Contrastive Graph Prompt-tuning for Cross-domain Recommendation

ZIXUAN YI, University of Glasgow, United Kingdom
IADH OUNIS, University of Glasgow, United Kingdom
CRAIG MACDONALD, University of Glasgow, United Kingdom

Recommender systems commonly suffer from the long-standing data sparsity problem where insufficient user-item interaction data limits the systems' ability to make accurate recommendations. This problem can be alleviated using cross-domain recommendation techniques. In particular, in a cross-domain setting, knowledge sharing between domains permits improved effectiveness on the target domain. While recent cross-domain recommendation techniques used a pre-training configuration, we argue that such techniques lead to a low fine-tuning efficiency, especially when using large neural models. In recent language models, *prompts* have been used for parameter-efficient and time-efficient tuning of the models on the downstream tasks - these prompts represent a tunable latent vector that permits to freeze the rest of the language model's parameters. To address the cross-domain recommendation task in an efficient manner, we propose a novel Personalised Graph Prompt-based Recommendation (PGPRec) framework, which leverages the efficiency benefits from prompt-tuning. In such a framework, we develop personalised and item-wise graph prompts based on relevant items to those items the user has interacted with. In particular, we apply Contrastive Learning (CL) to generate the pre-trained embeddings, to allow an increased generalisability in the pre-training stage and to ensure an effective prompt-tuning stage. To evaluate the effectiveness of our PGPRec framework in a cross-domain setting, we conduct an extensive evaluation with the top-*k* recommendation task and perform a cold-start analysis. The obtained empirical results on four Amazon Review datasets show that our proposed PGPRec framework can reduce up to 74% of the tuned parameters with a competitive performance and achieves an 11.41% improved performance compared to the strongest baseline in a cold-start scenario.

CCS Concepts: • **Information systems** → **Recommender systems**.

Additional Key Words and Phrases: Personalisation, Recommender system, Graph Neural Network

1 INTRODUCTION

Personalised recommendation techniques, which learn user preferences and find items related to their interest, have been increasingly developed in the last decades. In particular, for a recommender, the effective personalisation of the recommendation results, frequently rely on rich available data, such as historical user-item interactions, domain knowledge, as well as the user demographics and profiles [18]. However, in the recommendation literature, the performance of a personalised recommender system deployed on a single domain, often suffers from the commonly observed data sparsity issue. This refers to an insufficient number of user-item interactions, which hinders accurate recommendation generation [29]. Therefore, Cross-Domain Recommendation (CDR) [47, 63], which relies on shared users over pairs of domains to transfer relevant information from the source domain to

Authors' addresses: Zixuan Yi, z.yi.1@research.gla.ac.uk, University of Glasgow, Glasgow, Scotland, United Kingdom; Iadh Ounis, iadh.ounis@glasgow.ac.uk, University of Glasgow, Glasgow, Scotland, United Kingdom; Craig Macdonald, craig.macdonald@glasgow.ac.uk, University of Glasgow, Glasgow, Scotland, United Kingdom.

1046-8188/2023/8-ART $15.00
https://doi.org/10.1145/3618298

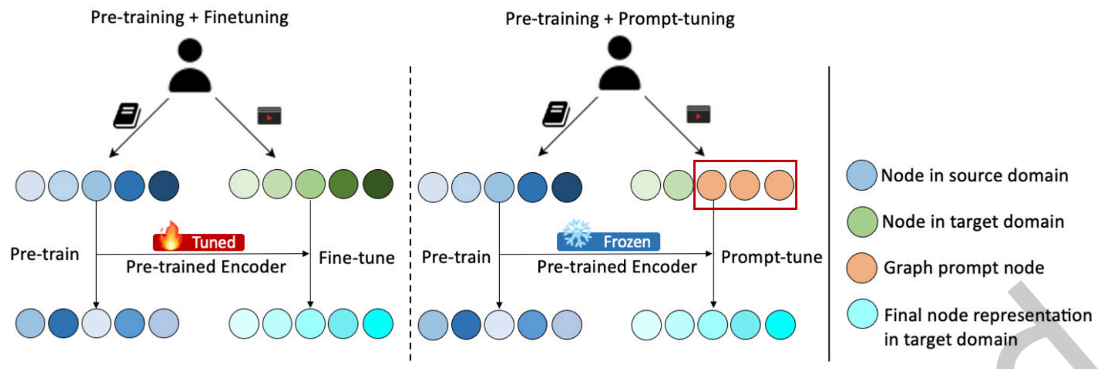

Fig. 1. Overview of conventional graph fine-tuning and personalised graph prompt-tuning in Cross-Domain Recommendation (CDR).

the target domain, is typically used to alleviate the negative effect of sparse interactions, so as to improve the recommendation performance when applied to a different target domain.

Inspired by recent advances in natural language processing [6] centred on the paradigm of first pre-training then fine-tuning, some cross-domain recommendation techniques [50, 66] considered the application of this pre-training and fine-tuning mechanism. In particular, a common development routine for shared users in cross-domain recommendation (CDR) involves first the pre-training of the recommendation model using data from the source domain to learn domain-invariant knowledge (e.g. user profiles). Subsequently, the knowledge contained in the pre-trained model is used to initialise the user/item embeddings in the target domain. Such a strategy has been shown to be effective in alleviating the mentioned data sparsity issue [56]. Moreover, a myriad of pre-training recommendation models [32, 50] have been proposed to leverage the structural data from the source domain, which can leverage collaborative filtering signals to yield better user/item representations by leveraging the high-order connectivity via graph convolutional operations[19]. However, the effectiveness of recommendation on the target domain can be negatively impacted if the pre-trained model learns from non-relevant domain-specific features in the source domain. Moreover, the fine-tuning of a pre-trained model causes all parameters of the model to be updated, even in the case where not all parameters need to be updated to obtain an effective model. This makes fine-tuning inefficient. Therefore, such an efficiency limitation provides a motivation for devising a new framework that can efficiently transfer effective pre-trained embeddings.

Recently, in the Natural Language Processing (NLP) field, prompt-tuning [1, 20] has been widely applied to address many NLP tasks and has shown its usefulness in extracting useful information from a pre-trained model, such that it can be adapted to a downstream task with less efforts. Several prompt-variants have been proposed, such as term filling-based text templates [1, 8] or continuous embeddings (i.e. soft prompt) [21, 35], which effectively bridge the pre-training process in various natural language processing tasks. In adapting prompt-tuning for recommender systems, several studies [5, 9, 55] have investigated the application of prompt techniques within the recommendation context, focusing on leveraging user and item information to generate customised prompts that can effectively address specific recommendation tasks. For example, Wu et al. [55] proposed a soft prefix prompt based on users' profiles to tackle the cold-start problem in a sequential recommendation setting. Another stream of work [5, 9] attempted to convert the available data resources, such as the user-item interactions, the user descriptions, the item metadata, and the user reviews, into natural language sequences. Next, they used a composition of such sequences as a prompt and the pre-trained models for efficiently addressing various downstream tasks of the recommender system (e.g., cold-start recommendation, few/zero-shot recommendation and user profile prediction). The aforementioned examples show that existing prompt tuning-based recommendation techniques rely on sequential patterns in the users' interacted items and mimic NLP models designed for sequential text. This demonstrates the effective application of the prompt-tuning techniques in the context of recommendation systems.

Inspired by the success of prompt-tuning in sequential recommendation, we introduce the prompt-tuning paradigm into Graph Neural Networks (GNN)-based recommendation models to alleviate their low efficiency limitation, as highlighted in [22]. However, it is challenging to design a prompt-tuning technique that makes a full use of knowledge from structural data rather than from sequential data in the existing recommendation techniques. Therefore, it is essential to bridge the GNN recommendation models with the existing prompt-tuning techniques, which use a sequential modelling of data. In this paper, we show that the language models in the NLP domain could be a special form of a GNN (as will be detailed in Section 3.3.1), which allows us to propose novel templates for the personalised graph prompts. These templates could then be applied to GNN-based recommenders. As an illustrative example, Figure 1 shows how our proposed prompt-tuning framework contrasts with conventional fine-tuning approaches in a CDR setting. The existing CDR solutions typically pre-train a graph encoder in a source domain. Then, they fine-tune the well-trained graph encoder with the interaction data for recommendation in the target domain. Therefore, in this paper, we propose a prompt-tuning framework instead of the fine-tuning process, to improve the tuning efficiency by leveraging personalised graph prompts while fixing the pre-trained parameters. To be more specific, we build personalised graph prompts with a series of relevant items, which we call the neighbouring-items[1] and leverage these prompts to efficiently enhance the final user/item representations. Hence, we argue that, in a CDR setting, GNN recommenders enriched with our proposed graph prompt can efficiently leverage the learned knowledge from the source domain to effectively improve the recommendation performance in the target domain. Furthermore, we propose a novel Personalised Graph Prompt-based Recommendation (PGPRec) framework, which encapsulates our proposed personalised prompts as well as a Contrastive Learning (CL)-empowered GNN recommender for cross-domain recommendation. Indeed, with the integration of a graph-based contrastive learning, the pre-trained model enforces the divergence of the learned user/item embeddings [54], which makes them generalisable to various target domains[2] with an improved recommendation performance.

To summarise, our contributions are four-fold:

- We propose a personalised graph prompt-based recommendation framework for cross-domain recommendation, which uses contrastive learning to enhance the user/item representations in both the pre-training and tuning stages and improves the efficiency of the tuning phase.
- We introduce item-wise personalised prompts to be used in a user-item bipartite graph to effectively enhance the user/item representations in the target domain.
- We conduct extensive experiments on four Amazon Review datasets. We perform a TOST test [42] to demonstrate that PGPRec shows a significantly improved efficiency in comparison to the state-of-the-art GNN recommenders while achieving a comparable recommendation performance.
- We further conduct a detailed analysis of the cold-start users in the target domain using four Amazon Review datasets. Our findings show that PGPRec significantly outperforms the strongest baseline in a cold-start scenario.

[1] Given an item, we denote by its "neighbouring-items" those items that share its same attributes in the target domain, i.e. the set of items with the same attributes. In this work, we define the neighbouring-items according to three types of available metadata – i.e. also_bought, also_viewed and bought_together – and refer to them as 'neighbouring-items' throughout this paper. [2] In this paper, we also evaluate the contribution of using the contrastive learning through the application of PGPRec to distinct source-target domain pairs.

The remainder of this paper is organised as follows: In Section 2, we position our proposed PGPRec framework in the literature. Section 3 describes our methodology and the specific implementations of the personalised graph prompts during the training stage as well as the optimisation objective of the cross-domain recommendation task. The experimental setup and the results of our empirical experiments are presented in Section 4, followed by concluding remarks and future work in Section 5.

2 RELATED WORK

In this section, we discuss related methods and techniques to our conducted study, namely pre-training recommenders, cross-domain recommenders and prompt-tuning techniques.

2.1 Pre-training in Recommendation

Recently, pre-training techniques, which learn knowledge from large-scale datasets for an improved model performance, have achieved several successes in addressing the recommendation task [64]. A typical approach for learning a recommendation model using the pre-training techniques involves first the initialisation of the model with knowledge obtained from a pre-training stage and subsequently the fine-tuning of the model using supervised signals from the target recommendation scenario. This pre-training and fine-tuning paradigm enables the model to effectively incorporate prior knowledge while adapting to the specific characteristics of the target domain [64]. For example, BERT4Rec [45] pre-trains a BERT transformer architecture to learn a sequential pattern of items to model the user behaviour sequences, thereby leading to a promising performance in sequential recommendation. Similarly, ASReP [27] pre-trains a transformer to generate fabricated historical items at the beginning of short sequences and then fine-tunes the transformer based on the new sequences to alleviate the cold-start problem.

On the other hand, there have been several efforts to pre-train recommenders by designing self-supervised auxiliary tasks to discover supervised signals from the raw data. Contrastive learning recommendation is increasingly considered to be a promising approach within the family of self-supervised learning recommendation approaches [60], which can be applied for various recommenders by perturbing the raw data. For example, CL4SRec [57] performs masking, cropping and reordering operations to augment the input items sequence then contrasts the augmentations to pre-train a transformer-based encoder for sequential recommendation. Another example of contrastive learning recommendation model is PCRec [50], which also uses contrastive learning to enhance the cross-domain recommendation. PCRec pre-trains a GIN encoder [58] by contrasting a random walk-based augmentation in the source domain and then transfers the pre-trained user/item embeddings to initialise a MF model in the target domain. As illustrated by the aforementioned models, contrastive learning has demonstrated its ability in learning generalisable representations in pre-training recommendations.

