

A Supplement to Method

In this section, the details of prompt design and the HPSERec procedures are addressed.

A.1 Distribution Balancing Module

In Section 3.2, we introduced the formulation of data imbalance. In this section, we provide a detailed description of the algorithmic procedure used to partition the item set into balanced subsets.

Let $I(\Theta) = \sum_{k=1}^L \mathcal{J}(V_k)$, where $\Theta = \{T_1, \dots, T_{L-1}\}$ is the set of threshold.

Algorithm 1 describes the dynamic programming approach used to identify the optimal threshold values Θ that minimize the overall imbalance score $I(\Theta)$.

Algorithm 1: Distribution Balancing Module

Data: An array V storing the set of items, and an array N storing the interaction counts for items in the set.

Result: The list of thresholds Θ .

```

1 Sort  $V$  in descending order based on item interaction counts;
2 Initialize  $dp[0 : |V|, 0 : L] \leftarrow \infty$ ;
3 Initialize  $path[0 : |V|, 0 : L] \leftarrow -1$ ;
4 for  $l \leftarrow 1$  to  $L$  do
5   for  $i \leftarrow 1$  to  $|V|$  do
6     for  $j \leftarrow l - 1$  to  $i$  do
7       Calculate  $J(V[j : i])$  by Eq. (7);
8       if  $H(V[j : i]) + dp[i][l - 1] < dp[i][l]$  then
9          $dp[i][l] \leftarrow H(V[j : i]) + dp[i][l - 1]$ ;
10         $\Theta[l - 1] \leftarrow j$ ;
11      end
12    end
13  end
14 end
15 return  $\Theta$ ;
```

Based on the threshold set Θ , the item subsets $V'_1, V'_2, \dots, V'_L \subseteq V$ are generated after partitioning. To better enable long-tail experts to guide the training of downstream experts, we propose that gradually expanding the range of input items for each expert, rather than providing disjoint item sets, is more effective in allowing long-tail experts to play an active role in subsequent training. The final subset assigned to each expert model is defined as:

$$V_k = \bigcup_{i=1}^k V'_i \quad (14)$$

where V_k represents the item input range for the k -th expert E_k in the feedforward module. When k approaches 1, the input user sequence contains only tail items, causing the model to focus on capturing the user's tail-item preferences. Conversely, as k approaches L , the input user sequence closely resembles the user's complete interaction history, prompting the model to capture the user's global interests.

A.2 Feedback Module

In this section, we present a detailed proof of the feedback module. For clarity of exposition, we begin by redefining the following notations.

User embeddings produced by the t -th expert can be defined as:

$$U_t = \{e_1^t, \dots, e_n^t\}, \quad (15)$$

while user embeddings produced by the $t + 1$ -th expert can be defined as:

$$U_t = e_1^{t+1}, \dots, e_n^{t+1}. \quad (16)$$

We define the cost matrix $C \in \mathbb{R}^{n \times n}$ using cosine distance:

$$C_{ij} = 1 - \frac{\langle u_i^t, u_j^{t+1} \rangle}{\|u_i^t\| \cdot \|u_j^{t+1}\|} \in [0, 2] \quad (17)$$

Since users are sampled uniformly during training, we assume that both user distributions are uniform:

$$\mu = \nu = \left(\frac{1}{n}, \dots, \frac{1}{n} \right) \in \Delta_n \quad (18)$$

We consider the following entropy-regularized optimal transport objective:

$$\min_{\gamma \in \Pi(\mu, \nu)} \langle \gamma, C \rangle - \varepsilon H(\gamma), \quad (19)$$

where $\gamma \in \mathbb{R}_+^{n \times n}$ is the transport plan, $H(\gamma) = -\sum_{i,j} \gamma_{ij} \log \gamma_{ij}$ is the Shannon entropy, and $\Pi(\mu, \nu) = \{\gamma \in \mathbb{R}_+^{n \times n} \mid \gamma \mathbf{1} = \mu, \gamma^\top \mathbf{1} = \nu\}$.

