Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5)

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, Yongfeng Zhang
Department of Computer Science, Rutgers University, NJ 08854, US
{sg1309,shuchang.syt.liu,zuohui.fu,yingqiang.ge,yongfeng.zhang}@rutgers.edu

ABSTRACT

For a long time, different recommendation tasks typically require designing task-specific architectures and training objectives. As a result, it is hard to transfer the learned knowledge and representations from one task to another, thus restricting the generalization ability of existing recommendation approaches, e.g., a sequential recommendation model can hardly be applied or transferred to a review generation method. To deal with such issues, considering that language can describe almost anything and language grounding is a powerful medium to represent various problems or tasks, we present a flexible and unified text-to-text paradigm called "Pretrain, Personalized Prompt, and Predict Paradigm" (**P5**) for recommendation, which unifies various recommendation tasks in a shared framework. In P5, all data such as user-item interactions, user descriptions, item metadata, and user reviews are converted to a common format — natural language sequences. The rich information from natural language assists P5 to capture deeper semantics for personalization and recommendation. Specifically, P5 learns different tasks with the same language modeling objective during pretraining. Thus, it serves as the foundation model for various downstream recommendation tasks, allows easy integration with other modalities, and enables instruction-based recommendation based on prompts. P5 advances recommender systems from shallow model to deep model to big model, and will revolutionize the technical form of recommender systems towards universal recommendation engine. With adaptive personalized prompt for different users, P5 is able to make predictions in a zero-shot or few-shot manner and largely reduces the necessity for extensive fine-tuning. On several recommendation benchmarks, we conduct experiments to show the effectiveness of P5. To help advance future research on Recommendation as Language Processing (RLP), Personalized Foundation Models (PFM), and Universal Recommendation Engine (URE), we release the source code, dataset, prompts, and pretrained P5 model at https://github.com/jeykigung/P5.

KEYWORDS

Recommender Systems; Natural Language Processing; Multitask Learning; Personalized Prompt; Language Modeling; Unified Model

1 INTRODUCTION

For the past decades, recommender systems have witnessed significant advancements and played an essential role in people's daily life, helping their micro decisions and fulfilling their demands with outstanding accuracy. In retrospect, we can summarize the development trend of modern recommender systems – towards a more comprehensive system that accommodates diverse features and a wide spectrum of application scenarios.

On one hand, feature engineering and learning in recommender systems has evolved greatly from simple to complex. In early ages, recommender systems typically adopt logistic regression or collaborative filtering [25, 35, 50, 52] which utilize user-item interaction records to model users' behavioral patterns. Later on, the contextual features such as user profile and item metadata are further integrated into the system through more sophisticated models such as factorization machines [48] and GBDT [20]. Recently, deep neural network models [3, 5, 19, 74] facilitate crossing and combination among even more diverse and sophisticated features. As a result, these models gain better representation ability compared with traditional feature engineering based approaches.

On the other hand, more recommendation tasks have emerged. Except for classical rating prediction and direct user-item matching-based recommendation tasks, recent works are broadening the spectrum to new tasks and scenarios such as sequential recommendation [21, 60, 63, 80], conversational recommendation [8, 61, 76], explainable recommendation [17, 31, 62, 70, 75, 77] and so on. While the approaches to the aforementioned recommendation tasks are often proposed separately, there is an evident trend of utilizing multiple recommendation tasks to jointly learn the transferable representations [31, 56, 57, 72]. Although existing recommender systems achieved great success, there is still a considerable gap between current solutions and the foreseeable intersection of the aforementioned trends – a comprehensive recommender system that can accommodate diverse features and different types of tasks. Since recommendation tasks usually share a common user–item pool and have overlapping contextual features, we believe it is promising to merge even more recommendation tasks into a unified framework so that they can implicitly transfer knowledge to benefit each other and enable generalization to other unseen tasks.

Inspired by the recent progress in multitask prompt-based training [1, 51, 67], in this work, we propose a unified "Pretrain, Personalized Prompt & Predict Paradigm" (denoted as **P5**). We show that P5 is possible to learn multiple recommendation related tasks together through a unified sequence-to-sequence framework by formulating these problems as prompt-based natural language tasks, where user–item information and corresponding features are integrated with personalized prompt templates as model inputs. P5 sheds light

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, Yongfeng Zhang
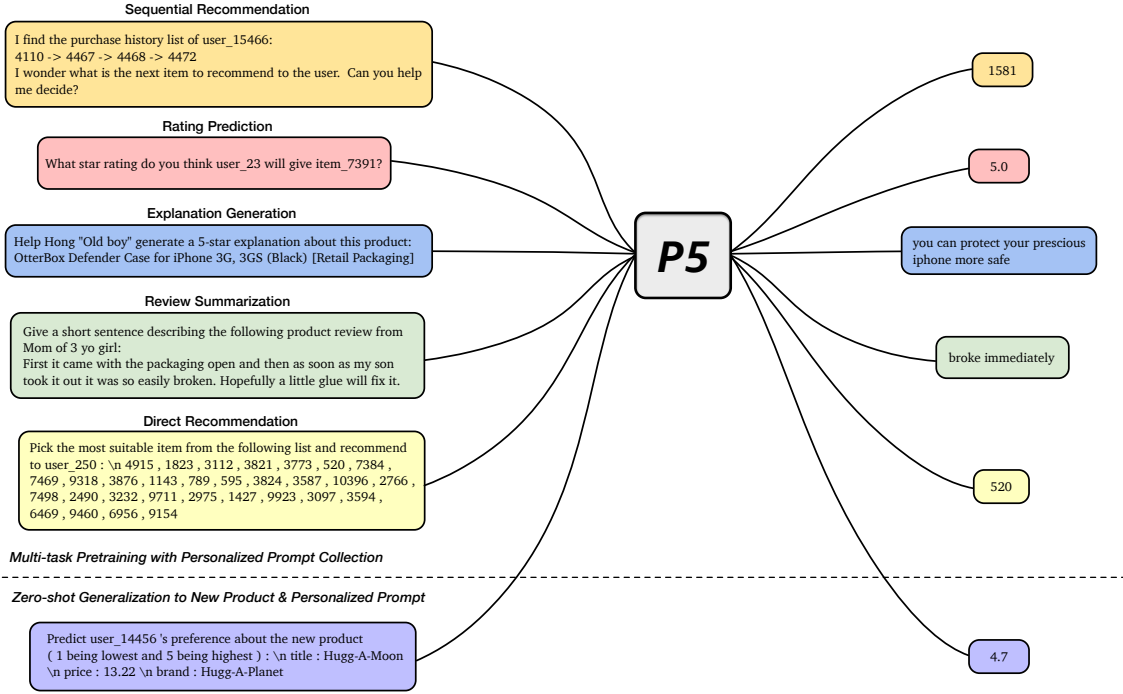
Figure 1: P5 pretrains on an encoder–decoder Transformer model that takes in textual inputs and produces target responses. We trained P5 on a multitask collection of personalized prompts. After multitask prompt-based pretraining on recommendation datasets, P5 achieves the capability of zero-shot generalization to unseen personalized prompts and new items.

on a promising technical route for unified and instruction-based recommendation. It has *three* main advantages:

1) P5 deeply immerses recommendation models into a full language environment, where all recommendation tasks are reformulated to NLP tasks with the help of personalized prompts. Since language grounding is sufficiently flexible and powerful to express various kinds of features in text templates, so there is no need to design feature-specific encoders. As a result, P5 can exploit the abundant semantics and knowledge inside the training corpora;

2) P5 integrates multiple recommendation tasks into a shared text-to-text encoder-decoder architecture and trains them with the same language modeling loss rather than designing task-specific architectures and objective functions. In other words, P5 treats all personalized tasks as a conditional text generation problem;

3) Trained with instruction-based prompts, P5 attains sufficient zero-shot performance when generalizing to novel personalized prompts or unseen items in other domains.

In our experiments, we study how P5 performs compared with task-specific approaches on all five task families as well as evaluating P5's zero-shot generalization ability. We also conduct several ablation studies to justify the design details of P5 framework. Overall, our main contributions can be outlined as follows:

- To the best of our knowledge, this is the first work to propose a unified "Pretrain, Personalized Prompt & Predict Paradigm" which integrates various recommendation related tasks into a shared conditional language generation framework.
- We create a collection of personalized prompts that cover five different recommendation task families.

- According to the experimental results, P5 achieves promising performances on the five task families when taking seen prompt templates as model inputs.
- P5 shows sufficient zero-shot generalization ability for novel personalized prompts and new items in unseen domains.

2 RELATED WORK

Unified Frameworks. Many prior works have pursued to solve various tasks in a unified model. As early pioneers, T5 [47] and GPT-3 [2] unifies NLP downstream tasks through text-to-text encoder–decoder framework and autoregressive language modeling, respectively. They both allow effective knowledge sharing among different tasks based on a common pretrained language model. Following this trend, recent advances started to focus on unifying large-scale language tasks [1, 51, 67] or cross-modality applications [6, 66, 71] through a shared sequence-to-sequence framework, where different types of tasks and modalities are all expressed in the format of natural language. However, aforementioned methods never consider personalization in their sequence-to-sequence models. Recently, a line of work [56, 57, 72] attempt to learn universal user representations which are easily transferrable to downstream tasks. One limitation of these methods is that they still require additional finetuning on downstream datasets. In contrast, our P5 first takes personalization into an encoder-decoder Transformer model that can generalize to a wide spectrum of recommendation related application scenarios – tasks that naturally require personalization. Moreover, with the help of prompt-based pretraining, P5 acquires zero-shot generalization ability when transferring to unseen prompts and items.
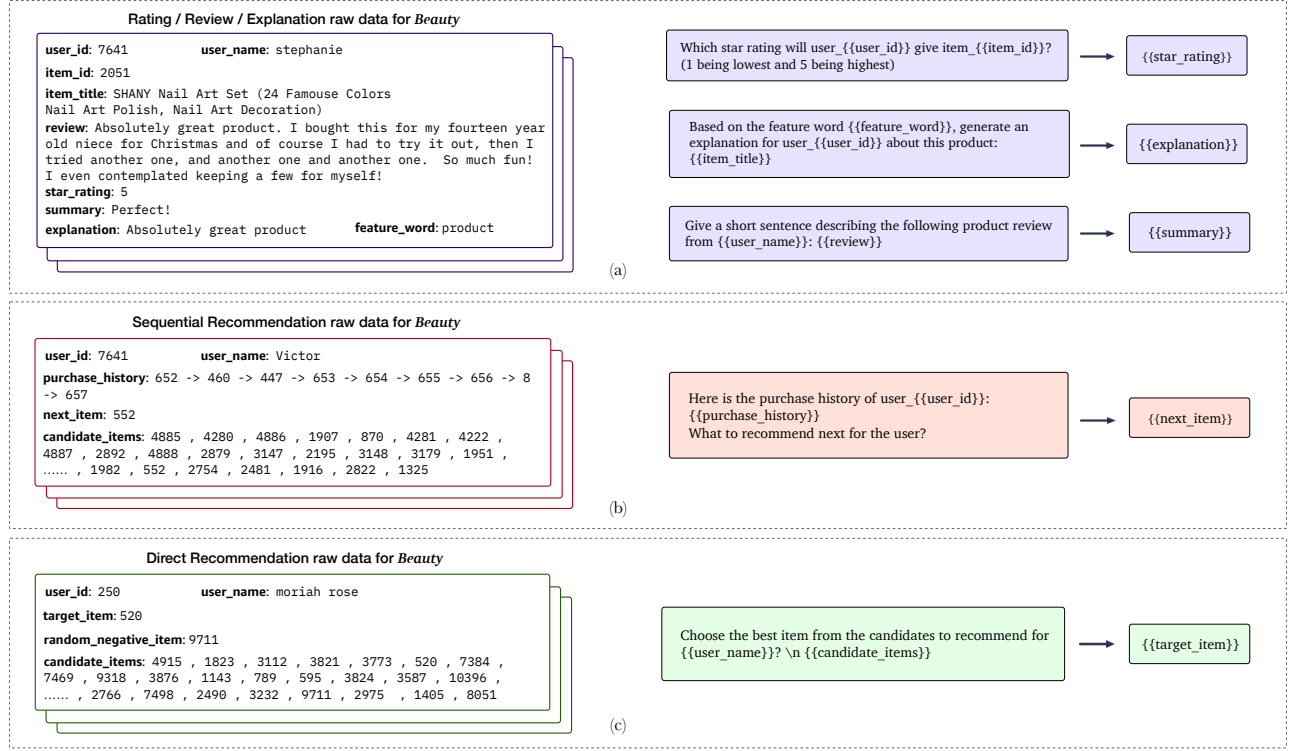
Figure 2: Building input–target pairs from raw data according to our designed personalized prompt templates – simply substituting the fields in the prompts with the corresponding information in raw data. The raw data for the five task families of P5 are from three separate sources. Specifically, rating/review/explanation prompts (a) have shared raw data. Sequential recommendation (b) and direct recommendation (c) uses similar raw data, but the former particularly requires the user interaction history. The complete collection of P5 personalized prompts are provided in the Appendix.

