BVAE: Behavior-aware Variational Autoencoder for Multi-Behavior Multi-Task Recommendation

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

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

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

Problem Definition

Multi-behavior Multi-task Recommendation (MMR)

- Input:
  - users set $\mathcal{U} = \{u\} = \{1, 2, \cdots, n\}$
  - items set $\mathcal{I} = \{i\} = \{1, 2, \cdots, m\}$
  - behaviors set $\mathcal{B} = \{k\} = \{1, 2, \cdots, c\}$

  Specifically, we have a set of items $\mathcal{I}_u^{\mathcal{B}_k}$ with respect to user u and behavior k, and a set of (user, item) pairs $\mathcal{R}^{\mathcal{B}_k} = \{(u, i)\}$ with respect to behavior k.
  For convenience, we use $\mathcal{B}_0$ denote the union set of all kinds of behaviors.
- Goal: to recommend a personalized ranked list of items for user u with respect to behavior $\mathcal{B}_k$, and these items have not previously interacted with the user under that behavior, i.e., $\mathcal{I} \backslash \mathcal{I}_u^{\mathcal{B}_k}$.

Challenge

1. The *efficiency* challenge. Compared with the split multiple single-behavior models, the joint model should be optimized in terms of both time complexity and space complexity.
2. The *scalability and adaptability* challenge. The addition of new behaviors always bring performance degeneration such as the negative transfer and seesaw phenomenon. Moreover, the relationship between new behavior and original behaviors may be weakly correlated or even competitive.

Related Work (1/2)

- One-Class Collaborative Filtering
    - VAE [Liang et al., 2018] is a typical AE-based method that generates raw user feedback using polynomial distribution, has robustness to some contexts that may be more suitable for solving MMR problems.
- Heterogeneous One-Class Collaborative Filtering
    - VAE++ [Ma et al., 2022] designs a target representation enhancement module and a target representation refinement module, which mitigate the challenges posed by data sparsity and behavioral heterogeneity.

Related Work (2/2)

- Multi-Task Learning Ideology
  - MMOE [Ma et al., 2018] is a typical expert sharing model that belongs to a special kind of soft sharing situation. It utilizes a softmax gating network to assemble experts learned by different tasks, which became the basis for numerous subsequent works.
  - UWL [Kendall et al., 2018] facilitates more defined tasks to receive higher weights via homoscedastic uncertainty in Bayesian modeling. This uncertainty is only task-dependent, which may be more suitable for soft parameter sharing models.

Overall of Our Solution

1. BVAE is an extension of VAE. When there is only one kind of behavior, it will degenerate into VAE.
2. We design the behavior aware semi-encoder (BASE). It learns users' behavior-aware latent representations from both global interaction and behavioral preference representation ways.
3. We design a dual-gate system: the global feature filtering (GFF) and the target feature fusion (TFF) network. It can enhance the robustness of the model.
4. We use the standard deviation of the VAE encoder as the uncertainty of the corresponding task to weigh loss (SDWL). It allows our model to automatically adjust the weight of each behavior without additional parameters, which ensures the model's flexibility.

The Framework of the BVAE

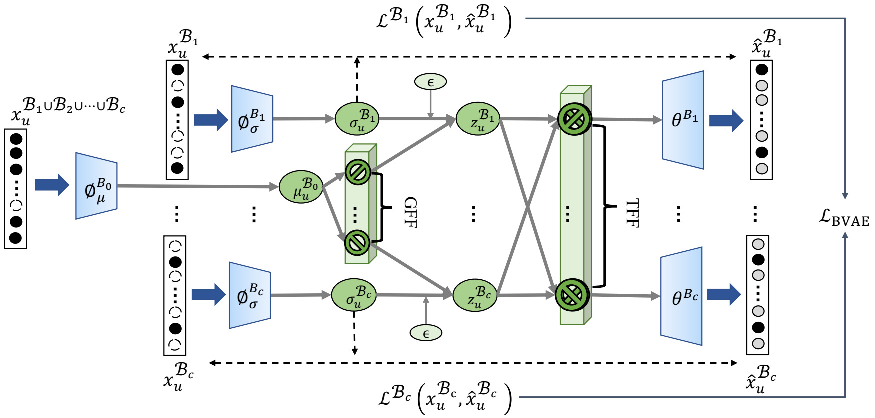

$\mathcal{L}^{\mathcal{B}_1}\left(x_u^{\mathcal{B}_1}, \hat{x}_u^{\mathcal{B}_1}\right)$
$x_u^{\mathcal{B}_1}$
$\hat{x}_u^{\mathcal{B}_1}$
$x_u^{\mathcal{B}_1 \cup \mathcal{B}_2 \cup \cdots \cup \mathcal{B}_c}$
$\phi_\mu^{\mathcal{B}_0}$
$\phi_\sigma^{\mathcal{B}_1}$
$\sigma_u^{\mathcal{B}_1}$
ϵ
$z_u^{\mathcal{B}_1}$
$\theta^{\mathcal{B}_1}$
$\mu_u^{\mathcal{B}_0}$
GFF
TFF
$\mathcal{L}_{\mathrm{BVAE}}$
$\phi_\sigma^{\mathcal{B}_c}$
$\sigma_u^{\mathcal{B}_c}$
$z_u^{\mathcal{B}_c}$
$\theta^{\mathcal{B}_c}$
$x_u^{\mathcal{B}_c}$
$\hat{x}_u^{\mathcal{B}_c}$
$\mathcal{L}^{\mathcal{B}_c}\left(x_u^{\mathcal{B}_c}, \hat{x}_u^{\mathcal{B}_c}\right)$