Let $f(\gamma) = \langle \gamma, C \rangle - \varepsilon H(\gamma)$.

Lemma 1 *The objective function $f(\gamma)$ is strictly convex over its domain.*

Proof. The cost term $\langle \gamma, C \rangle$ is linear in γ , and the entropy term $H(\gamma)$ is strictly concave. Therefore, the regularized term $-\varepsilon H(\gamma)$ is strictly convex. The sum of a linear and strictly convex function remains strictly convex.

Lemma 2 *The feasible set $\Pi(\mu, \nu) \subset \mathbb{R}^{n \times n}$ is non-empty, closed, convex, and compact.*

Proof. The constraints $\gamma \mathbf{1} = \mu$, $\gamma^\top \mathbf{1} = \nu$, and $\gamma \geq 0$ define a convex polytope. The entries of γ are bounded within the interval $[0, 1]$ and their total sum is constant, ensuring compactness. The set is non-empty since $\gamma = \mu \nu^\top$ is a valid coupling that satisfies the marginal constraints.

Proposition 1 *The entropy-regularized optimal transport problem*

$$\min_{\gamma \in \Pi(\mu, \nu)} \langle \gamma, C \rangle - \varepsilon H(\gamma) \quad (20)$$

admits a unique optimal solution $\gamma^ \in \Pi(\mu, \nu)$.*

Proof. From Lemmas 1 and 2, the objective function is strictly convex and the feasible set is compact and non-empty. Therefore, by the fundamental theorem of convex optimization, a unique global minimizer exists.

Proposition 2 *The unique minimizer γ^* has the following closed-form structure:*

$$\gamma^* = \text{diag}(u) \cdot K \cdot \text{diag}(v), \quad K = \exp\left(-\frac{C}{\varepsilon}\right) \quad (21)$$

for some positive scaling vectors $u, v \in \mathbb{R}_+^n$.

Proof. Consider the Lagrangian:

$$\mathcal{L}(\gamma, \alpha, \beta) = \sum_{i,j} \gamma_{ij} C_{ij} - \varepsilon \sum_{i,j} \gamma_{ij} \log \gamma_{ij} + \sum_i \alpha_i \left(\mu_i - \sum_j \gamma_{ij} \right) + \sum_j \beta_j \left(\nu_j - \sum_i \gamma_{ij} \right) \quad (22)$$

Taking the derivative with respect to γ_{ij} and setting it to zero yields:

$$\gamma_{ij} = \exp\left(\frac{\alpha_i + \beta_j - C_{ij}}{\varepsilon} - 1\right) = u_i \cdot K_{ij} \cdot v_j \quad (23)$$

where $u_i = \exp\left(\frac{\alpha_i - 1}{\varepsilon}\right)$, $v_j = \exp\left(\frac{\beta_j}{\varepsilon}\right)$.

784 **Theorem 1** Given a positive matrix $K \in \mathbb{R}_+^{n \times n}$ and marginal distributions $\mu, \nu \in \Delta_n$, there exist
 785 scaling vectors $u, v \in \mathbb{R}_+^n$ such that:

$$\gamma = \text{diag}(u)K \text{diag}(v) \in \Pi(\mu, \nu) \quad (24)$$

786 and the vectors u, v can be computed via the following iterative updates:

$$u^{(k+1)} = \frac{\mu}{Kv^{(k)} + \delta} \quad (25)$$

$$v^{(k+1)} = \frac{\nu}{K^\top u^{(k+1)} + \delta} \quad (26)$$

787 where $\delta > 0$ is a small constant to ensure numerical stability.

788 A.3 Training Strategy

789 For a clearer illustration of the training and inference process, we conclude them in Algorithm 2.

Algorithm 2: HPSERec

Data: S_u for $u \in U$, learning rate lr , hyperparameters $\alpha, \beta, \tau, \eta, \gamma, L$.