Prompt Learning. The success of GPT series especially GPT-3 [2] marked the beginning of prompt's popularization on NLP tasks. Trained with huge language data from the Web, GPT-3 exhibited the capability of solving NLP tasks when provided a number of input-output examples as exemplar prompts. Besides exemplar prompts, many prompt design methods have proliferated following the "pre-train, prompt, and predict" paradigm [37]. One type of the methods [16, 23, 36, 40, 58] explored prompt search for proper discrete prompts. Meanwhile, another line of work [18, 28, 33, 38, 45, 81] exploited continuous vector embeddings as prompts. Compared with the aforementioned prompt types, instruction-based prompts contain detailed task descriptions and adhere more to the natural language format. Since instruction-based prompts are flexible and close to how humans communicate with each other, several pioneer works [11, 68] claim that learning from crowd-sourced NLP datasets is a promising route for general purpose NLP systems. Recent works such as FLAN [67] and T0 [51] finetuned pretrained language models on large-scale NLP datasets verbalized via human-readable prompts. As a result, such multitask prompt-based tuning brings powerful models that exhibit strong zero-shot ability on unseen tasks. Inspired by the success of these approaches, we create a collection of personalized prompts and then train a sequence-to-sequence model on a variety of recommendation related tasks verbalized according to the constructed personalized prompts.

NLP for Recommendation. Recommendation has been interacting with NLP techniques for a long time. The main work mostly address four lines of research: 1) explainable recommendation [4, 10, 30–32, 75, 77] where NLP models help generating text explanations for a given recommendation; 2) sequential recommendation as language modeling [9, 60, 80] which considers user interaction histories as word token sequences; 3) text feature extraction [69, 74, 79] which aims to extract informative text encodings that can improve the performance of recommendation; and 4) conversational recommendation [8, 12–14, 22, 61, 76] that reasons the intent of users and gives recommendation in an interactive dialog format. In our work, we explicitly covers the tasks of sequential recommendation and explanation generation, and additionally offers insights on how to formulate a unified NLP framework for other recommendation problems including rating prediction, top-k recommendation, and review summarization. Furthermore, pretrained with instruction-based prompts that share similarity with conversational recommendation, our P5 benefits from the natural language environment and improves the performance on a series of recommendation tasks.

Zero-shot and Cold Start Recommendation. Recommender systems' performances heavily rely on the available training data, but there are always zero-shot cases where the history records are limited. The evidences of performing well on such startup cases signal a good generalization ability of recommendation models.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, Yongfeng Zhang
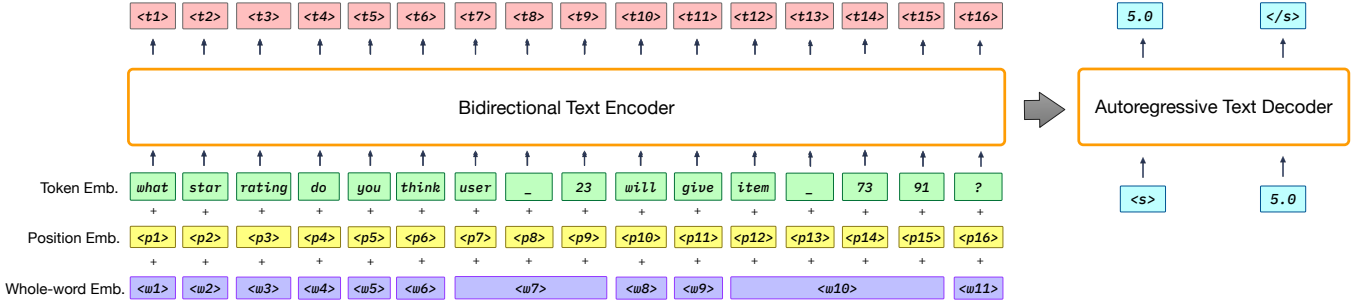
Figure 3: An illustration of the P5 architecture. For the example prompt input "What star rating do you think user_23 will give item_7391?", P5 adopts an encoder–decoder framework: first encodes the input with a bidirectional text encoder, and then generates the answer through a text decoder autoregressively. In contrast to task-specific recommendation models, our P5 relies on multitask prompt-based pretraining on a large-scale personalized prompt collection, which makes P5 able to adapt to different task families and even generalize to novel personalized prompts.

One widely studied problem under this setting is the cold-start recommendation where users [26] or items [53] are new to the system with no previous interaction records. Solutions to this problem either learn to model content features [15, 29, 44, 55] so that inference can be made without interaction records or learn to transfer representations from auxiliary domains [42, 56, 59, 72, 82]. Another line of work for zero-shot or few-shot recommendation discusses the quick adaptation to the new domain instead of providing recommendation for cold-start cases only. Solutions typically follow the meta learning [27, 64] or causal learning [34] frameworks that make the model robust to domain adaptations. In our work, we ask P5 model pretrained on an auxiliary domain to solve tasks on target domains, where the users are known to P5 but the items have never been seen by the model before.

3 PERSONALIZED PROMPT COLLECTION

To facilitate the multitask prompt-based pretraining for recommendation, we create a collection of personalized prompt templates. The collection covers five different task families – **rating**, **sequential recommendation**, **explanation**, **review**, and **direct recommendation**. Each of these task families contains multiple personalized prompts to help P5 discover various aspects about users and items. As mentioned in [51], a prompt is considered as consisting of an input template and a target template, along with a collection of associated metadata. In this work, we further define a *personalized prompt* as a prompt that includes personalized fields for different users and items. For example, a user's preference can be indicated through either an ID number or a description of the user such as name, gender, age, etc. Moreover, the expected model output of a given personalized prompt should also vary according to its item field. This implies the change of user's preferences towards different items. Such item fields can be represented by either item ID numbers or item metadata that contains detailed descriptions.

We designed basic P5 personalized prompt collection for each task family. For **rating** prediction task family, we divide the prompts into three categories: 1) Given the information about a user and an item, directly predict the rating score ranging from 1 to 5; 2) Predict whether a user will rate an item a given score. The expected output is yes or no; 3) Predict if a user likes or dislikes an item. Here we consider a star rating equal to or greater than 4 to be a

like preference of the user, whereas lower scores indicate a *dislike* preference. For **sequential** recommendation task family, we create three types of prompts: 1) Directly predict the next item based on user interaction history; 2) Given user interaction history, choose the possible next item from a candidate list, where only one item is positive; 3) Based on user interaction history, predict whether a given item will be interacted next by the user. For **explanation** task family, we ask P5 model to generate a textual explanation to justify a user's preference towards a given item. There are two prompt categories in this task family: 1) Directly generate an explanation sentence with user/item information; 2) Generate explanation based on a feature word as hint [31]. For each category, there could be other auxiliary information included such as the review headline and the star rating. For **review** related task family, we create two types of prompts: 1) Summarize review comment to a shorter review title; 2) Predict the corresponding rating score based on the given review comment. For **direct** recommendation, we also create two types of prompts: 1) Predict whether to recommend an item to a user, the answer should be yes or no; 2) Select the most suitable item from a list of candidate items to recommend to the user. We provide some example prompts in Figure 2, and the complete collection of personalized prompts are provided in the Appendix.

With the prompts, we can directly build input–target pairs from raw data. As illustrated in Figure 2, we can simply substitute the fields in braces with the corresponding information in the raw data and thus create training input–target pairs or zero-shot testing personalized prompts. The training data and pre-training tasks will distill the rich semantics from diverse modalities into the user and item tokens for preference understanding and personalization. Note that we divide the raw data into three parts—rating/review/explanation share the same raw data, while sequential and direct recommendation differ in terms of whether to use interaction history as input information. During pretraining, we mix the input–target pairs from different task families together to serve as the training data. To enhance P5's robustness and zero-shot generalization, for each raw datum, we only sample a portion of rather than all of the personalized prompts in each task family. In sequential and direct recommendation task families, we also randomly select a group of negative items for those prompts that require a candidate list.

4 THE P5 PARADIGM AND MODEL

4.1 The P5 Architecture

The collection of personalized prompts introduced in the previous section makes it convenient to create a large amount of available pretraining data that covers a wide range of recommendation related tasks. Thanks to the prompt templates, all pretraining data shares a unified format of input–target token sequences, which breaks the boundaries among different tasks. We claim that pretraining multiple recommendation tasks under a unified framework of conditional generation can facilitate all involving tasks together. By immersing P5 in the full language environment throughout the pretraining stage, we also expect its zero-shot generalization capability of understanding unseen personalized prompts with detailed item descriptions. That is the reason why P5 is called a unified "Pretrain, Personalized Prompt, and Predict Paradigm".

