Adaptive In-Context Learning with Large Language Models for Bundle Generation

Zhu Sun
A*STAR Centre for Frontier AI Research; Singapore University of Technology and Design
Singapore, Singapore

Kaidong Feng*
Yanshan University
Qinhuangdao, China
kaidong3762@gmail.com

Jie Yang
Delft University of Technology
Delft, the Netherlands

Xinghua Qu
Shanda AI-Lab; Tianqiao and Chrissy Chen Institute
Singapore, Singapore

Hui Fang
Shanghai University of Finance and Economics
Shanghai, China

Yew-Soon Ong
A*STAR Centre for Frontier AI Research; Nanyang Technological University
Singapore, Singapore

Wenyuan Liu
Yanshan University
Qinhuangdao, China

ABSTRACT

Most existing bundle generation approaches fall short in generating fixed-size bundles. Furthermore, they often neglect the underlying user intents reflected by the bundles in the generation process, resulting in less intelligible bundles. This paper addresses these limitations through the exploration of two interrelated tasks, i.e., personalized bundle generation and the underlying intent inference, based on different user sessions. Inspired by the reasoning capabilities of large language models (LLMs), we propose an adaptive in-context learning paradigm, which allows LLMs to draw tailored lessons from related sessions as demonstrations, enhancing the performance on target sessions. Specifically, we first employ retrieval augmented generation to identify nearest neighbor sessions, and then carefully design prompts to guide LLMs in executing both tasks on these neighbor sessions. To tackle reliability and hallucination challenges, we further introduce (1) a self-correction strategy promoting mutual improvements of the two tasks without supervision signals and (2) an auto-feedback mechanism for adaptive supervision based on the distinct mistakes made by LLMs on different neighbor sessions. Thereby, the target session can gain customized lessons for improved performance by observing the demonstrations of its neighbor sessions. Experiments on three real-world datasets demonstrate the effectiveness of our proposed method.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Neural networks**.

*corresponding author

SIGIR '24, July 14–18, 2024, Washington, DC, USA

ACM ISBN 979-8-4007-0431-4/24/07
https://doi.org/10.1145/3626772.3657808

KEYWORDS

Recommendation, Bundle Generation, User Intent Inference, Large Language Models, In-Context Learning

ACM Reference Format:
Zhu Sun, Kaidong Feng, Jie Yang, Xinghua Qu, Hui Fang, Yew-Soon Ong, and Wenyuan Liu. 2024. Adaptive In-Context Learning with Large Language Models for Bundle Generation . In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3626772.3657808

1 INTRODUCTION

Product bundling has evolved into a crucial marketing strategy for promoting products, catering to both physical and online retailers [46, 58]. A bundle refers to a group of products recommended or sold together as a package. These products are bundled together due to various reasons, e.g., having complementary or alternative relationships [2, 7]. For instance, as depicted in Figure 1, if a customer is shopping for a camera, a bundle recommendation may include not only the camera itself but also accessories like lenses, camera bags, tripods, and memory cards - all packaged together at a discounted price. Therefore, product bundling can offer a beneficial solution for both customers and businesses. On the one hand, it facilitates the discovery of new items, prevents the formation of filter bubbles, and presents opportunities for potential promotions, ultimately enhancing the long-term customer experience. On the other hand, it can significantly increase product sales and drive business revenue, promoting overall economic growth within societies [49].

Given the substantial benefit, a growing body of research can be found on exploring product bundling. Most of the studies assume the pre-existence of bundles and directly dive into downstream tasks, e.g., bundle recommendation. In particular, they take co-consumed products [30] or user-generated lists [7, 20, 21] as synthetic bundles, or rely on manually pre-defined bundles by retailers [13, 17, 33, 36]. However, the co-consumed products may not always reflect common intents; user-generated lists are generally limited to specific domains (e.g. music and books); and pre-defined bundles are restricted by quantity and diversity due to the high cost of producing such bundles [49].

Figure 1: Example bundles for (1) a camera and its accessories; and (2) mystery, thriller, and historical fiction.

Recognizing the demand for high-quality bundles, some research endeavors to develop methods for bundle generation. Early works create bundles by specifying hard constraints. The generated bundles may (1) possess a limited budget or maximum savings/customer adoption/expected revenue [5, 16, 58–60, 74]; or (2) contain compatible products in related categories, style, or functionality [27]. Recently, deep learning (DL) based methods [3, 8, 54] have emerged to learn the latent association of products for bundle generation. Nevertheless, these methods exhibit certain limitations, for instance, they can only form either fixed-size bundles or are limited to producing a single bundle for each user. Most importantly, they overlook the requirement for a consistent user intent underlying all products in the same bundle: products in a high-quality bundle should all reflect the same user intent that is semantically interpretable, describing a consistent user need (or purpose) in interacting with products (e.g., camera accessories, clothes for parties, or movies for Friday nights). Without such a constraint of intents, the generated bundles become less relevant and intelligible to users, thereby failing to meet their actual needs in applications.

To fill the gap, we propose simultaneously performing two interrelated tasks, i.e., *generating personalized bundles* and *inferring underlying intents* from user sessions[1]. This is motivated by the fact that users are inclined to explore relevant products, either as alternatives or complements, based on their specific intents during a session [28, 51]. In doing so, (1) user sessions can serve as valuable sources to create high-quality and personalized bundles; (2) the inferred user intents can enhance the interpretability of bundles, just as a well-defined bundle can clearly reflect the user's intent. However, performing both tasks at the semantic level poses significant challenges as it entails comprehending various potential motivations and contexts behind user actions and preferences, which can be intricate and ever-evolving.

To tackle the above challenges, we devise an adaptive in-context learning (AICL) paradigm leveraging the advanced reasoning capabilities of large language models (LLMs)[2]. This paradigm empowers LLMs to draw tailored lessons from closely related tasks, using them as demonstrations while tackling the target task. Specifically, we first adopt the retrieval augmented generation [26] to identify the nearest neighbor sessions for each target session, and then create prompts to instruct LLMs to perform both tasks in neighbor sessions. To enhance reliability and mitigate the hallucination issue, we further develop (1) a self-correction strategy to foster mutual improvements in both tasks without supervision signals; and (2) an auto-feedback mechanism to recurrently offer adaptive supervision by comparing LLMs' output and the labels. Subsequently, we guide LLMs to provide a summary of rules derived from the entire task execution process to prevent recurring errors in the future. Finally, the two tasks in the target session are performed by observing demonstrations of its neighbor sessions. Different neighbors may possess distinct mistakes made by LLMs, thereby receiving different feedback. It thus enables LLMs to seek adaptive and customized lessons for improved performance on the target session.

Our contributions are three-fold. **First**, we propose a new research question to perform two interrelated tasks, i.e., bundle generation and intent inference, based on user sessions. As such, the generated bundles are more intelligible and aligned with users' actual needs. **Second**, we design a novel adaptive in-context learning paradigm for our defined tasks, which enables LLMs to seek tailored lessons from neighbor sessions as demonstrations. To achieve this, we devise step-by-step strategies evolving from mutual self-correction (self-supervision) to adaptive auto-feedback (auto external-supervision), and finally rules summarization (self-supervision). This is a novel idea in the context of using LLMs for recommendation. **Lastly**, we conduct experiments on three public datasets. The results show that AICL surpasses baselines on the bundle generation task, and the inferred intents are of high quality, comparable to or even exceeding those annotated by humans.