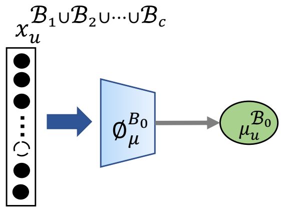
Figure: The Framework of the BVAE with *c* kinds of behaviors.

Behavior-aware Semi-Encoder (1/2)

We use $\phi_{\mu}^{\mathcal{B}_0}$ to denote the union mean vector semi-encoder, whose output the vector $\boldsymbol{\mu}_u^{\mathcal{B}_0} \in \mathbb{R}^{1\times d}$ will be use as the latent representation's mean of VAE, as follow,

$$\boldsymbol{\mu}_u^{\mathcal{B}_0} = f(\mathbf{x}_u^{\mathcal{B}_0}\mathbf{W}_{\mu_0} + \mathbf{b}_{\mu_0}), \quad (1)$$

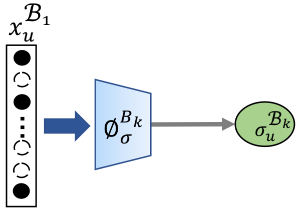
where $\mathbf{x}_u^{\mathcal{B}_0} \in \{0,1\}^{1\times m}$ is multi-hot vectors with respect to user u and union behavior. $\mathbf{W}_{\mu_0} \in \mathbb{R}^{m\times d}$ and $\mathbf{b}_{\mu_0} \in \mathbb{R}^{1\times d}$ are the weight matrix and bias vector, respectively, and $f(\cdot)$ is an activation function for the hidden layer.


$x_u^{\mathcal{B}_1 \cup \mathcal{B}_2 \cup \cdots \cup \mathcal{B}_c}$
$\phi_\mu^{B_0}$
$\mu_u^{\mathcal{B}_0}$

Behavior-aware Semi-Encoder (2/2)

Corresponding, the behavior-aware standard deviation vector semi-encoder $\phi_{\sigma}^{\mathcal{B}_k}$ can be represented as follows,

$$\boldsymbol{\sigma}_u^{\mathcal{B}_k} = \exp^{f(\mathbf{x}_u^{\mathcal{B}_k}\mathbf{W}_{\sigma_k}+\mathbf{b}_{\sigma_k})}, \qquad (2)$$



where $\mathbf{x}_u^{\mathcal{B}_k} \in \{0,1\}^{1\times m}$ is multi-hot vectors with respect to user u and behavior k, $k \in \{1,2,\cdots,c\}$. $\mathbf{W}_{\sigma_k} \in \mathbb{R}^{m\times d}$ and $\mathbf{b}_{\sigma_k} \in \mathbb{R}^{1\times d}$ are the weight matrices and bias vectors, respectively, and $f(\cdot)$ is an activation function for the hidden layer.

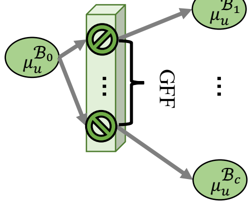
Global Feature Filtering Network

For each behavior $\mathcal{B}_k$, the mean of the latent representation $\boldsymbol{\mu}_u^{\mathcal{B}_k} \in \mathbb{R}^{1\times d}$ needs to be filtered through corresponding filtering gate g_k, as following,

$$g_k = softmax(\boldsymbol{\mu}_u^{\mathcal{B}_0}\mathbf{W}_{g_k} + b_{g_k}), \tag{3}$$

$$\boldsymbol{\mu}_u^{\mathcal{B}_k} = g_k \otimes \boldsymbol{\mu}_u^{\mathcal{B}_0}, \tag{4}$$

where $\mathbf{W}_{g_k} \in \mathbb{R}^{d\times 1}$ and $b_{g_k} \in \mathbb{R}$ are the weight and bias of filtering gate, respectively, and $\otimes$ is the element-wise product.