In terms of the model architecture, our P5 is established upon a basic encoder–decoder framework. We employ Transformer [65] blocks to build both the encoder and decoder. Suppose the embeddings of an input token sequence is $\mathbf{x} = [x_1, \cdots, x_n]$. As depicted in Figure 3, before feeding the embedding sequence into the bidirectional text encoder $\mathcal{E}(\cdot)$, we add positional encodings $\mathcal{P}$ to the raw embeddings to capture their position information in the sequence. Furthermore, to make P5 aware of the personalized information contained in the input sequence, we also apply whole-word embeddings $\mathcal{W}$ to indicate whether consecutive sub-word tokens are from the same original word. For instance, if we directly represent the item with ID number 7391 as "item_7391", then the word will be split into 4 separate tokens (i.e., "item", "_", "73", "91") by Sentence-Piece tokenizer [54]. With the assistance of the shared whole-word embedding "⟨w10⟩" (e.g., in Figure 3), P5 can better recognize the important field with personalized information. Another alternative is to represent each user/item by an independent extra token (e.g., "⟨item_7391⟩"). However, this may incur huge amounts of additional tokens when there is a large pool of users and items. Hence, in this paper, we adopt multiple sub-word units to represent a user or item.

Afterwards, the text encoder takes the sum of the aforementioned three embeddings $\mathbf{e} = [e_1, \cdots, e_n]$ and outputs their contextualized representations $\mathbf{t} = [t_1, \cdots, t_n] = \mathcal{E}(\mathbf{e})$. The decoder $\mathcal{D}(\cdot)$ then attends to both the previously generated tokens $\mathbf{y}_{<j}$ and the encoder output $\mathbf{t}$ and predicts the probability distribution of future tokens: $P_\theta(\mathbf{y}_j \mid \mathbf{y}_{<j}, \mathbf{x}) = \mathcal{D}(\mathbf{y}_{<j}, \mathbf{t})$. During the pretraining stage, P5 learns the model parameters θ by minimizing the negative log-likelihood of label tokens $\mathbf{y}$ conditioned on input text $\mathbf{x}$ in an end-to-end manner:

$$\mathcal{L}_\theta^{\text{P5}} = -\sum_{j=1}^{|\mathbf{y}|} \log P_\theta(\mathbf{y}_j \mid \mathbf{y}_{<j}, \mathbf{x}) \qquad (1)$$

This same objective function is shared by all recommendation tasks under P5. As a result, we unify recommendation tasks with one model, one loss, and one data format.

4.2 Recommendation with Pretrained P5

After pretraining, P5 can directly perform different tasks with either seen or unseen personalized prompts. For rating, explanation, and review tasks, we simply use greedy decoding to generate answers. In contrast, sequential and direct recommendation tasks usually

Table 1: Basic statistics of the experimental datasets.

Dataset	Sports	Beauty	Toys	Yelp
#Users	35,598	22,363	19,412	30,431
#Items	18,357	12,101	11,924	20,033
#Reviews	296,337	198,502	167,597	316,354
#Sparsity (%)	0.0453	0.0734	0.0724	0.0519

require an item list as target output. In view of this, for sequential recommendation, we apply beam search to generate a list of potential next items and evaluate it under the all-item setting. For direct recommendation, we predict the recommended items from a candidate set $\mathbf{S} = \{S_1, \cdots, S_m\}$, where only one of the m candidates is positive. Here, we also use beam search to decode a list of potential target items with the highest scores and then conduct evaluations. Both of the above decoding processes can be written as:

$$\mathbf{C} = [C_1, \cdots, C_B] = \text{Beam\_Search}(\mathcal{D}, \mathbf{t}, B) \qquad (2)$$

where B denotes the beam size and $\mathbf{C}$ is the output item list.

5 EXPERIMENTS

In this section, we evaluate the performance of the proposed P5 approach on real-world data and compare it with various representative methods targeting at different task families. Through the performance comparison and ablation studies, we aim to answer the following research questions regarding our unified "Pretrain, Personalized Prompt, and Predict Pargadigm" (P5):

- **RQ1:** How does our unified P5 framework perform compared with task-specific methods on all five task families?
- **RQ2:** Does P5 have enough zero-shot generalization ability when transferring to unseen personalized prompts for either existing or new items?
- **RQ3:** How do scaling factors such as model size, number of task families, and number of prompts affect the performance of P5?
- **RQ4:** Which is a better way to implement personalization in P5: adopting an independent extra token for each user or item (e.g., "⟨user_23⟩") or the default setting, i.e., tokenizing each user or item into multiple sub-word units (e.g., "user", "_", "23")?
- **RQ5:** How long does it take for P5 to conduct pretraining? Is it efficient to make inference with the pretrained P5 model? We provide statistics on training and inference time in the Appendix.

5.1 Experimental Setup

Datasets. We conduct extensive experiments over *four* real-world datasets. The Amazon[1] datasets are collected from *Amazon.com* platform with user ratings and reviews on 29 categories of products. In this paper, we adopt three of them to evaluate our method, namely *Sports & Outdoors*, *Beauty*, as well as *Toys & Games*. Besides, Yelp[2] dataset contains a large number of user ratings and reviews for business recommendation. We follow [80] and use transaction records between January 1, 2019 to December 31, 2019. Due to space limit and that the results on Yelp show similar trends with other datasets, we put the experimental results on *Yelp* dataset in the Appendix. The detailed statistics of these datasets are presented in Table 1.

---

[1]https://nijianmo.github.io/amazon/
[2]https://www.yelp.com/dataset

Task splits. For rating, explanation, and review task families, we randomly split each dataset into training (80%), validation (10%) and testing (10%) sets, and ensure that there is at least one instance included in the training set for each user and item. To obtain the ground-truth explanations, following the natural language explanation works [30, 31], we first extract item feature words from the reviews with the help of the Sentires toolkit[3] [77, 78], and then extract the sentences from reviews that comment on one or more item feature words as users' explanation about their preference. In terms of sequential recommendation task family, for each user interaction sequence, the last item is used as the test data, the item before the last one is used as the validation data, and the remaining data is used for training. To avoid data leakage during pretraining, we follow the training split of sequential recommendation to build the training set for direct recommendation task family.

Implementation Details. Our P5 model utilizes the pretrained T5 checkpoints [47] as backbone. According to the size of T5 backbone, we create two versions of P5, namely P5-small (**P5-S**) and P5-base (**P5-B**). For P5-small, there are 6 layers for both encoder and decoder, the model dimensionality is 512 with 8-headed attention. For P5-base, encoder and decoder both have 12 Transformer blocks. The model has an embedding dimensionality of 768 and a 12-headed attention. For tokenization, we directly use the SentencePiece [54] tokenizer with a vocabulary size of 32,128 for parsing sub-word units. We pretrain P5 for 10 epochs with AdamW optimization [39] on four NVIDIA RTX A5000 GPUs. The batch size is set to 16 for P5-base and 32 for P5-small. We choose 1×10^{-3} as the peak learning rate and set the maximum length of input tokens to 512. The warmup strategy is used to adjust the learning rate during training, the warmup stage is set to be the first 5% of all iterations. When negative sampling is needed for training, we use 1:1 positive vs. negative sampling for both P5 and baselines.

Our default *pretrain–predict combination* adopts the **last** prompt in each task family for zero-shot evaluation while all remaining prompts are utilized for multitask prompted pretraining. For rating prediction, we use Gaussian sampling to convert the original integer scores to float numbers rounded to 1 decimal place. In this way, we can avoid overfitting the limited score types. After this change, we increase the number of score classes from 5 to 41. For sequential recommendation, we set the beam size B to 20. For direct recommendation, the beam size is also 20 and the candidate pool contains 100 items, which consist of one ground-truth item and 99 sampled negative ones that the user has not interacted with.

Metrics. For rating prediction, we adopt two standard metrics— Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). For sequential recommendation and direct recommendation task families, we employ top-k Hit Ratio (HR@k) and Normalized Discounted Cumulative Gain (NDCG@k) to evaluate the performance and report HR@1, 5, 10 and NGCG@5, 10. For explanation generation and review summarization, we evaluate different methods with BLEU-4, as well as ROUGE-1, ROUGE-2, and ROUGE-L. RMSE and MAE are "the lower, the better", while all other metrics are "the higher, the better". For all tables in the following, **bold** numbers refer to the best performance, while underlined numbers indicate the second best performance.

---

[3]https://github.com/evison/Sentires

Table 2: Performance comparison on rating prediction.

Methods	Sports		Beauty		Toys	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
MF	**1.0234**	0.7935	**1.1973**	0.9461	**1.0123**	0.7984
MLP	1.1277	0.7626	1.3078	0.9597	1.1215	0.8097
P5-S (1-6)	1.0594	**0.6639**	1.3128	**0.8428**	1.0746	<u>0.7054</u>
P5-B (1-6)	1.0357	0.6813	<u>1.2843</u>	0.8534	1.0544	0.7177
P5-S (1-10)	1.0522	<u>0.6698</u>	1.2989	<u>0.8473</u>	1.0550	0.7173
P5-B (1-10)	<u>1.0292</u>	0.6864	1.2870	0.8531	<u>1.0245</u>	**0.6931**

5.2 Baselines for Multiple Tasks

To demonstrate P5's competence on a wide range of recommendation related tasks, we gather a collection of representative approaches for difference task families.

Rating Prediction and Direct Recommendation. These tasks take the user–item rating/interaction data, but no content or side information is provided. We aim to justify whether the models are able to provide accurate rating prediction or recommendation lists that align with the user preferences. We use **MF** [25] and **MLP** [5] under mean square root loss as rating prediction baselines. For direct recommendation, we use **BPR-MF** [49], **BPR-MLP** [5], and a state-of-the-art contrastive learning-based collaborative filtering model **SimpleX** [43] as baselines.

Sequential Recommendation. We adopt several representative sequential recommendation approaches as our baselines. **Caser** [63] treats sequential recommendation as a Markov Chain and employs convolutional neural networks to model user interests. **HGN** [41] adopts a hierarchical gating networks to learn user behaviors from the perspectives of both long and short terms. **GRU4Rec** [21] is originally proposed for session-based recommendation. It utilizes GRU [7] to model the user click history sequence. **BERT4Rec** [60] mimics the BERT-style masked language modeling and learns a bidirectional representation for sequential recommendation. **FDSA** [73] focuses on the feature transition patterns by modeling feature sequence with a self-attention module. **SASRec** [24] adopts self-attention mechanism in a sequential recommendation model, which reconciles the properties of Markov Chains and RNN-based approaches. **S³-Rec** [80] leverages self-supervised objectives to help sequential recommendation model better discover the correlations among different items and their attributes. We use the implementation of S³-Rec and its baselines for comparison[4].

Explanation Generation. For performance comparison, we consider several baselines with regard to the task of explanation generation. **Attn2Seq** [10] learns to encode attributes into vectors, and then invokes an attention mechanism to generate reviews conditioned on the attribute vector. **NRT** [32] utilizes GRU [7] to generate explanations based on user and item IDs. **PETER** [31] is a simple and effective framework that attempts to utilize user and item IDs to generate explanations. It is built upon a modified attention mask of the Transformer architecture. There is also a variant **PETER+**, which takes a hint feature word to assist the explanation generation.

