Large-scale Comb-K Recommendation

Houye Ji
Beijing University of Posts and Telecommunications
Beijing, China
jhy1993@bupt.edu.cn

Junxiong Zhu
Alibaba Group
Hangzhou, China
xike.zjx@taobao.com

Chuan Shi*
Beijing University of Posts and Telecommunications
Beijing, China
shichuan@bupt.edu.cn

Xiao Wang
Bai Wang
xiaowang@bupt.edu.cn
wangbai@bupt.edu.cn
Beijing University of Posts and Telecommunications
Beijing, China

Chaoyu Zhang
Zixuan Zhu
chaoyu.zcy@alibaba-inc.com
tzuhsuan.thc@alibaba-inc.com
Alibaba Group
Hangzhou, China

Feng Zhang
Yanghua Li
feichen.zf@taobao.com
yichen.lyh@taobao.com
Alibaba Group
Hangzhou, China

ABSTRACT

Promotion recommendation, as a new recommendation paradigm in recent years, plays an important role in stimulating the purchase desire of users and maximizing the total revenue. Different from previous recommendations (e.g., item/group recommendation), promotion recommendation aims to select a set of K items based on *all user preferences* in selection phase and maximize the total revenue in delivery phase. Although these two phases are closely related with each other, existing methods usually focus on item selection in selection phase, largely ignoring the delivery phase and leading to sub-optimal performance. To solve the promotion recommendation problem, we propose the comb-K recommendation model, a constrained combinatorial optimization model which seamlessly integrates the selection phase and delivery phase with delicately designed constraints. When selecting K items, the comb-K recommendation is able to simultaneously search the optimal combination of item selection and delivery with the full consideration of all user preferences. Specifically, we propose a novel heterogeneous graph convolutional network to estimate user preference and propose the user-level comb-K recommendation model through solving a binary combination optimization problem. In order to handle combination explosion for large-scale users, we furtherly cluster massive users into limited groups and present a group-level comb-K recommendation model in which a novel heterogeneous graph pooling network is proposed to perform user clustering and estimate group preference. In addition, considering the "long tail" phenomenon in e-commerce, we design a restricted neighbor heuristic search to accelerate the solving process. Extensive experiments on four datasets demonstrate the superiority of comb-K model for large-scale promotion recommendation. On billion-scale data, when clustering 2.5×10^7 users into 10^3 groups, our model is able to preserve 98.7% personalized preferences in group-level and significantly improves the Total Click and Hit Ratio by 9.35% and 7.14%, respectively.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

graph neural networks, recommendation, graph mining

ACM Reference Format:
Houye Ji, Junxiong Zhu, Chuan Shi, Xiao Wang, Bai Wang, Chaoyu Zhang, Zixuan Zhu, Feng Zhang, and Yanghua Li. 2018. Large-scale Comb-K Recommendation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In the era of information explosion, the recommender system has become the most effective way to help users to discover what they are interested in enormous data. Many diverse recommendation paradigms have emerged, such as item recommendation [10], social recommendation [5], group recommendation [2], and intent recommendation [4, 33]. These recommendation paradigms focus on recommending (or ranking) preferred items to *each user* based on his personalized preference, widely known as the top-K recommendation.

With the prosperous development of e-commerce, a new recommendation paradigm, promotion recommendation, has accompanied and attracted enormous attention. The promotion recommendation selects and delivers limited promotional items with lightning deals or limited-time discounts for *all users*, stimulating the purchase desire of users and improving the total revenue. Figure 1 shows a promotion recommendation example in Amazon. A typical promotion recommendation mainly consists of two phases: (1) Selection Phase. During this phase, a selection model selects K promotional items (termed as the *K-set*) via fully considering

*Corresponding author

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

all user preferences, satisfying the requirements of all users in most cases. (2) Delivery Phase. During this phase, a delivery model delivers different subsets of K-set to different users due to users' limited attention. These delivered items constitute the real revenue of K-set. Selection phase and delivery phase are actually interdependent and deeply influence each other. Specifically, although the delivery model tries to deliver the whole K-set to users, each user can actually only view a subset of the K-set due to limited attention (a.k.a., delivery window phenomenon).

Existing methods for promotion recommendation used in industry, such as Taobao and Amazon, only focus on item selection in selection phase, largely ignoring the delivery phase. Generally, they estimate user preferences, rank the items by their accumulated preferences on all users, and greedy select top-K items as K-set for delivery. These methods usually ignore the delivery window limitation in real applications. More importantly, they separately process item selection and delivery, without considering their mutual influence. So the estimated maximum revenue in selection phase does not lead to the real maximum revenue in delivery phase (see Section 3.3 for details).

Considering the unique characteristics of promotion recommendation, a good K-set selection strategy needs to address the following requirements.

- How to seamlessly integrate the selection phase and delivery phase. Considering the reality of the delivery window phenomenon, when selecting K-set in selection phase, we need to deeply consider whether the selected items will be successfully delivered to users and indeed generate revenue due to delivery window. Integrating the two phases as a whole may help us to select a more valuable K-set and maximize the real revenue in delivery phase.
- How to estimate the user preferences precisely, taking full advantage of complex interactions and rich attributes in the promotion scenario. Preference estimation is the cornerstone of the item selection and directly affects the total revenue. Both complex interactions and rich attributes provide valuable information from different aspects, improving the preference estimation.
- How to handle large-scale users in promotion recommendation. All user preferences should be fully considered for K item selection. However, there exist more than ten million users in mainstream promotion platform, leading to unacceptable computioanl cost. Designing an efficient and effective way to integrate massive users preferences is an urgent problem that needs to be solved.

In this paper, we are the first to thoroughly investigate the promotion recommendation problem and propose a novel comb-K recommendation model to improve the promotion revenue. Comb-K recommendation model is to solve a binary combinatorial optimization problem which considers the constraint of item selection and the constraint item delivery simultaneously, integrating the selection phase and delivery phase as a whole and achieving a preferable promotion effect. Different from traditional top-K recommendation models, comb-K recommendation is able to select a set of K items based on *all user preferences* with *the constraints*.

Figure 1: Show case of promotion recommendation in Amazon.

Specifically, user-level comb-K recommendation takes the cumulative users preferences on delivered items as the objective function, which is an approximation of the total revenue. Then, it takes all user preferences as constants, searches the optimal combination of item selection and item delivery simultaneously to maximize the objective function. Considering the rich heterogeneous information, we propose a novel heterogeneous graph convolutional network to estimate user-item preferences precisely. However, user-level comb-K recommendation cannot scale for large-scale users due to combinatorial explosion. Furthermore, we cluster massive users into limited groups and present a group-level comb-K recommendation in which a novel heterogeneous graph pooling network is proposed to perform user clustering based on their preferences and estimate group-item preference. In addition, considering the "long tail" phenomenon in e-commerce, we further design a fast strategy called restricted neighbor heuristic search to further accelerate the solving process of comb-K recommendation.

The main contributions are summarized as follows.

- We deeply investigate the problem of promotion recommendation with the full consideration of the selection phase and the delivery phase, stimulating the purchase desire of users and improving the total revenue of promotion platform.
- Considering the delivery window phenomenon, we innovatively propose the comb-K recommendation model to solve the promotion recommendation problem, a constrained combinatorial optimization problem which models the item selection and the item delivery simultaneously. We propose heterogeneous graph convolution and heterogeneous graph pooling to respectively estimate user preference and group preference, serving as the cornerstone of user-level and group-level comb-K recommendation. In addition, we design a restricted neighbor heuristic search strategy to accelerate the solving process.
- Extensive experiments on large-scale datasets show the superiority of the comb-K recommendation. The comparisons of

user-level and group-level preference demonstrate the effectiveness of heterogeneous graph pooling. When clustering 2.5×10^7 users into 10^3 groups, the proposed heterogeneous graph pooling is able to preserve 98.7% personalized preferences. On billion-scale data, group-level comb-K recommendation improves Total Click and Hit ratio by 9.35% and 7.14%, respectively. The superiority of restricted neighbor heuristic search is also confirmed on synthetic datasets.

