Personalized Bundle List Recommendation

Jinze Bai
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
baijinze@pku.edu.cn

Chang Zhou*
Alibaba Group
ericzhou.zc@alibaba-inc.com

Junshuai Song
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
songjs@pku.edu.cn

Xiaoru Qu
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
quxiaoru@pku.edu.cn

Weiting An
Alibaba Group
weiting.awt@alibaba-inc.com

Zhao Li
Alibaba Group
lizhao.lz@alibaba-inc.com

Jun Gao
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
gaojun@pku.edu.cn

ABSTRACT

Product bundling, offering a combination of items to customers, is one of the marketing strategies commonly used in online e-commerce and offline retailers. A high-quality bundle generalizes frequent items of interest, and diversity across bundles boosts the user-experience and eventually increases transaction volume. In this paper, we formalize the personalized bundle list recommendation as a structured prediction problem and propose a bundle generation network (BGN), which decomposes the problem into quality/diversity parts by the determinantal point processes (DPPs). BGN uses a typical encoder-decoder framework with a proposed feature-aware softmax to alleviate the inadequate representation of traditional softmax, and integrates the masked beam search and DPP selection to produce high-quality and diversified bundle list with an appropriate bundle size. We conduct extensive experiments on three public datasets and one industrial dataset, including two generated from co-purchase records and the other two extracted from real-world online bundle services. BGN significantly outperforms the state-of-the-art methods in terms of quality, diversity and response time over all datasets. In particular, BGN improves the precision of the best competitors by 16% on average while maintaining the highest diversity on four datasets, and yields a 3.85x improvement of response time over the best competitors in the bundle list recommendation problem.

KEYWORDS

Bundle Recommendation; Bundle Generation; Diversity

*Corresponding Author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313568

ACM Reference Format:
Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, Jun Gao. 2019. Personalized Bundle List Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313568

1 INTRODUCTION

A bundle is a collection of products or services for sale as a whole, and the bundle list recommendation is to recommend a personalized list of bundles. Traditional business domains, e.g., communication services, offline retailers and supermarkets, take bundling as a critical marketing strategy, and they usually make bundles by human insights or non-personalized mining methods [2]. Modern e-commerce websites and online service business, e.g., Amazon, Taobao, Steam and Netflix, also deploy new applications [22, 32], which recommend and sale a list of bundles rather than a list of items.

Customers, sellers, as well as the recommendation platform, could benefit from the personalized bundle recommendation service. For customers, high-quality bundles broaden user interests and indicate the complementary products directly in the shortest path after purchase. It is well known in classic marketing research that carefully-designed bundle brings mutual promotions.

For sellers, bundling increases per customer transaction. The average order size for walmart.com is 2.3 [36]. By attaching some highly correlated products, the seller can launch more products and increase gross merchandise volume (GMV). Besides, buying a bundle of products may cost less than buying each product separately for both customers and sellers. For example, it is a common practice to waive shipping fee if the items in one order exceed a certain amount in e-commerce websites.

For the recommendation system, bundling is a more efficient way to organize and present products. The recommendation list is structured by bundles where the products are highly correlated

Figure 1: Bundle List Recommendation Example

in contrast to the list of items, and multiple diversified bundles avoid monotonous choices for users. The recommendation system shows bundle as a whole and thus presents more products per unit area. It is crucial to reduce swipe-up operations for improving user experiences for the mobile application.

In Figure 1, we illustrate a bundle list example. The items in the bundle should be highly correlated. Besides, diversity across bundles becomes a critical metric as an interpretable, business-oriented goal, which is independent of click-through rate (CTR) and precision, trying to avoid tedious choices and quantify the user-experience in bundle recommendation system. The identical high-quality item tends to be generated in different bundles [22], which compose the confusing and monotonous bundle list, so there is a trade-off between precision/likelihood and diversity. Other meaningful problems, e.g., how to set the price discount for maximizing the profit, how to compose the picture of the bundle, are beyond the scope of this paper and can be the future work. Our paper focuses on how to generate high-quality and diversified bundles.

There are several challenges for the personalized bundle recommendation problem. First, the recommendation system should generate high-quality bundles automatically and flexibly without manual involvement. Frequent itemsets mining algorithms provide high-quality bundling but are not personalized. For modeling the distribution of bundle, we use the co-purchase data or pre-defined bundles to supervise the quality of bundles with the user context. However, the sequence generation methods suffer from inadequate representation for rich features in bundle recommendation scenario, because the weight matrix of traditional softmax only considers the item_id. Besides, in practice, we may need the larger size of bundles for the purpose of sellers' promotion, but seq2seq usually suffers from generating short sequences [14].

Second, diversity across bundles emerges as a critical metric for bundle recommendation. Different bundles with the same high-quality item may reach high ranking scores at the same time, so duplicated items tend to be seen in these generated bundle list [22]. Besides duplication, similar items due to the similarity of features may produce confusing and tedious bundle list and decrease user interest. A bundle recommendation system should consider the cross-bundle diversity considering both duplication and similarity of items. Here, the diversity is a explicit metric instead of some hidden relationships of similarity learned from data. The explicit similarity pre-defined by side information and prior knowledge provides the exact signal for the purpose of measurement and adjustment. We use the determinantal point processes (DPPs) as the objective function which considers the global negative correlations during inference to decompose the quality and diversity.

Last, compared to the traditional item list recommendation, the search space is the doubly exponential number of items for the bundle recommendation, since a bundle can contain arbitrary-length combination theoretically and a list contains multiple bundles [9]. Bundle sparsity should be carefully considered because the training data in the form of co-purchase contains few bundle ground truths w.r.t. the whole search space. Most of the bundles in search space are low-quality, so we need a method to generate high-quality candidate bundles efficiently for each user, instead of the ranking method applied in all the possible bundles.

In this paper, we aim to the problem of the personalized bundle list recommendation and propose a **Bundle Generation Network (BGN)**. We summarize the contributions of this paper as follows:

- It is the first attempt to address the bundle list recommendation problem by the sequence generation approach considering both the quality and diversity. We decompose the generation model based on determinantal point processes, and integrate the neural network with DPP selection to produce the high-quality and diversified bundle list.
- We propose a feature-aware softmax in bundle recommendation scenario, for making up the inadequate representation of traditional seq2seq for rich features, which improves the modeling of quality significantly. The loss of feature-aware softmax is a generalization of optimizing BPR loss at each step.
- We conduct extensive experiments on both the real-world dataset and public datasets. Our model improves the precision of the best baseline by 16% on average while maintaining the highest diversity on four datasets, and shows the fastest response time in the bundle list recommendation problem. Besides, BGN can control the bundle size with a tolerable reduction in precision.

2 RELATED WORKS

We present the existing bundle recommendation approaches and introduce the sequence generation model (Seq2seq) and Determinantal Point Processes (DPPs) as related works.

2.1 Bundle Recommendation

Product bundling is one of the marking strategies commonly used in both offline retailers and online e-Commerce websites. Frequent itemsets mining algorithms like Apriori [2] provide high-quality bundling but are not personalized. Parameswaran et al. [21] study some complex constraints like cross-item dependencies in personalized curriculum bundling. The complexity of top-K bundle recommendation is discussed comprehensively in [9]. All above only use frequency of items, which cannot utilize rich features and fail to generalize on similar items and users.

