ECCR: Explainable and Coherent Complement Recommendation based on Large Language Models

Zelong Li[1], Yan Liang[2], Ming Wang[2], Sungro Yoon[2], Jiaying Shi[2], Xin Shen[2], Xiang He[2], Chenwei Zhang[2], Wenyi Wu[2], Hanbo Wang[2], Jin Li[2], Jim Chan[2], Yongfeng Zhang[1]
Rutgers University[1], Amazon.com[2]
{zelong.li,yongfeng.zhang}@rutgers.edu,
{ynliang,mingww,sungroy,sjiaying,xinshen,xianghe,cwzhang,wenyiwu,hanbwang,jincli,jamchan}@amazon.com

ABSTRACT

A complementary item is an item that pairs well with another item when consumed together. In the context of e-commerce, providing recommendations for complementary items is essential for both customers and stores. Current models for suggesting complementary items often rely heavily on user behavior data, such as co-purchase relationships. However, just because two items are frequently bought together does not necessarily mean they are truly complementary. Relying solely on co-purchase data may not align perfectly with the goal of making meaningful complementary recommendations. In this paper, we introduce the concept of "coherent complement recommendation", where "coherent" implies that recommended item pairs are compatible and relevant. Our approach builds upon complementary item pairs, with a focus on ensuring that recommended items are well used together and contextually relevant. To enhance the explainability and coherence of our complement recommendations, we fine-tune the Large Language Model (LLM) with coherent complement recommendation and explanation generation tasks since LLM has strong natural language explanation generation ability and multi-task fine-tuning enhances task understanding. We have also devised an LLM-compatible method for compressing and quantizing user behavior information into language model tokens. Experimental results indicate that our model can provide more coherent complementary recommendations than existing state-of-the-art methods, and human evaluation validates that our approach achieves up to a 48% increase in the coherent rate of complement recommendations.

KEYWORDS

Recommender Systems; Large Language Model; Explainable Machine Learning

ACM Reference Format:
Zelong Li, Yan Liang, Ming Wang, Sungro Yoon, Jiaying Shi, Xin Shen, Xiang He, Chenwei Zhang, Wenyi Wu, Hanbo Wang, Jin Li, Jim Chan, Yongfeng Zhang. 2024. ECCR: Explainable and Coherent Complement Recommendation based on Large Language Models. In *1st Workshop on Generative AI for Recommender Systems and Personalization at the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In today's era of information abundance, Recommendation Systems (RS) have become pivotal tools within the e-commerce landscape. They help the users discover the items that align with their interests and preferences. Researchers across various domains have dedicated their efforts to tackling diverse RS challenges, and among these, complementary RS stands out as an important and challenging area.

Complementary item pairs or sets often arise from user co-purchase behavior and are typically linked by shared purpose or intent [1]. Effective e-commerce complementary RS can significantly enhance the customer experience and yield positive business outcomes for the store. A store with valuable complementary RS can motivate users to purchase item pairs or sets, boosting overall sales. Simultaneously, customers benefit by saving time, as they can avoid the need for additional searches to find related essential items, resulting in an improved shopping experience.

To assess how well the complementary items match, we propose the concept "coherent", which includes two aspects, *Compatibility* and *Relevance. Compatibility* plays a crucial role in determining the suitability of item pairs, particularly in the case of electronic items. When an item pair lacks compatibility, it cannot be used together to achieve a common goal. For instance, in Figure 1, the cellphone and the screen protector are incompatible, rendering them unsuitable for shared purposes, such as device protection. We define *Relevance* items if they share the same value of at least one important attributes, like brand, color, style, and so on. In Figure 2, the umbrella has the same brand and design style as the handbag, leading to a better match than other candidate umbrella items.

Current complementary RS, such as P-Companion [1], often rely on category transition and co-purchase user behavior to build models. However, these methods encounter two primary challenges. First, not all co-purchased pairs qualify as coherent complements, particularly in domains like Fashion and Food. For example, while users may buy various dresses or chocolates, these items represent variations rather than true complements, as they serve the same purpose instead of a joint intent when used together. Second, the category-level coherence cannot replace the item-level one. Like Figure 1, while a screen protector may complement a cellphone at the category level, customers might not consider it a suitable recommendation if the items are incoherent.

To tackle the issues of irrelevant and incompatible complementary RS, we formalize the concept of *coherent complement recommendations* and utilize Large Language Models (LLMs) to generate

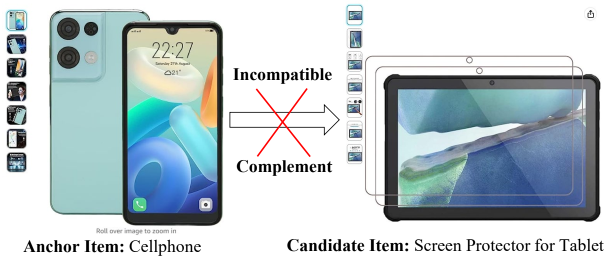

Figure 1: Incompatible complementary recommendation example.

Figure 2: Relevant complementary recommendation example.

explanations for coherent complement items as Figure 2. This aids users and stores in understanding why these item pairs are coherent. Recent advancements in LLMs [2–4] have highlighted their strong abilities in explanation generation on RS. To optimize our approach, we integrate the tasks of coherent complement recommendation and explanation generation into the LLM fine-tuning process. This integration is inspired by prior works [2, 3], suggesting that multi-task LLM fine-tuning enhances understanding when the tasks are interrelated. Besides, we develop an LLM-compatible item indexing algorithm to leverage user behavior information. This algorithm empowers the fine-tuned LLM to offer more effective complement recommendations and better generalization ability by incorporating text features and user behavior information.

To summarize, we make the following contributions to this work:

- We propose the "coherent complement recommendation" and demonstrate its effectiveness in enhancing the coherence of complementary items compared to existing RS. We achieve this by leveraging the LLM and providing high-quality explanations.
- We contend that using item titles directly may limit the capabilities of the LLM. As a solution, we develop an LLM-compatible item indexing algorithm that incorporates text features and user behavior information to enhance recommendation performance.
- We collaborate with domain experts to assess the quality improvement in coherent complement recommendations achieved by our fine-tuned LLM compared to existing models. We also evaluate the quality of the explanations provided.