2 RELATED WORK

2.1 Recommendation with Prebuilt Bundles

Many works assume the presence of prebuilt bundles and immediately delve into the downstream task of bundle recommendation. Early methods generate bundles by satisfying certain constraints, e.g., limited cost [17, 33]. Later, factorization-based methods decompose user-item and user-bundle interactions to learn users' interests over items and bundles, respectively [6, 35]. Recently, DL-based methods (e.g., DAM [9], AttList [20], CAR [21], BRUCE [2], and BundleGT [55]) adopt the attention mechanism to learn item-bundle affinity and user-bundle preference. Other methods adopt graph or hypergraph convolutional networks to better infer users' preference towards bundles, such as BGCN [7], BundleNet [13], CrossCBR [36], MIDGN [71], UHBR [65], SUGER [70], and DGMAE [42]. However, these methods are all based on prebuilt bundles, i.e., either co-consumed products, user-generated lists, or predefined ones by retailers as summarized in Table 2. They ignore the fact that (1) co-consumed products may not consistently represent shared intentions; (2) user-generated lists are typically confined to specific domains (e.g., music and books); and (3) pre-defined bundles are constrained by their limited quantity and diversity, primarily due to the high production costs associated with them.

2.2 Recommendation with Bundle Generation

Several studies explore bundle recommendation with generation. In the early stage, bundles are created via frequent itemsets mining algorithm [1, 49]. Later, they are formed by adhering to specific hard constraints. For instance, greedy-based methods create bundles by minimizing the cost [16, 58], or fulfilling other requirements [38, 39]. Heuristic methods form bundles by maximizing customer adoption [60], expected revenue [5], or sharing the same category [27]. Preference elicitation methods produce bundles via users' preference for cost and quality [14, 59]. Others frame bundle

[1]A user session is a sequence of actions (e.g., clicks, purchases) performed by a user on a platform or website with the products during a short period (e.g., a single visit) [51, 56].
[2]We use GPT-3.5-turbo in our study without further statement.

Table 1: Approaches with bundle generation. '?' means the answer is not found based on the paper and source code if available.

	[1]	[49]	[16]	[58]	[39]	[38]	[60]	[5]	[59]	[14]	[74]	[27]	[3]	[10]	[54]	[12]	[23]	[8]	Ours
Dynamic Bundle Size	✘	✘	✔	✔	✔	✔	✘	✘	✘	✔	✘	✘	✘	✔	?	✘	✔	✔	✔
Multiple Bundles	✔	✔	✘	✔	✘	✘	✔	✔	✔	✘	✘	✘	✔	✘	✘	✘	✘	✔	✔
Personalized Bundles	✘	✘	✔	✔	✔	✔	✔	✔	✔	✔	✔	✘	✔	✔	✔	✔	✔	✔	✔
Intent Inference	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✔

Table 2: Approaches with different prebuilt bundles.

Type	Methods
Co-consumed Products	[30]
User Generated Lists	[2, 6, 7, 9, 13, 20, 21, 35, 36, 42, 55, 65, 70, 71]
Predefined by Retailers	[2, 13, 17, 33, 36, 42, 55]

generation as a Quadratic Knapsack Problem [74] to maximize the expected reward. Recent studies mainly resort to DL techniques for bundle generation, such as Seq2Seq based methods [3, 10, 54], and graph-generation based method [8]. Other works treat it as combinatorial optimization [12] or Markov Decision Process [23] and adopt reinforcement learning to compose bundles. However, they suffer from various drawbacks (see Table 1): some methods can only generate fixed-size bundles or a single bundle for each user [39, 74]; some overlook personalization [1, 27]; and others exhibit high complexity and limited scalability [8]. Most importantly, they generally fail to understand user intent at the semantic level during bundle generation. Consequently, the created bundles may be less comprehensible and aligned with users' actual needs.

2.3 Intent-Aware Session Recommendation

Our proposal of simultaneously generating personalized bundles and inferring underlying intents from user sessions is related to intent-aware session recommendation [28, 51]. Specifically, early work learns the main intent in a session to help infer user preference [28, 34, 64]. However, only learning the main intent may limit the model performance, as items in a session may often reveal multiple intents. Hence, later works capture multiple intents in a session [29, 50, 51]. However, they can only learn a fixed number (one or multiple) of latent intents in the session, which is overly rigid and cannot faithfully unveil user intents in a session. In contrast, our study aims to generate an adaptive number of bundles and underlying intents at the semantic level based on user sessions. Closet to our work is the method proposed in [73] that can learn multiple user intents in a session; its assumption, i.e., items belonging to the same category indicate the same intent, however, may not always hold in reality. Our method instead, allows to generate multiple intents of any type, not restricted to item categories.

2.4 LLMs for Recommendation

The remarkable achievements of LLMs have led to their widespread adoption for more effective recommendation [18, 57, 68]. Many works adopt *in-context learning* (ICL) to align LLMs for recommendation. For instance, Zhai et al. [66] transform knowledge graphs into knowledge prompts for more explainable recommendation. Other studies [11, 22, 45] highlight ChatGPT's potential to mitigate the cold start issue and provide explainable recommendations. Another line of research exploits *parameter-efficient fune-tuning* (PEFT) to align LLMs for recommendation, such as TallRec [4], PALR [63], InstructRec [69], and HKFR [61]. Despite the effectiveness of these LLM-based methods, they are all designed for individual item recommendation. On the contrary, our study attempts to leverage the capability of LLMs through ICL for personalized bundle (a set of associated items) generation and underlying intent reasoning. Instead of using randomly sampled few-shot examples [45], we employ the retrieval augmented generation to retrieve the dataset and identify the most correlated neighbor sessions. On this basis, we create demonstrations via the proposed self-correction and auto-feedback strategies. This process empowers LLMs to take customized and adaptive lessons from neighboring sessions, ultimately leading to enhanced performance in the test session.

2.5 Prompting Methods for LLMs

The fact that LLMs have seen extensive application across diverse tasks and domains has also made a strong impact in the research communities, where an increasing amount of research work is being found on designing prompts for LLMs utilization. Many advanced prompting methods have been introduced to guide LLMs in generating more specific, accurate and high-quality responses, such as Chain-of-Thought [53], Tree-of-Thought [62], Self-Consistency [52], Self-Reflection [37], Generated Knowledge [31], Least-to-Most [72] and Retrieval Augmentation [26]. These methods offer promising opportunities in recommendation contexts, yet come also with the crucial challenge of creating appropriate prompts that are tailored to specific recommendation tasks. In response to this challenge, our study introduces novel strategies specifically designed to leverage LLMs effectively for bundle generation and intent inference.

3 THE PROPOSED METHODOLOGY

We design an adaptive in-context learning (AICL) paradigm for LLMs to simultaneously perform two interrelated tasks: *generating bundles* and *inferring underlying intents*, from user sessions. This is motivated by the fact that users tend to explore highly correlated products, either alternatives or complements, based on their specific intents during a session [51]. The two tasks defined can be mutually reinforced and enhanced. Specifically, effective user intents can help identify relevant products to form improved bundles and enhance interpretability. Meanwhile, a well-defined bundle, in turn, can provide a clearer elucidation of the user's intent.

Model Overview. Our core idea is to enable LLMs to seek tailored and adaptive lessons from closely related tasks as demonstrations while performing the target task. This is different from existing ICL-based recommendation methods [11, 45], which rely on randomly sampled examples and static instructions. Our AICL mainly consists of three modules as shown in Figure 2.

- *Neighbor Session Retrieval* exploits the retrieval augmented generation [26] to identify from the entire dataset the most correlated sessions for each target session regarding products contained. Such neighbor sessions will be used to generate demonstrations.

