CONTROLREC: EDITABLE INTENT MODELING FOR CONTROLLABLE GENERATIVE RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative recommender systems directly generate items from behavioral history, but most leave user intent under-specified and difficult to revise. Sequential recommenders primarily infer preferences from past behavior, while conversational recommenders elicit immediate needs but often make limited use of long-term behavioral preferences. We propose ControlRec, a unified framework for generative recommendation with implicit and explicit intent guidance. Without dialogue, it infers personalized intents by softly assigning historical interactions to shared preference prototypes; with dialogue, it updates these intents from natural-language feedback while preserving behavioral personalization. ControlRec comprises three core designs: (1) A semantic alignment adapter bridges natural language and the semantic IDs (SIDs) space, enabling expressed preferences to guide item generation. (2) A multi-intent condition encoder constructs personalized intent conditions from historical interactions and shared preference prototypes, and updates an editable intent state from natural-language feedback when dialogue is available, preserving behavioral personalization. (3) An intent-factorized diffusion decoder guides generation via parallel intent-specific denoising, preserving distinct concurrent preferences while enforcing shared negative constraints. Experiments on three Amazon domains and evaluations derived from ReDial demonstrate competitive next-item recommendation performance and effective intent refinement. On a challenging target-recovery subset, successive feedback progressively improves target recovery up to 3.22 times, and on ReDial the model achieves strong positive, negative, and multi-intent control. Furthermore, ControlRec supports freely expressed natural-language requests, providing a promising interface for controllable generative recommendation. Code will be released soon, and an interactive online demo is available[1].

1 INTRODUCTION

Modern generative recommender systems have moved beyond conventional graph-based and sequential collaborative filtering (He et al., 2020; Kang & McAuley, 2018; Sun et al., 2019) to directly generate item identities, often represented as semantic IDs (SIDs) (Rajput et al., 2023; Deng et al., 2025). Yet most still map behavioral history directly to the next item without explicitly representing the user's current intent. History may capture multiple preferences, but it does not indicate which preference should guide the current intent or reflect newly expressed needs and constraints. Users may instead want something different, add new constraints, or change their mind during real-world interaction, such as asking for a cheaper option or switching from one category to another.

Existing approaches capture only parts of this process. First, multi-interest sequential recommenders infer multiple latent preferences from behavioral history (Li et al., 2019; Cen et al., 2020; Tan et al., 2021; Chai et al., 2022; Li et al., 2026), but their activation and relative contribution are typically controlled by internal routing or aggregation rather than explicit user edits. Second, conversational recommenders use natural-language interaction to elicit users' current intent (Gao et al., 2021; Wei et al., 2025). However, many CRSs summarize the dialogue into a shared preference representation or assume a single recommendation target. This makes it difficult to preserve multiple concurrent intents during recommendation (Wang et al., 2022a; Zhang et al., 2022). In addition, long-term

[1]Online demo: https://control-demo-mimphzrsxk8bygcajafwxm.streamlit.app

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.



Figure 1: Direct history-to-item generation versus our intent-mediated generation paradigm.

interaction history is not always maintained as a personalized state that can be explicitly updated through conversational feedback (Zhou et al., 2020b; Li et al., 2022). Motivated by these observations, we formulate generative recommendation as an intent-mediated process in which behavioral history provides a personalized prior over plausible intents, natural-language feedback updates an explicit intent state, and multiple active intents are preserved as separate conditions during item generation. This formulation turns user intent from a hidden latent factor into an editable intermediate state that connects long-term behavioral preferences with immediate user needs.

Diffusion models are amenable to conditional control through iterative denoising guided by textual or structural signals, as demonstrated in image and video generation (Ho et al., 2020; Ho & Salimans, 2021; Ho et al., 2022; Zhang et al., 2023). Applied to recommendation, masked diffusion can revise multiple SID positions jointly without committing early to a single autoregressive prefix. Existing diffusion-based recommenders, however, mainly focus on preference reconstruction or behavior-conditioned next-item generation (Liu et al., 2026; Qu & Nobuhara, 2025), without supporting explicit intent control through natural-language feedback. Extending them to editable intent-mediated recommendation requires addressing three challenges. **(1) Language–SID grounding:** natural-language feedback and discrete SIDs have no direct semantic correspondence, making it difficult to translate feedback into item-generation conditions. **(2) Personalized intent updating:** using feedback to change active intents and their relative importance may overwrite personalization derived from behavioral history. **(3) Multi-intent interference:** combining several active user intents into a single condition may blur or collapse them during denoising.

To address these challenges, we propose CONTROLREC, an intent-controlled generative recommender built on masked discrete diffusion. First, **Semantic Alignment Adapter** learns bidirectional alignment between natural language and item SIDs, allowing user feedback to condition the discrete item space while retaining the original history-only recommendation path. Second, the **Multi-Intent Condition Encoder** initializes an explicit intent state from behavioral history and maintains it across turns. A frozen large language model (LLM) parses free-form feedback into structured edits, allowing individual preferences to be revised while retaining other active requirements. The updated state provides positive-intent conditions, their relative weights, and shared negative guidance for personalized generation. Finally, the **Intent-Guided Diffusion Decoder** injects intent conditions into the denoising hidden states, enabling direct intervention in the generation process. This allows item generation to adapt to users' evolving needs and constraints while retaining history-based personalization. Together, these components connect language, editable intent, and personalized generation into a unified recommendation process. Experiments show that CONTROLREC remains competitive with strong sequential and generative recommenders across three Amazon domains. On hard cases missed by history-only recommendation, feedback increases Recall@10 and NDCG@10 to 3.22× and 3.49× their history-only values, while CONTROLREC also supports positive, negative, and multi-intent control on ReDial. **Our main contributions are summarized as follows:**

- We first introduce a unified intent-mediated formulation of generative recommendation, in which behavioral history provides a personalized prior, language feedback updates an explicit and editable intent state, supporting both history-only and language-guided recommendation within a general framework.
- We propose ControlRec, which instantiates the formulation through language SID semantic alignment, editable multi-intent modeling, and intent-guided diffusion generation. These designs enable progressive refinement of user intent and coordinated control of positive preferences, negative constraints, and concurrent needs while retaining personalization.
- Experiments on three Amazon domains and ReDial demonstrate competitive dialogue-free recommendation, multi-turn recovery of hard cases, and improved multi-intent coverage from intent-factorized decoding over joint conditioning.