For the single bundle recommendation, Zhu et al. [36] propose a personalized method based on integer quadratic programming. This method considers the first and the second terms of cross-item dependencies. However, the estimation of the second term is heuristic, and this method ignores the higher order terms of cross-item dependencies, which gives a sub-optimal result. The bundle recommendation problem to groups rather than users is investigated by [23]. For the bundle list recommendation, Xie et al. [32] generate top-k bundles via sampling and preference elicitation. This method uses an additive utility function with a linear combination of the corresponding (aggregate) feature values, which fails to capture the complicated dependencies. In [22], Pathak and Gupta build an optimization criterion upon the latent factor model, Bayesian Personalized Ranking (BPR) [24], with the mean pair-wise Pearson correlation of the items. This method generates personalized bundles by a two-stage approach, combining the corresponding preference scores with a greedy annealing schedule, which might be caught in the local optimum due to randomness and instability of annealing.

2.2 Sequence Generation Approaches

In recent years, sequence generation methods achieve great successes in many tasks like Machine Translation [26] and Dialog Generation [29]. They adopt an encoder-decoder architecture to generate sequences, and use the beam search strategy [30] to produce the maximum possible sequences approximately.

In recommendation systems, some works based on RNN, CNN, attention extract the user embedding from the behaviors for each user [31, 33, 34]. The recurrent model and the memory network are also adopted for sequential recommendation [7, 18]. The difference between the sequential recommendation and the bundle recommendation is that, the former uses the encoder only, while the latter uses the encoder to model the user context and uses the decoder to model the bundle distribution. The decoder usually has separate trainable parameters without sharing with the encoder. Besides, The bundle recommendation produces multiple bundles, so we need to know when to end the generation and control diversity of bundles.

2.3 Determinantal Point Processes

Determinantal Point Processes (DPPs), first introduced to model fermion behavior, have gained popularity due to their elegant balancing of quality and diversity. DPPs have been studied for their theoretical properties [11, 16], and their machine learning applications including summarization [5, 17], object retrieval [1] and conversation generation [25]. Structured Determinantal Point Processes (SDPPs) [15] extend the DPP model to handle an exponentially-sized set of particles (structures) via a natural factorization combined with the second-order message passing, which leads to tractable algorithms for inference.

Recent works use DPPs for item list recommendation in [10, 35] to solve cross-item diversity, but to the best of our knowledge, there are no studies about cross-bundle diversity. In this paper, we use SDPP as the tool to decompose the quality/diversity term.

3 PERSONALIZED BUNDLE LIST RECOMMENDATION

Notation. Let $\mathcal{I}$ be the whole set of items that one could select from and $|N|$ be the number of all items. A bundle b is any combination out of $\mathcal{I}$ which consists of items, and T denotes the number of items in a bundle. The items are unordered in bundles. When regarding the bundle as a sequence, we usually give the order by sorting the items of the bundle in terms of their prices in a descendant order, because the cheaper items are more substitutable for the bundle and could endure more exposure bias in the sequence generation model. Let $\mathcal{B}$ be the set of all possible bundles, so $|\mathcal{B}|$ equals $\sum_{t=1}^{T}\binom{N}{t}$ with the fixed maximum bundle size T, where $\binom{N}{t} = \frac{N!}{t!(N-t)!}$ is the combination of t out of N.

$$\mathcal{I} = \{1, 2, \ldots, N\} \qquad |\mathcal{I}| = N$$

$$\mathcal{B} = \{b = \{i_1, i_2, \ldots, i_T\} | i \in \mathcal{I}\} \qquad |b| = T,\ |\mathcal{B}| = \sum_{t=1}^{T}\binom{N}{t}$$

$$\mathcal{Y} = \{y = \{b_1, b_2, \ldots, b_K\} | b \in \mathcal{B}\} \quad |y| = K,\ |\mathcal{Y}| = \binom{|\mathcal{B}|}{K}$$

Here K is the size of a recommended bundle list y. Still, we consider the order does not matter in the bundle list. Let $\mathcal{Y}$ be the set of all possible bundle list. The user context C_u is usually represented as the item sequence he/she has clicked/bought in the history.

Problem Formalization. Let $\tilde{\mathcal{P}}(\cdot,\cdot) : \mathcal{Y} \times \{C_u\} \to \mathbb{R}^+$ be a universal compatibility function measuring the score (compatibility) of the bundle list y and the user context C_u, and we refer $\tilde{\mathcal{P}}(y|C_u)$ to a unnormalized conditional probability of y given C_u. Then, the personalized bundle list recommendation problem can be formalized as a structured prediction problem [3], which tries to find the bundle list $\hat{y}$ satisfying:

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \tilde{\mathcal{P}}(y = \{b_1, b_2, \ldots, b_K\} | C_u) \tag{1}$$

Let $\mathcal{P}(y|C_u)$ be the normalized conditional probability of $\tilde{\mathcal{P}}$ over y. $\mathcal{P}(y|C_u)$ is actually modeling a distribution over the *doubly exponential* number of N, so there are $\binom{\sum_{t=1}^{T}\binom{N}{t}}{K}$ possible structures of y given C_u naively, which poses an extreme computational challenge. In this paper, we consider the compatibility function $\tilde{\mathcal{P}}$ related to both the quality and diversity.

The bundle list recommendation is a generalization of the single bundle recommendation problem and single-item list recommendation problem. When each bundle is independent given the user context, we have $\mathcal{P}(y = \{b_1, b_2, \ldots, b_K\}|C_u)) = \prod_k \mathcal{P}(b_k|C_u)$. In this case, there are no correlations among the bundle list, and the

problem becomes the single bundle recommendation problem [36], sometimes considered as the personalized bundle ranking problem [22]. Furthermore, when the bundle size T equals one it is the single-item list recommendation problem.

4 BUNDLE GENERATION NETWORK

Here we propose a *Bundle Generation Network* (BGN) to solve the personalized bundle list recommendation problem considering both the quality and diversity.

In the following subsections, we introduce how to decompose the bundle list probability into the quality/diversity part by SDPP and factorize them respectively. Then, we illustrate the measurement of diversity and design an improved neural network with the feature-aware softmax to learn the quality part. Besides, we propose a masked beam search to control the bundle size and avoid duplicated items. Finally, we demonstrate how to integrate these parts to BGN to tackle the process of generation for the bundle recommendation.

4.1 SDPP-based Bundle Generation Model

We define a *structured determinantal point process* (SDPP) [15] on the probability space $(\mathcal{B}, \mathcal{Y}, \mathcal{P})$. When Y is a random variable of bundle recommendation list drawn according to the probability mass function (PMF) $\mathcal{P}$, the distribution is given by:

$$\mathcal{P}(\mathrm{Y} = y) \triangleq det(L_y)/Z \tag{2}$$

Here, L is a symmetric positive semi-definite kernel of SDPP, with the size of $|\mathcal{B}| * |\mathcal{B}|$. L_y denotes the restriction of L to the entries indexed by elements of y, and $Z = \sum_{y' \in \mathcal{Y}} det(L_{y'})$ is the normalization constant. There is an intuitive geometric interpretation of the kernel L, that is, $\det(L_y)$ is the square of the volume spanned by its associated feature vectors [16]. Notice that, each element of y refers to a bundle which has a certain structure, so we consider that the DPP is structured.

