LEARN BEFORE YOU FUSE: PRESERVING COMPLEMENTARITY IN GENERATIVE RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Semantic IDs (SIDs) compress continuous item representations into discrete codes for generative recommendation. A single SID, however, preserves only one coarse organization of the item space, while quantization removes fine-grained item information. Different representation-learning objectives can induce different geometries over the same item space, suggesting that independently constructed SID views can capture complementary relational structures, while dense representations retain item-level details obscured by quantization. Yet simply combining such representations does not automatically yield complementary predictors. We identify a recurring failure mode: models exploit readily accessible signals before alternative predictive structures are sufficiently established, leaving complementary pathways under-trained. To address this, we propose Learn-Before-Fuse (LBF), which first establishes a complementary predictor before introducing readily exploitable signals, and instantiates it in CVGR^2, a Complementary-View Generative Recommendation and Re-Ranking framework. CVGR^2 models complementary SID views through interleaved cross-view interaction while preserving view-specific prediction spaces. Following LBF, it establishes the SID predictor before adding dense refinement; its reranker likewise learns history-conditioned item-level relevance before incorporating multi-view SID-based matching evidence. Across diverse item dense representations, LBF consistently improves over joint-learning in both retriever and reranker, with clear gain isolated in our extensive ablations. Across ten public datasets, CVGR^2 improves mean Recall@10 by 23.06% over the strongest baseline. Code is available at `https://anonymous.4open.science/r/MSGR2-40EF`.

1 INTRODUCTION

Semantic-ID-based generative recommendation (GR) (Rajput et al., 2023) represents each item with a sequence of discrete tokens and formulates next-item prediction as autoregressive generation. Semantic IDs (SIDs) are obtained by quantizing continuous item representations, thereby transferring an embedding space into a coarse discrete organization of items. Most existing SID-based methods construct this prediction space from a single item representation, and therefore inherit the particular relational geometry emphasized by its representation-learning objective (Penha et al., 2025). Moreover, quantization inevitably obscures fine-grained information retained in the original continuous space (Yang et al., 2025a; Liang et al., 2026). This suggests improving next-item prediction along two complementary dimensions: alternative relational structures from independently constructed SID views, and fine-grained item evidence from dense representations. Together, they can provide a richer basis for prediction than any single quantized view.

The difficulty is that complementary information does not automatically become complementary predictive capability. Different pathways may enter downstream training at very different levels of maturity. Consider an item-level collaborative representation learned from the same interaction logs as the GR model. Before GR training begins, this representation has already organized items around patterns predictive of next-item behavior. The SID predictor, in contrast, must learn from scratch how user histories map to hierarchical token sequences. If the dense collaborative signal is exposed from the beginning, the optimizer obtains an immediate shortcut for reducing training loss by reusing this already-formed structure. Shared model parameters can then be shaped around this accessible pathway before the SID-specific autoregressive structure has sufficiently developed, weakening the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

pressure to learn the complementary SID predictor. We refer to this difference in how readily a pathway can reduce the current training loss as *Training-time Optimization Accessibility (TOA)*.

TOA is not unique to representations pretrained on the downstream interaction data. We observe the same effect, although more mildly, with independently pretrained text representations: they still provide readily usable item information, which impedes the SID predictor from developing its own predictive structure from scratch.

This motivates *Learn-Before-Fuse (LBF)*. Rather than treating fusion as a symmetric joint-learning problem, LBF first establishes the predictor whose complementary inductive bias must be learned from scratch, and only then introduces the more readily exploitable signal for refinement. The principle is about *training order*, rather than simply delaying all auxiliary information: pathways with comparable optimization accessibility can still be learned jointly. LBF instead targets asymmetric cases in which an already accessible signal can bypass the formation of a useful alternative predictor.

We instantiate LBF in CVGR^2, a *Complementary-View Generative Recommendation and Re-Ranking* framework. CVGR^2 constructs multiple SID views from independently learned item representations and jointly models them with an Interleaved Temporal-View (ITV) encoder, allowing cross-view interaction while retaining view-specific autoregressive prediction spaces. Dense item representations complement the quantized inputs with continuous item information to refine autoregressive prediction. Because the SID views are decoded in separate code spaces, their predictions provide separate item-ranking lists that must ultimately be reconciled. CVGR^2 therefore introduces a history-aware reranker, which uses the continuous item representation to directly model history-conditioned item-level relevance and combines it with SID-based ranking signals from the generative decoders.

CVGR^2 specifies how complementary signals are combined, while LBF uses their relative TOA to determine when they should be fused. The SID views have similar TOA because their autoregressive predictors are learned from scratch, but their TOA is much lower than that of well-learned dense representations; LBF therefore trains the SID views jointly and adds dense refinement only afterwards. The same principle applies to reranking: history-conditioned item-level relevance must be learned from scratch, whereas ranking signals from the trained SID decoders are immediately available. LBF thus learns the item-level ranker first and incorporates SID-based matching evidence later.

Contributions. **(1)** We identify a training-time failure mode where readily exploitable signals can prevent complementary predictive structures from being fully learned. **(2)** We address its two sides with CVGR^2, which determines *how* complementary signals are fused, and LBF, which determines *when* they should be fused. **(3)** Empirically, early dense injection degrades SID generation across all three tested representation families, whereas LBF consistently improves their utilization and preserves a strong standalone SID predictor. LBF further yields an isolated and significant relative Recall@10 gain in reranking. Across ten public datasets, the complete CVGR^2 framework improves mean Recall@10 by 23.06% over the strongest evaluated baseline.

2 CVGR^2

2.1 Relational Geometry and Complementary Views

Given a user u with a chronologically ordered interaction history $\mathcal{H}_u = [i_1, \ldots, i_L]$, our goal is to predict the next item $i^+ \in \mathcal{I}$. SID-based generative recommendation represents each item with a sequence of discrete tokens and predicts the target SID autoregressively.

We use *relational geometry* to denote the similarity and neighborhood structure induced over the item set by a continuous item representation. Quantizing such a representation maps its continuous geometry into a coarse hierarchical organization used for SID prediction. For representation view $v \in \mathcal{V}$, we independently apply the GAOQ quantizer of ReSID (Liang et al., 2026) and obtain

$$\mathrm{SID}^{(v)}(i) = [c_{i,v,1}, \ldots, c_{i,v,S}], \tag{1}$$

where each view has an independent codebook. Representations learned under different objectives can induce different relational geometries; their independently quantized SIDs therefore provide candidate views with distinct inductive biases for next-item prediction. We do not assume these views to be complementary by construction, and evaluate their predictive complementarity empirically.

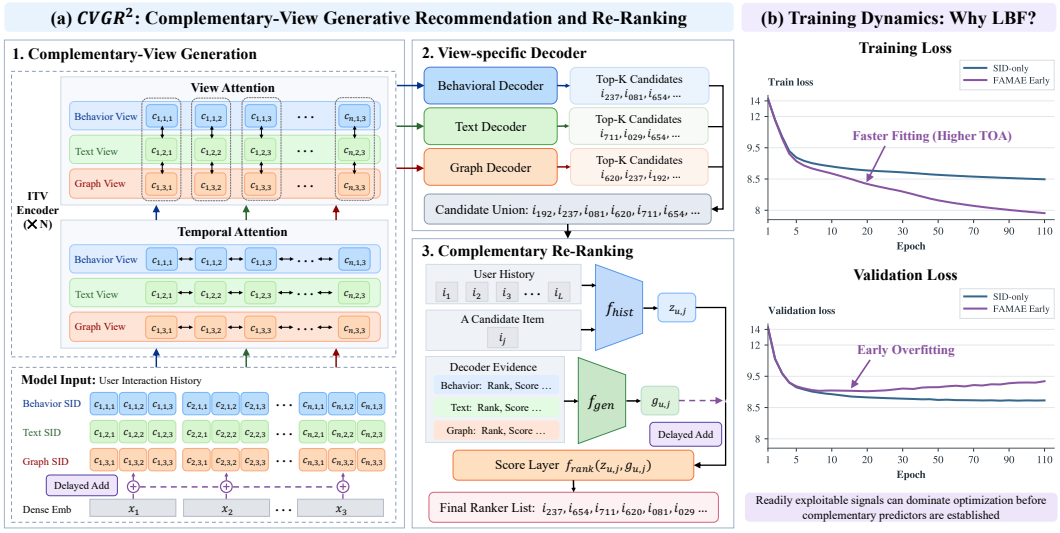

Figure 1: Overview of CVGR^2 and the motivation for Learn-Before-Fuse. **(a)** CVGR^2 combines multiple SID views for autoregressive generation, dense residual refinement, and history-aware item-level re-ranking. **(b)** When an already-learned recommendation-oriented dense representation is introduced from the start, training loss decreases much faster than in SID-only training, while validation loss turns upward early; SID-only training continues to reduce training and validation losses together.

Quantization also obscures part of the fine-grained information retained in the continuous representation. We therefore keep a dense item representation as a residual information pathway: the SID views provide structured discrete prediction spaces, while the dense representation supplies continuous item information that is not directly accessible through the quantized codes. Together, these two forms of complementarity enrich next-item prediction beyond a single quantized view. Figure 1 summarizes both sides of our design. Figure 1(a) shows how CVGR^2 combines complementary SID, dense, and ranking signals, while Figure 1(b) illustrates why their training order matters.

2.2 Learn-Before-Fuse

The representation design above creates potential complementarity, but joint training does not guarantee complementary predictive capability. Different pathways can vary substantially in how readily the current model can exploit them to reduce training loss. We call this property *Training-time Optimization Accessibility (TOA)*. TOA is relative to the current model state and distinct from predictive complementarity: two useful pathways can have very different accessibility during optimization. We use TOA as a qualitative ordering between pathways rather than as a calibrated numerical quantity.