Another line of work [32, 56, 61] attempted to use extra knowledge to enhance the positive effect of pre-training. For example, UPRec [56] encapsulates various pre-training tasks based on user attributes and social relations to learn comprehensive user representations for an improved user-item sequence modelling. Another example, PeterRec [61], also pre-trains the user representations based on the user-item interactions, but applies the learned model to a different domain (i.e. the target domain in a cross-domain setting). As can be seen from the above work, cross-domain recommendation is well-aligned with the pre-training and fine-tuning paradigm, which aims to transfer useful knowledge from the source domain to benefit the effectiveness of the resulting recommendations in the target domain. Moreover, the existing Cross-Domain Recommendation (CDR) models typically transfer knowledge learned from sequential patterns while very few approaches attempt to benefit from the learned structural knowledge [36]. Therefore, differently from the existing approaches that learn from sequential data, we propose PGPRec to leverage contrastive learning to pre-train the GNN-based recommenders. Our proposed

PGPRec framework extracts intrinsic and structural knowledge from the pre-trained models in a cross-domain recommendation setting.

2.2 Cross-domain Recommendation

In general, cross-domain recommendation (CDR) is an application of transfer learning to recommendation scenarios involving two domains, namely a *source* domain (which provides us with additional useful knowledge, in order to reduce data sparsity and therefore improve recommendations on a separate *target* domain. While different CDR scenarios exist, namely shared-users, shared-items and shared-nothing,[3] in this paper we focus upon shared-users, where user profiles on the source domain are used to permit improved personalisation of recommendations in the target. For example, CMF [47] – a classical CDR approach – jointly factorises the rating matrices from two domains with a shared global user embedding matrix. Another approach, CoNet [14], introduces a cross-connection unit to transfer the user-item interaction features between two domains. Similarly, CBMF [33] uses a cluster-based matrix to learn the correlation between the user clusters and the item clusters in different domains and then uses this matrix to alleviate the cold-start problem. With the rise of Graph Neural Networks (GNNs) in recommender systems, PPGN [65] adopts the GNN technique to explore the high-order connectivity between users and items on a joint interaction graph of two domains so as to allow the knowledge-transfer with shared user features.

Among the CDR approaches, those in the literature [50, 62, 64, 66] that have focused on pre-training in a source domain and fine-tuning in the target domain are the most relevant to our proposed PGPRec framework. For instance, EMCDR [29] pre-trains a multi-layer perceptron (MLP) as a mapping function in a source domain and then transfers the learned user representation into the target domain for cold-start users. Similarly, PCRec [50] first pre-trains a graph encoder and then transfers the learned user features to the target domain for an improved performance after fine-tuning the resulting recommendation model. However, the existing approaches assume a prediction objective during pre-training on the source domains, which could lead to sub-optimal representations that can impede the effectiveness of the resulting recommender models [50]. Unlike these existing approaches, we adopt contrastive pre-training to encourage the pre-trained models to learn more generalisable features, with an emphasis on transferring domain-invariant knowledge. This contrastive pre-training strategy allows for a higher divergence in the representation of learned features from the source domain, providing the model with a better initialisation point for diverse target domains. By focusing on the transfer of domain-invariant knowledge, our approach fosters the development of recommendation systems that can effectively harness the power of cross-domain techniques, thereby contributing to the field of domain adaptation in recommendation systems. Beyond addressing the issue of sub-optimal training objectives, the pre-training and fine-tuning paradigm has a critical limitation when applied to cross-domain recommendation, namely its low efficiency [2]. Indeed, the necessity to train the model twice results in significant time and computational costs associated with the model's parameters. To address this low-efficiency challenge, we propose the PGPRec framework, a tailored solution for cross-domain recommendation systems, which uses a personalised graph prompt-tuning technique to adapt the users' preferences in the target domain. By leveraging pre-trained knowledge and personalised graph prompts, PGPRec provides a more efficient solution for cross-domain recommendation. On the other hand, in order to investigate the cold-start problem in cross-domain recommendation, we also leverage contrastive pre-training and personalised graph prompts to benefit the cold-start users, i.e. those users that have sparse interactions in a target domain.

[3] Actually, improvements identified in shared-nothing cross-domain recommendation have been shown to simply be due to increased model capacity [30].

2.3 Prompt-tuning

A prompt originally refers to a prefixed plain text and is combined with the input of the pre-trained models for an improved semantic understanding of the task [37]. For example, GPT-3 [1] leverages manually devised prompts for transfer learning in Natural Language Processing (NLP). With the follow-up work on prompt-variants [16, 43], there has been a growing number of recent prompt design methods, including hard and soft prompts, following the "*pre-train, prompt, and predict*" paradigm [24]. Hard prompts are discrete textual terms [1, 41] while soft prompts are one or many continuous learned embeddings [20, 21, 25]. In particular, soft prompts are randomly initialised and then optimised via parameter-tuning. Consequently, prompt-tuning, using soft prompts, narrows the gap between the objectives of the source tasks and that of the downstream tasks by transforming the inputs to the target model in a certain format [53]. In addition, the prompt-tuning methods only rely on fine-tuning a small set of parameters within the soft prompts to achieve a competitive performance compared to the fine-tuned models, which endows the final user/item representations in an efficient manner.

To the best of our knowledge, only a limited number of work have recently applied prompt-tuning to recommender systems. In order to address various downstream tasks in recommender systems, [5, 9] attempted to follow the techniques from the NLP domain in leveraging the prompt-tuning paradigm. M6-Rec [5] considered the user behaviour as a sequence of 'text' and addressed both the click-through-rate (CTR) prediction and the explainable recommendation tasks with the assistance of randomly initialised soft prompts [9]. Similarly, P5 [9] converted the user descriptions, the item metadata, and the user reviews to natural language sequences as prompts to effectively leverage a pre-trained transformer model. On the other hand, Wu et al. [55] treated cold-start recommendation as the downstream task of sequential recommendation and enhanced the performance by building prefixed soft prompts based on the user profiles. Although prompt-tuning has also been applied to sequential recommendations, the existing approaches relied on the sequential pattern of the input data. In this work, we argue that another possible strategy is to transfer the structural pattern of a user from the user-item bipartite graphs. Indeed, we introduce novel personalised graph prompts, which leverage the neighbouring-items and the tunable continuous vectors as hard and soft graph prompts, to address cross-domain recommendation with improved training efficiency. Moreover, to investigate the characteristics of prompt-tuning on graphs, we ensure that PGPRec leverages the pre-trained knowledge and informative personalised graph prompts to benefit the cold-start users, which have sparse interactions in a target domain.

3 METHODOLOGY

In this section, we first present the cross-domain recommendation task (Section 3.1). Next, we describe the composition of our proposed personalised graph prompts and training process (Section 3.2). We also describe how to derive the personalised graph prompt-tuning paradigm by justifying the equivalence between the transformer and GNN (Section 3.3), along with the detailed implementations and objectives during the training phases (Section 3.4).

3.1 Preliminaries

In this paper, we focus on addressing the ranking-based recommendation task in a cross-domain setting. Conceptually, we consider two domains, the source domain D_{S} and the target domain D_{T}. The set of users in both domains is shared, and denoted by U (of size $m = |U|$). Let the set of items in I_{S} (of size $n_s = |D_{\mathrm{S}}|$) and I_{T} (of size $n_t = |D_{\mathrm{T}}|$), respectively. Then, we aim to make effective and efficient recommendations to users in U with a ranked list of items from the target domain D_{T} , while leveraging the learned knowledge from the source domain D_{S}. In particular, we devise two bipartite graphs $\mathcal{G}_S$ and $\mathcal{G}_T$ for both the source and target domains, where the nodes represent users/items and the edges indicate interactions between the users and items. Formally speaking, with the interaction graphs $\mathcal{G}_S$ and $\mathcal{G}_T$, we pre-train a graph encoder f in a selected source domain D_{S} and adapt it to the target domain D_{T} for an enhanced knowledge of the user preferences so as to effectively and efficiently recommend the top-k items related to their interests. The notations that we will use throughout this paper are summarised in Table 1.

Table 1. Notations used in this paper to describe the proposed PGRec framework.

Symbol	Description
D_{S}	the source domain
D_{T}	the target domain
I_{S}	the set of items in D_{S}
I_{T}	the set of items in D_{T}
U	the set of users which is shared by both the source domain and the target domain
$\mathcal{G}_S$	the user-item interaction graph in D_{S}
$\mathcal{G}_T$	the user-item interaction graph in D_{T}
$\mathcal{V}$	the node set
$\mathcal{E}$	the edge set
M, M'	the masking vectors on the edge set $\mathcal{E}$
N_u	the neighbour nodes of user u in the interaction graph
N_i	the neighbour nodes of item i in the interaction graph
$R(u)$	the ground-truth set of items that user u has interacted with
$\hat{R}(u)$	the ranked list of items generated by a recommender
h^{ℓ}	the node representation at the ℓ-th layer
$\boldsymbol{e_u}$,$\boldsymbol{e_i}$	the embedding vector of the ℓ-th layer for users and items
$\boldsymbol{e_{p_h}}$	the embedding vector of the neighbouring-item as a hard prompt
$\boldsymbol{e_{p_s}}$	the random initialised embedding vector as a soft prompt
B	the batch size
λ_1, λ_2	the regularization term
Θ	the parameters of the model

3.2 The PGPRec Framework

First, we provide an overview of our proposed recommendation framework, Personalised Graph Prompt-based Recommendation (PGPRec). In particular, Figure 2 shows and illustrates the major components included in our proposed framework. Specifically, for our cross-domain recommender, we adopt the GNN technique to model the data from each domain, for capturing the high-order features and allowing the transfer of additional structural knowledge aside from the typical user/item features (e.g. the user profiles and item attributes). To instantiate the discussed GNN technique, we are not limited to the commonly used GCN model [19]. Indeed, PGPRec is also flexible in allowing to use other GNN-based techniques, such as GAT [49], NGCF [51] or LightGCN [12]. Inspired by the recent success of applying prompt-tuning in the NLP domain, we also introduce the graph prompts to assist the cross-domain recommendation. In particular, as a rationale for applying prompt-tuning to a graph model, we argue that the GNN technique can be considered as another type of a transformer model. Note that we explain and justify this argument in Section 3.3. In our PGPRec framework, we propose a mixture of hard and soft graph prompts for personalised recommendation, which we collectively denote as *personalised graph prompts*. First, we describe the hard graph prompts. For each user, we develop personalised prompts according to the associated relevant items (denoted as the neighbouring-items). Given an item, we denote by its "neighbouring-items" those

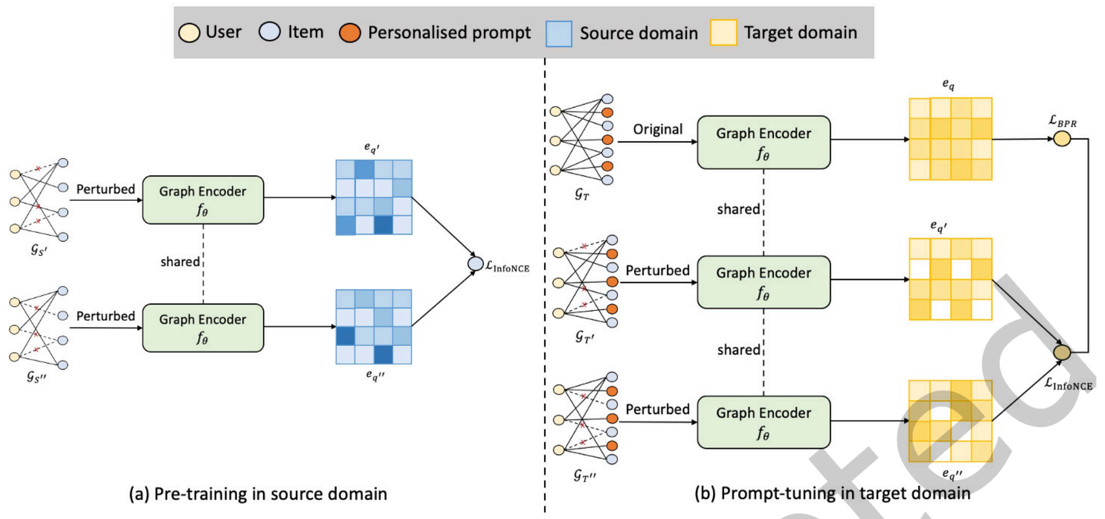

Fig. 2. An illustration of the PGPRec framework using a specific example. The framework is mainly composed of two steps. (a) In the pre-training stage, PGPRec trains a GNN model to learn the source domain's transferable knowledge through a contrastive objective. (b) Then PGPRec uses the pre-trained GNN model and the personalised graph prompts to enable the prompt-tuning in the target domain through multi-task learning.

items that share its same attributes in the target domain, i.e. the set of items with the same attributes. For example, in this paper, we refer to the use of relevant items according to three available attributes in the metadata (i.e. also_bought, also_viewed and bought_together) in the used Amazon datasets. However, such neighbouring-items are not necessarily fixed and can be naturally changed or extended depending on the used datasets. Next, we consider such neighbouring-items as adjacent nodes to a target user node. For example, consider a user u_0 that has interacted with items i_0 and i_1. Other users that interacted with item i_0 *also_bought* items i_2 and i_3. Then, items i_2 and i_3 will be considered as the adjacent nodes to the user node u_0. Afterwards, we use such neighbouring-items as a type of personalised graph prompts and call them as 'hard graph prompts'. Note that we ignore the users' interacted items as adjacent nodes when devising prompts to avoid the possible over-fitting of the final learned model. In addition, we also introduce a different type of personalised graph prompts, namely 'soft graph prompts', which consists of a pre-defined number of randomly initialised embedding vectors. Overall, we define both the hard graph prompts and the soft graph prompts as the personalised graph prompts of a given user. Finally, such personalised graph prompts will be leveraged to assist the graph encoder that is pre-trained from the source domain to improve the recommendation efficiency and effectiveness. In particular, to guide the pre-trained model in learning richer user/item representations, we embrace the use of the contrastive learning technique in the pre-training stage. To further enhance the model tuning in the target domain, we combine the contrastive learning loss and the BPR loss in a multi-task manner. In next section, we introduce the rationale and motivation for leveraging the personalised graph prompts in the tuning stage.

