VQ-DCR: Vector-Quantized Disentangled Item Cold-Start Recommendation

Li Zou[1], Changhong Li[1], Guohui Li[2,†]
[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
[2]School of Software Engineering, Huazhong University of Science and Technology, Wuhan, China
{li_zou, changhongli1, guohuili}@hust.edu.cn

***Abstract*—Generating collaborative embeddings for cold-start items relying solely on auxiliary content remains a fundamental challenge. Direct alignment methods predict ID embeddings from content features, but sparse supervision and a clustered target space make the regression unstable and biased toward frequent item clusters. RVQ discretizes content into ID-like codes, but its residual hierarchy concentrates most energy in early codebooks and leaves later codebooks with low-energy residuals, so they contribute little during training and inference. Static code summation ignores cross-attribute interactions and collapses compositional signals into a single fixed rule. We propose VQ-DCR to address both issues. We replace RVQ with Finite Scalar Quantization (FSQ) and align the discrete factors with collaborative signals via behavior-aware distillation. An item-side Transformer combiner then contextualizes factor embeddings with self-attention, and a lightweight continuous branch compensates for quantization loss. Across three datasets, VQ-DCR consistently improves cold-start ranking and yields more balanced code energy and usage.**

***Index Terms*—Item Cold-Start Recommendation, Generative Recommendation, Multimodal Recommendation, Vector Quantization.**

I. Introduction

Collaborative Filtering (CF) is a standard paradigm in recommender systems, learning user and item representations from historical interactions [1], [2]. However, CF relies on ID-based embeddings and therefore struggles with the cold-start setting, where new items have little or no interaction history. Existing solutions typically address this by incorporating auxiliary content (e.g., text and images) and learn mappings from content space to the collaborative embedding space, ranging from early attribute-to-feature alignment [3] to recent contrastive objectives [4], [5]. These methods enforce instance-level alignment (MSE/contrastive), but under sparse cold-start supervision they often learn a brittle mapping and generalize poorly to unseen items.

Generative recommendation provides an alternative by discretizing content features into code sequences that act as ID-like tokens in the collaborative space [6]. A representative method adopts residual vector quantization (RVQ): it quantizes the residual of the previous layer and sums codes across layers to reconstruct an item embedding. Crucially, this residual-then-sum design has two limitations in cold-start settings.

A primary limitation is the hierarchical entanglement in RVQ. Its greedy quantization forces early codebooks to capture the bulk of the variance, leaving deeper layers with low-energy residuals that contribute little to the representation. In sparse content scenarios (e.g., short tags or brief descriptions) [7], the dominant variance often corresponds to coarse-grained categories (e.g., "action movie"), while specific attributes (e.g., "Cyberpunk", "Dystopian") are compressed into deeper layers with lower ℓ_2 norms of the selected code vectors. We define the code energy of a codebook as the average ℓ_2 norm of the selected code vectors from that codebook across training instances. Figure 1 shows that this energy decays rapidly in deeper codebooks (i.e., later layers in RVQ), which reduces their contribution during training and inference. As a result, dominant features overshadow fine-grained signals, yielding representations that capture only coarse patterns.

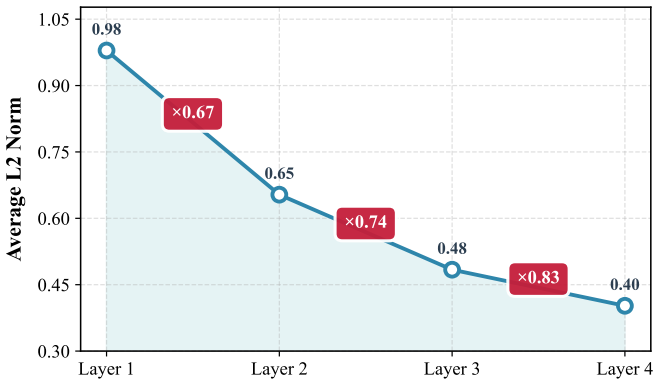


Fig. 1: Energy decay in RVQ on CiteULike with embedding dimension 64. For each codebook (layer), we report the average ℓ_2 norm of the selected code vectors over training instances; deeper codebooks exhibit substantially lower energy.

The second challenge is *static fusion*. Existing generative methods often aggregate discrete codes via summation or shallow MLPs, and these static aggregators provide limited capacity to model interactions among factors. This can lead to generic item representations that do not reflect the compositional nature of items. For instance, the attribute "red" may be decisive for clothing items (e.g., dresses) but much less informative for smartphones. A static fusion mechanism misses dependencies among factors and therefore cannot adapt attribute utility across domains.

To address these challenges, we propose **VQ-DCR**, a framework for **D**isentangled **C**old-start **R**ecommendation. First, to mitigate hierarchical entanglement, we introduce *Finite Scalar*

†Corresponding author. The research work is supported by National Natural Science Foundation of China (No. 62272176), and National Key Research and Development Program of China (No. 2022YFC3802101).

Quantization (FSQ) [8] to cold-start recommendation. Unlike the recursive approximation of RVQ, FSQ quantizes content features along parallel scalar dimensions, encouraging factor-wise disentanglement and reducing dependence on a greedy hierarchy. We further align this latent space with collaborative signals via a *behavior-aware distillation*, transferring knowledge from warm-start items to the quantizer. Second, to overcome static fusion, we propose a *Transformer-based combiner* that treats FSQ codes as a semantic sequence. Using self-attention over the code sequence, the combiner models dependencies among disentangled factors and produces a more expressive item representation than simple summation. Finally, we introduce a *residual-compensated prediction* mechanism. A lightweight continuous branch complements the discrete structural embedding, recovering instance-specific details lost during discretization.