One of the benefits of defining a DPP probability model is that we can decompose quality and diversity in a very natural way. For any element L_{ab} of positive semi-definite matrix L we could always find a quality/diversity decomposition:

$$\begin{aligned} L_{ab} &= q(a)\phi(a)^T\phi(b)q(b) \\ &= q(a)S(a,b)q(b) \end{aligned} \tag{3}$$

where $a, b \in \mathcal{B}$, L_{ab} denotes the element indexed by bundle a and b, $q(b) \in \mathbb{R}$ denotes the quality term of bundle, and $\phi(b) \in \mathbb{R}^D$ denotes the normalized diversity feature with $||\phi(b)||_2 = 1$. Here we define a similarity function/metric $S(a,b) : \mathcal{B} \times \mathcal{B} \to \mathbb{R}$ which satisfies $S(a,b) = \phi(a)^T\phi(b)$ and keeps the corresponding similarity matrix S positive semi-definite. We reformalize Equation (3) in the matrix way as

$$L = QSQ \tag{4}$$

where Q denotes a diagonal matrix $\mathrm{diag}\{q(b)_{b \in \mathcal{B}}\}$. The decomposition balances the quality and the diversity. Then, the optimization criterion of the personalized bundle list recommendation is to find a bundle list y maximizing the $\mathcal{P}(y|C_u;\theta)$ given the user context C_u:

$$\arg\max_{y} \ \det(S_y) \cdot \prod_{b \in y} q^2(b|C_u;\theta) \tag{5}$$

One major challenge for solving the personalized bundle list recommendation is that, the optimization criterion of the bundle list has the complexity over doubly exponential number of N. According to the decomposition in Equation (5), we reduce the complexity to a quality term q over the measure space $\mathcal{B}$, and a diversity term S over the measure space $\mathcal{B} \times \mathcal{B}$. However, both of the terms still have exponential space which is intractable.

SDPP considers that any elements in measure space $\mathcal{B}$ can factorize over a set of factors F [15], where a factor $\alpha \in F$ is a small subset of the parts of a structure. Then, we continue to factorize the quality term and the diversity term respectively:

$$q(b) = \prod_{\alpha \in F} q_\alpha(b_\alpha) \qquad \phi(b) = \sum_{\alpha \in F} \phi_\alpha(b_\alpha) \tag{6}$$

The factorization defines how we interpret the model structure and keeps the model tractable. Then we shall specify the factorization for the diversity and quality respectively, and integrate them to produce high-quality and diversified bundle list.

4.2 The Measurement and Factorization of Diversity

The diversity we measure in our work is a interpretable metric/goal, trying to quantify the user-experience in bundle recommendation system, which is independent of the likelihood from data. This implicit relationship learned from data may not be the same 'diversity' as we target, and the correlations among items from data have been considered into the quality part. Whereas, the explicit similarities defined by side information and prior knowledge provide exact signals for the purpose of the measurement and adjustment. Besides, the recommendation system is usually lake of the diversity-labeled ground-truth list, which makes it hard to supervise the complete likelihood.

With Equation (3) and Equation (6) and , we factorize the bundle similarity function to the item similarity function:

$$\begin{aligned} S(a,b) &= \phi(a)^T\phi(b) \\ &= (\sum_{\alpha \in F} \phi_\alpha(a_\alpha)^T)(\sum_{\beta \in F} \phi_\beta(b_\beta)) \\ &= \sum_{\alpha,\beta} \phi_\alpha(a_\alpha)^T\phi_\beta(b_\beta) \\ &= \sum_{i \in a, j \in b} s_{a,b}(i,j) \end{aligned} \tag{7}$$

Here, $s_{a,b}(i,j)$ measures the similarity of items rather than bundles, which is independent of user context. The similarity funciton usually should be consistent with the metrics of diversity predefined according to domain knowledge. For example, in our experiments the $s_{a,b}(i,j)$ is given by Jaccard similarity of bundle:

$$s_{a,b}(i,j) = \frac{1}{|a \cup b|}\delta(i = j) \tag{8}$$

where $\delta(\cdot)$ denotes the indicator function. Note that the bundles do not contain duplicated items inside. This function can maintain the positive semi-definition of L [4] and prevent the identical items shown in the different bundles. The function keeps consistent local measurement with Equation (19), and SDPP optimizes the global negative correlations. Besides, the metric of diversity could also be

defined by calculating the similarity of attributes, which provides the fine-grained diversity of attributes.

4.3 Modeling the Quality Part with Feature-Aware Softmax

Here, we focus on modeling the quality part $q(b|C_u;\theta)$ by a typical *Encoder-Decoder* framework, where the encoder extracts the information from the user context and the decoder generates the high-quality candidate bundle list with a proposed feature-aware softmax.

For the encoder, the user context can be represented as the purchase history of items. We use a text-CNN [13] to extract the information, which has recently been shown of effectiveness in text classification and information extraction [33, 34]. We show the basic network architecture in Figure 2 and omit the discussion of the details because it is not one of our main contributions. The text-CNN is parallelizable and faster than bi-LSTM, and it could also be easily improved by Transformer [27] based on the self-attention architecture in the future work.

For the decoder, the quality of a bundle $q(b|C_u;\theta)$ is an unnormalized joint probability parameterized in terms of a generic θ, so we factorize it in the Bayesian formula way. Note that the Bayesian formula always holds and models the complete distribution of quality, the assumption here is sharing the parameters in a sequence way.

$$\begin{aligned} q(b|C_u;\theta) &\propto p(b=\{i_1,i_2,\ldots,i_T\}|C_u;\theta) \\ &= \prod_{t=1}^{T} p(i_t|i_1,...,i_{t-1},C_u;\theta) \end{aligned} \tag{9}$$

In Equation (9), we factorize the joint probabilities of any bundle by multiplying all conditional probabilities for each of its elements. By sharing the same weights of those conditional probabilities, we use a sequence generation model with the (stacked) LSTM cell with the well-known Luong Attention [19] to calculate the probability. Let x_t be the decoder's input and t be the item position in the bundle. We initialize h_0 with the encoder's output. Still, we omit the common details of LSTM and attention.

$$\begin{aligned} h_0 &= \text{text-CNN}(C_u) \\ x_t &= [\text{item\_emb}(i_t), \text{cate\_emb}(i_t), ...] \\ h_t &= \text{stacked-LSTM-attention}(h_{t-1}, x_{t-1}, C_u) \\ p(i_t|i_1,...,i_{t-1},C_u;\theta) &= \text{feature-aware-softmax}(h_t)_{i_t} \end{aligned} \tag{10}$$

The traditional softmax layer contains a weight matrix $W^{(h+1)*N} = [w_1, w_2, ..., w_N]$, where w_i could be considered as an embedding of item_id of i. Note that, we eliminate the bias term here just for writing convenience.

$$\text{softmax}(h_t, W)_j = \frac{exp(h_t^T w_j)}{\sum_{\hat{j}=1}^{N} exp(h_t^T w_{\hat{j}})} \tag{11}$$

However, the item candidates in recommendation system usually have more belonging features rather than just item_id. The traditional softmax could result in inadequate representation for weight matrix owing to the lack of features.

