

Align, Don't Fuse: Structure-Preserving Alignment of Collaborative and Semantic Signals for LLM-Enhanced Recommendation

Jooweon Choi¹, MiHyeon Kim³, Eunju Lee²
JuneHyoung Kwon¹, YoungBin Kim^{1,2}

¹Department of Artificial Intelligence, Chung-Ang University

²Graduate School of Advanced Imaging Sciences, Multimedia and Film, Chung-Ang University

³KT Corporation

{jwjcwoi0910,dmswn5829,dirchdmltnv,ybkim85}@cau.ac.kr, mihyeon.gim@kt.com

Abstract

Large Language Models (LLMs) provide rich semantic representations for recommender systems, but their semantic similarity may not align with collaborative preference patterns captured by collaborative filtering (CF). Directly combining LLM and CF representations can therefore distort the structural relationships encoded in the interaction graph. We propose **Structure-Preserving Alignment (SPA)**, which reformulates CF-LLM integration as language-supervised representation alignment. SPA derives collaborative anchors from frozen CF representations and aligns multiple LLM-generated semantic views toward these anchors through cross-profile view competition. The aligned collaborative representations are then used for structure-preserving two-channel retrieval, which constructs behavior-aware context for downstream LLM-based preference prediction. Experiments on multiple datasets and CF backbones show that SPA consistently improves click-through rate (CTR)-style binary preference prediction.

1 Introduction

Large Language Models (LLMs) have recently opened new opportunities for recommender systems by transforming textual descriptions of users and items into rich semantic representations (Wu et al., 2024; Zhu et al., 2025). These language-derived representations capture high-level knowledge about user interests, item attributes, and latent intent, providing contextual signals that are not explicitly encoded in traditional ID-based recommendation models (Tsai et al., 2024; Kim et al., 2024; Lin et al., 2025). Such semantic knowledge is particularly valuable under limited downstream supervision, where interaction data alone may be insufficient to reliably estimate user preferences. These observations suggest that language can serve as a powerful auxiliary supervision signal for recommendation models.

Despite this promise, incorporating LLM-derived semantics into collaborative recommendation remains challenging. Collaborative filtering (CF) embeddings are learned from user-item interaction structures and therefore capture behavioral similarity induced by the interaction graph (Wang et al., 2019; He et al., 2020; Choi et al., 2025). In contrast, LLM embeddings are optimized to capture linguistic and conceptual similarity in a semantic space (Wu et al., 2024; Tsai et al., 2024). Because these representations are learned from fundamentally different signals, semantic similarity does not necessarily correspond to collaborative preference similarity. Directly combining the two representations can therefore weaken the collaborative relationships that make CF effective.

To bridge this gap, recent methods have explored integrating collaborative signals with language-derived representations or LLM-based semantic information (Xu et al., 2024; Ren and Huang, 2025; Liu et al., 2025). However, existing approaches generally focus on incorporating semantic information into recommendation rather than explicitly preserving the collaborative neighborhood structure during semantic integration.

We argue that integrating language signals into recommendation should be treated as a representation alignment problem rather than a feature fusion problem. Instead of directly mixing heterogeneous representations, language-derived semantics should guide collaborative representation learning while preserving the behavioral relationships captured by CF, especially when multiple LLM-generated descriptions provide complementary yet potentially inconsistent semantic views.

To address these challenges, we propose **Structure-Preserving Alignment (SPA)**, a framework that integrates language-derived semantics into collaborative recommendation through language-supervised representation alignment. SPA treats collaborative representations from a

frozen CF encoder as structural anchors and uses multi-view semantic profiles generated by an LLM as training-time supervision signals. Rather than using contrastive learning as a generic representation-learning objective, SPA uses asymmetric alignment to pull semantic views toward collaborative anchors derived from frozen CF representations, while preventing semantic supervision from updating the collaborative encoder. In addition, SPA introduces cross-profile view competition across alternative semantic views of the same entity, providing rotating multi-view regularization over complementary LLM-generated profiles.

After alignment, the projected collaborative representations are used for structure-preserving two-channel retrieval, which constructs behavior-aware context for downstream LLM-based preference prediction. During inference, SPA uses only projected collaborative representations for retrieval, while semantic profiles are used only during alignment training. This design connects CF-anchored alignment with retrieval-based LLM prediction while empirically retaining local collaborative structure in the projected representations, without requiring semantic profiles at inference time.

Extensive experiments across multiple CF backbones and few-shot LLM fine-tuning settings demonstrate that SPA consistently improves click-through rate (CTR) prediction performance. Following prior recommendation studies that formulate CTR prediction as a standard recommendation task (Yuan et al., 2022), we evaluate SPA under a CTR prediction setting, formulated as binary user-item preference prediction, where the task is to infer whether a user will prefer a target item rather than ranking a candidate set.

The contributions of this work are summarized as follows:

- We propose SPA, a framework that reformulates CF-LLM integration as language-supervised representation alignment, where frozen collaborative embeddings provide stable structural references and semantic supervision is introduced only through trainable projection networks. We evaluate structure retention explicitly through neighborhood- and correlation-based metrics.
- We introduce a CF-anchored contrastive alignment formulation that uses multiple LLM-generated user/item profiles as semantic supervision and incorporates cross-profile view

competition across alternative views of the same entity, providing rotating same-entity multi-view regularization across complementary semantic descriptions.

- We connect the aligned collaborative space to structure-preserving two-channel retrieval for downstream LLM-based preference prediction, leading to consistent improvements across multiple CF backbones and few-shot fine-tuning settings.

2 Related Work

2.1 LLM-based Recommendation and Text-Behavior Alignment

Recent advances in LLMs have influenced recommender systems by enabling semantic reasoning over textual descriptions of users and items. ReLLa (Lin et al., 2024) constructs retrieval-enhanced prompts from user interaction histories for LLM-based few-shot preference prediction, TALLRec (Bao et al., 2023) adapts LLMs through instruction tuning, and BIGRec (Bao et al., 2025) investigates generative preference modeling. GPT4Rec (Li et al., 2023) explores generative retrieval for recommendation, while LLM-KT (Wang et al., 2026) studies multi-level alignment between interaction representations and LLMs.

Despite these advances, aligning language-derived representations with behavioral representations remains challenging because the two representation types are learned under different objectives and reside in heterogeneous latent spaces. In contrast, our work focuses on structure-preserving alignment, where collaborative embeddings serve as structural anchors and LLM-derived semantic views provide training-time supervision.

2.2 CTR Prediction Models

CTR prediction is a fundamental recommendation task that estimates the probability of user interactions such as clicks or purchases. Prior CTR models include feature-interaction methods such as Wide & Deep (Cheng et al., 2016), AFM (Xiao et al., 2017), DeepFM (Guo et al., 2017), and DCN (Wang et al., 2017), as well as recent user-preference modeling approaches such as CREU (Xiang et al., 2025) and MWUF (Zhu et al., 2021). Our work complements these models by introducing CF-anchored alignment that incorporates LLM-derived semantic knowledge while

preserving local collaborative neighborhoods for interaction-level preference prediction.

3 Method

We propose **Structure-Preserving Alignment (SPA)**, a framework that integrates language-derived semantic knowledge into collaborative recommendation, as shown in Figure 1, while better preserving local collaborative neighborhoods learned from user–item interactions. Rather than directly fusing CF and semantic representations, SPA uses collaborative representations as structural references and aligns LLM-derived semantic views toward their projected collaborative anchors using in-batch contrast and cross-profile view competition. Semantic profiles are therefore used only as training-time alignment supervision, while inference uses only projected collaborative representations.

The framework is organized into four stages: (i) multi-view semantic profile generation and collaborative anchor construction, (ii) cross-profile semantic alignment, (iii) structure-preserving two-channel retrieval, and (iv) LLM-based preference prediction. SPA follows a staged pipeline rather than a single end-to-end differentiable objective. The CF backbone is first trained and frozen; projection networks are optimized via contrastive alignment; a discrete Top- K retrieval module constructs behavioral context outside the gradient path of the alignment objective; and the LLM predictor is fine-tuned separately on the Yes/No preference objective. The overall algorithm is provided in Appendix A.

3.1 Multi-View Semantic Profile Generation

To introduce semantic knowledge beyond interaction data, we generate multiple semantic views for users and items using a large language model. Specifically, we employ Gemini-2.5-Flash (Comanici et al., 2025) to produce structured semantic descriptions conditioned on user interaction histories and item metadata. These LLM-generated views provide complementary semantic interpretations of user preferences and item characteristics, which later serve as supervision signals for the alignment stage.

User semantic views. Let $\mathcal{U}$ denote the set of users and $\mathcal{I}$ denote the set of items. Let $\mathcal{H}_u^{\text{train}}$ denote the profiling history of user u with interaction feedback and item metadata, and let $\mathcal{H}_{u,t}$ denote

the target-specific history excluding the target interaction and label.

Using a prompt template $\mathcal{P}_u$, the LLM generates multiple textual views from the user-level profiling history $\mathcal{H}_u^{\text{train}}$:

$$\mathcal{S}_u = \{s_u^{(v)}\}_{v \in \mathcal{V}_u} = \text{LLM}(\mathcal{P}_u, \mathcal{H}_u^{\text{train}}). \quad (1)$$

These semantic profiles are generated once per user and cached for semantic alignment; they are not regenerated separately for individual user–target prediction instances. Here, $\mathcal{V}_u$ denotes the set of semantic views used for user profiling. In our implementation, $|\mathcal{V}_u| = 6$, representing textual descriptions derived from three behavioral perspectives (recent, early, and global), each expressed in two complementary forms: analysis and summary. These semantic views capture diverse interpretations of user preference behavior.

Item semantic views. For each item $j \in \mathcal{I}$, we generate multiple textual views using item metadata. Let $\mathcal{Q}_j$ denote the metadata associated with item j , such as title, category, or textual description.

Using a prompt template $\mathcal{P}_j$, the LLM generates semantic item views:

$$\mathcal{T}_j = \{t_j^{(v)}\}_{v \in \mathcal{V}_j} = \text{LLM}(\mathcal{P}_j, \mathcal{Q}_j). \quad (2)$$

Likewise, item semantic profiles are generated once from item metadata and cached for semantic alignment rather than regenerated for individual prediction instances. Here, $\mathcal{V}_j$ denotes the set of semantic views used for item profiling. In our implementation, we set $|\mathcal{V}_j| = 3$. These views represent different semantic perspectives of the same item. Detailed view definitions and prompt designs are provided in Appendices E and H, respectively.

Semantic encoding. The generated textual views are converted into dense semantic representations using a frozen pretrained encoder $\psi(\cdot)$ implemented with mean pooling over the final-layer hidden states of Llama-3.1-8B-Instruct (Grattafiori et al., 2024). The semantic encoder is used only for offline representation extraction and remains frozen throughout SPA training; the downstream prediction model is fine-tuned separately with LoRA (Hu et al., 2022).

$$\mathbf{z}_{u,v}^{\text{txt}} = \psi(s_u^{(v)}), \quad \mathbf{z}_{j,v}^{\text{txt}} = \psi(t_j^{(v)}). \quad (3)$$

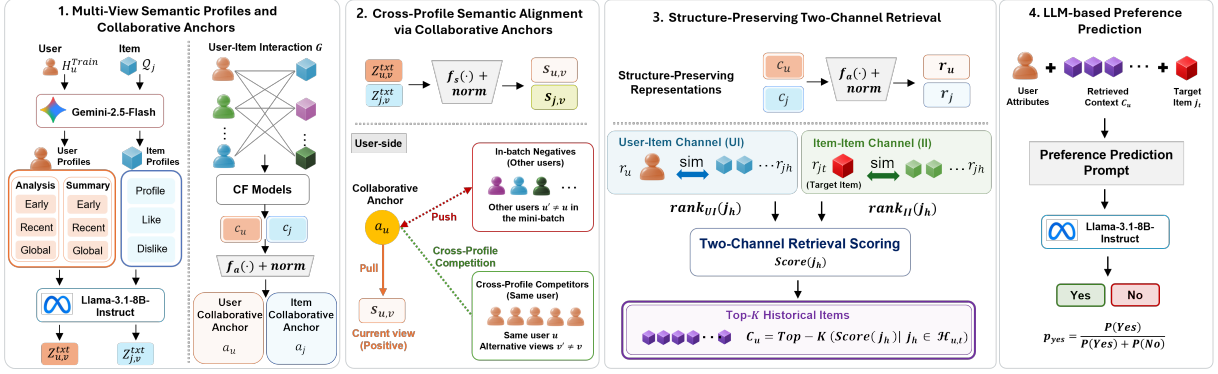


Figure 1: Overview of SPA as a four-stage pipeline. Given user–item interactions, SPA first generates multi-view semantic profiles and collaborative embeddings, then aligns semantic representations to collaborative anchors using in-batch negatives and cross-profile view competition. The aligned representations are used for structure-preserving two-channel retrieval, which constructs behavior-aware context for downstream LLM-based preference prediction.

Here, $z_{u,v}^{\text{txt}}$ and $z_{j,v}^{\text{txt}}$ denote the semantic embeddings corresponding to the v -th textual views of user u and item j , respectively, and both lie in the semantic representation space $\mathbb{R}^{d_s}$, where d_s denotes the semantic embedding dimension.

For each user and item, we collect the encoded semantic views as $\mathcal{Z}_u = \{z_{u,v}^{\text{txt}} \mid v \in \mathcal{V}_u\}$ and $\mathcal{Z}_j = \{z_{j,v}^{\text{txt}} \mid v \in \mathcal{V}_j\}$. These multi-view embeddings are used as supervision signals in the subsequent alignment stage.

3.2 Collaborative Anchor Construction

We obtain collaborative representations from the user–item interaction graph using a graph-based CF backbone. Let the collaborative training graph be $\mathcal{G} = (\mathcal{U}, \mathcal{I}, \mathcal{E})$, where $\mathcal{E}$ contains interactions available to collaborative representation learning under the dataset-specific construction described in Appendix B.1. Held-out target interactions are excluded from $\mathcal{E}$. A CF encoder $f_{\text{CF}}(\cdot)$ such as LightGCN (He et al., 2020), AutoCF (Xia et al., 2023), or SGL (Wu et al., 2021) learns latent embeddings for both users and items:

$$\mathbf{c}_u = f_{\text{CF}}(u), \mathbf{c}_j = f_{\text{CF}}(j), \quad (4)$$

where $\mathbf{c}_u, \mathbf{c}_j \in \mathbb{R}^{d_c}$ denote collaborative embeddings capturing behavioral similarity induced by the interaction topology, with d_c denoting the collaborative embedding dimension.

During alignment, the CF encoder is frozen and only the projection networks are optimized. This prevents semantic supervision from updating the underlying CF encoder and keeps the original collaborative embeddings as stable references during alignment, thereby isolating the effect of language-derived supervision. Thus, the frozen collaborative

embeddings $\mathbf{c}_u$ and $\mathbf{c}_j$ provide stable structural references during alignment. The projected anchors used in the contrastive objective are produced by the trainable projection network $f_a(\cdot)$; therefore, the raw CF representations remain fixed while the projection space can adapt to semantic supervision.

$$\mathbf{a}_u = \text{norm}(f_a(\mathbf{c}_u)), \mathbf{a}_j = \text{norm}(f_a(\mathbf{c}_j)). \quad (5)$$

The projected vectors $\mathbf{a}_u$ and $\mathbf{a}_j$ define collaborative anchor representations that serve as reference points for subsequent semantic alignment. This design restricts semantic adaptation to the projection space while leaving the original CF embeddings unchanged; the extent to which the projected representations retain collaborative structure is evaluated empirically in Appendix J.