In summary, our main contributions are as follows:

- We identify hierarchical energy decay in RVQ and static fusion of discrete codes as two key bottlenecks in generative cold-start recommendation, and propose VQ-DCR to address them.
- We introduce FSQ as a parameter-efficient and non-hierarchical quantization alternative for recommendation, together with a Transformer-based combiner that models interactions among disentangled codes.
- Experiments on three public datasets show that VQ-DCR outperforms state-of-the-art baselines, supported by analyses on semantic energy distribution and code utilization.

II. Related Work

Item Cold-Start Recommendation. Standard approaches for item cold-start rely on auxiliary content (e.g., text and images) to infer collaborative-space item representations for unseen items. Representative methods learn content-to-CF generators and train them with robust objectives or additional alignment signals, such as DropoutNet [9], Heater [10], and contrastive/alignment-based variants [3]–[5]. GAR [11], GoRec [12] and CGRC [13] incorporate adversarial training or graph reconstruction to improve the quality of generated embeddings. However, sparse supervision leads to unstable mappings and representations that fail to preserve the underlying collaborative structure.

Vector Quantization for Recommendation. Vector quantization (VQ) [14] provides discrete tokens that can serve as ID-like surrogates with combinatorial capacity [15]. Recent models use residual quantization to produce code sequences for retrieval or recommendation [6], [16]–[18]. However, residual quantization may induce hierarchical entanglement with energy decay, where later codebooks contribute less. Moreover, codes are often aggregated with static fusion, which limits the ability to model compositional dependencies among factors. These limitations motivate our non-hierarchical quantization and contextualized code fusion.

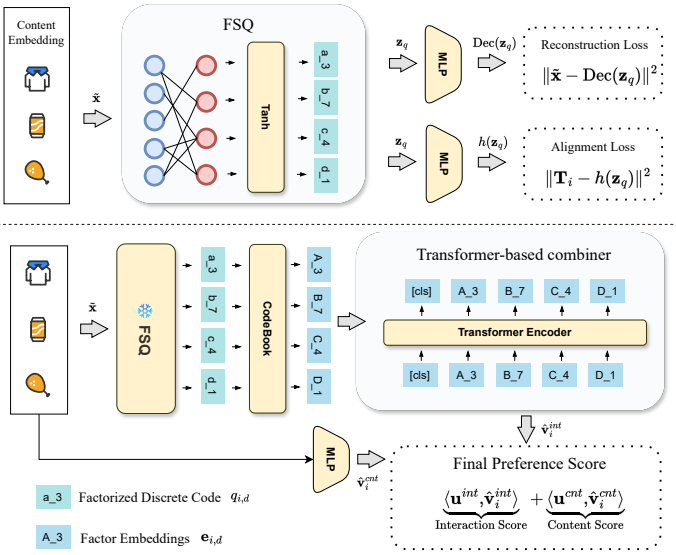


Fig. 2: Overview of VQ-DCR.

III. Method

VQ-DCR addresses item cold-start by generating collaborative-space item representations from multimodal content $\mathbf{x}_i$. As shown in Figure 2, we adopt a two-stage training pipeline. In Stage 1, *Behavior-Aware Disentangled Quantization* learns a Finite Scalar Quantization (FSQ) model supervised by warm-item collaborative signals, producing disentangled discrete indices that are aligned with interaction behavior. In Stage 2, *Contextualized Interaction Generation* freezes the Stage 1 quantizer and trains (1) an item-side Transformer combiner that aggregates the discrete index sequence into a structural embedding $\hat{\mathbf{v}}_i^{int}$, and (2) a lightweight continuous branch that outputs a residual embedding $\hat{\mathbf{v}}_i^{cnt}$. Finally, a fusion scoring function combines user embeddings with $\hat{\mathbf{v}}_i^{int}$ and $\hat{\mathbf{v}}_i^{cnt}$ to produce the preference score for ranking.

A. Behavior-Aware Disentangled Quantization

Stage 1 learns factorized discrete codes from pre-extracted multimodal content features. Let $\mathbf{x}_i \in \mathbb{R}^C$ denote the content feature vector of item i. We aim to produce D disentangled discrete factors, where $D = |\mathbf{L}|$ and $\mathbf{L} = (L_1, \ldots, L_D)$ specifies the number of scalar quantization levels per factor. The goal is to obtain factorized discrete indices that avoid the hierarchical entanglement of RVQ, while aligning the discrete latent space with collaborative signals from warm-start items.

To ensure consistent feature distribution and stable quantization, we first apply feature-wise mean centering. We compute a global mean vector $\boldsymbol{\mu}_{\text{data}}$ on the training set:

$$\boldsymbol{\mu}_{\text{data}} = \frac{1}{|\mathcal{I}_{\text{train}}|} \sum_{j \in \mathcal{I}_{\text{train}}} \mathbf{x}_j \in \mathbb{R}^C, \tag{1}$$

and obtain the centered feature vector $\tilde{\mathbf{x}}_i$:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \boldsymbol{\mu}_{\text{data}}. \tag{2}$$

We then encode content features into a continuous latent space and apply LayerNorm for stability:

$$\mathbf{z}_e = \mathrm{LN}(\mathrm{Enc}(\tilde{\mathbf{x}}_i)) \in \mathbb{R}^{d_z}. \tag{3}$$

Here, $\mathrm{Enc}(\cdot)$ is a multi-layer perceptron that maps $\tilde{\mathbf{x}}_i$ to a latent of dimension d_z, and $\mathrm{LN}(\cdot)$ denotes LayerNorm, which stabilizes quantization by normalizing latent statistics.