Inspired by pointer network [28], we propose **feature-aware (FA) softmax**, which modifies the softmax operator with feature-aware weight matrix instead of a stationary weight matrix, and makes the softmax sensitive to dynamic information. Feature-aware softmax uses the dynamic weight matrix built by $F(x_i)$, where F is a nonlinear transformation function for all kinds of features, so that the softmax could be sensitive to features besides item_id.

$$\begin{aligned} E^{(h+1)*N} &= [e_1, e_2, ..., e_N]^1, \qquad e_i = F(x_i) \\ \text{feature-aware-softmax}(h_t)_j &= \text{softmax}(h_t, E)_j \end{aligned} \tag{12}$$

We use the co-purchased data $\mathcal{D} = \{b, C_u\}_{1:|\mathcal{D}|}$ to guide the learning process. The loss could be defined as the mean cross entropy for the softmax layer of each step, which guarantees the same form as Equation (9) and the model is to maximize the likelihood of a single bundle.

$$\mathcal{L}_{\text{MLE}} = -\frac{1}{T}\sum_{t=1}^{T} \log p(i_t|i_1,...,i_{t-1},C_u;\theta) \tag{13}$$

However, the feature-aware softmax might be time-consuming during training owing to constructing the weight matrix E dynamically. So we apply feature-aware softmax with sampled softmax loss [12], denoted as the sampled feature-aware softmax loss, which accelerates the training process.

We show that the sampled feature-aware softmax is a generalization of optimizing the average BPR [24] at each step. The sampled feature-aware softmax samples a part of negative items at each step plus the ground truth of the next step, and then builds the corresponding weight matrix. We regard h_t as the user vector of BPR, e_{i_t} as the positive item vector, and e_{s_t} as the negative item vector, where s_t is the sampled negative item at step t. When setting the sample number to be 1, the loss of the sampled feature-aware softmax is given by:

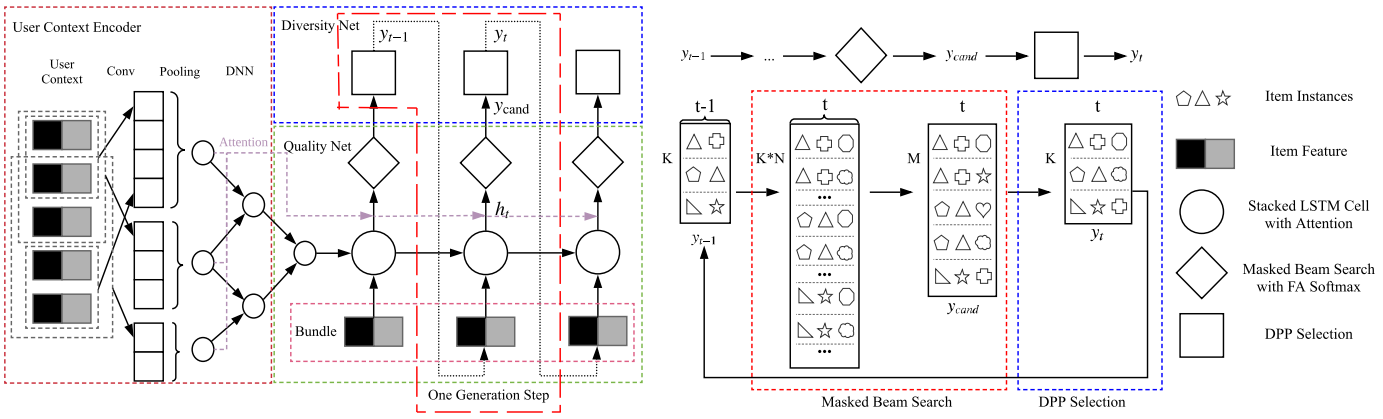
$$\begin{aligned} \mathcal{L}_{\text{sample-1}} &= -\frac{1}{T}\sum_{t=1}^{T} \log \frac{exp(h_t^T e_{i_t})}{exp(h_t^T e_{i_t}) + exp(h_t^T e_{s_t})} + \lambda_\theta ||\theta||^2 \\ &= -\frac{1}{T}\sum_{t=1}^{T} \log \sigma(h_t^T (e_{i_t} - e_{s_t})) + \lambda_\theta ||\theta||^2 \\ &= -\frac{1}{T}\sum_{t=1}^{T} \log p(\theta| \geq_{t,C_u}) = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\text{BPR}}(t) \end{aligned} \tag{14}$$

where $\sigma(\cdot)$ is the logistic sigmoid function, and $\geq_{t,C_u}$ is the pairwise preference at each step as [24]. So the feature-aware softmax with one negative item is equivalent to optimizing average BPR at each time in equation (14).

4.4 Overall Process of Bundle Generation

The goal of bundle list recommendation is to generate the high-quality and diversified bundles list y maximizing the $p(y|C_u)$ in Equation (5). We integrate and summarize the overall process of bundle generation in this subsection.

Figure 2(a) shows the overall inference process of BGN. We initialize h_0 with the output of text-CNN and then produce the bundle list by expanding the generated prefix of bundles at each step. Figure 2(b) shows how the bundle list y_t expands at each generation step. Each small shape icon denotes an item, each row



a. Overall Inference Process

b. One Generation Step

Figure 2: Inference Process of Bundle Generation Network.

of the rectangular block is a generated bundle prefix, and t indicates the aligned size of the generated bundle prefix. At each generation step, we use the beam search to produce $K * N$ candidate bundles. Beam search [30] is a heuristic search algorithm that explores the search space by expanding the most promising item in a limited set, which is widely used [26] in sequence generation approaches. Here, we use the beam search to prune the low-quality ones down to y_{cand} with width M, which is crucial for the efficiency. The submodular assumption [16] of DPPs is widely used during inference in practice including recommendation systems [6, 10], where one can find the solution in polynomial time. Following Equation (5), we choose the bundle b from y_{cand} one by one maximizing $P(y \cup \{b\})$, which is given by:

$$\begin{aligned} &\arg \max_{b \in y_{\text{cand}}} \log\det S_{y\cup\{b\}} + \sum_{\beta \in y \cup \{b\}} \log q^2(\beta|C_u) \\ \Leftrightarrow &\arg \max_{b \in y_{\text{cand}}} \log p(b|C_u) + \lambda \log\det S_{y\cup\{b\}} \end{aligned} \tag{15}$$

Here, λ is a hyper-parameter controlling the tradeoff between quality and diversity. When we increase λ, we pay more attention to the diversity among the generated bundles than the precision of the final list, and vice versa.

In addition, we need to avoid duplicated items in the bundle, and would like to generate bundles with the larger size in practice. Usually, We add a particular end token of an item to mark the end of bundle generation so that we can produce the various-length bundle. However, BGN uses the co-purchase data or pre-defined bundles to supervise the quality part. The distribution of groundtruth bundle size may not be consistent with expected distribution for target applications. In practice, one may need larger bundle size due to sellers' promotion. Besides, the traditional beam search often has a significant length bias and squeezes the distribution of bundle size to a much more skewed one that prefers shorter size, as we show in Figure 4(b) in our experiments. If we simply select the dataset with larger bundle size when training, it could cause more sparsity issue owing to the reduction of the training set.