2 RELATED WORK

2.1 Complementary Recommendation

Complementary recommendation, a critical scenario in recommendation systems, has predominantly relied on user purchase and browsing patterns for its construction. Previous research, exemplified by [1, 5–8], has concentrated on modeling complementarity by analyzing customers' historical purchase behaviors and the correlation among product attributes, including title semantics and image-based features. Recent investigations [9, 10] in this field have delved into advanced methodologies involving graph neural networks [11–13]. These methods aim to exploit prior knowledge pertaining to internal product-to-product graph structures to effectively constrain complementary recommendations within predefined boundaries. Another research avenue revolves around the development of sequential recommendations [14–16], wherein models learn complementarity implicitly from users' shopping sequences. However, despite the diversity in modeling architectures, these approaches overwhelmingly emphasize product-level correlations originating from users' purchase and browsing activities. This accentuates a notable concern: the recommended items may lack precise relevance and high compatibility with the anchor product. Consequently, it could lead to an unsatisfactory user experience where recommended products, although seemingly complementary at a high-level product category, ultimately lack relevance and compatibility at the individual product level. This situation poses a substantial risk to customer trust and overall satisfaction.

2.2 Large Language Model for Recommendation

Recent advancements in Large Language Models (LLMs) have garnered significant attention in the field of Natural Language Processing (NLP). Notably, models like OpenAI GPT [17] and Meta LLaMA [18], built on deep transformer architectures, have demonstrated the transformative potential of LLMs in various applications, including recommendation systems. Recent surveys [19, 20] highlight that LLMs are prominently employed in recommendation tasks in two primary ways: 1). *Universal Embeddings*: LLMs provide universal embeddings that can be used to initialize existing recommendation models [21–23], and 2). *Zero-Shot and Few-Shot Approaches*: LLMs produce recommendations by either leveraging their inherent knowledge as a zero-shot approach or by using a few examples within the prompt as a few-shot approach [24–26]. LLMs have also found applications in other recommendation scenarios, such as item generation [2, 3, 27] and re-ranking recommendation permutations [26]. However, none of these works have specifically focused on harnessing LLMs to achieve a deep understanding of item-to-item relationships. Deep product understanding is crucial to ensure that recommended items exhibit high compatibility and relevance. This understanding plays a pivotal role in enhancing the quality of item-pair or item-set recommendations, which benefits customer shopping experience and store business impact.

3 BACKGROUND

3.1 Preliminary Knowledge

There are three main types of behaviors to describe user interactions in online e-commerce, such as Amazon, eBay, Taobao, etc.

- **Co-View** ($\mathcal{B}_{cv}$): Users who viewed item x also viewed item y.
- **View to purchase** ($\mathcal{B}_{vp}$): Users who viewed item x eventually bought item y.
- **Co-Purchase** ($\mathcal{B}_{cp}$): Users who bought item x also bought item y.

These three types of behaviors are used to construct the user behavior graph, where edges of "Co-View" and "View to purchase" are considered as substitutes, and edges of "Co-Purchase" are treated as complements [28]. In order to further refine the product relationship, on one hand, the combination $(\mathcal{B}_{cv} \cap \mathcal{B}_{vp}) - \mathcal{B}_{cp}$ is often used to learn the substitute product relationship like Product Embedding (P2V) [1], which in our work is a 128-dimension vector representing an item learned by GAT [29]. With this refined logic, it can improve the substitute product relationship quality based on human labels. On the other hand, the combination $\mathcal{B}_{cp} - (\mathcal{B}_{cv} \cup \mathcal{B}_{vp})$ is used to model complementary product relationship.

3.2 Motivation

We use a state-of-the-art complement recommendation model, P-companion [1], as an example to demonstrate the existing defects. Although P-companion is an end-to-end model, it can be described as two steps. The first step is category transition. We define *anchor item* as the item for which we want to provide complementary items, and in this step, the P-companion model trained on the user behavior graph provides multiple categories that are complementary to the category of the anchor item. For example, if the anchor item is an iPhone 11 with *Cellphone* as the category, then the reasonable complementary categories could be *Phone Cases*, *Screen Protectors*, and *Headphones*. In the second step, the P-companion predicts the items under each complementary category and ranks them based on their similarity to the anchor item.

Based on the model design, P-companion focuses on the category-level complement and the item-level similarity, which, however, may not lead to important item-level coherence. Figure 1 shows a recommendation example provided by P-companion. In this example, the category of the anchor item is a cellphone, and the complementary candidate item is a screen protector for a tablet. Based on common sense, a cellphone and a screen protector are complementary at the category level, but these two items together are not a coherent complementary match because they cannot be used together (different sizes) or have any match on important attribute values (such as color or brand). Also, since the P-companion does not provide an explanation, we do not know why the P-companion model provides such a recommendation, and we can only speculate their similarity may come from image tone. Thus, explanations are important for the complement recommendations.

As a result, in response to the problems of existing complementary recommendation models, we propose the concept of *coherent complement recommendation*. We preliminarily define the coherent complement item as a great match with the anchor item, reflected in two aspects: *Compatibility* (i.e., can be used together) and *Relevance*. We will further formalize these two aspects in Section 4.

4 METHODOLOGY

Our task is to provide an explainable and coherent complement recommendation. In this section, we will answer: 1. how to further formalize the concept of *coherent complement*; 2. how to build explainable coherent complement datasets; 3. how the model learns to provide coherent complement recommendations from datasets.

4.1 Data Construction for Coherent Complementary Datasets

In Section 3.2, we define the *coherent complement* concept in two aspects: *Compatibility* and *Relevance*, which are, however, still vague. An intuitive solution is to introduce customer evaluation or expert labeling, but such a solution cannot be scaled up due to huge human labor. Thus, we propose a dataset construction method based on our data source. The following part is the data source introduction:

- **Catalog Data** (*CAT*): Each item is described by a dictionary. The keys are pre-defined attribute names, while the values are corresponding properties. The value is *None* if the attribute does not apply to this item. For example, the dictionary of an iPhone11 product could be {"color": "black", "brand": "Apple", "flavor": "None", ...}.
- **Product Type Important Attribute List** (I): *Product Type* is a special attribute of the dictionary in the *Catalog Data*. Knowledge experts manually select around 10 ranked attributes based on their impoartance for each product type. For example, the important attribute list of *Cellphone* includes *brand*, *color*, and *connectivity_technology*.
- **Compatible Data** (*COM*): A previous work of Amazon that uses DistilBERT [30, 31] and neural networks to predict the compatibility of item pairs in *Electronics* category. The *Electronics* includes several common product types, like *Cellphone* and *TV*. The data contains a list of predicted compatible item pairs.
- **User Behavior Data** (*CP*): We use $\mathcal{B}_{cp} - (\mathcal{B}_{cv} \cup \mathcal{B}_{vp})$ to build the user behavior dataset for complementary item pairs.

$$
\begin{aligned}
&(i_a, i_c) \in CC \leftrightarrow Complement \wedge (Relevant \vee Compatible)\\
&Complement \leftrightarrow (i_a, i_c) \in CP\\
&Compatible \leftrightarrow (i_a, i_c) \in COM\\
&Relevant \leftrightarrow \exists t \in \big(I(PT(i_a)) \cap I(PT(i_c))\big) \wedge CAT(i_a, t) = CAT(i_c, t)
\end{aligned} \tag{1}
$$

According to the data, we instantiate the coherent concept in two ways. If the co-purchase item pair satisfies at least one of the following conditions, we would see it as coherent complement. First *Compatible*, if the two items co-appear as a pair in **Compatible Data**, which means they can be perfectly used together for a joint intent. Second *Relevant*, if the item pair shares the same value of at least one **important attribute** based on their *product types*. For example, "brand" is an important attribute for both *Coffee* and *Coffee Maker*. If a coffee brand is the same as a coffee maker's brand, they should be coherent and very related to each other. Finally, we require the two items to appear as a pair in the **Co-Purchase** dataset since the co-purchase in user behavior data is a strong signal of complement. To formalize this step, we define i_a, i_c as anchor and candidate items, *CC* as coherent complementary dataset, and then we have Eq.(1) as the criteria. After building the *CC* dataset based on the criteria, we will use it for LLM fine-tuning and evaluation.

