We process the above datasets as follows:
  - If there are duplicate records of (user, item, behavior) in a session, we only keep the earliest records.
  - For the items that are purchased fewer than k times, all these items will be removed (JD: k=20, Tmall: k=20, UB: k=10).
  - To delete cold-start users, we remove the sessions with fewer than k purchase records (JD: k=5, Tmall: k=10, UB: k=5).
  - Split records into three sets: training, validation, and test. Sorting all records in ascending order according to their timestamps. The first 80% of records are assigned to the training set, the next 10% to the validation set, and the remaining 10% to the test set.
  - Users with purchasing records in the validation set or test set but not in the training set will be removed.
- Note that we don't use the leave-one-out method to split the dataset to prevent the utilization of future records from other related users during the prediction of a user's current behavior.

Datasets(3/3)

Table: Statistics of the datasets used in the experiments.

Dataset	#Users	#Items	#View	#Favorite	#Cart	#Purchase
JD	10,690	13,465	254,003	9,289	–	71,872
Tmall	17,202	16,177	998,032	121,146	–	240,828
UB	20,443	30,947	632,029	27,745	84,244	133,708

Evaluation Metrics

- To evaluate the performance of our VCGAE, we adopt four representative evaluation metrics in the field of top-N recommendation, i.e., precision (Prec@N), recall (Rec@N), normalized discounted cumulative gain (NDCG@N) and hit ratio (HR@N).
- Users tend to pay more attention to the top few recommended items[Collins et al., 2018]. Therefore, we report results for $N = 10$.

Baselines(1/2)

- Five single-behavior algorithms:
  - Multi-VAE [Liang et al., 2018] is an extension of the variational autoencoder (VAE) model, designed to reconstruct the input data for single-behavior collaborative filtering tasks.
  - VGAE [Kipf and Welling, 2016] combines graph convolutional networks (GCNs) with variational inference to generate latent variables of graph-structured data.
  - SCVG [Ding et al., 2021] is a model that builds upon the VGAE architecture, which employs a novel semi-deterministic variational architecture to infer latent variables.
  - NGCF [Wang et al., 2019] is a state-of-the-art GNN-based algorithm that utilizes GCNs to model user-item interactions.
  - LightGCN [He et al., 2020] is a simplified version of NGCF, while still achieving competitive performance in some scenarios.

Baselines(2/2)

- Four multi-behavior algorithms:
  - EHCF [Chen et al., 2020] is a non-sampling method that utilizes linear functions to capture the relationships among users' different types of behaviors.
  - MBGCN [Jin et al., 2020] is a GCN-based model that designs a unified graph to represent the multiple user-item interaction.
  - VAE++ [Ma et al., 2022] is a VAE-based method that designs two encoder modules to enhance the representations of nodes by utilizing three types of signals, including the target behavior, the auxiliary behavior, and their mixed behaviors.
  - GHCF [Chen et al., 2021] adopts the operations of graph convolution with the framework of multi-task learning to model users' different types of behaviors.

Parameter Configurations

- For a fair comparison, we fix the embedding size d to 100 for all models [Ma et al., 2022] and optimize them with the Adam optimizer with batch size fixed to 500.
- For the three deep learning-based models, i.e., Multi-VAE, VAE++, and EHCF, we select the learning rate from $\{0.0001, 0.001, 0.01, 0.05\}$, and the dropout ratio is set to 0.5 to prevent overfitting.
- For Multi-VAE, we follow the settings in [Liang et al., 2018] and adopt a structure with 1 hidden layer in MLP.
- For VAE++, the parameter settings are consistent with Multi-VAE.

Parameter Configurations

- For the seven GNN-based models, i.e., VGAE, SCVG, NGCF, LightGCN, MBGCN, GHCF, and our VCGAE, the number of layers of graph neural network L is searched from $\{1, 2, 3, 4\}$.
- For GHCF, we explore various combinations of weights for each behavior to find the best configuration.
- For our VCGAE, the KL regularization coefficient β is searched from $\{0.01, 0.1, 1\}$, and the behavior contrastive regularization coefficient λ is selected from $\{0.01, 0.1, 1\}$.
- It should be noted that, for every test user, we utilize the predicted scores to rank all of their uninteracted items, rather than a sample of uninteracted items [Dallmann et al., 2021].

Main Results (RQ1) (1/4)

Table: Recommendation performance of five single-behavior models, four multi-behavior models, and our VCGAE on JD, Tmall, and UB.