Result: Last expert model parameters \mathcal{W}_L .

```

1 Get item sets  $V_1, \dots, V_L$  by Algorithm 1;
2 for  $i \leftarrow 1$  to  $E$  (number of epochs) do
  /* Feedforward stage */
3   for  $j \leftarrow 1$  to  $L$  do
4     if  $j > 1$  then
5       Calculate  $\mathcal{L}_{KD}$  by Eq. (9);
6     end
7     Set  $\mathcal{L}_{forw}$  by Eq. (13);
8     Apply Adam optimizer to  $\mathcal{L}_{forw}$ ;
9     Perform back-propagation to  $\mathcal{L}_{forw}$  getting gradients  $\mathcal{G}$ ;
10    Update  $\mathcal{W}_j$  based on  $\mathcal{G}$ ;
11  end
  /* Feedback stage */
12  for  $j \leftarrow L$  to 1 do
13    Calculate  $\mathcal{L}_{back}$  by Eq. (12);
14    Perform back-propagation to  $\mathcal{L}_{back}$  getting gradients  $\mathcal{G}$ ;
15    Update  $\mathcal{W}_j$  based on  $\mathcal{G}$ ;
16  end
17 end
18 return  $\mathcal{W}_L$ ;

```

790 B Experimental Settings

791 In this section, we will refer to more details about the experimental settings.

792 B.1 Dataset and Preprocessing

793 We compare HPSERec with baseline models using three real-world datasets from online services:
 794 Yelp, Amazon Beauty, and Amazon Music. For data pre-processing, we follow prior research by
 795 filtering out users with fewer than five interactions. Table 3 presents the statistics of the datasets after
 796 pre-processing.

797 B.2 Backbone and Baseline

798 We compare our proposed method with the following baseline methods.

- 799 • **SASRec** employs a self-attention mechanism to model the user’s entire interaction sequence
 800 and predict the next potential item for interaction.

Table 3: Statistics of datasets. #Int denotes interaction.

Dataset	#Items	#Users	#Int	Avg $ S_u $
Beauty	57,289	52,204	394,908	5.6
Yelp	15,720	4,722	192,214	3.8
Music	20,356	20,165	132,595	5.1

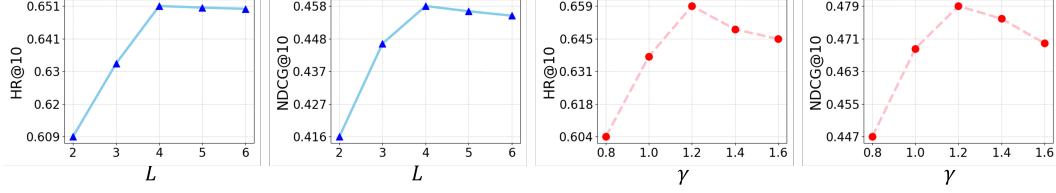


Figure 5: The hyper-parameter experiments on the weight of number of subsets L and the weight of the regularization penalty γ . The result are based on the Music dataset with the SASRec model.

- **BERT4Rec** adopts the Cloze task to train a bidirectional model, extending SASRec for better sequential modeling.
- **CITIES** applies a self-attention mechanism, leveraging head items to improve the representation of tail items.
- **MELT** optimizes the representation of both long-tailed items and users, marking its first effort in this area.
- **RLMRec** leverages LLMs to generate user and item profiles, effectively capturing their interaction preferences.
- **LLMInit** uses LLM embeddings to initialize the embedding layer of the element in the SRS model.
- **LLM-ESR** enhances both long-tailed items and users through a dual view modeling framework combined with a retrieval-augmented self-distillation method.