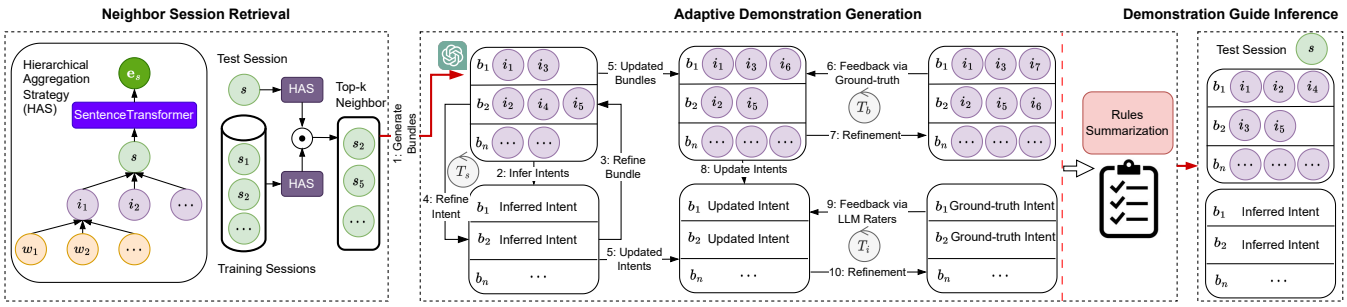

Figure 2: The overall framework of AICL. We take one test session and its top-1 neighbor session as an example for illustration.

- *Adaptive Demonstration Generation* creates prompts to instruct LLMs to perform both tasks in neighbor sessions. To enhance reliability and mitigate the hallucination issue, we develop a self-correction strategy to foster mutual improvements in both tasks without the need for any supervision signals. Afterwards, an auto-feedback mechanism is devised to recurrently provide adaptive supervision by comparing the outputs of LLMs and the ground truth (labels). Finally, it directs LLMs to provide a summary of rules derived from the entire task execution process to prevent recurring errors in the future.
- *Demonstration Guided Inference* observes demonstrations of neighbor sessions to perform the two tasks in the corresponding target session. Different neighbor sessions may encounter distinct mistakes/errors made by LLMs, thereby receiving diverse feedback. Hence, it empowers LLMs to seek tailored and adaptive lessons for improved performance on the target session.

Equipped with the three modules, our AICL paradigm is capable of effectively creating multiple, personalized, and intelligible bundles with adaptive sizes given any user session.

3.1 Neighbor Session Retrieval (NSR)

Existing ICL-based recommendation methods [11, 32, 45] randomly sample few-shot examples to instruct LLMs (e.g., ChatGPT) for the inference process. However, when the sampled examples are less relevant to the target task, LLMs receive only a restricted amount of useful knowledge to guide them [24]. To resolve this issue, the Neighbor Session Retrieval module (NSR) employs the retrieval augmented generation [26] to retrieve the entire dataset and identify the most correlated sessions (i.e., nearest neighbor sessions) for each target session regarding products contained. By doing so, we can acquire a wealth of valuable knowledge and lessons that can significantly enhance performance.

To this end, we devise a hierarchical aggregation strategy to get the representation of each session. First, for each item, we process its title via the natural language processing toolkit - NLTK (nltk.org) and regular expressions, to remove stop words and special characters (e.g., &, #), etc. The processed item title is treated as its description denoted as $i \leftarrow [w_1, w_2, \cdots]$, where w_x means an individual word. Then, we concatenate descriptions of items within one session to represent its session description denoted as $s \leftarrow [i_1, i_2, \cdots]$. Subsequently, we feed the session description into SentenceTransfomers (all-MiniLM-L6-v2) [41] to get its latent representation. given by, $\boldsymbol{e}_s = \text{SentenceTransformer}(s)$. Finally, for each target (test) session, we calculate its cosine similarity with all training sessions using the learned latent representations, to identify its top-k nearest neighbors. Such neighbor sessions are used for generating demonstrations to enhance LLMs' performance on the test session.

It is noteworthy that one may consider using the embeddings (i.e., representations) generated by LLMs (e.g., OpenAI text-embedding-ada-002) to calculate the similarity. Instead, we choose SentenceTransformers due to three aspects. (1) **Lower Time Complexity**. The dimension of the embedding output by SentenceTransformer (all-MiniLM-L6-v2) is 384, whereas the dimension output by OpenAI (text-embedding-ada-002) is 1536. The smaller embedding size will greatly reduce the time complexity for finding the nearest neighbors. (2) **Less API Usage Cost**. SentenceTransformer is an open-source Python framework, whereas we have to pay when using LLMs such as OpenAI (text-embedding-ada-002). In comparison, SentenceTransformer helps save much financial cost, especially for large-scale session datasets. (3) **Comparable Performance**. SentenceTransformers can obtain comparable performance as LLMs such as OpenAI (text-embedding-ada-002), which can be verified by the experimental results in Section 4.2.3.

3.2 Adaptive Demonstration Generation (ADG)

Next, ADG designs proper prompts to instruct LLMs to perform both tasks (i.e., bundle generation and intent reasoning) on these neighbor sessions, with the goal of creating demonstrations for improved performance on the target session.

First, a prompt that asks LLMs to generate bundles is created as - *A bundle can be a set of alternative or complementary products that are purchased with a certain intent. Please detect bundles from a sequence of products. Each bundle must contain multiple products. Here are the products and descriptions: {[product X: title, . . .]}. The answer format is: {'bundle number':['product number']}. No explanation for the results.* As a result, LLMs will generate bundles and output them in the requested format. Subsequently, another prompt is passed to LLMs to infer the intent behind each generated bundle as - *Please use 3 to 5 words to generate intents behind the detected bundles, the output format is: {'bundle number':'intent'}.* Note that we adopt the average number of words in the ground truth intents (i.e., 3.4) as a constraint to prevent overly long intents. To further enhance the reliability and mitigate the hallucination issue, we design the following strategies for more robust performance.

3.2.1 Mutual Self-Correction. Given the generated bundles and intents, we design a self-correction strategy to foster mutual improvements in both tasks without the need for any supervision signals. As emphasized, the two tasks are interrelated, that is, the user intent can increase the interpretability of bundles and help identify relevant products to form bundles, while the well-defined bundles can clearly reflect the user's intent. We thus create a prompt to exploit the inferred intent to refine the generated bundles - *Given the generated intents, adjust the detected bundles with the product descriptions. The output format is: {'bundle number':['product number']}.* If there is any adjustment, the adjusted bundles, in turn, are further employed to refine the intents - *Given the adjusted bundles, regenerate the intents behind each bundle, the output format is: {'bundle number':'intent'}* . We repeat the above process T_s times or until there is no further adjustment.

3.2.2 Adaptive Auto-Feedback. Despite the effectiveness of the self-correction strategy, explicit supervision is more helpful to better guide LLMs. Hence, we proceed to design an auto-feedback mechanism to recurrently provide adaptive instructions for LLMs regarding the generated bundles and intents, thereby chasing further performance enhancement. We start from the generated bundles. Based on the ground truth bundles and potential mistakes made by LLMs, we define five types of supervision signals:

- Type 1: correct and should be kept;
- Type 2: invalid and should be removed (not containing any products in the ground truth bundles);
- Type 3: containing unrelated products to be removed;
- Type 4: missing some products and should append other related products (bundle size>1);
- Type 5: missing some products and should contain at least two related products (bundle size=1).

Since multiple bundles may be generated by LLMs given a session, we calculate the Jaccard similarity between the generated bundles and ground truth bundles. Thus, we can match and compare them, and then automatically provide the corresponding supervision signals. Accordingly, we pass the prompt to LLMs to refine its generated bundles - *Here are some tips for the detected bundles in your answer: {[bundle X is Type X, . . .]}; Adjust the bundles based on the tips in your answer. Please output the adjusted bundles with the format: {'bundle number':['product number']}.* We repeat such a process T_b times or until only the Type 0 signal is returned. Since LLMs could make different mistakes for various generated bundles in different sessions, the auto-feedback mechanism can recurrently offer adaptive supervision based on the ground truth bundles.