2 RELATED WORK

2.1 Recommender System

Recommender systems mainly focus on how to recommend items to users based on their preferences, including item recommendation [10, 25], social recommendation [5], group recommendation [2], and intent recommendation [4, 33]. [10] first models item recommendation system with deep neural network and [5] further leverages social information to improve item recommendation. Several works [25, 27] adopt graph neural networks to capture structural information and further improve the item recommendation. [2] generalizes item recommendation to group recommendation which recommends top-K items to a group, considering the preference of a whole group. [4, 12, 33] aim to recommend the intent to the user, rather than the specific items. On the other hand, list-wise models [1, 8, 14] recommend a whole list to users via minimizing the ranking loss and achieve whole-list ranking optimization.

Some works [13, 23, 31] focus on designing diverse accelerate strategies for large-scale recommender systems. Fast graph encoder [31] directly applies small world graph on all items and navigates to derive recommended items. [13] reduces the candidate items based on the user clusters and accelerates the top-K item recommendation. And, [23] proposes Preference Hash which represents all the users with limited hash buckets to handle large-scale users.

2.2 Graph Neural Network

Graph neural networks (GNNs) [9, 17, 24] generalize deep learning to graph-structured data. [17] proposes GCN via a localized approximation of spectral graph convolutions. GraphSAGE [9] leverages neighbor sampling to perform inductive prediction. [18, 24, 28] design diverse aggregators to improve node embedding. Some works [4, 6, 11, 26, 30, 32, 34] extend GNNs to the heterogeneous graph with more delicate aggregators and [4, 8, 12, 20, 25, 33] leverage graph convolutional network to learn user and item embedding for diverse recommendation tasks.

In addition to graph convolution, some studies tried to extend pooling operations to graphs [7, 16, 19, 29], which aims to coarsen and reduce the size of the graph with differentiable pooling network, analogous to image downsampling in CNNs. DiffPool [29] computes soft clustering assignments of nodes from the original graph to nodes in the pooled graph. [19] leverages self-attention for hierarchical pooling, considering both node features and graph topology. [16] adopts memory mechanism to jointly learn node representations and coarsen the graph in a hierarchical manner.

Table 1: Notations and Explanations.

Notation	Explanation
$\{u_i \mid i \le M\}$	User set with size M
$\{v_j \mid j \le M\}$	Item set with size N
$\{g_p \mid p \le P\}$	Group set with size P
K	Number of selected items
W	Size of delivery window
$\mathcal{S}$	K-set containing K selected items
R	Total revenue of K-set
$\mathbf{h_i}$	Embedding of user u_i
$\mathbf{h_j}$	Embedding of item v_j
$\mathbf{z_p}$	Embedding of group g_p
$\hat{r}_{i,j}$	Preference of user u_i on item v_j
$\hat{r}_{p,j}$	Preference of group g_p on item v_j
a_p	Size of group g_p
$I_{i,p} \in \{0,1\}$	User u_i is assigned to group g_p or not
$X_j \in \{0,1\}$	Item v_j is selected or not
$Y_{i,j} \in \{0,1\}$	Item v_j is delivered to user u_i or not
$Y_{p,j} \in \{0,1\}$	Item v_j is delivered to group g_p or not

3 PROBLEM MODELING

In this section, we first model the promotion scenario as the attributed heterogeneous graph. Then, we define the promotion recommendation and introduce naive top-K based methods to solve it. Then, we propose the comb-K recommendation model in both user-level and group-level. The notations we will use throughout the article are summarized in Table 1.

3.1 Preliminary

The promotion scenario which contains complex interactions and rich attributes can be unified modeled as an attributed heterogeneous graph $\mathcal{G}$.

Definition 1. ***Attributed Heterogeneous Graph.*** *An attributed heterogeneous graph, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{F})$, where $\mathcal{V} = \mathcal{V}_U \cup \mathcal{V}_I$ is the node sets, $\mathcal{E} = \mathcal{E}_{UU} \cup \mathcal{E}_{UI}$ is the edge sets, $\mathbf{F} \in \mathbb{R}^{|\mathcal{V}| \times |F|}$ is an attribute matrix of nodes. Here $\mathcal{V}_U = \{u_i\}_{1 \le i \le M}$ and $\mathcal{V}_I = \{v_j\}_{1 \le i \le N}$ are the sets of M users and N items, respectively. $\mathcal{E}_{UU} = \langle \mathcal{V}_U, \mathcal{V}_U \rangle$ denotes User-User interaction and $\mathcal{E}_{UI} = \langle \mathcal{V}_U, \mathcal{V}_I \rangle$ denotes User-Item interaction.*

Given a promotion scenario, we define the promotion recommendation and its objective function Obj, as follows:

Definition 2. ***Promotion Recommendation.*** *Given a promotion scenario $\mathcal{G}$, promotion recommendation aims to select a set of K items $\mathcal{S} = \{v_j \mid X_j = 1\} \subset \mathcal{V}_I$ (termed as the K-set) which maximizes the total revenue R on all users $\mathcal{V}_\mathcal{U}$, as follows:*

$$\begin{aligned} Obj &= \max_X R(\mathcal{S}|\mathcal{G}; \theta), \\ s.t.\ X_j &\in \{0,1\}, \textstyle\sum_{j \le N} X_j = K, \end{aligned} \tag{1}$$

where θ is the parameters for function of generating $\mathcal{S}$ from $\mathcal{G}$, $X_j \in \{0,1\}$ is a binary selection indicator which indicates whether the item v_j is selected or not. The constraint $\sum_{j \le N} X_j = K$ restricts the size of K-set.

When selecting K items in selection phase, we can only estimate the total revenue of K items. For example, we can use the CTR[1] or user-item preference to estimate the revenue of single item and then estimate the total revenue of K items. When delivering K items in delivery phase, we can get the real revenue of K items. For example, we can use the Total Click or GMV[2] as the real revenue.

It is worth noting that the delivery model only delivers a subset of K-set to each user and only the delivered items constitute the real revenue. The unique workflow including item selection and delivery in the promotion recommendation leads to such phenomenon. We call it as the delivery window phenomenon and define the size of the subset delivered to each user as the *delivery window* W. Intuitively, the delivery window is highly related to the real revenue and potentially influences the item selection in the promotion recommendation. Considering the delivery window for items selection may improve the real revenue of K items in the delivery phase.

In the following sections, we first introduce naive top-K based methods, which ignore the delivery window limitation and separately process item selection and delivery without considering their mutual influence. Then, we introduce the comb-K based methods, which seamlessly integrate both item selection and delivery for K items selection.

3.2 Naive Top-K based Methods

In this section, we introduce two naive top-K based methods for promotion recommendation including CTR based method and preference based method.

CTR based Top-K Method. Considering that higher CTR usually implies the item is popular with users, a naive top-K based method for promotion recommendation is to rank all items based on their CTRs and selects top-K items as the K-set. It is a greedy method which maximizes the estimated revenue of K-set via finding the optimal selection indicator X_j, as follows:

$$\begin{aligned} &\max_X \textstyle\sum_{j\le N} \hat{l}_j \cdot X_j, \\ &s.t.\ X_j \in \{0,1\}, \textstyle\sum_{j\le N} X_j = K, \end{aligned} \tag{2}$$

where $\hat{l}_j$ is the constant which denotes the estimated CTR of item v_j and we use it as the estimated revenue of item v_j.