3.3 Personalised Graph Prompts

In this section, we first justify the equivalence between Graph Neural Networks and the transformer to explain the rationale of introducing our personalised graph prompts. Then, we further describe our proposed graph prompts and illustrate them with GNN recommender models. Furthermore, we explain the rationale behind the incorporation of graph prompts in cross-domain recommendation, emphasising the benefits and implications of this approach within the context of recommendation systems.

3.3.1 Comparing a Transformer and GNNs. **Transformer:** The transformer architecture consists of a composition of transformer layers [48]. Each transformer layer has two parts: a self-attention module and a position-wise feed-forward network (FFN). Let h^ℓ denote the input of the self-attention module on a certain layer. The propagation rule of the transformer updates the hidden feature h at position i of a sentence S from layer ℓ to layer $\ell+1$ as follows:

$$h_i^{\ell+1} = \sum_{j \in \mathcal{S}} w_{ij} \left(W_V^\ell h_j^\ell \right) \tag{1}$$

$$w_{ij} = \text{softmax}_j \left(W_Q^\ell h_i^\ell \cdot W_K^\ell h_j^\ell \right) \tag{2}$$

where $j \in S$ denotes the set of words in the sentence and W_Q, W_K, W_V are learned linear weights. In natural language processing, the transformer sums over all the words in a sentence and outputs the next hidden feature by applying a weighted summation on the values.
Graph Neural Network: Modern GNNs [19, 49] follow a learning schema that iteratively updates the representation of a node by aggregating the representations of its first or higher-order neighbours. Let h^ℓ denote the representation of a node v_i at the l-th layer. GNNs update the hidden features h of node i at layer ℓ via a non-linear transformation of the node's own features added to the aggregation of features from each neighbouring node $j \in N(i)$:

$$h_i^{\ell+1} = \sigma \left(W_U^\ell h_i^\ell + \sum_{j \in \mathcal{N}(i)} w_{ij} \left(W_V^\ell h_j^\ell \right) \right) \tag{3}$$

where W_U, W_V are learned weights of the GNN layer and σ is a non-linear transformation. GNNs exploit high-order connectivity by summing over the feature vectors h_j^ℓ from the adjacent nodes with w_{ij} set to 1 and output the next hidden feature. Given w_{ij} is a learned weight that is dependent on nodes i and j, the GNN layer can be interpreted as a layer in a single head-based GAT [49] and is similar to the calculation of the hidden features in a transformer layer (see Equation (1)).

To conclude, a transformer sums over all words in a sentence and a GNN sums over the local neighbourhood. Moreover, a transformer uses self-attention to have a weighted sum for feature transformation since self-attention can be seen as the inference of a soft adjacency matrix. Indeed, compared to the transformer architecture, a GNN can be considered as a simple transformer that applies a linear-weighted sum on a randomly permuted sentence, since the neighbouring nodes compose a 'sentence' and the 'words' are in random order without the positional embeddings. Inversely, the transformers can also be viewed as a special case of GNNs on a fully connected graph of words [17].

3.3.2 Prompt-tuning in a Transformer and GNNs. Equation (1) and Equation (2) describe the dot-product attention mechanism, which provides a weighted summation over the words in a sentence. Here, we derive an equivalent form of Equation (1) as follows:

$$\begin{aligned} h_i^{\ell+1} &= \text{Attn} \left(h_i^\ell W_Q, h_j^\ell W_K, h_j^\ell W_V \right) \\ &= \text{Attn} \left(Q W_Q, K W_K, V W_V \right) \end{aligned} \tag{4}$$

where W_Q, W_K, W_V are learned weights on the layer input Q and the key-value pairs.

The mechanism of prompt-tuning changes the attention module through prepending the tunable vectors to the original attention keys and values at every layer [11, 21]. Specifically, two sets of prefix vectors $P_K, P_V \in \mathbb{R}^{l \times d}$ are concatenated with the original attention key K and value V. Here, we provide an alternative view of prompt-tuning in a transformer:

$$\begin{aligned} h_i^{\ell+1} &= \text{Attn}\left(QW_Q, (P_K \parallel KW_K), (P_V \parallel VW_V)\right) \\ &= \text{softmax}\left(QW_Q \left(P_K \parallel KW_K\right)^\top \begin{bmatrix} P_V \\ VW_V \end{bmatrix}\right) \\ &= (1-\lambda)\,\text{softmax}\left(QW_Q W_K^\top K^\top\right) VW_V + \lambda\,\text{softmax}\left(QW_Q P_K^\top\right) P_V \\ &= (1-\lambda)\,\text{Attn}\left(QW_Q, KW_K, VW_V\right) + \lambda\,\text{Attn}\left(QW_Q, P_K, P_V\right) \end{aligned} \tag{5}$$

where $\parallel$ is the concatenation operator, and λ is a scalar that represents the sum of normalised attention weights on the prefixes:

$$\lambda = \frac{\sum_i \exp\left(QW_Q P_K^\top\right)_i}{\sum_i \exp\left(QW_Q P_K^\top\right)_i + \sum_j \exp\left(QW_Q W_K^\top K^\top\right)_j} \tag{6}$$

Note that the first term in Equation (5), $\text{Attn}\left(QW_Q, KW_K, VW_V\right)$, is the original attention without prefixes, whereas the second term is the attention modification term, which changes the attention module through linear interpolation on the original output of a transformer:

$$h_i^{\ell+1} \leftarrow (1-\lambda)h_i^{\ell+1} + \lambda \Delta h_j^{\ell+1}, \quad \Delta h_j^{\ell+1} = \text{softmax}\left(QW_Q P_K^\top\right) P_V \tag{7}$$

Inspired by the success of prompt-tuning methods, we prepend the mixture node embeddings to be the item-wise graph prompts in the CDR scenario. Specifically, we append such node embeddings as prompts to the output of the pre-trained GNNs to update the learned features of nodes. These prompts further incorporate additional items or nodes, referred to as neighbouring-items or continuous learned vectors discussed in Section 1, for a target user. As a consequence, each user node obtains a specific number of m prompts. Next, we update the embeddings of the items using their respective adjacent user nodes. When we apply the prompt-tuning to GNNs, we can obtain an updated propagation rule on top of Equation (3):

$$h_i^{\ell+1} = \sigma\left(W_U h_i^\ell + (1-\lambda)\sum_{j \in N_{(i)}} (W_V h_j^\ell) + \lambda \sum_{r \in N_{(i)}} (P_V{}' h_r^\ell)\right) \tag{8}$$

where r is a neighbouring-items node such as $r \in N(i)$. r is used to act as a new adjacency node of the given node, thereby creating a new edge connection in the graph; $\lambda = \frac{|N_r|}{|N_j|+|N_r|}$ is an item-related weight factor. $|N_j|$ and $|N_r|$ denote the first-hop neighbours item j and the neighbouring-item r, respectively. Similar to W_V, $P_V^{'}$ is a learned weight matrix of the GNN layer. It is worth noting that during the tuning stage, the second term of Equation (8), without $(1-\lambda)$, is the output from a pre-trained graph encoder with frozen parameters. Furthermore, the third term of Equation (8), specifically the learned matrix $P_V^{'}$, is the only learned matrix within the context of the target domain D_T. As a result, tuning only on a small set of parameters is expected to improve efficiency in the tuning stage. The number of tuning parameters of the graph encoder is determined by considering the learned parameters associated with the number of both hard and soft prompt nodes. These nodes are aggregated as supplementary nodes within the graph encoder, and their combined parameters hence contribute to the total number of the learned parameters. As such, we argue that the resulting GNN model using the personalised graph prompts is capable of efficiently leveraging the useful knowledge from the pre-trained GNN models, while also reducing the tuning parameters required during the tuning stage.

3.3.3 Hard Prompts and Soft Prompts. In this section, we provide a comprehensive overview of both hard and soft prompts in our PGPRec framework, detailing their selection, initialisation, and integration within the framework. **Hard Prompts.** Informative personalised prompts necessitate the inference of correlations among items. Indeed, the implementation of a straightforward yet efficient approach for determining item correlations is essential for generating personalised hard prompts that enable sufficient learning within a target domain. Following [26], we introduce a model-based method to infer correlations between the interacted items and their corresponding neighbouring items within the target domain. Liu et al. [26] suggested that a model-based correlation measurement is more desirable than alternative methods such as relying solely on the user frequency counts of the given items. Since the item representations are jointly learned with the graph encoder during the prompt-tuning phase, this correlation score is inherently model-based. In this work, we use the dot-product to measure the correlation between items. Let $\mathbf{e_j}$ be the representations of all the $\mathbf{j}$ items that a given user has interacted with, and similarly $\mathbf{e_r}$ denotes the representations of all of the neighbouring-items $\mathbf{r}$ in the target domain. It follows that the model-based correlation score can be defined as follows:

$$\mathrm{Cor_e}(\mathbf{j}, \mathbf{r}) = \mathbf{e_j} \cdot \mathbf{e_r} \tag{9}$$

In our proposed PGPRec framework, we select the top-m items from the neighbouring-items $\mathbf{r}$, determined by the correlation score, as hard prompts where m represents the number of hard prompts integrated within the framework. The hard prompts, which originate from the actual nodes within the graph and which are updated independently, directly affect the total number of parameters. As mentioned in Section 3.3.2, the number of these hard prompts is pre-determined based on the top-m correlation score, leading to a consistent parameter scale. This selection approach ensures that the most relevant and informative neighbouring items are used for generating the hard prompts, thereby enhancing the overall recommendation performance in the target domain. **Soft Prompts.** Different from the hard prompts, the soft prompts adopt a flexible and adaptive approach to capture the underlying item relationships within the target domain. In particular, the soft prompts are randomly initialised and subsequently embedded into trainable vectors. These soft prompts serve as auxiliary nodes that establish connections with the target users within the target domain. This process facilitates the collection of contextual information from the user-item interactions, thereby enabling our PGPRec framework to generate more effective user/item representations. As a consequence, the number of parameters increases proportionally to an increase in the number of soft prompts. Furthermore, since the soft prompts are trainable, they can adapt to the data during the training process, thereby aligning the user/item embeddings with the underlying structure of the target domain data. As such, this adaptability results in smoother user/item representations, since the soft prompts can better capture more fine-grained user/item embeddings compared to the hard prompts.