We now proceed with the inferred intents. We first ask LLMs to re-infer intents for the above updated bundles - *Please use 3 to 5 words to generate intents behind the detected bundles, the output format is: {'bundle number':'intent'}.* Afterwards, similar to the bundle auto-feedback generation, we define 3 types of supervision signals for the inferred intents as below:

- Type 1: (Naturalness) be more natural;
- Type 2: (Coverage) cover more products within the bundle;
- Type 3: (Motivation) have a more motivational description.

We seek to compare the intents inferred by LLMs and the ground truth in three aspects. In particular, *Naturalness* indicates whether the intent is easy to read and understand; *Coverage* implies to what extent the items in the bundle can be covered by the intent; and *Motivation* suggests whether the intent contains motivational description, i.e., describing the purpose of the bundle by activities, events, or actions. For example, the intent 'assembling computer' is motivational, whilst 'different computer accessories' is not. Based on this, we can provide the corresponding supervision signal defined above. However, it may necessitate human evaluation, potentially leading to labor-intensive tasks. To remedy this, we adopt two LLMs (another ChatGPT and Claude-2) as raters to automatically conduct the intent assessment task via the Intent Assessment Prompt demonstrated on the right side.

For the sake of robustness, we instruct each rater to repeat the evaluation process three times and calculate the average as the final rating. For each metric, we compare the rating between the generated intent and ground truth. If any rater provides a lower rating to the generated intents on any metric, we then provide the corresponding supervision signals to LLMs for further refinement via the prompt - *Here are some tips for the generated intents in your answer: regenerate intent X to {[Type X, · · ·]}. Please output the regenerated intents with the format: {'bundle number':'intent'}.* We repeat the above process either T_i times or until the ratings of generated intents are no lower than those of ground truth across the three metrics for both raters.

Prompt: *Intent Assessment*

The intent should describe the customer's motivation well in the purchase of the product bundles. You are asked to evaluate two intents for a bundle, using three metrics: Naturalness, Coverage, and Motivation. The details and scales of each metric are listed below.
Naturalness:
1 - the intent is difficult to read and understand
2 - the intent is fair to read and understand
3 - the intent is easy to read and understand
Coverage:
1 - only a few items in the bundle are covered by the intent
2 - around half items in the bundle are covered by the intent
3 - most items in the bundle are covered by the intent
Motivation:
1 - the intent contains no motivational description
2 - the intent contains motivational description
Following are the bundles that we ask you to evaluate:
{[product X: title, . . .]}, {intent X, intent GT}
Please answer in the following format: {'intent number': ['Naturalness':score, 'Coverage':score, 'Motivation':score]}.

3.2.3 Rules Summarization. Beyond the conversation above, we further instruct LLMs to derive useful rules from the entire task execution process to prevent recurring errors in the future, with the prompt - *Based on the conversations above, which rules do you find when detecting bundles?*. Here are examples of some generated rules: (1) products with similar intents are grouped together in a bundle; (2) missing products can be appended to the bundles if they are related to the intent; (3) the adjusted bundles should reflect the intent and include relevant products from the sequence; (4) the

intent behind a bundle can be inferred from the combination of products and their intended use; and (5) the regenerated intents should be descriptive and motivational.

3.3 Demonstration Guided Inference (DGI)

Given the constructed demonstration, we ask LLMs to perform the two tasks on the corresponding test session via the prompt - *Based on the rules above, detect bundles for the below product sequence: {[product X: title, . . .]}. The answer format is: {'bundle number':['product number']}. No explanation for the results* and - *Please use 3 to 5 words to generate intents behind the detected bundles, the output format is {'bundle number':'intent'}.* By observing demonstrations of neighbor sessions, LLMs can seek tailored and adaptive lessons for improved performance on the test session.

3.4 Complexity Discussion

The time complexity of our method mainly comes from three modules: (1) Neighbor Session Retrieval, (2) Adaptive Demonstration Generation, and (3) Demonstration Guided Inference. For (1), using SentenceTransformer to obtain the session embedding is quite fast. The main complexity lies in the pairwise similarity calculation, i.e., $O(|\mathcal{S}_r| \times |\mathcal{S}_t|)$, where $|\mathcal{S}_r|$ and $|\mathcal{S}_t|$ are the total number of training and test sessions, respectively. For (2), it involves calling ChatGPT API for iterative adjustment via self-correction and adaptive auto-feedback, which constitutes the bulk of the complexity. This process mainly depends on the network latency and server load. For (3), it involves doing inference with the demonstration as context using ChatGPT API, which depends on the token size of the context and factors mentioned in (2).

In real-world applications, the computation within each of the three modules can be done in parallel to speed up the whole process. Besides, during inference, for each test session, step (1) can be sped up by employing Product Quantization (PQ) [25] to compress text embedding as quantization-based representation [67], thereby reducing the cost of similarity computation. Step (2) involves multiple iterations and refinements using LLMs for generating demonstrations, which constitute the bulk of the complexity and can be done offline and stored in the database in advance. This is because the demonstrations are generated using training sessions only. We can offline generate them on all training sessions or a reasonable number of representative training sessions. In summary, our method is reasonably feasible for practical application.

4 EXPERIMENTS AND RESULTS

We conduct extensive experiments on three public datasets to demonstrate the effectiveness of our proposed AICL paradigm. For reproducibility [47], our code is available at https://github.com/BundleRec/bundle_generation.

4.1 Experimental Setup

4.1.1 Datasets. We adopt three public bundle datasets created by a resource paper in SIGIR 2022 [48, 49]. In particular, they design a crowdsourcing task to annotate high-quality bundles and the corresponding intents from user sessions in Amazon datasets [19] with three domains, i.e., Electronic, Clothing, and Food. The statistics are summarized in Table 3. For each dataset, we chronologically split the session data into training, validation, and test sets with a ratio of 7:1:2. To the best of our knowledge, they are the ONLY bundle datasets with user sessions and well-labeled intents. Other widely-used bundle datasets, such as Steam, Netease, Youshu [2], Goodreads [21], and iFashion [42] cannot be utilized in our study due to the unavailability of bundle intents.

Table 3: The statistics of the three bundle datasets.

	Electronic	Clothing	Food
#Users	888	965	879
#Items	3499	4487	3767
#Sessions	1145	1181	1161
#Bundles	1750	1910	1784
#Intents	1422	1466	1156
#User-Item Interactions	6165	6326	6395
#User-Bundle Interactions	1753	1912	1785
Average Bundle Size	3.52	3.31	3.58

4.1.2 Baselines. We compare our proposed AICL with seven baselines. **Freq** [49] is the frequent itemsets mining method. **BBPR** [39] is the greedy method with the predictions of BPRMF [43]. **POG** [10] is the Transformer-based encoder-decoder model to generate personalized outfits. **BYOB** [12] treats bundle generation as a combinatorial optimization problem with reinforcement learning. **T5** [40] is a Transformer-based seq2seq model. We use the version with 220M parameters and fine-tune it with our training data. **Zero-shot** directly adopts LLMs to generate bundles and infer intents from user sessions. **Few-shot** exploits LLMs to perform the two tasks with few-shot examples. Furthermore, for a comprehensive comparison, we consider different variants for Few-shot by changing the way of selecting few-shot examples, including (1) *Few-shot-random* randomly selecting different examples for each test session as the demonstration; (2) *Few-shot-fix* randomly selecting the same examples, and use them as demonstrations for all test sessions; and (3) *Few-shot-top* using the nearest neighbor sessions (same as in AICL) for each test session as the demonstration. We empirically find that Few-shot-fix generally achieves the best performance among all variants. Thus, we report the results produced by Few-shot-fix in our study. *It is noteworthy that we do not compare with BUNT [23], Conna [54] and BGGN [8]. This is because BUNT requires explicit user queries, and the source codes of Conna and BGGN are not available. We failed to reproduce them without the model details.*

4.1.3 Evaluation Metrics. Following [48, 49], we adopt three metrics to evaluate the quality of generated bundles: *Precision*, *Recall*, and *Coverage*. At the session level, *Precision* and *Recall* measure how many bundles (subsets included) have been correctly predicted for each session. Meanwhile, *Coverage*, at the bundle level, measures how many items are correctly covered by each hit bundle compared to the ground truth bundle. Due to space limitations, the detailed explanation of these metrics can be found in [48, 49].