Preference based Top-K Method. Another naive top-K based method for promotion recommendation is to rank all items based on all user preferences and selects top-K items as the K-set. It explicitly considers all user preferences to maximize the estimated revenue via finding the optimal selection indicator X_j, as follows:

$$\begin{aligned} &\max_X \textstyle\sum_{i\le M, j\le N} \hat{r}_{i,j} \cdot X_j, \\ &s.t.\ X_j \in \{0,1\}, \textstyle\sum_{j\le N} X_j = K, \end{aligned} \tag{3}$$

where $\hat{r}_{i,j}$ denotes the estimated user-item preference, and we use it as the estimated revenue of item v_j on user u_i.

Both CTR based method and preference based method only consider the selection phase with the selection indicator X_j to select K items in the top-K manner, largely ignoring the reality of the delivery window and leading to sub-optimal performance.

[1]Click-through Rate (CTR) is formulated as: CTR = $\frac{\text{Number of click-throughs}}{\text{Number of impressions}} \times 100(\%)$

[2]Gross Merchandise Volume

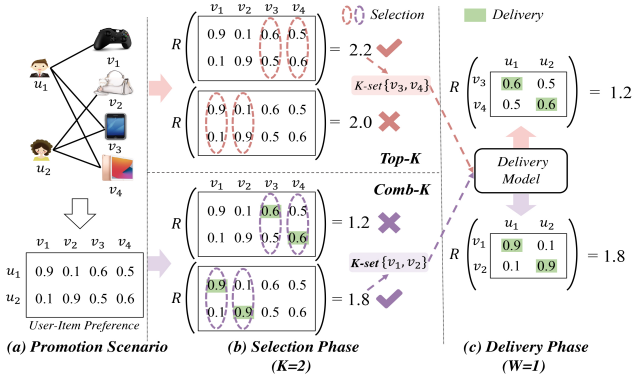


Figure 2: An illustrative example of item selection via top-K recommendation and comb-K recommendation.

3.3 User-level Comb-K Recommendation

To seamlessly integrate the selection phase and delivery phase in promotion recommendation, we propose the comb-K recommendation, a binary combinatorial optimization model with delicately designed constraints.

Comparing to naive top-K based methods, the comb-K recommendation further considers an additional delivery indicator $Y_{i,j} \in \{0,1\}$, indicating whether item v_j will be delivered to user u_i in delivery phase. The user-level comb-K recommendation aims to maximize the estimated revenue of K-set on all users via finding the optimal combination of two decision variables: item selection X_j and item delivery $Y_{i,j}$, as follows:

$$\begin{aligned} &\max_{X,Y} \textstyle\sum_{i\le M, j\le N} \hat{r}_{i,j} \cdot X_j \cdot Y_{i,j}, \\ &s.t.\ X_j \in \{0,1\},\ Y_{i,j} \in \{0,1\}, \\ &\qquad Y_{i,j} \le X_j, \\ &\textstyle\sum_{j\le N} X_j = K,\ \sum_{j\le N} Y_{i,j} = W, \end{aligned} \tag{4}$$

where the constraint $\sum_{j\le N} Y_{i,j} = W$ means each user can only view W items in K-set (a.k.a, delivery window), the constraint $Y_{i,j} \le X_j$ means only the selected item v_j will be delivered to user u_i. For example, $X_j = 1, Y_{i,j} = 0$ means although item v_j is selected in selection phase but it would not be delivered to user u_i in delivery phase.

Why the constraint of delivery window W works? In Figure 2, we give an illustrative example to show the superiority of the delivery window (a.k.a., top-K v.s. comb-K). Given a promotion scenario with 2 users $\{u_1, u_2\}$, 4 items $\{v_1, v_2, v_3, v_4\}$ and corresponding user preference $\hat{r}_{i,j}$, we show how the item selection changes with/without the constraint of delivery window. Here we assume $K = 2, W = 1$ which means each user can only view one of the two selected items.

- Without the constraint of delivery window W. Top-K based method selects $\{v_3, v_4\}$ as the K-set $\mathcal{S} = \{v_3, v_4\}$, since the set has the maximum estimated revenue 2.2. However, in delivery phase, the top-K recommendation can only select one optimal item due to delivery window $W = 1$, and thus the real revenue of K-set is actually 1.2.
- With the constraint of delivery window W. User-level comb-K recommendation selects $\{v_1, v_2\}$ as the K-set $\mathcal{S} = \{v_1, v_2\}$,

since the set has the maximum estimated revenue 1.8. More importantly, the real revenue of $\{v_1, v_2\}$ in delivery phase is still 1.8, which is consistent with the estimated revenue in delivery phase.

Although the estimated revenue of top-K based K-set (e.g., 2.2) is higher than comb-K based K-set (e.g., 1.8), but the real revenue in delivery phase is opposite. As can be seen, the real revenue of comb-K based K-set (e.g., 1.8) is higher than top-K based K-set (e.g., 1.2). The reason is that comb-K recommendation takes full consideration of the delivery window for K items selection in the selection phase, so the real revenue in delivery phase could be maximum.

In summary, the user-level comb-K recommendation shows the superiority over traditional top-K recommendation via seamlessly integrating item selection and delivery for K items selection.

3.4 Group-level Comb-K Recommendation

For large-scale users, the complexity of user-level comb-K recommendation is combinatorial explosion and unsolvable. It motivates us to reduce the complexity of comb-K recommendation and makes it scalable for large-scale users. Intuitively, different users sharing similar preferences and may belong to the same group. For example, people who like doing some sports belong to sports buff. Since the number of groups is significantly less than the number of users, if we can cluster M users into P groups ($P \ll M$) and conduct comb-K recommendation in group-level, the complexity of comb-K recommendation could be significantly reduced.

Group-level comb-K recommendation aims to maximize the estimated revenue of K-set on all groups via finding the optimal combination of two decision variables: item selection X_j and item delivery $Y_{p,j}$, as follows:

$$\begin{gathered} \max_{X,Y} \sum_{p\le P, j\le N} a_p \cdot \hat{r}_{p,j} \cdot X_j \cdot Y_{p,j}, \\ s.t.\ X_j \in \{0,1\},\ Y_{p,j} \in \{0,1\}, \\ Y_{p,j} \le X_j, \\ \sum_j X_j = K,\ \sum_{j\le N} Y_{p,j} = W, \end{gathered} \tag{5}$$

where $\hat{r}_{p,j}$ is the constant which denotes the estimated preference of group g_p on item v_j, a_p denotes the size of group g_p, $Y_{p,j}$ denotes whether item v_j will be delivered to group g_p in delivery phase, $\sum_{j\le N} Y_{p,j} = W$ denotes users in group g_p are able to view W items in delivery phase, $Y_{p,j} \le X_j$ means only the selected item v_j will be delivered to group g_p. Note that group-level comb-K recommendation not only considers group-item preference $\hat{r}_{p,j}$ but also considers the size of group a_p. It makes sense because the group with more users (the majority) should have more votes than the minority. And we can take $a_p \cdot \hat{r}_{p,j}$ as the estimated revenue of item v_j on group g_p.

Group-level comb-K recommendation also considers all users preferences in the group manner, but its complexity is significantly reduced because we only need to consider the preferences of P groups rather than M users ($P \ll M$). That is to say, group-level comb-K recommendation can be applied to large-scale users. The only remaining question for group-level comb-K recommendation is how to cluster massive users into limited groups and estimate the group-item preference.

4 COMB-K RECOMMENDATION SOLUTION

In this section, we give the solutions to both user-level and group-level comb-K recommendation. The solving process of the comb-K recommendation mainly consists of two steps: (1) Preference estimation. We estimate user preference $\hat{r}_{i,j}$ and group preference $\hat{r}_{p,j}$ via heterogeneous graph convolution and heterogeneous graph pooling, respectively. (2) Combinatorial optimization solution. We propose a fast solving strategy called restricted neighbor heuristic search to accelerate the solving process of the comb-K recommendation.