We then apply Finite Scalar Quantization (FSQ) to obtain D discrete factors. FSQ first transforms the latent $\mathbf{z}_e$ into a pre-quantization vector $\mathbf{s} \in \mathbb{R}^D$, where s_d is the d-th dimension of $\mathbf{s}$, and independently quantizes each factor into an integer index $q_d \in \{0, \ldots, L_d - 1\}$:

$$\tilde{s}_d = \frac{L_d - 1}{2}\left(\tanh(s_d) + 1\right), \quad q_d = \mathrm{round}(\tilde{s}_d). \tag{4}$$

Here, $\tanh(\cdot)$ bounds s_d to $[-1, 1]$, $\tilde{s}_d$ rescales it to $[0, L_d - 1]$, and $\mathrm{round}(\cdot)$ yields the discrete index. The resulting code is $\mathbf{q} = [q_1, \ldots, q_D] \in \prod_{d=1}^{D}\{0, \ldots, L_d - 1\}$.

We use the Straight-Through Estimator (STE) to back-propagate through the rounding operation, enabling end-to-end training despite the discrete indices. Conceptually, FSQ outputs both (i) a factorized discrete code $\mathbf{q}$ and (ii) its continuous quantized counterpart $\mathbf{z}_q$ for downstream decoding and distillation:

$$(\mathbf{z}_q, \mathbf{q}) = \mathrm{FSQ}_{\mathbf{L}}(\mathbf{z}_e), \tag{5}$$

where $\mathbf{z}_q \in \mathbb{R}^{d_z}$ and $\mathbf{q} \in \prod_{d=1}^{D}\{0, \ldots, L_d - 1\}$. In practice, FSQ maps the factor-wise quantized values back via a learnable output projection Proj_{out} to obtain $\mathbf{z}_q$, making the quantized latent compatible with the decoder and the distillation head.

We train FSQ with an autoencoding reconstruction objective on the content feature vector:

$$\mathcal{L}_{\text{rec}} = \|\tilde{\mathbf{x}}_i - \mathrm{Dec}(\mathbf{z}_q)\|_2^2. \tag{6}$$

To inject collaborative supervision, we distill knowledge from warm-start item embeddings learned by a CF backbone. Let $\mathbf{T}_i \in \mathbb{R}^{d_z}$ be the teacher embedding of item i learned by a pre-trained CF backbone on warm-start items. We minimize the alignment loss:

$$\mathcal{L}_{\text{distill}} = \|\mathbf{T}_i - h(\mathbf{z}_q)\|_2^2. \tag{7}$$

This loss forces the model to focus on behavior-relevant semantics, discarding irrelevant content noise. Therefore, the Stage 1 objective is

$$\mathcal{L}_{\text{stage1}} = \mathcal{L}_{\text{rec}} + \lambda\,\mathcal{L}_{\text{distill}}. \tag{8}$$

B. Contextualized Interaction Generation

We replace static pooling with an item-side Transformer combiner that learns factor-to-factor interactions through self-attention. Static aggregators such as summation/mean pooling (or shallow MLPs) compress factors with fixed rules, which makes it difficult to express *compositional semantics*: the contribution of a fine-grained attribute may depend on other attributes co-occurring in the same item (e.g., "red" is decisive for dresses but less informative for smartphones). To address this, we treat the FSQ factors as a short semantic sequence and use self-attention to model factor-to-factor dependencies in a data-driven way. The combiner runs on items only, so we can precompute item embeddings in batches and reuse them at serving time.

Given the content feature vector $\mathbf{x}_i \in \mathbb{R}^C$, we apply the same feature-wise mean centering as in Stage 1, utilizing the pre-computed global mean $\boldsymbol{\mu}_{\text{data}}$:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \boldsymbol{\mu}_{\text{data}}. \tag{9}$$

Using the same $\boldsymbol{\mu}_{\text{data}}$ for both training and inference ensures consistent normalization and stable index prediction. We then use the frozen Stage 1 quantizer to obtain the disentangled discrete indices:

$$\mathbf{q}_i = [q_{i,1}, \ldots, q_{i,D}] = f_{\text{VQ}}(\tilde{\mathbf{x}}_i), \quad \mathbf{q}_i \in \prod_{d=1}^{D}\{0, \ldots, L_d - 1\}, \tag{10}$$

where $f_{\text{VQ}}(\cdot)$ denotes the pre-trained FSQ-based model (kept fixed in this stage), ensuring the factorization learned in Stage 1 is preserved.

To map each discrete factor to a semantic vector, we maintain D independent embedding tables $\{\mathbf{E}_d\}_{d=1}^{D}$ with $\mathbf{E}_d \in \mathbb{R}^{L_d \times d_z}$:

$$\mathbf{e}_{i,d} = \mathbf{E}_d[q_{i,d}] \in \mathbb{R}^{d_z}. \tag{11}$$

Stacking all factor embeddings yields an item attribute sequence:

$$\mathbf{S}_i = [\mathbf{e}_{i,1}, \mathbf{e}_{i,2}, \ldots, \mathbf{e}_{i,D}] \in \mathbb{R}^{D \times d_z}. \tag{12}$$

Although the discrete code space grows exponentially as $|\mathcal{Q}| = \prod_{d=1}^{D} L_d$ (and $|\mathcal{Q}| = L^D$ when $L_d = L$), the number of learnable code vectors in the embedding tables grows only linearly: $N_{\text{tab}} = \sum_{d=1}^{D} L_d$. This is a key advantage of FSQ in cold-start recommendation: the representation capacity can be enlarged effectively by increasing D while keeping each L small, leading to a compact set of N_{tab} code vectors to learn. In contrast, due to the hierarchical entanglement and energy decay in RVQ, scaling the number of residual codebooks is often less effective; practitioners tend to enlarge L to increase capacity, which increases the codebook size and makes training harder under sparse cold-start data.