Concerning the inferred intents, our initial plan was to perform an automatic evaluation using the widely-used ROUGE [44] to evaluate *n*-grams of the generated intents with ground truth. However, our empirical observations reveal that the generated intents, while semantically aligned with the ground truth, exhibit distinct expressions. Hence, using ROUGE may not accurately gauge and reflect the true quality of these intents. Thus, we carefully design human evaluation to examine the quality of intents with the three metrics

Table 4: The performance on bundle generation. For the sake of robustness, we run each method five times to report the average results; the best results are highlighted in bold; the runner-up is underlined; and '†' refers to our method significantly outperforms the best-performed baselines with a paired t-test (p-value < 0.05).

		Electronic			Clothing			Food		
		Precision	Recall	Coverage	Precision	Recall	Coverage	Precision	Recall	Coverage
Non-LLMs	Freq	0.423	0.597	0.701	0.532	0.566	0.698	0.491	0.525	0.684
	BBPR	0.260	0.122	0.433	0.239	0.211	0.449	0.210	0.183	0.416
	POG	0.339	0.250	0.412	0.312	0.221	0.399	0.365	0.266	0.393
	BYOB	0.340	0.294	0.361	0.311	0.273	0.457	0.304	0.253	0.427
LLMs	T5	0.553	0.553	0.502	0.572	0.581	0.507	0.575	0.574	0.451
	Zero-shot	0.580	0.820	0.720	0.603	0.752	0.788	0.604	0.815	0.748
	Few-shot	0.587	0.825	0.724	0.595	**0.836**	0.781	0.647	0.833	0.749
	AICL	**0.679**†	**0.859**†	**0.741**†	**0.677**†	0.788	**0.839**†	**0.698**†	**0.851**†	**0.755**†

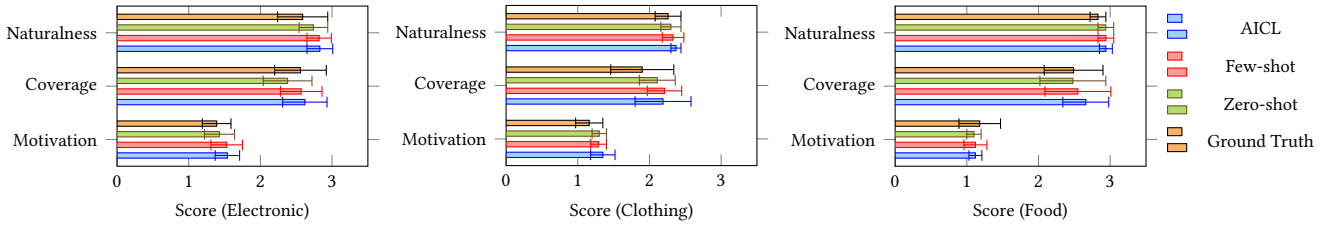

Figure 3: Human evaluation on inferred intents. The bar and horizontal line are mean and standard deviation values, respectively.

Naturalness, Coverage, and Motivation defined in Section 3.2.2. In our study, we ask 15 participants to rate the intents generated by the workers (ground truth) and three LLMs (Zero-shot, Few-shot, and AICL) for 60 bundles (20 per domain).

4.1.4 Hyper-parameter Settings. The best parameter settings for all methods are found based on the performance of the validation set or the suggested values in the original papers. For Freq, we apply a grid search in $\{0.0001, 0.001, 0.01\}$ for the *support* and *confidence* values. The best settings are 0.001 on all datasets. The embedding size is set to 20 for BBPR, POG, and BYOB for a fair comparison. The negative samples for BBPR and BYOB are set to 2. For BBPR, the initial bundle size is 3, and the number of neighbors is 10. For BYOB, the bundle size is set as 3. For POG and BYOB, the size of candidate item set is 20. The batch size for POG, BYOB, and T5 are set as 64, 64, and 4 respectively. The learning rate is searched in $\{0.0001, 0.001, 0.01\}$ for BBPR, POG, and BYOB, and in $\{0.00002, 0.00005, 0.00007, 0.0001\}$ to fine-tune T5. The best settings are 0.01, 0.001, 0.001, and 0.00005 for the four methods, respectively. For Zero- and Few-shot, we use the same prompts as AICL (bundle generation and intent reasoning parts only). For Few-shot, we use one example to construct the demonstration. For a fair comparison, we set $k = 1$ for our AICL. Besides, we apply a grid search in $\{1, 2, 3, 4, 5\}$ for T_s, T_b, and T_i. The optimal settings are $T_s = T_i = 1$ and $T_b = 4$.

4.2 Results and Analysis

4.2.1 Performance of Bundle Generation. Table 4 shows the performance of all methods on the bundle generation task. Several interesting observations can be noted. **(1)** LLM-based methods generally surpass Non-LLM ones, exhibiting the superiority of LLMs on our defined tasks. *Regarding the Non-LLM methods*, **(2)** the straightforward Freq outperforms all model-based methods (BBPR, POG, and BYOB), possibly because the data sparsity issue causes the model-based methods to be under-trained. In contrast, Freq initially identifies frequent patterns at the category level, effectively mitigating such an issue. **(3)** Among the three model-based methods, the DL-based ones (POG and BYOB) exhibit better performance, underscoring the effectiveness of DL techniques. *In terms of LLM-based methods*, **(4)** T5 (220M) performs the least effectively, primarily due to its relatively small model size in comparison to GPT-3.5 with 154 billion parameters. **(5)** Few-shot exceeds Zero-shot, showcasing the usefulness of demonstrations in ICL. **(6)** AICL generally delivers the top performance across all datasets, providing solid evidence of the effectiveness and efficiency of its distinctive design.

4.2.2 Performance of Intent Reasoning. Figure 3 displays the rating scores of intents generated by Zero-shot, Few-shot, AICL, and the workers (ground truth). Other baselines are not compared as they cannot generate intents. We randomly select 20 correctly generated bundles and their intents in each domain. In total, 60 bundles are assessed on three metrics, i.e., Naturalness, Coverage, and Motivation as defined in Section 3.2.2. The overall trends on all datasets are similar. First, AICL achieves the best performance in most cases, either with higher mean values or lower standard deviation values. This helps confirm the superiority of AICL on effective intent reasoning. Second, Few-shot generally exceeds Zero-shot, validating the usefulness of demonstrations on guiding LLMs for improved performance. Third, the ground truth intents annotated by workers are defeated by at least one of the three LLM-based methods (except 'Motivation' on Food). This might be attributed to workers often prioritizing the speed of task completion to maximize their earnings, potentially at the expense of work quality [15]. Furthermore, it underscores the advanced capabilities of reasoning and natural language generation in LLMs, emphasizing their substantial potential in the context of crowdsourcing tasks.

4.2.3 Ablation Study. We compare AICL with its six variants to examine the efficacy of each component. In particular, AICL_w/o_top randomly samples one session in the training set to replace the

Table 5: The results of ablation study on the bundle generation task. We run each variant five times to report the average results.