4.1 Preference Estimation

4.1.1 User Preference Estimation. User preference is the cornerstone of user-level comb-K recommendation and directly affects the quality of the K-set, so we need to precisely estimate the user preference. Considering the rich heterogeneous information in the promotion scenario, we design a heterogeneous graph convolutional network to learn user and item embedding and then estimate user preference.

Specifically, we initialize user embedding via concatenating its feature embedding, as follows:

$$\mathbf{e}_i = \Big\Vert_{f=1}^{|F|} \mathbf{e}_i^f, \tag{6}$$

where $||$ denotes the vector concatenation, $\mathbf{e}_i$ and $\mathbf{e}_i^f$ denote the initial embedding and the f-th feature embedding of user u_i, respectively. The same process can be done for the initial item embedding $\mathbf{e}_j$.

Then, we aggregate different types of neighbors and then fuse them to learn comprehensive node embedding. Specifically, given one user u_i and k_1 user-related relations $\{\Phi_1^U, \Phi_2^U, \cdots, \Phi_{k_1}^U\}$, user-related heterogeneous graph convolutional network (termed as $HeteGNN^U$) is able to get k_1 relation-specific user embeddings, as follows:

$$\mathbf{h}_i^{\Phi_1^U}, \mathbf{h}_i^{\Phi_2^U}, \cdots, \mathbf{h}_i^{\Phi_{k_1}^U} = HeteGNN^U(u_i; \Phi_1^U, \Phi_2^U, \cdots, \Phi_{k_1}^U). \tag{7}$$

Here we simply average the neighbor embeddings to get the relation-specific user embedding, as follows:

$$\mathbf{h}_i^{\Phi_{k_1}^U} = \text{Average}\left(\left\{\mathbf{e}_n \mid \forall n \in \mathcal{N}_i^{\Phi_{k_1}^U}\right\}\right), \tag{8}$$

where $\mathcal{N}_i^{\Phi_{k_1}^U}$ denotes the relation $\Phi_{k_1}^U$ based neighbors of user u_i. The final user embedding is the concatenation of multiple relation-specific embeddings, as follows:

$$\mathbf{h}_i = \mathbf{h}_i^{\Phi_1^U} || \mathbf{h}_i^{\Phi_2^U} || \cdots || \mathbf{h}_i^{\Phi_{k_1}^U}. \tag{9}$$

The same process can be done for items and the final embedding of item v_j is denoted as $\mathbf{h}_j$.

Taking user embedding $\mathbf{h}_i$ and item embedding $\mathbf{h}_j$ as inputs, we can estimate the user preference $\hat{r}_{i,j}$ via MLP_{U-I}, shown as follows:

$$\hat{r}_{i,j} = MLP_{U-I}(\mathbf{h}_i || \mathbf{h}_j). \tag{10}$$

After precisely estimating user preference $\hat{r}_{i,j}$, we can perform user-level comb-K recommendation via solving Eq. 4.

4.1.2 Group Preference Estimation. When applying comb-K recommendation to large-scale users, we need to cluster massive users into limited groups and estimate group preference, facing the following challenges: (1) Clustering evaluation. In promotion recommendation, we do not have the ground-truth of user clustering. So even if we cluster users into groups, we cannot directly evaluate the clustering results. (2) Task-specific clustering. Traditional clustering models (e.g., K-Means) are usually unsupervised models and may not suit for specific-task (e.g., promotion recommendation). How to find the optimal user clustering for promotion recommendation is still an open problem.

To tackle the above challenges, we propose a novel heterogeneous graph pooling network which clusters users into groups based on their preferences in an end-to-end manner via minimizing the recommendation loss. So the clustering result is special optimized for recommendation task and the users in the same group trend to share similar preferences. More importantly, we can directly evaluate the clustering results via comparing the performance of user-item recommendation and group-item recommendation. For example, if the AUC of group-item recommendation is close to the AUC of user-item recommendation (i.e., $\text{AUC}_{G-I}/\text{AUC}_{U-I} \approx 1$), then we can conclude the proposed heterogeneous graph pooling clusters the users who share similar preferences into the same group.

Specifically, taking M user embeddings $\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_M\}$ as inputs, the proposed heterogeneous graph pooling network is able to cluster them into P groups $\{g_1, g_2, \cdots, g_P\}$ and generate corresponding P group embeddings $\{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_P\}$, shown as follows:

$$\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_P = C(\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_M), \tag{11}$$

where C denotes neural network performing the heterogeneous graph pooling.

Intuitively, when performing user clustering, we need to consider the similarities between the users and the groups, because different users should make different contributions in different groups. Inspired by recent self-attention mechanism [16, 24], we design a self-attentive clustering model which is able to learn the user-group similarity and then updates the group embedding via attentive user aggregation. Specifically, we initialize P center embeddings $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_P\}$ which denote the center of each group and calculate the similarity between user u_i and group g_p, denoted as s_{ip}, as follows:

$$s_{ip} = \mathbf{h}_i \cdot \mathbf{v}_p^\top. \tag{12}$$

Larger s_{ip} means user u_i is closer to the center of group g_p. Then, we can obtain the probability of user u_i belonging to group g_p via softmax function,

$$w_{ip} = \frac{\exp(s_{ip})}{\sum_{p' \le P} \exp(s_{ip'})}. \tag{13}$$

With the learned w_{ip} as coefficients, we aggregate user embeddings and get the group embedding $\mathbf{z}_p$, as follows:

$$\mathbf{z}_p = \mathbf{z}'_p + \sum_{i \le M} w_{ip} \cdot \mathbf{h}_i, \tag{14}$$

where $\mathbf{z}'_p$ denotes the bias vector for group g_p. However, Eq. 14 needs to conduct sum operator over M users where is not practicable on large-scale users. Inspired by web-scale K-Means [21], we design an incremental update strategy to approximate $\sum_{i \le M} w_{ip} \cdot \mathbf{h}_i$, which aggregates batched user embeddings in a moving average manner,

$$\mathbf{var} \leftarrow (1-\gamma) \cdot \mathbf{var} + \gamma \cdot \sum_{i \in B} w_{ip} \cdot \mathbf{h}_i, \tag{15}$$

where $\mathbf{var}$ is an approximation of $\sum_{i \le M} w_{ip} \cdot \mathbf{h}_i$, B denotes mini-batch data, γ is an increment factor (e.g., 0.01). Experimentally, we find that Eq. 15 often provides satisfactory results and can be applied to group-level comb-K recommendation.

Taking group embedding $\mathbf{z}_p$ and item embedding $\mathbf{h}_j$ as inputs, we can estimate the group preference $\hat{r}_{p,j}$, as follows:

$$\hat{r}_{p,j} = MLP_{G-I}(\mathbf{z}_p || \mathbf{h}_j). \tag{16}$$

Lastly, we can get the clustering assignment $I_{i,p}$ via argmax operator, as follows:

$$I_{i,p} \leftarrow \arg\max_p (\mathbf{h}_i \cdot \mathbf{v}_p^\top). \tag{17}$$

Here $I_{i,p} = 1$ means user u_i is assigned to group g_p. It makes sense because the user should be assigned to the group with the largest similarity. After that, the size of group a_p can be obtained via $\sum_{i \le M} I_{i,p}$.

4.1.3 Loss Functions. After obtaining estimated user preference $\hat{r}_{i,j}$ and estimated group preference $\hat{r}_{p,j}$, we need to optimize the model via minimizing the loss function. The loss function of user preference $\mathcal{L}_{U-I}$ is shown as follows:

$$\mathcal{L}_{U-I} = \sum_{(i,j) \in \mathcal{D}} \left(r_{i,j} \log \hat{r}_{i,j} + (1 - r_{i,j}) \log (1 - \hat{r}_{i,j}) \right), \tag{18}$$

where $\mathcal{D}$ denotes the dataset, $r_{i,j}$ denotes the ground truth of user preference (a.k.a., whether user u_i will click item v_j).