3.3 Cross-Profile Semantic Alignment via Collaborative Anchors

We treat collaborative representations as anchors and semantic representations as auxiliary views, resulting in an asymmetric alignment formulation. Rather than directly fusing heterogeneous embeddings, SPA aligns semantic views with collaborative anchors through a contrastive learning objective. User-side and item-side alignment losses follow the same objective form and optimize a shared collaborative projection space, allowing projected user and item representations to be directly compared during retrieval. For clarity, we present the user-side formulation below; the item-side formulation is provided in Appendix G.

Semantic View Representations. Let $z_{u,v}^{\text{txt}}$ and $z_{j,v}^{\text{txt}}$ denote semantic embeddings obtained from

the textual profiles generated by the LLM, where v indexes semantic views. To compare semantic views with collaborative anchors in a shared alignment space, we project each semantic embedding through a semantic projection network $f_s(\cdot)$:

$$\mathbf{s}_{u,v} = \text{norm}(f_s(\mathbf{z}_{u,v}^{txt})), \mathbf{s}_{j,v} = \text{norm}(f_s(\mathbf{z}_{j,v}^{txt})). \quad (6)$$

Both collaborative anchors and semantic projections are ℓ_2 normalized so that similarity comparisons can be performed using cosine similarity.

Positive, In-Batch Negative, and Cross-Profile Competitor Construction. For each semantic view v , we construct a positive pair between the collaborative anchor and the corresponding semantic representation of the same entity, e.g., $(\mathbf{a}_u, \mathbf{s}_{u,v})$ for users and $(\mathbf{a}_j, \mathbf{s}_{j,v})$ for items. To learn meaningful alignment, we use two types of contrastive counterparts: in-batch negatives, which are semantic representations from other entities sampled within the same shuffled mini-batch, and cross-profile competitors, which correspond to alternative semantic views of the same entity. Importantly, cross-profile competitors are not permanent negatives. During an epoch, each semantic view is selected as the positive in its own entity-view training instance while participating as a cross-profile competitor for the remaining view-specific instances. This rotating optimization introduces same-entity multi-view competition across complementary semantic descriptions.

Contrastive Alignment Objective. Let $\mathcal{B}$ denote a shuffled mini-batch of entity-view training instances, where each training instance is represented as (u, v) consisting of a user u and one of its semantic views v . Accordingly, $(u_b, v_b) \in \mathcal{B}$ denotes an arbitrary entity-view training instance sampled from the same mini-batch. Let $\mathcal{V}_u$ denote the set of semantic views associated with user u . For each training instance $(u, v) \in \mathcal{B}$, the objective encourages the semantic representation $\mathbf{s}_{u,v}$ to align with its collaborative anchor $\mathbf{a}_u$, contrasts it against entity-view instances of other users in the mini-batch, and incorporates the remaining semantic views of the same user as cross-profile competitors within the view-specific normalization.

We define the in-batch negative term as

$$D_{u,v}^{\text{in}} = \sum_{\substack{(u_b, v_b) \in \mathcal{B} \\ u_b \neq u}} \exp(\text{sim}(\mathbf{a}_u, \mathbf{s}_{u_b, v_b})/\tau). \quad (7)$$

We define the cross-profile competition term as

$$D_{u,v}^{\text{cp}} = \sum_{v' \in \mathcal{V}_u \setminus \{v\}} \exp(\text{sim}(\mathbf{a}_u, \mathbf{s}_{u, v'})/\tau), \quad (8)$$

and the positive pair term as

$$D_{u,v}^{\text{pos}} = \exp(\text{sim}(\mathbf{a}_u, \mathbf{s}_{u,v})/\tau). \quad (9)$$

The user-side alignment loss is then defined as

$$\mathcal{L}_{\text{align}}^{\text{user}} = -\frac{1}{|\mathcal{B}|} \sum_{(u,v) \in \mathcal{B}} \log \frac{D_{u,v}^{\text{pos}}}{D_{u,v}^{\text{pos}} + D_{u,v}^{\text{in}} + D_{u,v}^{\text{cp}}}. \quad (10)$$

Here, $\text{sim}(\cdot, \cdot)$ denotes cosine similarity, and the denominator consists of the positive, in-batch negative, and cross-profile competition terms. The in-batch term captures entity-view training instances of other users in the shuffled mini-batch, whereas the cross-profile term incorporates the remaining semantic views of the same user into the view-specific contrastive normalization. These alternative views act as cross-profile competitors for the current training instance, but are not assigned permanent negative roles because each view becomes the positive in its own instance. Across the view-specific losses, this rotating competition regularizes the relative alignment of complementary semantic views around the same collaborative anchor. The objective does not explicitly enforce pairwise semantic separation among same-entity views; alternative views participate only as competitors within the current view-specific normalization, while each view remains a positive in its own training instance.

Structure-Preserving Representation. After alignment, inference uses only the projected collaborative representations:

$$\mathbf{r}_u = \text{norm}(f_a(\mathbf{c}_u)), \mathbf{r}_j = \text{norm}(f_a(\mathbf{c}_j)). \quad (11)$$

These inference representations are obtained by applying the trained collaborative projection network $f_a(\cdot)$ to the frozen CF embeddings. Although semantic embeddings are not used at inference, they guide representation learning during training, allowing the final representations to better preserve collaborative structure while reflecting language-derived supervision.

3.4 Structure-Preserving Two-Channel Retrieval.

After semantic alignment, SPA obtains projected collaborative representations for users and items.

Let $\mathbf{r}_u$ and $\mathbf{r}_j$ denote the aligned representations for user u and item j , respectively. At inference, the collaborative projections $f_a(\cdot)$ map both user and item CF embeddings into the same alignment space used for retrieval, whereas semantic projections $f_s(\cdot)$ are used only during alignment training.

Given a user u , a target item j_t , and the target-specific history $\mathcal{H}_{u,t}$, SPA constructs a compact behavioral context through a two-channel retrieval strategy. The first channel captures global preference affinity by measuring similarity between the user representation and historical items: $s_{UI}(j_h) = \text{sim}(\mathbf{r}_u, \mathbf{r}_{j_h})$. The second channel captures target-specific contextual relevance by measuring similarity between the target item and historical items: $s_{II}(j_h) = \text{sim}(\mathbf{r}_{j_t}, \mathbf{r}_{j_h})$. Instead of combining raw similarity scores directly, SPA aggregates ranking signals from the two channels, as their similarity-score distributions differ across the user-item and item-item channels. Let $\text{rank}_{UI}(j_h)$ and $\text{rank}_{II}(j_h)$ denote the zero-based rank positions under the two channels, such that the highest-ranked historical item has rank 0, and let λ denote the channel weight. The final retrieval score is defined as

$$\text{Score}(j_h) = (1 - \lambda) \frac{1}{\text{rank}_{UI}(j_h) + 1} + \lambda \frac{1}{\text{rank}_{II}(j_h) + 1}. \quad (12)$$

The Top- K historical items according to $\text{Score}(\cdot)$ form the retrieval context:

$$\mathcal{C}_u = \text{Top-}K(\text{Score}(j_h) \mid j_h \in \mathcal{H}_{u,t}). \quad (13)$$

The user-item channel captures global collaborative preference patterns, whereas the item-item channel emphasizes target-specific contextual relevance within the user’s interaction history. By combining both signals, SPA constructs a compact yet target-aware behavioral context $\mathcal{C}_u$ that summarizes the most informative historical evidence for downstream preference prediction.

Because the Top- K retrieval operation is discrete, it is used only to construct the behavioral context for downstream LLM-based prediction and is outside the gradient path of the alignment objective.

3.5 LLM-based Preference Prediction

After the retrieval stage, the retrieved historical items form the collaborative retrieval context $\mathcal{C}_u$. This context summarizes behaviorally relevant interactions for the user with respect to the target

item. We convert $\mathcal{C}_u$ into a prompt, which is used for downstream LLM-based preference prediction.

The instruction-tuned model is based on Llama-3.1-8B-Instruct (Grattafiori et al., 2024). Given a prompt containing available user attributes, the constructed history context, and the target item, the model predicts whether the user will prefer the target item. User attributes are included only when provided by the corresponding dataset. Following the instruction format of ReLLa (Lin et al., 2024), the model outputs a binary response (Yes/No), which is used for CTR prediction. Details of prompt construction and mixed-dataset training are provided in Appendix I. For evaluation, we convert the Yes/No output distribution into a predicted preference probability: $p_{\text{yes}} = p(\text{Yes}) / (p(\text{Yes}) + p(\text{No}))$. Area Under the ROC Curve (AUC) and Log Loss are computed using p_{yes} , and Accuracy (ACC) is computed by thresholding it at 0.5.

4 Experiments

We conduct experiments to answer the following research questions:

RQ1: Does SPA outperform strong CTR baselines and existing LLM-based recommendation models? **RQ2:** How important is cross-profile contrastive alignment for learning behavior-consistent representations? **RQ3:** How sensitive is SPA to key semantic and retrieval configurations? **RQ4:** Is SPA robust across different graph-based CF backbones (LightGCN, AutoCF, SGL)?

4.1 Experimental Settings

Datasets. We evaluate SPA on three widely used recommendation benchmarks: MovieLens-1M (ML-1M) (Harper and Konstan, 2015), BookCrossing (Ziegler et al., 2005), and Amazon-Movies (Hou et al., 2026). Following prior LLM-based recommendation protocols (Lin et al., 2024; Bao et al., 2023), we retain users with at least five historical interactions. Dataset statistics, label construction, sample construction, and feature fields are summarized in Appendix B.

Evaluation Metrics. We report AUC, Log Loss, and ACC. AUC evaluates discrimination between positive and negative instances, Log Loss measures probabilistic fit, and ACC reflects classification correctness. Following CTR practice (Guo et al., 2017; Cheng et al., 2016), even small AUC improvements

Table 1: (RQ1) Performance comparison on three datasets. Bold AUC values indicate the best overall results, and underlined values indicate the strongest non-SPA baseline. Rel.Impr denotes SPA’s relative AUC improvement over each matched baseline and is reported only for few-shot LLM-based models. Full-shot CTR comparisons are contextual due to different information budgets.

Model	ML-IM				BookCrossing				Amazon-Movies				
	AUC↑	Log Loss↓	ACC↑	Rel.Impr	AUC↑	Log Loss↓	ACC↑	Rel.Impr	AUC↑	Log Loss↓	ACC↑	Rel.Impr	
Traditional CTR Models (Full Training Data)													
Full-shot	WD	0.5305	0.6996	0.5382	–	0.5171	0.5376	0.5376	–	0.7172	0.5352	0.7035	–
	DeepFM	0.7432	0.6680	0.6756	–	0.6391	0.6883	0.5823	–	0.7273	0.3041	0.6275	–
	AFM	0.5113	0.6978	0.5367	–	0.5080	0.6910	0.4776	–	0.5725	0.3327	0.5705	–
	DCN	0.7276	0.7369	0.6645	–	0.5956	0.6903	0.5788	–	<u>0.7399</u>	0.2812	0.6393	–
	MWUF	<u>0.7435</u>	0.6722	0.6751	–	<u>0.6509</u>	0.6882	0.5872	–	0.7313	0.2943	0.7195	–
	CREU	0.7434	0.6533	0.6752	–	0.6412	0.6876	0.5827	–	0.7315	0.3002	0.7134	–
LLM-based Models (Few-shot LLM Fine-tuning)													
Few-shot 1%	ReLLa	<u>0.6734</u>	0.6563	0.6202	17.95%	<u>0.6786</u>	0.6607	0.5905	5.16%	<u>0.7217</u>	0.4075	0.8516	10.36%
	TALLRec	0.5536	0.6847	0.5424	43.48%	0.6421	0.6520	0.6182	11.14%	0.6716	0.5293	0.7887	18.60%
	BIGRec	0.5540	0.6841	0.5505	43.38%	0.6007	0.7298	0.5292	18.79%	0.7187	0.3980	0.8345	10.83%
	SPA	0.7943	0.5603	0.7179	–	0.7136	0.6497	0.6258	–	0.7965	0.3242	0.8741	–
Few-shot 10%	ReLLa	<u>0.6870</u>	0.6650	0.6281	16.72%	0.7117	0.6169	0.6607	2.80%	<u>0.7658</u>	0.3564	0.8747	6.37%
	TALLRec	0.6007	0.6809	0.5704	33.49%	0.6963	0.6495	0.6382	5.07%	0.7446	0.3427	0.8602	9.40%
	BIGRec	0.5909	0.6845	0.5625	35.71%	<u>0.7221</u>	0.6004	0.6617	1.32%	0.7254	0.3355	0.8759	12.30%
	SPA	0.8019	0.5377	0.7269	–	0.7316	0.6446	0.6349	–	0.8146	0.3103	0.8816	–

are considered meaningful. Detailed metric definitions are provided in Appendix C.

Baselines. We compare SPA with representative CTR models—WD (Cheng et al., 2016), DeepFM (Guo et al., 2017), AFM (Xiao et al., 2017), DCN (Wang et al., 2017), MWUF (Zhu et al., 2021), and CREU (Xiang et al., 2025)—and LLM-based recommenders, including ReLLa (Lin et al., 2024), TALLRec (Bao et al., 2023), and BIGRec (Bao et al., 2025). We use the official implementations of the baselines and adapt them to the same processed train/test splits, binary preference labels, and evaluation protocol used for SPA. For ReLLa, TALLRec, and BIGRec, we additionally use Llama-3.1-8B-Instruct as a common backbone and the same 1%/10% downstream fine-tuning subsets, while retaining each method’s original input construction and training configuration unless otherwise stated.

Implementation Details. Experiments are conducted on NVIDIA A100 GPUs. We use Llama-3.1-8B-Instruct (Grattafiori et al., 2024) in two distinct roles: a frozen semantic encoder for offline profile encoding and a downstream preference predictor fine-tuned with LoRA.

For the default Llama-3.1-8B-Instruct semantic encoder, the semantic embedding dimension is $d_s = 4096$. The CF embedding dimension is $d_c = 64$, and the projection output dimension is $d_p = 256$. The projection networks use a two-layer architecture with hidden dimension 512 and ReLU activation. We set the contrastive temperature to $\tau = 0.07$ and the semantic perturbation magnitude

to $\epsilon = 0.1$. During alignment training, semantic embeddings are perturbed using sign-preserving normalized Gaussian noise before being passed to the semantic projection network.

The projection networks are optimized using Adam with a learning rate of 1×10^{-3} , weight decay 1×10^{-5} , and batch size 2048. Alignment is trained for 50 epochs. The CF encoder and semantic representations remain fixed during alignment; only the lightweight projection networks are optimized.