A high-TOA pathway can provide an immediate shortcut for reducing training loss before a lower-TOA predictor has developed its own predictive structure. Figure 1(b) gives a representative example: an already-learned item-level collaborative representation rapidly lowers training loss when exposed from the beginning, but this faster fitting does not translate into continued validation improvement. The same tendency also appears, although more mildly, with pretrained text representations, showing that high TOA is not restricted to signals learned from the same training data.

This motivates *Learn-Before-Fuse (LBF)*. Pathways with comparable TOA can be learned jointly. When complementary pathways have substantially mismatched TOA, LBF first establishes the lower-TOA predictor and then introduces the more readily exploitable pathway for refinement. The principle therefore concerns training order, rather than prescribing late fusion for all auxiliary information.

2.3 COMPLEMENTARY-VIEW GENERATION

Complementary SID views. For each view $v \in \mathcal{V}$, let $x_i^{(v)}$ denote an item representation learned under a view-specific objective. Applying GAOQ independently to the representation spaces produces separate SID codebooks and prediction spaces, allowing each view to retain the coarse organization induced by its own relational geometry instead of compressing all views into a single SID space. The concrete representation families used in our experiments are described in Section 3.1.

Interleaved temporal–view modeling. Jointly modeling multiple SID views requires the model to capture temporal dependencies within each view while exchanging information across views. Directly collapsing all views may obscure their individual structures, whereas processing them independently prevents cross-view interaction. We therefore design an *Interleaved Temporal-View (ITV) Encoder* that alternates Temporal Attention and View Attention.

For a history of length L, we expand each item into its S SID tokens and arrange the resulting representations as $H^{(0)} \in \mathbb{R}^{T\times V\times d}$, where $T = LS$ follows the chronological item order and SID-slot order, $V = |\mathcal{V}|$, and d is the hidden dimension. The ITV Encoder contains N Transformer layers and alternates the two attention operations:

$$H^{(\ell+1)} = \mathcal{A}_{a_\ell}^{(\ell)}\left(H^{(\ell)}\right), \qquad a_\ell \in \{\text{temporal}, \text{view}\} \text{ alternates with } \ell. \tag{2}$$

Here, $\mathcal{A}_{\text{temporal}}^{(\ell)}$ is applied independently within each view to model the SID sequence over the interaction history, whereas $\mathcal{A}_{\text{view}}^{(\ell)}$ exchanges information among views at the same historical position and SID slot. This alternation preserves a separate representation stream for each SID view while progressively incorporating cross-view information.

The final ITV representations are passed to view-specific autoregressive decoders. Decoder D_v predicts the target SID in the corresponding code space, and the generation objective remains the sum of the standard view-specific autoregressive losses:

$$\mathcal{L}_{\text{gen}}^{(v)} = -\sum_{s=1}^{S} \log p_v\left(c_{i^+,v,s} \mid c_{i^+,v,<s}, H_{:,v,:}^{(N)}\right), \qquad \mathcal{L}_{\text{gen}} = \sum_{v\in\mathcal{V}} \mathcal{L}_{\text{gen}}^{(v)}. \tag{3}$$

Dense residual refinement. Although the SID views preserve coarse structures from multiple relational geometries, quantization still removes part of the continuous item information. Let $x_i \in \mathbb{R}^{d_x}$ denote the dense representation of item i, and let $e_{i,v,s}^{\text{disc}}$ denote its discrete input embedding in view v at SID slot s. We inject the dense information additively:

$$h_{i,v,s} = e_{i,v,s}^{\text{disc}} + r_{i,s}, \qquad r_{i,s} = P_s(x_i), \tag{4}$$

where $P_s : \mathbb{R}^{d_x} \to \mathbb{R}^d$ is a slot-specific projection and $r_{i,s}$ is shared across SID views. The SID pathway remains the structured base for autoregressive prediction, while the dense residual supplies continuous item information obscured by quantization.

LBF for generation. At the start of GR training, the SID views have comparable TOA and substantially lower TOA than the already-learned dense representation. LBF therefore learns the SID predictors jointly and activates dense refinement only after the multi-view SID predictor is established.

2.4 COMPLEMENTARY RE-RANKING

Complementary ranking views. Because the SID views use separate codebooks, they are decoded by separate autoregressive heads and produce multiple item-ranking signals that must ultimately be reconciled. Each decoder returns candidate items together with autoregressive probabilities and beam-search statistics; aggregating these outputs yields an SID-based matching view of item relevance. However, this view is inherited from SID decomposition and generation rather than learned directly for item-level ranking. We therefore introduce a second ranking view that directly models history-conditioned item-level relevance.

For each generated candidate j, a history encoder uses the continuous item representations to encode the candidate together with the user's interaction history:

$$z_{u,j} = f_{\text{hist}}(\mathcal{H}_u, j). \tag{5}$$

This representation provides a direct item-level ranking signal without relying on SID decoding.

Ranking fusion. Let $f_{u,j}$ collect the SID-decoder evidence for candidate j, including view-specific generation scores and ranks together with cross-view support and uncertainty information, and let $g_{u,j} = f_{\text{gen}}(f_{u,j})$ denote its encoded representation. The final ranking function combines the direct item-level representation and the SID-based matching evidence:

$$s(u, j) = f_{\text{rank}}(z_{u,j}, g_{u,j}). \tag{6}$$

The reranker is optimized over the items generated by the SID decoders using a standard listwise ranking objective; implementation details are provided in Appendix E.

LBF for re-ranking. History-conditioned item-level relevance must be learned from scratch and therefore has lower TOA than the matching evidence already available from the trained SID decoders. LBF first establishes the item-level ranker, then freezes it and introduces the SID-based matching pathway for refinement.

3 EXPERIMENTS

Our experiments address four research questions. **RQ1**: How does CVGR^2 compare with strong sequential and SID-based generative recommenders? **RQ2**: Do SID views induced by different relational geometries provide complementary predictive evidence, and can ITV exchange information while preserving view-specific predictions? **RQ3**: When dense representations are more readily exploitable than the SID predictor, does LBF improve their utilization and preserve SID predictive capability? **RQ4**: Are history-conditioned item-level relevance and SID-based matching evidence complementary ranking views, and does LBF improve their fusion?

3.1 EXPERIMENTAL SETTINGS

Data and protocol. We conduct experiments on ten datasets of the Amazon Reviews 2023 benchmark (Hou et al., 2024): *Musical Instruments* (MI), *Video Games* (VG), *Industrial & Scientific* (IS), *Baby Products* (BP), *Arts, Crafts & Sewing* (ACS), *Sports & Outdoors* (SO), *Toys & Games* (TG), *Health & Household* (HH), *Beauty & Personal Care* (BPC), and *Books* (BK). We follow the preprocessing and sequential evaluation protocol of ReSID (Liang et al., 2026). For each user, interactions are chronologically ordered, with the last interaction used for testing and the penultimate interaction for validation. Dataset statistics and preprocessing details are provided in Appendix C.

Representation instantiation. We instantiate three SID views using FAMAE behavioral representations (Liang et al., 2026), Sentence-T5 textual representations (Ni et al., 2022), and Node2Vec graph representations (Grover & Leskovec, 2016), each independently quantized with GAOQ. The final generator uses graph-based dense refinement, and the history-aware reranker uses frozen text-based item representations. Construction and optimization details are provided in Appendices B and E.

Compared methods. We compare against two categories of baselines. *(i) Item-ID-based sequential recommenders:* HGN (Ma et al., 2019), SASRec (Kang & McAuley, 2018), BERT4Rec (Sun et al., 2019), and S^3-Rec (Zhou et al., 2020). For a fair comparison, all sequential baselines are augmented with item-side information. *(ii) SID-based generative recommenders:* TIGER (Rajput et al., 2023), LETTER (Wang et al., 2024a), EAGER (Wang et al., 2024b), ETEGRec (Liu et al., 2024), ReSID (Liang et al., 2026), and LIGER (Yang et al., 2025a). We use CVGR^2-SUM for the generator-only variant ranked by SumProb and CVGR^2 for the complete framework with history-aware re-ranking. Detailed baseline descriptions are provided in Appendix D.

Evaluation and implementation. We report Recall@K and NDCG@K for $K \in \{5, 10\}$. All test results are obtained from checkpoints selected exclusively on validation data. Evaluation and implementation details are provided in Appendix E.

Table 1: Overall R@10 performance (%) on ten datasets from Amazon Reviews 2023. All sequential baselines use item's side-information. Avg. Gain denotes the relative improvement of CVGR^2 over each method in terms of mean R@10 across the ten datasets. The best result in each dataset is in **bold**, and the second-best is underlined.