The loss function of group preference is similar to Eq. 18. However, the ground truths of group preferences do not exist in dataset. Inspired by *variable substitution* [35], we use the ground truths of user preferences to substitute the ground truths of group preferences based on the clustering assignment $I_{i,p}$. For example, if $I_{i,p} = 1$, then $r_{p,j} = r_{i,j}$. Based on the above substitution operation, we can obtain the loss function of group-item recommendation $\mathcal{L}_{G-I}$, as follows:

$$\mathcal{L}_{G-I} = \sum_{(i,j) \in \mathcal{D}} I_{i,p} \left(r_{i,j} \log \hat{r}_{p,j} + (1 - r_{i,j}) \log (1 - \hat{r}_{p,j}) \right). \tag{19}$$

4.1.4 Optimization Strategies. To estimate user preference, we can directly minimize $\mathcal{L}_{U-I}$ and optimize the model parameters. To estimate group preference, we try two training strategies:

Pretrain & Fine-tuning. Deep neural network and clustering model are rather sensitive to initialization. To stabilize the training process, we first pretrain the user/item embedding via minimizing $\mathcal{L}_{U-I}$. Then, we minimize $\mathcal{L}_{G-I}$ to fine-tune them and learn group embedding. If we directly optimize $\mathcal{L}_{G-I}$ without pretrain, the proposed model cannot converge correctly.

Interval Training. Another training strategy is to minimize $\mathcal{L}_{U-I}$ and $\mathcal{L}_{G-I}$ intervally. Specifically, for each training step, we first optimize parameters via minimizing $\mathcal{L}_{U-I}$ and then optimize parameters via minimizing $\mathcal{L}_{G-I}$. It is actually the multi-task learning with the shared user/item embedding, so they can mutually reinforce each other and improve the performance. And experimentally, we found interval training is able to improve the effectiveness

Table 2: The statistics of the datasets.

Dataset	Split	#Users	#Items	#Samples
1-day	Training	119,988	3,888	13,217,835
	Val & Sel.	96,680	2,924	7,219,181
	Eval.	61,205	1,849	3,042,656
2-day	Training	119,988	3,888	13,217,835
	Val & Sel.	96,680	2,924	7,219,181
	Eval.	88,668	2,398	5,990,529
3-day	Training	119,988	3,888	13,217,835
	Val & Sel.	96,680	2,924	7,219,181
	Eval.	109,441	3,362	10,311,133
Large	Training	24,564,098	16,801	503,814,664
	Val & Sel.	3,357,660	7,292	63,701,395
	Eval.	22,867,517	11,696	356,650,383

of group-item prediction and generates more precise clustering assignment (details are shown in Section 5.5).

4.2 Combinatorial Optimization Solution

Taking the estimated preferences (e.g., $\hat{r}_{i,j}$ and $\hat{r}_{p,j}$) as the constants, we can solve the comb-K recommendation via finding the optimal combination of two decision variables: item selection (e.g., X_j) and item delivery (e.g., Y_{ij} and $Y_{p,j}$). We have tried to use commercial integer programming solver (e.g., Gurobi[3]) as the comb-K recommendation solver, but it is not special designed for the promotion recommendation and fails to solve the comb-K recommendation in the limited time. It inspires us to design a fast search strategy to accelerate the solving process of comb-K recommendation. Here we propose a fast search strategy called restricted neighbor heuristic search (RNHS) which is special designed for the promotion scenario and significantly faster than classical integer programming (IP) with high precision.

Recall a well-known heuristic search strategy for combinatorial optimization, called 2-opt [3], which swaps elements between initial set and candidate set until initial set achieves the maximum objective function. As discussed in Section 3.2, the K-set $\mathcal{S}$ selected by top-K based methods achieves sub-optimal performance. Here we take the K-set selected by Eq. 3 as the initial set $\mathcal{S}_{init}$ and the rest as the candidate set $\mathcal{S}_{cand} = \mathcal{V}_I - \mathcal{S}_{init}$, $|\mathcal{S}_{cand}| = N - K$. Then, we can apply 2-opt to swap elements between $\mathcal{S}_{init}$ and $\mathcal{S}_{cand}$ until initial set achieves the maximum objective function of comb-K recommendation. We take the latest initial set $\mathcal{S}_{init}$ as the K-set $\mathcal{S}$ for comb-K recommendation.

However, the previous 2-opt is still not fast enough to address the requirement of large-scale comb-K recommendation because $|\mathcal{S}_{cand}| = N - K$ remains vary large. Considering the "long tail" of items in e-commerce which means only a few items attract a great deal of consumer interest, we only need to focus on "head" items and construct a much smaller candidate set. Here we propose the restricted neighbor heuristic search, which restricts the size of the candidate set $\mathcal{S}_{cand}$ and significantly reduces the size of solution space. The proposed RNHS mainly consists of three steps: (1) rank all items based on user preference or group preference; (2) select top-K items as the initial set $\mathcal{S}_{init}$ and select the top K+1~2·K items as the candidate set $\mathcal{S}_{cand}$, $|\mathcal{S}_{cand}| = K$; (3) swap items in $\mathcal{S}_{init}$ and $\mathcal{S}_{cand}$ until maximizing the objective function of comb-K recommendation. We take the latest initial set $\mathcal{S}_{init}$ as the K-set $\mathcal{S}$ for comb-K recommendation.

5 EXPERIMENTS

We conduct experiments on large-scale promotion recommendation to evaluate the proposed method and answer the following research questions:

- **RQ1:** Does the proposed comb-K recommendation outperform top-K recommendation in the promotion scenario?
- **RQ2:** How does the delivery window affect the performance of comb-K recommendation?
- **RQ3:** How well are the user-level preferences preserved in group-level with different training strategies?
- **RQ4:** Does the proposed RNHS solve the comb-K recommendation efficiently?

5.1 Datasets

We extract four datasets with different scales from the promotion scenario in Taobao platform[4] to verify the effectiveness of the proposed comb-K recommendation. Here we split each dataset into three parts, shown as follows:

- **Training.** We train the preference estimation model (e.g., heterogeneous graph convolution and heterogeneous graph pooling) to estimate user/group preference.
- **Validation & Selection.** We test the preference estimation model and compare the effectiveness of user preference $\hat{r}_{i,j}$ and group preference $\hat{r}_{p,j}$ (**RQ3**). Then, we select a set of K items $\mathcal{S}$ via different methods (e.g., top-K v.s. comb-K).
- **Evaluation.** We evaluate the performance (e.g., total revenue) of a set of K items $\mathcal{S}$ selected by different methods (**RQ1**).

Specifically, we collect samples in 2020/04/28-2020/04/30 for training, 2020/05/01-2020/05/02 for validation & selection, and the next 1,2,3 days for evaluation, marked as **1-day**, **2-day**, and **3-day**, respectively. Meanwhile, we also extract a billion-scale dataset (marked as **Large**), which uses 2020/05/25-2020/05/31 for training, 2020/06/01-2020/06/03 for validation & selection, and 2020/06/04-2020/06/06 for evaluation. For the Large dataset, we extract 200+ features for both user and item, such as *user_ID*, *user_age*, *user_gender*, *item_ID*, *item_category*, *item_brand*, and so on. For the rest datasets, we only select *user_ID* and *item_ID* as features. Each sample contains an interaction $\langle u_i, v_j \rangle$ and corresponding label $r_{i,j} \in \{0, 1\}$ which indicates whether user u_i will click the item v_j. Here we select User-User (UU) and User-Item (UI) to extract different neighbor information. The statistics of datasets are summarized in Table 2.

5.2 Baselines and Metrics

We compare the proposed comb-K recommendation with traditional top-K recommendation, especially state-of-the-art heterogeneous GNN based models, to show its superiority in the promotion

[3]http://www.gurobi.com/

[4]https://www.taobao.com

Table 3: The comparisons of comb-K recommendation v.s. top-K recommendation. Best results are indicated in bold.