For controlling the distribution of bundle size, as well as eliminating duplicated items, we improve the softmax with masks during beam search, which denotes the masked beam search. The masked beam search subtracts a mask vector $m_t \in \mathbb{R}^N$ to the logits for decreasing the probability in beam search during inference, which is given by:

$$\text{masked\_softmax}(h_t, E, m_t)_j = \frac{exp(h_t^T e_j - m_{t,j})}{\sum_{\hat{j}=1}^{N} exp(h_t^T e_{\hat{j}} - m_{t,\hat{j}})} \tag{16}$$

where $m_{t,j}$ is the j-th element of m_t. Notice that, the mask vector reduces the certain unnormalized probabilities exponentially. In the bundle recommendation problem, we set m_t to be:

$$m_{t,j} = \begin{cases} \max(C - t, 0) & \text{j is the end token} \\ +\infty & \text{j has been generated before } t \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

where C is a positive hyper-parameter for shifting the distribution of the bundle size, which gives a threshold for end token. The reduction effect due to the shifting hyper-parameter C decays exponentially with time step and dribbles away after C steps. Besides, the mask vector can avoid the duplicated elements in the bundle. We can set $m_{t,j}$ to be infinity when we definitely don't want to generate the item j at time step t.

The overall algorithm is shown in Algorithm 1, where line 5-15 show the single generation step and line 10-12 show the DPP selection according to Equation (15). Each generation step consists of the stacked LSTM cell, the masked beam search with feature-aware softmax, and the DPP selection. We consider the hidden dimension of LSTM cell is constant here. The time complexity of generating candidate list y_{cand} using beam search is $O(TMN)$, and we usually choose M larger than K to ensure that the candidate set is sufficiently diversified. The time complexity of the DPP selection at each step is $O(MK^\omega)$ where the K^ω term comes from the determinant calculation and ω denotes 2.37. Note that we do not need $O(N^2)$ space to compute and store all item pair's similarity because we only compute S_y rather than entire S. The total time complexity of inference process for BGN is $O(TMN + TMK^\omega)$.

5 EXPERIMENT

We organize the experiment section as follows. First, we clarify the experiment setup, measurement metrics, and all competitors. Second, we illustrate the performance gain of feature-aware softmax

Algorithm 1 Bundle Generation Network during Inference

Input: User context C_u, Similarity function $S(a, b)$, Quality/diversity hyper-parameter λ, Maximum bundle size T, Beam search width M, Output list size K
Output: Diversified bundle list y with size K
$h_0 \leftarrow$ text-CNN(C_u)
$y_0 \leftarrow \varnothing$
Building the weight matrix E of the feature-aware softmax according to Equation (12)
for $t \in [1, ..., T]$ **do**
    $h_t \leftarrow$ Stacked LSTM Cell with Attention(h_{t-1}, y_{t-1}, C_u)
    $y_{\text{cand}} \leftarrow$ Masked Beam Search with FA Softmax(h_t, E, m_t)
    // y_{cand} contains M bundles with maximum size t
    $y_t \leftarrow \varnothing$
    // DPP Selection
    while $|y_t| \neq K$ **do**
        Choose b from y_{cand} according to Equation (15)
        $y_{\text{cand}} \leftarrow y_{\text{cand}} - \{b\}$
        $y_t \leftarrow y_t \cup \{b\}$
    end while
    //y_t has K bundles with maximum bundle size t
end for
return $y \leftarrow y_T$

and the performance impact of the masked beam search. Then, we show the balances the diversity and the precision. Finally, we compare with all competitors to see the advantages of our methods on metrics of precision, diversity, response time for generation and AUC for ranking.

5.1 Experiment Setup

Dataset. We use three public datasets and one industrial dataset: Electro and Clothe are two public datasets of Amazon in [20] where bundles are generated from co-purchase records synthetically. Steam is a public dataset where bundles are extracted from video game distribution network in [22]. Taobao is an industrial dataset where we collect the real sales transaction of bundles from the online bundle list recommendation services. We show the statistics for each dataset in Table 1.

Dataset	Electro	Clothe	Steam	Taobao
Items	63,001	23,033	10,978	104,328
Users	192,403	39,387	88,310	2,767
Categories	801	484	-	3,607
Records	1,689,188	278,677	87,565	54,344
Bundles	327,874	72,784	615	1,236
Average Bundle Size	1.50	2.47	5.73	4.06

Table 1: Statistics of each dataset.

Amazon Dataset[1]. We use two subsets (Electro and Clothe) of Amazon product data in [20], which have already been reduced to satisfy the 5-core property, such that each of the remaining users and items has five reviews. We regard the co-purchase items as a bundle, where items are sorted in terms of their prices in a descendant order. The behavior history of each user can be represented as $(b_1, b_2, ..., b_k, ..., b_K)$ where b_k is a bundle consisting of co-purchase items. We make the first k bundle behaviors of each user as the user context to predict the $k+1$ th bundle behavior in the training set and validation set, where $k = 1, 2, ..., K-2$, and we use the first $K-1$ bundle behaviors to predict the last one in the test set. The average bundle size is 1.50 for Electro and is 2.47 for Clothe, and we notice that many bundles consist of only one item in these two datasets. For alleviating the sparsity of bundle purchase records and keeping enough training records, we consider the bundles with one item is valid in training set. However, in the test set, we only use bundles consisting of more than one items to measure the performance, which isolates the measurement from the single-item list recommendation. The features that we use contain user id, item id, category id, and bundle id. The category is a feature of items.

Steam Dataset[2]. This dataset is collected in [22] from the Steam video game distribution platform. The item is a certain game in this dataset. We partition all purchase records into 70%/10%/20% training/validation/test splits. As in Amazon dataset and Taobao dataset, Some combinations of items in the test set may never be seen in the training set, which leads to the difficulty in generation. Note that there is no feature of the category in the Steam dataset.

Taobao Dataset. This dataset is collected from an online bundle list recommendation service of a commercial e-business, Taobao[3]. Different from the synthetic bundles consisting of co-purchase items in Amazon dataset, the bundles in the Taobao dataset are predefined manually by the sellers. Correspondingly, they have higher quality and longer average bundle size. We conduct the careful collection which guarantees that each user has at least three bundle behaviors. We use the last one of bundle behaviors in the test set and the others in the training set. The user context consists of items which users have purchased. The features that we use contain user id, item id, category id, and bundle id.

Measurements. For the bundle generation problem, the metrics we mainly consider are the top-k precision ***pre@k*** measuring the quality and the cross-bundle diversity ***div***. The *pre@k* is the average precision with the ground truth over each user and each position in the final bundle list. The *div* is defined as 1 − Jaccard Similarity over each pair of the final bundle list.

$$pre@k = \frac{1}{|U|} \sum_{u} \frac{1}{|K|} \sum_{b \in y_u} \frac{|b \cap gt_u|}{|b \cup gt_u|} \tag{18}$$

$$div = \frac{1}{|U|} \sum_{u} \frac{1}{|K|(|K|-1)} \sum_{a,b \in y_u} (1 - \frac{|a \cap b|}{|a \cup b|}) \tag{19}$$

where gt_u is the ground truth bundle that a user buys or clicks. According to the dataset setup above, there is only one bundle in gt_u for the Amazon and Taobao dataset, and there are multiple bundles in gt_u for the Steam dataset. Let y_u be the final recommendation bundle list with size K, U be all users in the test set. These two metrics can be applied to any case no matter whether the bundle is predefined. Note that, the repeated items belonging to different

[1] http://jmcauley.ucsd.edu/data/amazon

[2] http://jmcauley.ucsd.edu/data/steam.tar.gz

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

bundles hit the ground truth and increase the *pre@k*, but they decrease *div*.