2 Related Work

Generative Recommendation Systems. Recent recommendation systems have moved beyond fixed-catalog ranking toward generative recommendation, representing items as sequences of SIDs and generating them autoregressively (Wu et al., 2025; Rajput et al., 2023), or casting recommendation as a unified language-processing task with personalized prompts (Geng et al., 2022). This paradigm typically involves two stages: item tokenization and sequence generation (Guo et al., 2026; Ye et al., 2026). For tokenization, FORGE (Fu et al., 2026) studies SID construction and proposes training-free metrics for SID quality. For generation, OnePiece (Dai et al., 2026) and OneReason (Team et al., 2026) introduce structured reasoning for industrial recommendation. Conversational recommender systems further incorporate natural-language feedback into recommendation, with recent methods exploring parameter-efficient language modeling and richer conversational context (Wang et al., 2022b; Ravaut et al., 2024; Wei et al., 2025). However, most such approaches condition generation on the dialogue context without explicitly modeling how each turn modifies an existing personalized preference state. Our work instead uses generative item representations to support explicit preference edits over multiple turns.

Diffusion for Recommendation. Diffusion models provide an alternative generative framework based on iterative denoising (Ho et al., 2020), where conditional guidance enables flexible control over the generation process. This makes diffusion a natural fit for recommendation, where user preferences can serve as generation conditions. DiffRec (Wang et al., 2023) applies diffusion to interaction vectors, while ContRec (Qu et al., 2026) uses continuous user and item tokens and conditions diffusion on LLM reasoning. InDiRec (Qu & Nobuhara, 2025) uses clustered user preferences to guide diffusion-based generation. LLaDA-Rec further introduces masked discrete diffusion for parallel SID generation (Shi et al., 2025; Liu et al., 2026). Existing approaches mainly focus on preference reconstruction or history-conditioned next-item generation without explicit signal control, we first achieve intent-guided parallel diffusion generation under conversational control, bridging the semantic gap between natural language and SIDs.

3 Preliminaries

3.1 Problem Formulation

Generative recommendation. Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items, respectively. For a user $u \in \mathcal{U}$, the historical interaction sequence is $\mathcal{H}_u = (i_{u,1}, \ldots, i_{u,L_u})$, where $i_{u,\ell} \in \mathcal{I}$, and the ground-truth next item is $i_u^+ \in \mathcal{I}$. A standard generative recommender maps the behavioral history directly to the identity of the next item:

$$\hat{i}_u^+ = \arg\max_{i \in \mathcal{I}} p_\theta(i \mid \mathcal{H}_u)\,, \tag{1}$$

where an item is generated as a sequence of SID codes (Shi et al., 2025). This history-only formulation provides a personalized behavioral prior, but it does not expose which preference should control the current recommendation.

Editable-intent generative recommendation. At interaction turn t, free-form feedback determines an editable intent state S_u^t that complements the behavioral history. Extending Eq. 1, editable-intent recommendation predicts

$$\hat{i}_u^{+,t} = \arg\max_{i \in \mathcal{I}} p_\theta\big(i \mid \mathcal{H}_u, S_u^t\big)\,. \tag{2}$$

Here, $\mathcal{H}_u$ retains the personalized behavioral context, while S_u^t represents the current positive intents and negative constraints. In ControlRec, positive intents instantiate separate generation branches, whereas negative constraints provide shared exclusion guidance across those branches. The construction and execution of S_u^t are detailed in Section 4.

3.2 Discrete Diffusion Modeling

Existing diffusion-based generative recommenders condition masked SID generation on the user's behavioral history. In contrast to autoregressive generation, which predicts SID codes in a fixed left-to-right order, discrete diffusion generates an item by iteratively reconstructing a partially masked

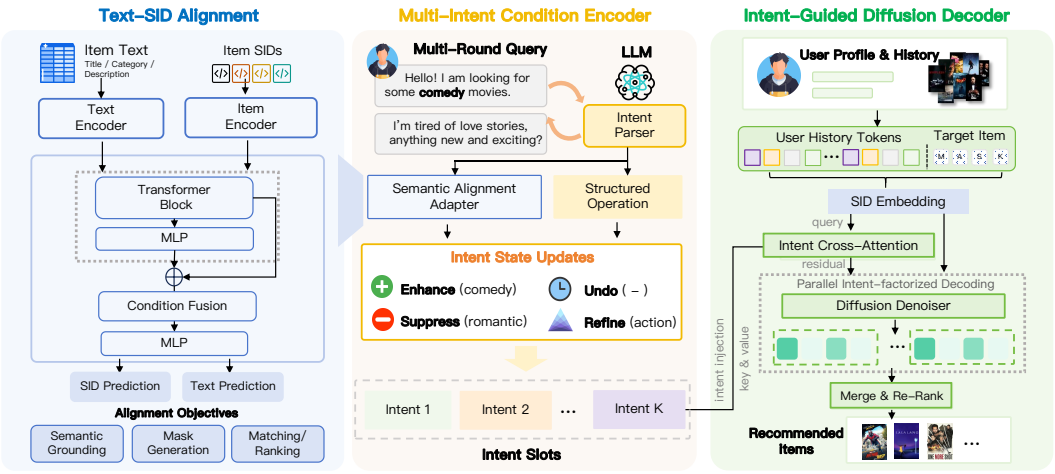

Figure 2: The overall architecture of CONTROLREC, with two condition-construction modules and an intent-factorized diffusion decoder: (a) Semantic Alignment Adapter establishes the semantic interface between natural-language intent and item SIDs; (b) Multi-Intent Condition Encoder updates an editable intent state and constructs personalized conditions; (c) the Intent-Guided Diffusion Decoder incorporates these conditions into denoising to steer personalized item generation.