For downstream preference prediction, LoRA uses rank $r=16$, $\alpha=32$, and dropout 0.05. The downstream predictor is optimized using AdamW with a learning rate of 5×10^{-4} , batch size 256, and 10 training epochs. Unless otherwise specified, hyperparameters were fixed globally across all datasets before test-set evaluation and were not tuned on the test set or separately optimized for individual datasets. The 1% and 10% few-shot settings apply only to downstream LLM fine-tuning instances. CF training, semantic profiling, and alignment use the non-held-out interactions permitted by the dataset-specific construction described in Appendix B.1; no held-out evaluation target or label is used by these stages. Detailed offline costs for semantic profile generation, semantic encoding, and alignment training are reported in Appendix P.

4.2 Experimental Results

(RQ1) Performance Comparison. Table 1 reports the performance comparison between SPA, traditional CTR models, and recent LLM-based recommendation approaches. Because preprocess-

ing, label construction, few-shot sampling, and evaluation protocols vary across studies, reported scores may differ from those in prior work. For the LLM-based comparisons, we adapt the official baseline implementations to the common processed splits, downstream fine-tuning budgets, and binary user-item preference prediction protocol described above.

SPA achieves the highest AUC across all three datasets under both few-shot regimes. For RQ1, both the sequential-history context length and the Top- K collaborative retrieval size are set to 15. In the 1% setting, we use AutoCF for all datasets; in the 10% setting, we use AutoCF for ML-1M and BookCrossing and SGL for Amazon-Movies.

First, SPA achieves performance competitive with or stronger than several full-shot CTR baselines. However, because SPA leverages pretrained LLM-derived semantic knowledge whereas traditional CTR models rely mainly on interaction data and structured features, this comparison should be interpreted as contextual evidence under different information budgets rather than as a controlled efficiency comparison. For example, on ML-1M, the strongest full-shot CTR baseline MWUF achieves an AUC of 0.7435, whereas SPA reaches 0.8019 using only 10% of the LLM fine-tuning prediction instances. Similar trends are observed on BookCrossing and Amazon-Movies, suggesting that LLM-derived semantic supervision remains beneficial when aligned with collaborative anchors, even with limited downstream LLM fine-tuning data.

Second, SPA consistently improves over existing LLM-based recommenders under few-shot LLM fine-tuning settings. In the 1% few-shot regime, SPA achieves the best results across all datasets. On ML-1M, SPA improves the AUC from 0.6734 (ReLLa) to 0.7943, with consistent improvements observed across the other datasets. These results show that the full SPA pipeline benefits from combining multi-view semantic supervision with collaborative representations under limited downstream LLM fine-tuning; the contribution of cross-profile competition is isolated separately in RQ2.

Overall, these results suggest that SPA benefits from using LLM-derived semantics as alignment supervision rather than directly fusing heterogeneous representations.

(RQ2) Ablation Study. Table 2 presents an ablation study under the 1% few-shot setting with

Table 2: (RQ2) Ablation study under the 1% few-shot setting with $K = 10$. We analyze the contribution of semantic supervision and cross-profile contrastive alignment in SPA.

Model	ML-1M AUC↑	BookCrossing AUC↑	Amazon-Movies AUC↑
<i>Single-Modality Baselines</i>			
CF-only	0.7773	0.6938	0.7552
Text-only	0.7591	0.7099	0.7595
<i>Fusion Baseline</i>			
Naive Fusion	0.7765	0.6240	0.7889
<i>SPA Variants</i>			
SPA w/o Cross-profile	0.7863	0.7068	0.7619
SPA w/o Augmentation	0.7876	0.7135	0.7923
SPA w/ Drop50	0.7846	0.7090	0.7914
SPA	0.7880	0.7164	0.7983

$K = 10$. We use the best-performing backbone for each dataset according to Appendix M: AutoCF for ML-1M and SGL for BookCrossing and Amazon-Movies. The single-modality baselines show that both collaborative and language-derived signals contain useful information for recommendation. The CF-only model performs well on ML-1M, while the Text-only model performs competitively on BookCrossing and Amazon-Movies. However, both remain inferior to SPA, indicating that integrating behavioral structure with language-derived semantics is important for robust recommendation.

The naive fusion baseline, which directly concatenates CF and LLM embeddings, performs substantially worse than SPA, suggesting that such direct combination is insufficient and may even degrade performance due to the structural mismatch between the two representation spaces. Thus, Naive Fusion serves as a targeted control for testing the fusion-vs-alignment hypothesis rather than an exhaustive representative of all fusion-based LLM recommenders.

Removing cross-profile competition (SPA w/o Cross-profile) consistently reduces performance across datasets, indicating that cross-profile view competition provides effective multi-view regularization across complementary semantic descriptions.

The robustness variants further demonstrate the stability of SPA. Removing semantic augmentation (SPA w/o Augmentation), i.e., disabling additional perturbation on semantic embeddings, slightly degrades performance, while Drop50, which randomly drops 50% of semantic embedding features, shows that SPA remains effective even under substantial semantic feature removal. This indicates

that SPA learns robust collaborative embeddings guided by semantic supervision, rather than relying on fragile or noisy textual cues.

Overall, the full SPA model consistently achieves the best performance across all datasets. These results suggest that SPA’s gains arise from the synergistic interaction among multi-view semantic supervision, CF-anchored alignment, and behavior-aware retrieval, rather than from a single isolated component.

To further understand how SPA reshapes the representation space, we visualize the learned embeddings using t-SNE and quantitatively evaluate structure preservation using Top-10 neighborhood overlap, Spearman’s ρ , and Pearson’s r ; detailed results are provided in Appendix J. Additional controlled comparisons against Gated Fusion, CLIP-style Alignment, and KD-style Alignment under a unified protocol are reported in Appendix L.

(RQ3) Configuration Sensitivity. To assess the robustness of semantic, alignment, and retrieval-related design choices, we evaluate SPA under different LLM semantic encoders (Llama-3.1-8B-Instruct, Qwen2.5-7B-Instruct (Yang et al., 2024), and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023)), semantic-view configurations, two-channel retrieval weights λ , and contrastive temperatures τ . For retrieval, $\lambda = 0$ uses only the user-item channel, capturing global preference but ignoring target-specific relevance, whereas $\lambda = 1$ uses only the item-item channel, emphasizing local relevance but losing user-specific signals; the best performance is achieved around $\lambda = 0.6$, showing that combining both channels provides the most informative behavioral context. For semantic encoders, SPA remains effective across Llama, Qwen, and Mistral, demonstrating the alignment mechanism’s flexibility across diverse latent spaces. Although Llama-3.1-8B-Instruct achieves the strongest overall results, the performance gaps vary across datasets, indicating that the quality of the semantic embedding space can materially affect downstream recommendation performance. For view configuration, performance improves with richer semantic views, indicating that diverse behavioral perspectives provide more robust alignment supervision than single-view descriptions. For alignment sharpness, SPA performs best around $\tau = 0.07$; lower temperatures may overemphasize specific textual features, whereas higher temperatures may weaken the structural sig-

nal from collaborative anchors. Full sensitivity results for the semantic encoder, semantic-view configuration, retrieval weight, contrastive temperature, and projection-network architecture are provided in Appendix K. Appendix N further compares raw cosine weighting, z-score calibrated aggregation, and the proposed reciprocal-rank aggregation.

(RQ4) Robustness Across Different CF Backbones. To evaluate the robustness of SPA, we compare its performance using three representative graph-based CF backbones: LightGCN, AutoCF, and SGL. As shown in Table 22 in Appendix M, we evaluate SPA across all datasets under both the 1% and 10% downstream LLM fine-tuning regimes with $K \in \{5, 10, 15\}$. SPA maintains strong performance across different backbones, datasets, fine-tuning regimes, and history lengths. Appendix O additionally examines SPA’s dependence on collaborative-anchor quality by intentionally undertraining the CF backbone.

5 Conclusion

In this work, we propose SPA, a structure-preserving alignment framework for integrating language-derived semantics into collaborative recommendation. Instead of directly fusing heterogeneous representations, SPA derives collaborative anchors from frozen CF representations and aligns multi-view semantic profiles using in-batch contrast and cross-profile view competition. This design enables language supervision to refine projected collaborative representations while empirically retaining local collaborative neighborhoods learned from interaction data. Together, CF-anchored alignment and two-channel retrieval improve CTR prediction under limited downstream supervision, showing an effective alternative to direct feature fusion for integrating language-derived semantics into recommendation.

6 Limitations

A broader challenge in language-supervised recommendation is that language-derived semantics can be noisy, incomplete, or temporally outdated, and their relationship to collaborative preference may vary across recommendation settings. In addition, LLM-generated semantic profiles may vary depending on prompt wording. Although SPA mitigates this issue by using multiple behavior-defined

semantic views, we do not claim invariance to arbitrary prompt formulations.

This work addresses a narrower question: whether language-derived supervision can improve collaborative recommendation when semantic signals are introduced through alignment to CF anchors, rather than being directly mixed into collaborative representations. Under this scope, SPA uses offline LLM-generated semantic profiles and freezes the collaborative encoder during alignment in order to isolate the effect of semantic supervision while better preserving local collaborative neighborhoods learned from user–item interactions. These choices leave open broader settings that require online semantic updates, fuller joint optimization between semantic and collaborative representations, and evaluation beyond offline CTR prediction, including top- N ranking, session-based recommendation, and strict cold-start generalization.

The current study evaluates SPA with graph-based CF backbones; extending the framework to sequential recommenders such as SASRec or BERT4Rec remains future work. SPA also depends on the quality of the collaborative anchor space. Our undertraining analysis shows that performance degrades when the underlying CF encoder is insufficiently optimized, particularly on BookCrossing and Amazon-Movies. Thus, SPA should not be interpreted as compensating for arbitrarily poor collaborative representations; robustness to severely degraded or cross-domain anchors remains an open direction.

Finally, SPA leverages pretrained LLM-derived semantic knowledge, whereas traditional CTR baselines rely mainly on interaction data and structured features. Therefore, comparisons with full-shot CTR models should be interpreted as contextual evidence under different information and computational budgets rather than as controlled efficiency comparisons. Moreover, the current few-shot protocol constrains only the downstream LLM fine-tuning stage, while CF training, semantic profile generation, and alignment use the dataset-specific non-held-out interaction data described in Appendix B.1. Evaluating SPA under end-to-end data-scarce conditions remains future work.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning

& Evaluation (IITP) grant funded by the Korea government (MSIT) [RS-2021-II211341, Artificial Intelligence Graduate School Program (Chung-Ang University)] and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00556246).

References

- Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yanchen Luo, Chong Chen, Fuli Feng, and Qi Tian. 2025. [A bi-step grounding paradigm for large language models in recommendation systems](#). *ACM Trans. Recomm. Syst.*, 3(4).
- Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. [TALLRec: An effective and efficient tuning framework to align large language model with recommendation](#). In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys '23*, page 1007–1014, New York, NY, USA. Association for Computing Machinery.
- Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishikesh Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. [Wide & Deep learning for recommender systems](#). In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS 2016*, page 7–10, New York, NY, USA. Association for Computing Machinery.
- Jooweon Choi, JuneHyoung Kwon, Yeonghwa Kim, and YoungBin Kim. 2025. [Hypergraph temporal multi-behavior recommendation](#). *Engineering Applications of Artificial Intelligence*, 145:110112.
- Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Naveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17*, page 1725–1731. AAAI Press.
- F. Maxwell Harper and Joseph A. Konstan. 2015. [The MovieLens datasets: History and context](#). *ACM Trans. Interact. Intell. Syst.*, 5(4).

- Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. [LightGCN: Simplifying and powering graph convolution network for recommendation](#). In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 639–648, New York, NY, USA. Association for Computing Machinery.
- Yupeng Hou, Jiacheng Li, Xiangjun Fu, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2026. [Bridging language and items for retrieval and recommendation: Benchmarking LLMs as semantic encoders](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3251–3265, San Diego, California, United States. Association for Computational Linguistics.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Giana Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. [Mistral 7B](#). *ArXiv*, abs/2310.06825.
- Sein Kim, Hongseok Kang, Seungyeon Choi, Donghyun Kim, Minchul Yang, and Chanyoung Park. 2024. [Large language models meet collaborative filtering: An efficient all-round LLM-based recommender system](#). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 1395–1406, New York, NY, USA. Association for Computing Machinery.
- Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and G  rard G. Medioni. 2023. [GPT4Rec: A generative framework for personalized recommendation and user interests interpretation](#). In *Proceedings of the 2023 SIGIR Workshop on eCommerce co-located with the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2023)*, Taipei, Taiwan, July 27, 2023, volume 3589 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Hao Zhang, Yong Liu, Chuhan Wu, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. 2025. [How can recommender systems benefit from large language models: A survey](#). *ACM Trans. Inf. Syst.*, 43(2).
- Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. [ReLLa: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation](#). In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 3497–3508, New York, NY, USA. Association for Computing Machinery.
- Yibin Liu, Ang Li, and Shijian Li. 2025. [Hierarchical interaction summarization and contrastive prompting for explainable recommendations](#). In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2025, Porto, Portugal, September 15–19, 2025, Proceedings, Part V*, page 485–501, Berlin, Heidelberg. Springer-Verlag.
- Xubin Ren and Chao Huang. 2025. [EasyRec: Simple yet effective language models for recommendation](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 17717–17732, Suzhou, China. Association for Computational Linguistics.
- Alicia Tsai, Adam Kraft, Long Jin, Chenwei Cai, Anahita Hosseini, Taibai Xu, Zemin Zhang, Lichan Hong, Ed H. Chi, and Xinyang Yi. 2024. [Leveraging LLM reasoning enhances personalized recommender systems](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13176–13188, Bangkok, Thailand. Association for Computational Linguistics.
- Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. [Deep & Cross Network for ad click predictions](#). In *Proceedings of the ADKDD'17*, ADKDD'17. Association for Computing Machinery, New York, NY, USA.
- Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. [Neural graph collaborative filtering](#). In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, page 165–174, New York, NY, USA. Association for Computing Machinery.
- Ziwei Wang, Jie Zhou, Qin Chen, Bo Jiang, Qingchun Bai, Liang Dou, and Liang He. 2026. [LLM-KT: Enhancing large language models with knowledge tracing via multi-level plug-and-play alignment](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 35775–35792, San Diego, California, United States. Association for Computational Linguistics.
- Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. [Self-supervised graph learning for recommendation](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 726–735, New York, NY, USA. Association for Computing Machinery.
- Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. 2024. [A survey on large language models for recommendation](#). *World Wide Web*, 27(5).

- Lianghao Xia, Chao Huang, Chunzhen Huang, Kangyi Lin, Tao Yu, and Ben Kao. 2023. [Automated self-supervised learning for recommendation](#). In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 992–1002, New York, NY, USA. Association for Computing Machinery.
- Yang Xiang, Li Fan, Chenke Yin, Menglin Kong, and Chengtao Ji. 2025. [Harnessing light for cold-start recommendations: Leveraging epistemic uncertainty to enhance performance in user-item interactions](#). In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, CIKM '25, page 5361–5365, New York, NY, USA. Association for Computing Machinery.
- Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 3119–3125. AAAI Press.
- Shuyuan Xu, Wenyue Hua, and Yongfeng Zhang. 2024. [OpenP5: An open-source platform for developing, training, and evaluating LLM-based recommender systems](#). In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 386–394, New York, NY, USA. Association for Computing Machinery.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Guanghu Yuan, Fajie Yuan, Yudong Li, Beibei Kong, Shujie Li, Lei Chen, Min Yang, Chenyun Yu, Bo Hu, Zang Li, Yu Xu, and Xiaohu Qie. 2022. TenRec: a large-scale multipurpose benchmark dataset for recommender systems. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.
- Xi Zhu, Yu Wang, Hang Gao, Wujiang Xu, Chen Wang, Zhiwei Liu, Kun Wang, Mingyu Jin, Linsey Pang, Qingsong Weng, Philip S. Yu, and Yongfeng Zhang. 2025. [Recommender systems meet large language model agents: A survey](#). *Foundations and Trends in Privacy and Security*, 7(4):247–396.
- Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. 2021. [Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks](#). In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1167–1176, New York, NY, USA. Association for Computing Machinery.
- Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. [Improving recommendation lists through topic diversification](#). In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, page 22–32, New York, NY, USA. Association for Computing Machinery.