	Electronic			Clothing			Food		
	Precision	Recall	Coverage	Precision	Recall	Coverage	Precision	Recall	Coverage
AICL_w/o_top	0.667	0.823	0.734	0.667	0.761	0.814	0.689	0.837	0.743
AICL_w/o_self	0.651	0.839	0.735	0.649	0.768	0.835	0.651	0.820	0.747
AICL_w/o_auto	0.635	0.811	0.719	0.652	0.768	0.816	0.663	0.825	0.729
AICL_w/o_context	0.556	0.786	0.721	0.577	0.723	0.826	0.583	0.814	0.723
AICL_w/o_rules	0.665	0.822	0.729	0.661	0.772	0.826	0.679	0.834	0.742
AICL_w/o_intent	0.636	0.825	0.731	0.655	0.769	0.821	0.671	0.841	0.741
AICL	**0.679**	**0.859**	**0.741**	**0.677**	**0.788**	**0.839**	**0.698**	**0.851**	**0.755**

Table 6: The performance comparison between Sentence-Transformer and LLMs on Electronic.

	Precision	Recall	Coverage
SentenceTransformer (all-MiniLM-L6-v2)	0.623	0.745	0.674
OpenAI (text-embedding-ada-002)	0.616	0.860	0.683

top neighbor. AICL_w/o_self removes the self-correction strategy from the demonstration. AICL_w/o_auto omits the auto-feedback mechanism from the demonstration. AICL_w/o_context removes both self-correction and auto-feedback modules from the demonstration. AICL_w/o_rules abandons the rules summarization from the demonstration. AICL_w/o_intent deletes intent reasoning from the demonstration. Due to space limitations, we only show the results of the bundle generation task, as presented in Table 5.

Overall, all the variants demonstrate lower performance compared to AICL, showcasing the contribution of each component to the improved performance. To be specific, AICL_w/o_top underperforms AICL, which indicates the importance of identifying highly correlated examples to generate demonstrations. Both AICL_w/o_self and AICL_w/o_auto perform worse than AICL, while gaining better performance compared with AICL_w/o_context, implying the significance of both self-correction and auto-feedback strategies. The fact that AICL defeats AICL_w/o_rules exhibits the usefulness of rules summarization in instructing LLMs. Besides, an obvious performance drop is observed on AICL_w/o_intent when compared with AICL. This helps reinforce our claim that effective user intents play a crucial role in identifying relevant products to form improved bundles.

Furthermore, our Neighbor Session Retrieval module exploits the open-source Python framework SentenceTransformer (all-MiniLM-L6-v2) to get session representations for similarity calculation instead of using LLMs (e.g., OpenAI text-embedding-ada-002) due to its comparable performance with less cost (time and money) as explained in Section 3.1. To verify our claim, we randomly sample 50 sessions from Electronic and use the two methods to help get the nearest neighbors. The results are presented in Table 6. Accordingly, their precision and coverage are comparable, while AICL with OpenAI embeddings possess higher recall. This also indicates the results of AICL reported in our paper may not be its upper bound, and there is still space for further improvements by using better sentence encoding models.

4.2.4 Hyper-Parameter Analysis. We further study the impact of essential hyper-parameters on AICL, including the rounds of self-correction (T_s) and auto-feedback for both tasks (T_b and T_i), as well as the number of neighbor sessions (k). First, we observe that around 29% of neighbor sessions adjust bundles and intents with $T_s = 1$,

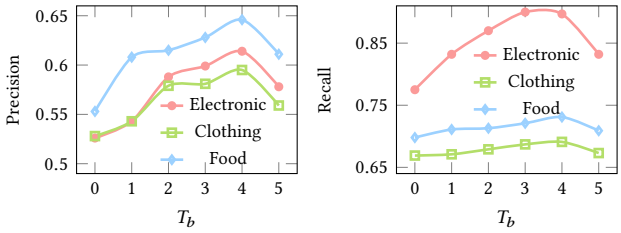

Figure 4: The impact of T_b on all neighbor sessions.

and the accuracy is improved by 5.9% w.r.t. Precision on average. In summary, the self-correction allows LLMs to reassess the response, leading to more self-consistent and effective results. Second, we apply a grid search in $\{1, 2, 3, 4, 5\}$ to check the impact of T_b depicted in Figure 4. As T_b increases, the performance initially rises, reaching its peak with $T_b = 4$, and subsequently declines as T_b continues to increase. Our empirical findings indicate an average improvement of 16.7% in Precision with the auto-feedback. Third, we find that with $T_i = 1$, most intents (65.7%) generated by LLMs are quite close to the ground truth intents annotated by workers. It reveals the great potential of LLMs in crowdsourcing tasks. Lastly, we observe that the best performance is attained with $k = 1$. Increasing the value of k does not consistently yield noticeable improvements and can, on occasion, even lead to marginal performance declines. This is intuitive as a large k substantially lengthens the context, which may confuse LLMs and result in decreased performance.

4.2.5 Case Study. A case study is performed to check the generated bundles and intents by Zero-shot, Few-shot, and AICL. Due to space limitations, we only show one sampled test session on Electronic in Figure 5. For *bundle generation*, it's evident that Zero-shot only manages to generate a portion of the Galaxy Tab and Protection bundle, overlooking the TV Box bundle. Few-shot, on the other hand, identifies three bundles, but the first is a subset of the ground truth, and the last, combining the iPad case and Galaxy Tab, is less coherent. In contrast, AICL consistently and effectively generates all bundles that perfectly align with the ground truth. For *intent reasoning*, all methods perform similarly in terms of Naturalness and Coverage. However, AICL demonstrates a superior performance in terms of Motivation. For example, the intent associated with the TV Box bundle 'Upgrade Your Streaming Experience' is more motivational than the intents 'Streaming Box' (ground truth) and 'Streaming Player' (Few-shot).

4.2.6 Limitations of LLMs for Bundle Generation. Despite the effectiveness of LLMs, we now discuss their limitations on our defined tasks. First, through empirical observations, we notice that LLM-based methods tend to produce smaller bundles in comparison to

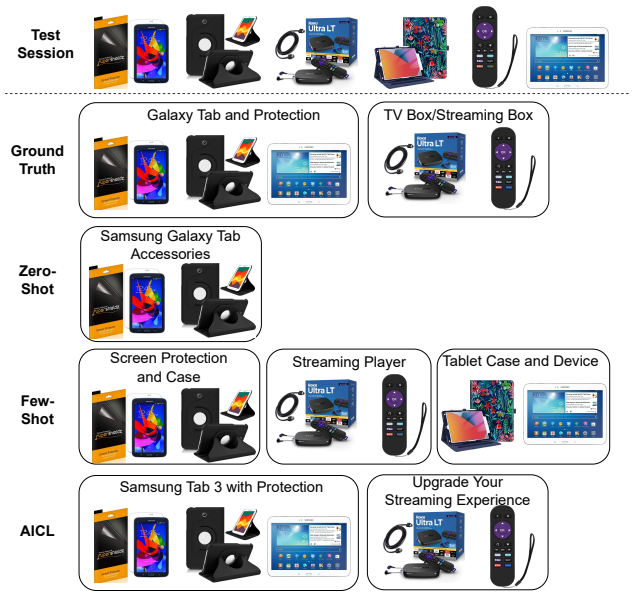

Figure 5: The generated bundles and intents on Electronic.