FSQ factors have no natural order, so we drop positional embeddings and make the `[CLS]` aggregation insensitive to the ordering of factor tokens. We prepend a learnable `[CLS]` token $\mathbf{c} \in \mathbb{R}^{d_z}$ and feed the sequence into a K-layer Transformer encoder:

$$\mathbf{X}_i = [\mathbf{c}; \mathbf{S}_i] \in \mathbb{R}^{(D+1) \times d_z}, \tag{13}$$

$$\mathbf{H}_i = \mathrm{TransEnc}(\mathbf{X}_i) \in \mathbb{R}^{(D+1) \times d_z}, \tag{14}$$

where $\mathrm{TransEnc}(\cdot)$ is a standard Transformer encoder stack (multi-head self-attention with residual connections and position-wise feed-forward blocks). Self-attention allows each factor embedding to be *contextualized* by other factors, thereby modeling domain-dependent attribute utility and alleviating the static fusion bottleneck.

We use the contextualized [CLS] output as the aggregated interaction representation:

$$\mathbf{h}_i^{cls} = \mathbf{H}_i[0] \in \mathbb{R}^{d_z}, \quad \hat{\mathbf{v}}_i^{int} = \mathbf{W}_o\, \mathbf{h}_i^{cls} + \mathbf{b}_o \in \mathbb{R}^{d_z}. \quad (15)$$

We choose [CLS] for two reasons. First, it provides a learnable global aggregator, avoiding hand-crafted pooling rules and allowing the model to decide how to summarize factors. Second, [CLS] interacts with all factor tokens through self-attention, making it a natural bottleneck to capture global compositional semantics in a fixed-dimensional vector that interfaces cleanly with downstream ranking objectives.

Efficiency and parallelization. The combiner is item-only: it depends solely on $\mathbf{x}_i$ (and the frozen quantizer) and is independent of user context. This enables efficient large-batch parallel inference over items, and the resulting $\hat{\mathbf{v}}_i^{int}$ can be precomputed and cached for fast online recommendation. Moreover, the sequence length is only $D + 1$, keeping the Transformer overhead lightweight in practice.

C. Residual-Compensated Prediction and Optimization

While the Transformer-based combiner produces a robust structural item embedding $\hat{\mathbf{v}}_i^{int}$ from discretized FSQ codes, quantization is inherently lossy and may discard instance-specific details (e.g., subtle visual/textual cues). To compensate for this information loss, we introduce a lightweight continuous branch and perform dual-stream scoring. We map the original continuous content feature $\mathbf{x}_i \in \mathbb{R}^C$ to a detail embedding:

$$\hat{\mathbf{v}}_i^{cnt} = \mathrm{MLP}_{cnt}(\mathbf{x}_i) \in \mathbb{R}^{d_z}. \quad (16)$$

The interaction-side user embedding $\mathbf{u}_u^{int} \in \mathbb{R}^{d_z}$ is obtained from the same pre-trained CF model as in Stage 1 (i.e., the teacher that provides collaborative embeddings for warm items), and thus lies in the collaborative space by construction. For the content-side user embedding, we aggregate the user's historical interacted items with mean pooling:

$$\mathbf{u}_u^{cnt} = \frac{1}{|\mathcal{I}_u|} \sum_{k \in \mathcal{I}_u} \hat{\mathbf{v}}_k^{cnt}, \quad (17)$$

where $\mathcal{I}_u$ denotes the set of items interacted by user u.

We compute an interaction score and a content score:

$$s_{ui}^{int} = \langle \mathbf{u}_u^{int}, \hat{\mathbf{v}}_i^{int} \rangle, \quad s_{ui}^{cnt} = \langle \mathbf{u}_u^{cnt}, \hat{\mathbf{v}}_i^{cnt} \rangle. \quad (18)$$

Under the mean fusion option, the final preference score is

$$\hat{y}_{ui} = \frac{1}{2} \left(s_{ui}^{int} + s_{ui}^{cnt} \right). \quad (19)$$

This design allows the discrete structural pathway to focus on stable semantics, while the continuous pathway provides residual signals that are hard to preserve through quantization.

We train the model by optimizing the fused score $\hat{y}_{ui}$ with the Bayesian Personalized Ranking (BPR) loss:

$$\mathcal{L} = \sum_{(u,i,j) \in \mathcal{O}} -\log \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda_{reg} \|\Theta\|^2, \quad (20)$$

where $\mathcal{O}$ is the set of sampled triplets, $\sigma(\cdot)$ is the sigmoid function, and Θ denotes trainable parameters.

TABLE I: Statistics of the datasets. "Inter." denotes interactions and "Dims" denotes feature dimensions.

Dataset	General Statistics				Cold Setting		Sparsity
	Users	Items	Inter.	Dims	Items	Inter.	
Movielens	6,040	3,706	1,000,209	206	742	192,651	95.53%
CiteULike	5,551	16,980	204,986	300	3,396	39,960	99.78%
XING	106,881	20,519	3,856,580	2,738	4,104	830,634	99.82%

IV. Experiment

Datasets and Inductive Protocol. We conduct our evaluation on three benchmark datasets, including MovieLens, CiteULike and XING, which cover diverse domains such as movie ratings, article bookmarks, and job applications. The detailed statistics are provided in Table I. To simulate a zero-shot cold-start scenario, we adopt a strict inductive splitting strategy where the item set is disjointly partitioned into 80% warm items and 20% cold items. During training, the model is exposed exclusively to warm-item interactions from the warm *training* split. Specifically, interactions on warm items are split into training (80%), validation (10%), and testing (10%) subsets for model selection and evaluation. Interactions involving cold items are strictly withheld from training and are used only for evaluation, split equally into validation and testing sets. This protocol ensures that the model must generalize to unseen items based purely on content features, preventing any collaborative signal leakage.