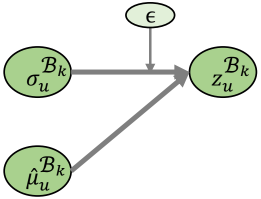


Latent Representation

Based on VAE's structure, we apply the reparameterization trick [Kingma and Welling, 2013, Rezende et al., 2014] to get the latent variable $\mathbf{z}_u^{\mathcal{B}_k}$ though a normal distribution ϵ as follows,

$$\epsilon \sim \mathcal{N}(\mathbf{0}, \mathrm{diag}(\mathbf{1})), \tag{5}$$

$$\mathbf{z}_u^{\mathcal{B}_k} = \boldsymbol{\mu}_u^k + \epsilon \otimes \boldsymbol{\sigma}_u^{\mathcal{B}_k}, \tag{6}$$



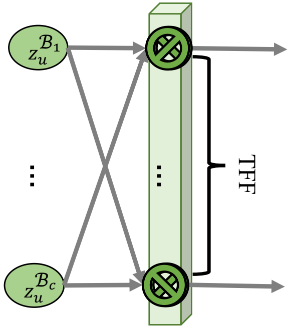
Target Feature Fusion Network

Similarly, the enhanced latent variable $\hat{\mathbf{z}}_u^{\mathcal{B}_k} \in \mathbb{R}^{1\times d}$ is a weighted sum of all latent variables, as following:

$$\hat{\mathbf{z}}_u^{\mathcal{B}_k} = \sum_{k'=1}^{c} G_{k,k'} \otimes \mathbf{z}_u^{\mathcal{B}_{k'}}, \tag{7}$$

$$G_{k,k'} = softmax(\mathbf{z}_u^{\mathcal{B}_{k'}} \mathbf{W}_{G_k} + b_{G_k}), \tag{8}$$

where $G_{k,k'}$ denotes Target feature fusion gate between any two behaviors k and k'. The weight matrices and the bias vector, denoted as $\mathbf{W}_{G_k} \in \mathbb{R}^{d\times 1}$ and $b_{G_k} \in \mathbb{R}$, are only related to the target behavior k.

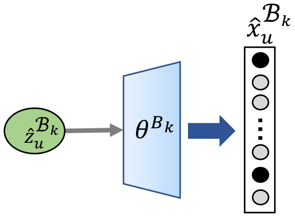


Behavior-aware Decoder

The behavior-aware decoder $\theta_u^{\mathcal{B}_k}$ consists of two parts: It inputs the enhanced latent variable $\hat{\mathbf{z}}_u^{\mathcal{B}_k}$ and outputs the probability distribution $\pi(\hat{\mathbf{z}}_u^{\mathcal{B}_k})$ through an MLP $f_\theta^k(\cdot)$ with decoder parameters θ and compress by the softmax function at first. Then reconstructs a vector $\hat{\mathbf{x}}_u^{\mathcal{B}_k}$ by sampling $N_u^{\mathcal{B}_k}$ times from the probability distribution which reflects the degree of user u prefer to behave k for each item, as follows,

$$\pi(\hat{\mathbf{z}}_u^{\mathcal{B}_k}) = \text{softmax}(f_\theta^k(\hat{\mathbf{z}}_u^{\mathcal{B}_k})), \tag{9}$$

$$\hat{\mathbf{x}}_u^{\mathcal{B}_k} \sim \text{Multi}(N_u^{\mathcal{B}_k}, \pi(\hat{\mathbf{z}}_u^{\mathcal{B}_k})), \tag{10}$$



Objective Function (1/2)

At first, the Kullback-Leibler (KL) loss. We use the KL divergence between the variational posterior $q_\phi(\mathbf{z}_u^{\mathcal{B}_k}|\mathbf{x}_u^{\mathcal{B}_k})$ and the prior $p(\mathbf{z}_u^{\mathcal{B}_k})$ to be the loss function. The KL loss can be seen as a regularization, which can be formulated as follows,

$$\mathcal{L}_{\mathrm{KL}}(\mathbf{z}_u^{\mathcal{B}_k}) = \mathrm{KL}(q_\phi(\mathbf{z}_u^{\mathcal{B}_k}|\mathbf{x}_u^{\mathcal{B}_k})||p(\mathbf{z}_u^{\mathcal{B}_k})), \tag{11}$$

Next, the reconstruction loss. For each behavior k, the reconstructed vector $\hat{\mathbf{x}}_u^{\mathcal{B}_k}$ needs to be as close as possible to the input vector $\mathbf{x}_u^{\mathcal{B}_k}$. Therefore, the reconstruction loss of behavior k can be formulated as follows,

$$\mathcal{L}(\hat{\mathbf{x}}_u^{\mathcal{B}_k}, \mathbf{x}_u^{\mathcal{B}_k}) \equiv \mathbb{E}_{q_\phi(\mathbf{z}_u^{\mathcal{B}_k}|\mathbf{x}_u^{\mathcal{B}_k})}[\log p_\theta(\mathbf{x}_u^{\mathcal{B}_k}|\hat{\mathbf{z}}_u^{\mathcal{B}_k})], \tag{12}$$

Objective Function (2/2)

Let $\beta \in [0, 1]$ to denote the parameter to weigh the regularization. The loss function for each individual single behavior is as follows,

$$\mathcal{L}_{\mathcal{B}_\text{k}} = \mathcal{L}(\hat{\mathbf{x}}_u^{\mathcal{B}_k}, \mathbf{x}_u^{\mathcal{B}_k}) - \beta \mathcal{L}_\text{KL}(\mathbf{z}_u^{\mathcal{B}_k}), \tag{13}$$