Paradigms	Models	Datasets					
		1-day		2-day		3-day	
		TC@K	HR@K	TC@K	HR@K	TC@K	HR@K
Top-K (Previous)	CTR	26883	0.462	50062	0.462	75393	0.408
	GREE	39000	0.576	66691	0.526	101069	0.454
	IntentGC	39121	0.576	65912	0.528	101212	0.458
	IntentGC+Avg	39005	0.574	65432	0.524	100682	0.450
	IntentGC+Att	38605	0.574	65554	0.526	100850	0.452
	MEIRec	38992	0.574	65912	0.530	99541	0.458
	MEIRec+Avg	38224	0.570	65573	0.526	99076	0.450
	MEIRec+Att	38743	0.572	65795	0.528	99239	0.454
	HGRec	38921	0.574	66231	0.528	100213	0.456
	HGRec+Avg	38815	0.568	65868	0.518	98993	0.448
	HGRec+Att	38873	0.572	65830	0.526	99717	0.452
Comb-K (Our)	User-level	39798	**0.594**	68069	**0.558**	**104664**	**0.490**
	Group-level	**40088**	0.588	**68543**	0.548	103926	0.482

recommendation. For traditional top-K based recommendation, we first estimate the item CTR or user/group preference (i.e., $\hat{l}_j$, $\hat{r}_{i,j}$, and $\hat{r}_{p,j}$) and then greedy select top-K items as the K-set. The previous top-K based methods which find the optimal selection indicator X_j, are shown as follows:

- **DNN**: We leverage 3-layers DNN to estimate the item CTR $\hat{l}_j$ based on its features and then select top-K items with the highest CTRs as the K-set.
- **GREE**[2]: GREE is a classical group recommendation model which recommends items to the group in the top-K manner. Here we take all users as one group, estimate group preference $\hat{r}_{p,j}$, and select top-K items with the highest group preference as the K-set.
- **MEIRec**[4]/**IntentGC**[33]/**HGRec**[22]: They are heterogeneous GNN based recommendation models. We use them to estimate user preference $\hat{r}_{i,j}$, rank the items by their accumulated preferences on all users, and select top-K items as the K-set.
- **MEIRec+Avg/Att**: It is the extend version of MEIRec. We aggregate all user embeddings learned by MEIRec as one group embedding via average/attention aggregator, estimate group preference $\hat{r}_{p,j}$, and then greedy select top-K items with the highest group preference. Note that the attention aggregator is actually the proposed heterogeneous graph pooling in Section 4.1.2.
- **IntentGC+Avg/Att**: It is the extend version of IntentGC. The setting is the same as MEIRec+Avg/Att.
- **HGRec+Avg/Att**: It is the extend version of HGRec. The setting is the same as MEIRec+Avg/Att.

For the comb-K recommendation, we first estimate preference and then find the optimal K-set in the combinatorial optimization manner. The proposed comb-K based methods which find the optimal combination of item selection (e.g., X_j) and item delivery (e.g., $Y_{i,j}$ and $Y_{p,j}$) simultaneously are shown as follows:

- **User-level**: It is the user-level comb-K recommendation. We use heterogeneous graph convolution to estimate user preference $\hat{r}_{i,j}$ and select K items via solving Eq. 4.
- **Group-level**: It is the group-level comb-K recommendation. We use heterogeneous graph pooling to estimate group preference $\hat{r}_{p,j}$ and select K items via solving Eq. 5.

Parameter Settings. All models are implemented with Tensorflow 1.8 on PAI[5] with Tesla P100 Cluster. For fair comparison, we randomly initialize model parameters with Gaussian distribution and optimize the model with Adam, and set the batch size to 1024, the learning rate to 0.001, the feature embedding to 8, the node embedding to 128, the regularization to 0.001, and the dropout rate to 0.6. For heterogeneous GNNs (e.g., MEIRec, IntentGC, HGRec, and our model), we sample 5 neighbors via both UU and UI to ensure fairness. And we uniformly set $K = 500$ and $W = 40$ to satisfy the requirements of Taobao promotion scenario.

Since the proposed comb-K recommendation needs to find the optimal combination of item selection X_j and item delivery $Y_{i,j}$ with more decision variables, we only leverage the proposed fast search strategy RNHS to solve it. Note that we only present the performance of group-level comb-K recommendation on the Large data because user-level comb-K recommendation with RNHS still cannot be solved in the limited time.

To evaluate the effectiveness of the selected K items in the promotion scenario, we select the following metrics:

Total Click. Total click (TC) measures how many times the selected items will be click in delivery phase, which is the core KPI[6] of promotion recommendation. Specially, $TC@K$ is defined as follows:

$$TC@K = \sum_{v_j \in \mathcal{S}} clk_j, \tag{20}$$

where clk_j denotes the click time of item v_j in delivery phase.

Hit Ratio. Hit ratio (HR) is a recall-based metric, measuring how many the testing ground-truth K items $\mathcal{S}'$ are in the selected

[5]https://www.aliyun.com/product/bigdata/product/learn
[6]Key Performance Indicator

K items $\mathcal{S}$ via different recommendation models. Here we use the K items with the highest click-through count in delivery phase as the ground truth. Specially, $HR@K$ is defined as follows:

$$HR@K = \frac{|\mathcal{S} \cap \mathcal{S}'|}{K}. \tag{21}$$

5.3 Recommendation Evaluation (RQ1)

We start by evaluating the effectiveness of K items selected by different models and show the superiority of comb-K recommendation over top-K recommendation in the promotion scenario. The comparison results on 1-day, 2-day, and 3-day datasets are reported in Table 3.

The major findings from the experimental results are summarized as follows:

- The proposed comb-K recommendation consistently outperforms all top-K based models on all datasets. The results indicate the superiority of comb-K recommendation over top-K recommendation, providing a more principled way to seamlessly integrate item selection and delivery for maximizing the real revenue in the promotion scenario.
- Comb-K recommendation usually performs better in user level because it fully considers the personalized preferences of M users rather than P group preferences ($P \ll M$). However, the gap is not significant, which demonstrates the effectiveness of group-level modeling. For large-scale users, user-level comb-K recommendation is unsolvable due to the combinatorial explosion, so we can only solve group-level comb-K recommendation with acceptable revenue loss. It implies that group-level comb-K recommendation is a more practical way for large-scale promotion recommendation, which balances the trade-off between effectiveness and efficiency.
- When aggregating user embedding to update group embedding, attentive group embedding usually outperforms the averaged group embedding. For example, HGRec+Att usually outperforms HGRec+Avg on both TC and HR. It demonstrates that the different users indeed make different contributions in different groups and shows the necessity of self-attentive user clustering in the proposed heterogeneous graph pooling.

5.4 Effect of Delivery Window (RQ2)

The key idea of comb-K recommendation is to seamlessly integrate item selection and delivery with the crucial constraint of delivery window. In this section, we study how different delivery windows affect the performance of comb-K recommendation. Here we use RNHS to solve both user-level and group-level comb-K recommendation with different delivery windows (e.g., 10-50), and show how the relative improvements (e.g., HR_{Comb-K}/HR_{Top-K}) change in Figure 3. To ensure fairness, we use the same user/group preferences (i.e., $\hat{r}_{i,j}$ and $\hat{r}_{p,j}$) for both top-K recommendation and comb-K recommendation.

Based on Figure 3, we have the following observations:

- With the growth of the delivery window, the relative improvements on TC and HR both raise first and then start to drop. The comb-K recommendation usually achieves the best performance when the delivery window is set to 30~40, which is consistent with the real user behavior. For example, on the Large dataset, comb-K recommendation with W=30~40 improves TC and HR significantly by 9.35% and 7.14%, respectively. Note that an inappropriate setting of the delivery window may drop the performance of comb-K recommendation. On 3-day dataset, group-level comb-K recommendation cannot outperform top-K recommendation when the delivery window W is set to 50. It demonstrates that the delivery window indeed improves the performance of comb-K recommendation.
- The delivery window phenomenon exists in both user-level and group-level. When varying the size of the delivery window, the performances of both user-level and group-level comb-K recommendations show similar trends, which both raise first and then start to drop. It demonstrates the delivery window phenomenon is also exists in group-level and also verifies the effectiveness of the group-level comb-K recommendation.