SID sequence (Nie et al., 2025; Shi et al., 2025). Let $\mathbf{s}^0$ denote the complete SID of an item and $\mathbf{s}^R$ its fully masked state. The joint probability of the reverse trajectory is factorized as

$$p_\theta(\mathbf{s}^{0:R-1} \mid \mathbf{s}^R, \mathcal{H}_u) = \prod_{r=1}^{R} p_\theta(\mathbf{s}^{r-1} \mid \mathbf{s}^r, \mathcal{H}_u), \tag{3}$$

where $\mathbf{s}^r$ is the partially masked SID at diffusion step r. At each step, the model predicts all masked positions jointly, retains high-confidence codes, and updates the remaining positions until a complete SID is obtained. The diffusion model is well suited to conditional generation, as external conditions can guide its iterative denoising process. We therefore retain $\mathcal{H}_u$ as the personalized context and introduce the updated intent state as an additional condition, decoding each active positive intent independently.

4 METHODS

As illustrated in Figure 2, CONTROLREC comprises two condition-construction modules and an intent-factorized decoder. The Semantic Alignment Adapter grounds natural-language descriptions in SID space (Section 4.1); the Multi-Intent Condition Encoder constructs branch-specific positive conditions and shared negative guidance (Section 4.2); and the Intent-Guided Diffusion Decoder performs separate denoising for positive intents under shared behavioral context (Section 4.3).

4.1 SEMANTIC ALIGNMENT ADAPTER

Although SIDs encode item semantics, they do not directly correspond to natural language. We therefore introduce a **Semantic Alignment Adapter** that jointly models item text and SIDs through bidirectional cross-modal reconstruction, providing aligned representations and logits for downstream conditioning.

Adapter architecture. For each item $i \in \mathcal{I}$, let x_i denote its text and $\mathbf{s}_i \equiv \mathbf{s}_i^0 = (s_{i,1}, \ldots, s_{i,D})$ its complete D-code SID, assigned by the Multi-Head VQ-VAE tokenizer in LLaDA-Rec (Shi et al., 2025). We encode x_i using Sentence-T5 (Ni et al., 2022) and $\mathbf{s}_i$ using the SID embedding module of LLaDA-Rec, and concatenate their embeddings into a unified sequence. A shared Transformer first contextualizes this sequence, followed by a direction-specific residual MLP that adapts the hidden states to the current prediction direction. Condition Fusion then mean-pools the visible

context and injects it into each target position through a residual MLP. Finally, the SID Prediction or Text Prediction head reconstructs the target modality. The history-only recommendation path does not use these cross-modal components, while the trained text-to-SID path provides representations and SID logits for language-based intent conditioning. Given the related text x_c associated with a validated category c, the trained text-to-SID path produces $(\mathbf{z}(c), \mathbf{B}(c)) = A_{\text{align}}(x_c)$, where $\mathbf{z}(c)$ is an aligned representation, $\mathbf{B}(c)$ contains depth-wise SID logits, and A_{align} denotes the trained adapter. They provide the embedding and logit conditions for downstream modules.

Alignment training. We train the adapter with bidirectional text–SID alignment ($\mathcal{L}_{\text{sem}}$), masked SID generation ($\mathcal{L}_{\text{mask}}$), and candidate discrimination ($\mathcal{L}_{\text{rank}}$). They respectively reconstruct text and SIDs across modalities, recover masked or next-item SIDs from history, and score history–candidate compatibility:

$$\mathcal{L}_{\text{align}} = \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}} + \lambda_{\text{rank}}\mathcal{L}_{\text{rank}}, \tag{4}$$

The coefficients balance the three objectives; task definitions and loss details are in Appendix A.1.

4.2 Multi-Intent Condition Encoder

The **Multi-Intent Condition Encoder** translates free-form queries into decoder conditions using a frozen LLM parser and the trained Semantic Alignment Adapter. Positive preferences produce intent conditions, while negative constraints provide shared SID-logit guidance.

Intent slot initialization. We initialize the intent slots by softly assigning the items in $\mathcal{H}_u$ to a learned codebook of behavioral preference prototypes and aggregating the assignments with recency weights. The top K_0 prototypes are projected into the decoder hidden space. The initial state is

$$S_u^0 = \left(\{(\mathbf{z}_{u,k}^0, \pi_{u,k}^0)\}_{k=1}^{K_0}, \mathbf{0}\right), \tag{5}$$

where $\mathbf{z}_{u,k}^0$ is the k-th initial intent slot, $\pi_{u,k}^0$ is its normalized history-derived weight, and $\mathbf{0}$ indicates that no suppression constraint is active. This initialization preserves multiple behavior-derived preference directions.

Operation-aware state update. At turn t, the LLM parses the current query into a structured editing operation. A deterministic state updater applies the valid operations accumulated through turn t, after which a condition builder constructs the state consumed by the decoder:

$$o_u^t = \text{LLM}\big(q_u^t\big), \qquad S_u^t = \text{Build}\big(\text{ApplyEdits}(o_u^{1:t}), S_u^0\big). \tag{6}$$

Here, q_u^t is the user's current query and o_u^t is its parsed operation. Each operation specifies an action, a validated category, and a strength. ApplyEdits accumulates *Enhance*, *Suppress*, *Refine*, and *Undo* as positive addition, negative addition, category replacement, and constraint removal, respectively. These edits update the persistent preference constraints; Build reconstructs the decoder state rather than modifying slots in place. Further details are provided in the appendix.