Review Related. For review summarization, we adopt pretrained **T0** [51] and **GPT-2** [46] checkpoints hosted by Hugging Face[5] as baselines. For review preference prediction, we only use **T0** to make comparisons because GPT-2 cannot perform this task.

---

[4]https://github.com/RUCAIBox/CIKM2020-S3Rec
[5]https://huggingface.co/

Table 3: Performance comparison on sequential recommendation.

Methods	Sports				Beauty				Toys			
	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10	HR@5	NDCG@5	HR@10	NDCG@10
Caser	0.0116	0.0072	0.0194	0.0097	0.0205	0.0131	0.0347	0.0176	0.0166	0.0107	0.0270	0.0141
HGN	0.0189	0.0120	0.0313	0.0159	0.0325	0.0206	0.0512	0.0266	0.0321	0.0221	0.0497	0.0277
GRU4Rec	0.0129	0.0086	0.0204	0.0110	0.0164	0.0099	0.0283	0.0137	0.0097	0.0059	0.0176	0.0084
BERT4Rec	0.0115	0.0075	0.0191	0.0099	0.0203	0.0124	0.0347	0.0170	0.0116	0.0071	0.0203	0.0099
FDSA	0.0182	0.0122	0.0288	0.0156	0.0267	0.0163	0.0407	0.0208	0.0228	0.0140	0.0381	0.0189
SASRec	0.0233	0.0154	0.0350	0.0192	0.0387	0.0249	0.0605	0.0318	0.0463	0.0306	0.0675	0.0374
S^3-Rec	0.0251	0.0161	0.0385	0.0204	0.0387	0.0244	0.0647	0.0327	0.0443	0.0294	0.0700	0.0376
P5-S (2-3)	0.0272	0.0169	0.0361	0.0198	0.0503	0.0370	0.0659	0.0421	**0.0648**	**0.0567**	**0.0709**	**0.0587**
P5-B (2-3)	0.0364	0.0296	0.0431	0.0318	**0.0508**	**0.0379**	**0.0664**	**0.0429**	0.0608	0.0507	0.0688	0.0534
P5-S (2-13)	0.0258	0.0159	0.0346	0.0188	0.0490	0.0358	0.0646	0.0409	0.0647	0.0566	0.0705	0.0585
P5-B (2-13)	**0.0387**	**0.0312**	**0.0460**	**0.0336**	0.0493	0.0367	0.0645	0.0416	0.0587	0.0486	0.0675	0.0536

Table 4: Performance comparison on explanation generation.

Methods	Sports				Beauty				Toys			
	BLUE4	ROUGE1	ROUGE2	ROUGEL	BLUE4	ROUGE1	ROUGE2	ROUGEL	BLUE4	ROUGE1	ROUGE2	ROUGEL
Attn2Seq	0.5305	12.2800	1.2107	9.1312	0.7889	12.6590	1.6820	9.7481	1.6238	13.2245	2.9942	10.7398
NRT	0.4793	11.0723	1.1304	7.6674	0.8295	12.7815	1.8543	9.9477	1.9084	13.5231	3.6708	11.1867
PETER	**0.7112**	12.8944	1.3283	9.8635	1.1541	14.8497	2.1413	11.4143	**1.9861**	14.2716	3.6718	11.7010
P5-S (3-3)	0.5902	**60.8892**	17.7514	18.0010	2.4094	**62.0077**	21.4176	25.1863	1.0794	59.4588	18.9272	19.8567
P5-B (3-3)	0.6213	58.7260	18.5533	18.4670	2.6606	61.7001	22.3824	25.9821	1.5248	60.0415	19.9905	20.8306
PETER+	2.4627	24.1181	5.1937	18.4105	3.2606	25.5541	5.9668	19.7168	4.7919	28.3083	9.4520	22.7017
P5-S (3-9)	**7.2129**	**67.4004**	**36.1417**	**30.8359**	5.8239	**68.6839**	37.1359	31.4482	**8.5465**	**69.7741**	40.0564	**34.0364**
P5-B (3-9)	3.5598	64.7683	34.0162	26.3184	6.5846	68.3067	37.0722	31.7545	7.7315	69.2266	39.7407	33.3348
P5-S (3-12)	5.8446	66.5976	35.5160	29.2766	5.3233	68.3174	36.8730	30.7071	8.2840	69.7158	40.1340	33.9770
P5-B (3-12)	4.6977	65.4562	34.9379	27.7223	**6.9169**	68.2195	36.9858	**32.1314**	7.2597	68.8920	39.5056	32.8794

5.3 Performance Comparison on Different Task Families (RQ1)

In this section, we pretrain P5 with prompts from all five task families to verify its multitask learning ability. According to the default pretrain–predict task combination, we leave Prompt 1-10, Prompt 2-13, Prompt 3-12, Prompt 4-4, and Prompt 5-8 for zero-shot evaluation and pretrain P5 with the remaining personalized prompts. The performances of P5 and relevant baselines on the five task families are presented in Table 2 to Table 7. For each task family, we choose one or more **seen** prompts as supplement to the aforementioned zero-shot **unseen** prompts to perform evaluations.

5.3.1 *Rating Prediction*.
Prompt 1-6 and Prompt 1-10 are used for evaluating P5's performance on rating prediction. The performance comparison is presented in Table 2. We can see that when testing with seen Prompt 1-6, P5-B gets better MAE and slightly higher RMSE on all three datasets compared with MF. When testing with unseen Prompt 1-10, P5-B can achieve similar performance as Prompt 1-6. Moreover, P5-S usually has better MAE but higher RMSE. It seems that P5 is overfitting these data since the task complexity of rating prediction is relatively lower than other recommendation tasks. Overall, these results show that it is feasible to perform rating prediction on a conditional text generation framework.

5.3.2 *Sequential Recommendation*.
As illustrated in Table 3, Prompt 2-3 and Prompt 2-13 are employed for the evaluation of sequential recommendation under all-item setting, i.e., using all items as candidates rather than sampling 100 or 1,000 items for

Table 5: Performance on review preference prediction.

Methods	Sports		Beauty		Toys	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
T0 (4-2)	0.6728	0.3140	0.6925	0.3324	0.8282	0.4201
T0 (4-4)	0.6503	0.2984	0.7066	0.3663	0.8148	0.4230
P5-S (4-2)	0.7293	0.3529	**0.6233**	**0.3051**	**0.6464**	**0.3125**
P5-B (4-2)	**0.6487**	**0.2847**	0.6449	0.3168	0.6785	0.3342
P5-S (4-4)	0.7565	0.3395	0.6262	0.3113	0.6577	0.3174
P5-B (4-4)	0.6563	0.2921	0.6515	0.3106	0.6730	0.3342

ranking. From the table, we can see that P5-B surpasses all competitive baselines with a relatively large gap on both seen (Prompt 2-3) and unseen (Prompt 2-13) prompts. On *Toys*, P5-S can get even better performance than P5-B. While on *Beauty* and *Sports*, P5-B achieves the advantage over P5-S. The results show that the P5 architecture is effective in modeling the user interaction history and conducting next item prediction with the help of beam search.

5.3.3 *Explanation Generation*.
In Table 4, Prompt 3-9 and Prompt 3-12 are used to evaluate P5's performance on explanation generation under feature-based setup, while Prompt 3-3 is used for direct explanation generation without providing a hint word. We can see that for both seen (Prompts 3-3 & 3-9) and unseen (Prompt 3-12) prompts, P5 achieves both the best and the second best performance on most cases. For feature-based prompts (Prompts 3-9 & 3-12), P5-B outperforms PETER+ by a large margin on all domains.

5.3.4 *Review Related*.
We take Prompts 4-2 and 4-4 to compare P5's performance with T0 on review preference prediction, as shown in Table 5. We can see that P5-S achieves better RMSE and

Table 6: Performance comparison on review summarization.

Methods	Sports				Beauty				Toys			
	BLUE4	ROUGE1	ROUGE2	ROUGEL	BLUE4	ROUGE1	ROUGE2	ROUGEL	BLUE4	ROUGE1	ROUGE2	ROUGEL
T0 (4-1)	**8.0881**	43.4226	13.9629	19.4267	6.1775	41.4487	13.2033	18.2904	_7.1445_	_44.8651_	15.1572	18.9544
GPT-2 (4-1)	2.6586	43.6313	15.2021	5.2144	2.2032	41.0148	13.5772	4.2457	2.0760	43.0985	14.4126	4.4048
P5-S (4-1)	7.1781	_44.6943_	_20.0491_	_26.5941_	**7.5644**	**44.0703**	**18.3076**	**26.4375**	7.3916	**46.0153**	**19.3441**	**27.1528**
P5-B (4-1)	_7.5727_	**47.7859**	**21.6872**	**28.1312**	_7.0708_	_44.0486_	_18.1725_	_26.2031_	6.0264	43.7403	_17.3446_	_25.0362_

Table 7: Performance comparison on direct recommendation.

Methods	Sports					Beauty					Toys				
	HR@1	HR@5	NDCG@5	HR@10	NDCG@10	HR@1	HR@5	NDCG@5	HR@10	NDCG@10	HR@1	HR@5	NDCG@5	HR@10	NDCG@10
BPR-MF	0.0314	0.1404	0.0848	0.2563	0.1220	0.0311	0.1426	0.0857	0.2573	0.1224	0.0233	0.1066	0.0641	0.2003	0.0940
BPR-MLP	0.0351	0.1520	0.0927	0.2671	0.1296	0.0317	0.1392	0.0848	0.2542	0.1215	0.0252	0.1142	0.0688	0.2077	0.0988
SimpleX	0.0331	_0.2362_	_0.1505_	0.3290	0.1800	0.0325	0.2247	0.1441	0.3090	0.1711	0.0268	**0.1958**	**0.1244**	_0.2662_	_0.1469_
P5-S (5-1)	0.0638	0.2096	0.1375	0.3143	0.1711	0.0600	0.2021	0.1316	0.3121	0.1670	0.0405	0.1538	0.0969	0.2405	0.1248
P5-B (5-1)	0.0245	0.0816	0.0529	0.1384	0.0711	0.0224	0.0904	0.0559	0.1593	0.0780	0.0187	0.0827	0.0500	0.1543	0.0729
P5-S (5-4)	0.0701	0.2241	0.1483	_0.3313_	_0.1827_	_0.0862_	_0.2448_	_0.1673_	_0.3441_	_0.1993_	0.0413	0.1411	0.0916	0.2227	0.1178
P5-B (5-4)	0.0299	0.1026	0.0665	0.1708	0.0883	0.0506	0.1557	0.1033	0.2350	0.1287	0.0435	0.1316	0.0882	0.2000	0.1102
P5-S (5-5)	_0.0766_	0.2195	0.1495	0.3187	0.1814	0.0826	0.2378	0.1613	0.3289	0.1906	_0.0453_	0.1404	0.0933	0.2192	0.1185
P5-B (5-5)	0.0618	0.1812	0.1226	0.2669	0.1501	0.0601	0.1607	0.1113	0.2308	0.1339	0.0438	0.1367	0.0910	0.2106	0.1147
P5-S (5-8)	**0.0967**	**0.2519**	**0.1763**	**0.3530**	**0.2089**	**0.1020**	**0.2688**	**0.1869**	**0.3665**	**0.2184**	**0.0588**	_0.1793_	_0.1203_	**0.2691**	**0.1491**
P5-B (5-8)	0.0631	0.1834	0.1245	0.2643	0.1505	0.0576	0.1572	0.1086	0.2306	0.1321	0.0442	0.1374	0.0914	0.2160	0.1167

MAE on *Beauty* and *Toys*, while P5-B shows better performance on *Sports*. Additionally, we take Prompt 4-1 to evaluate P5's ability on review summarization, as shown in Table 6. For this task, P5-S clearly outperforms T0 and GPT-2 except for BLUE4 score on *Sports*. It is worth noting that GPT-2 and T0 has 1.5B and 11B parameters, respectively. This shows that P5 can achieve better performance than these competitive baselines with a much smaller model size.