Method	MI	VG	IS	BP	ACS	SO	TG	HH	BPC	BK	Avg. Gain
HGN	4.6886	5.4091	3.2148	2.7780	2.7142	1.9707	1.9291	2.0159	2.1948	3.9676	+104.64%
SASRec	6.3905	9.1364	5.3782	4.2475	4.6368	3.1899	4.0110	2.8745	3.3985	6.2910	+27.53%
BERT4Rec	6.1357	8.7932	4.8756	3.8952	4.3878	2.8641	3.5623	2.7350	3.1184	5.6425	+37.36%
S^3-Rec	6.1085	8.4322	4.4140	3.8595	4.2022	2.9693	3.3820	2.7416	3.2168	6.3910	+38.24%
TIGER	5.9190	8.8383	4.6028	4.0583	4.4620	2.7946	3.2183	2.7617	2.9882	4.0820	+44.53%
LETTER	6.2282	9.1442	4.8166	4.1582	4.6396	2.8844	3.4012	2.7704	3.1513	4.5707	+38.09%
EAGER	5.2274	8.6333	4.0567	3.3053	3.7491	2.3618	2.8914	2.2323	2.6510	4.9719	+57.68%
ETEGRec	5.7880	8.6875	3.9300	3.6881	3.8885	2.3611	2.6966	2.2890	2.7418	2.8423	+62.41%
LIGER	6.6107	9.6052	4.9808	4.1988	4.3261	2.8716	3.4089	2.3489	3.0058	2.5024	+44.09%
ReSID	6.4465	9.2664	5.1173	4.4072	4.8498	3.2309	3.9247	3.1703	3.5733	7.3707	+23.06%
CVGR^2-SUM	**6.9530**	9.9313	**5.4181**	4.7149	5.3648	3.5769	4.5406	3.3861	3.9225	8.6569	+11.92%
CVGR^2	6.8936	**11.4650**	5.3805	**5.1350**	**5.9505**	**3.9725**	**5.5959**	**3.8887**	**4.4748**	**10.4410**	–

3.2 Overall Performance

Table 1 reports R@10 across all ten datasets. Complete R@5, R@10, N@5, and N@10 results are provided in Appendix F.

Compared with ReSID, the strongest baseline on average, CVGR^2-SUM improves mean R@10 by 9.95% and outperforms it on all ten datasets, showing that the generator-side design already provides a strong improvement before re-ranking. The complete CVGR^2 improves mean R@10 by 23.06% over ReSID and achieves the best result on eight of the ten datasets. Relative to CVGR^2-SUM , re-ranking adds a further 11.92% in mean R@10. MI and IS are the only datasets where re-ranking does not improve the direct generator result; they also have among the smallest effective reranker training sets in our evaluation, which may be the main cause, i.e., an under-trained ranker. Appendix G reports the corresponding data-sufficiency statistics as a descriptive diagnostic.

3.3 Complementary SID Views and ITV

We first test whether the selected SID views provide complementary predictive evidence and whether cross-view interaction improves their joint modeling without collapsing view-specific predictions. Table 2 compares the corresponding modeling variants on BP and BPC, with dense refinement disabled to isolate the contributions.

Independent Generators train the three SID predictors separately and aggregate their item-level scores only after decoding. Temporal-only jointly optimizes three view-specific streams with shared temporal modeling but no cross-view attention. Input Mean Fusion collapses aligned view representations before sequence modeling. ITV preserves separate view streams while alternating temporal and cross-view attention. The encoder depth is held fixed across variants: each predictor uses the same number of Transformer blocks.

Independent Generators reach 4.6037 and 3.7610 R@10 after aggregation on BP and BPC, respectively, exceeding every single-view generator on both datasets. Because these models are trained independently, this gain directly establishes predictive complementarity among the selected SID views rather than an effect of parameter sharing or cross-view interaction. Temporal-only yields a small additional gain on BP (4.6263) but falls below independent aggregation on BPC (3.5525), suggesting that joint parameterization alone does not consistently improve multi-view prediction.

Input Mean Fusion strengthens every individual decoder relative to the single-view generators on both datasets, but its aggregated R@10 falls below Independent Generators on BP (4.5584) and improves only slightly on BPC (3.7835). ITV also increases agreement, but achieves the strongest aggregated R@10 on both datasets and retains slightly more distinct candidate evidence than Input Mean Fusion.

Table 2: Multi-view SID modeling and predictive complementarity on BP and BPC, with dense refinement disabled. Avg. Union Size is the mean number of unique items after merging three beam-50 decoder outputs. 1-view Share is the fraction of union-recalled targets retrieved by exactly one decoder.

Variant	R@10 (%)				Avg. Union Size	1-view Share among Union Hits (%)
	Behavior	Text	Graph	CVGR^2-SUM		
BP						
Behavioral SID only	4.3939	–	–	–	–	–
Textual SID only	–	4.3992	–	–	–	–
Graph-based SID only	–	–	3.4810	–	–	–
Independent Generators	4.3939	4.3992	3.4810	4.6037	98.21	39.0377
Temporal-only Encoder	4.4192	4.3706	3.4984	4.6263	96.43	37.7568
Input Mean Fusion	4.4605	4.4705	4.2554	4.5584	80.47	26.6954
ITV Encoder	**4.5544**	**4.5164**	**4.3120**	**4.6496**	81.40	27.2261
BPC						
Behavioral SID only	3.6156	–	–	–	–	–
Textual SID only	–	3.2921	–	–	–	–
Graph-based SID only	–	–	2.9698	–	–	–
Independent Generators	3.6156	3.2921	2.9698	3.7610	98.83	37.1010
Temporal-only Encoder	3.4171	3.1206	2.8326	3.5525	98.80	37.7136
Input Mean Fusion	3.6743	**3.5959**	**3.5993**	3.7835	83.65	26.9653
ITV Encoder	**3.7566**	3.5851	3.4514	**3.8294**	85.65	28.0088

Table 3: LBF for dense refinement on BP. "Dense Off" disables the dense pathway only at inference time without retraining. SID-only is the original three-view ITV generator without dense item embeddings. All entries report test R@10 (%).

Representation	SID-only	Early	Early, Dense Off	LBF	LBF, Dense Off
Text	4.6496	4.5604	2.4969	**4.6895**	4.6589
Graph	4.6496	4.1715	1.8411	**4.7149**	4.6003
FAMAE	**4.6496**	3.3465	0.4861	4.5810	4.5570

This suggests that explicit cross-view interaction is most effective when the view-specific prediction spaces are preserved rather than collapsed. Appendix G reports all pairwise SID-view combinations.

3.4 LBF for Dense Refinement

Figure 1(b) shows a representative training asymmetry: exposing a recommendation-oriented dense representation from the start rapidly lowers training loss while validation performance deteriorates before the SID predictor is fully established. We now test whether this behavior generalizes across dense representation sources and whether LBF preserves the SID predictor.

Table 3 jointly evaluates the effect of LBF on final recommendation performance and on the predictive capability retained by the SID pathway. LBF outperforms Early introduction for all three representation families. The gain is modest for the independently pretrained textual representation and substantially larger for the graph and FAMAE representations learned from recommendation interactions. This pattern is consistent with the TOA interpretation: already-learned signals that are more closely aligned with the downstream interactions can be exploited more readily during GR training. LBF improves how such dense information is utilized, but does not guarantee positive incremental value: the FAMAE variant recovers substantially from 3.3465 to 4.5810, yet remains below the SID-only reference of 4.6496. A plausible explanation is that FAMAE is trained with a history-conditioned next-item objective that is closely aligned with SID autoregressive prediction.

Table 4: Complementary ranking views and LBF on BP and BPC. All variants rank the same generated candidate sets within each dataset.

Ranking method	Configuration			R@10 (%)	
	Item relevance	SID evidence	Schedule	BP	BPC
CVGR^2-SUM	No	Fixed	–	4.7149	3.9225
SID-evidence-only Fusion	No	Learned	–	4.6716	3.8610
Item-level-only Reranker	Yes	No	–	4.4126	4.0067
Joint Fusion	Yes	Learned	Joint	4.7961	4.4298
CVGR^2	Yes	Learned	LBF	**5.1350**	**4.4748**

This makes its dense signal highly accessible during optimization, while leaving less incremental information once the SID predictor is fully established.

The Dense-Off intervention further reveals what changes inside the learned predictor. Removing the dense pathway from Early models reduces R@10 by 2.06–2.86 points across the three representations, with the FAMAE model collapsing from 3.3465 to 0.4861. In contrast, the same intervention reduces the LBF models by only 0.02–0.11 points, leaving their performance close to the SID-only predictor. Thus, Early fusion shifts substantial predictive reliance toward the readily exploitable dense pathway, whereas LBF first preserves a strong SID-specific predictor and subsequently uses continuous information as refinement. Appendix G provides dense-to-SID pathway-strength trajectories as a complementary diagnostic, showing how the relative strength of the dense residual and SID embedding pathways evolves during training. Importantly, these ratios are not treated as direct numerical measurements of TOA.

3.5 Complementary Ranking Views and LBF

The generator provides SID-based matching evidence through decoder probabilities, beam ranks, and cross-view support, while the re-ranker adds a second ranking view by directly learning history-conditioned item-level relevance. Table 4 first tests whether the two ranking views are complementary, and then evaluates LBF by comparing joint and staged training with the same two pathways. All variants rank the same generated candidate sets.

The SID-evidence-only learned fusion performs similarly to fixed SumProb despite using a more flexible learned transformation of decoder evidence, showing that additional score modeling alone does not explain the final gain. The item-level-only reranker is weaker than SumProb on BP but stronger on BPC, suggesting that neither ranking view uniformly dominates the other. Joint Fusion reaches R@10 of 4.7961 on BP and 4.4298 on BPC, exceeding either isolated ranking view in both comparisons, a pattern consistent with predictive complementarity.

LBF further raises R@10 to 5.1350 and 4.4748 on BP and BPC, respectively, corresponding to relative gains of 7.07% and 1.02% over Joint Fusion with the same two information pathways. This isolates training order as an independent contributor: the lower-TOA item-level ranking function is established before the already-available SID matching evidence is introduced. A support-stratified analysis of where the two ranking views contribute is provided in Table 5.