3.3.4 Why Prompt-tuning in GNNs? Applying prompt-tuning in GNNs offers several potential advantages for cross-domain recommendation tasks, such as improved tuning efficiency, the provision of additional collaborative signals for the target domains, and an easier deployment in large-scale graphs. **(1) Tuning efficiency:** Tuning a smaller set of parameters within GNNs using both hard and soft prompts is essential, as it allows for the customisation of the model, enabling it to capture the unique characteristics specific to the target domain. As illustrated in Equation (8), prompt-tuning in GNNs leverages pre-trained graph encoders and concentrates on a limited set of parameters by tuning solely with the personalised prompts. Subsequently, the obtained embedding is combined with that generated by the pre-trained graph encoder, therefore leveraging the previously pre-trained knowledge while maintaining the model efficiency. **(2) Additional collaborative signals:** Prompt-tuning in GNNs offers a flexible solution capable of incorporating supplementary collaborative signals for the target domain [7, 46]. In our proposed PGPRec framework, the personalised graph prompts, which include both the hard and soft prompts, can indeed serve as auxiliary side information for the target domain. These prompts can be tailored to enhance the recommendation performance in specific cases, thereby likely improving the accuracy and effectiveness of the cross-domain recommender system. **(3) Easier deployment:** Applying personalised graph prompts on adequately pre-trained GNNs leads to less parameters to store [46], rendering it a more convincing option for large-scale graph applications where the training of GNNs necessitates significant computational resources and memory. In our proposed PGPRec framework, by freezing the GNNs once they are pre-trained and tuning the graph prompts in an efficient and scalable manner, as described in Section 3.3.2, PGPRec can potentially be deployed for various applications without the need for extensive training.

3.4 Contrastive Learning in PGPRec

Unlike the existing approaches, we adopt the Contrastive Learning (CL) scheme [34] to ensure an effective training in both the pre-training and fine-tuning phases. In the pre-training stage, we employ contrastive learning to optimise the graph encoder with the data from the source domain D_S. Specifically, the contrastive pre-training stage comprises three main components: 1) a graph augmentation tool, which builds distinct sub-graphs for generating the representation variants of identical nodes, 2) a graph encoder to model the developed sub-graphs, and 3) a corresponding contrastive loss function for the graph encoder optimisation. On the other hand, to further enhance the recommendation performance in the tuning stage, we also leverage the contrastive learning technique and introduce a joint-learning loss that comprises both the contrastive loss and the commonly used BPR [38] loss. To better illustrate the training process of PGPRec, Algorithm 1 provides the training pseudo-code.

3.4.1 *Contrastive Pre-training of PGPRec.*

We now describe the contrastive learning component in our PGPRec framework. Following [54], we generate two augmented sub-graphs via the edge dropout technique and feed them into the graph encoder f. As mentioned in Section 3.2, PGPRec is a model-agnostic framework that is flexible to accommodate the commonly used GNN models. In this paper, we adopt a special case of LightGCN [4] [12] as the graph encoder, which is a simple yet efficient GNN recommender [31]. Specifically, we use the edge dropout augmentation with the best reported performance in SGL [54] which is an effective graph contrastive learning method. Then, we can obtain two augmented sub-graphs as follows:

$$\mathcal{G}_{S'} = (\mathcal{V}, M \odot \mathcal{E}), \quad \mathcal{G}_{S''} = (\mathcal{V}, M' \odot \mathcal{E}) \tag{10}$$

where $\mathcal{V}$ is the node set, while M and M' are two different masking vectors on the edge set $\mathcal{E}$.

By using the augmented sub-graphs and a graph encoder, we can generate different embedding representations of an identical node (i.e. positive pair). For example, consider a given node q. We can obtain its embeddings e_q and $e_{q'}$ after encoding two sub-graphs with the graph encoder f. Then, to obtain a negative pair of sub-graphs, we apply an in-batch random sampling strategy (e.g. a pair of node representations e_q and e_k of two different nodes q and k). By contrasting the positive and negative pairs, we expect that the resulting user/item representations of users/items can effectively improve the recommendation performance when applied to the target domain. As such, the learned feature embeddings from the source domain allow the model to start from a better initialisation point and provide promising results after a light fine-tuning [10] in various target domains.

Next, after obtaining the positive and negative pairs, we follow SimCLR [3] to generate a better representation via data augmentations and adopt the contrastive loss, InfoNCE [34], to maximise the agreement of the positive pairs and minimise that of the negative pairs:

$$\mathcal{L}_{cl}^{user} = -\log \frac{\exp\left(\mathbf{e}_q^\top \mathbf{e}_{q'}/\boldsymbol{\tau}\right)}{\sum_{i=1}^{n} \exp\left(\mathbf{e}_q^\top \mathbf{e}_k/\boldsymbol{\tau}\right)} \tag{11}$$

[4] Since the only trainable model parameters in LightGCN are the user/item embeddings at the 0-th layer, LightGCN cannot transfer the structure knowledge with the feature transformation matrices from the source domain. Hence, following [59], we adopt LightGCN with the attention aggregator to explicitly learn transferable knowledge.

where τ is a hyper-parameter that adjusts the dynamic range of the resulting loss value. Analogously, we obtain the contrastive loss of the item side $\mathcal{L}_{cl}^{item}$. Combining these two losses, we obtain an objective function for the contrastive task as follows:

$$\mathcal{L}_{cl} = \mathcal{L}_{cl}^{user} + \mathcal{L}_{cl}^{item} \tag{12}$$

3.4.2 Contrastive Tuning of PGPRec. Once we obtain the pre-trained GNN model from the source domain, we transfer the learned model to the target domain. Recall the discussion in Section 3.2 where we introduced two variants of the personalised graph prompt, namely the soft and hard graph prompts in order to improve the recommendation efficiency and effectiveness in a CDR setting. Differently from the pre-training phase, we optimise both a pairwise ranking task objective and a contrastive learning objective $\mathcal{L}_{cl}$ to further integrate the prediction signals during the tuning phase:

$$\begin{aligned} \mathcal{L} =& \mathcal{L}_{rec} + \lambda_1 \mathcal{L}_{cl} + \lambda_2 \left\|\Theta\right\|_2^2 \\ \text{where } \mathcal{L}_{rec} =& \sum_{(u,i,j)\in D_s} -\log\sigma(\boldsymbol{e_u}^\top(\boldsymbol{e_i} - \boldsymbol{e_j})) \end{aligned} \tag{13}$$

where $\mathcal{L}_{rec}$ is the Bayesian Personalised Ranking (BPR) loss [38], $\boldsymbol{e_u}$ is the user embedding, $\boldsymbol{e_i}$ denotes the positive item embedding and y_{ui} is the ground truth value, $D_s = \{(u,i,j)|(u,i) \in R^+, (u,j) \in R^-\}$ is the set of the training data, R^+ indicates the interacted user-item pairs and R^- indicates user-item pairs from the users' unseen items. Moreover, $\sigma(\cdot)$ is the sigmoid function, Θ is the set of model parameters in $\mathcal{L}_{rec}$ while λ_1 and λ_2 are hyper-parameters to control the strengths of the CL and L_2 regularisation, respectively.

4 EXPERIMENTS

To demonstrate the efficiency of PGPRec and provide evidence for its effectiveness , we conduct experiments to answer the following four research questions:
RQ1: How do the PGPRec framework perform in cross-domain recommendation compared with existing baselines?
RQ2: How do different hard graph prompts and soft graph prompts and their combination impact the recommendation performance?
RQ3: Are the personalised graph prompts more efficient than the fine-tuning in cross-domain recommendation?
RQ4: Are the personalised graph prompts effective in a cold-start scenario?

4.1 Experimental Setup

4.1.1 Datasets. To evaluate the effectiveness of our PGPRec framework, we conduct experiments on four *Amazon Review datasets* [5], namely Electronics (Elec for short), Cell Phones (Phone for short), Accessories, Sports and Outdoors (Sport for short) & Clothing Shoes and Jewelry (Cloth for short). Since the Amazon Review datasets include information from items in different domains, they are suitable for evaluating cross-domain recommendation [52]. In particular, these datasets are considered into 4 pairs of source-target datasets as shown in Table 2. Each pair of datasets shares the common users between the source and target datasets. The exact statistics of the used pairs of datasets are listed in Table 2. By comparing the statistics across all datasets in Table 2, it is clear that each source-target dataset pair exhibits a similar density level. When processing the datasets, we transform the ratings into 1 or 0, indicating whether the user has rated the item or not. Following [23], we denote those users that have more than 5 interactions in a given dataset as the regular users while the cold-start user are those users with less than five ratings.

[5] https://jmcauley.ucsd.edu/data/amazon/

Algorithm 1: The overall training process of the PGPRec framework

```
Input:
        The user-item interaction graphs $\mathcal{G}_S$ and $\mathcal{G}_T$;
        The batch size $B$;
        The hyper-parameters $\lambda_1, \lambda_2$;
        The ID representations of neighbouring-item $\boldsymbol{e}_{\boldsymbol{p}_h}$ as hard prompts;
Output:
        The final user/item embeddings $e_u$, $e_i$;
1  Initialise epoch $t = 0$;
2  Initialise the model parameters $\Theta$ with the default Xavier distribution;
3  // Pre-training parameters in the source domain $\mathcal{D}_S$
4  repeat
5      $t = t + 1$;
6      Obtain the original ID representations $\boldsymbol{e}_{\boldsymbol{u}}, \boldsymbol{e}_{\boldsymbol{i}}$ for each user $u$ and item $i$ based on $\mathcal{G}_S$ ;
7      Obtain the sub-graphs $\mathcal{G}_{S'}$ and $\mathcal{G}_{S''}$ with a graph perturbation on $\mathcal{G}_S$ with Eq.(10);
8      Obtain the perturbed user/item representations $e_q$, $e_{q'}$ based on sub-graphs $\mathcal{G}_{S'}$ and $\mathcal{G}_{S''}$ with Eq.(3) ;
9      Calculate the InfoNCE losses $\mathcal{L}_{cl}^{user}$ and $\mathcal{L}_{cl}^{item}$ with $e_q$ and $e_{q'}$, according to Eq.(11);
10     Calculate the combined InfoNCE loss $\mathcal{L}_{cl}$ with Eq.(12) ;
11     Backpropagation and update model parameters $\Theta$ with $\mathcal{L}$;
12 until convergence;
13 // Tuning parameters in the target domain $\mathcal{D}_T$
14 Initialise epoch $t = 0$;
15 repeat
16     $t = t + 1$;
17     Obtain the original ID representations $\boldsymbol{e}_{\boldsymbol{u}}, \boldsymbol{e}_{\boldsymbol{i}}$ for each user $u$ and item $i$ based on $\mathcal{G}_T$ ;
18     Acquire the hard prompt representations $\boldsymbol{e}_{\boldsymbol{p}_h}$ through neighbouring-items;
19     Acquire the soft prompt representations $\boldsymbol{e}_{\boldsymbol{p}_s}$ with random initialised embedding vectors;
20     Encode the user/item representations $\boldsymbol{e}_{\boldsymbol{u}}, \boldsymbol{e}_{\boldsymbol{i}}$ through the pre-trained model parameters $\Theta$ with Eq. (3);
21     Combine the embeddings of $\boldsymbol{e}_{\boldsymbol{p}_h}$ and $\boldsymbol{e}_{\boldsymbol{p}_s}$ into the user/item representations $\boldsymbol{e}_{\boldsymbol{u}}, \boldsymbol{e}_{\boldsymbol{i}}$ with Eq. (8);
22     Calculate the BPR loss $\mathcal{L}_{BPR}$ with Eq. (13);
23     Obtain the perturbed user/item representations $e_q$, $e_{q'}$ based on sub-graphs $\mathcal{G}_{T'}$ and $\mathcal{G}_{T''}$ with Eq. (10)
        ;
24     Calculate the InfoNCE loss $\mathcal{L}_{cl}$ for contrastive learning with $e_q$ and $e_{q'}$, according to Eq. (11);
25     Calculate the joint learning loss $\mathcal{L}$ with Eq. (13) ;
26     Backpropagation and update the model parameters $\Theta$ with $\mathcal{L}$;
27 until convergence;
28 return the final user/item embeddings $e_u$, $e_i$ in the target domain $\mathcal{D}_T$
```

4.1.2 Evaluation Metrics. In this work, we adopt two widely used evaluation metrics to evaluate the performance of our PGPRec framework as well as the used baselines. Specifically, suppose there is a set of items to be ranked. Given a user u, let $\hat{R}(u)$ represent a ranked list of items that an algorithm produces, and let $R(u)$ represent a

Table 2. Statistics of the used Amazon Review datasets. The column 'Users' donates the number of overlapping users and the column 'Cold-start users' donates the number of cold-start users in the test set.