Multi-intent condition construction. The condition builder instantiates S_u^t in Eq. 2 by mapping each positive intent $c_{u,k}^{+,t}$ to a decoder condition $\mathbf{z}_{u,k}^t$ and converting the active negative constraints $\mathcal{N}_u^t$ into shared SID-logit guidance $\mathbf{B}_u^{-,t}$, yielding the decoder-facing representation:

$$S_u^t = \left(\{(\mathbf{z}_{u,k}^t, \pi_{u,k}^t)\}_{k=1}^{K_t}, \mathbf{B}_u^{-,t}\right), \tag{7}$$

Here, K_t is the number of active positive intents, $\mathbf{z}_{u,k}^t$ is the k-th intent condition, $\pi_{u,k}^t$ is its normalized branch prior, and $\mathbf{B}_u^{-,t}$ is the shared negative SID-logit guidance. Explicit positive constraints define the active branches; when none are present, the builder falls back to the history-derived slots S_u^0. In either case, $\mathcal{H}_u$ remains the shared personalization context, and negative guidance constrains all branches without instantiating an additional one.

4.3 Intent-Guided Diffusion Decoder

We build the decoder on the masked discrete diffusion backbone of LLaDA-Rec (Shi et al., 2025). Given the updated state S_u^t, the decoder combines the shared behavioral history with one positive

intent slot in each generation branch. The negative logit condition is applied to every branch, while the slot weights are used only for final list aggregation. The decoder uses intent information to steer successive SID predictions while retaining behavioral history as the personalization context.

Intent-conditioned denoising. For a fixed user u and turn t, the backbone encodes the history SIDs together with the partially masked target s_k^r. At each denoising step, the intent condition $z_{u,k}^t$ is injected into the hidden states of the masked target positions through cross-attention. This makes the subsequent SID predictions depend on both behavioral context and the active intent:

$$\mathbf{h}_k^r = \widetilde{\mathbf{h}}_k^r + \mathrm{CrossAttn}\Big(\widetilde{\mathbf{h}}_k^r, \mathbf{z}_{u,k}^t, \mathbf{z}_{u,k}^t\Big), \tag{8}$$

Here, $\tilde{h}_k^r$ and h_k^r denote the masked-position hidden states before and after intent injection; fixed user and turn indices are omitted. The denoiser uses the conditioned states to predict the masked SID codes and update s_k^r to s_k^{r-1}. Repeating this injection lets intent information guide the successive prediction steps.

Intent-factorized decoding and aggregation. For K_t positive intent slots, we replicate the masked target state and evaluate all branches in parallel along the batch dimension. The branches share model parameters, behavioral history, and $\mathbf{B}_u^{-,t}$, but keep their intent slots and denoising states separate. Negative constraints provide an additional control signal at the prediction stage. At each denoising step, the shared guidance $B_u^{-,t}$ adjusts the logits at SID position d to discourage codes associated with unwanted preferences:

$$\widetilde{\mathbf{a}}_{k,d}^r = \mathbf{a}_{k,d}^r - \alpha_d^r \mathbf{b}_{u,d}^{-,t}, \tag{9}$$

where $\mathbf{a}_{k,d}^r$ and $\widetilde{\mathbf{a}}_{k,d}^r$ are the SID logits before and after suppression, $\mathbf{B}_u^{-,t} = \{\mathbf{b}_{u,d}^{-,t}\}_{d=1}^D$ is the negative guidance, and $\alpha_d^r \geq 0$ controls its strength. When no suppression is active, $\mathbf{B}_u^{-,t} = \mathbf{0}$. For multiple active intents, each intent guides a separate generation branch under the same behavioral history and negative constraints, preserving a distinct condition for each preference. Each branch returns catalog-valid candidates $C_{u,k}^t$. We add $\log \pi_{u,k}^t$ to its candidate scores, merge duplicates by retaining the highest adjusted score, and return the top N unique items. Implementation and aggregation details are provided in Appendix A.5.

Conditional recommendation training. Following semantic alignment, we train the conditional recommendation model with the LLM and item SID codebook fixed. Given behavioral history and language feedback, the model learns to reconstruct the next-item SID by minimizing

$$\mathcal{L}_{\mathrm{rec}} = \mathcal{L}_{\mathrm{denoise}} + \lambda_s \mathcal{L}_{\mathrm{intent}}, \tag{10}$$

where $\mathcal{L}_{\mathrm{denoise}}$ reconstructs masked target SID tokens, training the decoder to use current intent together with behavioral context. The auxiliary objective $\mathcal{L}_{\mathrm{intent}}$ grounds intent slots in target-item semantics and discourages redundant slot representations, with λ_s balancing the two objectives. Condition-free examples are also included to retain recommendation from behavioral history alone.

5 Experiments

Our experiments aim to answer the following progressive questions: **(RQ1)**: Can ControlRec preserve strong next-item recommendation performance without explicit user feedback? **(RQ2)**: Can explicit intent refinement recover hard cases missed by a history-only recommender? **(RQ3)**: Can our model reliably execute positive preference edits, negative constraints, and multiple simultaneous intents from conversational feedback? **(RQ4)**: How do the core modules of our model contribute to improvements in recommendation performance?

5.1 Experimental Setup

Datasets and metrics. We evaluate ControlRec on Amazon Reviews 2023 (Hou et al., 2026) and ReDial (Kostric & Balog, 2026). For standard next-item recommendation, we use three official Amazon domains: Industrial and Scientific, Musical Instruments, and Video Games. We use ReDial

Table 1: Full-catalog next-item recommendation. **Bold** and underlined values indicate the best and second-best results across all methods, respectively. CONTROLREC results are reported as mean ± standard deviation over three random seeds.