Where does the ranking complementarity come from? To understand why the two ranking views are complementary, we stratify test instances whose targets appear in the generated candidate set by the number of SID decoders that retrieve the target. Table 5 reports the number of targets ranked in the Top-10 within each support group.

The two ranking views exhibit sharply different strengths across these support regimes. SID-based matching is strongest when the target is retrieved by all three decoders, where agreement across relational views directly reinforces the item-level matching score. History-conditioned item-level relevance behaves differently: it can promote a candidate directly from its compatibility with the user's interaction history, and is therefore particularly effective when SID support is sparse. The item-level-only ranker consequently recovers far more Top-10 targets in the one- and two-view groups, despite being weaker than CVGR^2-SUM overall.

Table 5: Re-ranking performance conditioned on SID-view support on BP. #Inst. is the number of test instances whose target is retrieved by exactly one, two, or all three SID decoders. Each method column reports the number of these targets ranked in the Top-10. Joint Fusion trains the two ranking pathways jointly from scratch.

SID Support	#Inst.	CVGR^2-SUM	SID-only Fusion	Item-level Only	Joint Fusion	CVGR^2
One view	6,352	9	15	**923**	62	247
Two views	4,973	206	187	**969**	348	578
Three views	11,737	6,866	6,814	4,735	6,793	**6,887**

This asymmetry explains why the two ranking views are genuinely complementary rather than simply two parameterizations of the same score. It also clarifies the failure of naive joint training. Although Joint Fusion has access to both pathways, its hit distribution remains much closer to the SID-evidence-dominated pattern: it gains little in the one-view regime and retains most of its successes in the three-view regime. This is consistent with the TOA interpretation—the already-formed SID matching evidence is immediately exploitable, while the history-conditioned item-level ranking function still needs to develop its own predictive structure.

LBF changes this balance. Notably, 415 of the 509 additional Top-10 hits obtained by LBF over Joint Fusion (81.5%) come from the one- and two-view support groups. Thus, most of the LBF gain arises precisely where SID-based matching evidence is sparse and the complementary history-conditioned ranking view has the greatest opportunity to contribute. By establishing the item-level ranker before introducing SID-based matching evidence, CVGR^2 retains substantially more of the sparse-support capability of the direct relevance view while still achieving the strongest result in the three-view regime. The benefit of LBF is therefore not to suppress strong SID evidence, but to prevent it from preempting the learning of a complementary ranking function.

Together with the generator-side results in Table 2, these analyses reveal two distinct forms of complementarity in CVGR^2 . Different SID views provide alternative relational evidence during generation, while the history-aware ranker provides an item-level relevance view that is especially useful when SID agreement is weak. CVGR^2 defines how these signals are combined, and LBF preserves their distinct predictive roles during learning.

4 Conclusion

We study how to preserve predictive complementarity when combining multiple information pathways in SID-based generative sequential recommendation. Our results show that complementary representations do not automatically yield complementary predictors: readily exploitable signals can dominate learning before alternative predictive structures are sufficiently established. To address this failure mode, CVGR^2 determines *how* complementary SID, dense, and ranking signals are combined, while LBF uses their relative TOA to determine *when* they should be fused. Experiments across ten public datasets demonstrate consistent improvements and show that LBF better preserves independently learned predictive capability in both generation and re-ranking. More broadly, our findings suggest that representation-level complementarity is only useful when optimization also allows the corresponding predictive structures to emerge.

AI use statement

In this work, we used generative AI tools to assist with language polishing and improving the clarity and presentation of the manuscript, as well as with code debugging, testing, and consistency checking during implementation and experimentation. Generative AI tools were not used to originate the core research idea, formulate the main research hypotheses, or design the proposed methodology and algorithmic contributions. They were also not used to generate experimental results or determine the conclusions of the study.

All AI-assisted text revisions were reviewed and edited by the authors. The authors take full responsibility for the final content of this work, including all text, code, claims, analyses, and artifacts produced with the aid of generative AI.

Ethics statement

This work uses publicly available Amazon Reviews 2023 benchmark data and does not involve new user data collection, human-subject experiments, or real-world deployment. We do not attempt to identify users or infer sensitive attributes.

Reproducibility statement

Appendix C describes dataset preprocessing and the chronological data splits. Appendix B specifies the multi-view generator, dense residual pathway, candidate consolidation, and re-ranking objectives. Appendix E provides representation and SID construction settings, model dimensions, optimization settings, checkpoint-selection criteria, evaluation metrics, and baseline implementation details. Additional experimental results and analyses are reported in Appendices F and G. The core implementation is available anonymously at `https://anonymous.4open.science/r/MSGR2-40EF`.

References

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, 2016.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11523–11532, 2022.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. Learning to rank in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8716–8723, 2024. doi: 10.1609/aaai.v38i8.28717.

Yu Liang, Zhongjin Zhang, Yuxuan Zhu, Kerui Zhang, Zhiluohan Guo, Wenhang Zhou, Zonqi Yang, Kangle Wu, Yabo Ni, Anxiang Zeng, Cong Fu, Jianxin Wang, and Jiazhi Xia. Rethinking generative recommender tokenizer: Recsys-native encoding and semantic quantization beyond llms. *arXiv preprint arXiv:2602.02338*, 2026.

Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. End-to-end learnable item tokenization for generative recommendation. *arXiv preprint arXiv:2409.05546*, 2024.

Chen Ma, Peng Kang, and Xue Liu. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 825–833, 2019.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, 2022.

Gustavo Penha, Edoardo D'Amico, Marco De Nadai, Enrico Palumbo, Alexandre Tamborrino, Ali Vardasbi, Max Lefarov, Shawn Lin, Timothy Heath, Francesco Fabbri, and Hugues Bouchard. Semantic IDs for joint generative search and recommendation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, 2025. doi: 10.1145/3705328.3759300.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H. Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10299–10315, 2023.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450, 2019.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 565–573, 2018. doi: 10.1145/3159652.3159656.

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2400–2409, 2024a.

Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, and Zhenhua Dong. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3245–3254, 2024b.

Liu Yang, Fabian Paischer, Kaveh Hassani, Jiacheng Li, Shuai Shao, Zhang Gabriel Li, Yun He, Xue Feng, Nima Noorshams, Sem Park, Bo Long, Robert D. Nowak, Xiaoli Gao, and Hamid Eghbalzadeh. Unifying generative and dense retrieval for sequential recommendation. *Transactions on Machine Learning Research*, 2025a.

Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, et al. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. *arXiv preprint arXiv:2503.02453*, 2025b.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1893–1902, 2020.

Jieming Zhu, Mengqun Jin, Qijiong Liu, Zexuan Qiu, Zhenhua Dong, and Xiu Li. CoST: Contrastive quantization based semantic tokenization for generative recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems*, pp. 969–974, 2024.

Jing Zhu, Mingxuan Ju, Yozen Liu, Danai Koutra, Neil Shah, and Tong Zhao. Beyond unimodal boundaries: Generative recommendation with multimodal semantics. *arXiv preprint arXiv:2503.23333*, 2025.

Table 6: Full recommendation results (%). Datasets: MI = Musical Instruments, VG = Video Games, IS = Industrial & Scientific, BP = Baby Products, ACS = Arts, Crafts & Sewing, SO = Sports & Outdoors, TG = Toys & Games, HH = Health & Household, BPC = Beauty & Personal Care, and BK = Books. All sequential baselines use item-side information, as in Table 1. The best result is in **bold**, and the second-best is underlined.

(A) MI / VG / IS / BP / ACS

Method	MI				VG				IS				BP				ACS			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
HGN	2.8679	4.6886	1.8345	2.4192	3.3068	5.4091	2.1226	2.7972	1.9186	3.2148	1.2170	1.6334	1.6916	2.7780	1.0849	1.4326	1.6609	2.7142	1.0699	1.4086
SASRec	3.9682	6.3905	2.4796	3.2572	5.3213	9.1364	2.9451	4.1721	3.3332	5.3782	1.8483	2.5030	2.5682	4.2475	1.5675	2.1066	2.7452	4.6368	1.5683	2.1764
BERT4Rec	3.7673	6.1357	2.4084	3.1691	5.3690	8.7932	3.3686	4.4675	2.9644	4.8756	1.8612	2.4761	2.4052	3.8952	1.5457	2.0238	2.7542	4.3878	1.7605	2.2859
S^3-Rec	3.7972	6.1085	2.4234	3.1669	5.1600	8.4322	3.2577	4.3104	2.7360	4.4140	1.6871	2.2266	2.3656	3.8595	1.5203	1.9990	2.6050	4.2022	1.6766	2.1896
TIGER	3.8476	5.9190	2.5134	3.1800	5.6890	8.8383	3.7351	4.7494	2.9723	4.6028	1.9190	2.4422	2.5622	4.0583	1.6925	2.1728	2.8534	4.4620	1.8685	2.3854
LETTER	4.0642	6.2282	2.6852	3.3796	5.9173	9.1442	3.8935	4.9278	3.0732	4.8166	1.9747	2.5363	2.6075	4.1582	1.7122	2.2088	2.9725	4.6396	1.9575	2.4938
EAGER	3.2678	5.2274	2.0965	2.7261	5.4712	8.6333	3.5659	4.5825	2.5171	4.0567	1.6298	2.1247	1.9895	3.3053	1.2611	1.6826	2.4005	3.7491	1.5513	1.9846
ETEGRec	3.7236	5.7880	2.4400	3.1049	5.5838	8.6875	3.6657	4.6612	2.5092	3.9300	1.6615	2.1176	2.2865	3.6881	1.5107	1.9608	2.4936	3.8885	1.6158	2.0628
LIGER	4.2406	6.6107	2.7350	3.4983	6.0745	9.6052	3.9031	5.0382	3.0791	4.9808	2.0016	2.6117	2.6474	4.1988	1.6968	2.1929	2.7776	4.3261	1.8052	2.3041
ReSID	4.1742	6.4465	2.7348	3.4643	5.9736	9.2664	3.9552	5.0147	3.4036	5.1173	2.1760	2.7260	2.8452	4.4072	1.8636	2.3647	3.1277	4.8498	2.0534	2.6055
$CVGR^2$-SUM	4.4432	**6.9530**	2.9195	3.7291	6.3773	9.9313	4.1887	5.3314	**3.4472**	**5.4181**	**2.2148**	**2.8507**	3.0236	4.7149	1.9945	2.5376	3.4559	5.3648	2.2732	2.8879
$CVGR^2$	**4.5166**	6.8936	**2.9733**	**3.7382**	**7.8271**	**11.4650**	**5.3754**	**6.5483**	3.4195	5.3805	2.1799	2.8096	**3.4477**	**5.1350**	**2.3363**	**2.8779**	**4.0926**	**5.9505**	**2.7897**	**3.3875**