For studying whether our method can adapt to the bundle ranking problem, we use the **AUC** to evaluate the performance, which is given by:

$$AUC = \frac{1}{|U|} \sum_{u} \frac{1}{|gt_u|} \sum_{b^+ \in gt_u} \delta(\hat{p_{u,b^+}} > \hat{p_{u,b^-}}) \tag{20}$$

where $\hat{p_{u,b}}$ is the predicted probability that a user may act on a bundle and $\delta(\cdot)$ denotes an indicator function. Note that, b^- is sampled uniformly from all valid bundles in the dataset, which are all co-purchase records consisting of more than two items for the Amazon dataset and all pre-defined bundles for the Steam and Taobao dataset.

Competitors. We mainly consider six methods as below:

- **BGN**. This denotes our Bundle Generation Network method. For CNN encoder of user context, we use various filters with the window size of $\{1, 2, 4, 8, 12, 16, 32, 64\}$, and each filter contains 12 output channels, so the final dimension of output channel is 96. The other latent dimension is set to be 64. We use two layers' stacked LSTM cell with attention in the decoder. The batch size is set to be 16. We use l2-loss in the training process, and the weight is set to be 5e-5. Adam is adopted as the optimizer. The beam search width is set to be 50. The sample number of sampled feature-aware softmax is set to be 1024. We use Equation (8) as the similarity function. λ is the quality/diversity hyper-parameter ranging from 0.0 to 5.0 with step 0.25. C is the shifting hyper-parameter controlling the distribution of bundle size, which ranges from 0 to 20 with step 1.0.
- **Freq**. This is a well-known frequent itemsets mining algorithm introduced in [2], and we generate bundles according to frequent itemsets discovered in co-purchase or pre-defined bundles. Then we recommend the most frequent k bundles to users. Note that this is not a personalized method.
- **BRP**. *BRP* (Bundle Recommendation Problem) [36] is a personalized single bundle recommendation algorithm, which optimizes the expected reward based on Mixed-Integer Quadratic Programming (MIQP). We use matrix factorization as the probability model which has the best performance in the original paper, and use Gurobi[4], one of the fastest public mathematical programming solver, to optimize the MIQP problem.
- **BBPR**. *BBPR* (Bundle Bayesian Personalized Ranking) is proposed by [22]. It builds the optimization criterion upon the Bayesian Personalized Ranking(BPR) [24], with the mean pair-wise Pearson correlation of the items. Combined with traditional matrix factorization techniques, it utilizes both item and bundle information including bundle size and item compatibility for better performance. As for the bundle generation, it proposes a greedy algorithm following an annealing schedule to give personalized bundles, which avoids ranking for all possible bundles. In the Steam dataset, we maintain the setup of typer-parameters as the original paper, and in other datasets, we deploy a small range of fine-tuning and report the best results.
- **RankAll**. *RankAll* ranks all the existing bundles in the training set and returns the top-K results, and we consider it as a strong baseline to validate whether the ranking methods are good enough for the bundle generation task regardless of the consumption of time and space. Note that, *RankAll* refers to rank all the existing bundles in the training set, instead of ranking all possible bundles in $\mathcal{B}$, which is too large to enumerate even for modern machines. Specifically, *RankAll* adopts the same loss as *BBPR*, but utilizes two text-CNN networks to extract the user context and the bundle information respectively. The hyper-parameter setting of text-CNN is agree with *BGN*. Ranking all the existing bundles in the training set is time-consuming and usually is unacceptable in practice.

Environment. All experiments are conducted on a machine with one GPU (NVIDIA Tesla P100 16G) and 96 CPUs (Intel Xeon Platinum 2.50GHz). We use Tensorflow [5] as the main toolkit of neural network.

5.2 Feature-Aware Softmax Performance Gain

The feature-aware softmax combines more features besides the item_id in the weight matrix of softmax. We demonstrate the performance gain of all datasets in Table 2, where 'w/o FA' denotes 'without feature-aware' and 'w/ FA' denotes 'with feature-aware' for short.

Dataset		Electro	Clothe	Steam	Taobao
pre@10	BGN w/o FA	0.32%	0.13%	-	1.75%
	BGN w/ FA	0.60%	0.38%	9.65%	2.82%
pre@5	BGN wo/ FA	0.38%	0.12%	-	1.83%
	BGN w/ FA	0.67%	0.43%	14.96%	3.09%

Table 2: Feature-Aware (FA) softmax Performance Gain

In Table 2, we observe that the proposed feature-aware softmax can improve the precision significantly, which yields the improvements on *pre*@10 by 0.87x, 1.92x and 0.61x on Electro, Clothe and Taobao dataset respectively. The gain brought by feature-aware softmax indicates that, compared to the traditional softmax, the feature-aware softmax considers more features in the weight matrix and alleviates the sparsity issue of item_id effectively in bundle generation. The performance gain also verifies the importance of features for recommendation tasks as in [8]. Note that there is no feature of categories in the Steam dataset, so there is no difference with feature-aware softmax.

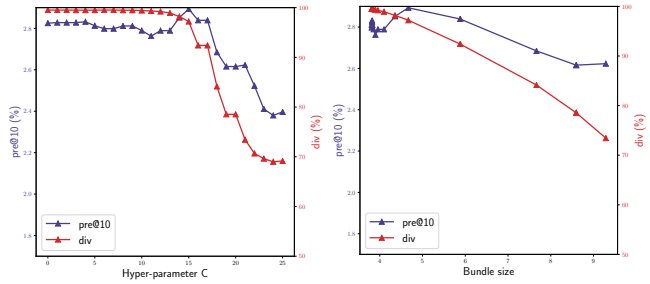
Note that all competitors except *Freq* consider the same complete features as BGN, but BGN with feature-aware softmax performs better because we factorize the bundle probability as a sequence of conditional probability, which is equivalent to optimizing generalized BPR loss directly at each step as shown in Equation (14).

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

[5] https://tensorflow.org

5.3 Performance Impact due to controlling the Bundle Size

The distribution of the training data may not be consistent with the distribution of the bundle size for the target application. According to the masked beam search, we study the performance impact due to controlling the Bundle Size.



(a) Hyper-parameter C (b) Bundle Size

Figure 3: Performance Impact of the Masked Beam Search (Taobao)

The masked beam search controls the distribution of bundle size by changing a shifting typer-parameter C according to Equation (17), which reduces the unnormalized probabilities exponentially of generating the end token. We show in the Taobao dataset how the distribution of bundle size is altered in Figure 4 and the effect on the precision and diversity in Figure 3.

In Figure 4, we notice that the masked beam search does shift the distribution of bundle size. The distribution shifts significantly compared to the original beam search when setting C to be 14, and with the increase of C the average bundle size shifts as we expected. However, the shifting parameter C enforces the network to generate the most likely items, although the generation should have ended. As shown in Figure 3, C is set from 0 to 20 with step 1.0, which leads to the bundle size varying from 3.8 to 9.2, then we show the performance impact about the precision and diversity respectively for the C and bundle size. We notice that the precision still holds when we shift the bundle size from 3.8 to 6.0, but both the precision and diversity decrease if we continue shifting. Though our shifting strategy is simple, it is good enough to control the distribution in a small range in practice and it might be promoted in the future work through considering the long-term reward in advance by reinforcement learning.