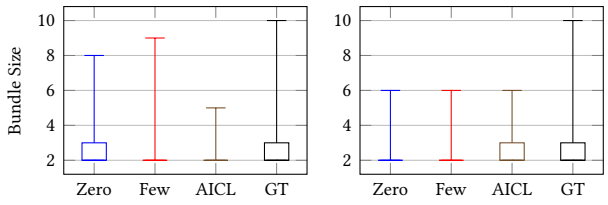

Figure 6: Bundle size distribution on Electronic and Clothing.

the ground truth (GT) bundles annotated by workers, as depicted in Figure 6. The smaller bundle size could limit the diversity of recommendations. However, it may align with real user behavior, as most online customers often purchase a small number, typically two, of items in one shopping session [74]. In this context, an overlarge bundle size may not provide significant benefits and could potentially divert users' attention. Second, despite our various attempts to emphasize the prompt constraint, namely, 'Each bundle must contain multiple products', GPT-3.5 sometimes generates bundles with only a single product. It could be attributable to its inherent hallucination issue, and such an issue can be partially resolved by more powerful LLMs, e.g., GPT-4. Specifically, we do a preliminary exploration to examine the performance of GPT-4 on sessions (5 per domain) with such an issue using GPT-3.5. The results are depicted in Table 7. Overall, improvements are observed in two key areas. Firstly, all compared methods exhibited fewer instances of 'bad cases' (i.e., bundles containing a single product) with GPT-4 compared to GPT-3.5. Secondly, the accuracy (precision, recall, and coverage) of most methods show enhancements with GPT-4 compared to GPT-3.5, although there are some exceptions, such as decreases in certain metrics. Moreover, Figure 7 illustrates the performance comparison on a real test session on Electronic, which has such a hallucination issue with GPT-3.5 but can be completely resolved with GPT-4.

Table 7: The performance comparison between GPT-3.5 and GPT-4, where '#Bad Case' refers to the number of sessions with the hallucination issue. Due to space limitation, we only present the results on Precision and Recall.

		GPT-3.5			GPT-4		
		Recall	Precision	#Bad Case	Recall	Precision	#Bad Case
Elect.	Zero-shot	0.733	0.667	5	0.733	0.633	2
	Few-shot	0.533	0.600	5	0.800	0.600	1
	AICL	0.800	0.700	5	0.833	0.700	1
Clothing	Zero-shot	0.533	0.517	5	0.567	0.450	2
	Few-shot	0.433	0.400	5	0.667	0.483	1
	AICL	0.611	0.556	5	0.667	0.583	1
Food	Zero-shot	0.800	0.733	5	0.800	0.667	2
	Few-shot	0.646	0.533	5	0.800	0.667	2
	AICL	0.800	0.733	5	0.800	0.700	1

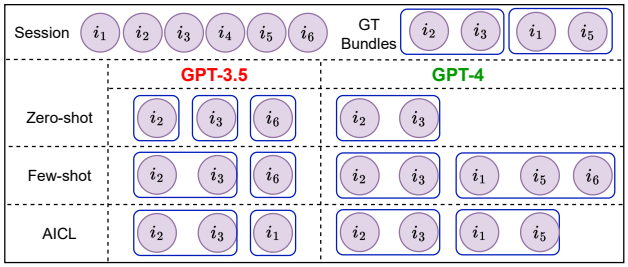

Figure 7: The comparison between GPT-3.5 and GPT-4.

5 CONCLUSION AND FUTURE WORK

Motivated by the advanced reasoning capability exhibited in LLMs, this paper initiates a pioneering exploration into two interrelated tasks, i.e., personalized bundle generation and the underlying intent inference, both rooted in users' behaviors within a session. To this end, we propose an adaptive in-context learning (AICL) paradigm equipped with three modules, i.e., neighbor session retrieval, adaptive demonstration generation, and demonstration guided inference. This empowers LLMs to seek tailored and adaptive lessons from neighbor sessions as demonstrations for performance improvements on the test session. As a result, it ultimately delivers an effective approach that is capable of generating multiple, personalized, and intelligible bundles with adaptive sizes given any user session. Experimental results on three public datasets verify the effectiveness of our AICL on both tasks.

For future endeavors, there are several potential directions, including (1) designing strategies to create larger bundles and better control the output format; (2) introducing self-correction and auto-feedback mechanisms in the inference stage, and (3) exploring the utilization of multi-modal data for further enhancement.

ACKNOWLEDGMENTS

We greatly acknowledge the support of National Natural Science Foundation of China (Grant No. 72371148 and 72192832), the Shanghai Rising-Star Program (Grant No. 23QA1403100), and the Natural Science Foundation of Shanghai (Grant No. 21ZR1421900). It was also supported by A*Star Center for Frontier Artificial Intelligence Research and in part by the Data Science and Artificial Intelligence Research Centre, School of Computer Science and Engineering at the Nanyang Technological University (NTU), Singapore.

REFERENCES

[1] Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proceedings of 20th International Conference on Very Large Scale Data Bases (VLDB)*, Vol. 1215. 487–499.

[2] Tzoof Avny Brosh, Amit Livne, Oren Sar Shalom, Bracha Shapira, and Mark Last. 2022. BRUCE: bundle recommendation using contextualized item embeddings. In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys)*. 237–245.

[3] Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, and Jun Gao. 2019. Personalized bundle list recommendation. In *The Web Conference (TheWebConf)*. 60–71.

[4] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: an effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 1007–1014.

[5] Moran Beladev, Lior Rokach, and Bracha Shapira. 2016. Recommender systems for product bundling. *Knowledge-Based Systems (KBS)* 111 (2016), 193–206.

[6] Da Cao, Liqiang Nie, Xiangnan He, Xiaochi Wei, Shunzhi Zhu, and Tat-Seng Chua. 2017. Embedding factorization models for jointly recommending items and user generated lists. In *Proceedings of the 40th international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 585–594.

[7] Jianxin Chang, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2020. Bundle recommendation with graph convolutional networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1673–1676.

[8] Jianxin Chang, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. 2021. Bundle recommendation and generation with graph neural networks. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 35, 3 (2021), 2326–2340.

[9] Liang Chen, Yang Liu, Xiangnan He, Lianli Gao, and Zibin Zheng. 2019. Matching user with item set: collaborative bundle recommendation with deep attention network.. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 2095–2101.

[10] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. 2019. POG: personalized outfit generation for fashion recommendation at Alibaba iFashion. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 2662–2670.

[11] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 1126–1132.

[12] Qilin Deng, Kai Wang, Minghao Zhao, Runze Wu, Yu Ding, Zhene Zou, Yue Shang, Jianrong Tao, and Changjie Fan. 2021. Build your own bundle - a neural combinatorial optimization method. In *Proceedings of the 29th ACM International Conference on Multimedia (MM)*. 2625–2633.

[13] Qilin Deng, Kai Wang, Minghao Zhao, Zhene Zou, Runze Wu, Jianrong Tao, Changjie Fan, and Liang Chen. 2020. Personalized bundle recommendation in online games. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)*. 2381–2388.

[14] Paolo Dragone, Giovanni Pellegrini, Michele Vescovi, Katya Tentori, and Andrea Passerini. 2018. No more ready-made deals: constructive recommendation for telco service bundling. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys)*. 163–171.

[15] Ujwal Gadiraju, Ricardo Kawase, Stefan Dietze, and Gianluca Demartini. 2015. Understanding malicious behavior in crowdsourcing platforms: the case of online surveys. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI)*. 1631–1640.

[16] Robert Garfinkel, Ram Gopal, Arvind Tripathi, and Fang Yin. 2006. Design of a shopbot and recommender system for bundle purchases. *Decision Support Systems (DSS)* 42, 3 (2006), 1974–1986.

[17] Yong Ge, Hui Xiong, Alexander Tuzhilin, and Qi Liu. 2014. Cost-aware collaborative filtering for travel tour recommendations. *ACM Transactions on Information Systems (TOIS)* 32, 1 (2014), 1–31.

[18] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 1096–1102.