(B) SO / TG / HH / BPC / BK

Method	SO				TG				HH				BPC				BK			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
HGN	1.2134	1.9707	0.7781	1.0209	1.1877	1.9291	0.7451	0.9829	1.2372	2.0159	0.7811	1.0310	1.3651	2.1948	0.8797	1.1461	2.3055	3.9676	1.3807	1.9149
SASRec	1.9708	3.1899	1.2175	1.6090	2.4299	4.0110	1.2687	1.7782	1.7616	2.8745	1.0840	1.4418	2.0802	3.3985	1.2640	1.6878	3.6937	6.2910	2.0161	2.8546
BERT4Rec	1.7491	2.8641	1.1116	1.4696	2.2320	3.5623	1.3769	1.8050	1.6832	2.7350	1.0589	1.3965	1.9234	3.1184	1.2254	1.6095	3.3694	5.6425	1.9901	2.7223
S^3-Rec	1.8401	2.9693	1.1723	1.5352	2.0862	3.3820	1.2843	1.7013	1.6970	2.7416	1.0669	1.4029	1.9813	3.2168	1.2546	1.6521	3.8093	6.3910	2.0716	2.9042
TIGER	1.8022	2.7946	1.1906	1.5087	2.1413	3.2183	1.4147	1.7612	1.8022	2.7617	1.2133	1.5219	1.9198	2.9882	1.2547	1.5985	2.6537	4.0820	1.7114	2.1921
LETTER	1.8655	2.8844	1.2352	1.5620	2.2125	3.4012	1.4668	1.8497	1.8028	2.7704	1.2145	1.5257	2.0267	3.1513	1.3303	1.6912	3.0812	4.5707	2.0855	2.5643
EAGER	1.4549	2.3618	0.9536	1.2450	1.9077	2.8914	1.2709	1.5870	1.3761	2.2323	0.8893	1.1640	1.6434	2.6510	1.0712	1.3947	3.4163	4.9719	2.3433	2.8440
ETEGRec	1.4790	2.3611	0.9650	1.2483	1.7802	2.6966	1.1629	1.4574	1.4269	2.2890	0.9216	1.1984	1.7661	2.7418	1.1501	1.4632	1.7585	2.8423	1.1039	1.4519
LIGER	1.8363	2.8716	1.1986	1.5315	2.2332	3.4089	1.4483	1.8273	1.4426	2.3489	0.9218	1.2132	1.8661	3.0058	1.2000	1.5655	1.6055	2.5024	1.0366	1.3248
ReSID	2.0972	3.2309	1.3714	1.7366	2.6140	3.9247	1.7355	2.1574	2.0805	3.1703	1.3951	1.7453	2.3035	3.5733	1.5035	1.9120	5.2966	7.3707	3.6928	4.3612
$CVGR^2$-SUM	2.3297	3.5769	1.5439	1.9453	3.0415	4.5406	2.0042	2.4867	2.2381	3.3861	1.4977	1.8666	2.5704	3.9225	1.6867	2.1213	6.2282	8.6569	4.4039	5.1857
$CVGR^2$	**2.7171**	**3.9725**	**1.8735**	**2.2772**	**4.0875**	**5.5959**	**2.9001**	**3.3863**	**2.7095**	**3.8887**	**1.8822**	**2.2610**	**3.0963**	**4.4748**	**2.1334**	**2.5768**	**7.9548**	**10.4410**	**5.8557**	**6.6582**

A Related Work

A.1 Sequential Recommendation

Sequential recommendation aims to learn users' evolving preferences from historical interaction sequences to predict future items. Early neural approaches use recurrent (Hidasi et al., 2015) and convolutional (Tang & Wang, 2018) networks to model user behavior, while self-attention has become widely used to capture dependencies among items in a sequence. SASRec (Kang & McAuley, 2018) employs unidirectional self-attention to model historical interactions, BERT4Rec (Sun et al., 2019) learns contextualized item representations through bidirectional masked-item prediction, and S^3-Rec (Zhou et al., 2020) enhances sequence representations with multiple self-supervised pretraining tasks. HGN (Ma et al., 2019) takes a different approach, using hierarchical gating to capture users' long- and short-term interests. Despite differences in architectures and training objectives, these methods typically follow a similar prediction procedure: historical items are mapped to embeddings, a sequence model summarizes the user's current interests, and candidate items are scored according to their relevance to the resulting representation.

A.2 SID-based Generative Recommendation

SID-based generative recommendation represents items as short sequences of discrete tokens and directly generates target identifiers from user histories. Items with similar representations can share code tokens, enabling knowledge sharing across related items. Compositional codes also represent a large item catalog with a relatively small shared vocabulary, reducing the storage required for separate trainable item embeddings (Rajput et al., 2023; Yang et al., 2025a). TIGER (Rajput et al., 2023) is a representative approach: it quantizes textual item representations into SIDs using RQ-VAE (Lee et al., 2022), trains an encoder–decoder model to autoregressively predict target SIDs, and generates recommendation candidates through beam search. Subsequent studies improve SID quality and its alignment with recommendation. CoST (Zhu et al., 2024) and LETTER (Wang et al., 2024a) explore improved quantization and learnable item tokenization, ETEGRec (Liu et al., 2024) jointly optimizes the item tokenizer and generative recommender, and ReSID (Liang et al., 2026) develops recommendation-oriented representation learning and quantization to improve information preservation and autoregressive predictability.

Beyond SID construction, another line of work uses multiple information sources to enrich the item relations captured by generative recommendation. EAGER (Wang et al., 2024b) models behavioral and semantic information with a shared encoder and separate decoders, using auxiliary tasks to promote collaboration between the two streams. MGR (Zhu et al., 2025) studies how modality choices and fusion strategies affect generative recommendation, using modality alignment and fusion mechanisms to exploit complementary semantics. These studies highlight the importance of both preserving the characteristics of different sources and enabling interactions among them. CVGR^2 focuses on jointly modeling independently constructed SID spaces: its ITV Encoder alternates within-view sequence modeling and cross-view information exchange, while view-specific representation streams and decoders allow each predictor to retain its own relational structure and benefit from complementary information in other views.

Other studies combine discrete generation with continuous representations to recover fine-grained item information and improve final ranking. LIGER (Yang et al., 2025a) unifies generative and dense retrieval, using continuous representations to strengthen candidate scoring. COBRA (Yang et al., 2025b) cascades sparse SID and dense vector generation and combines generation probabilities with nearest-neighbor similarity during inference. In generative retrieval, LTRGR (Li et al., 2024) further studies how ranking objectives can improve models trained with identifier-generation likelihood. These approaches demonstrate complementary roles for discrete codes, continuous representations, and relevance scoring. CVGR^2 uses dense item representations as residual refinement for SID generation and introduces a history-aware re-ranker that combines direct item-level relevance with SID-based matching evidence. Beyond how these signals are combined, our focus is also on when they should enter optimization: LBF first establishes the lower-TOA complementary predictor before introducing the more readily exploitable pathway.

B Additional Method Details

This section provides the formulation and implementation details omitted from the main method section for clarity. We follow the notation of Section 2 and describe the multi-view SID generation, candidate consolidation, and re-ranking objectives used by CVGR^2 .

B.1 SID Views and Autoregressive Generation

For each view $v \in \mathcal{V}$, item i is represented by an independently quantized Semantic ID

$$\mathrm{SID}^{(v)}(i) = [c_{i,v,1}, \ldots, c_{i,v,S}], \tag{7}$$

where S is the SID length and $c_{i,v,s}$ denotes the token at slot s in view v. The codebooks are independent across views, so the same token index in two views does not imply shared semantics.

Given interaction history $\mathcal{H}_u$, decoder D_v predicts the target SID autoregressively:

$$p_v\Big(\mathrm{SID}^{(v)}(i^+) \mid \mathcal{H}_u\Big) = \prod_{s=1}^{S} p_v\big(c_{i^+,v,s} \mid c_{i^+,v,<s}, \mathcal{H}_u\big)\,, \tag{8}$$

where $c_{i^+,v,<s} = [c_{i^+,v,1}, \ldots, c_{i^+,v,s-1}]$. The training objective sums the token-level negative log-likelihood over SID slots and views, as given in Eq. (3).

View instantiation. We instantiate three SID views using behavioral representations from FAMAE (Liang et al., 2026), textual representations from Sentence-T5 (Ni et al., 2022), and graph-based representations from Node2Vec (Grover & Leskovec, 2016). Each representation space is independently quantized with GAOQ (Liang et al., 2026) to produce its own SID codebook. Details are provided in Section E.1.