Appendix Overview

- **Appendix A: SPA Training and Inference Algorithm**
- **Appendix B: Dataset Statistics and Construction**
 - [B.1](#) Data Preprocessing
 - [B.2](#) Label Construction
 - [B.3](#) Feature Fields for CTR Baselines
- **Appendix C: Evaluation Metrics**
 - [C.1](#) LLM-based Prediction Score
 - [C.2](#) AUC
 - [C.3](#) Log Loss
 - [C.4](#) Accuracy
- **Appendix D: Reproducibility Details**
- **Appendix E: Semantic View Definitions**
 - [E.1](#) User Semantic Views
 - [E.2](#) Item Semantic Views
- **Appendix F: Cross-Profile Competitor Construction**
- **Appendix G: Item-side Alignment Formulation**
- **Appendix H: Semantic Profiling Prompt Templates**
 - [H.1](#) User Profiling Prompt Templates
 - [H.2](#) Item Profiling Prompt Templates
- **Appendix I: Prompt Construction and Preference Prediction**
 - [I.1](#) Final Preference Prediction Prompts
- **Appendix J: Embedding Space Visualization and Quantitative Structure Analysis**
 - [J.1](#) Qualitative Embedding Space Visualization
 - [J.2](#) Quantitative Structure-Preservation Metrics
- **Appendix K: Sensitivity to Key Semantic and Retrieval Configurations**
 - [K.1](#) Impact of Different LLM Backbones
 - [K.2](#) Impact of Semantic View Configuration
 - [K.3](#) Sensitivity to the Two-Channel Weight
 - [K.4](#) Contrastive Temperature Sensitivity
 - [K.5](#) Projection Network Sensitivity Analysis
- **Appendix L: Additional Controlled Fusion and Generic Alignment Baselines**
- **Appendix M: Additional Backbone Comparison**
- **Appendix N: Analysis of Two-Channel Retrieval Aggregation**
- **Appendix O: Collaborative Anchor Quality Analysis**
- **Appendix P: Training and Inference Cost Analysis**
 - [P.1](#) CF Training
 - [P.2](#) Semantic Profile Generation and Encoding
 - [P.3](#) Alignment Training
 - [P.4](#) Actual Offline Cost

A SPA Training and Inference Algorithm

Algorithm 1: Training and Inference Pipeline of SPA

```

1 Input: Training interaction graph  $\mathcal{G}$ , profiling
   histories  $\mathcal{H}_u^{\text{train}}$ , target-specific histories  $\mathcal{H}_{u,t}$ , item
   metadata  $\mathcal{Q}_j$ 
2 Output: Binary preference prediction for a target
   user–item pair
3 Stage 1: Profile Generation and CF Training
4 Generate multi-view semantic profiles  $\mathcal{S}_u, \mathcal{T}_j$  and
   encode into dense embeddings  $\mathcal{Z}_u, \mathcal{Z}_j$  via text
   encoder  $\psi(\cdot)$ 
5 Train CF backbone  $f_{\text{CF}}(\cdot)$  on the full training graph
 $\mathcal{G}$  to obtain collaborative embeddings  $\mathbf{c}_u, \mathbf{c}_j$ ; freeze
 $f_{\text{CF}}$ 
6 Stage 2: Cross-Profile Semantic Alignment
7 for each alignment epoch do
8   for each entity side  $e \in \{\text{user, item}\}$  do
9     Sample a shuffled mini-batch of entity–view
     training instances  $\mathcal{B}$ 
10    Apply semantic augmentation to semantic
    view embeddings
11    Project collaborative embeddings and
    semantic view embeddings using the
    shared projection networks  $f_a$  and  $f_s$ 
12    Construct positive pairs, in-batch negatives
    from other entities, and cross-profile
    competitors from alternative views of the
    same entity
13    Update the shared projection networks  $f_a$ 
    and  $f_s$  using the corresponding alignment
    loss
14  end
15 end
16 Obtain final inference representations:
17  $\mathbf{r}_u = \text{norm}(f_a(\mathbf{c}_u))$  and  $\mathbf{r}_j = \text{norm}(f_a(\mathbf{c}_j))$ 
18 Stage 3: Structure-Preserving Two-Channel
Retrieval
19 for each target triplet  $(u, j_t, \mathcal{H}_{u,t})$  do
20   Score each  $j_h \in \mathcal{H}_{u,t}$  using user–item and
   item–item ranking signals
21    $\text{Score}(j_h) = \frac{1-\lambda}{\text{rank}_{UI}(j_h)+1} + \frac{\lambda}{\text{rank}_{II}(j_h)+1}$ 
22   Retrieve Top- $K$  items from  $\mathcal{H}_{u,t}$  to construct
   behavioral context  $\mathcal{C}_u$ 
23 end
24 Stage 4: LLM-Based Preference Prediction
25 Construct sequential prompts  $\mathcal{T}_{\text{seq}}$  and collaborative
   prompts  $\mathcal{T}_{\text{collab}}$  from available user attributes,
   history context, and target item  $j_t$ 
26 Sample  $\mathcal{T}_{\text{seq}}$  and  $\mathcal{T}_{\text{collab}}$  with equal probability
   during LoRA fine-tuning on the 1%/10%
   prediction-instance subset using the binary Yes/No
   objective
27 Inference after Training
28 Retrieve  $\mathcal{C}_u$  via two-channel retrieval using projected
   collaborative representations
29 Compute  $p_{\text{yes}} = p(\text{Yes}) / (p(\text{Yes}) + p(\text{No}))$  with the
   fine-tuned prediction LLM

```

B Dataset Statistics and Construction

We conduct experiments on three widely used recommendation benchmarks: MovieLens-1M (ML-1M), BookCrossing, and Amazon-Movies. Table 3 summarizes the dataset statistics after preprocessing.

Table 3: Dataset statistics after preprocessing. #Interactions denotes the number of observed user–item interaction records, not the number of constructed prediction instances. #Fields denotes the number of categorical feature fields; #Features denotes the total number of unique feature values across all fields.

Dataset	#Users	#Items	#Interactions	#Fields	#Features
ML-1M	5,395	3,631	853,761	8	16,939
BookCrossing	4,653	4,289	87,891	8	912,268
Amazon-Movies	12,364	5,000	123,491	4	24,750

B.1 Data Preprocessing

Following prior LLM-based recommendation protocols (Lin et al., 2024; Bao et al., 2023), we construct CTR prediction samples as binary user–item preference prediction instances for all datasets.

For ML-1M and Amazon-Movies, all interactions are sorted by global timestamp and split sequentially into training and test sets with a ratio of 8:1. For each prediction instance in these timestamped datasets, the target-specific history $\mathcal{H}_{u,t}$ contains only interactions occurring before the target interaction; later interactions are excluded from retrieval and prompt construction.

For BookCrossing, which does not provide timestamps, the interacted items of each user are randomly shuffled; the final item after shuffling is used as the prediction target, and the remaining items are used as the user history.

The resulting BookCrossing user-level prediction instances are randomly divided into training and test subsets with a ratio of 9:1. This split applies to downstream prediction instances rather than to the construction of the collaborative graph.

For collaborative representation learning on BookCrossing, we use a transductive leave-one-out construction. For each user, the held-out target interaction is excluded from the CF graph, while the user’s remaining non-target interactions are retained. This allows test users to obtain collaborative representations without exposing the held-out target interaction or its label to the CF backbone.

Consequently, BookCrossing contributes one user-level prediction instance per user, whereas

ML-1M and Amazon-Movies construct interaction-level prediction instances from timestamp-split target interactions.

For ML-1M and Amazon-Movies, the CF graph is constructed from interactions available under the chronological training split, with held-out test target interactions excluded. Across all datasets, evaluated users and target items are required to have valid collaborative representations in the resulting CF graph.

For each prediction instance, the target-specific history $\mathcal{H}_{u,t}$ excludes the target interaction and its label and is used for retrieval and prompt construction. Across all datasets, held-out evaluation target interactions and their labels are excluded from the CF training signal, semantic profile generation, alignment training, retrieval histories, and prediction prompts.

B.2 Label Construction.

We convert dataset feedback values into binary preference labels.

- **ML-1M** contains user–movie ratings from 1 to 5. Ratings ≥ 4 are labeled as positive and the rest as negative.
- **BookCrossing** contains user–book ratings from 0 to 10. Ratings ≥ 5 are labeled as positive and the rest as negative.
- **Amazon-Movies** contains user–movie ratings from 1 to 5. Ratings ≥ 4 are labeled as positive and the rest as negative.

B.3 Feature Fields for CTR Baselines.

For traditional CTR baselines, we use dataset-specific structured feature fields. All categorical features are label-encoded and mapped to trainable dense embeddings. The feature fields are summarized as follows:

- **ML-1M**: user and item identifiers, user demographic attributes including gender, age, occupation, and zipcode, and item metadata including title and genre.
- **BookCrossing**: user and item identifiers, user attributes including age and location, and book metadata including publication year, title, author, and publisher.
- **Amazon-Movies**: user and item identifiers, item category, and title information.

C Evaluation Metrics

We evaluate model performance using AUC, Log Loss, and ACC.

C.1 LLM-based prediction score.

For LLM-based binary preference prediction, the model is prompted to generate a Yes/No response. We convert the Yes/No output distribution into a predicted preference probability:

$$p_i = \frac{p(\text{Yes})}{p(\text{Yes}) + p(\text{No})}. \quad (14)$$

AUC and Log Loss are computed using p_i , and Accuracy is computed by thresholding p_i at 0.5.

C.2 AUC.

The Area Under the ROC Curve (AUC) measures the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance. Formally,

$$\text{AUC} = \frac{1}{|P||N|} \sum_{i \in P} \sum_{j \in N} \left[\mathbf{1}(s_i > s_j) + \frac{1}{2} \mathbf{1}(s_i = s_j) \right]. \quad (15)$$

where P and N denote the sets of positive and negative samples, respectively, and s denotes the predicted score. For LLM-based prediction, we use p_i as the predicted score s_i .

C.3 Log Loss.

Log Loss evaluates the quality of probabilistic predictions:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N (y_i \log p_i + (1 - y_i) \log(1 - p_i)), \quad (16)$$

where $y_i \in \{0, 1\}$ is the ground-truth label and p_i is the predicted probability.

C.4 Accuracy.

Accuracy (ACC) measures the proportion of correct predictions:

$$\text{ACC} = \frac{1}{N} \sum_{i=1}^N \mathbf{1}(\hat{y}_i = y_i), \quad (17)$$

where $\hat{y}_i$ is obtained by thresholding p_i at 0.5.

D Reproducibility Details

To improve reproducibility, we summarize the unified protocol used across SPA and all baselines. All methods are trained and evaluated under the same processed splits, binary label construction, and binary user-item preference prediction protocol. Dataset statistics, sample construction, and feature fields are described in Appendix B, and evaluation metrics including LLM-based Yes/No probability scoring are described in Appendix C. For SPA, the CF backbone is trained first and then frozen, projection networks are optimized with the contrastive alignment objective, and the downstream LLM predictor is fine-tuned using retrieved behavioral context. Prompt templates are provided in Appendices H and I.

LLM Baseline Reproduction. We control the backbone, data, supervision budget, and evaluation setting when comparing against ReLLa¹, TALL-Rec², and BIGRec³. Specifically, all three methods are run with Llama-3.1-8B-Instruct, using the same processed train/test partitions and the same 1% and 10% subsets of labeled prediction instances used for downstream instruction tuning. Binary preference labels and evaluation procedures are also shared across the LLM-based methods. For each baseline, we start from its publicly released official implementation and modify only the components needed to support the common backbone and evaluation setting. The remaining method-dependent choices are kept consistent with the original implementation, including how interaction histories and available input fields are organized, as well as the corresponding LoRA and optimization configurations.

E Semantic View Definitions

This appendix defines the semantic views used in SPA. We construct multiple semantic views for each user and item so that the alignment stage can observe complementary language-derived descriptions of the same entity. This design is motivated by the fact that LLM-generated text may emphasize different aspects of user behavior or item characteristics depending on prompt formulation. Using multiple views therefore provides more robust semantic supervision and enables cross-profile con-

trastive alignment across alternative textual descriptions.

E.1 User Semantic Views

To characterize complementary aspects of user preference behavior, we generate multiple semantic views for each user from three history-based perspectives:

- **Recent interaction behavior**, which reflects preference signals expressed in the later portion of the available interaction history;
- **Early interaction behavior**, which captures preference patterns from the earlier portion of the available interaction history;
- **Global preference behavior**, which represents the user’s overall preference tendency across the available interaction history.

For BookCrossing, the terms *recent* and *early* are retained only as view identifiers. Because interaction timestamps are unavailable, they refer to the later and earlier segments of the preprocessing-defined interaction order and do not imply chronological recency or temporal preference evolution.

For each perspective, the LLM produces two complementary textual forms:

- an **analysis view**, which provides a detailed interpretation of behavioral patterns and inferred preferences;
- a **summary view**, which provides a concise high-level description of the user’s dominant interests.

Combining the three behavioral perspectives with the two textual forms yields six user semantic views:

$$\mathcal{V}_u = \{r_a, r_s, e_a, e_s, g_a, g_s\}, \quad (18)$$

where r , e , and g denote recent, early, and global behavioral perspectives, respectively, and a and s denote analysis and summary forms.

These six views provide diverse yet complementary semantic observations of the same user. In SPA, they serve as multi-view semantic supervision signals for alignment with collaborative anchor representations.

¹<https://github.com/LaVieEnRose365/ReLLa>

²<https://github.com/SAI990323/TALLRec>

³<https://github.com/SAI990323/BIGRec>

E.2 Item Semantic Views

For each item, we generate multiple semantic views from item metadata such as title, category, and textual description. The goal is to capture not only the intrinsic semantic content of the item, but also its potential compatibility or mismatch with user preferences.

Specifically, the LLM produces three complementary item views:

- **Profile view**, describing the intrinsic semantic content of the item;
- **Like-oriented view**, describing the types of users or preferences that are likely to be attracted to the item;
- **Dislike-oriented view**, describing the types of users or preferences for which the item may be a poor match.

These views define the item semantic view set:

$$\mathcal{V}_j = \{p_r, l_o, d_o\} \quad (19)$$

where p_r , l_o , and d_o denote profile, like-oriented, and dislike-oriented semantic views.

Together, these views provide complementary semantic descriptions of the same item from both content and preference-compatibility perspectives. They are used as multi-view semantic supervision signals during alignment.