We understand the coherent criteria are imperfect since the same value of only one important attribute may lead to incoherent complement item pairs, and the item pair only needs to meet one of the conditions of *Compatible* and *Relevant*. However, the relaxed criteria could mitigate the data sparsity, and we will introduce human labeling in the final evaluation in Section 5.5 to show the criteria have better coherence than the current complement recommendation.

4.2 Data Construction for Explanation Datasets

Since *coherent* is a fresh concept we define in Section 4.1, there are no off-the-shelf explanation datasets for this task. A simple solution is to design an explanation template and fill the matched important

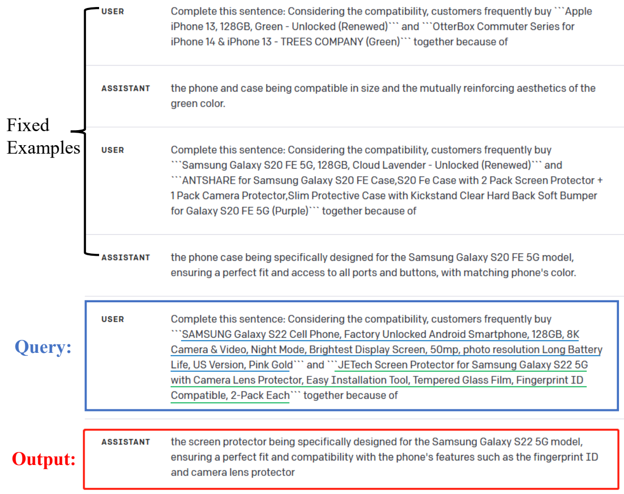

Figure 3: Explanation on *Cellphones* generated by ChatGPT[1].

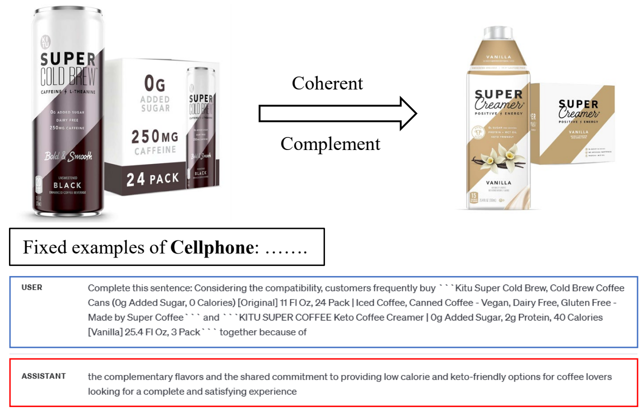

Figure 4: Explanation on *Coffee* generated by ChatGPT.

attribute value as Eq.(1) in the template sentence, like:

$$\{item_x\} \text{ and } \{item_y\} \text{ are coherent complement because they are compatible / their } \{attribute\} \text{ are both } \{value\}. \tag{2}$$

However, this kind of explanation is too rigid and not attractive to customers, even with more manual templates. Considering that ChatGPT[1] can generate vivid sentences, we use it to generate the explanation with a coherent complementary item pair [32]. In real-world e-commerce application scenarios, customers do not tend to read verbose sentences, we expect the explanation to be concise and precise. Thus, we manually write two examples so that ChatGPT can generate the output with the format as we expect by few-shot learning. Specifically, as shown in Figure 3, we always integrate two fixed examples at the beginning of the prompt with the item titles in the last utterance to generate coherent complement explanations.

Besides, we notice that with the inherent knowledge of various domains, ChatGPT is able to provide reasonable explanations under other product types, even if the fixed examples are related to *Cellphones*. Figure 4 shows an explanation generation case of *Coffee* using the same fixed examples of *Cellphones* in the prompt as Figure 3. The explanation of the coffee and the creamer is specific to their common item-level features, low calorie and keto-friendly, and convinces the customers of the benefits of buying them together.

4.3 Explainable and Coherent Complement Recommendation Task

We break the Explainable and Coherent Complement Recommendation (ECCR) task into two parts. First, we need to determine the coherent complement product pairs, and second, we provide an explanation of why they are coherent and complementary.

For the coherent complementary recommendation task (short as the Recommendation Task), following the negative sampling schema, we sample k (as a hyper-parameter) negative items based on the strategy in Section 5.1 for each item pair in CC defined in Eq.(1). Thus, the input of the Recommendation Task includes an anchor item and a list of candidate items with the size of $k+1$, and the backbone model expects to select the best item choice. For the explanation generation task (short as the Explanation Task), the backbone model receives an input containing a pair of items in CC. The expected output is part of a sentence with at most 50 words, and the output should contain enough information to express the reason for coherent complementary matching.

The Explanation Task is a form of language generation, where the Large Language Model (LLM) performs well. Besides, recent OpenP5 work [2] has shown that if we properly union multiple related tasks and fine-tune the same backbone LLM on them, the LLM can learn from multiple angles and each single task performance will be boosted. Considering such an advantage, we also include two classification tasks as auxiliary tasks during the model training process: the complement classification task and the substitution classification task (short as the Classification Task). For these two auxiliary tasks, the model needs to answer whether these two products have a complement/substitution relation with the given item pair, and the expected output should be either *Yes* or *No*.

To union the task forms, we transfer the Recommendation and Classification Tasks into the language generation task, the same as the Explanation Task, and fine-tune one LLM on all four tasks together. We provide some prompts for each task in the Appendix.

To summarize, we have four types of tasks for LLM fine-tuning:

- **Main Task 1: Recommendation Task.** Listwise coherent complementary recommendation.
- **Main Task 2: Explanation Task.** Generate a short explanation for a coherent complementary item pair.
- **Auxiliary Task 1: Complement Classification Task.** Pairwise coherent complementary recommendation.
- **Auxiliary Task 2: Substitution Classification Task.** Pairwise substitution recommendation.

We will analyze the influence of integrating these two auxiliary tasks in the ablation study in Section 5.4.

