

Entities	Description	CDs & Vinyl	Clothing	Cell Phones	Beauty
		Number of Entities			
User	User in recommender system	75,258	39,387	27,879	22,363
Item	Product to be recommended to users	64,443	23,033	10,429	12,101
Feature	A product feature word from reviews	202,959	21,366	22,493	22,564
Brand	Brand or manufacturer of the product	1,414	1,182	955	2,077
Category	Category of the product	770	1,193	206	248
Relations	Description	Number of Relations per Head Entity			
		14.58 ± 39.13	7.08 ± 3.59	6.97 ± 4.55	8.88 ± 8.16
Purchase	User $\xrightarrow{\text{purchase}}$ Item	14.58 ± 39.13	7.08 ± 3.59	6.97 ± 4.55	8.88 ± 8.16
Mention	User $\xrightarrow{\text{mention}}$ Feature	2, 545.92 ± 10, 942.31	440.20 ± 452.38	652.08 ± 1335.76	806.89 ± 1344.08
Described_by	Item $\xrightarrow{\text{described_by}}$ Feature	2, 973.19 ± 5, 490.93	752.75 ± 909.42	1, 743.16 ± 3, 482.76	1, 491.16 ± 2, 553.93
Belong_to	Item $\xrightarrow{\text{belong_to}}$ Category	7.25 ± 3.13	6.72 ± 2.15	3.49 ± 1.08	4.11 ± 0.70
Produced_by	Item $\xrightarrow{\text{produced_by}}$ Brand	0.21 ± 0.41	0.17 ± 0.38	0.52 ± 0.50	0.83 ± 0.38
Also_bought	Item $\xrightarrow{\text{also_bought}}$ Item	57.28 ± 39.22	61.35 ± 32.99	56.53 ± 35.82	73.65 ± 30.69
Also_viewed	Item $\xrightarrow{\text{also_viewed}}$ another Item	0.27 ± 1.86	6.29 ± 6.17	1.24 ± 4.29	12.84 ± 8.97
Bought_together	Item $\xrightarrow{\text{bought_together}}$ another Item	0.68 ± 0.80	0.69 ± 0.90	0.81 ± 0.77	0.75 ± 0.72

Table 1: Descriptions and statistics of four Amazon e-commerce datasets: CDs & Vinyl, Clothing, Cell Phones and Beauty.

the corresponding reasoning paths $\{p_n(u, i_n)\}$. One straightforward way is to sample n paths for each user u according to the policy network $\pi(\cdot|s, \tilde{A}_u)$. However, this method cannot guarantee the diversity of paths, because the agent guided by the policy network is likely to repeatedly search the same path with the largest cumulative rewards. Therefore, we propose to employ beam search guided by the action probability and reward to explore the candidate paths as well as the recommended items for each user. The process is described as Algorithm 1. It takes as input the given user u , the policy network $\pi(\cdot|s, \tilde{A}_u)$, horizon T , and predefined sampling sizes at each step, denoted by K_1, \dots, K_T . As output, it delivers a candidate set of T -hop paths \mathcal{P}_T for the user with corresponding path generative probabilities Q_T and path rewards \mathcal{R}_T . Note that each path $p_T(u, i_n) \in \mathcal{P}_T$ ends with an item entity associated with a path generative probability and a path reward.

For the acquired candidate paths, there may exist multiple paths between the user u and item i_n . Thus, for each pair of (u, i_n) in the candidate set, we select the path from \mathcal{P}_T with the highest generative probability based on Q_T as the one to interpret the reasoning process of why item i_n is recommended to u . Finally, we rank the selected interpretable paths according to the path reward in \mathcal{R}_T and recommend the corresponding items to the user.

4 EXPERIMENTS

In this section, we extensively evaluate the performance of our PGPR method on real-world datasets. We first introduce the benchmarks for our experiments and the corresponding experimental settings. Then we quantitatively compare the effectiveness of our model with other state-of-the-art approaches, followed by ablation studies to show how parameter variations influence our model.

4.1 Data Description

All experiments are conducted on the Amazon e-commerce datasets collection [7], consisting of product reviews and meta information from Amazon.com. The datasets include four categories: CDs and Vinyl, Clothing, Cell Phones and Beauty. Each category is considered as an individual benchmark that constitutes a knowledge graph

containing 5 types of entities and 7 types of relations. The description and statistics of each entity and relation can be found in Table 1. Note that once the type of head entity and relation are provided, the type of tail entity is uniquely determined. In addition, as shown in Table 1, we find that *Mention* and *Described_by* account for a very large proportion among all relations. These two relations are both connected to the *Feature* entity, which may contain redundant and less meaningful words. We thus adopt TF-IDF to eliminate less salient features in the preprocessing stage: For each dataset, we keep the frequency of feature words less than 5,000 with TF-IDF score > 0.1 . We adopt the same data split rule as in previous work [32], which randomly sampled 70% of user purchases as the training data and took the rest 30% as test. The objective in the KGRE-Rec problem is to recommend items purchased by users in the test set together with reasoning paths for each user-item pair.

4.2 Experimental Setup

Baselines & Metrics. We compare our results against previous state-of-the-art methods. **BPR** [20] is a Bayesian personalized ranking model that learns latent embeddings of users and items. **BPR-HFT** [16] is a Hidden Factors and Topics (HFT) model that incorporates topic distributions to learn latent factors from reviews of users or items. **VBPR** [8] is the Visual Bayesian Personalized Ranking method that builds upon the BPR model but incorporates visual product knowledge. **TransRec** [6] invokes translation-based embeddings for sequential recommendation. It learns to map both user and item representations in a shared embedding space through personalized translation vectors. **DeepCoNN** or Deep Cooperative Neural Networks [36] are a review-based convolutional recommendation model that learns to encode both users and products with reviews assisting in rating prediction. **CKE** or Collaborative Knowledge base Embedding [31] is a modern neural recommender system based on a joint model integrating matrix factorization and heterogeneous data formats, including textual contents, visual information and a structural knowledge base to infer the top- N recommendations results. **JRL** [32] is a start-of-the-art joint representation learning model for top- N recommendation that utilizes