Evaluation Metrics. We evaluate model performance using a full-ranking protocol, where each ground-truth interaction is ranked against all items in the specified candidate pool that the user has not interacted with in the training data. We report Recall@20 and NDCG@20 as the primary metrics. Specifically, we analyze performance across three distinct scenarios: *Cold*, which ranks cold items among cold candidates to directly assess generative quality; *Warm*, which ranks warm items to ensure the generative module does not degrade collaborative knowledge[1]; and *All*, which ranks targets within the entire corpus to evaluate the model's robustness in hybrid scenarios.

Implementation Details. All models are implemented in the PyTorch framework for fair comparison. We utilize a pre-trained NCL model trained on the warm training split as the CF backbone to provide teacher ID embeddings and user embeddings $\mathbf{u}_u^{int}$. The latent dimension is fixed to $d_z = 64$, and the batch size is set to 2048 for all experiments. We optimize with AdamW, and apply early stopping if validation NDCG@20 does not improve for 10 consecutive epochs. For VQ-DCR, we use an item-side Transformer combiner with $K = 1$ encoder layer and 4 attention heads. To facilitate hyperparameter search, we set all FSQ levels to the same value $L_d = L$ for all factors, and tune $L \in \{4, 6, 8, 10\}$ and the number of factors $D \in \{5, 6, 7, 8\}$ on the validation set. The distillation weight λ is tuned via grid search in $\{0.1, \ldots, 5.0\}$.

[1]For warm-item evaluation, some cold-start baselines directly use the pre-trained CF ID embeddings as warm item representations, which may yield identical warm-item results across methods. In contrast, VQ-DCR still uses the generated item representation for warm items to explicitly evaluate the impact of generation on warm retrieval.

TABLE II: Overall performance comparison. The best result in each row is **bolded** and the runner-up is underlined.

Dataset	Metric	Direct Generation			Robust Training			Auxiliary Objectives				VQ		
		DUIF [19]	DeepMusic [20]	CVAR [21]	DropoutNet [9]	AMR [22]	Heater [10]	CLCRec [4]	GAR [11]	ALDI [23]	CGRC [13]	SaviorRec [6]	**VQ-DCR**	Improv.
MovieLens	R_{all}	0.0829	0.1368	0.0936	0.0586	0.0745	0.1313	0.1310	0.1393	0.1423	0.1088	0.1465	**0.1614**	+10.17%
	N_{all}	0.1180	0.1643	0.1363	0.0767	0.1153	0.1805	0.1700	0.1904	0.1926	0.1568	0.1982	**0.2107**	+6.31%
	R_{cold}	0.2154	0.1656	0.2218	0.1627	0.1274	0.1206	0.1622	0.2327	0.2150	0.2317	0.2218	**0.2594**	+11.47%
	N_{cold}	0.1990	0.1536	0.2067	0.1476	0.1217	0.1128	0.1034	0.2210	0.2045	0.2111	0.2102	**0.2421**	+9.55%
	R_{warm}	0.1335	0.3075	0.1692	0.1046	0.1625	0.2902	**0.3100**	0.2351	0.3075	0.2343	0.3075	0.3037	-2.03%
	N_{warm}	0.1181	0.2566	0.1536	0.0843	0.1460	0.2418	**0.2599**	0.2042	0.2566	0.2041	0.2566	0.2512	-3.35%
CiteUlike	R_{all}	0.0962	0.1432	0.0880	0.0542	0.0558	0.1058	0.1663	0.1812	0.1825	0.1408	0.1781	**0.1925**	+5.48%
	N_{all}	0.0644	0.1081	0.0614	0.0363	0.0371	0.0840	0.1283	0.1278	0.1281	0.0995	0.1252	**0.1331**	+3.74%
	R_{cold}	0.2640	0.2217	0.2576	0.1948	0.1935	0.1674	0.2813	0.2843	0.2539	0.3010	0.2468	**0.3209**	+6.61%
	N_{cold}	0.1558	0.1254	0.1525	0.1102	0.1127	0.1005	0.1754	0.1669	0.1473	0.1781	0.1425	**0.1865**	+4.72%
	R_{warm}	0.1298	**0.3074**	0.1430	0.0991	0.0872	0.2649	0.2316	0.2439	**0.3074**	0.2137	**0.3074**	0.3041	-1.07%
	N_{warm}	0.0628	**0.1752**	0.0733	0.0488	0.0418	0.1472	0.1201	0.1322	**0.1752**	0.1120	**0.1752**	0.1715	-2.11%
XING	R_{all}	0.1770	0.2267	0.1760	0.0803	0.0974	0.2076	0.2097	0.2420	0.2540	0.2005	0.2494	**0.2681**	+5.55%
	N_{all}	0.1374	0.1682	0.1411	0.0589	0.0716	0.1769	0.1749	0.1863	0.1952	0.1657	0.1915	**0.2051**	+5.07%
	R_{cold}	0.2930	0.2327	0.2867	0.1971	0.1836	0.1623	0.2234	0.2992	0.2883	0.3067	0.2818	**0.3371**	+9.91%
	N_{cold}	0.1973	0.1478	0.1921	0.1259	0.1067	0.1228	0.1545	0.1898	0.1927	0.2071	0.1873	**0.2255**	+8.89%
	R_{warm}	0.3552	**0.4955**	0.3462	0.1423	0.2511	0.4795	0.4921	0.4321	**0.4955**	0.4645	**0.4955**	0.4861	-1.90%
	N_{warm}	0.2111	**0.3064**	0.2132	0.0793	0.1322	0.2953	0.3023	0.2599	**0.3064**	0.2859	**0.3064**	0.3002	-2.02%