Multi-view SID input. For item i_l at interaction position l, SID slot s, and view v, the discrete input is

$$x_{l,s,v} = e_v^{\mathrm{SID}}(c_{i_l,v,s}) + e_s^{\mathrm{slot}}, \tag{9}$$

where SID embedding tables are view-specific and e_s^{slot} is shared across views.

B.2 ITV Encoder Implementation

For a batch of histories, the discrete input is organized by interaction position, SID slot, and view. After combining the interaction-position and SID-slot dimensions, the encoder input has shape

$$H^{(0)} \in \mathbb{R}^{B\times T\times V\times d}, \qquad T = LS, \tag{10}$$

where B is the batch size, $V = |\mathcal{V}|$, and d is the hidden dimension.

The ITV encoder alternates temporal and view attention while preserving the view axis. For temporal attention, the view dimension is folded into the batch dimension,

$$[B, T, V, d] \to [BV, T, d], \tag{11}$$

and self-attention is applied separately to each view's SID sequence. For view attention, the temporal dimension is folded into the batch dimension,

$$[B, T, V, d] \to [BT, V, d], \tag{12}$$

and self-attention exchanges information across views at the same historical position and SID slot. The resulting sequence of operations follows the alternating pattern defined in Eq. (2). Each view-specific decoder then reads its corresponding view slice from the final encoder state.

B.3 Dense Residual Pathway

Let $x_i \in \mathbb{R}^{d_x}$ denote the graph-based dense item representation used for generator refinement. For each SID slot s, we compute a slot-specific residual

$$r_{i,s} = \mathrm{RMSNorm}(W_s x_i + b_s). \tag{13}$$

The residual is shared across SID views at the same slot and added to the discrete SID and slot embeddings (slot position encoding):

$$h_{i,v,s} = e_v^{\mathrm{SID}}(c_{i,v,s}) + e_s^{\mathrm{slot}} + r_{i,s}. \tag{14}$$

Under LBF, Generator Stage 1 disables $r_{i,s}$ and learns the multi-view SID predictor alone. Generator Stage 2 initializes from the validation-selected Stage 1 checkpoint and activates the graph-based dense pathway. Optimization details are provided in Section E.

B.4 Candidate Consolidation and SID-Based Matching Evidence

Candidate consolidation. Each decoder D_v returns $K = 50$ SID sequences by beam search. Let $p_v(\mathbf{c}_{v,k} \mid \mathcal{H}_u)$ denote the autoregressive probability of the k-th returned sequence, as defined in Eq. (8). We normalize these probabilities over the returned beams:

$$\bar{p}_{v,k} = \frac{p_v(\mathbf{c}_{v,k} \mid \mathcal{H}_u)}{\sum_{r=1}^{K} p_v(\mathbf{c}_{v,r} \mid \mathcal{H}_u)}. \tag{15}$$

Generated SID sequences are mapped back to item IDs, and invalid sequences are discarded. If the k-th beam from decoder D_v maps to item j, we assign $p_v(j \mid \mathcal{H}_u) = \bar{p}_{v,k}$. Items not retrieved by D_v receive a score of zero.

Let $\mathcal{C}_u^v$ denote the set of items retrieved by decoder D_v, and let

$$\mathcal{C}_u = \bigcup_{v \in \mathcal{V}} \mathcal{C}_u^v \tag{16}$$

be the merged candidate set. After this mapping, predictions from different SID code spaces become comparable at the item level.

A simple generator-only ranking rule aggregates the evidence assigned to the same item across views:

$$s_{\mathrm{SumProb}}(u, j) = \sum_{v \in \mathcal{V}} p_v(j \mid \mathcal{H}_u). \tag{17}$$

SumProb aggregates view-specific scores at the item level, allowing candidates retrieved by multiple views to receive contributions from each supporting decoder. Ranking $\mathcal{C}_u$ by s_{SumProb} gives the generator-only ranking used in our ablations.

SID-based matching evidence. The learned re-ranker retains richer SID-based matching evidence than SumProb alone. For each user–candidate pair, we construct a 16-dimensional feature vector $f_{u,j}$ with three groups of features:

View-specific evidence (9 dimensions). For each decoder D_v, we include the beam-normalized item score $p_v(j \mid \mathcal{H}_u)$, the reciprocal rank $1/r_v(j)$, and a binary retrieval indicator. Here, $r_v(j)$ is the rank of the SID corresponding to item j in the returned beam list, and the indicator equals one if the decoder retrieves item j. All three features are set to zero for unretrieved items.

Cross-view statistics (4 dimensions). We include the sum, maximum, and standard deviation of the three view-specific scores, and the number of views retrieving item j.

Decoder uncertainty (3 dimensions). We include the entropy of each decoder's normalized beam-score distribution. These values are shared across candidates for the same user.

A generation-evidence encoder (MLP) maps these features to

$$g_{u,j} = f_{\mathrm{gen}}(f_{u,j}). \tag{18}$$

This representation is combined with the history-conditioned item-level representation $z_{u,j}$ defined in Eq. (5).

B.5 History-Aware Re-Ranking and LBF Training

History-conditioned item-level ranking. The re-ranker uses frozen Sentence-T5 textual item representations for both historical items and candidates. For each candidate $c \in \mathcal{C}_u$, we form the item sequence $[i_1, \ldots, i_L, c]$. Let $j_t = i_t$ for $t \le L$ and $j_{L+1} = c$. The input representation at each sequence position is

$$e_{u,c,t} = P_{\text{hist}}(x^{\text{text}}_{j_t}) + e^{\text{pos}}_t, \qquad t = 1, \ldots, L+1, \tag{19}$$

where $x^{\text{text}}_{j_t}$ is the frozen textual item representation, P_{hist} is a trainable projection, and e^{pos}_t is the recency-aware positional embedding at position t. The sequence

$$E_{u,c} = [e_{u,c,1}, \ldots, e_{u,c,L+1}]$$

is then processed by a Transformer encoder. The hidden state at the candidate position defines the history-conditioned item-level representation:

$$z_{u,c} = f_{\text{hist}}(E_{u,c})_{L+1}. \tag{20}$$

SID-based matching fusion. Let $g_{u,c}$ denote the SID-based matching representation constructed from the multi-view decoder evidence in Section B.4. The two ranking views are combined additively in the scorer:

$$h_{u,c} = W_h z_{u,c} + b_h + W_g g_{u,c}. \tag{21}$$

The remaining scoring layers f_{score} map $h_{u,c}$ to a scalar relevance score:

$$s(u,c) = f_{\text{score}}(h_{u,c}). \tag{22}$$

For a generated candidate set containing the target item i^+, both stages use the same listwise softmax objective:

$$\mathcal{L}_{\text{rank}} = -\log \frac{\exp s(u, i^+)}{\sum_{c \in \mathcal{C}_u} \exp s(u,c)}. \tag{23}$$

The target item is never manually inserted into the generated candidate set. Training instances whose target is absent from $\mathcal{C}_u$ are excluded from the ranking loss, whereas such instances remain misses during validation and testing.

LBF for re-ranking. In Stage 1, we train the history-conditioned pathway and scorer using $h_{u,c} = W_h z_{u,c} + b_h$. In Stage 2, we freeze this base ranker and introduce the generation-evidence branch, yielding $h_{u,c} = W_h z_{u,c} + b_h + W_g g_{u,c}$. Only the generation-evidence encoder and adapter W_g are optimized. The adapter W_g is initialized to zero to preserve the base ranker's initial scores.

C Dataset Details

Following prior work (Wang et al., 2024a;b; Liu et al., 2024; Liang et al., 2026), we adopt the standard 5-core setting, removing users and items with fewer than five interactions. The remaining interactions are ordered chronologically for each user. The last interaction is used for testing, the second-to-last interaction for validation, and earlier interactions for training. Training examples are constructed using a sliding window with a maximum history length of 32.

We use the same structured item fields as ReSID (Liang et al., 2026), including the store identifier and three category levels. Items missing any required field are removed. Table 7 reports the statistics of the resulting datasets.

D Compared Methods

We compare our framework with representative methods from two categories.

(i) Item-ID-based Sequential recommenders.

Table 7: Statistics of the Amazon Reviews 2023 datasets.

Dataset	#Users	#Items	#Interactions	Density
Musical Instruments	57,359	23,742	490,522	0.036%
Video Games	94,515	24,685	772,218	0.033%
Industrial & Scientific	50,886	25,142	394,989	0.031%
Baby Products	150,642	35,024	1,189,171	0.023%
Arts, Crafts & Sewing	196,980	87,449	1,706,484	0.010%
Sports & Outdoors	409,309	151,411	3,333,753	0.005%
Toys & Games	431,411	156,537	3,652,250	0.005%
Health & Household	796,014	183,230	6,860,582	0.005%
Beauty & Personal Care	712,259	193,383	5,785,124	0.004%
Books	775,503	485,218	8,680,494	0.002%

- **HGN** (Ma et al., 2019) models users' short- and long-term interests through a hierarchical gating mechanism over historical interaction sequences.
- **SASRec** (Kang & McAuley, 2018) employs causal self-attention to capture sequential dependencies among historical interactions for next-item prediction.
- **BERT4Rec** (Sun et al., 2019) adopts bidirectional Transformer encoding and masked-item prediction to learn contextualized user representations.
- **S^3-Rec** (Zhou et al., 2020) enhances sequential recommendation through multiple self-supervised objectives followed by next-item fine-tuning.

(ii) SID-based generative recommenders.