F Cross-Profile Competitor Construction

SPA performs semantic alignment over entity–view training instances. For each instance in a mini-batch, its semantic representation forms the positive pair with the corresponding collaborative anchor, while entity–view instances from other entities in the mini-batch serve as in-batch negatives and the remaining semantic views of the same entity serve as cross-profile competitors. These cross-profile competitors are view-specific competitors rather than permanently negative semantic representations, since each view serves as the positive in its own training instance. As illustrated in Figure 2, each entity–view training instance in a mini-batch forms one positive pair, and the resulting view-specific losses are averaged within the mini-batch.

The number of cross-profile competitors is fixed by the predefined semantic-view configuration rather than treated as a tunable hyperparameter: each user has six semantic views and thus five

cross-profile competitors, whereas each item has three semantic views and thus two cross-profile competitors.

During each epoch, the training loader iterates over all entity–view instances, so every semantic view of each entity serves once as the positive while acting as a cross-profile competitor for the remaining view-specific instances. This rotating construction provides a same-entity multi-view regularizer across complementary semantic descriptions, rather than selecting a single behavior-consistent semantic view. Cross-profile competitors are used only during semantic alignment training. The item-side alignment follows the same procedure with its corresponding semantic-view configuration.

G Item-side Alignment Formulation

For completeness, we provide the item-side formulation of the alignment objective used in SPA. The item-side formulation follows the same objective form as the user-side formulation, with item indices replacing user indices.

Let $\mathcal{B}$ denote a shuffled mini-batch of entity–view training instances, where each training instance is represented as (j, v) consisting of an item j and one of its semantic views v . Accordingly, $(j_b, v_b) \in \mathcal{B}$ denotes an arbitrary entity–view training instance sampled from the same mini-batch.

Let $\mathcal{V}_j$ denote the set of semantic views associated with item j .

For each training instance $(j, v) \in \mathcal{B}$, the objective encourages the semantic representation $\mathbf{s}_{j,v}$ to align with its collaborative anchor $\mathbf{a}_j$, contrasts it against entity–view training instances of other items in the mini-batch, and incorporates the remaining semantic views of the same item as cross-profile competitors within the view-specific normalization.

We define the in-batch negative term as

$$D_{j,v}^{\text{in}} = \sum_{\substack{(j_b, v_b) \in \mathcal{B} \\ j_b \neq j}} \exp(\text{sim}(\mathbf{a}_j, \mathbf{s}_{j_b, v_b})/\tau). \quad (20)$$

and the cross-profile competition term as

$$D_{j,v}^{\text{cp}} = \sum_{v' \in \mathcal{V}_j \setminus \{v\}} \exp(\text{sim}(\mathbf{a}_j, \mathbf{s}_{j, v'})/\tau), \quad (21)$$

and the positive pair term as

$$D_{j,v}^{\text{pos}} = \exp(\text{sim}(\mathbf{a}_j, \mathbf{s}_{j,v})/\tau). \quad (22)$$

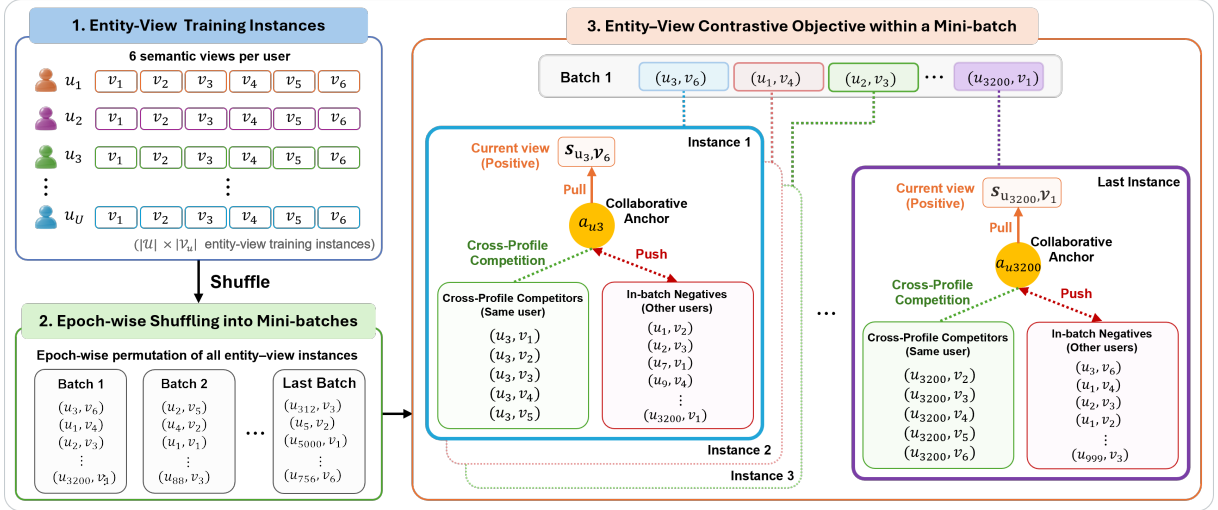


Figure 2: **Cross-profile competitor construction.** Entity-view instances are shuffled into mini-batches. For each instance, its semantic view forms the positive pair with the corresponding collaborative anchor; views of other entities serve as in-batch negatives, whereas alternative views of the same entity serve as cross-profile competitors. Each view becomes the positive in its own instance over an epoch.

The item-side alignment loss is then defined as

$$\mathcal{L}_{\text{align}}^{\text{item}} = -\frac{1}{|\mathcal{B}|} \sum_{(j,v) \in \mathcal{B}} \log \frac{D_{j,v}^{\text{pos}}}{D_{j,v}^{\text{pos}} + D_{j,v}^{\text{in}} + D_{j,v}^{\text{cp}}}. \quad (23)$$

Here, $\text{sim}(\cdot, \cdot)$ denotes cosine similarity. As in the user-side formulation, the denominator consists of the positive, in-batch negative, and cross-profile competition terms. The alternative semantic views of the same item act as cross-profile competitors for the current training instance, but are not assigned fixed negative roles, since each view becomes the positive in its own training instance. The per-instance losses are averaged over the mini-batch during optimization.

H Semantic Profiling Prompt Templates

This appendix presents the prompt templates used to generate the user and item semantic views defined in Appendix E. These prompts are designed to produce structured and complementary language descriptions that are later encoded into semantic embeddings for alignment.

For users, we design prompts that capture complementary preference patterns from the available user history, including recent/early segments when order is available and global behavioral patterns. For items, we design prompts that capture both intrinsic semantic content and preference-oriented interpretations, including likely attraction and potential mismatch signals. This separation is important because SPA relies on multiple semantic

observations of the same entity rather than a single textual description.

H.1 User Profiling Prompt Templates

We use two user profiling prompt types. The history-based prompt compares different portions of the available user history, enabling the LLM to capture local preference patterns and broader tendencies. The general profile prompt summarizes the user’s overall preference behavior across the available user history. Together, these prompts support the construction of segment-level and global semantic views in both analysis and summary forms. The detailed prompt templates are provided in Tables 4–7. In implementation, the history-based template is instantiated separately for the recent and early history segments, while the general-profile template is instantiated once over the full profiling history. Each request returns both an analysis and a summary output, resulting in three generation requests and six user semantic views per user.

H.2 Item Profiling Prompt Templates

We use two item profiling prompt types. The content profiling prompt generates a factual semantic description of the item itself and is used to construct the profile view. The preference-oriented prompt generates user-attraction and user-mismatch descriptions and is used to construct the like-oriented and dislike-oriented views. This separation allows SPA to model both intrinsic content semantics and preference compatibility signals. The detailed

Table 4: History-based user profiling prompt used to generate recent and early user semantic views for ML-1M. The prompt encourages the LLM to compare recent interactions with earlier history and to produce both detailed analysis and concise summary outputs. The same template is used for Amazon-Movies.

ML-1M History-based User Profile Prompt
You are a movie preference analyst who interprets a user’s movie rating history. Your goal: • Explain why the user rated the most recent movie as they did. • Identify any preference shift or consistency relative to the earliest record. • Conclude with a concise User Description summarizing general taste. Follow these steps: 1. Analyze the latest ratings for genre, tone, pacing, and emotional themes. 2. Derive at least 3 key Preference_Analysis insights highlighting changes or consistent traits. 3. In Rating_Reasoning, explain step-by-step reasoning (≥ 3 steps) for why the user gave the most recent movie its rating, grounding each step in evidence from the earlier rating or inferred taste patterns. 4. Explicitly classify the pattern as “Shift” or “Consistency.” 5. End with one concise User_Description (≤ 100 words) summarizing the user’s age, gender, and overall viewing preference. 6. Assign Confidence: High / Medium / Low depending on how much evidence the two ratings provide. 7. Output must strictly follow the JSON format below. Output Format: <pre>{ "Analysis": { "Preference_Analysis": ["Insight 1", "Insight 2", "Insight 3"], "Rating_Reasoning": { "Steps": ["1) ...", "2) ...", "3) ...", "4) (optional) ..."], "Preference_Trend": "Shift / Consistency" } }, "Summary": { "User_Description": "A concise summary of the user’s overall movie preferences within 100 words.", "Confidence": "High / Medium / Low" } }</pre> User Information: <pre>{text}</pre>

prompt templates are provided in Tables 8–11.

I Prompt Construction and Preference Prediction

To capture both local behavioral signals and structure-aware collaborative signals, we train the LLM using a mixed dataset composed of two complementary prompt types.

- **Sequential prompt** ($\mathcal{T}_{seq}$): constructed from the last K interactions when temporal order is available, or otherwise from K interactions selected from the available user history, capturing local behavioral context.
- **Collaborative prompt** ($\mathcal{T}_{collab}$): constructed from the retrieval context $\mathcal{C}_u$ obtained from the two-channel retrieval strategy, represent-

ing structurally relevant historical interactions.

During training, prompts from $\mathcal{T}_{seq}$ and $\mathcal{T}_{collab}$ are sampled with equal probability. Both prompt types share the same instruction format, differing only in how the interaction history is constructed. This strategy allows the model to learn both local preference patterns and collaborative relevance signals.

I.1 Final preference prediction prompts.

After structure-preserving retrieval, the selected behavioral context is converted into dataset-specific prompts and provided to the LLM together with the target item. The model is then asked to predict a binary Yes/No preference decision.

These prompts preserve a consistent decision format across datasets while adapting to avail-

Table 5: General user profiling prompt used to generate global user semantic views for ML-1M. The prompt summarizes stable preference patterns over the full interaction history and produces both analysis and summary outputs. The same template is used for Amazon-Movies.

ML-1M General User Profile Prompt
You are a movie preference analyst who interprets a user’s entire movie rating history. Your goal: • Explain the reasoning behind the user’s most recent movie rating. • Identify any preference shift or consistency compared to prior viewing history. • Conclude with a concise user description summarizing their general taste. Follow these detailed steps: 1. Analyze past ratings, genres, tones, and themes the user tends to favor or avoid. 2. Compare the most recent rating with prior tendencies to detect preference shift or consistency. 3. Provide at least three evidence-based insights in “Preference_Analysis”. 4. In “Rating_Reasoning”, describe step-by-step reasoning (≥ 3 steps) for why the user rated the latest movie as they did. 5. Summarize with “User_Description” — one concise paragraph (≤ 100 words) describing the user’s general taste, personality, and demographic traits. 6. Assign “Confidence” (“High” / “Medium” / “Low”) based on evidence strength. 7. Follow the exact JSON format below. Output Format: <pre>{ "Analysis": { "Preference_Analysis": ["Insight 1", "Insight 2", "Insight 3"], "Rating_Reasoning": { "Steps": ["1) ...", "2) ...", "3) ...", "4) (optional) ..."], "Preference_Trend": "Shift / Consistency" } }, "Summary": { "User_Description": "A concise summary of the user’s overall movie preferences within 100 words.", "Confidence": "High / Medium / Low" } }</pre> User Information: <pre>{text}</pre>

able metadata. ML-1M incorporates demographic attributes such as gender, age, and occupation, BookCrossing uses reader attributes including location and age, while Amazon-Movies relies solely on interaction history since explicit user metadata are unavailable.

The final prompt templates for each dataset are shown in Tables 12–14.

J Embedding Space Visualization and Quantitative Structure Analysis

To qualitatively analyze how SPA integrates collaborative and semantic representations, we visualize the learned embedding spaces using t-SNE across three datasets: ML-1M, BookCrossing, and Amazon-Movies. We present both item-level and user-level visualizations before and after alignment, as well as comparisons across different fusion strategies.

J.1 Qualitative Embedding Space Visualization

Although t-SNE does not preserve exact global distances, it provides an intuitive view of the structural organization of the learned embedding space and helps illustrate how SPA reshapes the structure of collaborative and semantic embeddings. To avoid relying solely on qualitative visualization, we additionally evaluate structure preservation using Top-10 neighborhood overlap, Spearman’s ρ , and Pearson’s r . Formal definitions and quantitative results are provided in Appendix J.2.

Figure 3 illustrates the distribution of item embeddings before and after applying SPA alignment. Across all datasets, CF embeddings form dense clusters reflecting interaction-driven preference structures, while semantic embeddings are distributed in separate semantic clusters. This separation reveals a clear modality gap between collaborative and language-derived representations.

Table 6: History-based user profiling prompt for BookCrossing. This prompt is used to generate history-segment user semantic views by contrasting different portions of the available user history and producing both analysis and summary outputs. Because BookCrossing does not provide interaction timestamps, temporal terms such as “most recent,” “latest,” and “earliest” in the prompt refer to the preprocessing-defined interaction order rather than true chronological recency.

BookCrossing History-based User Profile Prompt
You are a book preference analyst who interprets a user’s reading and rating history. Your goal: • Explain why the user rated the most recent book as they did. • Identify any preference shift or consistency relative to their earliest recorded book. • Conclude with a concise User Description summarizing overall reading preferences. Instructions: 1. Analyze the latest book rating for genre, writing style, and themes or character focus. 2. Derive at least 3 key Preference_Analysis insights about how the user’s literary taste has evolved or remained consistent. 3. In Rating_Reasoning, explain step-by-step (at least 3 steps) why the user gave the most recent book its rating, grounding each reason in past ratings or inferred reading tendencies. 4. Explicitly classify the observed pattern as “Shift” or “Consistency.” 5. End with one concise User_Description (within 100 words) summarizing the user’s age, possible gender, and overall reading preference. 6. Assign Confidence: High / Medium / Low depending on how strong the evidence from the provided history is. 7. Output must strictly follow the JSON format below. Output Format: <pre>{ "Analysis": { "Preference_Analysis": ["Insight 1", "Insight 2", "Insight 3"], "Rating_Reasoning": { "Steps": ["1) ...", "2) ...", "3) ...", "4) (optional) ..."], "Preference_Trend": "Shift / Consistency" } }, "Summary": { "User_Description": "A concise summary of the user’s overall book preferences within 100 words.", "Confidence": "High / Medium / Low" } }</pre> User Information: <pre>{text}</pre>

After applying SPA, embeddings from both modalities converge into a shared representation space. Importantly, the aligned embeddings maintain coherent cluster structures, suggesting that SPA integrates semantic information while better maintaining local collaborative neighborhoods.