[19] Ruining He and Julian McAuley. 2016. Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. 507–517.

[20] Yun He, Jianling Wang, Wei Niu, and James Caverlee. 2019. A hierarchical self-attentive model for recommending user-generated item lists. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. 1481–1490.

[21] Yun He, Yin Zhang, Weiwen Liu, and James Caverlee. 2020. Consistency-aware recommendation for user-generated item list continuation. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*. 250–258.

[22] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Majumder, Nathan Kallus, and Julian Mcauley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. 720–730.

[23] Zhankui He, Handong Zhao, Tong Yu, Sungchul Kim, Fan Du, and Julian McAuley. 2022. Bundle mcr: towards conversational bundle recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems (RecSys)*. 288–298.

[24] Zixian Huang, Jiaying Zhou, Gengyang Xiao, and Gong Cheng. 2023. Enhancing in-context learning with answer feedback for multi-span question answering. In *Natural Language Processing and Chinese Computing (NLPCC)*. 744–756.

[25] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 33, 1 (2010), 117–128.

[26] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 7969–7992.

[27] Pigi Kouki, Ilias Fountalis, Nikolaos Vasiloglou, Nian Yan, Unaiza Ahsan, Khalifeh Al Jadda, and Huiming Qu. 2019. Product collection recommendation in online retail. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)*. 486–490.

[28] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*. 1419–1428.

[29] Yinfeng Li, Chen Gao, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1997–2002.

[30] Guannan Liu, Yanjie Fu, Guoqing Chen, Hui Xiong, and Can Chen. 2017. Modeling buying motives for personalized product bundle recommendation. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 3 (2017), 1–26.

[31] Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2022. Generated knowledge prompting for commonsense reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

[32] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149* (2023).

[33] Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. 2011. Personalized travel package recommendation. In *IEEE 11th International Conference on Data Mining (ICDM)*. 407–416.

[34] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1831–1839.

[35] Yidan Liu, Min Xie, and Laks VS Lakshmanan. 2014. Recommending user generated item lists. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys)*. 185–192.

[36] Yunshan Ma, Yingzhi He, An Zhang, Xiang Wang, and Tat-Seng Chua. 2022. CrossCBR: cross-view contrastive learning for bundle recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 1233–1241.

[37] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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

[39] Apurva Pathak, Kshitiz Gupta, and Julian McAuley. 2017. Generating and personalizing bundle recommendations on steam. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1073–1076.

[40] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research (JMLR)* 21, 1 (2020), 5485–5551.

[41] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[42] Yuyang Ren, Zhang Haonan, Luoyi Fu, Xinbing Wang, and Chenghu Zhou. 2023. Distillation-enhanced graph masked autoencoders for bundle recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1660–1669.

[43] 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 (UAI)*. 452–461.

[44] Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for*

Computational Linguistics (TACL) 8 (2020), 264–280.

[45] Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. 2023. Large language models are competitive near cold-start recommenders for language-and item-based preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 890–896.

[46] Oren Sar Shalom, Noam Koenigstein, Ulrich Paquet, and Hastagiri P Vanchinathan. 2016. Beyond collaborative filtering: the list recommendation problem. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. 63–72.

[47] Zhu Sun, Hui Fang, Jie Yang, Xinghua Qu, Hongyang Liu, Di Yu, Yew-Soon Ong, and Jie Zhang. 2022. Daisyrec 2.0: Benchmarking recommendation for rigorous evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2022).

[48] Zhu Sun, Kaidong Feng, Jie Yang, Hui Fang, Xinghua Qu, Yew-Soon Ong, and Wenyuan Liu. 2024. Revisiting bundle recommendation for intent-aware product bundling. *ACM Transactions on Recommender Systems (TORS)* (2024).

[49] Zhu Sun, Jie Yang, Kaidong Feng, Hui Fang, Xinghua Qu, and Yew Soon Ong. 2022. Revisiting bundle recommendation: datasets, tasks, challenges and opportunities for intent-aware product bundling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2900–2911.

[50] Md Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian McAuley. 2020. Attentive sequential models of latent intent for next item recommendation. In *Proceedings of The Web Conference (TheWebConf)*. 2528–2534.

[51] Shoujin Wang, Liang Hu, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Longbing Cao. 2019. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 3771–3777.

[52] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

[53] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*. 24824–24837.

[54] Penghui Wei, Shaoguo Liu, Xuanhua Yang, Liang Wang, and Bo Zheng. 2022. Towards personalized bundle creative generation with contrastive non-autoregressive decoding. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 2634–2638.

[55] Yinwei Wei, Xiaohao Liu, Yunshan Ma, Xiang Wang, Liqiang Nie, and Tat-Seng Chua. 2023. Strategy-aware bundle recommender system. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1198–1207.

[56] Huizi Wu, Hui Fang, Zhu Sun, Cong Geng, Xinyu Kong, and Yew-Soon Ong. 2023. A generic reinforced explainable framework with knowledge graph for session-based recommendation. In *IEEE 39th International Conference on Data Engineering (ICDE)*. 1260–1272.

[57] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2023. A survey on large language models for recommendation. *arXiv preprint arXiv:2305.19860* (2023).

[58] Min Xie, Laks VS Lakshmanan, and Peter T Wood. 2010. Breaking out of the box of recommendations: from items to packages. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys)*. 151–158.

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

[60] De-Nian Yang, Wang-Chien Lee, Nai-Hui Chia, Mao Ye, and Hui-Ju Hung. 2012. On bundle configuration for viral marketing in social networks. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*. 2234–2238.

[61] Fan Yang, Zheng Chen, Ziyan Jiang, Eunah Cho, Xiaojiang Huang, and Yanbin Lu. 2023. PALR: personalization aware llms for recommendation. *arXiv preprint arXiv:2305.07622* (2023).

[62] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[63] Bin Yin, Junjie Xie, Yu Qin, Zixiang Ding, Zhichao Feng, Xiang Li, and Wei Lin. 2023. Heterogeneous knowledge fusion: a novel approach for personalized recommendation via llm. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 599–601.

[64] Feng Yu, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: target attentive graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1921–1924.

[65] Zhouxin Yu, Jintang Li, Liang Chen, and Zibin Zheng. 2022. Unifying multi-associations through hypergraph for bundle recommendation. *Knowledge-Based Systems (KBS)* 255 (2022), 109755.

[66] Jianyang Zhai, Xiawu Zheng, Chang-Dong Wang, Hui Li, and Yonghong Tian. 2023. Knowledge prompt-tuning for sequential recommendation. In *Proceedings of the 29th ACM International Conference on Multimedia (MM)*. 6451–6461.

[67] Han Zhang, Hongwei Shen, Yiming Qiu, Yunjiang Jiang, Songlin Wang, Sulong Xu, Yun Xiao, Bo Long, and Wen-Yun Yang. 2021. Joint learning of deep retrieval model and product quantization based embedding index. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1718–1722.

[68] Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys)*. 993–999.

[69] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: a large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001* (2023).

[70] Zhenning Zhang, Boxin Du, and Hanghang Tong. 2022. Suger: a subgraph-based graph convolutional network method for bundle recommendation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*. 4712–4716.

[71] Sen Zhao, Wei Wei, Ding Zou, and Xianling Mao. 2022. Multi-view intent disentangle graph networks for bundle recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 4379–4387.

[72] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2023. Least-to-most prompting enables complex reasoning in large language models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

[73] Nengjun Zhu, Jian Cao, Yanchi Liu, Yang Yang, Haochao Ying, and Hui Xiong. 2020. Sequential modeling of hierarchical user intention and preference for next-item recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*. 807–815.

[74] 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 and Development in Information Retrieval (SIGIR)*. 657–666.