4.4 Item Indexing

Recent work [33] shows that item indexing methods play an important role in LLM for recommendation tasks. Previous works that use the LLM for recommendation mainly focus on user-item interaction, and thus, some clustering methods based on user-item interaction graphs and using Out-Of-Vocabulary (OOV) for item

[1]https://chat.openai.com

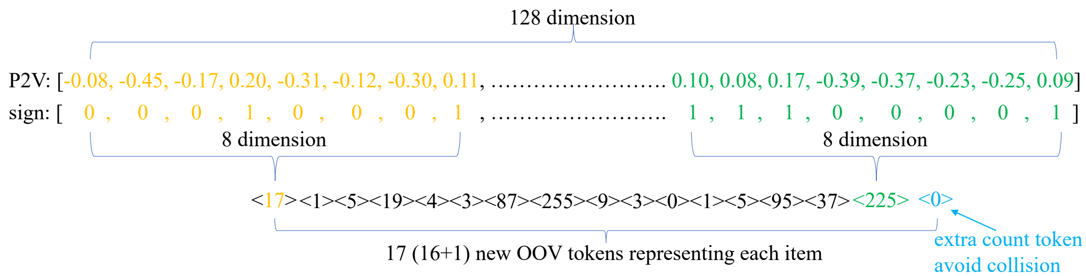

Figure 5: A P2V quantization process example.

indexing would be effective. The reason is that similar items on user-item interaction graphs may share some OOV tokens by clustering, which benefits backbone model learning.

Based on the observation, we propose a P2V Quantization Indexing method for our tasks. The P2V vector learns from text features and user behavior information, as introduced in Section 3.1, and similar products under these two perspectives have similar P2V embeddings. We design the following algorithm, which balances the new OOV token size and item representation length for LLM.

Figure 5 demonstrates an example of our P2V quantization process. First, we separate the 128 dimensions into 16 groups in order, each containing 8 dimensions. Since the absolute value of each dimension is usually not more than 1.0, we map each dimension to a Boolean value, denoted as *sign*. If the value of this dimension is positive, *sign* will be 1; otherwise, *sign* will be 0. Then, for the 8 01-values in the same group, we use the binary-to-decimal algorithm to map to a decimal OOV token value. Please note that we follow [33] work to use a bracket-surrounded number to stand for the OOV token, and thus, <1>, a newly defined OOV token, is different from the number 1 for LLM. To formalize this process, we have:

$$
\begin{aligned}
t_i &= \; < \sum_{j=0}^{j<8} pos(v_{8i+j}) \cdot 2^j > \; \text{ where } i = 0 \ldots 15 \\
pos(x) &= \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}
\end{aligned} \tag{3}
$$

where t_i is the i-th token of the item representation and v_j is the j-th dimension value of the item P2V embedding vector. We notice that in this way if two items have the same sign value for all 128 dimensions, they will have the same quantization representation. Thus, like dealing with Hash collision, we add an extra count token to distinguish different items with the same sign value.

Reviewing the quantization process, each item will be represented by 17 new OOV tokens, and the first 16 tokens will be between <0> and <255> since it is calculated from an 8-bit binary number. Also, in our experiments, the size of the largest item set with the same sign value for all 128 dimensions is smaller than 256. Thus, the backbone LLM will learn 256 extra OOV token embedding from random initialization during the fine-tuning process.

To show the effectiveness of our P2V quantization indexing algorithm, we compare the other two indexing methods, title representation and sequential indexing, in Section 5. Title representation is to directly use the item titles in the prompt. Sequential indexing uses numbers defined in the original vocabulary to index the item from 1 according to the appearance order in the dataset [2], and the candidate order in the Recommendation Task will be randomly shuffled. An example of the sequential indexing is *item*_38.

Dataset	Total item	Anchor item	Rec task samples	Complement classification	Substitution classification	Exp samples
Cellphone	54,439	3,031	82,017	164,034	175,078	6,835
Handbag	246,853	12,843	201,951	403,902	209,331	20,322
Coffee	46,082	4,062	54,737	109,474	55,658	5,601
All	347,374	19,936	338,705	677,410	440,067	32,758

Table 1: Statistics of the training datasets.

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 ***Dataset.*** Our experiments are conducted on a real-world dataset collected from *Amazon.com* including three different product types, namely *Cellphone*, *Coffee*, and *Handbag*. Under each of the three product types, we collect **Catalog Data**, **Product Type Important Attribute List**, **Compatible Data** (only available for *Cellphone*) and **User Behavior Data** as mentioned in Section 4.1, and the **P2V Data** mentioned in Section 4.4.

Since the number of the items in *Amazon.com* is extremely large, we only use the popular Amazon product as the anchor items as the other items are usually less important and the size is much larger. We follow the method in Section 4.1 to build the coherent complement dataset. The anchor items with fewer than 10 coherent complementary items are filtered out as unseen anchor item testing, since they do not show in the training or validation datasets. We expect the LLM to gain domain knowledge and be able to generalize to unseen cases. For the remaining data, we split the coherent complementary items into training, validation, and testing datasets in 8:1:1 for each anchor item. Also, we follow the solution in Section 4.2 to build the explanation dataset for coherent complementary item pairs. Considering the strong language learning and generation ability, we only apply 10% stratified sampling based on product type pairs for all training, validation, and testing datasets, respectively. Finally, the training dataset statistics are shown in Table 1, and the testing and unseen anchor item testing dataset statistics are shown in the Appendix. From the statistics, we can see that the *Handbag* is the largest one among the three product types, and the *Cellphone* has the smallest number of anchor items but each anchor item has the most coherent complement items, reflected in the number of the Recommendation Task samples over the number of anchor items.

For the Recommendation Task, we set the negative candidate set size $k = 20$ for each coherent complement item pair. We build the negative candidate pool for each anchor item from three sources: 1.

LLM	#Param	Fine-tuning time (min / epoch)	Recommendation HR@5	Recommendation NDCG@5	Explanation (%) BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L	Classification F1-Complement	F1-Substitution
T5-S	60M	**19.0**	0.5545	0.3812	36.7309	56.8252	39.0976	50.8865	**0.8950**	0.9388
T5-B	220M	69.0	**0.5737**	**0.3889**	**37.1520**	**57.1968**	**39.7864**	**51.4705**	0.8881	**0.9392**
LLaMA-3B	3B (tunable 200M)	693.5	0.3170	0.2002	36.7023	56.5216	39.4460	50.8438	0.7573	0.8640

Table 2: Performance of different backbone LLMs using P2V quantization indexing on *Cellphone*.

Dataset	Cellphone		Coffee		Handbag	
Model	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
CP	0.4810	0.2909	0.4908	0.2695	0.4888	0.2767
PMSC	0.3726	0.3065	0.3697	0.2323	0.4045	0.2563
P-Companion	0.5098	0.2872	0.5170	0.2798	0.3582	0.1949
T5-S (Ours)	**0.5545**	**0.3812**	**0.7026**	**0.5774**	**0.7266**	**0.6019**

Table 3: Performance of different models on Recommendation Task.