Figure 4 presents the same analysis for user embeddings. The separation between collaborative and semantic embeddings is even more pronounced for users than for items across all datasets. For example, in BookCrossing the collaborative embeddings form a dense central cluster while semantic embeddings appear as several distinct semantic clusters.

After alignment, SPA reduces this modality gap and produces a unified representation space in which both collaborative and semantic information

are integrated.

Figure 5 compares the embedding spaces produced by different representation learning strategies. CF-only embeddings reflect collaborative preference structure but lack clear semantic grouping. Text-only embeddings produce tight semantic clusters but do not reflect interaction-driven preference signals.

In the t-SNE projection, Naive Fusion produces a more dispersed organization than the original CF representation, whereas SPA exhibits more coherent local grouping. Because t-SNE provides only a qualitative view of the embedding geometry, we rely on the quantitative neighborhood-overlap and correlation metrics in Table 15 for structure-preservation comparisons.

Figure 6 shows the corresponding visualization

Table 7: General user profiling prompt for BookCrossing. This prompt summarizes reading preferences over the available user history and is used to generate global user semantic views in both analysis and summary forms. Because BookCrossing does not provide interaction timestamps, temporal terms such as “most recent” and “latest” in the prompt refer to the preprocessing-defined interaction order rather than true chronological recency.

BookCrossing General User Profile Prompt
You are a book preference analyst who interprets a user’s entire reading and rating history. Your goal: • Explain the reasoning behind the user’s most recent book rating. • Identify any preference shift or consistency compared to prior reading history. • Conclude with a concise user description summarizing their general literary taste. Follow these detailed steps: 1. Analyze past ratings, genres, writing styles, and themes the user tends to favor or avoid. 2. Compare the most recent rating with prior tendencies to detect preference shift or consistency. 3. Provide at least three evidence-based insights in “Preference_Analysis”. 4. In “Rating_Reasoning”, describe step-by-step reasoning (at least 3 steps) for why the user rated the latest book as they did. 5. Summarize with “User_Description” — one concise paragraph (within 100 words) describing the user’s general reading taste, personality, and demographic traits. 6. Assign “Confidence” (“High” / “Medium” / “Low”) based on evidence strength. 7. Follow the exact JSON format below. Output Format: <pre>{ "Analysis": { "Preference_Analysis": ["Insight 1", "Insight 2", "Insight 3"], "Rating_Reasoning": { "Steps": ["1) ...", "2) ...", "3) ...", "4) (optional) ..."], "Preference_Trend": "Shift / Consistency" } }, "Summary": { "User_Description": "A concise summary of the user’s overall book preferences within 100 words.", "Confidence": "High / Medium / Low" } }</pre> User Information: <pre>{text}</pre>

for user embeddings. CF-only representations appear widely dispersed because collaborative signals alone provide limited semantic grouping. Text-only representations form tight semantic clusters reflecting language-derived preference descriptions.

In the t-SNE projection, Naive Fusion produces a more dispersed organization, whereas SPA exhibits more coherent local neighborhood patterns. Because t-SNE provides only qualitative evidence, quantitative structure-preservation comparisons are reported in Table 15.

J.2 Quantitative Structure-Preservation Metrics

To complement the qualitative t-SNE visualizations, we quantitatively evaluate how well each learned representation retains the structure of the original CF embedding space. These metrics are computed using AutoCF representations obtained

under the dataset-specific collaborative-graph construction described in Appendix B.1.

For local structure, we identify the 10 nearest neighbors of each user or item in the original CF space and in the representation produced by each method using cosine similarity, excluding the entity itself. We define the Top-10 neighborhood overlap as

$$\text{Overlap@10}(e) = \frac{|\mathcal{N}_{10}^{\text{CF}}(e) \cap \mathcal{N}_{10}^{\text{method}}(e)|}{10}. \quad (24)$$

We report the average of $\text{Overlap@10}(e)$ over all evaluated entities in each dataset–entity group. Higher overlap indicates stronger retention of local collaborative neighborhoods.

For global structure, we compare pairwise cosine similarities in the original CF space with those in each learned representation space. Spearman’s ρ measures rank-level agreement between the two

Table 8: Item content profiling prompt used to generate the profile view for ML-1M. Given the title and genre, the LLM produces a concise factual description of the item’s intrinsic semantic content. The same template is used for Amazon-Movies.

ML-1M Item Description Prompt
You are a professional movie analyst. Your task: Write a concise and informative description of the movie or TV show titled “{movie_title}”, whose genre is “{genre}”, considering the following aspects: • A brief plot overview (avoid major spoilers) • The main themes and genre focus • The setting or historical context • Key characters and the actors portraying them • The director and notable stylistic or cinematic elements • The overall tone and mood • Its cultural influence or deeper insight, if relevant Rules: 1. The description must be factual, concise, and under 80 words. 2. Do not use bullet points. 3. Return only JSON in the exact format below. Output Format: <pre>{ "movie_title": "{movie_title}", "genre": "{genre}", "description": "A concise, spoiler-free summary under 80 words.", "word_count": 0 }</pre>

sets of pairwise similarities, while Pearson’s r measures their linear correlation.

As shown in Table 15, SPA achieves the highest Top-10 neighborhood overlap across all datasets and entity types, indicating strong retention of the local collaborative neighborhoods used by retrieval. The global correlation metrics show a more nuanced pattern: for ML-1M items, Naive Fusion attains higher Spearman’s ρ and Pearson’s r than SPA, whereas SPA achieves stronger correlation fidelity for the remaining dataset–entity combinations. Accordingly, our notion of structure preservation focuses on empirical retention of collaborative neighborhood relationships rather than exact preservation of the entire CF geometry.

K Sensitivity to Key Semantic and Retrieval Configurations

This appendix provides additional analyses of key semantic and retrieval configurations in SPA, including the choice of LLM semantic encoder, semantic-view configuration, two-channel retrieval weight λ , and contrastive temperature τ . The proposed hyperparameter configuration was fixed before test-set evaluation; the following analyses vary

individual components only to assess sensitivity around the predefined configuration. Unless otherwise specified, all sensitivity experiments are conducted under the 1% few-shot setting using AutoCF with $K = 10$, and performance is measured by AUC. We fix the collaborative backbone to isolate the effect of each configuration variable.

K.1 Impact of Different LLM Backbones

To assess whether SPA depends on a specific semantic encoder, we replace the default Llama-3.1-8B-Instruct (Grattafiori et al., 2024) encoder with Qwen2.5-7B-Instruct (Yang et al., 2024) and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023) under the unified sensitivity setting.

Table 16: Impact of different LLM semantic encoders. All models are evaluated under the same 1% few-shot setting using AutoCF with $K = 10$, and performance is measured by AUC.

Dataset	Llama-3.1-8B	Qwen2.5-7B	Mistral-7B
ML-1M	0.7880	0.7844	0.7849
BookCrossing	0.7126	0.7074	0.6835
Amazon-Movies	0.7953	0.7661	0.7649

Table 9: Preference-oriented item profiling prompt used to generate the like-oriented and dislike-oriented item views for ML-1M. The prompt asks the LLM to characterize user groups that are likely to prefer or dislike the item. The same template is used for Amazon-Movies.

ML-1M User-group Preference Analysis Prompt
You are a professional movie audience analyst. Your task: Identify and describe both (1) what types of users would most likely enjoy, and (2) what types of users would most likely dislike the movie or TV show titled “{movie_title}”, whose genre is “{genre}”. Consider the following aspects: • Typical audience demographics (age range, gender tendency) • Personality traits or lifestyle preferences • Viewing motivations (e.g., emotional engagement, intellectual curiosity, escapism) • Preferred or disliked storytelling and pacing styles • Similar movies or genres that align or conflict with their taste Rules: 1. Each summary must be concise, specific, and under 100 words. 2. Do not use bullet points. 3. Return only JSON in the exact format below. Output Format: <pre>{ "movie_title": "{movie_title}", "genre": "{genre}", "user_like_summarization": "A concise description (under 100 words) of what types of users would most likely enjoy this movie.", "user_dislike_summarization": "A concise description (under 100 words) of what types of users would most likely not enjoy this movie.", "word_count_like": 0, "word_count_dislike": 0 }</pre>

As shown in Table 16, the results show that SPA remains effective across alternative semantic encoders. Although the default Llama-3.1-8B encoder achieves the strongest overall performance, Qwen2.5-7B and Mistral-7B preserve the main performance trend, indicating that SPA is not tightly coupled to a single semantic encoder. At the same time, the remaining performance differences suggest that the quality of the semantic embedding space still influences final recommendation performance.

K.2 Impact of Semantic View Configuration

To investigate the impact of semantic diversity on representation learning, we vary the number and type of semantic views used for alignment. This analysis examines whether SPA benefits from using multiple behavior-defined user views and preference-oriented item views, rather than relying on a single textual profile. For datasets without explicit timestamps, such as BookCrossing, recent

and early views refer to different segments of the available user history rather than strict chronological intervals. Following the view definitions in Appendix E, we compare three configurations:

- **V1 (Coarse)** uses one user view and one item view: *global-summary* (g_s) for users and *profile view* (p_r) for items.
- **V2 (Mid)** uses three user views and two item views: *recent-summary* (r_s), *early-summary* (e_s), and *global-summary* (g_s) for users, and *profile* (p_r) and *like-oriented* (l_o) views for items.
- **V3 (Proposed)** uses the full SPA configuration with six user views and three item views: *recent-analysis* (r_a), *recent-summary* (r_s), *early-analysis* (e_a), *early-summary* (e_s), *global-analysis* (g_a), and *global-summary* (g_s) for users, and *profile* (p_r), *like-oriented* (l_o), and *dislike-oriented* (d_o) views for items.

Table 10: Item content profiling prompt for BookCrossing. Given bibliographic metadata, the LLM produces a concise factual description of the item’s intrinsic semantic content, which is used as the profile view.

BookCrossing Item Description Prompt					
You are a professional book analyst.					
Your task:					
Write a concise and informative description of the book titled “{book_title}” by “{book_author}”, published in “{publication_year}” by “{publisher}”.					
Consider the following aspects:					
• A brief plot overview (avoid major spoilers) • The main themes and literary focus • The setting or historical context • Key characters and their roles • The author’s style or narrative techniques • The overall tone and mood • Its cultural or philosophical significance, if relevant					
Rules:					
1. The description must be factual, concise, and under 80 words.					
2. Do not use bullet points.					
3. Return only JSON in the exact format below.					
Output Format:					
<pre>{ "book_title": "{book_title}", "book_author": "{book_author}", "publication_year": "{publication_year}", "publisher": "{publisher}", "description": "A concise, spoiler-free summary under 80 words.", "word_count": 0 }</pre>					

Table 17: Impact of semantic view configuration on SPA. We vary the number of user and item semantic views used for alignment and report AUC under the 1% few-shot setting using AutoCF with $K = 10$.

Setting	User Views	Item Views	ML-1M	BookCrossing	Amazon-Movies
V1	1	1	0.7800	0.6834	0.7650
V2	3	2	0.7850	0.7111	0.7659
V3	6	3	0.7880	0.7126	0.7953

As shown in Table 17, performance improves as the semantic view configuration becomes richer. V2 outperforms V1 across all datasets, suggesting that history-segment user summaries and preference-oriented item views provide useful alignment supervision. The full V3 configuration achieves the best performance, indicating that richer semantic diversity further benefits CF-anchored alignment. We interpret this result as evidence that semantic diversity is beneficial for SPA, rather than as proof that every individual view is strictly necessary.

K.3 Sensitivity to the Two-Channel Weight

We analyze the sensitivity of SPA to the weighting parameter λ in the two-channel retrieval mechanism. Specifically, we vary $\lambda \in \{0.0, 0.3, 0.6, 1.0\}$ to examine the relative contribution of the user-item and item-item channels. Here, $\lambda = 0$ uses only the user-item channel, whereas $\lambda = 1$ uses only the item-item channel.

Table 18: Marginal contribution of two-channel retrieval under the 1% few-shot setting using AutoCF with $K = 10$. $\lambda = 0$ uses only the user-item channel, while $\lambda = 1$ uses only the item-item channel. Performance is measured by AUC.

λ	ML-1M	BookCrossing	Amazon-Movies
0.0	0.7474	0.7084	0.7863
0.3	0.7787	0.7102	0.7919
0.6	0.7880	0.7126	0.7953
1.0	0.7853	0.7100	0.7946

As shown in Table 18 and Figure 7, the proposed setting $\lambda = 0.6$ achieves the best AUC across all datasets. Compared with the user-item-only set-

Table 11: Preference-oriented item profiling prompt for BookCrossing. The prompt generates complementary descriptions of reader groups that are likely to prefer or dislike the item, supporting the construction of like-oriented and dislike-oriented item views.

BookCrossing User-group Preference Analysis Prompt			
You are a professional book audience analyst.			
Your task: Identify and describe both (1) what types of readers would most likely enjoy, and (2) what types of readers would most likely dislike the book titled "{book_title}" by "{book_author}", published in "{publication_year}" by "{publisher}".			
Consider the following aspects:			
• Typical reader demographics (age range, education level, reading habits) • Personality traits or lifestyle preferences • Reading motivations (e.g., emotional resonance, intellectual depth, escapism) • Preferred or disliked narrative and pacing styles • Similar books or literary genres that align or conflict with their taste			
Rules:			
1. Each summary must be concise, specific, and under 100 words.			
2. Do not use bullet points.			
3. Return only JSON in the exact format below.			
Output Format:			
<pre>{ "book_title": "{book_title}", "book_author": "{book_author}", "publication_year": "{publication_year}", "publisher": "{publisher}", "user_like_summarization": "A concise description (under 100 words) of what types of readers would most likely enjoy this book.", "user_dislike_summarization": "A concise description (under 100 words) of what types of readers would most likely not enjoy this book.", "word_count_like": 0, "word_count_dislike": 0 }</pre>			

ting, incorporating the item–item channel substantially improves retrieval quality, especially on ML-1M. At the same time, the item–item-only setting is slightly weaker than the two-channel configuration, indicating that target-specific relevance is most effective when combined with global user preference signals. These results show that the two-channel retrieval mechanism provides more informative behavioral context than either single-channel variant alone.

K.4 Contrastive Temperature Sensitivity

We analyze the sensitivity of SPA to the contrastive temperature τ , which controls the sharpness of the alignment distribution between collaborative anchors and semantic views.

Table 19: Sensitivity to the contrastive temperature τ under the 1% few-shot setting using AutoCF with $K = 10$. Performance is measured by AUC.

τ	ML-1M	BookCrossing	Amazon-Movies
0.03	0.7838	0.6856	0.7646
0.07	0.7880	0.7126	0.7953
0.10	0.7865	0.6787	0.7657

As shown in Table 19, SPA performs best around $\tau = 0.07$ across the evaluated datasets. Lower or higher temperatures reduce performance, suggesting that alignment sharpness affects the quality of semantic supervision.

K.5 Projection Network Sensitivity Analysis

To evaluate the selected projection architecture, we conduct a one-factor-at-a-time sensitivity analysis over projection depth, hidden dimension, and activation function. Each factor is varied independently while all remaining factors and experimen-

Table 12: Final preference prediction prompt for ML-1M. The prompt conditions on user demographic attributes and interaction history to predict preference for a target movie via binary Yes/No inference.