A. Overall Performance Comparison

Table II presents the performance of VQ-DCR and competing methods on three datasets. VQ-DCR achieves the best performance in the Cold setting across all datasets, with relative improvements of 11.47%, 6.61%, and 9.91% in Recall@20 on MovieLens, CiteUlike, and XING, respectively. These gains suggest that FSQ with behavior-aware distillation better preserves behavior-relevant structure when generating embeddings from content.

We also observe that SaviorRec is competitive on MovieLens but degrades on CiteUlike and XING, falling behind CGRC/ALDI in *Cold* and *All*. An explanation is that RVQ-style hierarchical residual quantization can be sensitive under sparse interaction data due to hierarchical entanglement and energy decay, making it harder to fully exploit deeper codebooks. In contrast, VQ-DCR uses FSQ to quantize along parallel scalar factors and aggregates factor embeddings with an item-side Transformer encoder. This combiner leverages self-attention to capture compositional dependencies among factors, which is more expressive than static fusion methods (e.g., summation) when item attributes exhibit complex correlations.

In the Warm setting, VQ-DCR performs on par with or slightly below the strongest baselines (e.g., within 2% gap). This minor gap is expected as baselines like DeepMusic and ALDI directly leverage pre-trained ID embeddings. The drop is minimal, because the residual compensation branch can help preserve instance-level details and reduce the performance gap on warm items.

B. Ablation Study

To quantify the contribution of each component in VQ-DCR, we conduct ablation studies on MovieLens and CiteULike, as summarized in Table III.

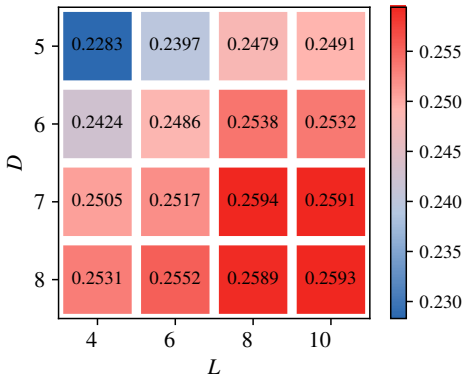
TABLE III: Ablation study on key components of VQ-DCR.

Variant	MovieLens		CiteULike	
	R@20	N@20	R@20	N@20
VQ-DCR (Full)	**0.2594**	**0.2421**	**0.3209**	**0.1865**
Impact of Quantization				
w/ Standard VQ	0.2446	0.2299	0.3088	0.1741
w/ LFQ	0.2518	0.2376	0.3194	0.1818
Impact of Fusion Strategy				
w/ Mean Pooling (No Attn)	0.2257	0.2111	0.2955	0.1704
Impact of Architecture				
w/o Behavior Alignment	0.2459	0.2289	0.3136	0.1792
w/o Residual MLP	0.2343	0.2183	0.3050	0.1733
Only Residual MLP	0.2138	0.1959	0.2764	0.1637

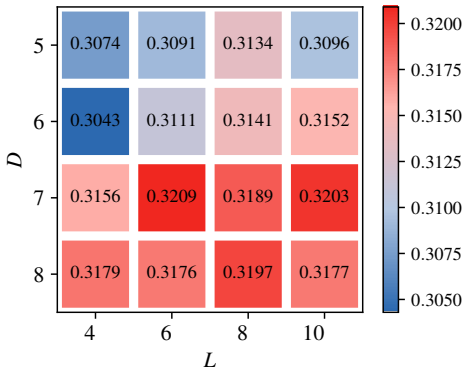
We replace FSQ with Standard VQ [14] and LFQ [24] while keeping the rest of the pipeline unchanged. Standard VQ leads to a clear degradation (e.g., Recall@20 drops from 0.2594 to 0.2446 on MovieLens), suggesting that standard codebook-based VQ is harder to optimize under sparse cold-start supervision. LFQ performs closer to FSQ but remains consistently worse, suggesting that FSQ is a better fit for learning factorized discrete codes aligned with recommendation signals.

Replacing the Transformer-based combiner with mean pooling over factor embeddings substantially degrades performance on both datasets. This validates our hypothesis regarding static fusion: simply aggregating disentangled factors cannot capture their compositional dependencies, whereas self-attention provides a more expressive mechanism for modeling factor interactions.

Removing behavior alignment (i.e., training Stage 1 without the distillation loss) consistently hurts ranking quality, showing that collaborative distillation is important for aligning discrete codes with interaction semantics. When ablating a stream (e.g.,



(a) Movielens



(b) CiteUlike

Fig. 3: Performance with quantization parameters.

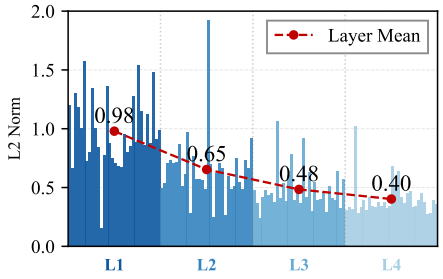
removing the residual or structural pathway), we score with the remaining stream only. Removing the residual MLP branch also causes a noticeable drop, indicating that the continuous branch complements the discrete structural representation by recovering instance-level details that are difficult to preserve through quantization. Finally, the variant that removes the structural stream performs the worst, highlighting that the discrete structural bottleneck is essential for generalization in cold-start recommendation.