Dataset	Metrics	Multi-VAE	VGAE	SCVG	NGCF	LightGCN	EHCF	MBGCN	VAE++	GHCF	VCGAE
JD	Prec@10	0.0288	0.0254	0.0300	0.0300	0.0323	0.0322	<u>0.0339</u>	0.0311	0.0315	**0.0342**
	Rec@10	0.1308	0.1080	0.1246	0.1203	0.1357	0.1391	0.1401	0.1402	<u>0.1428</u>	**0.1453**
	HR@10	0.1733	0.1497	0.1671	0.1695	0.1815	0.1881	<u>0.1923</u>	0.1919	0.1900	**0.1950**
	NDCG@10	0.0835	0.0793	0.0790	0.0794	0.0856	0.0888	0.0911	0.0921	<u>0.0924</u>	**0.0960**
Tmall	Prec@10	0.0013	0.0007	<u>0.0014</u>	0.0010	0.0012	0.0013	<u>0.0014</u>	0.0013	0.0013	**0.0014**
	Rec@10	<u>0.0062</u>	0.0041	0.0058	0.0048	0.0055	<u>0.0062</u>	0.0057	<u>0.0062</u>	<u>0.0062</u>	**0.0064**
	HR@10	0.0119	0.0074	0.0120	0.0097	0.0103	<u>0.0121</u>	<u>0.0121</u>	0.0119	0.0118	**0.0128**
	NDCG@10	0.0036	0.0034	0.0035	0.0027	<u>0.0038</u>	0.0036	0.0036	<u>0.0038</u>	0.0035	**0.0042**
UB	Prec@10	0.0024	0.0014	0.0023	0.0017	0.0023	0.0052	0.0038	0.0048	<u>0.0062</u>	**0.0082**
	Rec@10	0.0142	0.0086	0.0134	0.0102	0.0133	0.0280	0.0171	0.0257	<u>0.0336</u>	**0.0466**
	HR@10	0.0200	0.0123	0.0189	0.0154	0.0190	0.0454	0.0291	0.0409	<u>0.0533</u>	**0.0684**
	NDCG@10	0.0086	0.0065	0.0074	0.0061	0.0077	0.0170	0.0120	0.0164	<u>0.0205</u>	**0.0371**

Main Results (RQ1) (2/4)

We can have the following observations:

- Our VCGAE outperforms other baseline models across all three datasets on four evaluation metrics. Such remarkable performance improvement can be attributed to the effectiveness of our designed framework, which can better learn the users' behavioral preferences.
- The performance enhancements on UB are considerably more significant than those on JD and Tmall. One possible explanation for this could be that UB contains four different types of behavior, which enables our VCGAE to capture user preferences and behavior transition relationships more comprehensively.

Main Results (RQ1) (3/4)

- The multi-behavior recommendation models (e.g., GHCF, VAE++, MBGCN) generally outperform single-behavior models (e.g., Multi-VAE, SCVG, LightGCN). This demonstrates that incorporating the auxiliary behavior data can often lead to a more comprehensive understanding of user preferences, addressing the issue of sparse target behavior data, and thereby enhancing the recommendation performance.
- However, this trend is less pronounced on Tmall, possibly due to the noise in auxiliary behaviors or their weak correlation with the target behavior, limiting the benefits of multi-behavior modeling.

Main Results (RQ1) (4/4)

From the perspective of the multi-behavior, we can have the following observations:

- VCGAE outperforms VAE++ in the VAE-based methods and surpasses GHCF in the GNN-based methods. Particularly, the effects are more pronounced on UB, demonstrating that the combination of VAE and GNNs can lead to significant improvement in the multi-behavior recommendation.
- The GNN-based models for multi-behavior recommendation achieve good performance in many cases, demonstrating the effectiveness in capturing high-order connectivity over user-item interaction graphs, allowing for a more comprehensive understanding of the complex relationships between users and items.

Ablation Study (RQ2) (1/4)

Table: Recommendation performance of our VCGAE by removing different components, i.e., behavior-aware graph embedding learning (BGEL), behavior transition network (BTN), and behavior contrastive regularization (BCR), respectively, for ablation studies on JD, Tmall, and UB.