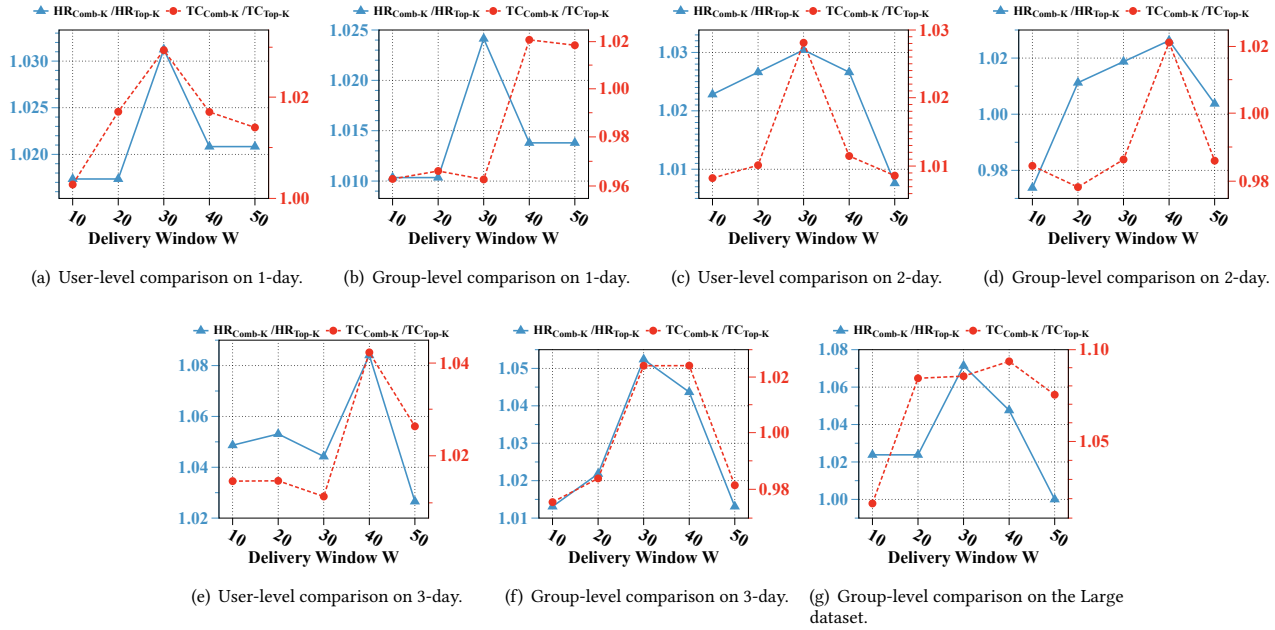
Note that we do not show user-level comparison on the Large dataset because the large-scale comb-K recommendation in user level is unsolvable even with RNHS. This further confirms the necessity of group-level comb-K recommendation in large-scale promotion recommendation.

5.5 User Preference v.s. Group Preference(RQ3)

The user clustering and group preferences serve as the basis of group-level comb-K recommendation and directly affect its performance. In this section, we compare the performance of user preference $\hat{r}_{i,j}$ with group preference $\hat{r}_{p,j}$ to verify the effectiveness of user clustering and group preferences simultaneously. Specifically, we take the AUC ratio (=$\text{AUC}_{G-I}/\text{AUC}_{U-I}$) as the evaluation metric. If the AUC ratio ≈ 1, then we can conclude the proposed heterogeneous graph pooling is able to cluster users into proper groups and personalized user preferences are well-preserved in group level. We also show how different training strategies and the number of groups affect the heterogeneous graph pooling. Specifically, we vary the number of groups and train the heterogeneous graph pooling to estimate the group preference via Pretrain & Fine-tuning and Interval Training. We also use compression ratios (=#Users/#Groups) to measure the degree of preference compression.

The experimental details are shown in Figure 4 and we have the following findings:

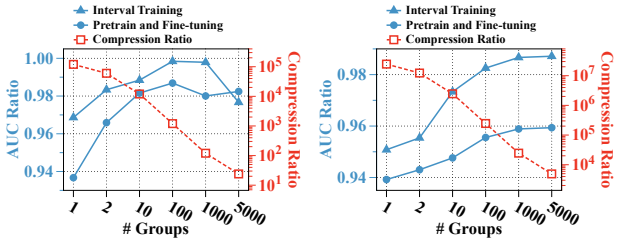
- Benefitted from the proposed heterogeneous graph pooling, the AUC ratios are close to 1 with high compress ratios, which means users are clustered into proper groups and the major user preferences are well-preserved in group preference. For example, on the Large dataset with around 2.5×10^7 users, if we cluster all users into 10^3 groups, the AUC ratio is up to 98.7% even when the compression ratio is up to 2.5×10^4, indicating both effectiveness and efficiency of the proposed heterogeneous graph pooling.
- With the growth of the number of groups, the AUC ratios first raise up and then keep stable. The optimal number of groups is related to the number of users. For example, we



(a) User-level comparison on 1-day. (b) Group-level comparison on 1-day. (c) User-level comparison on 2-day. (d) Group-level comparison on 2-day.

(e) User-level comparison on 3-day. (f) Group-level comparison on 3-day. (g) Group-level comparison on the Large dataset.

Figure 3: Effect of delivery window. We select TC_{Comb-K}/TC_{Top-K} and HR_{Comb-K}/HR_{Top-K} to show how comb-K recommendation with different delivery windows is superior to top-K recommendation. Note that we only show group-level comparison on the Large dataset because large-scale comb-K recommendation in user-level is unsolvable.



(a) The comparison on 1-day dataset. (b) The comparison on the Large dataset.

Figure 4: User preference v.s. Group preference. We show how the AUC ratios change with regard to the training strategies and the number of groups.

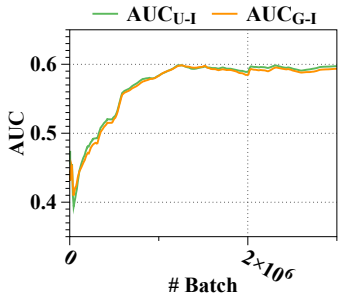


Figure 5: Test AUC of user-item recommendation and group-item recommendation on the Large dataset.

only need 10^2 groups to preserve all users preferences on 1-day dataset, while the Large dataset shows the highest AUC ratio when the number of group is set to 10^3. It makes sense because we need enough groups (e.g., 10^3) to preserve diverse users preferences and too many groups may introduce some redundancies.

- Different training strategies affect the performance of user clustering and interval training shows its superiority over pretrain & fine-tuning. The reason is that interval training is similar to multi-task training, making user preference and group preference mutually reinforce each other and improving the performance.

We also show the test AUC of user-item recommendation and group-item recommendation on the Large dataset in Figure 5. Obviously, group-item preference is a good approximation of user-item preference because their performances are largely consistent with little inevitable information loss. In summary, user preferences are well-preserved in group level via the proposed heterogeneous graph pooling.