Dataset	Method	Recommendation HR@5	NDCG@5	Explanation (%) BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L
Cellphone	Title	0.4658	**0.4137**	36.6610	**56.8873**	**39.1667**	**51.1066**
	Seq	0.4916	0.3570	35.9842	56.1791	38.4759	50.5354
	P2V	**0.5545**	0.3812	**36.7309**	56.8252	39.0976	50.8865
Coffee	Title	0.5767	0.5209	15.6374	39.8639	19.9234	33.1805
	Seq	**0.7428**	**0.6333**	14.5328	40.0190	19.1537	32.7500
	P2V	0.7026	0.5774	**15.7357**	**41.0696**	**20.4086**	**33.9282**
Handbag	Title	0.6315	0.5838	12.8479	**40.0494**	16.4402	**32.7442**
	Seq	0.6709	0.5721	12.5280	39.7307	16.0910	32.3682
	P2V	**0.7266**	**0.6019**	**12.9881**	39.9086	**16.4851**	32.5529

Table 4: Performance of different indexing methods on T5-S [34].

Dataset	Cellphone		Coffee		Handbag	
Metrics	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
Title	0.4628	0.4100	0.5093	0.4364	0.5039	0.4531
Sequential	0.5735	0.4376	0.5623	**0.4556**	0.5366	0.4235
P2V	**0.6255**	**0.4963**	**0.5796**	0.4437	**0.5890**	**0.4659**

Table 5: Recommendation task performance of different indexing methods on T5-S [34] using the datasets with unseen anchor items.

Dataset	Cellphone		Coffee		Handbag	
Metrics	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
learn 1 PT, test 1 PT	**0.5545**	**0.3812**	0.7026	0.5774	**0.7266**	**0.6019**
learn 3 PT, test each	0.5498	0.3674	**0.7931**	**0.6481**	0.6803	0.5427

Table 6: Recommendation task performance of using different numbers of datasets on T5-S [34] fine-tuning with P2V quantization.

co-purchase but incoherent items (i.e., $Complement \wedge \neg(Compatible \vee Relevant)$ in Eq.(1)); 2. substitution items; 3. other unrelated items (if the first two are insufficient to fill the k candidates). The negative candidates for every Recommendation Task query are stratified sampled from the negative candidate pool according to the rate of different data sources. For the coherent complement classification task, the positive pairs are from the coherent complement dataset based on Eq.(1), and for each positive pair, sample one negative case from the negative candidate pool of this anchor item. For the substitution classification task, the positive cases are from $(\mathcal{B}_{cv} \cap \mathcal{B}_{vp}) - \mathcal{B}_{cp}$ of the anchor item as mentioned in Section 3.1, and the negative cases are the coherent complement item pairs.

Dataset	Setting	BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L
Cellphone	learn 1 PT, test 1 PT	**36.7309**	**56.8252**	**39.0976**	**50.8865**
	learn 3 PT, test each	35.3478	55.3783	37.8605	49.7384
Coffee	learn 1 PT, test 1 PT	**15.7357**	**41.0696**	**20.4086**	**33.9282**
	learn 3 PT, test each	14.0519	39.1990	18.6288	32.0069
Handbag	learn 1 PT, test 1 PT	**13.1495**	**40.4601**	**16.8321**	**32.9428**
	learn 3 PT, test each	12.0516	39.3285	15.7426	31.9599

Table 7: Explanation task performance of using different numbers of datasets on T5-S [34] fine-tuning with P2V quantization.

5.1.2 ***Evaluation Metrics.*** For the Recommendation Task, we apply beam search to the backbone LLM output generation and use the top-5 generated different outputs from the backbone LLM and apply Hit Ratio (HR) @ 5 [35] and Normalized Discounted Cumulative Gain (NDCG) @ 5 [36, 37] as evaluation metrics. For the Explanation Task, we use typical metrics BLEU-4 [38], as well as ROUGE-1, ROUGE-2, and ROUGE-L [39] to compare the difference between the ground truth explanation (generated by ChatGPT) and the model generated output for quantitative quality evaluation. We also have two auxiliary classification tasks, complement and substitution, and we use the F1-score as a precision-recall balanced metric for evaluation. All the metrics are the higher the better.

We understand the ground-truth data of Recommendation and Explanation Tasks may not be reliable since it is the first time to propose the "coherent complement" concept and build coherent complementary datasets to the best of our knowledge. Thus, apart from offline metrics, we introduce human labeling in the evaluation phase, and the detail is shown in Section 5.5.

5.1.3 ***Baseline Models.*** We compare our LLM performance on the Recommendation Task with the following baseline models:

- **Co-Purchase (CP)** is one of the most straightforward ways. We output the items that are in the Co-Purchase ($\mathcal{B}_{cp}$) and not in Co-View ($\mathcal{B}_{cv}$) or View-to-Purchase $\mathcal{B}_{vp}$ for recommendation.
- **PMSC** [28] adopts relation-aware parameters to model multiple item relations with path logic constraint loss and later feeds into a neural network for classification. Each product item has its source and target embeddings for query and candidate contexts.
- **P-Companion** [1] is an end-to-end framework, leveraging product type transitions and then a transfer metric learning network for diversified complementary product recommendation.

We do not have a baseline model for the Explanation Task as it is a new research question. We compare our generated explanations with ChatGPT's explanations by human labeling shown in Section 5.5. For the Classification Tasks, since they are auxiliary tasks, we do not compare the model with other baseline models, but compare our model among different experiment settings.

5.1.4 ***Implementation Details.*** Our fine-tuning implementation borrows ideas from OpenP5 [3], which is a framework using LLM pre-trained checkpoints for downstream recommendation tasks. We select three open-source LLM checkpoints, T5-Small (T5-S) [34], T5-Base (T5-B) [34], and OpenLLaMA-3B (LLaMA-3B) [40], as our backbone model choices. Due to the large number of parameters of LLaMA-3B, we apply QLoRA with 4-bit model quantization [41, 42] and the rank as 8 on LLaMA-3B for memory and time efficiency, while for T5-S and T5-B, we update all their parameters in our fine-tuning process. The loss used for fine-tuning and the validation dataset evaluation is token-level Cross-Entropy (CE).

Apart from backbone LLM selections, we have three different item indexing options: our P2V quantization algorithm, title representation, and sequential indexing, as mentioned in Section 4.4. Since the input token length limitation of T5 models is 512 for the best practice, for the title representation, we keep the first at most 20 words and add "..." with the assumption that the beginning of the title contains most of the key information. Also, we have three product-type datasets, and we can also fine-tune our models on these three datasets together or separately. Thus, we have four different dataset settings. Besides, we use the unseen anchor item dataset to demonstrate our backbone model's generalization ability on unseen cases of the fine-tuning product type.