At last, inspired by UWL [Kendall et al., 2018], we use the standard deviation vector as a task-dependent uncertainty, weighting all the single behavior loss function, called SDWL in short. Those behaviors with low variance, i.e., with more explicit meaning, exerts a more significant influence in guiding the overall training process. After SDWL, the overall loss function of BVAE the weighted sum of single behavior loss functions, which can be formulated as follows,

$$\mathcal{L}_\text{BVAE} = \sum_{k=1}^{c} \left(\frac{1}{2{\sigma_u^{\mathcal{B}_k}}^2} \mathcal{L}_{\mathcal{B}_\text{k}} + log \sigma_u^{\mathcal{B}_k} \right). \tag{14}$$

Research Questions

- **RQ1.** How does our BVAE perform compared with the related state-of-the-art recommendation methods?
- **RQ2.** How do the components of our BVAE, such as BASE, TFF, GFF, and SDWL, affect the overall performance?
- **RQ3.** How do the hyperparameters of our BVAE, such as the dimensionality of modules and the quantity of recommended items, affect the evaluation result?
- **RQ4.** To what extent does our BVAE demonstrate adaptability to varying quantities and distributions of behavioral feedback?
- The processed datasets, source code, and scripts necessary to reproduce the results can be available at `https://github.com/WitnessForest/BVAE`.

Datasets(1/2)

- Jdata 2019 (JD) [1]
- UserBehavior (UB) [2]
- We process the dataset as follows :
    - (i) *Data deduplication.* We only retain the initial pair for the same (user, item, behavior) tuples and delete subsequent repeated interactions.
    - (ii) *Remove items* that have been purchased fewer than 10 times in UB and 20 times in JD.
    - (iii) *Remove sessions* containing fewer than 5 purchase records.
    - (iv) *Split the dataset into training, validation, and test sets.* Sorting all records according to the timestamp in ascending order. Then take the data front of 80% timeline as the training set, 80%-90% as the validation set, and the data after 90% as the test set.
    - (v)*Remove users* who interacted in the validation and test sets, but did not purchase an item in the training set.

[1] https://jdata.jd.com/html/detail.html?id=8

[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=649

Datasets(2/2)

Table: Statistics of two processed datasets, including the whole set with its density, the training set (tr.), the validation set (val.), and the test set (te.) with their different users and items distribution.

Statistics	JD	Density	JD (tr.)	JD (val.)	JD (te.)	UB[3]	Density	UB (tr.)	UB (val.)	UB (te.)
#Users	10,690	–	10,690	5,435	4,770	20,443	–	20,443	15,899	16,161
#Items	13,465	–	12,820	6,792	6,072	30,947	–	30,734	21,984	21,943
#Purchase	71,872	0.50‰	60,967	5,892	5,013	133,708	0.21‰	107,489	13,088	13,131
#Click	254,003	1.76‰	217,977	20,059	15,967	632,029	1.00‰	511,020	60,014	60,995
#Favorite	9,289	0.06‰	7,380	955	954	27,745	0.04‰	22,270	2,748	2,727
#Cart	8,343	0.06‰	0	1,818	6,525	84,244	0.13‰	68,207	8,191	7,846

[3]The original paper had a clerical error at items number and purchase number of the UB dataset, where the data of the whole set was misspelled as the data of the training set, which has been fixed in this slide.

Evaluation Metrics

- We use four kinds of metrics to evaluate the final personalized top-K items ranking list, i.e., precision (Prec@K), recall (Rec@K), normalized discounted cumulative gain (NDCG@K), and hit rate (HR@K) [Valcarce et al., 2018].
- Since users are usually interested in only a few items that can be displayed on the page, the experimental results will be reported mainly for $K = 5$.
- All optimal values for the aforementioned parameters are determined based on the NDCG@5 metric on the validation set.

Baselines(1/2)

- OCCF algorithms:
  - NGCF[Wang et al., 2019] is a typical graph-based OCCF method that applies graph convolutional neural networks to recommender system.
  - VAE[Liang et al., 2018] is an autoencoder-based OCCF method. It is a generative model that reconstructs user feedback using polynomial distribution. It has two versions, i.e., uses the target behavior feedback (T.) or the union behavior feedback (U.) as input. VAE (T.) can be seen as a particular case of our BVAE when there is only one behavior. VAE (U.) can be seen as a strategy that OCCF algorithms adopt to exploit multi-behavior.