5.6 RNHS Evaluation (RQ4)

Following the prior work [15], we random synthesize datasets with different scales Sc (e.g., Sc={3, 6, · · · , 1536}) for comb-K recommendation and compare the proposed RNHS with traditional integer programming (IP). We do not synthesize larger datasets (e.g., Sc=3072)

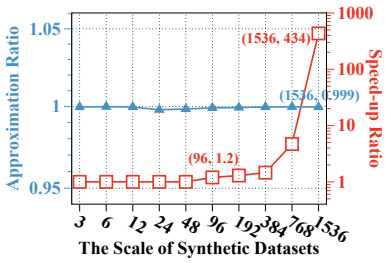


Figure 6: Comparing the efficiency and effectiveness of the proposed RNHS with integer programming on synthetic datasets with different scales. The higher approximation ratio, the better RNHS is. The higher speed-up ratio, the faster RNHS is.

because integer programming is too slow to solve it. For each synthetic dataset with scale Sc, we mock up Sc users, Sc items, and randomize user-item preferences via uniform distribution $U \sim [0, 1]$. Then, we run RNHS to solve comb-K recommendation with the constraints $K = \sqrt{Sc}, O = \sqrt{K}$ on all synthetic datasets and use the approximation ratio ($=Obj_{RNHS}/Obj_{IP}$) and speed-up ratio ($=Time_{IP}/Time_{RHNS}$) to show its superiority over traditional integer programming.

The comparison results are shown in Figure 6 and we have the following observations:

- For all datasets with different scales, Obj_{RNHS} is always very close to Obj_{IP} (a.k.a., $Obj_{RNHS}/Obj_{IP} \approx 1$) which means the proposed RNHS is able to ensure sufficiently accurate solutions. For example, on synthetic dataset with Sc=1536, the approximation ratio Obj_{RNHS}/Obj_{IP} is up to 0.999 and the gap is negligible.
- With the growth of the scale of the dataset, the speed-up ratio grows exponentially, indicating the efficiency of the proposed RNHS. For example, on synthetic dataset with Sc=96, the speed-up ratio is just 1.2x. But, on synthetic dataset with Sc=1536, RNHS achieves 434x speedup for solving comb-K recommendation with approximation ratio 0.999.

6 CONCLUSION

In this paper, we deeply investigate the problem of promotion recommendation, which aims to select a set of K items based on *all user preferences* in selection phase and maximize the total revenue in delivery phase. To solve it, we present the comb-K recommendation model, a constrained combinatorial optimization model integrating the selection phase and delivery phase seamlessly with delicately designed constraints, which searches the optimal combination of item selection and delivery simultaneously. Specifically, we propose a heterogeneous graph convolution to estimate user preferences and present a user-level comb-K recommendation model with the full consideration of all user preferences. For large-scale users, we furtherly cluster massive users into limited groups via a novel heterogeneous graph pooling and present a group-level comb-K recommendation model with the full consideration of all group preferences. In addition, we design a restricted neighbor heuristic search (RNHS) to accelerate the search process of comb-K recommendation. Extensive experiments on four datasets show the superiority of the comb-K recommendation over top-K recommendation in large-scale promotion recommendation. On billion-scale data, when clustering 2.5×10^7 users into 10^3 groups, our model is able to preserve 98.7% user preferences in group level and significantly improves the Total Click and Hit Ratio by 9.35% and 7.14%, respectively.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. U20B2045, 61772082, 61702296, 62002029, U1936104, 61972442). It is also supported by National Social Science Fund of China (18CSH019), Beijing Social Science Fund(20JCC096), and BUPT Excellent Ph.D. Students Foundation (No. CX2020311).

REFERENCES

[1] Qingyao Ai, Keping Bi, J. Guo, and W. Croft. 2020. Learning a Deep Listwise Context Model for Ranking Refinement. In *SIGIR*. 135–144.
[2] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang. Hong. 2018. Attentive Group Recommendation. In *SIGIR*. 645–654.
[3] G. A. Croes. 1958. A Method for Solving Traveling-Salesman Problems. *Operations Research* 6 (1958), 791–812.
[4] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation. In *KDD*. 2478–2486.
[5] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *WWW*. 417–426.
[6] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. In *WWW*. 2331–2341.
[7] Hongyang Gao and Shuiwang Ji. 2019. Graph U-Nets. In *ICML*. 2083–2092.
[8] Y. Gong, Y. Zhu, L. Duan, Qingwen Liu, Ziyu Guan, Fei Sun, Wenwu Ou, and K. Q. Zhu. 2019. Exact-K Recommendation via Maximal Clique Optimization. In *KDD*.
[9] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.
[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
[11] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In *WWW*. 2704–2710.
[12] Houye Ji, Junxiong Zhu, Xiao Wang, Chuan Shi, Bai Wang, Xiaoye Tan, Yanghua Li, and Shaojian He. 2021. Who You Would Like to Share With? A Study of Share Recommendation in Social E-commerce. In *AAAI*.
[13] Jyun-Yu Jiang, Patrick H Chen, Cho-Jui Hsieh, and Wei Wang. 2020. Clustering and Constructing User Coresets to Accelerate Large-scale Top-K Recommender Systems. In *WWW*. 2177–2187.
[14] Ray Jiang, Sven Gowal, Timothy A. Mann, and Danilo J. Rezende. 2018. Beyond Greedy Ranking: Slate Optimization via List-CVAE. In *ArXiv*.
[15] E. Khalil, Hanjun Dai, Yuyu Zhang, B. Dilkina, and L. Song. 2017. Learning Combinatorial Optimization Algorithms over Graphs. In *NIPS*. 6348–6358.
[16] Amir Hosein Khasahmadi, Kaveh Hassani, Parsa Moradi, Leo Lee, and Quaid Morris. 2020. Memory-Based Graph Networks. *ICLR*.
[17] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[18] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *ICLR*.
[19] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-Attention Graph Pooling. In *ICML*. 6661–6670.
[20] Z. Li, Shen Xin, Yuhang Jiao, Xuming Pan, Pengcheng Zou, Xianling Meng, Chengwei Yao, and Jiajun Bu. 2020. Hierarchical Bipartite Graph Neural Networks: Towards Large-Scale E-commerce Applications. In *ICDE*. 1677–1688.
[21] D. Sculley. 2010. Web-Scale K-Means Clustering. In *WWW*. 1177–1178.
[22] Jinghan Shi, Houye Ji, Chuan Shi, Xiao Wang, Zhiqiang Zhang, and Jun Zhou. 2020. Heterogeneous Graph Neural Network for Recommendation. In *ICML Workshop*.
[23] Shaoyun Shi, Weizhi Ma, Min Zhang, Yongfeng Zhang, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. 2020. Beyond User Embedding Matrix: Learning to Hash for Modeling Large-Scale Users in Recommendation. In *SIGIR*. 319–328.

[24] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[25] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*. 165–174.
[26] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *WWW*. 2022–2032.
[27] Xiao Wang, R. Wang, C. Shi, G. Song, and Q. Li. 2020. Multi-Component Graph Convolutional Collaborative Filtering. In *AAAI*.
[28] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *ICML*. 5453–5462.
[29] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *NIPS*. 4800–4810.
[30] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and H. Kim. 2019. Graph Transformer Networks. In *NeurIPS*. 11983–11993.
[31] Minjia Zhang, X. Liu, Wenhan Wang, Jianfeng Gao, and Yuxiong He. 2018. Navigating with Graph Representations for Fast and Scalable Decoding of Neural Language Models. *NIPS*, 6308–6319.
[32] Yizhou Zhang, Yun Xiong, Xiangnan Kong, Shanshan Li, Jinhong Mi, and Yangyong Zhu. 2018. Deep collective classification in heterogeneous information networks. In *WWW*. 399–408.
[33] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. 2019. IntentGC: a Scalable Graph Convolution Framework Fusing Heterogeneous Information for Recommendation. In *KDD*. 2347–2357.
[34] Qiwei Zhong, Yang Liu, Xiang Ao, Binbin Hu, Jinghua Feng, Jiayu Tang, and Qing He. 2020. Financial Defaulter Detection on Online Credit Payment via Multi-view Attributed Heterogeneous Information Network. In *WWW*.
[35] Dennis Zill, Warren S Wright, and Michael R Cullen. 2011. *Advanced engineering mathematics*. Jones & Bartlett Learning.