- **TIGER** (Rajput et al., 2023) extracts textual item representations, quantizes them into Semantic IDs with RQ-VAE, and autoregressively generates the SID of the next item.
- **LETTER** (Wang et al., 2024a) proposes learnable item tokenization that incorporates hierarchical semantics, collaborative signals, and code-assignment diversity for generative recommendation.
- **EAGER** (Wang et al., 2024b) constructs semantic and behavioral identifiers separately and models them with a two-stream generative architecture.
- **ETEGRec** (Liu et al., 2024) jointly optimizes item tokenization and downstream generative recommendation in an end-to-end framework.
- **ReSID** (Liang et al., 2026) learns recommendation-oriented item representations and constructs SIDs that preserve task-relevant information while improving autoregressive predictability.
- **LIGER** (Yang et al., 2025a) combines SID-based generative candidate retrieval with dense sequential scoring for recommendation. It also augments the prediction with dense representations.

E Implementation Details

E.1 Representation and SID Construction

We use three types of item embeddings to construct the SID views: behavioral embeddings from FAMAE, textual embeddings from Sentence-T5-xxl, and graph-based embeddings from Node2Vec.

Each embedding space is independently quantized with GAOQ (Liang et al., 2026) to produce a three-token SID. We use balanced hierarchical clustering and globally aligned code indices, with the same first- and second-level branching factors across views for each dataset. The third token distinguishes items sharing the same two-token prefix. Table 8 reports the dataset-specific code sizes.

Table 8: Dataset-specific GAOQ code sizes (b_1, b_2, b_3).

MI	VG	IS	BP	ACS	SO	TG	HH	BPC	BK
$(32, 40, 19)$	$(32, 64, 13)$	$(24, 80, 14)$	$(32, 64, 18)$	$(64, 96, 15)$	$(128, 128, 10)$	$(192, 192, 5)$	$(50, 512, 8)$	$(96, 192, 11)$	$(256, 256, 8)$

Behavioral representation. We use the FAMAE item representation from ReSID (Liang et al., 2026). FAMAE uses a two-layer Transformer with four attention heads, hidden size 128, FFN size 512, and dropout 0.1. It is trained with a multi-field masking objective over the item identifier, store identifier, and three category fields, yielding recommendation-oriented item embeddings.

Textual representation. We extract item text embeddings using the pretrained Sentence-T5-xxl encoder (Ni et al., 2022). These embeddings provide the continuous representation space for the textual SID view.

Graph-based representation. We construct a directed item graph from users' interaction sequences in the training set. We add a forward edge for every ordered pair of items in each sequence and an additional reverse edge when their positional distance is at most five. We use inverse user frequency (IUF) weighting to reduce the influence of highly active users: user u contributes $1/\log(1 + m_u)$ to each such edge, where m_u is the number of distinct items in the user's training sequence. An edge is retained only if it belongs to both its source node's top-100 outgoing edges and its destination node's top-100 incoming edges, ranked by weight. We then learn Node2Vec embeddings (Grover & Leskovec, 2016) on this graph using dimension 512, walk length 40, 10 walks per node, context size 3, 10 negative samples, $p = 2$, and $q = 1$.

E.2 Generator

The generator uses hidden size 128, FFN size 512, four attention heads, dropout 0.1, and the encoder order Temporal–View–Temporal–View. Each SID view has an independent autoregressive decoder.

Generator Stage 1 is optimized with AdamW using batch size 2048, learning rate 0.005, weight decay 10^{-5}, five warmup epochs, and cosine learning-rate decay. Training runs for at most 500 epochs. The checkpoint with the highest validation SumProb R@10 is selected.

Generator Stage 2 initializes from the selected Stage 1 checkpoint and activates three slot-specific projections for the graph-based representation. Stage 2 uses learning rate 0.001, three warmup epochs, and at most 200 additional epochs. The checkpoint is again selected by validation SumProb R@10. Each decoder uses beam size 50 during candidate generation.

E.3 Reranker

The candidate cache stores the union of the three beam-50 decoder lists. Historical items and candidates use frozen Sentence-T5 textual representations. The text projection uses dimensions $768 \to 256 \to 128$. The history-aware encoder contains two Transformer layers, four attention heads, and FFN size 256.

The base reranker is optimized with AdamW using batch size 256, learning rate 0.001, weight decay 10^{-5}, and dropout 0.1 for at most 50 epochs. The checkpoint with the best validation R@10 is selected.

The generation feature encoder applies LayerNorm to the 16-dimensional feature vector, followed by a $16 \to 64$ linear layer, GELU activation, another LayerNorm, and dropout. The adapter projects the 64-dimensional feature representation into the first 256-dimensional hidden layer of the scorer. During the score-integration stage, all base-reranker parameters are frozen and only the generation feature encoder and adapter are optimized for at most 30 epochs using learning rate 0.001.

E.4 Evaluation Metrics

Under the leave-one-out protocol, each evaluation instance has one ground-truth next item. Let $\mathcal{U}_{\text{eval}}$ denote the evaluation users, i_u^+ the target item for user u, and $\hat{i}_{u,k}$ the item at rank k in the

recommendation list. Recall@K measures the fraction of targets appearing in the Top-K list:

$$\text{Recall@}K = \frac{1}{|\mathcal{U}_{\text{eval}}|} \sum_{u \in \mathcal{U}_{\text{eval}}} \mathbf{1}\left[i_u^+ \in \{\hat{i}_{u,1}, \ldots, \hat{i}_{u,K}\}\right]. \tag{24}$$

With one relevant item per instance, Recall@K is equivalent to Hit Rate@K. NDCG@K additionally discounts hits at lower ranks. Since the ideal DCG equals one in this setting, it is computed as

$$\text{NDCG@}K = \frac{1}{|\mathcal{U}_{\text{eval}}|} \sum_{u \in \mathcal{U}_{\text{eval}}} \sum_{k=1}^{K} \frac{\mathbf{1}[\hat{i}_{u,k} = i_u^+]}{\log_2(k+1)}. \tag{25}$$

Here, $\mathbf{1}[\cdot]$ is the indicator function. We use $K \in \{5, 10\}$ and report both metrics as percentages. The averages include all evaluation instances; targets absent from the generated candidate set contribute zero to both metrics.

E.5 Baseline Implementations

Baseline architectures and method-specific hyperparameters follow the corresponding papers and implementations.

E-stage and sequential baselines. HGN, SASRec, BERT4Rec, and S^3-Rec use the sequential-baseline setup of ReSID (Liang et al., 2026), retaining their respective architectures and learning objectives. Embeddings have dimension 128; the Transformer-based encoders use two layers, four attention heads, FFN size 512, and dropout 0.1. Item and side-information embeddings are combined by summation. Training uses AdamW with learning rate 0.001, weight decay 10^{-5}, batch size 2048, and at most 500 epochs. S^3-Rec uses self-supervised pretraining followed by next-item fine-tuning. For the generative baselines, ReSID learns behavioral representations with FAMAE, while TIGER, LETTER, EAGER, and LIGER use Sentence-T5-xxl text embeddings where textual representations are required. EAGER retains its behavioral stream, and ETEGRec retains joint tokenizer–recommender learning.

Q-stage. ReSID applies GAOQ to FAMAE representations, using the dataset-specific branching factors described in Section E.1. TIGER, LETTER, EAGER, ETEGRec, and LIGER retain their method-specific tokenization procedures and reported quantization settings. TIGER and LIGER use RQ-VAE; LETTER and ETEGRec use RQ-VAE-based tokenizers with their respective regularization and joint-learning objectives; EAGER uses hierarchical k-means to construct behavioral and textual SIDs.

G-stage and retrieval. Following ReSID's generative-baseline setup, the T5-style backbone uses four encoder layers, four decoder layers, hidden size 128, FFN size 512, four attention heads, and dropout 0.1, while preserving method-specific decoding branches and objectives. Optimization uses AdamW with learning rate 0.005, weight decay 10^{-5}, cosine learning-rate decay, batch size 2048 (512 for LIGER), and at most 500 epochs. Beam search uses width 50. LIGER uses the same backbone and adds dense-representation modules following its original formulation (Yang et al., 2025a). These modules are trained jointly with SID generation using its dense retrieval objective.

F Full Results of Main Experiments

The main paper reports the compact R@10 comparison across all ten datasets. Table 6 reports the complete R@5, R@10, N@5, and N@10 results for the same methods as Table 1. All sequential baselines use item-side information.

G Additional Results and Analyses

This section complements the main experiments with analyses that are useful for interpreting the proposed design. We first expand the SID-view ablation to pairwise view combinations and examine candidate-set complementarity. We then analyze how LBF changes the internal allocation between the dense and SID pathways, and finally study how the two re-ranking views behave across datasets. Additional analyses examine projection sharing, re-ranker input representations, and efficiency.

Table 9: Full multi-view SID ablation on BP. Dense item refinement is disabled in all generators. B, T, and G denote behavioral, textual, and graph-based SIDs. The decoder and ranking columns report R@10 (%). Target Cov. is measured over all test instances, and Avg. Union is the mean number of unique candidates after merging the Top-50 decoder lists.