Dataset	Method	Prec@10	Rec@10	HR@10	NDCG@10
JD	w/o BGEL	0.0277	0.1194	0.1644	0.0779
	w/o BTN	0.0332	0.1388	0.1857	0.0885
	w/o BCR	**0.0349**	**0.1489**	**0.1978**	0.0944
	VCGAE	0.0342	0.1453	0.1950	**0.0960**
Tmall	w/o BGEL	0.0005	0.0027	0.0047	0.0016
	w/o BTN	0.0012	0.0056	0.0110	0.0036
	w/o BCR	0.0013	0.0060	0.0124	0.0041
	VCGAE	**0.0014**	**0.0064**	**0.0128**	**0.0042**
UB	w/o BGEL	0.0011	0.0069	0.0103	0.0040
	w/o BTN	0.0025	0.0140	0.0209	0.0084
	w/o BCR	0.0048	0.0270	0.0394	0.0183
	VCGAE	**0.0082**	**0.0466**	**0.0684**	**0.0371**

Ablation Study (RQ2) (2/4)

We have the following observations:

- **"w/o BGEL".** The performance of our VCGAE without BGEL declines on all three datasets, demonstrating the effectiveness of graph neural networks (GNNs) in learning node representations.
- **"w/o BTN".** The performance of our VCGAE without BTN declines on all three datasets. This demonstrates the usefulness of the behavior transition network in capturing behavior transition relationships, especially with a higher number of behaviors.
- **"w/o BCR".** The performance of our VCGAE without BCR decreases on both Tmall and UB. This indicates the advantage of our designed BCR module in capturing semantic correlations between auxiliary behaviors and target behavior. The effect on JD is less pronounced due to the fewer target behaviors, potentially introducing noise from auxiliary behaviors when constructing behavior pairs.

Impact of Auxiliary Behavior Data (RQ3) (1/2)

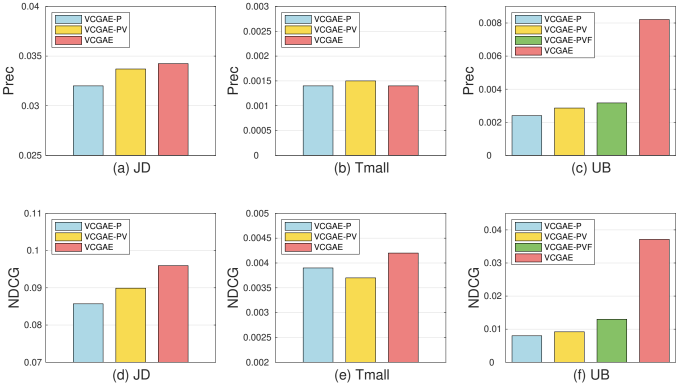


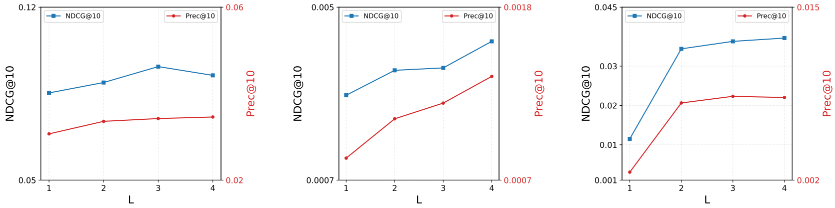
Figure: Recommendation performance of our VCGAE by leveraging different auxiliary behavior data for ablation studies on JD, Tmall, and UB.

Impact of Auxiliary Behavior Data (RQ3) (2/2)

We have the following observations:

- As the number of auxiliary behaviors increases, the recommendation performance also improves in general. Moreover, the best results are achieved when all behavior data is utilized in most cases. This observation showcases the effectiveness of our VCGAE in integrating diverse behavioral data to improve the performance of the recommendation system.
- The NDCG@10 score of VCGAE-PV is lower than that of VCGAE-P on Tmall. This could be attributed to the large volume of view behavior data in the Tmall dataset, which may contain a certain amount of noise, negatively affecting the recommendation performance.

Hyperparameter Sensitivity (RQ4) (1/3)



(a) JD (b) Tmall (c) UB

Figure: Recommendation performance of our VCGAE with different depths of GNNs on JD, Tmall, and UB.

Hyperparameter Sensitivity (RQ3) (2/3)

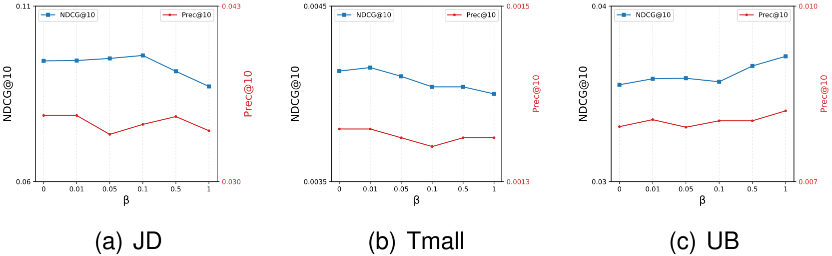


Figure: Recommendation performance of our VCGAE with different values of β on JD, Tmall, and UB.