5.3.5 Direct Recommendation. Finally, Prompts 5-1, 5-4, 5-5 and 5-8 are applied to evaluate the direct recommendation task under the 1-out-of-100 evaluation setting. For binary question prompts (5-1 & 5-4), which are discriminative prompts, we use the softmax generation probability of "yes" to rank the candidate items. For open question prompts (5-5 & 5-8), which are generative prompts, we use beam-search (Eq.(2)) to generate the top-k list. The results are presented in Table 7. From the table, we can see that P5-B and P5-S have great advantages over BPR-MF and BPR-MLP on all three datasets. Comparing with SimpleX, we can see that P5 works especially well on top-1 item ranking, which is more than two times better than SimpleX on HR@1. Besides, P5 also achieves the best result on most of the other metrics. The success of P5 on direct recommendation shows the competence of the sequence-to-sequence generation framework in recommendation domain.

5.4 Zero-shot Generalization to Unseen Prompts and Items in New Domain (RQ2)

5.4.1 Transfer to Unseen Personalized Prompts. In this section, we transfer the pretrained P5 models to the previously held-out prompts during pretraining. These unseen prompts are from the same task families, and the testing items have been seen by P5 during pretraining at least once. The experimental results are also reported in Table 2 to Table 7. As previously discussed in Section 5.3, P5 achieves surprisingly good performances on various task families when being challenged by unseen prompts. On some specific

Table 8: Statistics on domain transfer evaluation sets.

Dataset	Sports	Beauty	Toys
#Users	290	439	487
#Items	381	586	886
#Reviews	478	1,237	1,183

datasets, the performances of P5 on unseen prompts even surpass seen prompts, e.g., P5-B gets the best performances under Prompt 2-13 on *Sports* and under Prompt 3-12 on *Beauty*. These results show that multitask prompted pretraining empowers P5 enough robustness to understand unseen prompts with wording variations.

5.4.2 Transfer to Items in New Domain. Next, we increase the difficulty level of zero-shot transfer. We collect a group of 741 users that exist in all the three domains with their interaction and review histories in other domains. The detailed statistics of these domain transfer evaluation sets are illustrated in Table 8. We then challenge P5-B pretrained on one domain with unseen prompts from the Task Family Z, whose item fields are filled with the information from a new product domain. For example, we ask the P5 model pretrained on the *Toys* domain about an existing user's preference towards an item in the *Beauty* domain. The full results on all six directions are reported in Table 9. From the table, we notice P5 still maintains sufficient performances for rating prediction (Prompts Z-2 & Z-3), like/dislike prediction (Prompts Z-1 & Z-4), as well as explanation generation with feature word (Prompt Z-6). In contrast, direct explanation generation without feature word (Prompts Z-5 & Z-7) is very difficult for P5 because it lacks awareness of relevant knowledge in the new domain. In Figure 4, we provide some example explanations generated by P5-B under the setup of zero-shot domain transfer (Prompt Z-6). We can see that P5 is able to catch different users' rating preferences and hint feature words, then integrate them with the knowledge learned from previous domain to generate plausible explanations.

Toys -> Beauty

Case 1 **Input**: Based on the word color , help user_4549 write a 5-star explanation for this new product : \n title : Bain De Terre Serum Anti-Frizz Recovery Complex 1.7 oz. \n price : 3.86 \n brand : Bain de Terre
Target Output: I have color treated fine hair but lots of it
P5's Output: The colors are bright

Case 2 **Input**: Based on the word shampoo , help user_1491 write a 1-star explanation for this new product : \n title : CLEAR SCALP & HAIR BEAUTY Volumizing Root Boost Nourishing Shampoo, 12.9 Fluid Ounce \n price : 1.66 \n brand : Clear
Target Output: I tried the shampoo and it hardly foams up
P5's Output: Not worth the price

Beauty -> Sports

Case 3 **Input**: Based on the word tool , help user_3877 write a 5-star explanation for this new product : \n title : SOG Specialty Knives & Tools TF6-CP Trident Knife with Straight Edge Assisted Folding 3.75-Inch Steel Tanto Shape Blade and GRN Handle, Satin Finish \n price : 57.6 \n brand : SOG Specialty Knives
Target Output: this is an excellent tool
P5's Output: This is a great tool tool

Case 4 **Input**: Based on the word saw , help user_326 write a 2-star explanation for this new product : \n title : Rome's #1705 Square Pie Iron with Steel and Wood Handles \n price : 20.26 \n brand : Rome Industries
Target Output: After looking closely we saw 2 of the rivots that are supposed to hold it on were missing
P5's Output: I saw no visible improvement

Sports -> Toys

Case 5 **Input**: Based on the word game , help user_29782 write a 3-star explanation for this new product : \n title : Deal Breaker Coaster Card Game \n price : 13.95 \n brand : Zobmondo!!
Target Output: I think this game could prove interesting if played while drinking
P5's Output: I'm a big fan of this game and it is a great game

Case 6 **Input**: Based on the word cars , help user_2121 write a 5-star explanation for this new product : \n title : Ridemakerz Marvel Universe Spiderman and Venom Xtreme Customz Swap Set \n price : 14.89 \n brand : RIDEMAKERZ
Target Output: These are great looking little plastic cars
P5's Output: I have several cars and this one is the best

Figure 4: Example cases of zero-shot domain transfer on Z-6 task. We demonstrate three transfer directions: *Toys* to *Beauty*, *Beauty* to *Sports*, and *Sports* to *Toys*.
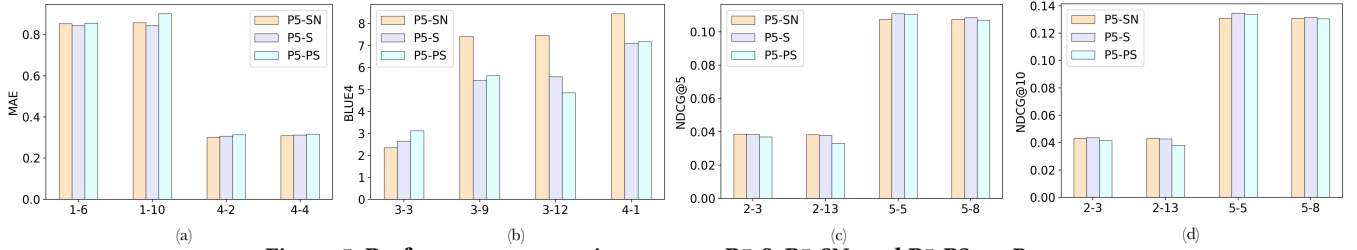

Figure 5: Performance comparison among P5-S, P5-SN, and P5-PS on *Beauty*.

5.5 Ablation on Model Size (RQ3)

In this section, we will discuss the influence of model size on the performance of P5 on different recommendation tasks. Here, we train two size variants of P5, namely P5-small and P5-base. The parameter numbers of these two P5 models are 60.75M and 223.28M, respectively. From Table 2 to Table 7, we can see that although P5-S is only 1/4 of the size of P5-B, P5-S can beats P5-B on a series of tasks and datasets. For example, P5-S achieves better sequential recommendation, review preference prediction, and direct recommendation (Prompts 5-5 & 5-8) performances than P5-B on *Toys*. In contrast, P5-B shows advantages on sequential recommendation and review preference prediction tasks for *Sports*. Since *Sports* contains more users, items and reviews and has a lower sparsity, it requires a model with higher capacity to discover latent correlation among different personalized factors. The findings indicate that larger P5 models may be needed when the dataset is large, while for smaller datasets, smaller P5 models could be enough. As a result, we should decide an appropriate model size that matches the scale of the training data.

5.6 Ablation on Task Scaling (RQ3)

Moreover, we explore whether multitask prompted pretraining is superior than pretraining on each task family alone. We pretrain P5-small on *Beauty* dataset with prompts from every single task family, resulting in five models – P5-S1, P5-S2, P5-S3, P5-S4, and P5-S5. We then compare P5-S on various recommendation tasks with the corresponding single task P5 model. The performance comparison between P5-S and **P5-SN** ($N \in [1, 2, 3, 4, 5]$) is illustrated in Figure 5. As shown in the figure, P5-S achieves comparable or better performance than P5-SN on rating prediction, sequential recommendation and direct recommendation tasks, while on text generation tasks such as explanation generation (Prompts 3-9 & 3-12) and review summarization (Prompt 4-1), P5-SN is better than P5-S. This indicates that multitask modeling (P5-S) seeks a good balance among tasks and improves recommendation performance by leveraging the power of language understanding. Besides, both P5-S and P5-SN perform better than or comparable with state-of-the-art baselines on all tasks, as shown in Table 2 through Table 7, which demonstrates the power of P5 for recommendation.
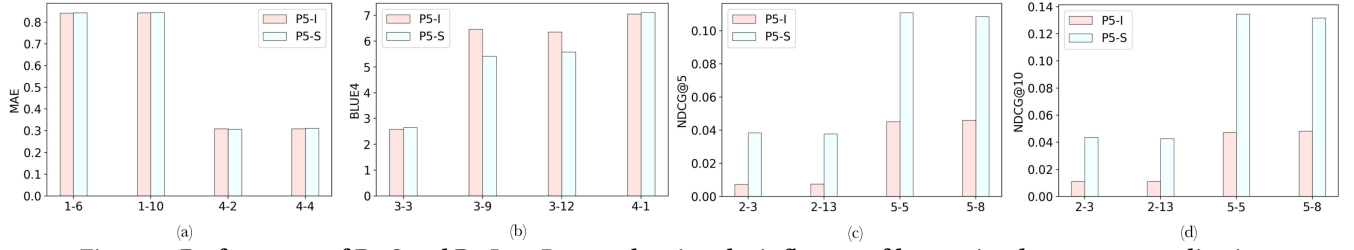
Figure 6: Performance of P5-S and P5-I on *Beauty* showing the influence of how to implement personalization.

