A EVALUATION METRICS DEFINITION

Given target bundle $\mathcal{B}_u^*$ and all items the model proposed as bundle $\hat{\mathcal{B}}_u$ (note that it is not $\check{\mathcal{B}}_u$, $\check{\mathcal{B}}_u$ is all items that user u accepted), we follow [47] to define metrics as:

$$\text{Precision} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{|\mathcal{B}_u^*\cap\hat{\mathcal{B}}_u|}{|\hat{\mathcal{B}}_u|} \qquad \text{Recall} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{|\mathcal{B}_u^*\cap\hat{\mathcal{B}}_u|}{|\mathcal{B}_u^*|}$$

$$\text{Accuracy} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{|\mathcal{B}_u^*\cap\hat{\mathcal{B}}_u|}{|\mathcal{B}_u^*\cup\hat{\mathcal{B}}_u|} \qquad \text{F1-Score} = \frac{2\cdot\text{Accuracy}\cdot\text{Precision}}{\text{Accuracy}+\text{Precision}}$$

B DATASET PROCESSING

B.1 Steam Dataset

We pre-process the Steam raw dataset[12], which contains user-item interactions and user-bundle interactions. We focus on bundle interactions only, so filter out the users' interactions with individual items or bundles that contain only single item. Then, to make users in Steam datasets can be evaluated using our protocol, we filter out users who have only one bundle. We treat item *tags* in metadata (which is a list of phrases per item, such as "[Earl Access, Video Production]") as the *attributes* in Bundle MCR. Apart from attributes, we also use item *genres* (which is also a list of phrases per item, such as "[Indie, Sports]") in metadata as *category* in Bundle MCR. After indexing items, attributes and categories we follow the data splitting mentioned in Section 6.1 to randomly split the processed dataset, other datasets below follow the same data splitting process.

B.2 MovieLens Dataset

This MovieLens 10M raw dataset[13] contains (user, item, timestamp) tuples as well as item metadata. We first group user-item interactions by timestamp, with the second-level granularity. Then, if the number of (user, item) pairs with the same timestamp is large than 1, we assume these items construct a bundle. We focus on bundle interactions only, and to ensure the quality of this noisy dataset (because bundles are not explicitly defined as Steam dataset), we filter out users and items that appear not more than 3 times. Moreover, we use *tags* in item metadata as the *attributes* in Bundle MCR, which are a list of phrase per item, such as "[kung fu, Zombies]". We process tags into lower case and filter out all punctuations to normalize these tags. To ensure the tag quality, we only use frequent tags (appearing more than 5 times) as our final attributes. Moreover, we use *genres* in metadata and also only use frequent genres as our *categories* in Bundle MCR. We index items, categories, attributes and randomly split the dataset.

B.3 Clothing Dataset

We download clothing subcategory (i.e., categoty titled *Clothing, Shoes & Jewelry*) from Amazon Review 2018[14], we do the same processing as Appendix B.2 to obtain user-bundle interactions, where the difference is the timestamp granularity is day-level in Amazon Review dataset. Clothing metadata provides item *category* features, we use it by removing the most general category (i.e., *Clothing, Shoes & Jewelry*, which is the category for all items thus meaningless in our task) and filter out categories that appear less than 10 times because many infrequent and meaningless sentences exist in metadata for some reason. Moreover, we found user will specify the item style when submitting reviews, such

[12] https://cseweb.ucsd.edu/~jmcauley/datasets.html#steam_data
[13] https://grouplens.org/datasets/movielens/10m/
[14] https://nijianmo.github.io/amazon/index.html

as "Format: Hardcover", which are collected as dictionaries in metadata. So we collect all style words mentioned in item reviews, normalizing these words by using lower case and filtering our words that appear less than 10 times. We use the clean style words as attribute in Bundle MCR and index items, categories, attributes, then randomly split the dataset.

B.4 iFashion Dataset

We download iFashion dataset[15]. We treat the outfit (consisting of multiple items such as pants, shoes) as bundle, and only use the user-bundle interactions as other dataset processing. Then, similar to [43], to ensure data quality, we use the 10-core dataset. We use the *category* provided by iFahsion as the *category* in Bundle MCR, which are hashed ids. Then, we tokenize the titles (which are in Chinese) with a Chinese tokenizer Jieba[16], and remove all Chinese punctuations to get frequent words (with frequency ≥ 10) as *attributes* in Bundle MCR.

C BUNT DETAILS

For BUNT hyper-parameters tuning, we did grid hyper-parameter searching (following the same granularity of all baselines) as: (1) we first search learning rate lr = {1e-4, 5e-4, ..., 5e-3} and batch size bs = [32, 64, 128, 256]; (2) then use the best model to search weight decay from {0, 1e-6, 1e-4, 1e-2, 1}, dropout from [0, 0.1, ..., 0.9]. We post the hyper-parameters that used in our offline pretraining and online fine-tuning. The common hyper-paramters are `embed_size`=32, `batch_size`=32, λ=0.1, ρ=0.5 and the others are listed below:

- **Steam:**
  - Offline: `n_layers`=2, `n_heads`=4, `lr`=1e-4, `dropout`=0, `weight_decay`=0;
  - Online: `final_metric`=f1, `rollout_step`=512, others are default values in `Stable-Baselines` PPO.
- **MovieLens:**
  - Offline: `n_layers`=2, `n_heads`=2, `lr`=1e-4, `dropout`=0, `weight_decay`=1e-6;
  - Online: `final_metric`=f1, `rollout_step`=512, others are default values in `Stable-Baselines` PPO.
- **Clothing:**
  - Offline: `n_layers`=2, `n_heads`=4, `lr`=5e-4, `dropout`=0, `weight_decay`=1e-6;
  - Online: `final_metric`=prec, `rollout_step`=512, others are default values in `Stable-Baselines` PPO.
- **iFashion:**
  - Offline: `n_layers`=1, `n_heads`=2, `lr`=5e-4, `dropout`=0.1, `weight_decay`=1e-6;
  - Online: `final_metric`=rec, `rollout_step`=1024, others are default values in `Stable-Baselines` PPO.

D HUMAN EVALUATION DETAILS

We randomly samples 1000 (in total) users and 1000 pairs of conversation trajectories from <BUNT-Learn, SCPR*> or <BUNT-Learn, FM-Learn> in Steam and MovieLens datasets. For Steam dataset, each item (video game) has image url, which is also shown in our conversation trajectories. For MovieLens dataset, we cannot show the poster images because they are not provided in the raw dataset. Therefore, the user has to read the name of movie, which might require more domain knowledge (compared with movie names, video game thumbnail images are very straightforward). Each pair of conversation trajectories is posted to five MTurk[17] workers, who are required to measure the subjective quality by browsing the conversations and selecting the best model from the given pair. The question we posted is *"Imagine*

[15] https://github.com/wenyuer/POG
[16] https://github.com/fxsjy/jieba
[17] https://requester.mturk.com/

that you are the user, the items you love is posted. Please select the conversation that makes you feel it is comfortable and useful.". The workers can only choose one from these two conversation trajectories per question. The order of conversation trajectories are shuffled. We use the answers from high-quality workers who spend more than 30 seconds and the `LifeTimeAcceptanceRate` is 100%, and count the majority votes per pair. For these 388 valid results, we count the majority votes and discard questions where two answers tie (it happens even though we set 5 workers, some are removed by the mentioned pre-filtering.) These workers are paid $0.01 per valid answer.