In the backbone LLM fine-tuning process, we design 10 different prompts for each task discussed in Section 4.3, but only sample 2 prompts for each data in each fine-tuning epoch. This approach following OpenP5 [3] is to balance efficiency and data variety for better LLM learning. Also, to mitigate the data imbalance and potential catastrophic forgetting issues [43], each data batch only contains one task under one product-type dataset, and we alternate data batches derived from different tasks and different product-type datasets and repeat small-size data iterations until all tasks and datasets finish in one epoch [3].

Here are other hyper-parameter settings. We use AdamW [44] as the model optimizer with a learning rate of 0.001. The batch size is among $\{8, 16, 32, 64\}$ by grid search. For the Recommendation Task, we set the negative candidate number as $k = 20$, and we adhere to the negative candidate pool construction method outlined in Section 4.3. The maximum number of fine-tuning epochs is set to 30. If, during fine-tuning, the CE loss on the validation dataset exhibits a continuous increase for more than 5 epochs or if the smallest CE loss observed on the validation dataset occurred over 5 epochs ago, the fine-tuning process will be halted to prevent over-fitting. We select the model of the epoch with the smallest validation loss to report the performance in the testing dataset. When testing, we randomly select one prompt for each task. Different prompts do not have significant testing performance differences.

5.2 Performance Comparison

Since we have three backbone model choices, four different product-type dataset settings, and three item representation methods, due to the page limitation, we only display several small tables for each conclusion rather than showing an extremely huge table for experiment results. The bold numbers in all tables represent the best performance under each setting.

5.2.1 ***Model Comparison.*** We conduct experiments on *Cellphone* dataset with our P2V quantization indexing method on different backbone LLMs. The results are shown in Table 2. The running time is based on 8x NVIDIA A10G (24GB) GPUs. We can see that all three backbones perform similarly on the Explanation Task, but T5-S and T5-B much outperform LLaMA-3B on the Recommendation and Classification Tasks. One possible reason could be that we only tune 200M parameters of LLaMA-3B due to our limited GPU memory, which takes only a small part of its 3B parameters. Besides, the performance of T5-S and T5-B has insignificant performance differences, but due to over three times better efficiency, we use T5-S as the default backbone model in the following experiments.

Table 3 displays the results of our model and baseline models on the Recommendation Task on different datasets. Our model performs much better than every baseline, which validates that our model can improve the coherent aspect when providing the complement recommendations.

5.2.2 ***Indexing Method Comparison.*** To compare the performance of different indexing methods, we employ T5-S as the backbone and present the results of the Recommendation Task using the testing datasets in Table 4. It is evident that in the Recommendation Task, P2V quantization outperforms the sequential indexing and title-based methods in half of the cases. In the Explanation Task, while the results are comparable, the sequential method consistently ranks lowest, with P2V quantization generally outperforming the title-based approach. Notably, the *Cellphone* and *Handbag* results using the P2V and title methods exhibit a BLEU-ROUGE trade-off, reminiscent of the precision-recall trade-off.

Table 5 provides an overview of the Recommendation Task performance across the three indexing methods in unseen anchor item testing datasets. In most cases, the P2V method prevails, affirming the superior generalization capability of our proposed indexing algorithm when faced with unseen anchor item scenarios. This enhanced generalization arises from the integration of text features and user behavior information into P2V latent embedding, which is subsequently mapped in the quantization indexing. Such adaptability is particularly valuable since it accommodates scenarios where not all items are used for training, and new items are continuously introduced as potential anchor items.

5.2.3 ***Multiple Product-type Datasets Co-training.*** We do not expect to fine-tune an LLM for each product-type (PT) dataset since many product types exist in the real world. Hence, we aim to assess whether the performance consistency is maintained when training a single LLM on all three product-type datasets, as opposed to training a separate model for each product type. Table 6 and 7 display the performance comparison using T5-S with P2V quantization indexing on the Recommendation and Explanation Tasks, respectively. Note that if the backbone model is trained on a single PT dataset, it will test on that PT; if the model learns from three PT datasets, it will test on all three PT datasets. We can see that through a slight drop, the performance of three PT training is not significantly worse than one PT training, even if the three PTs vary widely in semantics. Thus, it is an applicable solution to fine-tune an LLM on the union of all PT datasets. Another interesting discovery is that LLM using three-PT training reaches a huge performance gain in the *Coffee* dataset on the Recommendation Task. As mentioned 5.1.4, the fine-tuning process alternates data batches of different tasks and PTs and repeats small-size data iterations until all tasks and datasets finish in one epoch. Since the Recommendation Task

Tasks	Recommendation		Classification	
Metrics	HR@5	NDCG@5	F1 (Comp)	F1 (Subs)
w/o Comp Classification	0.5146	0.3639	/	0.9318
w/o Subs Classification	0.4656	0.3283	0.8920	/
w/o Extra collision token	0.4790	0.3352	0.8922	0.9309
w/o Ablation setting	**0.6612**	**0.5202**	**0.9045**	**0.9469**

Table 8: Average results over 3 PT datasets in ablation study.

on *Coffee* is the smallest data among all tasks and datasets (except for the Explanation Tasks), it got the most adequate training among all PT datasets and tasks under three-PT training, and thus, the LLM performs better on this task-PT combination. Another piece of evidence is that the Recommendation performance on *Handbag* drops most and keeps close on *Cellphone* when switching to three-PT training as the data sample size is *Handbag* > *Cellphone* > *Coffee* and the repeated training times are *Coffee* > *Cellphone* > *Handbag*.

5.3 Online A/B Test

To evaluate the online performance of our proposed solution, we used revenue gain as the metric for production evaluation. We compared our proposed solution (the treatment group) to the baseline solution based on the framework [1] (the control group). Our approach involves suggesting products corresponding to the items customers have added to their shopping carts. After conducting a 2-week online testing, we observe significant commercial success:

- Annual revenue was improved by 0.1%;
- Annual sales were improved by 0.2%.

5.4 Ablation Study

We conduct the ablation study to validate the effectiveness of our proposed design. We have the following three ablation options:

- Option 1: remove the complement classification task.
- Option 2: remove the substitution classification task.
- Option 3: remove the extra collision token for P2V quantization (thus using 16 tokens to represent each item).

We use the T5-S as the backbone with P2V quantization and the average results over 3 PT datasets are in Table 8. We can see that all the ablation settings perform worse than our original setting. Among the three options, removing the substitution task drops the most, indicating substitution relation learning benefits the coherent complement recommendation task. Besides, removing the complement classification drops the least as it is a variant of the Recommendation Task. Thus, if the data is too large compared to limited fine-tuning time, Option 1 may be the first consideration.