5.4 Balance Between Precision and Diversity

According to Algorithm 1, we produce the diversified bundle list by combining the beam search and DPPs, which chooses the item maximizing the objective function of the quality and diversity in Equation (15). The hyper-parameter λ balances between the precision and the diversity. The metric of diversity is a separate metric from CTR and precision, trying to quantify the user-experience in a bundle recommendation system. When we increase λ, we pay more attention to the diversity among the generated bundles than the precision of the final list, and vice versa. We select K bundles from M candidates generated from the beam search with $M = 50$, $K = 10$ at each step, and λ is set from 0 to 5 with step 0.25 and C is set to be 0.

We show in Figure 5 that in all datasets, the precision decreases along with the growth of diversity as we expected. We also mark the other competitors' results as points in the figure, and we omit the competitors whose precision is too poor. Feature-aware softmax and beam search guarantee that BGN has a high precision base, meanwhile, DPP selection ensures to improve the diversity steadily with a tolerable reduction in precision. With adjustment of the λ, the curve formed by BGN is completely at the top right of all competitors. We achieve the highest precision and diversity in contrast with all competitors in all datasets.

5.5 Comparison in the Bundle Generation Problem

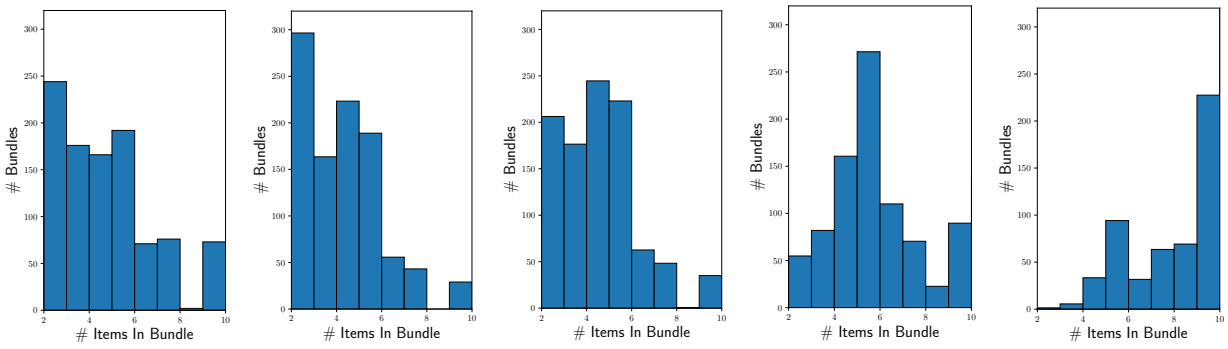
By comparing the proposed BGN against the other competitors including, *Freq*, *BRP*, *BBPR*, *RankAll*, *Seq2seq*, we report and analyze their performance in terms of the precision metric *pre@k*, the diversity metric *div*, and the average response time per user of these methods during inference, denoted by *avg-time* in Table 3. Note that the shifting hyper-parameter C of BGN is set to be 0 in this part becasue we have shown the performance results of shifting the bundle size in Figure 3.

Freq performs poorly in term of precision on all datasets, because it is a non-personalized method, but *Freq* performs quite diverse in Amazon dataset. We think that it is because the bundles of two Amazon datasets are generated synthetically by co-purchase records, which contain more diversified bundles naturally, whereas, the bundles of Steam and Taobao dataset is pre-defined by sellers.

BRP produces only one bundle, so there is no diversity reported. We repeat the results K times to produce the list for achieving the higher precision, which is well-defined because our metric of precision averages on each position. With regard to precision, *BRP* performs better than *Freq* owing to the personalization and consideration of the the first and second cross-item dependencies, but it ignores the higher order terms and gives a sub-optimal result. Besides, compared with other methods, solving the mixed-integer quadratic programming (MIQP) problem is time-consuming.

BBPR produces a personalized list of bundles instead of one. This method utilizes both item and bundle information including bundle size and the pair-wise Pearson correlation of the items, which performs better than *Freq* and *BRP* on most of the dataset. However, the annealing schedule leads to randomness in generating bundles and is easily caught in the local optimum due to instability of annealing.

RankAll performs better than *BBPR* and *BRP* on the metrics of precision and diversity on all datasets, because it uses CNN as a context extractor capturing more combinatorial generalization of features. But for the response time, *RankAll* is the most time-consuming approach on almost all datasets. Because the inference time of *RankAll* is approximately proportional to the number of bundles, whereas, other methods are approximately proportional to the number of items. Essentially, *RankAll* is not a generative model, but a ranking model which ranks the existing bundles to give the final recommendation list. It wastes most of the time scoring the low-quality bundles and it is not scalable in practice. Furthermore, it



(a) Groundtruth (b) Original Beam Search (c) C=14 (d) C=17 (e) C=20

Figure 4: Bundle Size Distribution Shifting by the Masked Beam Search (Taobao)

Dataset	Synthetic Bundle						Real bundle					
	Electro			Clothe			Steam			Taobao		
	pre@10	*div*	*avg-time*	*pre@10*	*div*	*avg-time*	*pre@10*	*div*	*avg-time*	*pre@10*	*div*	*avg-time*
Freq	0.260%	95.56%	-	0.096%	94.07%	-	5.831%	69.44%	-	0.192%	60.00%	-
BRP	0.324%	-	4.85s	0.149%	-	3.96s	6.750%	-	0.44s	0.332%	-	5.47s
BBPR	0.308%	95.87%	0.68s	0.122%	85.56%	0.34s	7.168%	87.32%	0.29s	0.357%	95.49%	1.27s
RankAll	0.443%	88.87%	55.97s	0.280%	89.23%	7.94s	9.497%	97.63%	0.11s	1.540%	98.19%	1.38s
BGN ($\lambda = 0$)	**0.607%**	87.22%	**0.05s**	**0.387%**	81.63%	**0.03s**	**9.655%**	98.01%	**0.04s**	**2.825%**	99.50%	**0.24s**
BGN ($\lambda = 5$)	0.469%	**96.68%**	0.09s	0.235%	**95.75%**	0.06s	9.617%	**99.08%**	0.06s	2.739%	**99.88%**	0.28s

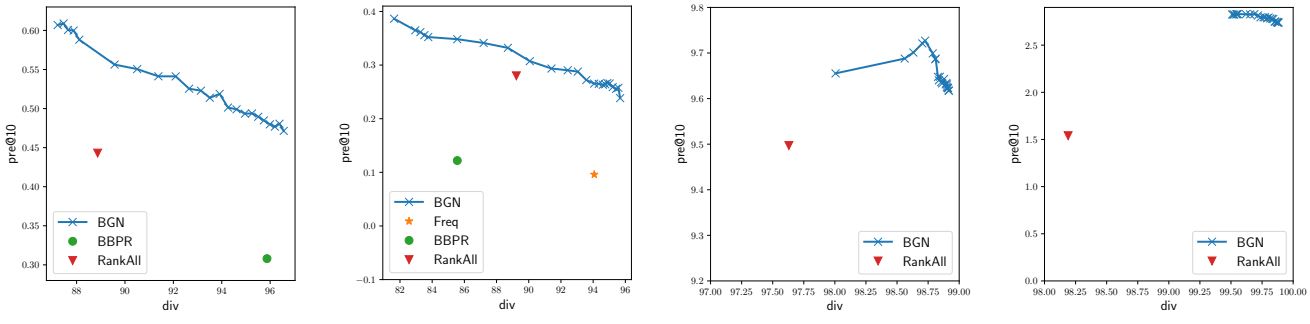
Table 3: *pre*@10, *div*, *avg-time* of Different Methods in the Bundle List Recommendation Problem.

could not produce new bundles beyond the existing ones in training data, which limits the precision.