Table 9: Performance on zero-shot domain transfer.

Directions	Z-1 & Z-4	Z-2 & Z-3	Z-5 & Z-7		Z-6	
	Accuracy	MAE	BLUE4	ROUGE1	BLUE4	ROUGE1
Toys -> *Beauty*	0.7922	0.8244	1.8869	61.1919	5.4609	66.4931
Toys -> *Sports*	0.8682	0.6644	0.7405	60.9575	2.2601	62.0353
Beauty -> *Toys*	0.8073	0.7792	0.0929	41.3061	11.8046	64.8701
Beauty -> *Sports*	0.8676	0.6838	0.0346	39.7191	6.6409	66.9222
Sports -> *Toys*	0.8230	0.7443	0.0687	42.9310	13.3408	69.7910
Sports -> *Beauty*	0.8057	0.8102	0.0790	41.0659	13.1690	66.7687

5.7 Ablation on Prompt Scaling (RQ3)

As mentioned in implementation details, our default pretrain–predict task combination follows the leave-one-out strategy. However, do we need so many prompts during pretraining to enable P5's zero-shot generalization ability? In this section, we explore to reduce the number of pretraining prompts and then make comparisons with the P5 model pretrained under default setup. To this end, we choose a collection of pretraining prompts that has the minimum number of prompts to cover all important personalized fields. Specifically, this combination contains the following 18 personalized prompts: {1-5, 1-6, 1-8, 1-9, 2-1, 2-3, 2-8, 2-11, 3-2, 3-3, 3-6, 3-9, 4-1, 4-2, 4-3, 5-2, 5-5, 5-7}. Similar to the default pretrain–predict combination, the last prompt in each task family is for zero-shot evaluation. We name this prompt scaling variant of P5-small as **P5-PS** and then pretrain P5-PS on *Beauty* dataset. The performance comparison between P5-S and P5-PS is also presented in Figure 5. From the figure, we can observe that P5-S beats P5-PS on most tasks except for some generation tasks (i.e., Prompts 3-3, 3-9 & 4-1). Interestingly, P5-S outperforms P5-PS on Prompt 3-12 – a zero-shot explanation generation task. In fact, P5-S also shows its superiority on other zero-shot tasks such as Prompts 1-10, 2-13, and 5-8. Overall, we can find that larger number of high quality personalized prompts can generally help P5 achieve better performances on various recommendation tasks especially zero-shot tasks with unseen prompts.

5.8 How to Implement Personalization (RQ4)

In this section, we discuss different strategies to implement personalization in P5. The default practice is using SentencePiece tokenizer to split personalized fields into multiple sub-word units and meanwhile using whole-word embedding to preserve the field information (Figure 3). A straightforward alternative is creating an independent extra token for each user and item. Here we name this P5-small variant as **P5-I** and also pretrain it on *Beauty* dataset.

While the former utilizes collaborative learning to implicitly optimize the latent correlations among different sub-word tokens, the latter learns a unique personalized representation for every extra token. The performance comparison between P5-S and P5-I is shown in Figure 6. We can see that P5-I achieves similar performances as P5-S on regression tasks (Prompts 1-6 & 1-10 for rating prediction, Prompts 4-2 & 4-4 for review-based rating regression) and review summarization tasks (Prompt 4-1). Also, P5-I is slightly better than P5-S on explanation generation tasks (Prompts 3-3, 3-9 & 3-12). However, P5-I significantly underperforms P5-S by a large margin on both sequential and direct recommendation tasks (all prompts in Figure 6 (c) & (d)). The reason behind P5-I's lower performance lies in that the newly introduced huge number of extra tokens and embeddings cannot be well trained compared with the original sub-word units initialized from T5. This shows that our default setting can achieve better recommendation and overall performances with the help of collaborative learning while keeping a small and constant amount of learnable tokens.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we present P5 which unifies different recommendation tasks into a shared language modeling and natural language generation framework. By designing a collection of personalized prompts covering five recommendation task families, we transfer all raw data such as the user-item interactions, user descriptions, item metadata, and user reviews to the same format – input-target text pairs. We then pretrain P5 in a full language environment to help it discover deeper semantics for various recommendation tasks. According to our experiments, P5 can beat or achieve similar performance with several representative approaches on all five task families. Moreover, P5 shows the generalization ability on performing zero-shot transfer to new items, new domains, and new personalized prompts. In the future, we will continue exploring to further enlarge the model size of P5 and employ more powerful base models such as GPT-3, OPT, and BLOOM. Besides, P5 is a very flexible paradigm and it is promising to further extend P5 to diverse modalities and more tasks such as conversational recommendation, comparative recommendation, cross-platform recommendation, or even various search tasks by incorporating user queries into P5. Finally, in this work, we designed explicit prompts since they are intuitive, flexible, and close to the natural way of how humans communicate with each other, which enables instruction-based recommendation, while in the future, we will also investigate prompt search and/or latent prompt techniques to achieve instruction prompts or leverage retrieval-enhanced generation to further boost P5's performance on downstream tasks.

ACKNOWLEDGMENT

REFERENCES

[1] Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. ExT5: Towards Extreme Multi-Task Scaling for Transfer Learning. In *International Conference on Learning Representations*.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.

[3] Hanxiong Chen, Shaoyun Shi, Yunqi Li, and Yongfeng Zhang. 2021. Neural collaborative reasoning. In *Proceedings of the Web Conference 2021*. 1516–1527.

[4] Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2019. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 765–774.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[6] Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying Vision-and-Language Tasks via Text Generation. In *ICML*.

[7] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1724–1734.

[8] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 815–824.

[9] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. 2021. Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation. In *Fifteenth ACM Conference on Recommender Systems*. 143–153.

[10] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *EACL*.

[11] Avia Efrat and Omer Levy. 2020. The Turking Test: Can Language Models Understand Instructions? *arXiv preprint arXiv:2010.11982* (2020).

[12] Zuohui Fu, Yikun Xian, Shijie Geng, Gerard De Melo, and Yongfeng Zhang. 2021. Popcorn: Human-in-the-loop Popularity Debiasing in Conversational Recommender Systems. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 494–503.

[13] Zuohui Fu, Yikun Xian, Yongfeng Zhang, and Yi Zhang. 2020. Tutorial on conversational recommendation systems. In *fourteenth ACM conference on recommender systems*. 751–753.

[14] Zuohui Fu, Yikun Xian, Yaxin Zhu, Shuyuan Xu, Zelong Li, Gerard De Melo, and Yongfeng Zhang. 2021. HOOPS: Human-in-the-Loop Graph Reasoning for Conversational Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2415–2421.

[15] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. 2010. Learning attribute-to-feature mappings for cold-start recommendations. In *2010 IEEE International Conference on Data Mining*. IEEE, 176–185.

[16] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL-IJCNLP*.

[17] Yingqiang Ge, Shuchang Liu, Zuohui Fu, Juntao Tan, Zelong Li, Shuyuan Xu, Yunqi Li, Yikun Xian, and Yongfeng Zhang. 2022. A survey on trustworthy recommender systems. *arXiv:2207.12515* (2022).

[18] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: Pre-trained Prompt Tuning for Few-shot Learning. *arXiv preprint arXiv:2109.04332* (2021).

[19] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 1725–1731.

[20] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. 1–9.

[21] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[22] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–36.

[23] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics* 8 (2020), 423–438.

[24] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[25] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[26] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. 208–211.

[27] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. Melu: Meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1073–1082.

[28] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *EMNLP*.

[29] Jingjing Li, Mengmeng Jing, Ke Lu, Lei Zhu, Yang Yang, and Zi Huang. 2019. From zero-shot learning to cold-start recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 4189–4196.

[30] Lei Li, Yongfeng Zhang, and Li Chen. 2020. Generate neural template explanations for recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 755–764.

[31] Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized Transformer for Explainable Recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 4947–4957.

[32] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 345–354.

[33] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*.

[34] Yunqi Li, Hanxiong Chen, Juntao Tan, and Yongfeng Zhang. 2022. Causal factorization machine for robust recommendation. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*. 1–9.

[35] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.

[36] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What Makes Good In-Context Examples for GPT-3? *arXiv preprint arXiv:2101.06804* (2021).

[37] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586* (2021).

[38] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT Understands, Too. *arXiv preprint arXiv:2103.10385* (2021).

[39] Ilya Loshchilov and Frank Hutter. 2018. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[40] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

[41] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 825–833.

[42] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: An embedding and mapping approach.. In *IJCAI*, Vol. 17. 2464–2470.

[43] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A Simple and Strong Baseline for Collaborative Filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 1243–1252.

[44] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[45] Guanghui Qin and Jason Eisner. 2021. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 5203–5212.

[46] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* (2019).

[47] 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. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[48] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[49] 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* (Montreal, Quebec, Canada) *(UAI '09)*. AUAI Press, Arlington, Virginia, USA, 452–461.

[50] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 175–186.

[51] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask Prompted Training Enables Zero-Shot Task Generalization. In *International Conference on Learning Representations*.

[52] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[53] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. 253–260.

[54] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1715–1725.

[55] Shaoyun Shi, Min Zhang, Xinxing Yu, Yongfeng Zhang, Bin Hao, Yiqun Liu, and Shaoping Ma. 2019. Adaptive feature sampling for recommendation with missing content feature values. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1451–1460.

[56] Kyuyong Shin, Hanock Kwak, Kyung-Min Kim, Minkyu Kim, Young-Jin Park, Jisu Jeong, and Seungjae Jung. 2021. One4all User Representation for Recommender Systems in E-commerce. *arXiv preprint arXiv:2106.00573* (2021).

[57] Kyuyong Shin, Hanock Kwak, Kyung-Min Kim, Su Young Kim, and Max Nihlen Ramstrom. 2021. Scaling Law for Recommendation Models: Towards General-purpose User Representations. *arXiv preprint arXiv:2111.11294* (2021).

[58] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*.

[59] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 650–658.

[60] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[61] Yueming Sun and Yi Zhang. 2018. Conversational recommender system. In *The 41st international acm sigir conference on research & development in information retrieval*. 235–244.

[62] Juntao Tan, Shuyuan Xu, Yingqiang Ge, Yunqi Li, Xu Chen, and Yongfeng Zhang. 2021. Counterfactual explainable recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 1784–1793.

[63] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 565–573.

[64] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. 2017. A meta-learning perspective on cold-start recommendations for items. *Advances in neural information processing systems* 30 (2017).

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[66] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework. *arXiv preprint arXiv:2202.03052* (2022).

[67] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*.

[68] Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E Peters. 2020. Learning from Task Descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1361–1375.

[69] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 6389–6394.