Baselines(1/2)

- HOCCF algorithms:
  - EHCF[Chen et al., 2020] is an MLP-based HOCCF method. It models the complex relationships between behaviors and employs an efficient optimization method based on the entire quantity of items.
  - VAE++[Ma et al., 2022] is a VAE-based HOCCF method. It designs the target representation enhancement and target representation refinement modules to join the information learned by the target behavior and auxiliary behaviors.
- MTL algorithms:
  - MMOE[Ma et al., 2018] is one of the most typical MTL method. It sets up multiple learnable experts and combines their knowledge for different tasks through different task gates.
  - CGC[Tang et al., 2020] is a single-layer case of PLE [Tang et al., 2020]. Its major difference from MMOE is distinguishes between task-specific experts and shared experts.

Parameter Configurations

- To ensure a just and unbiased comparison, all the dimension of the latent factors are fixed to $d = 100$.
- For VAE, we follow the settings in [Liang et al., 2018], adopt a structure with 1 hidden layer and a batch size of 500. To mitigate the risk of overfitting, we set the dropout ratio ρ to 0.5. Furthermore, the learning rate is chosen from the set $\{0.0001, 0.001, 0.01\}$ and optimized using mini-batch Adam.
- For VAE++ and our BVAE, the parameter settings are consistent with VAE.
- For NGCF and EHCF, the parameters are selected following the respective papers.

Performance Comparison (RQ1) (1/5)

- The final reported results are the average and variance of three runs on the test set using the optimal parameters.
- We bold the best results and mark the second-best results with an underline.
- The "$*$" denotes the p-value of the significance test between our BVAE and the second-best is $p < 0.05$.
- In addition, from the distribution of data and the importance of behaviors, we believe that clicks and purchases are the major user behaviors and thus mark them in bold.

Performance Comparison (RQ1) (2/5)

Table: Recommendation performance of our BVAE.

Dataset		UB				JD			
Behavior	Method	Prec@5	Rec@5	NDCG@5	HR@5	Prec@5	Rec@5	NDCG@5	HR@5
#Purchase	NGCF	0.0023±0.0001	0.0070±0.0003	0.0049±0.0001	0.0108±0.0004	0.0335±0.0003	0.0721±0.0001	0.0588±0.0001	0.1086±0.0009
	VAE (T.)	0.0031±0.0001	0.0094±0.0004	0.0069±0.0002	0.0140±0.0005	0.0327±0.0008	0.0792±0.0001	0.0649±0.0005	0.1096±0.0016
	VAE (U.)	0.0028±0.0001	0.0081±0.0003	0.0057±0.0002	0.0129±0.0005	0.0296±0.0006	0.0684±0.0018	0.0544±0.0008	0.1002±0.0029
	EHCF	0.0069±0.0002	0.0186±0.0007	0.0140±0.0005	0.0313±0.0009	0.0364±0.0002	0.0853±0.0015	0.0675±0.0010	0.1170±0.0019
	VAE++	0.0059±0.0000	0.0164±0.0006	0.0128±0.0000	0.0268±0.0001	0.0369±0.0006	0.0914±0.0003	0.0730±0.0006	0.1247±0.0018
	BVAE	**0.0074**±0.0003	***0.0206**±0.0008	***0.0161**±0.0005	**0.0328**±0.0011	***0.0388**±0.0003	**0.0916**±0.0002	***0.0743**±0.0002	**0.1289**±0.0024
#Click	NGCF	0.0049±0.0001	0.0075±0.0001	0.0072±0.0001	0.0237±0.0004	0.0304±0.0004	0.0440±0.0005	0.0431±0.0005	0.1126±0.0014
	VAE (T.)	0.0067±0.0001	0.0093±0.0001	0.0092±0.0004	0.0307±0.0007	0.0331±0.0004	0.0505±0.0005	0.0480±0.0002	0.1201±0.0012
	VAE (U.)	0.0059±0.0001	0.0077±0.0003	0.0082±0.0001	0.0275±0.0008	0.0327±0.0003	0.0498±0.0005	0.0480±0.0006	0.1203±0.0019
	EHCF	0.0040±0.0002	0.0050±0.0002	0.0052±0.0003	0.0185±0.0012	0.0245±0.0004	0.0375±0.0008	0.0353±0.0004	0.0853±0.0020
	VAE++	0.0090±0.0001	0.0141±0.0000	0.0142±0.0002	0.0422±0.0004	0.0333±0.0004	0.0517±0.0005	0.0494±0.0003	0.1230±0.0014
	BVAE	***0.0109**±0.0003	***0.0189**±0.0002	***0.0181**±0.0001	***0.0510**±0.0016	***0.0375**±0.0004	***0.0575**±0.0016	***0.0551**±0.0006	***0.1367**±0.0015
#Favourite	NGCF	0.0010±0.0002	0.0028±0.0004	0.0023±0.0001	0.0048±0.0010	0.0188±0.0004	0.0366±0.0012	0.0306±0.0005	0.0552±0.0011
	VAE (T.)	0.0031±0.0001	0.0094±0.0004	0.0048±0.0007	0.0097±0.0000	0.0390±0.0006	0.0767±0.0018	0.0559±0.0011	0.1220±0.0022
	VAE (U.)	0.0029±0.0002	0.0080±0.0007	0.0055±0.0005	0.0131±0.0010	0.0313±0.0011	0.0621±0.0046	0.0523±0.0022	0.1030±0.0060
	EHCF	0.0029±0.0000	0.0094±0.0003	0.0066±0.0004	0.0143±0.0003	0.0348±0.0008	0.0722±0.0048	0.0577±0.0004	0.1142±0.0033
	VAE++	0.0033±0.0004	0.0110±0.0018	0.0085±0.0014	0.0161±0.0019	0.0490±0.0021	0.1038±0.0046	0.0819±0.0008	0.1515±0.0083
	BVAE	**0.0036**±0.0002	**0.0119**±0.0005	**0.0097**±0.0007	**0.0180**±0.0010	**0.0513**±0.0008	**0.1077**±0.0011	***0.0859**±0.0002	**0.1523**±0.0039
#Cart	NGCF	0.0010±0.0000	0.0037±0.0001	0.0026±0.0001	0.0049±0.0002	–	–	–	–
	VAE (T.)	0.0016±0.0004	0.0048±0.0016	0.0035±0.0009	0.0082±0.0021	–	–	–	–
	VAE (U.)	0.0022±0.0001	0.0069±0.0003	0.0048±0.0003	0.0106±0.0003	–	–	–	–
	EHCF	0.0022±0.0005	0.0065±0.0016	0.0044±0.0009	0.0107±0.0022	–	–	–	–
	VAE++	0.0034±0.0002	0.0105±0.0005	0.0082±0.0003	0.0159±0.0006	–	–	–	–
	BVAE	**0.0036**±0.0001	**0.0114**±0.0007	**0.0086**±0.0001	**0.0171**±0.0009	–	–	–	–