BGN achieves the best precision, diversity and fastest response time at the same time on all datasets. The high precision benefits from the the feature-aware softmax. However, when setting λ to be 0, BGN has low diversity especially in the synthetic bundle because traditional seq2seq is easy to generate similar sentences. When we simply set the λ to be 5, BGN balances the quality and diversity, which improves the precision of the best baselines by 16% on average while maintaining the highest diversity on four datasets. Besides, BGN is a sequnce generation approach without ranking or recursive searching. The time complexity of BGN during inference is proportional to the number of items. So our model achieves fastest response time in bundle list recommendation problem. Specifically, BGN improves the response time by 6.5x, 4.6x, 0.8x, 3.5x on four datasets respectively over the best competitors when λ equals 5.

5.6 Comparison in the Bundle Ranking Problem

BGN is designed for the bundle list generation problem. However, we can adopt it in the bundle ranking problem, by feeding the candidate bundles to the inputs of the decoder and taking the negative loss as the ranking score, so that we compare with other competitors in the bundle ranking problem. Note that, the absolute values of the AUC in the bundle ranking problem are usually lower than the one in the item ranking problem because the sample space for bundles is larger and more sparse.



(a) Electro (b) Clothe (c) Steam (d) Taobao

Figure 5: Balance between Precision and Diversity.

AUC	Electro	Clothe	Steam	Taobao
BBPR	74.85%	66.20%	90.27%	60.24%
RankAll	75.51%	66.97%	**99.66%**	60.93%
BGN	**78.11%**	**69.99%**	99.36%	**61.27%**

Table 4: AUC Tests in the Bundle Ranking Problem.

In Table 4, we show the measurement of AUC for different methods, and BGN improves the best baseline by 2.04% on average on four datasets. Actually, in the bundle ranking setup, all competitors in Table 4 can be regarded as the encoder-decoder framework, where the encoder extracts the user context information and the decoder extracts the bundle information. The major differences are the architectures of specific neural network and the loss function. With the help of the feature-aware softmax, BGN is good at utilizing the feature information, because the weight matrix of softmax is built dynamically and its loss is a generalization of optimizing the average BPR at each step. The Steam dataset has no category feature, so BGN is worse than *RankAll* slightly on this dataset.

6 CONCLUSIONS

This paper studies a problem of personalized bundle list recommendation and proposes a bundle generation network (BGN). BGN decomposes the problem into the quality/diversity part to produce the high-quality and diversified bundle list. The proposed feature-aware softmax makes up the inadequate representation of traditional sequence generation for rich features in bundle recommendation scenario, which improves the modeling of quality significantly. We conduct extensive experiments on three public datasets and one industrial dataset. BGN achieves the best state-of-the-art results on the metrics of precision, diversity and response time on all datasets. Besides, BGN can control the bundle size with a tolerable reduction in precision.

ACKNOWLEDGMENTS

This work was partially supported by National Key Research and Development Program No. 2016YFB1000700, NSFC under Grant No. 61572040 and 61832001, and Alibaba-PKU joint Program.

REFERENCES

[1] Raja Hafiz Affandi, Emily Fox, Ryan Adams, and Ben Taskar. 2014. Learning the parameters of determinantal point process kernels. In *International Conference on Machine Learning*. 1224–1232.

[2] Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, Vol. 1215. 487–499.

[3] David Belanger and Andrew Mccallum. 2015. Structured prediction energy networks. (2015), 983–992.

[4] Mathieu Bouchard, Anne-Laure Jousselme, and Pierre-Emmanuel Doré. 2013. A proof for the positive definiteness of the Jaccard index matrix. *International Journal of Approximate Reasoning* 54, 5 (2013), 615–626.

[5] Wei-Lun Chao, Boqing Gong, Kristen Grauman, and Fei Sha. 2015. Large-Margin Determinantal Point Processes.. In *UAI*. 191–200.

[6] Laming Chen, Guoxin Zhang, and Hanning Zhou. 2018. Improving the Diversity of Top-N Recommendation via Determinantal Point Process. *Advances in Neural Information Processing Systems* (2018).

[7] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[8] Heng-Tze Cheng, Levent Koc, and Jeremiah Harmsen. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS 2016)*. 7–10. https://doi.org/10.1145/2988450.2988454

[9] Ting Deng, Wenfei Fan, and Floris Geerts. 2013. On the complexity of package recommendation problems. *SIAM J. Comput.* 42, 5 (2013), 1940–1986.

[10] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2016. Bayesian low-rank determinantal point processes. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 349–356.

[11] Jennifer A Gillenwater, Alex Kulesza, Emily Fox, and Ben Taskar. 2014. Expectation-maximization for learning determinantal point processes. In *Advances in Neural Information Processing Systems*. 3149–3157.

[12] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007* (2014).

[13] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[14] Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872* (2017).

[15] Alex Kulesza and Ben Taskar. 2010. Structured determinantal point processes. In *Advances in neural information processing systems*. 1171–1179.

[16] Alex Kulesza, Ben Taskar, et al. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning* 5, 2–3 (2012), 123–286.

[17] Hui Lin and Jeff A Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871* (2012).

[18] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware sequential recommendation. *arXiv preprint arXiv:1609.05787* (2016).

[19] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[21] Aditya Parameswaran, Petros Venetis, and Hector Garcia-Molina. 2011. Recommendation systems with complex constraints:A course recommendation perspective. *Acm Transactions on Information Systems* 29, 4 (2011), 1–33.

[22] Apurva Pathak, Kshitiz Gupta, and Julian Mcauley. 2017. Generating and Personalizing Bundle Recommendations on Steam. In *The International ACM SIGIR Conference*. 1073–1076.

[23] Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2016. Recommending packages to groups. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 449–458.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[25] Yiping Song, Rui Yan, Yansong Feng, Yaoyuan Zhang, Dongyan Zhao, and Ming Zhang. 2018. Towards a Neural Conversation Model With Diversity Net Using Determinantal Point Processes.. In *AAAI*.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[28] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*. 2692–2700.

[29] Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869* (2015).

[30] Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960* (2016).

[31] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. 2016. Personal recommendation using deep recurrent neural networks in NetEase. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, 1218–1229.

[32] Min Xie, Laks VS Lakshmanan, and Peter T Wood. 2014. Generating top-k packages via preference elicitation. *Proceedings of the VLDB Endowment* 7, 14 (2014), 1941–1952.

[33] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 425–434.

[34] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2017. ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. *CoRR* abs/1711.06632 (2017). arXiv:1711.06632 http://arxiv.org/abs/1711.06632

[35] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences* 107, 10 (2010), 4511–4515.

[36] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. 2014. Bundle recommendation in ecommerce. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 657–666.