B.3 Implementation Details

We conduct all experiments on an Intel Core i7-11700KF platform with dual NVIDIA GeForce RTX 3090 (24GB) GPUs. Besides, the implementation is based on Python 3.8.19 and PyTorch 2.0.0. For the backbone SRS models, the number of GRU layers is set to 1 for GRU4Rec, while the number of self-attention layers is fixed at 2 for SASRec and Bert4Rec. Also, the dropout rate is 0.6 for Bert4Rec. We use the Adam optimizer for parameter optimization with a learning rate of 1×10^{-4} . The embedding size is 128 for all baselines. We choose the Adam as the optimizer. For Eq. (5), the default value of α is set to 0.6. For Eq. (6), the default value of L is set to 4. For Eq. (7), the default value of γ is set to 1.2. For Eq. (9), the default value of τ is set to 1.2. For Eq. (13), the default value of β is set to 1.

C More Experimental Results

C.1 Hyper-parameter Analysis

Figure 5 illustrates the impact of several key hyperparameters on the performance of HPSERec. The hyperparameter L controls the number of item subsets generated by the Distribution Balancing module. As L increases from 2 to 6, the recommendation accuracy initially improves and then gradually declines. The suboptimal performance with a small L can be attributed to the large representational disparity between subsets, while an excessively large L may degrade the representation quality of upstream item subsets due to data sparsity. Regarding the regularization weight γ , which controls the penalty term in the partitioning objective, the optimal value is found to be 1.2. A smaller γ results in imbalanced subset sizes, whereas a larger γ overly suppresses the influence of item-level imbalance, thus impairing the quality of the learned partitions.

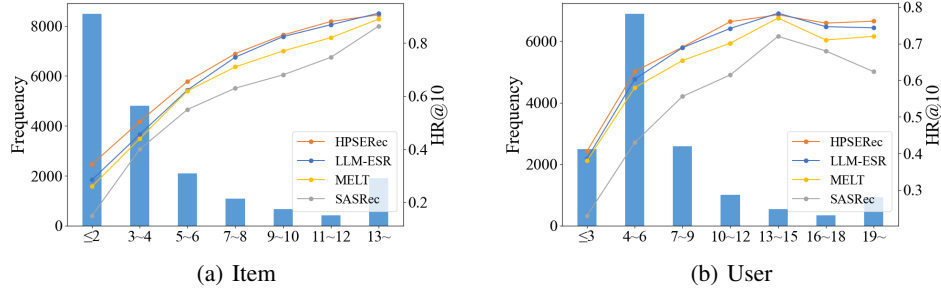


Figure 6: The results of the proposed HPSERec and competing baselines in user and item groups. The results are based on the Music dataset with the SASRec model.

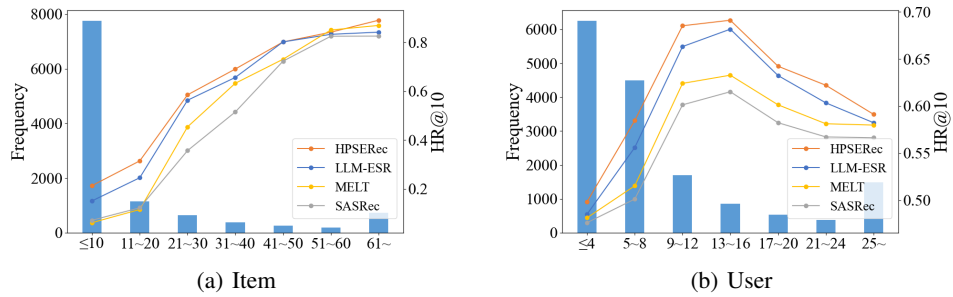


Figure 7: The results of the proposed HPSERec and competing baselines in user and item groups. The results are based on the Yelp dataset with the SASRec model.

834 C.2 Group Analysis

835 To further evaluate the effectiveness of HPSERec, we conduct group-wise performance analysis on
 836 two additional datasets, Music and Yelp. The results are presented in Figures 6 and Figures 7. From
 837 the results, we observe that the LLM-based framework consistently improves performance across all
 838 user and item groups. Moreover, HPSERec demonstrates a clear advantage under extreme long-tail
 839 scenarios, while maintaining competitive performance on head users and items, indicating its ability
 840 to enhance tail representations without compromising head accuracy.