Performance Comparison (RQ1) (3/5)

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

- In most cases, VAE (T.) and VAE (U.) perform better than NGCF, especially on sparse data. This shows the strong competitiveness of the VAE-based approach, especially it consumes less complexity.
- In most cases, VAE (T.) performs better than VAE (U.). It shows the significance to study the HOCCF and MMR problems because we cannot simply process multi-behavior data by inputting their union set.

Performance Comparison (RQ1) (4/5)

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

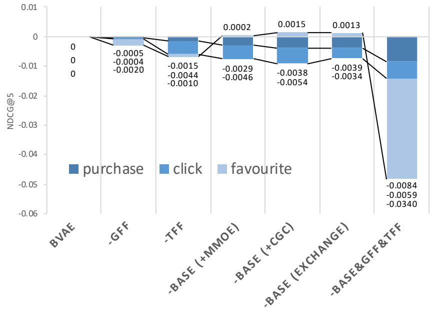
- EHCF performs better than VAE++ on the purchase behavior of the UB dataset. However, it leads to poor performance in other behaviors due to its prior setting of the sequential behavioral order. This demonstrates the limitations of the multi-behavior model based on the concatenated structure.
- In most cases, the HOCCF methods outperformance the OCCF methods. It suggests that utilizing multi-behavior feedback within the union model facilitates more accurate learning of users' actual preferences. Meanwhile, it also cautions that unreasonable structural design may cause the target task to be disturbed and degraded.

Performance Comparison (RQ1) (5/5)

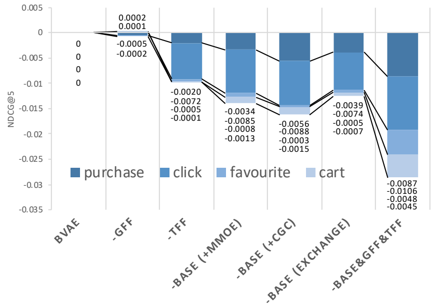
From the perspective of the MMR problem, we can have the following observations:

- Our BVAE achieves the best performance with respect to all behaviors on both JD and UB datasets compared with other baselines. In particular, there is a significant gap between our BVAE and the second-best baseline with respect to the most important and primary user behaviors, i.e., purchase and click. It indicates that our BVAE is generalizable and effective.
- Note that other methods cannot generate recommendation lists for different behaviors in a joint training. Our BVAE also has the advantage of addressing multi-behavior feedback and multi-task prediction.

Ablation Study (RQ2) (1/4)



(a) JD



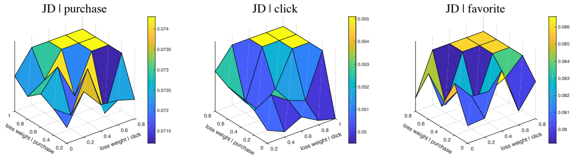
(b) UB

Figure: Variation of recommendation performance of our BVAE by removing different components, i.e., target representation enhancement (BASE), global feature filtering network (GFF), target fusion network (TFF), and BASE & GFF & TFF, respectively, for ablation studies on JD and UB.