ML-1M Preference Prediction Prompt	
The user is a {Gender}. The user’s job is {Job}. The user’s age is {Age}.	
The user watched the following movies in order in the past, and rated them:	
1. {Movie Title} ({rating} stars)	
2. {Movie Title} ({rating} stars)	
...	
K. {Movie Title} ({rating} stars)	
Based on the movies the user has watched, deduce if the user will like the movie ***{target_movie_title}***.	
Note that more stars the user rated the movie, the user liked the movie more.	
You should ONLY tell me yes or no.	

Table 13: Final preference prediction prompt for BookCrossing. The prompt conditions on user attributes and reading history to predict preference for a target book via binary Yes/No inference.

BookCrossing Preference Prediction Prompt	
The user’s location is {Location}. The user’s age is {Age}.	
The user read the following books in the past and rated them:	
1. {Book Title} ({rating} stars)	
2. {Book Title} ({rating} stars)	
...	
K. {Book Title} ({rating} stars)	
Based on the books the user has read, deduce if the user will like the book ***{target_book_title}***.	
Note that more stars the user rated the book, the user liked the book more.	
You should ONLY tell me yes or no.	

tal settings are fixed to the proposed configuration (AutoCF backbone, 1% few-shot, $K=10$).

Table 20: Sensitivity analysis of the projection network (AUC). **Bold** denotes the proposed configuration.

Design Factor	Variant	ML-1M	BookCrossing	Amazon Movies
Depth	1-layer	0.7837	0.6624	0.7664
	2-layer	0.7880	0.7126	0.7953
	3-layer	0.7848	0.6971	0.7695
Width	128	0.7859	0.6732	0.7611
	512	0.7880	0.7126	0.7953
	1024	0.7849	0.6740	0.7611
Activation	ReLU	0.7880	0.7126	0.7953
	GELU	0.7859	0.6870	0.7596
	Tanh	0.7825	0.6846	0.7594

As shown in Table 20, across all three datasets, the proposed two-layer projection network with hidden dimension 512 and ReLU activation consistently achieves the best recommendation performance. Alternative projection depths and hidden dimensions, as well as replacing ReLU with GELU

or Tanh, consistently result in lower recommendation performance. These results empirically support the selected projection configuration.

L Additional Controlled Fusion and Generic Alignment Baselines

To further compare SPA with representative feature fusion and generic alignment strategies, we additionally evaluate four controlled baselines under the same experimental protocol. All methods use the same data splits, AutoCF backbone, identical semantic inputs, the same 1% few-shot setting, the same retrieval configuration ($K = 10$), the same downstream LLM predictor, and the same evaluation protocol. The only difference lies in the integration module and its corresponding optimization objective.

Specifically, we compare the following representative methods:

- **Naive Fusion:** Direct concatenation of collaborative and semantic representations.

Table 14: Final preference prediction prompt for Amazon-Movies. The prompt conditions only on interaction history, as user metadata are unavailable, to predict preference for a target movie via binary Yes/No inference.

Amazon-Movies Preference Prediction Prompt	
The user has watched the following movies in the past, and rated them:	
1. {Movie Title} ({rating} stars)	
2. {Movie Title} ({rating} stars)	
...	
K. {Movie Title} ({rating} stars)	
Based on the movies the user has watched, deduce if the user will like the movie ***{target_movie_title}***.	
Note that more stars the user rated the movie, the user liked the movie more.	
You should ONLY tell me yes or no.	

- **Gated Fusion:** Learnable gating of collaborative and semantic representations.
- **CLIP-style Alignment:** Symmetric InfoNCE-based representation alignment without asymmetric collaborative anchoring or cross-profile semantic supervision.
- **KD-style Alignment:** Regression-based feature distillation for aligning collaborative and semantic representations without contrastive learning.

All methods are evaluated using the unified AutoCF backbone under the 1% few-shot setting with $K = 10$. This unified protocol differs from the dataset-specific backbone configuration reported in Table 2, which explains the different numerical values for BookCrossing and Amazon-Movies.

Table 21: Comparison with controlled fusion and generic alignment baselines under the unified AutoCF protocol (AutoCF, 1% few-shot, $K = 10$).

Method	ML-1M	BookCrossing	Amazon-Movies
Naive Fusion	0.7765	0.7119	0.7919
Gated Fusion	0.7716	0.6862	0.6798
CLIP-style	0.7842	0.6911	0.6669
KD-style	0.7727	0.6820	0.7618
SPA (Proposed)	0.7880	0.7126	0.7953

As shown in Table 21, SPA achieves the highest AUC across all three datasets under the unified evaluation protocol. The evaluated fusion and generic alignment baselines do not reproduce SPA’s downstream gains under the same AutoCF, 1% few-shot, and $K = 10$ setting. These controlled comparisons therefore provide empirical support for the proposed recommendation-specific formulation without implying that alignment is universally superior to feature fusion.

M Additional Backbone Comparison

This appendix provides detailed performance comparisons of SPA across different collaborative filtering backbones. We evaluate three representative graph-based CF models: LightGCN, AutoCF, and SGL. For each backbone, the CF encoder is first trained on the interaction graph to produce collaborative embeddings, which are then used as anchor representations in SPA. We report results under different history lengths ($K \in \{5, 10, 15\}$) and two downstream LLM fine-tuning regimes (1% and 10%).

As shown in Table 22, SPA maintains strong and stable performance across all backbones and datasets. Although the best-performing backbone varies slightly depending on dataset characteristics—for example, AutoCF performs best on ML-1M while SGL shows competitive results on the sparser BookCrossing and Amazon-Movies datasets—the overall performance trends remain consistent across different collaborative backbones.

These results indicate that the full SPA pipeline remains effective across different graph-based collaborative backbones, suggesting that its benefits are not tied to a single CF architecture.

N Analysis of Two-Channel Retrieval Aggregation

SPA retrieves historical evidence using two complementary retrieval channels: a user–item channel that captures personalized collaborative preference and an item–item channel that captures target-specific contextual relevance. Since the two channels provide complementary rankings over the user’s historical items, SPA aggregates their rank positions to construct the final retrieval context.

Although both channels compute cosine simi-

Table 15: Quantitative structure-preservation metrics, including Top-10 neighborhood overlap, Spearman’s ρ , and Pearson’s r , comparing each representation with the original CF embedding space. Metrics are computed using AutoCF under the dataset-specific collaborative-graph construction described in Appendix B.1.

Higher values indicate stronger fidelity to the original CF structure.

Dataset	Entity	Method	Top-10 Overlap $\uparrow$	Spearman’s $\rho\uparrow$	Pearson’s $r\uparrow$
ML-1M	User	Text-only	0.0053	0.0809	0.0480
		Naive Fusion	0.0118	0.5119	0.5660
		SPA	0.2210	0.7737	0.7600
	Item	Text-only	0.0137	0.0409	0.0269
		Naive Fusion	0.0416	0.4735	0.4409
		SPA	0.2451	0.4565	0.3642
BookCrossing	User	Text-only	0.0179	0.0114	0.0057
		Naive Fusion	0.0044	0.1084	0.1132
		SPA	0.1875	0.4896	0.5160
	Item	Text-only	0.0054	0.0346	0.0307
		Naive Fusion	0.0091	0.1068	0.1247
		SPA	0.1577	0.3836	0.4206
Amazon-Movies	User	Text-only	0.0150	0.0168	0.0141
		Naive Fusion	0.0277	0.2355	0.2663
		SPA	0.2558	0.4172	0.4622
	Item	Text-only	0.0203	0.0227	0.0127
		Naive Fusion	0.0534	0.1948	0.2205
		SPA	0.2707	0.3355	0.3870

larity in the same aligned embedding space, they correspond to different query types and exhibit substantially different score distributions. Table 23 reports the mean and standard deviation of cosine similarity scores computed over all training samples.

Table 23: Cosine similarity statistics of the two retrieval channels.

Dataset	User-Item (mean/std)	Item-Item (mean/std)
ML-1M	−0.021/0.064	0.400/0.268
BookCrossing	0.008/0.059	0.135/0.177
Amazon-Movies	0.002/0.062	0.217/0.236

The item–item channel consistently exhibits larger means and standard deviations than the user–item channel. Consequently, directly combining raw cosine similarity scores introduces channel-dependent bias unless explicit cross-channel score calibration is performed.

We compare three retrieval aggregation strategies:

- **Raw Cosine Weighted Sum:** Direct summation of cosine similarity scores from the user–item and item–item retrieval channels.
- **Z-score Calibrated Cosine Sum:** Channel-wise z-score normalization followed by cosine score summation.
- **Reciprocal-rank Aggregation (Proposed):** Rank-based aggregation that is invariant to monotonic score rescaling within each retrieval channel while avoiding explicit cross-channel score calibration.

Table 24: Comparison of retrieval aggregation strategies (AUC).

Aggregation Strategy	ML-1M	BookCrossing	Amazon -Movies
Raw Cosine Weighted Sum	0.7912	0.6828	0.7646
Z-score Calibrated Cosine Sum	0.7803	0.6878	0.7662
Reciprocal-rank Aggregation (Proposed)	0.7880	0.7126	0.7953

Table 22: (RQ4) Performance comparison across different structural backbones under few-shot LLM fine-tuning settings. Bold AUC values indicate the highest AUC among backbones for each dataset, K , and few-shot setting.

Backbone	K	ML-1M			BookCrossing			Amazon-Movies		
		AUC↑	Log Loss↓	ACC↑	AUC↑	Log Loss↓	ACC↑	AUC↑	Log Loss↓	ACC↑
Few-shot 1% LLM Fine-tuning										
LightGCN	5	0.7538	0.5870	0.6934	0.6794	0.6619	0.6280	0.7769	0.3236	0.8763
AutoCF		0.7747	0.5879	0.7044	0.6964	0.6516	0.6278	0.7761	0.3354	0.8787
SGL		0.7721	0.5822	0.7048	0.6983	0.6425	0.6460	0.7762	0.3279	0.8766
LightGCN	10	0.7679	0.5742	0.7011	0.7041	0.6524	0.6260	0.7967	0.3185	0.8808
AutoCF		0.7880	0.5768	0.7123	0.7126	0.6122	0.6565	0.7953	0.3166	0.8745
SGL		0.7842	0.5810	0.7119	0.7164	0.6574	0.6324	0.7983	0.3145	0.8761
LightGCN	15	0.7743	0.5713	0.7056	0.7124	0.6719	0.6221	0.7963	0.3200	0.8782
AutoCF		0.7943	0.5603	0.7179	0.7136	0.6497	0.6258	0.7965	0.3242	0.8741
SGL		0.7903	0.5518	0.7186	0.7089	0.6855	0.6153	0.7941	0.3212	0.8751
Few-shot 10% LLM Fine-tuning										
LightGCN	5	0.7641	0.5762	0.7000	0.6752	0.6628	0.6311	0.7863	0.3144	0.8788
AutoCF		0.7812	0.5586	0.7107	0.7026	0.6454	0.6468	0.7833	0.3189	0.8803
SGL		0.7795	0.5610	0.7113	0.6950	0.6403	0.6425	0.7861	0.3251	0.8795
LightGCN	10	0.7778	0.5643	0.7091	0.7160	0.6249	0.6448	0.8084	0.3040	0.8809
AutoCF		0.7954	0.5452	0.7221	0.7161	0.6266	0.6435	0.8078	0.3139	0.8813
SGL		0.7925	0.5488	0.7194	0.7200	0.6335	0.6486	0.8120	0.3021	0.8816
LightGCN	15	0.7830	0.5588	0.7122	0.7217	0.5969	0.6542	0.8095	0.3035	0.8824
AutoCF		0.8019	0.5377	0.7269	0.7316	0.6446	0.6349	0.8129	0.3009	0.8823
SGL		0.7979	0.5422	0.7243	0.7255	0.6060	0.6560	0.8146	0.3103	0.8816

As shown in Table 24, although raw cosine weighted summation performs slightly better on ML-1M, reciprocal-rank aggregation consistently achieves stronger performance on BookCrossing and Amazon-Movies. Overall, the results indicate that no single aggregation strategy is uniformly optimal across datasets. We therefore adopt reciprocal-rank aggregation because it provides a robust, calibration-free mechanism for combining complementary retrieval signals without requiring explicit cross-channel score calibration.

O Collaborative Anchor Quality Analysis

SPA aligns semantic representations toward collaborative embeddings produced by the CF backbone. Because the quality of these collaborative embeddings depends on the optimization of the underlying CF model, we analyze how collaborative anchor quality affects the downstream recommendation performance.

To evaluate this dependency, we intentionally undertrained the CF backbone by training it for only one epoch while keeping all remaining compo-

nents unchanged (AutoCF backbone, 1% few-shot, $K=10$). Table 25 reports the resulting recommendation performance.

Table 25: Effect of collaborative anchor quality (AUC).

Dataset	1-Epoch CF Anchor	Fully-trained CF Anchor	Δ AUC
ML-1M	0.7874	0.7880	-0.0006
BookCrossing	0.6958	0.7126	-0.0168
Amazon-Movies	0.7593	0.7953	-0.0360

The performance difference is negligible on ML-1M but becomes substantially larger on BookCrossing and Amazon-Movies. These results indicate that SPA inherits the quality of its collaborative anchor rather than being invariant to undertrained collaborative representations, highlighting the importance of a sufficiently optimized CF encoder.

We do not evaluate domain-mismatched collaborative encoders in this work. Since SPA relies on collaborative anchors learned from the target domain, its robustness under cross-domain settings remains an important direction for future investiga-

tion.

P Training and Inference Cost Analysis

Let N denote the number of users, M the number of items, and let V denote the number of semantic views for the corresponding entity type.

The computational cost of SPA consists of three main components.

P.1 CF Training.

Graph-based CF models require

$$\mathcal{O}(|\mathcal{E}|d_c)$$

per epoch, where $|\mathcal{E}|$ denotes the number of interactions and d_c denotes the collaborative embedding dimension.

P.2 Semantic Profile Generation and Encoding.

Semantic preprocessing consists of two distinct operations. LLM-based profile generation requires a fixed number of generation requests per entity in our implementation—three requests per user and two requests per item—because individual requests can return multiple semantic views. Once generated, encoding is performed once per semantic view, requiring

$$\mathcal{O}(N|\mathcal{V}_u| + M|\mathcal{V}_j|)$$

text-encoding operations.

P.3 Alignment Training

Contrastive alignment operates on shuffled mini-batches of B entity–view training instances, resulting in approximately

$$\mathcal{O}(B^2 + BV)$$

similarity computations, where the $\mathcal{O}(B^2)$ term arises from in-batch similarity computation and the $\mathcal{O}(BV)$ term from cross-profile comparisons. Because alignment updates only lightweight projection networks over frozen representations, its measured wall-clock cost is small in our experiments (1–2 minutes per configuration; Table 27). Overall, the CF backbone, semantic encoding, and alignment modules are prepared offline. At inference time, SPA performs retrieval followed by LLM-based prediction.

P.4 Actual Offline Cost

Table 26 summarizes the offline cost of semantic profile generation. We report the number of user and item generation requests separately, together with the total number of requests and the corresponding billing cost. Semantic profiles are generated once using Vertex AI BatchPredictionJob. Since profile generation is executed as a cloud batch service, we report generation requests rather than wall-clock execution time.