Dataset	Users	Cold-start users	Items	Interactions	Density
Elec	12,739	2,014	36,183	465,545	0.101%
Phone	12,739	2,014	12,772	178,973	0.110%
Sports	6,755	893	31,514	93,666	0.044%
Cloth	6,755	893	35,131	87,805	0.037%
Sports	4,017	617	19,396	52,202	0.067%
Phone	4,017	617	12,517	40,225	0.080%
Elec	11,611	1,803	49,730	202,095	0.035%
Cloth	11,611	1,803	47,265	137,198	0.025%

ground-truth set of items that user u has interacted with. For the top-k recommendation task, only the top-ranked items are important to consider. Therefore, we list the used metrics below:

- Recall at top-k positions: Recall is a metric for computing the fraction of relevant items out of all relevant items and is defined as follows:

$$\text{Recall@k} = \frac{1}{|U|}\sum_{u\in U}\frac{|\hat{R}(u)\cap R(u)|}{|R(u)|}, \tag{14}$$

  where $\hat{R}(u)$ is the ranked list of items generated by the recommenders and $|\hat{R}(u)|$ is the size of the ranked list $\hat{R}(u)$, $R(u)$ is the ground-truth set of items that user u has interacted with and $|R(u)|$ represents the size of the item set $R(u)$, U denotes the user set with size $|U|$. Here, $|\hat{R}(u)| = k$.
- Normalised Discounted Cumulative Gain at top-k positions: Normalised Discounted Cumulative Gain (NDCG) is a metric devried from Discounted Cumulative Gain (DCG) [15], which takes the positions of the correct recommended items into consideration, where the positions are discounted logarithmically [39]. NDCG accounts for the position of the hit by assigning higher scores to hits at the top ranks [13] and is defined as follows:

$$\text{NDCG@k} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{Z}\sum_{x=1}^{|\hat{R}(u)|}\frac{2^{I(\hat{R}_x(u)\in R(u))}-1}{\log_2(x+1)} \tag{15}$$

  where $\hat{R}_x(u)$ denotes for the item recommended at the x-th position, and Z is a normalised factor, which denotes the ideal value of DCG given $R(u)$.

We measure the recommendation effectiveness on four Amazon Review datasets described in Section 4.1.1 in terms of Recall and NDCG calculated to rank depth 10. To evaluate parameter-efficiency, we count the parameters associated with m graph prompts h_r and the weighted matrix P_V, as defined in Equation (8), and compare these to the parameter counts of the pre-trained GNN model. Additionally, we evaluate the time-efficiency by measuring the total tuning time, the time per epoch, and the number of epochs needed for convergence as per early-stopping criteria. Moreover, we follow the experimental setup in [4] and randomly split a given dataset into training, validation, and testing sets with an 8:1:1 ratio. For statistical significance comparisons with the baselines, we use the two one-sided equivalence test ($p < 0.05$) and apply the Holm–Bonferroni multiple testing correction, as per best practices in information retrieval [40]. Moreover, we calculate the used metrics based on the generated ranking list of items and report the average score over all test users.

4.1.3 Baselines. To evaluate the effectiveness of our proposed approach, we compare PGPRec with the following existing state-of-the-art baselines. The baselines can be roughly categorised into four groups: (1) NGCF, LightGCN and SGL are single domain collaborative filtering methods, (2) CMF, CoNet and PPGN are joint-learning methods for cross-domain recommendation, (3) NGCF, LightGCN, and SGL are graph pre-training and fine-tuning methods, (4) our PGPRec is a graph pre-training and prompt-tuning method. Table 3 summarises the baselines as well as PGPRec across different aspects, namely single-domain, joint-learning, and pre-training followed by fine-tuning. Below, we provide a brief description of all the used baselines.

CMF [47]: This is a joint-learning approach, which factorises matrices of multiple domains simultaneously by sharing the user latent factors. CMF first jointly learns on two domains and then optimises the target domain.

CDMF [28]: This is also a joint-learning approach. Different from CMF, CDMF leverages factorisation machines to model interactions between users and items across different domains. CDMF learns shared latent factors across these domains by representing user-item interactions as feature vectors and applying factorisation machines to model their relationships. Such an approach enables CDMF to capture more complex interaction patterns between users and items, as well as to share latent factors in cross-domain recommendations.

CoNet [14]: This is also a joint-learning method, which transfers knowledge across domains by leveraging cross-connections between the feed-forward neural networks. Different from CMF, CoNet optimises both source and target domains with a joint-learning objective.

PPGN [65]: PPGN is a joint-learning graph method, which fuses the interaction information of the two domains into a graph, and shares the features of users learned from the joint interaction graph by stacking multiple graph convolution layers. Finally, it inputs the learned embeddings to the domain-specific MLP structure to learn the matching function.

NGCF [51]: NGCF is a single-domain GCN-based model. It first captures the high-order connectivity information in the embedding function by stacking multiple embedding propagation layers. Next, it concatenates the obtained embeddings and uses the inner product to make predictions. In order to examine the effectiveness of single domain GNNs as well as the fine-tuning of GNNs in Cross-Domain Recommendation (CDR), we perform two different training strategies with NGCF: (1) single domain training where we directly train in the target domain (denoted by NGCF$_{SD}$) – this is the typical training strategy in single-domain recommenders, and (2) pre-training in a source-domain then fine-tuning in the target domain (denoted by NGCF$_{PT}$).

LightGCN [12]: This is also a single-domain GCN-based model that omits learned weight matrices from NGCF. It simplifies the design in the feature propagation component by removing the non-linear activation and the transformation matrices. Similar to NGCF, we also perform both the single domain training (denoted by LightGCN$_{SD}$) and the "pre-training then fine-tuning" process (denoted by LightGCN$_{PT}$).

SGL [54]: This is a single-domain CL-based method. With LightGCN as the encoder of the users/items, it adopts different augmentation operators such as edge dropout and node dropout, on the pre-existing features of the user/items. Similar to NGCF and LightGCN, we also perform both the single domain training (denoted by SGL$_{SD}$) and the "pre-training then fine-tuning" process (denoted by SGL$_{PT}$) but using an auxiliary contrastive loss as described in Section 3.4.1.

BiTGCF [23]: This joint-learning graph approach extends LightGCN for cross-domain recommendation tasks by employing dual linear graph encoders to generate user and item representations in each domain. Subsequently, a feature transfer layer is used to fuse user representations across domains, effectively capturing the underlying relationships and facilitating enhanced recommendations in the target domain.

PGPRec-BPR: This is a variant of PGPRec. Differently from PGPRec, which pre-trains with a contrastive loss, PGPRec-BPR uses a BPR loss to optimise the GNN model. Moreover, it tunes with the same joint loss (Equation (13)) as PGPRec. It is a pre-training and prompt-tuning method. Hence, PGPRec-BPR allows us to gauge the added-value of leveraging contrastive learning.

Table 3. Summary of compared approaches across different aspects.

Method	CMF	CDMF	CoNet	PPGN	BiTGCF	NGCF	LightGCN	SGL	PGPRec
Single-domain	×	×	×	×	×	✓	✓	✓	×
Joint-learning	✓	✓	✓	✓	✓	×	×	×	×
Pre-training & Fine-tuning	×	×	×	×	×	✓	✓	✓	×
Pre-training & Prompt-tuning	×	×	×	×	×	×	×	×	✓

4.1.4 Implementation Details and Hyper-parameter Settings. For a fair comparison between our PGPRec framework and the used baselines, we conduct all experiments on the same machine with a GeForce RTX 2080Ti GPU. Moreover, we use the learned parameters at the pre-training stage to initialise the model parameters at the tuning stage. In the pre-training stage, the dropout rate ρ of nodes and edges are set to 0.1 and the softmax temperature τ in the contrastive learning loss is set to 0.2, which are reported with the best performance of the original SGL paper [54]. Furthermore, in the tuning stage, we tune the hyper-parameters of our PGPRec framework on the validation set. The learning rate is selected from $\{10^{-2}, 10^{-3}, 10^{-4}\}$. For those hyper-parameters unique to PGPRec, we tune λ_1, λ_2, τ and ρ within the ranges of $\{0, 0.1, 0.2, ..., 1.0\}$, $\{0, 0.1, 0.2, ..., 1.0\}$, $\{0, 0.1, 0.2, ..., 1.0\}$ and $\{0, 0.1, 0.2, ..., 0.9\}$, respectively. We adopt two widely used evaluation metrics, namely Recall@K and NDCG@K to evaluate the performance of top-K recommendations, which were described in Section 4.1.2. We follow [44] and set K = 10 and report the average performance achieved for all users in the test set. The negative items of each user are defined as those having no interactions with the user. Following [12], the dimensions of user and item embedding are set to 64 to achieve a trade-off between performance and time cost, which is determined by a grid search in the range of $\{16, 32, 64, 128, 256\}$. We tune NGCF$_{PT}$, LightGCN$_{PT}$, SGL$_{PT}$ and PGPRec-BPR as described in Section 4.1.3 on the validation set. For the other cross-domain recommendation baselines corresponding to joint-learning approaches (CMF, CDMF, CoNet, PPGN, BiTGCF), we follow the reported optimal parameter settings by the authors of these baselines. For our PGPRec framework, we tune the hyper-parameters as described above on the validation set. Moreover, we adopt the Xavier initialisation to initialise all the models' parameters and use the Adam optimiser for the model optimisation with a batch size of 1024. We apply early-stopping during training, terminating the training when the validation loss does not decrease for 50 epochs.

4.2 PGPRec Effectiveness Evaluation (RQ1)

Table 4 reports the empirical results of our PGPRec method in comparison to all of the baselines, which were described in Section 4.1.3. We evaluate our PGPRec framework in comparison to three distinct recommendation approaches: GNN-based recommenders, single-domain recommenders and joint-learning recommenders. Specifically, we compare our PGPRec framework to the GNN-based recommenders (NGCF, LightGCN, SGL), which are trained in both the single domain (SD for short) setting as well as in the pre-training (PT for short) setting. In addition, we compare our PGPRec framework to the joint-learning methods (CMF, CDMF, CoNet, PPGN, BiTGCF). Among the evaluated baselines, LightGCN$_{SD}$ generally outperforms the joint-learning CDR methods, with the exception of BiTGCF. This observation highlights the importance of investigating the nonlinear interaction relationship between users and items through graph neural networks, as mentioned in Section 4.1.3. Notably, BiTGCF is a tailored approach that builds upon LightGCN. The comparison between LightGCN$_{SD}$ and BiTGCF further supports the effectiveness of the GNN-based methods in capturing complex user-item relationships within the cross-domain recommendation context. Furthermore, when comparing SGL$_{PT}$ with NGCF and LightGCN, as well as their single-domain (NGCF$_{SD}$, LightGCN$_{SD}$) and pre-trained variants (NGCF$_{PT}$, LightGCN$_{PT}$), SGL$_{PT}$ exhibits a superior performance in 88% of the cases (28 out of 32 instances), with a significant difference. This result emphasises the advantages of employing contrastive learning during both the pre-training and tuning

Table 4. Experimental results for PGPRec and the used baselines. The best performance is highlighted in bold and the second best result is highlighted with an underline. * and $^\triangle$ mean p <0.05 in the t-test and TOST test (AP=0.05) compared to the result of PGPRec with the Holm–Bonferroni correction. SD/PT are the abbreviations for Single-Domain and Pre-Training, respectively.

Dataset	Elec-Phone		Sport-Cloth		Sport-Phone		Elec-Cloth	
Methods	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
CMF	0.4015*	0.2543*	0.4309*	0.2685*	0.3348*	0.1927*	0.3054*	0.1836*
CDMF	0.4457*	0.2612*	0.4561*	0.2874*	0.3808*	0.2242*	0.3524*	0.2356*
CoNet	0.4514*	0.2696*	0.4912*	0.3128*	0.3854*	0.2331*	0.3546*	0.2353*
PPGN	0.4464*	0.2629*	0.4678*	0.2930*	0.3677*	0.2240*	0.3404*	0.2184
BiTGCF	0.5064*	0.2999*	0.5360*	$\underline{0.3315}^\triangle$	$\underline{0.4688}^*$	0.3072*	0.4209	0.2783*
NGCF$_{SD}$	0.4501*	0.2643*	0.4889*	0.3027*	0.3955*	0.2431*	0.3878*	0.2483*
LightGCN$_{SD}$	0.4776*	0.2993*	0.5327*	**0.3358**$^\triangle$	0.4557*	0.3006*	$\underline{0.4261}$	0.2699*
SGL$_{SD}$	0.4722*	0.2941*	0.5296*	0.3219*	0.4486*	0.2749*	0.4012*	0.2569*
NGCF$_{PT}$	0.4665*	0.2969*	0.4983*	0.3253$^\triangle$	0.4251*	0.2681*	0.3723*	0.2359*
LightGCN$_{PT}$	0.4821*	0.3044*	0.5196*	0.3268$^\triangle$	0.4543*	0.2986*	0.4137	0.2727*
SGL$_{PT}$	$\underline{0.5071}$	$\underline{0.3282}^\triangle$	$\underline{0.5398}$	0.3119*	**0.4721**	**0.3114**$^\triangle$	**0.4301**$^\triangle$	**0.2967***
PGPRec	**0.5213**	**0.3386**	**0.5678**	0.3281	0.4607	$\underline{0.3086}$	0.4233	$\underline{0.2854}$
%Diff.	2.80%	3.17%	3.27%	- 2.29%	- 2.41%	- 0.10%	- 1.58%	- 3.80%

stages, since it facilitates the generation of richer user representations, ultimately enhancing the recommendation performance. Comparing PGPRec and SGL$_{PT}$, we observe that PGPRec is competitive and comparable with SGL$_{PT}$ on all datasets. In fact, PGPRec has even a better performance in 75% of the cases (3 out of 4 instances) on the Elec-Phone and Sport-Cloth datasets, thereby demonstrating the general effectiveness of personalised graph prompts. However, PGPRec performs worse than SGL$_{PT}$ on the Sport-Phone & Elec-Cloth datasets. This may be due to PGPRec being more sensitive to differences between domains, such as variations in user-item interaction patterns or preferences, compared to SGL-PT. As a result, such sensitivity may affect the PGPRec's performance when transferring knowledge between distant domains. This observation warrants further investigation to better identify the factors influencing PGPRec's performance in such scenarios. We leave such an investigation to future work.