	Industrial and Scientific					Musical Instruments					Video Games				
Method	**R@1**	**R@5**	**R@10**	**N@5**	**N@10**	**R@1**	**R@5**	**R@10**	**N@5**	**N@10**	**R@1**	**R@5**	**R@10**	**N@5**	**N@10**
Item ID-based Methods															
GRU4Rec	0.0071	0.0184	0.0272	0.0128	0.0156	0.0094	0.0297	0.0453	0.0196	0.0246	0.0149	0.0461	0.0712	0.0307	0.0387
SASRec	0.0063	0.0240	0.0379	0.0152	0.0197	0.0089	0.0331	0.0525	0.0211	0.0273	0.0128	0.0516	0.0823	0.0323	0.0421
BERT4Rec	0.0045	0.0157	0.0264	0.0100	0.0134	0.0065	0.0255	0.0412	0.0160	0.0211	0.0082	0.0315	0.0530	0.0199	0.0267
FMLP-Rec	0.0046	0.0181	0.0300	0.0113	0.0151	0.0086	0.0299	0.0496	0.0193	0.0257	0.0099	0.0395	0.0649	0.0246	0.0328
DreamRec	0.0052	0.0184	0.0299	0.0118	0.0155	0.0069	0.0245	0.0423	0.0157	0.0214	0.0125	0.0381	0.0611	0.0253	0.0326
DiffuRec	0.0050	0.0190	0.0310	0.0119	0.0158	0.0077	0.0283	0.0465	0.0179	0.0237	0.0111	0.0425	0.0709	0.0268	0.0359
LRURec	0.0049	0.0169	0.0267	0.0110	0.0141	0.0071	0.0272	0.0431	0.0172	0.0223	0.0134	0.0480	0.0753	0.0308	0.0396
HSTU	0.0104	0.0306	0.0450	0.0207	0.0253	0.0132	0.0401	0.0605	0.0267	0.0333	0.0257	**0.0775**	**0.1153**	0.0517	**0.0639**
Semantic ID-based Methods															
VQ-Rec	0.0076	0.0248	0.0385	0.0162	0.0206	0.0099	0.0345	0.0532	0.0222	0.0282	0.0150	0.0497	0.0769	0.0325	0.0412
TIGER	0.0084	0.0282	0.0446	0.0183	0.0236	0.0105	0.0359	0.0566	0.0233	0.0300	0.0166	0.0529	0.0823	0.0348	0.0442
TIGER-SAS	0.0067	0.0221	0.0356	0.0144	0.0187	0.0102	0.0342	0.0521	0.0223	0.0280	0.0170	0.0548	0.0847	0.0360	0.0457
LETTER	0.0082	0.0273	0.0423	0.0179	0.0227	0.0114	0.0362	0.0562	0.0239	0.0303	0.0169	0.0552	0.0863	0.0362	0.0462
LC-Rec	0.0091	0.0280	0.0434	0.0186	0.0235	0.0119	0.0379	0.0587	0.0251	0.0318	0.0165	0.0567	0.0891	0.0366	0.0471
RPG	0.0087	0.0257	0.0395	0.0174	0.0218	0.0118	0.0362	0.0545	0.0241	0.0300	0.0209	0.0579	0.0853	0.0397	0.0485
LLaDA-Rec	0.0098	0.0310	0.0474	0.0203	0.0256	0.0128	0.0406	0.0623	0.0268	0.0337	0.0203	0.0623	0.0942	0.0415	0.0517
CONTROLREC	**0.0115** ±.00005	**0.0327** ±.00005	**0.0483** ±.00008	**0.0220** ±.00004	**0.0270** ±.00005	**0.0149** ±.00027	**0.0426** ±.00015	**0.0636** ±.00010	**0.0289** ±.00013	**0.0356** ±.00012	**0.0271** ±.00015	0.0765 ±.00057	0.1101 ±.00037	**0.0522** ±.00045	0.0630 ±.00038

to evaluate conversational preference editing and multi-intent control. Following prior work (Shi et al., 2025; Rajput et al., 2023), we report Recall@k and NDCG@k with $k \in \{1, 5, 10\}$. For ReDial, we additionally report Sat.@10, Viol.@10, and All-Req@10 for positive-intent satisfaction, negative-intent violation, and multi-intent coverage, respectively.

Baselines. For standard next-item recommendation, we compare CONTROLREC with representative sequential and generative recommendation methods, including GRU4Rec (Hidasi et al., 2016), SASRec (Kang & McAuley, 2018), BERT4Rec (Sun et al., 2019), FMLP-Rec (Zhou et al., 2022), DreamRec (Yang et al., 2023), DiffuRec (Li et al., 2023), and TIGER (Rajput et al., 2023). For intent-guided recovery, we compare with conversational recommenders (Chen et al., 2019; Zhou et al., 2020a; Wang et al., 2022a;b; Li et al., 2025), LLM-based agent recommenders, and LLaDA-Rec backbone (Shi et al., 2025). Other diffusion baselines lack conversational intent control.

Experimental Protocol. We use HSTU as a recommendation baseline to establish a fixed recovery benchmark. Specifically, we select the HSTU checkpoint with the best validation NDCG@10 and identify 43,577 test cases for which the ground-truth items are not included in its top-100 recommendations. Existing recommendation datasets do not systematically record the corrective user feedback needed to evaluate conversational intent control. We therefore simulate dialogues using an LLM to generate coarse-grained preference descriptions from the ground-truth target's category metadata. Construction details and statistics are provided in Appendix A.6.1. Each case follows a three-turn refinement trajectory: T1 *enhances* a target-consistent broad category, T2 *suppresses* an over-exposed sibling, and T3 *refines* to a target-consistent finer category. The generated utterances do not reveal the target title, item ID, SID, future behavior, or item-specific attributes. All experiments are conducted on NVIDIA A100 80GB GPUs with 3 random seeds.

5.2 OVERALL RECOMMENDATION PERFORMANCE (RQ1)

Tab. 1 reports the dialogue-free recommendation results. Without explicit user feedback, CONTROLREC derives four implicit user intents by softly assigning historical items to a learned preference-prototype codebook and selecting the top four prototypes to guide item generation. On Industrial and Musical datasets, CONTROLREC achieves the best Recall and NDCG at all reported cutoffs, e.g., improving over the second-best method by 10.6% on R@1 and 5.5% on N@10 on Industrial. On Video Games, CONTROLREC achieves the best R@1 and N@5 and remains highly competitive with HSTU on the remaining metrics. These results show that even when user intent is inferred implicitly from behavioral patterns, intent-guided generation can consistently achieve competitive recommendation performance with state-of-the-art methods.