5.5 Human Evaluation

Since it is the first time to propose and discuss the concept of coherent complement (CC), there are no benchmark datasets nor explanation datasets for CC item pairs. Thus, we introduce human labeling to the dataset and our LLM output evaluation.

There are two labeling tasks in total. In the first task, the knowledge experts assume they are customers and have decided to buy the anchor item. The experts need to label whether the provided CC items are a great match with the anchor item. We evaluate this task by sampling around 200 item pairs from three datasets: fine-tuned T5-S (our) output, CC dataset constructed through Eq.(1), and P-companion complement recommendation. The results under three different product-type datasets are shown in Table 9. We can see that our model output is much better than the existing Amazon complementary recommendation in the coherent aspect. More surprisingly, although our model is fine-tuned on the CC dataset, it outperforms CC data, probably due to LLMs' pre-trained knowledge and generalization ability.

	Cellphone	Gain	Coffee	Gain	Handbag	Gain
P-companion	0.630	0.00%	0.310	0.00%	0.420	0.00%
CC Dataset	0.740	+17.46%	0.355	+14.52%	0.565	+34.52%
T5-S (Our)	**0.790**	**+25.40%**	**0.380**	**+22.58%**	**0.625**	**+48.81%**

Table 9: Human labeling on the coherent rate.

	Cellphone			Coffee			Handbag		
	Corr	Info	Reason	Corr	Info	Reason	Corr	Info	Reason
ChatGPT	1.000	**1.000**	1.000	**1.000**	0.9057	**0.9811**	1.000	0.8903	0.8774
T5-S (Our)	1.000	0.9892	1.000	0.8868	0.9057	0.8679	1.000	**0.9613**	**1.000**

Table 10: Human Evaluation on the explanation quality.

The second task is to label the explanation data, one from ChatGPT generation as mentioned in Section 4.2, and the other from our fine-tuned T5-S model generation. We require the knowledge experts to label the explanation in three aspects: Correct (Corr), Informative (Info), and Reasonable (Reason). These three aspects imply the explanation does not provide wrong information, is not general but specific to this item pair, and provides the matching aspect convincing to customers. Examples in Section C.2 explain the three aspects. We first sample and filter out the item pairs that are not coherent complement, and report the results on the remaining about 200 data samples under each setting in Table 10. Both our fine-tuned T5-S and ChatGPT can provide a high rate of good explanations and our model is even better overall. Similar to the first human labeling task's situation, although the explanation training data is from ChatGPT, multi-task fine-tuning on domain data also benefits item understanding to provide a better explanation.

6 CONCLUSIONS AND FUTURE WORK

In this work, we propose the problem that current complement recommendations are not compatible or relevant enough with the anchor item, and based on this, we propose the concept of "coherent complement". We design the P2V quantization indexing method that integrates text features and user behavior information for Large Language Model (LLM) fine-tuning. Through extensive experiments, our multi-task fine-tuned LLM outperforms current complement recommendation models in the coherent aspect and can provide high-quality explanations validated by expert labeling.

There are many possible extensions of our work. First, the coherent rate of our dataset, though improved, still has an improvement space. New solutions can be proposed to construct a coherent complement dataset with better quality. Second, our indexing method is a primary solution to include user behavior information for LLM learning. Other quantization methods or even other information integration forms are worth a try. Last, the largest parameter numbers of our LLMs are 3B due to the limited GPU memory. With the rapid development of memory and efficiency improvement techniques for LLMs, other larger LLMs could be used for performance gain on the explainable and coherent complement recommendation tasks.

REFERENCES

[1] Junheng Hao, Tong Zhao, Jin Li, Xin Luna Dong, Christos Faloutsos, Yizhou Sun, and Wei Wang. P-companion: A principled framework for diversified complementary product recommendation. In *CIKM 2020*, 2020.

[2] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the Sixteenth ACM Conference on Recommender Systems*, 2022.

[3] Shuyuan Xu, Wenyue Hua, and Yongfeng Zhang. Openp5: Benchmarking foundation models for recommendation. *arXiv:2306.11134*, 2023.

[4] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*, 2022.

[5] Wang-Cheng Kang, Eric Kim, Jure Leskovec, Charles Rosenberg, and Julian McAuley. Complete the look: Scene-based complementary product recommendation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10532–10541, 2019.

[6] Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 785–794, 2015.

[7] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

[8] Yin Zhang, Haokai Lu, Wei Niu, and James Caverlee. Quality-aware neural complementary item recommendation. In *Proceedings of the 12th ACM conference on recommender systems*, pages 77–85, 2018.

[9] Yankai Chen, Tuan Truong, Xin Shen, Ming Wang, Jin Li, Jim Chan, and Irwin King. Topological representation learning for e-commerce shopping behaviors, 2023.

[10] Weiwen Liu, Yin Zhang, Jianling Wang, Yun He, James Caverlee, Patrick PK Chan, Daniel S Yeung, and Pheng-Ann Heng. Item relationship graph neural networks for e-commerce. *IEEE Transactions on Neural Networks and Learning Systems*, 33(9):4785–4799, 2021.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[12] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

[13] Yuchen Zhuang, Xin Shen, Yan Zhao, Chaosheng Dong, Ming Wang, Jin Li, and Chao Zhang. G-sto: Sequential main shopping intention detection via graph-regularized stochastic transformer. *arXiv preprint arXiv:2306.14314*, 2023.

[14] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. Text is all you need: Learning language representations for sequential recommendation. *arXiv preprint arXiv:2305.13731*, 2023.

[15] Huiyuan Chen, Yusan Lin, Menghai Pan, Lan Wang, Chin-Chia Michael Yeh, Xiaoting Li, Yan Zheng, Fei Wang, and Hao Yang. Denoising self-attentive sequential recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 92–101, 2022.

[16] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.

[17] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[18] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[19] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, et al. How can recommender systems benefit from large language models: A survey. *arXiv preprint arXiv:2306.05817*, 2023.

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

[21] Yiding Liu, Weixue Lu, Suqi Cheng, Daiting Shi, Shuaiqiang Wang, Zhicong Cheng, and Dawei Yin. Pre-trained language model for web-scale retrieval in baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3365–3375, 2021.

[22] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1652–1656, 2021.

[23] Qi Zhang, Jingjie Li, Qinglin Jia, Chuyuan Wang, Jieming Zhu, Zhaowei Wang, and Xiuqiang He. Unbert: User-news matching bert for news recommendation. In *IJCAI*, pages 3356–3362, 2021.

[24] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *arXiv preprint arXiv:2305.00447*, 2023.

[25] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR, 2023.

[26] Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*, 2023.

[27] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. Gpt4rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879*, 2023.

[28] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. A path-constrained framework for discriminating substitutable and complementary products in e-commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 619–627, 2018.