841 C.3 Ablation Studies

842 In this section, we present additional ablation experiments on two benchmark datasets, Yelp and
 843 Beauty, with results summarized in Table 4. Specifically, Row 1 corresponds to the baseline SASRec.
 844 Row 2 shows the performance when head and tail classes are partitioned using conventional frequency-
 845 based heuristics, and data augmentation is applied via the Feedforward module. Building on this,
 846 Row 3 incorporates the Feedback module to further enhance tail-side representations. Row 4 reflects
 847 the effect of applying our proposed Distribution Balancing module for data partitioning, coupled with
 848 Feedforward-based augmentation. Finally, Row 5 adds the Feedback module to Row 4, forming the
 849 complete HPSERec framework. Notably, although HPSERec does not include a dedicated module
 850 targeting tail users, we observe a consistent performance gain for this group. This is because user
 851 and item representations are embedded in the same vector space, and user embeddings are computed
 852 based on the items they interact with—thus, improving item representations inherently enhances user
 853 representations as well. Moreover, unlike many long-tail recommendation methods that favor tail
 854 performance at the cost of head accuracy, HPSERec achieves balanced improvements across head and
 855 tail users/items. The full model leads to consistent performance gains in both head and tail segments,
 856 demonstrating effective representation alignment and knowledge propagation across levels of data
 857 sparsity.

Table 4: The ablation study on the Yelp and Amazon Beauty dataset with SASRec as the backbone SRS model. The boldface refers to the highest score and the underline indicates the next best result of the models.

Dataset	Row	DB	FF	FB	Overall		Head User		Tail User		Head Item		Tail Item	
					HR@10	ND@10	HR@10	ND@10	HR@10	ND@10	HR@10	ND@10	HR@10	ND@10
Yelp	1				0.5866	0.3536	0.5945	0.3585	0.5848	0.3591	0.8002	0.4888	0.0890	0.0386
	2		✓		0.6181	0.3799	0.6135	0.3784	0.6142	0.3807	0.8055	0.4941	0.1061	0.0582
	3		✓	✓	0.6313	0.3974	0.6237	0.3897	0.6311	0.4008	0.8130	0.5031	0.1440	0.0874
	4	✓	✓		0.6719	0.4225	0.6440	0.4065	0.6738	0.4238	0.8243	0.5206	0.3200	0.1797
	5	✓	✓	✓	0.6827	0.4231	0.6583	<u>0.4027</u>	0.6884	0.4280	0.8361	0.5261	0.3252	0.1832
Beauty	1				0.4488	0.2861	0.5261	0.3611	0.4236	0.2690	0.7187	0.4815	0.1593	0.0856
	2		✓		0.4606	0.2977	0.5461	0.3873	0.4367	0.2875	0.7195	0.4844	0.1872	0.1083
	3		✓	✓	0.4826	0.3209	0.5522	0.3907	0.4898	0.3216	0.7257	0.5026	0.2152	0.1439
	4	✓	✓		0.5198	0.3417	0.5564	0.3965	0.5047	0.3402	0.7271	0.5125	0.2612	0.1712
	5	✓	✓	✓	0.5281	0.3665	0.5799	0.4148	0.5163	0.3557	0.7306	0.5229	0.3203	0.2060

D Limitation

Two potential limitations of this work should be noted. First, the proposed framework involves a number of hyperparameters, and identifying optimal configurations for specific tasks may require considerable tuning effort. Second, the subset partitioning algorithm in the Distribution Balancing module incurs relatively high computational complexity, making it more suitable for scenarios with a moderate number of items. In practical applications with large-scale item sets, further optimization or integration with incremental learning strategies may be necessary.