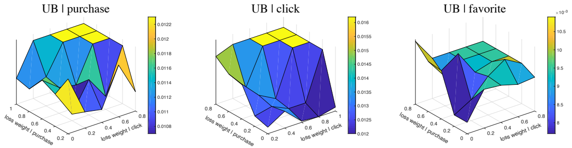
Ablation Study (RQ2) (2/4)

- Except for a slight increase in individual indicators after "-BASE" and "-GFF", most metrics dropped after removing these modules, especially after removing all three components at the same time.
- The impact of the GFF module is more significant on behaviors with sparser data, e.g., favorite and cart, while the TFF and BASE have a greater impact on behaviors with dense data.
- The MMOE, CGC, or "-BASE (Exchange)" perform slightly better than the BASE on the JD-favorite dataset, but drop more severely on other behavior and the UB dataset. Among them, the "-BASE (+CGC)" performs slightly worse than "BASE (+MMOE)" because the number of experts assigned to only 1 for each behavior may result in an inapposite allocation of share and behavior-specific experts. However, with the same parameter amount limit, our BASE module has the best comprehensive performance.

Ablation Study (RQ2) (3/4)



(a) NDCG@5 of BVAE on JD dataset



(b) NDCG@5 of BVAE on UB dataset

Figure: Recommendation performance of our BVAE by removing the SDWL.

Ablation Study (RQ2) (4/4)

Note that we omit the zero-weight result of the corresponding behavior because it leads the final evaluation close to zero. The part where the sum of the purchase loss weight and the click loss weight exceeds 1.0 are shown as the result of our BVAE for comparison purposes.

- Taking some combination of weights manually are slightly better than SDWL on the recommendation performance of favorite feedback. However, the SDWL performs better from the perspective of the joint effect of all actions.
- The results of manual parameter tuning are haphazard and irregular, which makes it difficult to find a suitable parameter for practical applications. In particular, the weight parameter ratio between behaviors may not be of the same order of magnitude, and the increase in the number of optional weight parameters will lead to a power increase in the time complexity of the model training. These validate the necessity of our SDWL modules.

Hyperparameter Sensitivity (RQ3) (1/4)

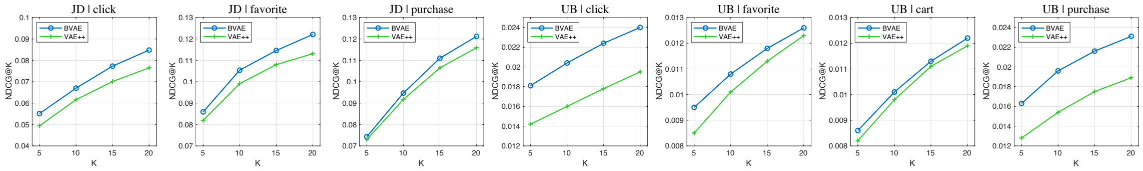


Figure: Recommendation performance of our BVAE (indicated by the blue line and circle symbols) and VAE++ (indicated by the green line and plus symbols) with different numbers of recommended items, which concern three behaviors on the JD dataset and four behaviors on the UB dataset.

Hyperparameter Sensitivity (RQ3) (2/4)

- Our BVAE outperforms the state-of-the-art related method VAE++ with respect to all kinds of behaviors on two datasets.
- As the number of K rises, VAE++ and our BVAE perform better. What's more, the gap between VAE++ and our BVAE increases for most behaviors, i.e., for all behaviors on the JD dataset and cart and purchase on the UB dataset.
- In addition, for the prediction tasks of different behavior numbers c, our BVAE has the space complexity $O(c)$ while VAE++ has the space complexity $O(c^2)$. These substantiate the powerful capabilities of our BVAE to address the MMR problems.

Hyperparameter Sensitivity (RQ3) (3/4)

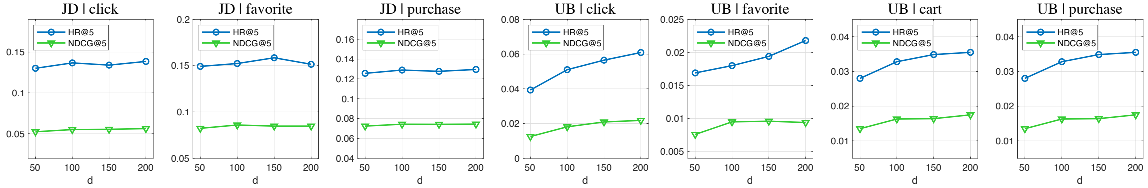
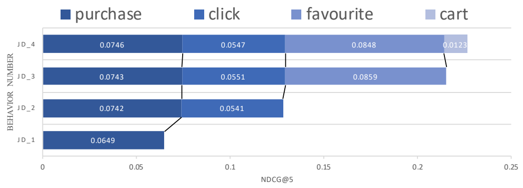


Figure: Recommendation performance of our BVAE with different numbers of latent dimensions *d*, including the evaluation metrics of NDCG@5 (indicated by the green line with triangle symbols) and HR@5 (indicated by the blue line and circle symbols), which separately concern three behaviors on JD and four behaviors on UB.