[29] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[30] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[31] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[32] Joyce Zhou and Thorsten Joachims. Gpt as a baseline for recommendation explanation texts. *arXiv preprint arXiv:2309.08817*, 2023.

[33] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. *arXiv:2305.06569*, 2023.

[34] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[35] Robin Burke, Bamshad Mobasher, Runa Bhaumik, and Chad Williams. Segment-based injection attacks against collaborative filtering recommender systems. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 4–pp. IEEE, 2005.

[36] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[37] Stefan Buttcher, Charles LA Clarke, and Gordon V Cormack. *Information retrieval: Implementing and evaluating search engines*. Mit Press, 2016.

[38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[39] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[40] Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023.

[41] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[42] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[43] Fajie Yuan, Guoxiao Zhang, Alexandros Karatzoglou, Joemon Jose, Beibei Kong, and Yudong Li. One person, one model, one world: Learning continual user representation without forgetting. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 696–705, 2021.

[44] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

A PROMPT FOR FINE-TUNING

As mentioned in Section 4.3, each task has ten distinct prompts in total. We provide one prompt example for each task here:

- **Recommendation Task**: We define the coherent complementary products of one anchor product if the complement product is compatible or shares a certain important attribute value with the

anchor product. Now given the anchor product {*anchor*}, what is the coherent complementary product among {*candidates*}?
- **Explanation Task** (Sentence Completion): Assume you are a shopping assistant. I am considering buying {*anchor*} and you think that buying {*candidate*} together will be compatible or improve my experience due to
- **Complement Classification Task**: Are there any significant synergies or interdependencies between the features or functionalities of {*anchor*} and {*candidate*}?
- **Complement Substitution Task**: A substitution product is defined as a product, whose use is directly related to another such that a decrease in demand for one product results in an increase for the other. In practice, customers usually view the two products together but only purchase one of them. Is {*candidate*} a substitution product for {*anchor*}?

Note: Braces surround the argument of the prompt.

B MORE DATA STATISTICS

Dataset	Total item	Anchor item	Rec Task samples	Complement classification	Substitution classification	Exp samples
Cellphone	54,439	3,031	8,706	17,412	20,305	694
Handbag	246,853	12,843	20,013	40,026	29,148	2,218
Coffee	46,082	4,062	5,331	10,662	7,808	616
All	347,374	19,936	34,050	68,100	57,261	3,528

Table 11: Statistics of the validation and testing datasets.

Dataset	Total item	Anchor item	Rec Task samples	Complement classification	Substitution classification
Cellphone	54,439	5,374	17,021	34,042	33,864
Handbag	246,853	68,515	205,120	410,240	407,360
Coffee	46,082	23,677	88,128	176,256	175,706
All	347,374	97,566	310,269	620,538	616,930

Table 12: Statistics of the unseen anchor item testing datasets.

C CASE STUDY

C.1 Coherent Complement Recommendation Task

Candidate Item:
Jocko Mölk Whey Protein Powder (Chocolate) - Keto, Probiotics, Grass Fed, Digestive Enzymes, Amino Acids, Sugar Free Monk Fruit Blend - Supports Muscle Recovery & Growth - 31 Servings (2lb Old Tub)

Anchor Item:
Real Good Coffee Company - Flavored Ground Coffee - Pumpkin Spice Light Roast Coffee - 100% Arabica Beans - Roasted and Ground in Seattle, WA - 2 Pound Bag

Figure 6: Incorrect coherent complement example from Eq.(1).

We present examples that provide different recommendations among various models. Figure 1 displays an example recommended by the P-companion [1] model, but it is classified as *NOT* coherent complement by our model. In Figure 6, we provide an example based

Candidate Item:
LOVE MEI LG V60 ThinQ Case with Tempered Glass Screen Protector Shockproof Scratch Proof Hybrid Metal and Silicone Gel Heavy Duty Armor Defender Tough Back Cover for V60 ThinQ / V60 (Black)

Anchor Item:
LG V60 ThinQ 5G LM-V600AM 128GB GSM Unlocked 6.8" P-OLED DisplayTriple 64MP + 13MP + 0.3MP Camera Phone Classy Blue

Explanation from our model:
the phone case being specifically designed for the LG V60 ThinQ model, ensuring a perfect fit and access to all ports and buttons, with the added protection of a tempered glass screen protector.

Figure 7: Case study from fine-tuned T5-S [34] with the explanation.

on Eq.(1), but it is also classified as *NOT* coherent complement by our model and domain experts. The two items are co-purchased, probably because they are consumable supplies. The same important attribute value is "2 pounds" under *item weight*, but domain experts argue that this point is insufficient for a coherent complement relation. The example also implies our criteria for the *CC* datasets have room for improvement. Figure 2 and 7 showcase the examples recommended by our model but not in the recommendations from P-companion or the *CC* dataset established by Eq.(1). Experts align with our model's recommendation and explanation.

C.2 Explanation Task

We present some explanation examples here.

Anchor	Caribou Coffee Reindeer Blend, Single-Serve Keurig K-Cup Pods, Dark Roast Coffee Pods, 88 Count
Candidate	Nestle Coffee mate Coffee Creamer, Peppermint Mocha, Liquid Creamer Singles, Non Dairy, No Refrigeration, Box of 50 Singles (Pack of 4)
ChatGPT (Correct)	the complementary flavors and aromas that enhance the overall coffee-drinking experience, with the dark roast of the Caribou Coffee Reindeer Blend perfectly complemented by the peppermint mocha flavor of the Nestle Coffee Mate creamer.
Ours (Incomplete Sentence)	the perfect combination of rich and flavorful coffee from Caribou Coffee Reindeer Blend, complemented by the indulgent and creamy taste of peppermint mocha coffee creamer from Nestle Coffee Mate, enhancing the overall coffee-

Table 13: Explanation example cases at *Correct* aspect.

Anchor	Vera Bradley Women's Recycled Lighten Up Reactive Sling Backpack Bookbag, Sunflower Sky, One Size US
Candidate	Vera Bradley Women's Cotton Deluxe Zip Id Case Wallet with RFID Protection
ChatGPT	the matching design and complementary functionality of both products.
Ours (Informative)	the cohesive design and matching patterns of the Vera Bradley brand, creating a coordinated and stylish look for their accessories.

Table 14: Explanation example cases at *Informative* aspect.

Anchor	SUPER Charcoal Roasted White Coffee Classic
Candidate	SUPER NutreMill Soy Milk Powder No Sugar
ChatGPT	the complementary flavors and nutritional combination they offer when consumed together.
Ours (Reasonable)	the complementary flavors and the ability to create delicious and creamy beverages by combining the rich and smooth taste of the super charcoal roasted white coffee with the creamy and indulgent smooth texture of the Nutremill Soy milk powder.

Table 15: Explanation example cases at *Reasonable* aspect.