Hence, in answer to RQ1, we conclude that PGPRec successfully leverages the contrastive learning loss to effectively pre-train a graph encoder and further enhances the recommendation performance by leveraging the personalised graph prompts on two of the four datasets used. As a result, our PGPRec framework successfully enriches the user representations in the target domain, by leveraging the knowledge from the pre-trained model.

4.3 Ablation Study (RQ2)

To investigate the impact of each component of our PGPRec framework, in Table 5, we compare the results of PGPRec to its variants that employ different types of personalised prompts (soft/hard) as well as PGPRec variants with varying pre-training loss functions (BPR/contrastive loss). In particular, we aim to examine the impact of these components on the overall performance of the framework. Table 5 presents the effect of these components on the overall performance of the framework. In particular, we evaluate the statistical significance of the difference in performance between PGPRec and its variants with the paired t-test ($p < 0.05$). We first compare PGPRec to its variants with different types of personalised prompts (PGPRec-soft and PGPRec-hard), which sorely use soft

Table 5. PGPRec performance in terms of Recall@10 and NDCG@10 on the used datasets. The best performance is highlighted in bold and the second best result is highlighted with an underline. * denotes a significant difference compared to the result of PGPRec using the paired t-test with the Holm-Bonferroni correction for $p < 0.05$.

Dataset	Elec-Phone		Sport-Cloth		Sport-Phone		Elec-Cloth	
Methods	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
PGPRec-BPR	0.4851*	0.3006*	0.5332*	0.3149*	0.4361*	0.2593*	0.3918*	0.2603*
PGPRec-soft	0.4716*	0.2845*	0.5216*	0.3063*	0.4468*	0.2610*	0.4033*	0.2662*
PGPRec-hard	0.5172*	0.3244*	0.5532*	0.3193*	0.4537*	0.2856*	0.4231	0.2814*
PGPRec	**0.5213**	**0.3386**	**0.5678**	**0.3281**	**0.4607**	**0.3086**	**0.4233**	**0.2854**

and hard prompts, respectively. From the table, we observe that for all four used datasets, PGPRec significantly outperforms PGPRec-soft and PGPRec-hard in 94% of the cases (15 out of 16 instances). This observation indicates the effectiveness of combining both hard and soft prompts for better user/item representations by transferring the knowledge from a source domain and tailoring the user/item representations to a target domain. On the other hand, comparing the variants PGPRec-soft and PGPRec-hard, we observe a noticeable performance gap between the two, with PGPRec-hard outperforming PGPRec-soft on all datasets and metrics. This finding indicates that the neighbouring-items carry more informative content than the randomly initialised learned vectors. Hence, these hard prompts can better assist user representations in adapting to the target domain. This comparison between PGPRec-soft and PGPRec-hard highlights the importance of leveraging the appropriate item-level information, as provided by hard prompts, to enhance the recommendation performance in cross-domain settings. Furthermore, we also compare PGPRec to PGPRec-BPR, which uses the BPR loss as a training loss function during the pre-training stage while PGPRec employs a contrastive loss when pre-training the graph encoder. From Table 5, we observe that PGPRec significantly outperforms PGPRec-BPR by a large margin across all used datasets, demonstrating the rationality of incorporating contrastive learning to ensure an effective pre-training. The observed enhanced performance highlights the importance of leveraging contrastive learning to extract valuable knowledge from the source domain, and ultimately benefiting the downstream target domain.

To further investigate the impact of different combinations of personalised prompts, Figure 3 shows how the performance of PGPRec changes as we use different numbers and combinations of hard and soft graph prompts. In particular, Figure 3 shows the results of a comprehensive ablation study using different combinations of hard prompt nodes and soft prompt nodes. For ease of illustration and visual clarity, in addition to the performance of PGPRec as we vary the number of used hard and soft prompts, the figure only reports the best-performing combination of soft and hard prompts, which was consistently the same across the used datasets. Note that we use Recall@10 to report the performance of different PGPRec variants, since using NDCG@10 also leads to the same conclusions across the used datasets. In particular, Figure 3 shows that the PGPRec variant (the bar in *orange*) that combines five hard prompt nodes and three soft prompt nodes achieves the best performance in both the Elec-Phone and Elec-Cloth datasets. This promising performance is due to the supplement of the neighbouring-items from the available metadata in the used datasets as well as the use of a continuous vector as a prompt. For the PGPRec variants that only use the hard prompts (bars in *blue*) in Figure 3, the performance of PGPRec improves with more neighbouring-items, which can be viewed as if more adjacent nodes of a target node are added/taken into consideration. However, the improvement becomes marginal when the number of hard prompt nodes increases to over five. In particular, we can observe that the performance of the variant with seven hard prompts is on a par with the variant in *orange* on the Sport-Phone and Elec-Cloth datasets, which

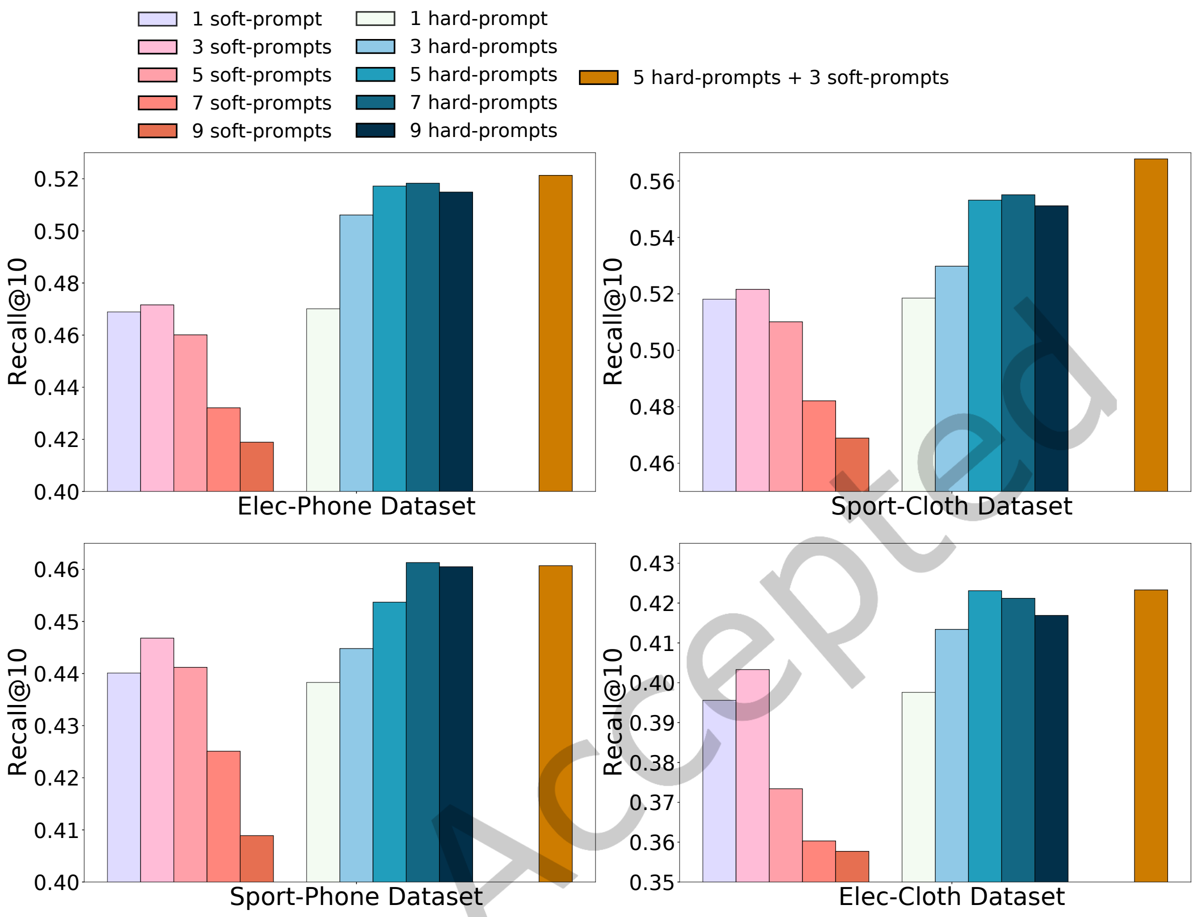

Fig. 3. Performance of the PGPRec variants in terms of Recall@10 on four Amazon Review datasets.

combines both hard and soft prompts. This further supports that the hard neighbouring-items contribute most in the personalised graph prompts while the contribution of the soft prompts is marginal. By comparing the results observed on the Elec-Phone and Elec-Cloth datasets, we find that the best number of hard prompts depends on the scale of the target domain dataset, which indicates that a larger scale target dataset always requires more prompts to facilitate more divergent user/item representations. Moreover, we observe from Figure 3 that the performance of CDR is further enhanced by the addition of soft prompt nodes. One possible reason is that embedding soft prompt nodes provides an additional supervised signal when optimising through contrastive learning. Indeed, through contrastive learning, the collaborative signal is further enhanced by mining graph data in a self-supervised manner. However, as observed in Figure 3, a large number of soft prompt nodes impedes the CDR recommendation performance, which shows the difficulty for the graph model to train a useful user/item embedding from scratch.

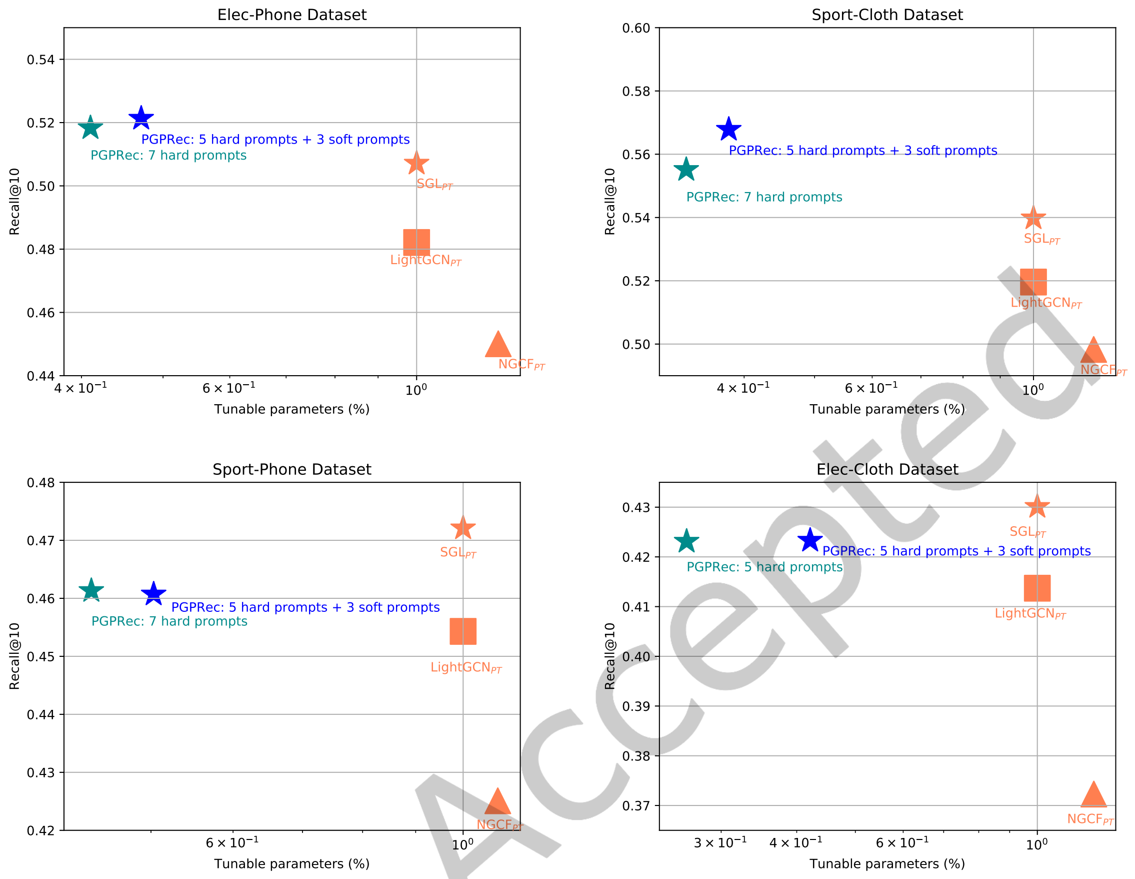

Fig. 4. Performance of different tuning methods on the four used datasets with the pre-trained GNN models

Hence, in answer to RQ2, we conclude that PGPRec successfully leverages hard graph prompts to learn effective user/item representations. It also further enhances performance by optimising the learned soft prompt nodes' embeddings.