Table 26: Offline cost of semantic profile generation.

Dataset	User	Item	Total	Cost (USD)
ML-1M	16,185	7,766	23,951	59.09
BookCrossing	47,826	542,750	590,576	338.71
Amazon-Movies	37,092	10,000	47,092	108.16
Total	101,103	560,516	661,619	505.96

Each user requires three generation requests to produce six semantic views, whereas each item requires two generation requests to produce three semantic views. Therefore, the reported request counts correspond to actual LLM generation requests rather than the number of generated semantic views. Since semantic profiles are generated before the final filtering stage, the reported requests reflect the entities processed during offline semantic profiling.

Table 27 reports the GPU time required for one-time semantic encoding and contrastive alignment. Semantic encoding is performed on frozen semantic profiles, while contrastive alignment trains only lightweight projection networks on frozen collaborative and semantic representations.

Table 27: Offline semantic encoding and alignment time (min).

Stage	ML-1M	BookCrossing	Amazon -Movies	Total
Semantic Encoding	66.8	40.1	120.7	227.6
Alignment Training	1–2	1–2	1–2	–

Both semantic profile generation and semantic encoding are performed only once during offline preprocessing. Contrastive alignment also incurs only a small one-time cost. Consequently, semantic profile generation, semantic encoding, and contrastive alignment introduce only offline preprocessing overhead. At inference time, SPA does not require Gemini-based profile generation or se-

mantic encoding; inference consists of two-channel retrieval over the projected collaborative representations followed by LLM-based preference prediction.

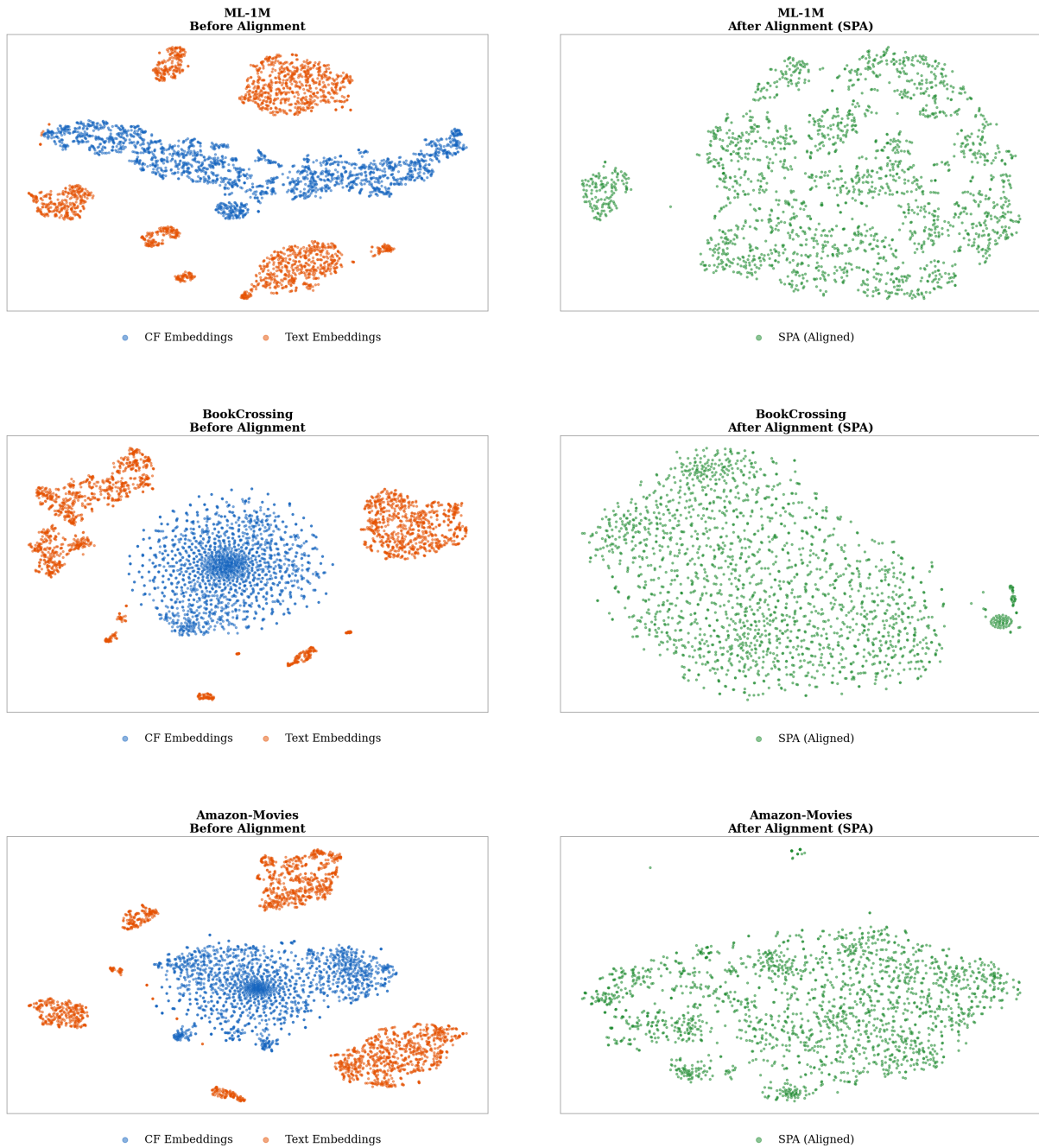


Figure 3: t-SNE visualization of item embeddings before and after applying SPA alignment on ML-1M, BookCrossing, and Amazon-Movies. Before alignment, CF embeddings and semantic embeddings occupy clearly separated regions of the representation space, indicating a substantial modality gap between interaction-derived and language-derived representations. After alignment, SPA brings the two modalities into a unified embedding space while maintaining meaningful local structural organization.

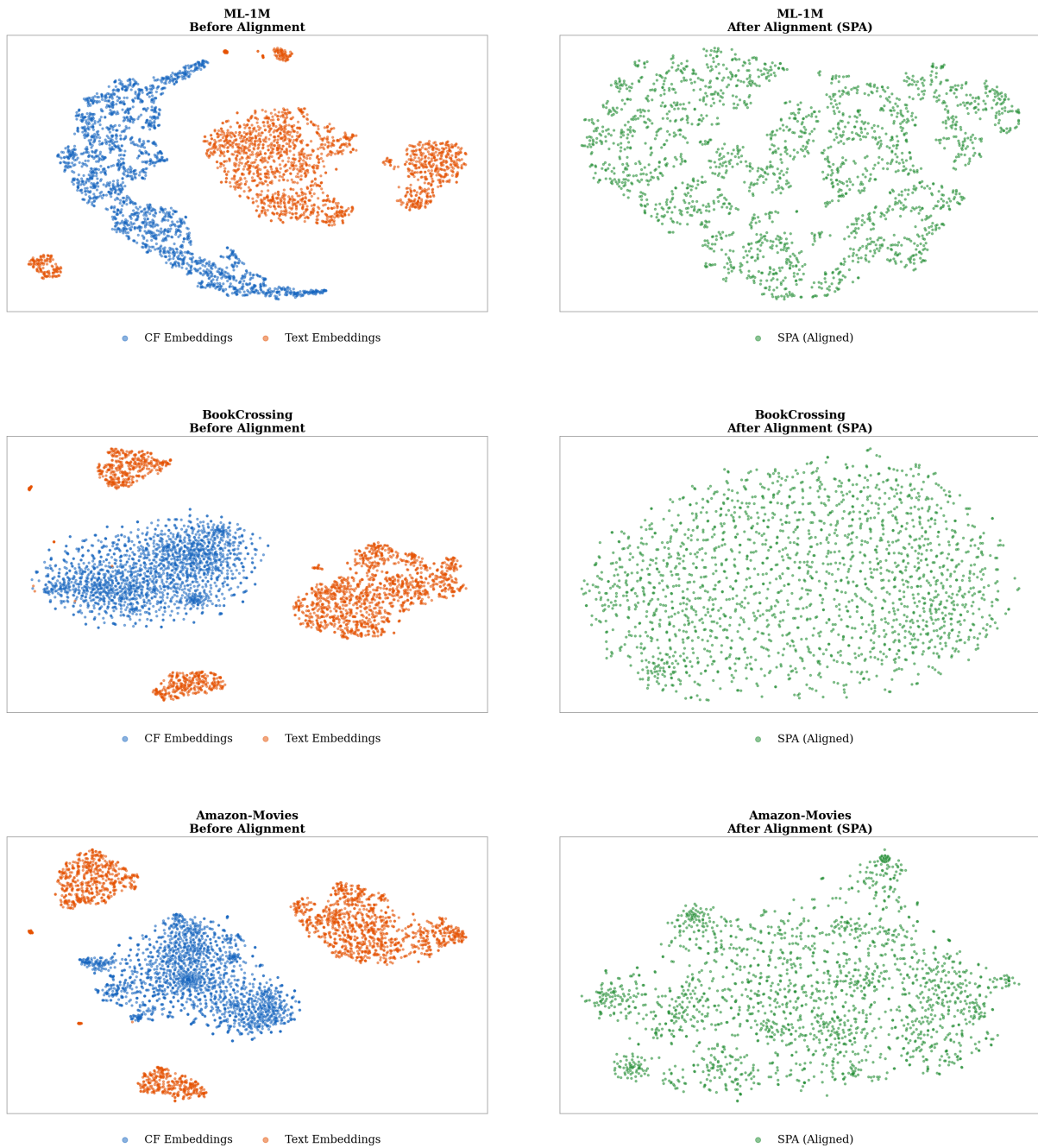


Figure 4: t-SNE visualization of user embeddings before and after SPA alignment across ML-1M, BookCrossing, and Amazon-Movies. Before alignment, collaborative and text representations occupy distinct regions, demonstrating a pronounced modality gap for user representations. After alignment, SPA integrates both modalities into a unified representation space while maintaining meaningful local structural patterns.



Figure 5: t-SNE comparison of item embeddings produced by different methods: CF-only, Text-only, Naive Fusion, and SPA. CF-only captures collaborative structure but lacks semantic grouping, while Text-only forms tight semantic clusters without reflecting interaction-based relationships. Naive Fusion disrupts the geometric structure of both modalities. In contrast, SPA better maintains local structural patterns while integrating collaborative and semantic signals.

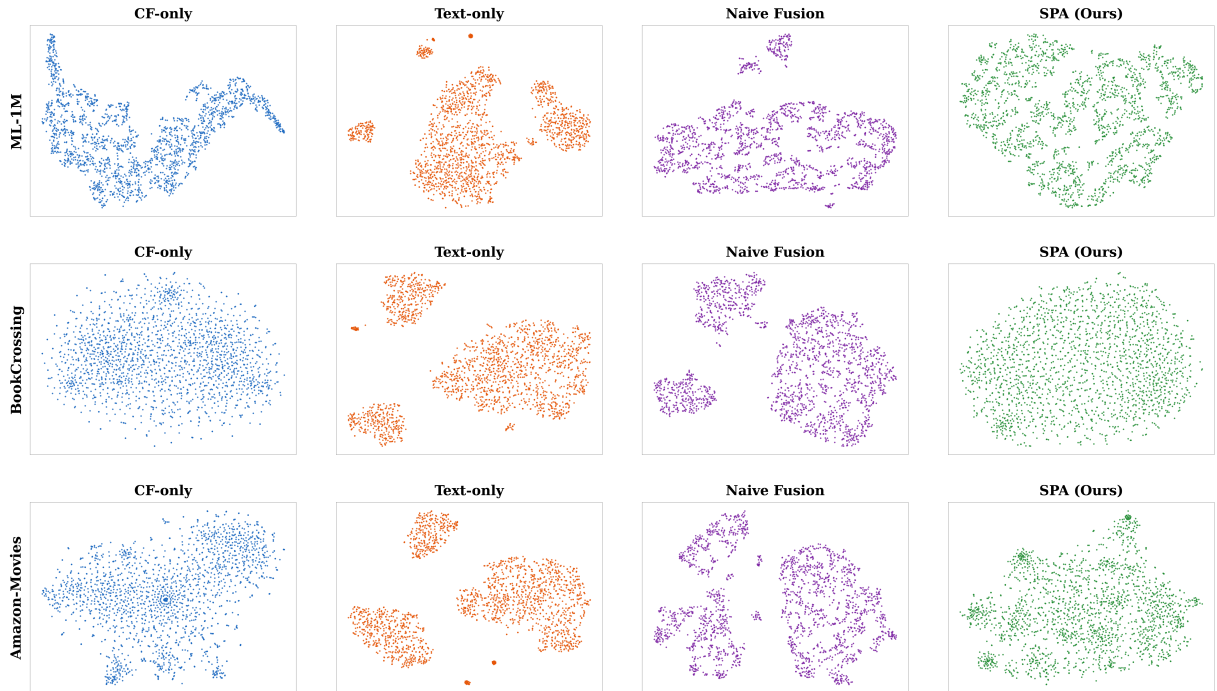


Figure 6: t-SNE comparison of user embeddings produced by CF-only, Text-only, Naive Fusion, and SPA across the three datasets. While CF-only embeddings exhibit diffuse collaborative structures and Text-only embeddings form tight semantic clusters, naive fusion disrupts both representations. SPA produces a unified embedding space that better maintains local structural patterns while incorporating collaborative preference signals.

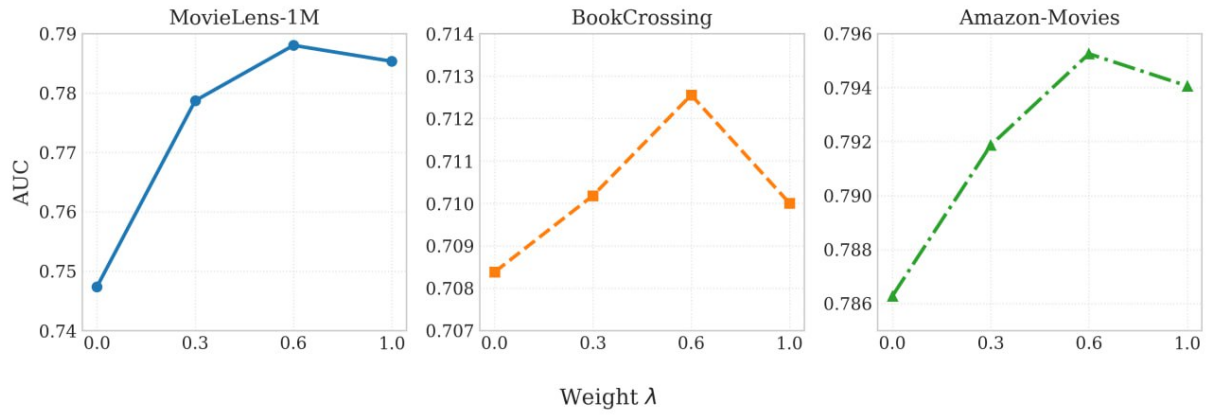


Figure 7: Sensitivity analysis of the two-channel weighting parameter λ . λ controls the balance between the user–item channel and the item–item channel in the retrieval mechanism. Across all datasets, the best performance is achieved around $\lambda = 0.6$, showing that combining global preference signals with target-specific item similarity provides the most effective retrieval context.