[70] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 285–294.

[71] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. 2021. Crossing the Format Boundary of Text and Boxes: Towards Unified Vision-Language Modeling. *arXiv preprint arXiv:2111.12085* (2021).

[72] Fajie Yuan, Guoxiao Zhang, Alexandros Karatzoglou, Joemon Jose, Beibei Kong, and Yudong Li. 2021. 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*. 696–705.

[73] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation.. In *IJCAI*. 4320–4326.

[74] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1449–1458.

[75] Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

[76] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th acm international conference on information and knowledge management*. 177–186.

[77] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.

[78] Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. 2014. Do users rate or review? Boost phrase-level sentiment labeling with review-level sentiment classification. In *SIGIR*.

[79] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*. 425–434.

[80] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1893–1902.

[81] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2021. Learning to Prompt for Vision-Language Models. *arXiv preprint arXiv:2109.01134* (2021).

[82] Feng Zhu, Yan Wang, Chaochao Chen, Jun Zhou, Longfei Li, and Guanfeng Liu. 2021. Cross-Domain Recommendation: Challenges, Progress, and Prospects. In *IJCAI*. https://doi.org/10.24963/ijcai.2021/639

APPENDIX

In this appendix, we first provide additional experimental results on *Yelp* dataset in Section A. Then we collect and report statistics on the training and inference time of P5 variants in Section B. Since personalized prompts constitute a very important part of our work, we thus offer the full list of the personalized prompts for Amazon datasets in Section C and Yelp dataset in Section D, respectively.

A   EXPERIMENTAL RESULTS ON *YELP* DATASET

For *Yelp* dataset, we also follow the default pretrain–predict task combination setup. Based on the personalized prompts presented in Section D, we leave Prompt 1-10, Prompt 2-13, Prompt 3-10, Prompt 4-3, and Prompt 5-8 for zero-shot evaluation and pretrain a P5-S model with the remaining prompts. We adopt the same baselines as described in Section 5.2 for performance comparison. Again, for each task family, we choose one or more **seen** prompts as supplement to the aforementioned zero-shot **unseen** prompts to perform evaluations. The performances of P5-S and relevant baseline models are shown in Table 10 to Table 14. As indicated in these experimental results, P5 also shows great capability on Yelp, especially for the sequential recommendation and explanation generation tasks.

B   STATISTICS ON TRAINING & INFERENCE TIME (RQ5)

In this section, we provide statistics on the training and inference time of P5 models, we collect the running time on the *Beauty* dataset. As mentioned in Section 5.1, we trained our P5 models on $4 \times$ A5000 GPUs. From our records, P5-S spent 6.7 hours to finish training, while P5-B took 24.3 hours due to the larger number of parameters. In terms of inference, we use a single A5000 GPU to conduct evaluations. The average inference time of P5-S and P5-B on different tasks are presented in Table 15. Among all tasks, sequential and direct recommendation tasks require much longer inference time than other tasks. This can be ascribed to the beam search step described in Eq. (2). Since we need to generate a list of recommended items, the larger the beam size B is, the longer the decoding will take. Besides, we can also observe that direct recommendation typically takes longer than sequential recommendation. The reason is that 100 candidates are included in the input of direct recommendation prompts, which usually have more tokens than that of sequential recommendation prompts. Overall, even though the pretraining of P5 takes hours to finish, but the inference is very fast. It is also promising to further reduce the training and inference time with the help of efficient Transformer techniques.

Table 10: Performance comparison on rating prediction.

Methods	Yelp	
	RMSE	MAE
MF	**1.2645**	1.0426
MLP	1.2951	1.0340
P5-S (1-6)	1.4868	1.0186
P5-S (1-10)	1.4685	**1.0054**

Table 11: Performance on sequential recommendation.

Methods	Yelp			
	HR@5	NDCG@5	HR@10	NDCG@10
Caser	0.0151	0.0096	0.0253	0.0129
HGN	0.0186	0.0115	0.0326	0.0159
GRU4Rec	0.0152	0.0099	0.0263	0.0134
BERT4Rec	0.0051	0.0033	0.0090	0.0045
FDSA	0.0158	0.0098	0.0276	0.0136
SASRec	0.0162	0.0100	0.0274	0.0136
S^3-Rec	0.0201	0.0123	0.0341	0.0168
P5-S (2-3)	0.0568	0.0402	**0.0707**	**0.0447**
P5-S (2-13)	**0.0574**	**0.0403**	0.0703	0.0445

Table 12: Performance on explanation generation.

Methods	Yelp			
	BLUE4	ROUGE1	ROUGE2	ROUGEL
Attn2Seq	0.8031	14.1185	1.9730	10.9220
NRT	0.8128	13.9256	1.9635	10.6980
PETER	0.5938	12.0065	1.8645	9.4645
P5-S (3-2)	**3.4515**	**63.8361**	**26.0032**	**26.9255**
PETER+	3.2827	27.2366	8.1941	21.1573
P5-S (3-7)	9.8952	70.4083	41.8598	34.5578
P5-S (3-10)	**9.9617**	**70.6757**	**41.9679**	**34.6106**

Table 13: Performance on review preference prediction.

Methods	Yelp	
	RMSE	MAE
T0 (4-2)	0.5383	0.2756
T0 (4-3)	**0.5359**	**0.2732**
P5-S (4-2)	0.6301	0.3113
P5-S (4-3)	0.6350	0.3150

Table 14: Performance on direct recommendation.

Methods	Yelp				
	HR@1	HR@5	NDCG@5	HR@10	NDCG@10
BPR-MF	**0.0813**	0.2251	0.1543	0.3312	0.1886
BPR-MLP	0.0489	0.1876	0.1184	0.3066	0.1566
SimpleX	0.0569	**0.3970**	**0.2538**	**0.5473**	**0.3020**
P5-S (5-1)	0.0603	0.2097	0.1354	0.3197	0.1708
P5-S (5-4)	0.0567	0.2065	0.1320	0.3190	0.1682
P5-S (5-5)	0.0696	0.2134	0.1423	0.3178	0.1758
P5-S (5-8)	0.0705	0.2136	0.1432	0.3219	0.1780

Table 15: Average inference time (in milliseconds) of P5 variants on different tasks on the *Beauty* dataset.

	per user		per user–item pair		per review	
Models	Sequential	Direct	Rating	Explanation	Summarization	Preference
	(2-13)	(5-8)	(1-10)	(3-12)	(4-1)	(4-4)
P5-S	53.96	57.40	4.93	13.41	8.66	6.25
P5-B	78.19	98.80	13.40	34.71	23.51	14.85

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, Yongfeng Zhang

C   FULL LIST OF PERSONALIZED PROMPTS FOR *AMAZON* DATASETS

C.1   Task Family 1: Rating Prediction

Prompt ID: 1-1

Input template: Which star rating will user_{{user_id}} give item_{{item_id}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-2

Input template: How will user_{{user_id}} rate this product: {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-3

Input template: Will user_{{user_id}} give item_{{item_id}} a {{star_rating}}-star rating? (1 being lowest and 5 being highest)

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 1-4

Input template: Does user_{{user_id}} like or dislike item_{{item_id}}?

Target template:
{{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: 1-5

Input template: Predict the user_{{user_id}} 's preference on item_{{item_id}} ({{item_title}})
-1 \n -2 \n -3 \n -4 \n -5

Target template: {{star_rating}}

Prompt ID: 1-6

Input template: What star rating do you think {{user_desc}} will give item_{{item_id}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-7

Input template: How will {{user_desc}} rate this product: {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-8

Input template: Will {{user_desc}} give a {{star_rating}}-star rating for {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 1-9

Input template: Does {{user_desc}} like or dislike {{item_title}}?

Target template:
{{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: 1-10

Input template: Predict {{user_desc}} 's preference towards {{item_title}} (1 being lowest and 5 being highest)

Target template: {{star_rating}}

C.2   Task Family 2: Sequential Recommendation

Prompt ID: 2-1

Input template: Given the following purchase history of user_{{user_id}}:
{{purchase_history}}
predict next possible item to be purchased by the user?

Target template: {{next_item}}

Prompt ID: 2-2

Input template: I find the purchase history list of user_{{user_id}}:
{{purchase_history}}
I wonder which is the next item to recommend to the user. Can you help me decide?

Target template: {{next_item}}

Prompt ID: 2-3

Input template: Here is the purchase history list of user_{{user_id}}:
{{purchase_history}}
try to recommend next item to the user

Target template: {{next_item}}

Prompt ID: 2-4

Input template: Given the following purchase history of {{user_desc}}:
{{purchase_history}}
predict next possible item for the user

Target template: {{next_item}}

Prompt ID: 2-5

Input template: Based on the purchase history of {{user_desc}}:
{{purchase_history}}
Can you decide the next item likely to be purchased by the user?

Target template: {{next_item}}

Prompt ID: 2-6

Input template: Here is the purchase history of {{user_desc}}:
{{purchase_history}}
What to recommend next for the user?

Target template: {{next_item}}

Prompt ID: 2-7

Input template: Here is the purchase history of user_{{user_id}}:
{{purchase_history}}
Select the next possible item likely to be purchased by the user from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-8

Input template: Given the following purchase history of {{user_desc}}:
{{purchase_history}}
What to recommend next for the user? Select one from the following items:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-9

Input template: Based on the purchase history of user_{{user_id}}:
{{purchase_history}}
Choose the next possible purchased item from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-10

Input template: I find the purchase history list of {{user_desc}}:
{{purchase_history}}
I wonder which is the next item to recommend to the user. Try to select one from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-11

Input template: User_{{user_id}} has the following purchase history:
{{purchase_history}}
Does the user likely to buy {{candidate_item}} next?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 2-12

Input template: According to {{user_desc}} 's purchase history list:
{{purchase_history}}
Predict whether the user will purchase {{candidate_item}} next?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 2-13

Input template: According to the purchase history of {{user_desc}}:
{{purchase_history}}
Can you recommend the next possible item to the user?

Target template: {{next_item}}

C.3 Task Family 3: Explanation Generation

Prompt ID: 3-1

Input template: Generate an explanation for user_{{user_id}} about
this product: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-2

Input template: Given the following review headline
{{review_headline}}
can you help generate an explanation of user_{{user_id}} for
item_{{item_id}}?