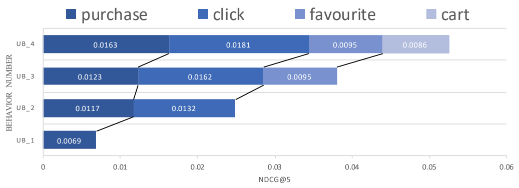
Hyperparameter Sensitivity (RQ3) (4/4)

- For the smaller dataset JD, the effect of the dimensionality change is not significant. In addition, assigning a large dimension to some very sparse data may decrease the recommendation performance due to overfitting.
- For the larger dataset UB, the recommendation performance is positively correlated with the latent dimension d.
- These showcase the possibilities of assigning the latent dimensions according to the data volume for each behavior feedback. Instead of assigning uniform parameters to all behaviors, trying different parameter assignments in input-related aspects such as dimensions can optimize memory utilization without compromising the quality of the overall recommendation.

Adaptability Study (RQ4) (1/2)



(a) NDCG@5 of BVAE on JD dataset



(b) NDCG@5 of BVAE on UB dataset

Figure: Recommendation performance of our BVAE by extending it with the different numbers of behavior types on two datasets. New behaviors are jointed in the order of ”purchase-click-favorite-cart”, considering their priority.

Adaptability Study (RQ4) (2/2)

- Both our BVAE performance becomes better when the click feedback, i.e., behavior with abundant information but noise, and the favorite feedback, i.e., behavior with useful knowledge but sparse, join to model user preferences. This shows that our model is extendable to learning the knowledge contained in other behaviors while offsetting others' weaknesses.
- The results on the UB dataset show that the cart feedback, i.e., behavior with helpful information but difficult to obtain, can significantly enhance the model.
- The results on the JD dataset showed that the empty single-behavior training set slightly degrades the recommendation performance for some behaviors, e.g., click and favorite, while not impact on the more explicit behavior, e.g., purchase. This showcases the ability of our model to be robustly adaptive in the face of extreme conditions.

Conclusion

- We propose a novel recommendation framework called BVAE, to fill the gap in the MMR problem.
- In BVAE, the BASE, GFF, and TFF modules extend the VAE to fit the MMR problem in terms of network structure. The SDWL module simplifies the loss minimization that avoids manual parameter tuning when the numbers of behaviors and tasks increase.
- Our BVAE thus has the adaptability to any amount of behaviors with different distributions. The validity of our design is confirmed via extensive empirical studies on two widely used e-commerce datasets.

Future Work

- We will consider introducing sequential information (e.g., timestamps [Zhang et al., 2022]) into our BVAE to accurately model users' dynamic preferences.
- We will explore the application potential of our BVAE in cross-domain [Zang et al., 2022] or multi-scenario [Xu et al., 2023] recommendation.
- Limited by the structure of CF autoencoders and to avoid the problem of an excessive number of projects during actual deployment, it may be necessary to combine with sampling or other methods.

Thank you!

- We thank the anonymous reviewers for constructive and expertcomments, and the support of National Natural Science Foundation of China No. 62172283, No. 62272315, and No. 61836005.
- We express our sincere appreciation to Mr. Weixin Li, for his helpful discussions and assistance.

Chen, C., Zhang, M., Ma, W., Zhang, Y., 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*, AAAI'20, pages 19–26.

Kendall, A., Gal, Y., and Cipolla, R. (2018).
Multi-task learning using uncertainty to weigh losses for scene geometry and semantics.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'18, pages 7482–7491.

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

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. (2018).
Variational autoencoders for collaborative filtering.
In *Proceedings of the 27th International Conference on World Wide Web*, WWW'18, pages 689–698.

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. H. (2018).
Modeling task relationships in multi-task learning with multi-gate mixture-of-experts.
In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD'18, pages 1930–1939.

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*, WSDM'22, 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 31st International Conference on Machine Learning*, ICML'14, pages 1278–1286.

Tang, H., Liu, J., Zhao, M., and Gong, X. (2020).
Progressive layered extraction (ple): A novel multi-task learning model for personalized recommendations.
In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys'20, pages 269–278.

Valcarce, D., Bellogín, A., Parapar, J., and Castells, P. (2018).
On the robustness and discriminative power of information retrieval metrics for top-n recommendation.
In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys'18, pages 260–268.

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*, SIGIR'19, pages 165–174.

Xu, S., Li, L., Yao, Y., Chen, Z., Wu, H., Lu, Q., and Tong, H. (2023).
Musenet: Multi-scenario learning for repeat-aware personalized recommendation.
In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, WSDM'23, pages 517–525.

Zang, T., Zhu, Y., Liu, H., Zhang, R., and Yu, J. (2022).
A survey on cross-domain recommendation: taxonomies, methods, and future directions.
ACM Transactions on Information Systems, 41(2):1–39.

Zhang, M., Wu, S., Yu, X., Liu, Q., and Wang, L. (2022).
Dynamic graph neural networks for sequential recommendation.
IEEE Transactions on Knowledge and Data Engineering.