C. Parameter Sensitivity Analysis

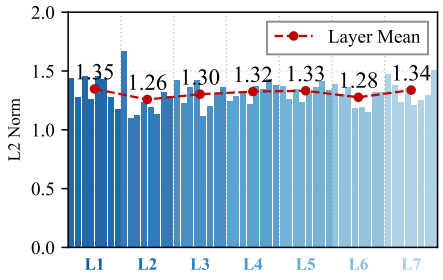
To study the effect of FSQ capacity, we vary two key hyperparameters: the number of quantization factors D and the number of levels per factor L. For simplicity, we set $L_d = L$ for all factors and conduct grid search with $L \in \{4, 6, 8, 10\}$ and $D \in \{5, 6, 7, 8\}$. Figure 3 reports Recall@20 on MovieLens and CiteULike.

Overall, increasing D and/or L tends to improve performance, consistent with the increased discrete capacity $|\mathcal{Q}| = L^D$. On MovieLens, we observe a monotonic improvement as capacity increases, and the performance saturates when $D \geq 7$ and $L \geq 8$, suggesting that sufficient discrete capacity is beneficial for capturing item semantics.

On the sparser CiteULike dataset, increasing D is generally more reliable than increasing L. In particular, larger L does not always translate to better performance, and the best results



(a) Residual VQ



(b) FSQ

Fig. 4: Comparison of Semantic Energy Distribution.

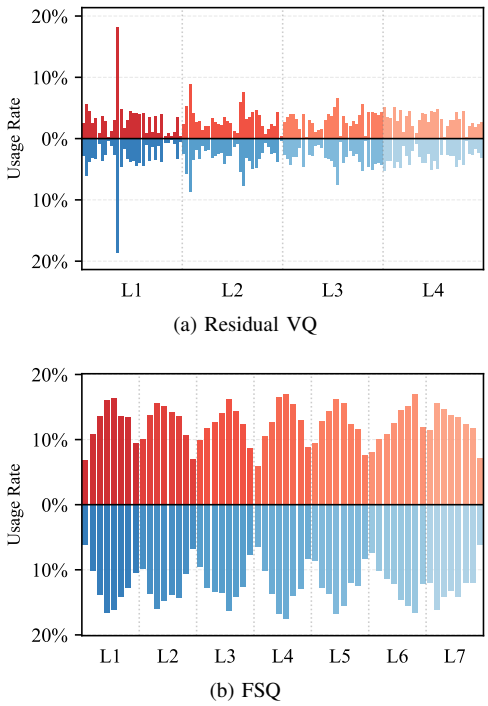
are achieved with moderate L and moderate-to-large D (e.g., $D = 7$, $L = 6$). This indicates that under sparse supervision, allocating capacity to more factors can be more effective than using overly fine-grained levels per factor. Following this observation, we use the best-performing configuration on the validation set as the default setting for each dataset.

D. Visualization Analysis

Semantic Energy Distribution. To compare the inductive bias of different quantization mechanisms, we analyze the average ℓ_2 norm of the code embeddings used to form the item representation. For RVQ, we compute the norm of the selected code vectors from each codebook layer. For FSQ in VQ-DCR, we compute the norm of factor embeddings retrieved from the D embedding tables, i.e., $\mathbf{e}_{i,d} = \mathbf{E}_d[q_{i,d}]$, and report the average ℓ_2 norm per factor d over items. We use this norm as a proxy for the magnitude of each component's contribution to the final representation. To make a fair comparison in terms of the discrete code space size, we choose an FSQ configuration whose theoretical state space size L^D is on the same order as the RVQ configuration used in SaviorRec (e.g., 32^4 vs. 8^7).

As shown in Figure 4(a), Residual VQ (used in SaviorRec) exhibits a clear energy decay across codebooks. The mean norm decreases from 0.98 in the first codebook to 0.40 in the fourth, indicating that later codebooks contribute substantially less than early ones. This observation is consistent with our discussion on hierarchical entanglement: the greedy residual-then-sum structure tends to allocate most representational magnitude to early codebooks, leaving deeper codebooks with low-energy residual signals that are more likely to be under-utilized.

In contrast, FSQ in VQ-DCR shows a markedly more balanced energy distribution across factors (Figure 4(b)). The factor norms remain stable (ranging from 1.26 to 1.35) without



(a) Residual VQ

(b) FSQ

Fig. 5: Codebook Usage Visualization.

a progressive decay, suggesting that FSQ mitigates hierarchical coupling and maintains comparable contribution magnitudes across factors. This balanced factorization is beneficial for the subsequent Transformer-based combiner, which relies on multiple factors to model compositional dependencies among item attributes.

Generalization to Cold Items. A key requirement in cold-start recommendation is to generalize from warm (training) items to unseen cold (test) items. We visualize code usage frequency for warm and cold items in Figure 5. For RVQ, we report the index histogram of each codebook layer; for FSQ, we report the index histogram of each factor $q_{i,d}$. Given the random split, the feature distributions of warm and cold items are expected to be similar. Indeed, both models exhibit mirrored patterns between the two sets. However, these distributions differ in representational entropy.