4.4 Efficiency of Graph Prompt-tuning (RQ3)

To answer RQ3, we investigate the superiority of personalised graph prompts in terms of parameter efficiency (Section 4.4.1) and tuning efficiency (Section 4.4.2).

4.4.1 Parameter Efficiency. First, we investigate how many tuned parameters can be reduced by using our personalised graph prompt-tuning on all four used datasets. Specifically, we examine the parameter efficiency by comparing the number of tuned parameters of the personalised graph prompts with the number of fine-tuned parameters of the GNN baselines. In particular, we use the two most effective PGPRec variants as reported in Section 4.3 and calculate the number of tuned parameters. Since efficiency and effectiveness are both critical in our study, we first compare the calculated number of tuned parameters in PGPRec as well as in the conventionally fine-tuned GNN baselines in Section 4.1.3 (NGCF$_{PT}$, LightGCN$_{PT}$ and SGL$_{PT}$)[6]. Figure 4 shows a comparison of the effectiveness and efficiency of the fine-tuned GNN baseline models in comparison to the variants of PGPRec. For conciseness, we use Recall@10 to report the effectiveness of PGPRec and the used baselines in Figure 4, since the use of NDCG@10 also leads to the same conclusions across the used datasets. As shown in Figure 4, LightGCN$_{PT}$ and SGL$_{PT}$ have the same number of tuned parameters but less so than NGCF$_{PT}$, which has additional learned weight matrices compared to LightGCN$_{PT}$ and SGL$_{PT}$ (see Section 4.1.3). Note also that SGL$_{PT}$ has the best effectiveness among the baselines. We observe that the two PGPRec variants need only 41% and 47% of the required fine-tuned parameters of SGL$_{PT}$ to achieve an even higher effectiveness on the Elec-Phone dataset. Similarly, the two PGPRec variants only need 33% and 38% of the tuned parameters required by SGL$_{PT}$ on the Sport-Cloth dataset. Next, we also compare our PGPRec variants with NGCF$_{PT}$. Consider the Elec-Phone dataset as an example. Figure 4 shows that the two PGPRec variants need only 34% and 39% of the required fine-tuned parameters of NGCF$_{PT}$, respectively. Similarly on the Sport-Cloth dataset, the two PGPRec variants need 27% and 31% of the fine-tuned parameter required by NGCF$_{PT}$, respectively. Similar conclusions can also be observed on the Sport-Phone and Elec-Cloth datasets, demonstrating the parameter efficiency of PGPRec. Overall, the results in terms of both effectiveness and efficiency on both datasets demonstrate that our personalised graph prompt-tuning is more parameter-efficient than the conventional fine-tuning when transferring pre-trained knowledge to a target domain. Meanwhile, our PGPRec framework can achieve a competitive performance compared to the fine-tuned GNN baselines on both datasets.

To investigate the parameter-efficiency impact of both hard and soft graph prompts within our PGPRec framework, we conduct a comparative analysis on the number of tuned parameters, taking into account the different types of graph prompts used. This comparative analysis encompasses two types of PGPRec variants - one exclusively employs hard prompts (indicated by the red line) while the other uses a combination of hard and soft prompts (indicated by the green line) - as well as the fine-tuned SGL$_{PT}$ baseline. Note that we use SGL$_{PT}$ for comparison in this experiment because it is the most effective baseline. For a fair comparison, both the PGPRec variants and SGL$_{PT}$ use the same GNN encoder (i.e LightGCN). Figure 5 shows the results of the two PGPRec variants in comparison to the fine-tuned SGL$_{PT}$ baseline on all four used datasets. From Figure 5, we observe that the effectiveness of the variant that uses seven hard prompts (in red) is on a par with the variant that combines five hard prompts with three soft prompts (in green) while reducing the number of tuned parameters on the x-axes of all four used datasets. This result suggests that the hard prompts are overall more efficient in estimating the users' preferences in the target domain. In particular, the PGPRec variant that leverages seven hard prompts only needs 41% and 33% of the tuned parameters of SGL$_{PT}$ to achieve an even better performance on both the Elec-Phone dataset and the Sport-Cloth dataset, respectively. This means that the personalised prompt-tuning allows to reduce 59% and 67% of the parameters in comparison to using SGL$_{PT}$, while also attaining an improved effectiveness. Similarly, the PGPRec variant that uses seven hard prompts can reduce by 54% the number of tuned parameters of SGL$_{PT}$ on the Sport-Phone dataset, and the PGPRec variant that uses five hard prompts can reduce by 74% the number of tuned parameters of SGL$_{PT}$ on the Elec-Cloth dataset, while ensuring a competitive performance on both datasets. To verify that the reduction in the number of tuned parameters does not lead to a significant degradation of effectiveness in comparison to SGL$_{PT}$ on all 4 used datasets, we apply a two one-sided equivalence test (TOST). This test is performed to ascertain an effectiveness equivalence with the SGL$_{PT}$ model. In Figure 5, a point marker (corresponding to a green or red dot) indicates that the corresponding performance is significantly equivalent with SGL$_{PT}$ using the TOST test. The TOST results validate our argument that only tuning on the parameters of graph prompts with all other pre-trained parameters remaining unchanged can achieve a competitive performance in comparison to SGL$_{PT}$ on all used datasets.

[6] Recall, that in cross-domain recommendation, all the pre-trained parameters of the GNN recommenders are fine-tuned in the target domain, while in PGPRec, we only fine-tune the parameters from the personalised graph prompts in the target domain.

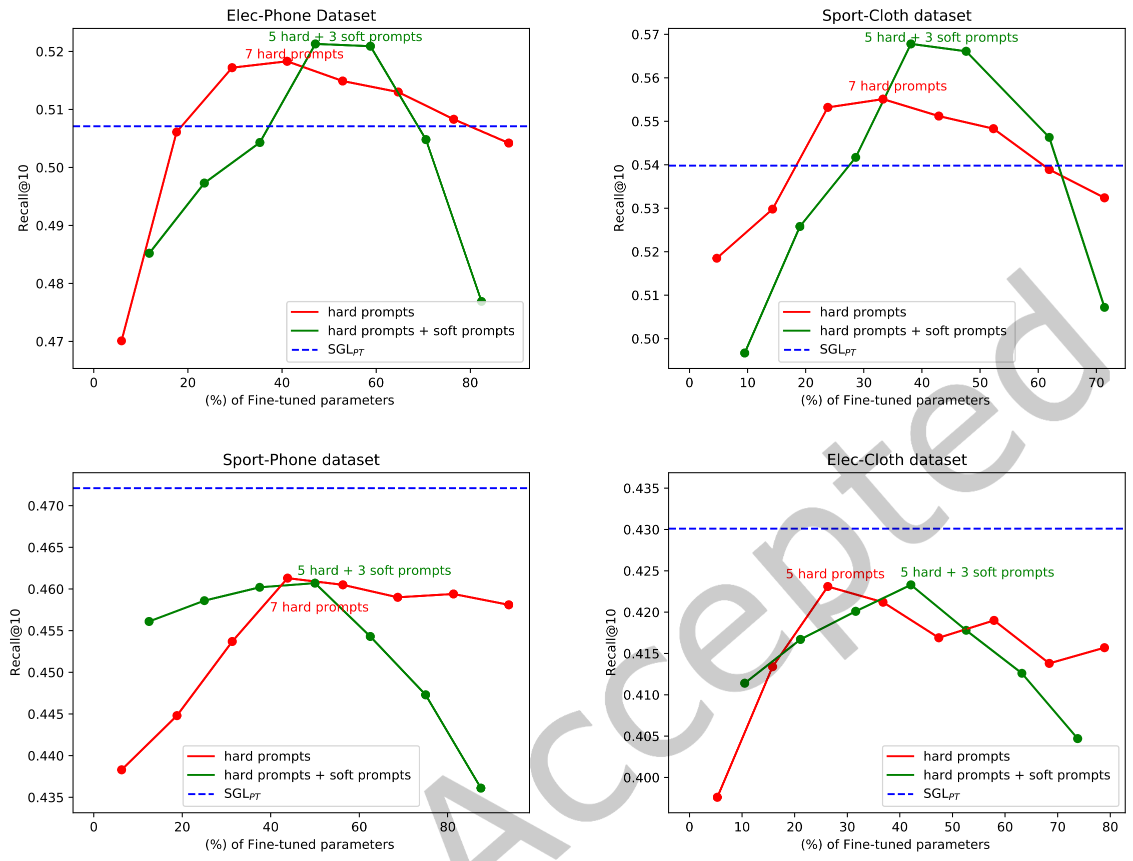

Fig. 5. A parameter comparison between two types of PGPRec variants: one with only hard prompts and another that combines hard and soft prompts. The peak performances of the two types of PGPRec variants, denoted by the red and green lines respectively, are highlighted with text annotations. A point marker, represented as a red dot for the PGPRec variant with hard prompts and a green dot for the PGPRec variant that combines hard and soft prompts, indicates that the corresponding performance is significantly equivalent over the SGL$_{PT}$ baseline using the TOST test.

4.4.2 Efficiency of Tuning Time. We now analyse the time efficiency of PGPRec. As described in Section 4.1.4, for a fair comparison, we use one Geforce RTX 2080Ti GPU to conduct the time efficiency experiments. For conciseness, we report the tuning times for PGPRec and the two most effective baselines (LightGCN$_{PT}$ and SGL$_{PT}$) on the Elec-Phone dataset, since the conclusions on the other datasets are similar. Table 6 compares the efficiency of various PGPRec variants on the Elec-Phone dataset in terms of tuning time. Specifically, we compare the effective PGPRec variants from Section 4.3, namely the variants with seven hard prompts (denoted by PGPRec$_{hard}$), three soft prompts (denoted by PGPRec$_{soft}$) and the mixture of five hard prompts along with three soft prompts (denoted by PGPRec$_{mixture}$) to LightGCN$_{PT}$ and SGL$_{PT}$. Note that we do not compare with NGCF$_{PT}$ as it has additional learned weight matrices compared to LightGCN$_{PT}$ and SGL$_{PT}$. From Table 6, we observe that all PGPRec variants have a faster training speed (seconds per epoch) than the LighGCN$_{PT}$ and SGL$_{PT}$ baselines, which indicates the efficiency of using prompts to tune on a small portion of parameters. Moreover,

Table 6. Efficiency comparison on the Elec-Phone Dataset in terms of tuning time. PGPRec$_{hard}$, PGPRec$_{soft}$ and PGPRec$_{mixture}$ refer to the PGPRec variants with hard prompts, soft prompts and mixture prompts, respectively.

Model	Time/Epoch	Nbr of Epochs	Training Time	Recall@10
LightGCN$_{PT}$	63s	294	309m	0.4821
SGL$_{PT}$	68s	138	157m	0.5071
PGPRec$_{mixture}$	57s	423	402m	0.5213
PGPRec$_{soft}$	26s	367	159m	0.4716
PGPRec$_{hard}$	44s	147	108m	0.5172

Table 7. Cold-start analysis results for PGPRec and the SGL$_{PT}$ baseline, where PT is the abbreviation for Pre-Training. 'Cold-start' denotes the cold-start users and 'Regular' denotes the regular users in the used datasets. * denotes a significant difference between PGPRec and SGL$_{PT}$ using the paired t-test with p<0.05.