Target template: {{explanation}}

Prompt ID: 3-3

Input template: Help user_{{user_id}} generate a {{star_rating}}-star
explanation about this product:
{{item_title}}

Target template: {{explanation}}

Prompt ID: 3-4

Input template: Generate an explanation for {{user_desc}} about this
product: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-5

Input template: Based on the following review headline:
{{review_headline}}
Generate {{user_desc}} 's purchase explanation about {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-6

Input template: Help {{user_desc}} generate a {{star_rating}}-star
explanation for item_{{item_id}}

Target template: {{explanation}}

Prompt ID: 3-7

Input template: Predict the star rating, then use {{feature_word}} as
feature word to generate user_{{user_id}} 's purchase explanation for
item_{{item_id}}

Target template: {{star_rating}}, {{explanation}}

Prompt ID: 3-8

Input template: What score will {{user_desc}} rate item_{{item_id}}?
Then give an explanation for the rating score. (1 being lowest and 5
being highest)

Target template: {{star_rating}}, {{explanation}}

Prompt ID: 3-9

Input template: Based on the feature word {{feature_word}}, generate
an explanation for user_{{user_id}} about this product:
{{item_title}}

Target template: {{explanation}}

Prompt ID: 3-10

Input template: Given the word {{feature_word}}, can you help
generate an explanation for {{user_desc}} about the product: \n
{{item_title}}

Target template: {{explanation}}

Prompt ID: 3-11

Input template: Using the word {{feature_word}}, write a
{{star_rating}}-star explanation for user_{{user_id}} about
item_{{item_id}}

Target template: {{explanation}}

Prompt ID: 3-12

Input template: According to the feature word {{feature_word}},
generate a {{star_rating}}-star explanation for {{user_desc}} about
item_{{item_id}}

Target template: {{explanation}}

C.4 Task Family 4: Review Related

Prompt ID: 4-1

Input template: Write a short sentence to summarize the following
product review from user_{{user_id}}:
{{review}}

Target template: {{summary}}

Prompt ID: 4-2

Input template: Given the following review written by
user_{{user_id}}:
{{review}}
Can you predict the associated star rating (1 being lowest and 5
being highest)?

Target template: {{star_rating}}

Prompt ID: 4-3

Input template: Give a short sentence describing the following
product review from {{user_desc}}:
{{review}}

Target template: {{summary}}

Prompt ID: 4-4

Input template: According to the following review written by
{{user_desc}}:
{{review}}
Predict the associated star rating (1 being lowest and 5 being
highest)

Target template: {{star_rating}}

C.5 Task Family 5: Direct Recommendation

Prompt ID: 5-1

Input template: Will user_{{user_id}} likely to interact with
item_{{item_id}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-2

Input template: Shall we recommend item_{{item_id}} to {{user_desc}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-3

Input template: For {{user_desc}}, do you think it is good to recommend {{item_title}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-4

Input template: I would like to recommend some items for user_{{user_id}}. Is the following item a good choice? {{item_title}}

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-5

Input template: Which item of the following to recommend for {{user_desc}}? {{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-6

Input template: Choose the best item from the candidates to recommend for {{user_desc}}? {{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-7

Input template: Pick the most suitable item from the following list and recommend to user_{{user_id}}: {{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-8

Input template: We want to make recommendation for user_{{user_id}}. Select the best item from these candidates: {{candidate_items}}

Target template: {{target_item}}

C.6 Task Family Z: Zero-Shot Generalization

Prompt ID: Z-1

Input template: Given the facts about the new product, do you think user {{user_id}} will like or dislike it? title: {{item_title}} brand: {{brand}} price: {{price}}

Target template: {{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: Z-2

Input template: Here are the details about a new product: title: {{item_title}} brand: {{brand}} price: {{price}} What star will {{user_desc}} probably rate the product?
-1 -2 -3 -4 -5

Target template: {{star_rating}}

Prompt ID: Z-3

Input template: Predict user_{{user_id}} 's preference about the new product (1 being lowest and 5 being highest):
title: {{item_title}} price: {{price}} brand: {{brand}}

Target template: {{star_rating}}

Prompt ID: Z-4

Input template: Will {{user_desc}} like or dislike the following product? title: {{item_title}} price: {{price}} brand: {{brand}}

Target template:
{{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: Z-5

Input template: Generate a possible explanation for {{user_desc}} 's preference about the following product: title: {{item_title}} brand: {{brand}} price: {{price}}

Target template: {{explanation}}

Prompt ID: Z-6

Input template: Based on the word {{feature_word}}, help user_{{user_id}} write a {{star_rating}}-star explanation for this new product: title: {{item_title}} price: {{price}} brand: {{brand}}

Target template: {{explanation}}

Prompt ID: Z-7

Input template: For the new product {{item_title}}, we would like to know whether {{user_desc}} will love it. If you think the user will love it, please help explain why.

Target template: {{explanation}}

D FULL LIST OF PERSONALIZED PROMPTS FOR *YELP* DATASET

D.1 Task Family 1: Rating Prediction

Prompt ID: 1-1

Input template: Which star rating will user_{{user_id}} give item_{{item_id}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-2

Input template: How will user_{{user_id}} rate this business: {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-3

Input template: Will user_{{user_id}} give item_{{item_id}} a {{star_rating}}-star rating? (1 being lowest and 5 being highest)

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 1-4

Input template: Does user_{{user_id}} like or dislike item_{{item_id}}?

Target template: {{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: 1-5

Input template: Predict the user_{{user_id}} 's preference on item_{{item_id}} ({{item_title}})
-1 \n -2 \n -3 \n -4 \n -5

Target template: {{star_rating}}

Prompt ID: 1-6

Input template: What star rating do you think {{user_desc}} will give item_{{item_id}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-7

Input template: How will {{user_desc}} rate this business: {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{star_rating}}

Prompt ID: 1-8

Input template: Will {{user_desc}} give a {{star_rating}}-star rating for {{item_title}}? (1 being lowest and 5 being highest)

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 1-9

Input template: Does {{user_desc}} like or dislike {{item_title}}?

Target template: {{answer_choices[label]}} (like/dislike) – like (4,5) / dislike (1,2,3)

Prompt ID: 1-10

Input template: Predict {{user_desc}} 's preference towards {{item_title}} (1 being lowest and 5 being highest)

Target template: {{star_rating}}

D.2   Task Family 2: Sequential Recommendation

Prompt ID: 2-1

Input template: Given the following visit history of user_{{user_id}}: {{visit_history}}
predict next possible business to be visited by the user?

Target template: {{next_item}}

Prompt ID: 2-2

Input template: I find the visit history list of user_{{user_id}}: {{visit_history}}
I wonder which is the next item to recommend to the user. Can you help me decide?

Target template: {{next_item}}

Prompt ID: 2-3

Input template: Here is the visit history list of user_{{user_id}}: {{visit_history}}
try to recommend next item to the user

Target template: {{next_item}}

Prompt ID: 2-4

Input template: Given the following visit history of {{user_desc}}: {{visit_history}}
predict next possible business for the user

Target template: {{next_item}}

Prompt ID: 2-5

Input template: Based on the visit history of {{user_desc}}: {{visit_history}}
Can you decide the next business likely to be visited by the user?

Target template: {{next_item}}

Prompt ID: 2-6

Input template: Here is the visit history of {{user_desc}}: {{visit_history}}
What to recommend next for the user?

Target template: {{next_item}}

Prompt ID: 2-7

Input template: Here is the visit history of user_{{user_id}}: {{visit_history}}
Select the next possible business likely to be visited by the user from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-8

Input template: Given the following visit history of {{user_desc}}: {{visit_history}}
What to recommend next for the user? Select one from the following items:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-9

Input template: Based on the visit history of user_{{user_id}}: {{visit_history}}
Choose the next possible visited business from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-10

Input template: I find the visit history list of {{user_desc}}: {{visit_history}}
I wonder which is the next item to recommend to the user. Try to select one from the following candidates:
{{candidate_items}}

Target template: {{next_item}}

Prompt ID: 2-11

Input template: User_{{user_id}} has the following visit history:
{{visit_history}}
Does the user likely to visit {{candidate_item}} next?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 2-12

Input template: According to {{user_desc}} 's visit history list:
{{visit_history}}
Predict whether the user will visit {{candidate_item}} next?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 2-13

Input template: According to the visit history of {{user_desc}}:
{{visit_history}}
Can you recommend the next possible business to the user?

Target template: {{next_item}}

D.3   Task Family 3: Explanation Generation

Prompt ID: 3-1

Input template: Generate an explanation for user_{{user_id}} about this business: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-2

Input template: Help user_{{user_id}} generate a {{star_rating}}-star explanation about this business: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-3

Input template: Generate an explanation for {{user_desc}} about this business: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-4

Input template: Help {{user_desc}} generate a {{star_rating}}-star explanation for item_{{item_id}}

Target template: {{explanation}}

Prompt ID: 3-5

Input template: Predict the star rating, then use {{feature_word}} as feature word to generate user_{{user_id}} 's visit explanation for item_{{item_id}}

Target template: {{star_rating}}, {{explanation}}

Prompt ID: 3-6

Input template: What score will {{user_desc}} rate item_{{item_id}}? Then give an explanation for the rating score. (1 being lowest and 5 being highest)

Target template: {{star_rating}}, {{explanation}}

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, Yongfeng Zhang

Prompt ID: 3-7

Input template: Based on the feature word {{feature_word}}, generate an explanation for user_{{user_id}} about this business: {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-8

Input template: Given the word {{feature_word}}, can you help generate an explanation for {{user_desc}} about the business: \n {{item_title}}

Target template: {{explanation}}

Prompt ID: 3-9

Input template: Using the word {{feature_word}}, write a {{star_rating}}-star explanation for user_{{user_id}} about item_{{item_id}}

Target template: {{explanation}}

Prompt ID: 3-10

Input template: According to the feature word {{feature_word}}, generate a {{star_rating}}-star explanation for {{user_desc}} about item_{{item_id}}

Target template: {{explanation}}

D.4 Task Family 4: Review Related

Prompt ID: 4-1

Input template: Predict the associated rating score of the review written by user_{{user_id}} (1 being lowest and 5 being highest): {{review}}

Target template: {{star_rating}}

Prompt ID: 4-2

Input template: Given the following review written by user_{{user_id}}:
{{review}}
Can you predict the associated star rating (1 being lowest and 5 being highest)?

Target template: {{star_rating}}

Prompt ID: 4-3

Input template: According to the following review written by {{user_desc}}:
{{review}}
Predict the associated star rating (1 being lowest and 5 being highest)

Target template: {{star_rating}}

D.5 Task Family 5: Direct Recommendation

Prompt ID: 5-1

Input template: Will user_{{user_id}} likely to interact with item_{{item_id}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-2

Input template: Shall we recommend item_{{item_id}} to {{user_desc}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-3

Input template: For {{user_desc}}, do you think it is good to recommend {{item_title}}?

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-4

Input template: I would like to recommend some items for user_{{user_id}}. Is the following item a good choice? {{item_title}}

Target template: {{answer_choices[label]}} (yes/no)

Prompt ID: 5-5

Input template: Which item of the following to recommend for {{user_desc}}?
{{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-6

Input template: Choose the best item from the candidates to recommend for {{user_desc}}?
{{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-7

Input template: Pick the most suitable item from the following list and recommend to user_{{user_id}}:
{{candidate_items}}

Target template: {{target_item}}

Prompt ID: 5-8

Input template: We want to make recommendation for user_{{user_id}}. Select the best item from these candidates:
{{candidate_items}}

Target template: {{target_item}}