Residual VQ (SaviorRec) exhibits a highly skewed utilization pattern: a small subset of codes are selected frequently while many codes are rarely used. This low diversity suggests that the discrete representation is dominated by a few frequent patterns, which limits the ability to represent diverse cold-start items. In contrast, VQ-DCR shows more balanced utilization across codes, and the warm/cold histograms are closely matched across factors. The high consistency between warm and cold histograms, confirms that FSQ effectively utilizes the space to capture diverse semantic nuances for unseen items.

E. Conclusion

This paper introduces VQ-DCR to address the bottlenecks of hierarchical energy decay and static fusion inherent in Residual VQ methods. Our framework constructs robust collaborative representations by combining behavior-aware FSQ, a contextual Transformer combiner, and a residual compensation mechanism. Across the MovieLens, CiteULike, and XING datasets, VQ-DCR demonstrates consistent improvements over state-of-the-art baselines on cold items, while maintaining competitive performance on warm items.

REFERENCES

[1] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *SIGIR*, 2020, pp. 639–648.

[2] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, "Neural graph collaborative filtering," in *SIGIR*. ACM, 2019, pp. 165–174.

[3] O. Barkan, N. Koenigstein, E. Yogev, and O. Katz, "CB2CF: a neural multiview content-to-collaborative filtering model for completely cold item recommendations," in *RecSys*, 2019, pp. 228–236.

[4] Y. Wei, X. Wang, Q. Li, L. Nie, Y. Li, X. Li, and T. Chua, "Contrastive learning for cold-start recommendation," in *MM*, 2021, pp. 5382–5390.

[5] Z. Zhou, L. Zhang, and N. Yang, "Contrastive collaborative filtering for cold-start item recommendation," in *WWW*, 2023, pp. 928–937.

[6] Y. Yao, Z. Li, S. Xiao, B. Du, J. Zhu, J. Zheng, X. Kong, and Y. Jiang, "Saviorrec: Semantic-behavior alignment for cold-start recommendation," *CoRR*, vol. abs/2508.01375, 2025.

[7] B. Tan, W. Ge, Y. Wang, X. Liu, J. Burtoft, H. Fan, and H. Wang, "PCR-CA: parallel codebook representations with contrastive alignment for multiple-category app recommendation," *CoRR*, vol. 2508.18166, 2025.

[8] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen, "Finite scalar quantization: VQ-VAE made simple," in *ICLR*, 2024.

[9] M. Volkovs, G. Yu, and T. Poutanen, "Dropoutnet: Addressing cold start in recommender systems," in *NeurIPS*, 2017, pp. 4957–4966.

[10] Z. Zhu, S. Sefati, P. Saadatpanah, and J. Caverlee, "Recommendation for new users and new items via randomized training and mixture-of-experts transformation," in *SIGIR*, 2020, pp. 1121–1130.

[11] H. Chen, Z. Wang, F. Huang, X. Huang, Y. Xu, Y. Lin, P. He, and Z. Li, "Generative adversarial framework for cold-start item recommendation," in *SIGIR*, 2022, pp. 2565–2571.

[12] H. Bai, M. Hou, L. Wu, Y. Yang, K. Zhang, R. Hong, and M. Wang, "Gorec: A generative cold-start recommendation framework," in *MM*, 2023, pp. 1004–1012.

[13] J. Kim, E. Kim, K. Yeo, Y. Jeon, C. Kim, S. Lee, and J. Lee, "Content-based graph reconstruction for cold-start item recommendation," in *SIGIR*, 2024, pp. 1263–1273.

[14] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *NeurIPS*, 2017, pp. 6306–6315.

[15] X. Luo, J. Cao, T. Sun, J. Yu, R. Huang *et al.*, "QARM: quantitative alignment multi-modal recommendation at kuaishou," in *CIKM*. ACM, 2025, pp. 5915–5922.

[16] S. Rajput, N. Mehta, A. Singh, R. H. Keshavan, T. Vu, L. Heldt, L. Hong, Y. Tay, V. Q. Tran, J. Samost, M. Kula, E. H. Chi, and M. Sathiamoorthy, "Recommender systems with generative retrieval," in *NeurIPS*, 2023.

[17] B. Zheng, Y. Hou, H. Lu, Y. Chen, W. X. Zhao, M. Chen, and J. Wen, "Adapting large language models by integrating collaborative semantics for recommendation," in *ICDE*, 2024, pp. 1435–1448.

[18] W. Ye, M. Sun, S. Shi, P. Wang, W. Wu, and P. Jiang, "DAS: dual-aligned semantic ids empowered industrial recommender system," in *CIKM*, 2025, pp. 6217–6224.

[19] X. Geng, H. Zhang, J. Bian, and T. Chua, "Learning image and user features for recommendation in social networks," in *ICCV*, 2015, pp. 4274–4282.

[20] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *NeurIPS*, 2013.

[21] X. Zhao, Y. Ren, Y. Du, S. Zhang, and N. Wang, "Improving item cold-start recommendation via model-agnostic conditional variational autoencoder," in *SIGIR*, 2022, pp. 2595–2600.

[22] J. Tang, X. Du, X. He, F. Yuan, Q. Tian, and T. Chua, "Adversarial training towards robust multimedia recommender system," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 855–867, 2020.

[23] F. Huang, Z. Wang, X. Huang, Y. Qian, Z. Li, and H. Chen, "Aligning distillation for cold-start item recommendation," in *SIGIR*, 2023.

[24] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. Minnen, Y. Cheng, A. Gupta, X. Gu, A. G. Hauptmann, B. Gong, M. Yang, I. Essa, D. A. Ross, and L. Jiang, "Language model beats diffusion - tokenizer is key to visual generation," in *ICLR*, 2024.