Hyperparameter Sensitivity (RQ3) (3/3)

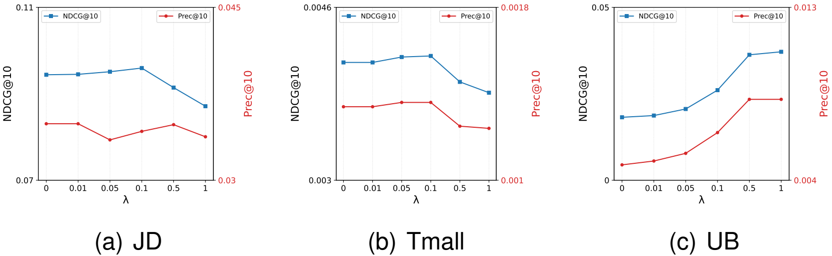


Figure: Recommendation performance of our VCGAE with different values of λ on JD, Tmall, and UB.

Conclusion

- We propose a novel model called VCGAE which combines the merits of variational autoencoder (VAE) and graph neural networks (GNNs) to address the problem of multi-behavior recommendation.
- We develop a behavior transition network that is capable of capturing the complex transition relationships that exist among different types of behaviors. This module enables us to learn about users' personal preferences based on their behavior patterns.
- We design a behavior contrastive regularization module that helps extract different correlations between users' auxiliary behaviors and their target behavior. By doing so, we are able to better understand the semantic strength of different behaviors and how they relate to each other.

Future Work

- We will consider extending our VCGAE by incorporating some side information, such as items' categories and users' social networks to further improve the recommendation accuracy.
- We plan to explore incorporating temporal dynamics into our model to capture evolving user preferences more effectively.

Thank you!

- We thank the support of National Natural Science Foundation of China No. 62172283, No. 62272315, and No. 61836005.
- We express our gratitude to Mr. Zinan Lin for his valuable assistance and insightful discussions.

Chen, C., Ma, W., Zhang, M., Wang, Z., He, X., Wang, C., Liu, Y., and Ma, S. (2021).
Graph heterogeneous multi-relational recommendation.
In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, number 5, pages 3958–3966.

Chen, C., Zhang, M., Zhang, Y., Ma, W., Liu, Y., and Ma, S. (2020).
Efficient heterogeneous collaborative filtering without negative sampling for recommendation.
In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, number 01, pages 19–26.

Collins, A., Tkaczyk, D., Aizawa, A., and Beel, J. (2018).
Position bias in recommender systems for digital libraries.
In *International Conference on Information*, pages 335–344. Springer.

Dallmann, A., Zoller, D., and Hotho, A. (2021).
A case study on sampling strategies for evaluating neural sequential item recommendation models.
In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 505–514.

Ding, Y., Shi, Y., Chen, B., Lin, C., Lu, H., Li, J., Tang, R., and Wang, D. (2021).
Semi-deterministic and contrastive variational graph autoencoder for recommendation.
In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 382–391.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., and Wang, M. (2020).
LightGCN: Simplifying and powering graph convolution network for recommendation.
In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 639–648.

Jin, B., Gao, C., He, X., Jin, D., and Li, Y. (2020).
Multi-behavior recommendation with graph convolutional networks.
In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 659–668.

Kingma, D. P. and Welling, M. (2013).
Auto-encoding variational bayes.
arXiv preprint arXiv:1312.6114.

Kipf, T. N. and Welling, M. (2016).
Variational graph auto-encoders.
arXiv preprint arXiv:1611.07308.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. (2018).
Variational autoencoders for collaborative filtering.
In *Proceedings of the 2018 World Wide Web Conference*, pages 689–698.

Ma, W., Chen, X., Pan, W., and Ming, Z. (2022).
VAE++: Variational autoencoder for heterogeneous one-class collaborative filtering.
In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, pages 666–674.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014).
Stochastic backpropagation and approximate inference in deep generative models.
In *Proceedings of the 35th International Conference on Machine Learning*, pages 1278–1286.

Wang, X., He, X., Wang, M., Feng, F., and Chua, T.-S. (2019).
Neural graph collaborative filtering.
In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–174.