Datasets		PGPRec (Recall@10)	SGL$_{PT}$ (Recall@10)	%Improv.
Elec-Phone	Cold-start	0.4671*	0.4314	8.27%
	Regular	0.7501*	0.7343	2.15%
Sport-Cloth	Cold-start	0.5185*	0.4653	11.41%
	Regular	0.6735*	0.6564	2.60%
Sport-Phone	Cold-start	0.4224*	0.4042	4.5%
	Regular	0.5483	0.5687	-3.59%
Elec-Cloth	Cold-start	0.3876*	0.3764	2.98%
	Regular	0.6192	0.6354	-2.62%

PGPRec$_{hard}$ takes the least training time while achieving a competitive performance compared with SGL$_{PT}$ and PGPRec$_{mixture}$. This demonstrates the efficiency of prompt-tuning compared with conventional fine-tuning as well as the effectiveness of hard prompts in enhancing the pre-trained embeddings. However, Table 6 also shows that PGPRec$_{soft}$ and PGPRec$_{mixture}$ take more epochs to converge, which indicates the difficulty of the graph encoder (LightGCN) in PGPRec to train useful soft graph prompts from scratch.

To answer RQ3, according to Figure 4 and Figure 5, we conclude that PGPRec empowered by personalised graph prompts is more parameter-efficient than fine-tuned GNNs such as NGCF$_{PT}$, LightGCN$_{PT}$ and SGL$_{PT}$. Moreover, as shown in Table 6, we find that only using hard graph prompts helps to further improve the efficiency of tuning time while maintaining a competitive effectiveness.

4.5 Cold-start Analysis (RQ4)

To investigate the effectiveness of transferring knowledge to the cold-start users using our personalised graph prompts, we examine our PGPRec framework in a cold-start scenario. Specifically, we use the most effective PGPRec variant from Section 4.3, namely the PGPRec variant with five hard prompts and three soft prompts, and perform a cold-start analysis on four Amazon Review datasets. Table 7 shows the performances of PGPRec for both the cold-start and regular users, in comparison to the best used baseline, SGL$_{PT}$, in terms of Recall@10 (similar trends can be observed with NDCG@10). The results show that PGPRec benefits the cold-start users more than SGL$_{PT}$ and significantly according to the paired t-test on all four datasets. This observation suggests that PGPRec successfully leverages the neighbouring-items and the learned vectors as graph prompts to bring useful information to estimate a user's preferences. In particular, PGPRec outperforms SGL$_{PT}$ by 8.27% and 11.41% on the Elec-Phone dataset and the Sport-Cloth datasets, respectively, thereby demonstrating the successful feature transfer from the source domain and the added-value of personalised graph prompts as additional information.

Significant improvements on the cold-start users in comparison to SGL$_{PT}$ are also obtained on the Sport-Phone and Elec-Cloth datasets, although their scale is smaller (4.5% and 2.98%, respectively. This latter observation indicates that the transferred features from the source domain are not sufficiently informative to enhance the prediction of cold-start users' preference in a distant target domain. In fact, given the scale of improvements in Table 7, we observe that our PGPRec framework actually benefits the cold-start users more than the regular users. For example, on the Sport-Cloth dataset, PGPRec improves the performance by 11.41% for the cold-start users in comparison to SGL$_{PT}$, while it only improves the performance by just 2.60% for the regular users. This suggests that PGPRec is particularly helpful in cold-start scenarios. This result suggests that the PGPRec framework successfully leverages the personalised graph prompts to act as additional adjacent items to a cold-start user, thereby enriching the representations of users with sparse interactions in the target domain.

Overall, in response to RQ4, we conclude that our PGPRec framework successfully leverages the pre-trained features and the personalised graph prompts as auxiliary information to effectively enhance the recommendation performance on cold-start users.

5 CONCLUSIONS

In this work, we proposed a Personalised Graph-based Prompt Recommendation (PGPRec) framework to enhance the efficiency of conventional fine-tuning in cross-domain recommendation. Specifically, we devised novel personalised graph prompts to further enrich the pre-trained embeddings in the target domain. In particular, we used the neighbouring-items as hard prompts and used randomly initialised embedding vectors as soft prompts. We leveraged the personalised graph prompts to enhance the efficiency of the tuning stage. Our results on four cross-domain datasets showed that PGPRec efficiently leverages the prompt-tuning along with the state-of-the-art graph recommenders and achieves a competitive performance compared with the strongest baseline (SGL$_{PT}$). Moreover, we conducted an ablation study investigating different combinations of the personalised graph prompts. We showed that the hard prompts make a key and marked contribution in the tuning stage while the soft prompts can provide limited improvement to top-k recommendation in the target domain. Furthermore, we showed that a PGPRec variant with five hard prompts can markedly reduce the number of the required tuned parameters by a large margin (e.g., a 74% reduction in the number of tunable parameters of the strongest baseline SGL$_{PT}$ on the Elec-Cloth dataset). By comparing the tuning time of the existing fine-tuned GNNs with that of several PGPRec variants, we showed that our PGPRec framework tunes faster (e.g., 31% faster on the Elec-Phone dataset), when using only hard prompts, leading to a marked training efficiency. Finally, we conducted a cold-start analysis to investigate whether our proposed PGPRec framework benefits the cold-start users. The results obtained on four datasets showed that PGPRec successfully alleviates the cold-start problem, and achieves effective cross-domain recommendations (e.g. an 11.41% performance improvement on the Sport-Cloth dataset in comparison to the strongest baseline SGL$_{PT}$) for cold-start users.

REFERENCES

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proc. of NeurIPS*.

[2] Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022. Revisiting parameter-efficient tuning: Are we really there yet?. In *Proc. of EMNLP*.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proc. of the ICML*.

[4] Qiang Cui, Tao Wei, Yafeng Zhang, and Qing Zhang. 2020. HeroGRAPH: A heterogeneous graph framework for multi-target cross-domain recommendation. In *Proc. of RecSys*.

[5] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-Rec: Generative pre-trained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084* (2022).

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

[7] Taoran Fang, Yunchao Zhang, Yang Yang, and Chunping Wang. 2022. Prompt tuning for graph neural networks. *arXiv preprint arXiv:2209.15240* (2022).

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

[9] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (RLP): A unified pretrain, personalised prompt & predict paradigm (P5). In *Proc. of RecSys*.

[10] Henry Gouk, Timothy M Hospedales, and Massimiliano Pontil. 2021. Distance-based regularisation of deep networks for fine-Tuning. In *Proc. of ICLR*.

[11] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *Proc. of ICLR*.

[12] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proc. of SIGIR*.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proc. of WWW*.

[14] Guangneng Hu, Yu Zhang, and Qiang Yang. 2018. Conet: Collaborative cross networks for cross-domain recommendation. In *Proc of CIKM*.

[15] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *TOIS* 20, 4 (2002).

[16] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *TACL* 8 (2020).

[17] Chaitanya Joshi. 2020. Transformers are graph neural networks. *The Gradient* 7 (2020).

[18] Saranya KG and G Sudha Sadasivam. 2014. A survey on personalized recommendation techniques. *IJRITCC* 2, 6 (2014).

[19] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proc. of ICLR*.

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

[21] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimising continuous prompts for generation. In *Proc. of ACL*.

[22] Juncheng Liu, Kenji Kawaguchi, Bryan Hooi, Yiwei Wang, and Xiaokui Xiao. 2021. Eignn: Efficient infinite-depth graph neural networks. In *Proc. of NeuIPS*.

[23] Meng Liu, Jianjun Li, Guohui Li, and Peng Pan. 2020. Cross domain recommendation via bi-directional transfer graph collaborative filtering networks. In *Proc. of CIKM*.

[24] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2022. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *Computing Surveys* 55, 9 (2022).

[25] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *Proc. of ACL*.

[26] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479* (2021).

[27] Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S Yu. 2021. Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer. In *Proc. of SIGIR*.

[28] Babak Loni, Yue Shi, Martha Larson, and Alan Hanjalic. 2014. Cross-domain collaborative filtering with factorisation machines. In *Proc. of ECIR*.

[29] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: an embedding and mapping approach.. In *Proc. of IJCAI*.

[30] Jarana Manotumruksa, Dimitrios Rafailidis, Craig Macdonald, and Iadh Ounis. 2019. On cross-domain transfer in venue recommendation. In *Proc. of ECIR*.

[31] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: ultra simplification of graph convolutional networks for recommendation. In *Proc. of CIKM*.

[32] Zaiqiao Meng, Siwei Liu, Craig Macdonald, and Iadh Ounis. 2023. Graph neural pre-training for enhancing recommendations using side information. *TOIS* 41, 3 (2023).

[33] Nima Mirbakhsh and Charles X Ling. 2015. Improving top-n recommendation for cold-start users via cross-domain information. *TKDD* 9, 4 (2015).

[34] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[35] Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proc. of NAACL*.

[36] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. GCC: Graph contrastive coding for graph neural network pre-training. In *Proc. of KDD*.

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

[38] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proc. of UAI.*

[39] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender Systems Handbook.*

[40] Tetsuya Sakai. 2021. On Fuhr's guideline for IR evaluation. In *Proc. of SIGIR Forum.*

[41] Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. In *Proc. of EACL.*

[42] Donald J Schuirmann. 1987. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of pharmacokinetics and biopharmaceutics* 15 (1987).

[43] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980* (2020).

[44] Nina Simmons-Mackie, Linda Worrall, Laura L Murray, Pam Enderby, Miranda L Rose, Eun Jin Paek, and Anu Klippi. 2017. The top ten: best practice recommendations for aphasia. *Aphasiology* 31, 2 (2017).

[45] 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 *Proc. of CIKM.*

[46] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. 2022. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *Proc. of KDD.*

[47] Jun Tang, Haiqun Jin, Shoubiao Tan, and Dong Liang. 2016. Cross-domain action recognition via collective matrix factorization with graph Laplacian regularization. *Image and Vision Computing* 55 (2016).

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeuIPS.*

[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *Proc. of the ICLR.*

[50] Chen Wang, Yueqing Liang, Zhiwei Liu, Tao Zhang, and S Yu Philip. 2021. Pre-training graph neural network for cross-domain recommendation. In *Proc. of CogMI.*

[51] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proc. of SIGIR.*

[52] Yongpeng Wang, Hong Yu, Guoyin Wang, and Yongfang Xie. 2020. Cross-domain recommendation based on sentiment analysis and latent feature mapping. *Entropy* 22, 4 (2020).

[53] Hui Wu and Xiaodong Shi. 2022. Adversarial soft prompt tuning for cross-domain sentiment analysis. In *Proc. of ACL.*

[54] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proc. of SIGIR.*

[55] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Fuzhen Zhuang, Xu Zhang, Leyu Lin, and Qing He. 2022. Personalized Prompts for Sequential Recommendation. *arXiv preprint arXiv:2205.09666* (2022).

[56] Chaojun Xiao, Ruobing Xie, Yuan Yao, Zhiyuan Liu, Maosong Sun, Xu Zhang, and Leyu Lin. 2021. UPRec: User-aware pre-training for recommender systems. *arXiv preprint arXiv:2102.10989* (2021).

[57] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *Proc. of ICDE.*

[58] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *Proc. of ICLR.*

[59] Zhen Yang, Ming Ding, Bin Xu, Hongxia Yang, and Jie Tang. 2022. STAM: A spatiotemporal aggregation method for graph neural network-based recommendation. In *Proc. of WWW.*

[60] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Jundong Li, and Zi Huang. 2023. Self-supervised learning for recommender systems: a survey. *TKDE* (2023).

[61] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. 2020. Parameter-efficient transfer from sequential behaviours for user modeling and recommendation. In *Proc. of the SIGIR.*

[62] Feng Yuan, Lina Yao, and Boualem Benatallah. 2019. DARec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns. In *Proc. of IJCAI.*

[63] Tianzi Zang, Yanmin Zhu, Haobing Liu, Ruohan Zhang, and Jiadi Yu. 2022. A survey on cross-domain recommendation: taxonomies, methods, and future directions. *TOIS* 41, 2 (2022).

[64] Zheni Zeng, Chaojun Xiao, Yuan Yao, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2021. Knowledge transfer via pre-training for recommendation: A review and prospect. *Frontiers in Big Data* 4 (2021).

[65] Cheng Zhao, Chenliang Li, and Cong Fu. 2019. Cross-domain recommendation via preference propagation graphnet. In *Proc. of CIKM.*

[66] Yongchun Zhu, Zhenwei Tang, Yudan Liu, Fuzhen Zhuang, Ruobing Xie, Xu Zhang, Leyu Lin, and Qing He. 2022. Personalized transfer of user preferences for cross-domain recommendation. In *Proc. of WSDM.*