5.3 DIALOGUE-ASSISTED HARD-CASE RECOVERY (RQ2)

Tab. 2 evaluates intent-guided recovery on the HSTU miss@100 subset. Our method slightly underperforms LLaDA-Rec in the history-only setting, possibly because population-level intent priors

Table 2: Assisted recovery on the frozen HSTU miss@100 subset. T0 is history-only; T1–T3 is the cumulative Broad / Suppress / Refine trajectory. The gray row serves as a rule-based reference.

	T0: History		T1: Broad		T2: Suppress		T3: Refine	
Method	**R@10**	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**
Conversational Recommenders								
KBRD	0.00771	0.00364	0.01308	0.00653	0.01427	0.00715	<u>0.02132</u>	<u>0.01118</u>
KGSF	0.00014	0.00005	0.00509	0.00234	0.01437	0.00663	0.02031	0.00952
BARCOR	0.00014	0.00005	0.00555	0.00255	0.00402	0.00190	0.00528	0.00239
UniCRS	0.00016	0.00006	0.00457	0.00193	0.00406	0.00172	0.00532	0.00235
LLM-based Agent Recommenders								
ChatCRS	–	–	<u>0.01574</u>	<u>0.00797</u>	<u>0.01572</u>	<u>0.00797</u>	0.01597	0.00812
Meta-Llama-3-8B-Instruct	0.00872	0.00461	0.00369	0.00185	0.00546	0.00260	0.00799	0.00393
Qwen2.5-7B text retriever	0.00831	0.00457	0.00434	0.00209	0.00615	0.00291	0.00737	0.00373
DeepSeek-R1-Distill-Qwen-32B	0.00783	0.00425	0.00248	0.00117	0.00287	0.00127	0.00422	0.00207
Qwen2.5-72B-Instruct	0.00808	0.00441	0.00264	0.00125	0.00326	0.00159	0.00418	0.00198
Diffusion-based Generative Methods								
LLaDA-Rec	**0.01157**	**0.00550**	0.01157	0.00550	0.01157	0.00550	0.01157	0.00550
LLaDA-Rec + Tag Reranking†	0.01157	0.00550	0.01928	0.00931	0.02139	0.01046	0.02302	0.01141
CONTROLREC	<u>0.01067</u>	<u>0.00494</u>	**0.01960**	**0.00959**	**0.02761**	**0.01392**	**0.03438**	**0.01726**

Table 3: Controlled dialogue recommendation on the ReDial dataset. **Bold** and <u>underlined</u> values indicate the best and second-best results, respectively.

	Positive Editing			Negative Editing			Multi-Intent Control		
Method	**R@10**	**N@10**	**Sat.@10↑**	**R@10**	**N@10**	**Viol.@10↓**	**R@10**	**N@10**	**All-Req@10↑**
KBRD	0.2286	0.1120	0.7103	0.2600	0.1283	0.3480	0.2333	0.1099	0.5733
KGSF	0.2229	0.1273	<u>0.7840</u>	0.2333	0.1322	0.5013	<u>0.2867</u>	<u>0.1579</u>	0.7067
BARCOR	<u>0.2571</u>	<u>0.1357</u>	0.7674	<u>0.2867</u>	<u>0.1543</u>	0.4280	0.2400	0.1258	0.7600
UniCRS	0.2400	0.1247	0.7629	0.2733	0.1426	0.4360	0.2533	0.1250	0.8667
PECRS	0.0114	0.0036	0.3823	0.0133	0.0042	0.3373	0.0133	0.0042	0.2133
Qwen2.5-7B	0.1714	0.0882	0.5697	0.1867	0.0982	0.3493	0.1667	0.0845	<u>0.9667</u>
DeepSeek-R1-Distill-Qwen-32B	0.1543	0.0900	0.5691	0.1733	0.1027	<u>0.3233</u>	0.1533	0.0879	0.8933
CONTROLREC	**0.2914**	**0.1446**	**0.9943**	**0.4333**	**0.2415**	**0.0100**	**0.4200**	**0.2175**	**0.9800**

favor common preference patterns that do not match these users' target interests. Nevertheless, CONTROLREC improves consistently as feedback accumulates, with R@10 increasing from 0.0107 at T0 to 0.0344 over T1–T3, and N@10 showing the same monotonic trend. We additionally include LLaDA-Rec with rule-based tag reranking, which uses matches between the expressed category constraints and catalog tags to rerank the generated candidates. CONTROLREC consistently outperforms this reference from T1 to T3. Appendix B.5 further applies the same tag reranking to both candidate sources. This indicates that the gain comes from changing the candidate pool rather than only reordering it. After three turns, CONTROLREC outperforms the strongest CRS baseline, KBRD, by 61.2% in Recall and 54.4% in NDCG. In contrast, the dialogue-unaware LLaDA-Rec remains unchanged, while direct LLM-based recommenders do not consistently benefit from accumulated feedback. Shuffled trajectories substantially reduce the gains, confirming that recovery depends on the specific intent refinements rather than the presence of language alone (Appendix B.2).

5.4 CONTROLLABLE CONVERSATIONAL RECOMMENDATION (RQ3)

We further evaluate whether CONTROLREC can handle different forms of conversational intent control on the ReDial dataset.

Positive editing. For positive requests, CONTROLREC achieves R@10=0.2914 and N@10=0.1446, while satisfying nearly all stated preferences with Sat.@10=0.9943. Compared with conventional CRS and direct LLM recommenders, CONTROLREC better preserves personalized ranking quality while responding to the newly expressed intent. This shows that positive feedback can effectively redirect the recommendation without discarding the user's historical preferences.