Variant	Views	Beh.	Text	Graph	Target Cov.	Avg. Union	$CVGR^2$-SUM	Re-rank
Single	B	4.3939	–	–	11.2248	50.00	4.3939	4.8580
Single	T	–	4.3992	–	10.9751	50.00	4.3992	4.7748
Single	G	–	–	3.4810	9.7360	50.00	3.4810	3.9684
ITV	B + T	4.5684	4.5331	–	13.7837	68.14	4.6576	5.0438
ITV	B + G	4.4951	–	4.2208	13.8722	69.63	4.5124	4.9745
ITV	T + G	–	4.4525	4.2454	13.8609	70.38	4.5198	4.8933
Independent	B + T + G	4.3939	4.3992	3.4810	16.4118	98.21	4.6037	4.8487
Temporal-only	B + T + G	4.4192	4.3706	3.4984	16.2719	96.43	4.6263	4.9665
ITV	B + T + G	4.5544	4.5164	4.3120	15.3144	81.40	4.6496	5.0604

Independent denotes three separately trained SID generators whose item-level scores are aggregated after decoding. Temporal-only jointly trains the three view streams without cross-view attention. All re-ranking results use the same history-aware backbone and LBF score-integration procedure.

G.1 Full Complementary-View SID Ablation

Table 9 extends the main multi-view analysis in Table 2 with all pairwise SID-view combinations and their re-ranking results. All two-view combinations use the ITV encoder, while the three-view variants compare independent generators, a Temporal-only encoder, and ITV. Dense refinement is disabled throughout so that the effects of SID-view complementarity and cross-view modeling are isolated.

All three pairwise ITV combinations outperform their constituent single-view generators after item-level score aggregation. Moreover, both active decoders improve in each pairwise model, with the graph-based decoder—the weakest single-view predictor—benefiting particularly strongly from cross-view interaction. These results show that the gains are not specific to one privileged view pair: the selected relational geometries provide useful information to one another while retaining separate SID prediction spaces.

The three-view comparison further separates raw candidate diversity from useful predictive complementarity. Independent generators produce the largest candidate union and the highest target coverage, but their aggregated R@10 remains below ITV. ITV substantially reduces the average union size while retaining most of the target coverage and improving both the individual decoders and the aggregated ranking. Thus, preserving complementarity does not mean maximizing disagreement among decoders; useful cross-view interaction can increase agreement while improving the quality of the shared evidence.

The pairwise and three-view results also show that adding more SID views does not monotonically improve simple probability aggregation: the B+T model slightly exceeds the three-view ITV model under $CVGR^2$-SUM . However, the three-view ITV model achieves the strongest re-ranking result, showing that useful evidence from an additional view need not be fully reflected by direct score summation.

The re-ranking column reveals a broader pattern. Across every evaluated generator configuration—from single-view predictors to pairwise and three-view models—history-aware re-ranking improves over the corresponding generator-only ranking. This indicates that direct history-conditioned item relevance provides complementary ranking information beyond SID-based matching across a broad range of generator structures, rather than only for the final ITV configuration.

G.2 Dense-to-SID Pathway Strength

The main dense-refinement experiment in Table 3 shows that LBF both improves the use of dense information and preserves a strong SID-only predictor under the Dense-Off intervention. We further examine whether Early and LBF training also produce different internal allocations between the two

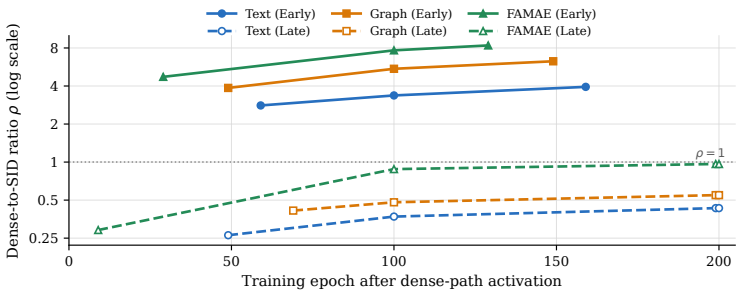

Figure 2: Dense-to-SID pathway ratios under Early and LBF dense-item refinement. Training progress is measured from the point at which the dense pathway is activated, and the vertical axis uses a logarithmic scale.

input pathways. Specifically, we measure

$$\rho_t = \frac{\mathbb{E}_{i,s}\left[\|r_{i,s}^{(t)}\|_2\right]}{\mathbb{E}_{i,v,s}\left[\|e_v^{\mathrm{SID}}(c_{i,v,s})\|_2\right]}. \tag{26}$$

The ratio is computed before the dense residual and SID embedding are added and over non-padding SID tokens. It tracks the relative representation magnitude of the two pathways and is used only as a pathway-strength diagnostic, not as a direct numerical estimator of TOA.

Early and LBF training produce sharply separated pathway-strength trajectories. At the validation-selected checkpoints, the Early ratios are 2.8067, 3.8625, and 4.7263 for the textual, graph-based, and behavioral variants, whereas the corresponding LBF ratios are only 0.2637, 0.4131, and 0.2899. The separation is not a transient consequence of the activation point: the Early ratios eventually increase to 3.9379, 6.2778, and 8.3806, while all LBF ratios remain below one.

The Early models therefore end training with a dense pathway whose representation magnitude exceeds that of the SID pathway, whereas LBF maintains the SID pathway as the dominant base representation. Notably, under Early training the ordering of pathway strength (behavioral $>$ graph $>$ text) mirrors the ordering of Dense-Off degradation in Table 3. This agreement between two different diagnostics is consistent with the view that the more readily exploited dense pathway increasingly carries the prediction when introduced from the beginning. LBF produces a qualitatively different allocation: dense information remains an additive refinement while the independently established SID predictor remains strong.

G.3 Re-Ranker Training Data Sufficiency

Table 10 complements the BP and BPC re-ranking ablations in Table 4 by showing how the two ranking views behave across all ten datasets. The two re-ranking stages for a dataset use the same generated candidate sets. A training instance is retained only when its target appears in the candidate union; validation and test instances whose targets are absent remain misses in the reported metrics.

The relative strength of the two ranking views varies substantially across datasets. On TG, HH, BPC, and BK, the History-only ranker already exceeds the direct generator, whereas the generator remains stronger on the other datasets. This role reversal provides cross-dataset evidence that neither SID-based matching nor history-conditioned item-level relevance uniformly dominates the other. Except for MI and IS, the complete model outperforms both isolated views, consistent with the two ranking functions providing complementary predictive evidence rather than one simply replacing the other. MI and IS are also the two datasets with the fewest effective re-ranker training instances. These results therefore indicate that the ranking views have dataset-dependent relative strength, and learning a robust history-conditioned ranking function is sensitive to sufficient direct supervision.

G.4 Additional Design Analyses

Dense projection sharing. Performance is relatively insensitive to the projection-sharing scheme. The original slot-specific projection gives the best result at 4.7149, but the differences among the four

Table 10: Re-ranker data sufficiency and R@10 across datasets. Retention is the fraction of original training instances whose targets occur in the candidate union.

Dataset	Original Train	Effective Train	Retention	CVGR^2-SUM	History-only	CVGR^2
MI	318,612	105,208	33.02%	6.9530	5.2920	6.8936
VG	489,224	246,269	50.34%	9.9313	9.6020	11.4650
IS	242,912	112,142	46.17%	5.4181	4.1378	5.3805
BP	737,966	191,518	25.95%	4.7149	4.4126	5.1350
ACS	1,116,325	312,537	28.00%	5.3648	5.1042	5.9505
SO	2,108,189	407,564	19.33%	3.5769	3.4582	3.9725
TG	2,360,495	522,707	22.14%	4.5406	5.1732	5.5959
HH	4,475,471	619,306	13.84%	3.3861	3.5156	3.8887
BPC	3,650,871	592,714	16.23%	3.9225	4.0067	4.4748
BK	6,359,670	1,735,124	27.28%	8.6569	9.3360	10.4410

Table 11: Efficiency comparison under the same hardware environment.

Model	Parameters (M)	Training Time (ms/step)	R@10 (%)
ReSID*	4.1499	62.44	4.2894
Independent Generators	5.5588	90.28	4.6037
CVGR^2-SUM	4.2701	70.21	4.7149

variants are small. Notably, making the projection view-specific provides no benefit, supporting our choice to share the continuous residual across SID views while allowing slot-specific adaptation.

Re-ranker input representations. Textual inputs achieve the highest re-ranking R@10 of 5.1350, compared with 4.6376 for graph-based inputs and 4.1895 for FAMAE inputs. Interestingly, the preferred dense representation is stage-dependent: graph representations perform best as generator-side residual refinement (Table 3), whereas textual representations are substantially stronger for direct history-conditioned re-ranking. This suggests that representation utility depends on the predictive role in which it is used, rather than there being a single dense representation that uniformly dominates all pathways.

G.5 Generator-Side Efficiency

Table 11 compares ReSID*, Independent Generators, and CVGR^2-SUM on BP using an NVIDIA L40S GPU, FP32 precision, a batch size of 32. Measurements use 5 warm-up iterations followed by 30 timed iterations over 8 batches. ReSID* uses nine encoder layers and nine decoder layers to approximately match the parameter count of CVGR^2-SUM.

Compared with three independent generators, CVGR^2-SUM uses 23.2% fewer parameters and reduces training time per step by 22.2%, while achieving higher R@10. Compared with the parameter-matched ReSID*, CVGR^2-SUM uses only 2.9% more parameters and 12.4% more training time per step, while improving R@10 by 9.9% relatively. These results show that ITV obtains multi-view gains substantially more efficiently than maintaining three independent generators. This comparison isolates generator-side cost and does not include the additional re-ranking stage.

Table 12: Dense projection sharing on BP.

Projection sharing	R@10 (%)
Slot (original)	4.7149
Shared	4.7095
View	4.6902
Slot $\times$ View	4.6935

Table 13: Re-ranker input representations on BP.

Input representation	R@10 (%)
Text	5.1350
Graph	4.6376
FAMAE	4.1895