Negative editing. We next evaluate whether the model can suppress an undesired preference that may already be prominent in the original ranking. CONTROLREC reduces Viol.@10 to 0.0100, substantially lower than all baselines, while also achieving the best R@10 and N@10. The result

indicates that the editable intent state can selectively remove unwanted preference signals without sacrificing relevant personalized items.

Multi-intent control. Finally, we consider a more challenging case where several intents must be satisfied simultaneously. CONTROLREC achieves R@10=0.4200, N@10=0.2175, and All-Req@10=0.9800. Direct LLM recommenders can achieve relatively high intent coverage but still perform worse on target ranking, whereas conventional CRS methods cover fewer simultaneous requirements. By factorizing concurrent intents into separate generation branches and merging them after decoding, CONTROLREC maintains both multi-intent coverage and recommendation quality.

5.5 ABLATION STUDY (RQ4)

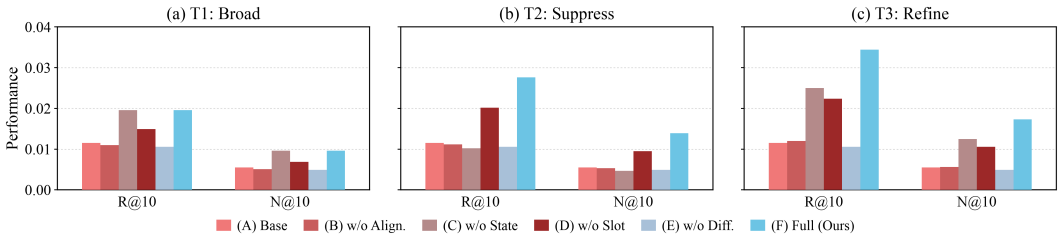

Figure 3: Ablation of individual design variants across the T1–T3 protocol.

As shown in Figure 3, we ablate four key components on top of the LLaDA-Rec backbone (Base): (a) the Semantic Alignment Adapter (**Align.**), (b) multi-turn preference state updating (**State**), (c) multi-intent slot conditioning (**Slot**), and (d) the Intent-Guided Diffusion Decoder (**Diff.**). We observe that Base and w/o Diff. remain essentially unchanged from T1 to T3, showing that intent conditions affect the ranking only when they are incorporated into SID generation. Align. mainly accounts for the first-turn gain: without semantic alignment, T1 is weaker and subsequent improvements are also limited. The effects of State and Slot are most evident at T2. Removing Slot weakens the model's response to the suppressed sibling, while removing State drops below Base, indicating that the latest utterance alone is insufficient without the accumulated preference state. Only the full model improves consistently through T3 (R@10/N@10 = 0.03438/0.01726), showing that the four components are complementary.

Discussion Our Amazon evaluation is constrained by the available interaction logs, which do not record users' corrective requests or natural-language preferences. We therefore use target-derived category-level feedback to construct a controlled evaluation protocol for progressively specified preferences. In practical use, however, users can express their needs directly in natural language and provide feedback at any point to progressively refine the generated recommendations. Such requests guide generation through continuous semantic conditioning, as illustrated by our interactive demo. CONTROLREC also retains efficient intent-conditioned decoding, with language interpretation accounting for most of the conversational inference time. Further qualitative examples and efficiency analysis are provided in the appendix.

6 CONCLUSION

In this work, we presented CONTROLREC, a unified framework for intent-mediated generative recommendation. By making user intent an explicit and editable intermediate state, CONTROLREC connects history-based personalization with natural-language guidance, allowing users to progressively revise their needs and steer item generation. Semantic alignment, persistent intent management, and intent-guided diffusion jointly support this process. Experiments on three Amazon domains and ReDial demonstrate competitive recommendation performance and effective control over positive preferences, negative constraints, and multiple concurrent intents. These results support intent modeling as a practical interface between users' evolving needs and personalized generation. Future work will investigate learning from naturally occurring feedback and adapting intent management to longer, more complex interactions.

AI USE STATEMENT

We used generative AI tools to assist with English language editing, manuscript expression refinement, code development, and code debugging. We did not use generative AI tools to generate experimental results or replace the authors' evaluation of the scientific validity of the work. All AI-assisted work was reviewed and verified by the authors. In particular, the authors checked the correctness of AI-assisted code and debugging suggestions, as well as the accuracy and clarity of the revised manuscript. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses publicly available datasets for research on recommendation. We do not collect new personal information or conduct experiments involving human subjects. The datasets are used in accordance with their respective usage conditions.

REFERENCES

Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2942–2951. Association for Computing Machinery (ACM), August 2020. doi: 10.1145/3394486.3403344.

Zheng Chai, Zhihong Chen, Chenliang Li, Rong Xiao, Houyi Li, Jiawei Wu, Jingxu Chen, and Haihong Tang. User-aware multi-interest learning for candidate matching in recommenders. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1326–1335. Association for Computing Machinery (ACM), July 2022. doi: 10.1145/3477495.3532073.

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1803–1813. Association for Computational Linguistics (ACL), January 2019. doi: 10.18653/v1/d19-1189.

Sunhao Dai, Jiakai Tang, Jiahua Wu, Kun Wang, Yuxuan Zhu, Bingjun Chen, Bangyang Hong, Yu Zhao, Cong Fu, Kangle Wu, Yabo Ni, Anxiang Zeng, Wenjie Wang, Xu Chen, Jun Xu, and See-Kiong Ng. Onepiece: Bringing context engineering and reasoning to industrial cascade ranking system. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 7117–7128. Association for Computing Machinery (ACM), August 2026. doi: 10.1145/3770855.3818456.

Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment, February 2025.

Kairui Fu, Tao Zhang, Shuwen Xiao, Ziyang Wang, Xinming Zhang, Chenchi Zhang, Yuliang Yan, Junjun Zheng, Xiangheng Kong, Shengyu Zhang, Kun Kuang, and Yuning Jiang. Forge: Forming semantic identifiers for generative retrieval in industrial datasets. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 8858–8869. Association for Computing Machinery (ACM), August 2026. doi: 10.1145/3770855.3817477.

Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126, 2021. doi: 10.1016/j.aiopen.2021.06.002.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 299–315. Association for Computing Machinery (ACM), September 2022. doi: 10.1145/3523227.3546767.

Xian Guo, Ben Chen, Siyuan Wang, Ying Yang, Mingyue Cheng, Chenyi Lei, Yuqing Ding, and Han Li. Onesug: The unified end-to-end generative framework for e-commerce query suggestion. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(17):14774–14782, March 2026. doi: 10.1609/aaai.v40i17.38497.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 639–648, 2020.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems 35*, pp. 8633–8646. Curran Associates, January 2022. doi: 10.52202/068431-0628.

Yupeng Hou, Jiacheng Li, Xiangjun Fu, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation: Benchmarking llms as semantic encoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3251–3265. Association for Computational Linguistics (ACL), January 2026. doi: 10.18653/v1/2026.acl-long.147.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. Institute of Electrical and Electronics Engineers (IEEE), November 2018. doi: 10.1109/icdm.2018.00035.

Ivica Kostric and Krisztian Balog. A standardized re-evaluation of conversational recommender systems on the redial dataset. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2952–2960. Association for Computing Machinery (ACM), July 2026. doi: 10.1145/3805712.3808573. URL https://doi.org/10.1145/3805712.3808573.

Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. Multi-interest network with dynamic routing for recommendation at tmall. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 2615–2623, 2019.

Chuang Li, Yang Deng, Hengchang Hu, Min-Yen Kan, and Haizhou Li. Chatcrs: Incorporating external knowledge and goal guidance for llm-based conversational recommender systems. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 295–312. Association for Computational Linguistics (ACL), January 2025. doi: 10.18653/v1/2025.findings-naacl.17.

Shuokai Li, Ruobing Xie, Yongchun Zhu, Xiang Ao, Fuzhen Zhuang, and Qing He. User-Centric conversational recommendation with Multi-Aspect user modeling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 223–233. Association for Computing Machinery, 2022. doi: 10.1145/3477495.3532074.

Zihao Li, Aixin Sun, and Chenliang Li. DiffuRec: A diffusion model for sequential recommendation. *ACM Transactions on Information Systems*, 42(3), 2023.

Zihao Li, Qiang Chen, Lixin Zou, Aixin Sun, and Chenliang Li. Multi-interest recommendation: A survey. *ACM Transactions on Information Systems*, 44(4):1–38, April 2026. doi: 10.1145/3789510.

Zhao Liu, Yichen Zhu, Yiqing Yang, Xiao Lv, Guoping Tang, Rui Huang, Qiang Luo, Ruiming Tang, and Guorui Zhou. Diffgrm: Diffusion-based generative recommendation model. In *Proceedings of the ACM Web Conference 2026*, pp. 5853–5864. Association for Computing Machinery (ACM), April 2026. doi: 10.1145/3774904.3792156.

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. Association for Computational Linguistics (ACL), January 2022. doi: 10.18653/v1/2022.findings-acl.146.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan LI. Large language diffusion models. In *Advances in Neural Information Processing Systems 38*, pp. 56354–56392. Curran Associates, January 2025. doi: 10.52202/085713-1689.

Haohao Qu, Shanru Lin, Yujuan Ding, Yiqi Wang, and Wenqi Fan. Diffusion generative recommendation with continuous tokens. In *Proceedings of the ACM Web Conference 2026*, pp. 7259–7270. Association for Computing Machinery (ACM), April 2026. doi: 10.1145/3774904.3792428.

Yuanpeng Qu and Hajime Nobuhara. Intent-aware diffusion with contrastive learning for sequential recommendation. In *Proceedings of the 48th international ACM SIGIR conference on research and development in information retrieval*, pp. 1552–1561, 2025.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, Maciej Kula, Ed Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems 36*, pp. 10299–10315. Curran Associates, January 2023. doi: 10.52202/075280-0452.

Mathieu Ravaut, Hao Zhang, Lu Xu, Aixin Sun, and Yong Liu. Parameter-efficient conversational recommender system as a language processing task. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 152–165. Association for Computational Linguistics (ACL), January 2024. doi: 10.18653/v1/2024.eacl-long.9.

Teng Shi, Chenglei Shen, Weijie Yu, Shen Nie, Chongxuan Li, Xiao Zhang, Ming He, Yan Han, and Jun Xu. Llada-rec: Discrete diffusion for parallel semantic id generation in generative recommendation, November 2025.

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.

Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 598–606. Association for Computing Machinery (ACM), March 2021. doi: 10.1145/3437963.3441811.

OneRec Team, Biao Yang, Boyang Ding, Chenglong Chu, Dunju Zang, Fei Pan, Han Li, Hao Jiang, Honghui Bao, Huanjie Wang, Jian Liang, Jiangxia Cao, Jiao Ou, Jiaxin Deng, Jinghao Zhang, Kun Gai, Lu Ren, Peiru Du, Pengfei Zheng, Rongzhou Zhang, Ruiming Tang, Shiyao Wang, Siyang Mao, Siyuan Lou, Teng Shi, Wei Yuan, Wenlong Xu, Xingchen Liu, Xingmei Wang, Xinqi Jin, Yan Sun, Yan Wang, Yifei Hu, Yingzhi He, Yufei Ye, Yuhao Wang, Yunhao Zhou, Yuqin Dai, Zhao Liu, Zhipeng Wei, Zhixin Ling, Ziming Li, Zixing Zhang, Ziyuan Liu, An Zhang, Changxin Lao, Chaoyi Ma, Chengru Song, Defu Lian, Fan Yang, Guowang Zhang, Hao Peng, Jiayao Shen, Jie Chen, Jun Xu, Junmin Chen, Kun Zhang, Kuo Cai, Mingxing Wen, Minmao Wang, Minxuan Lv, Qi Zhang, Qiang Luo, Sheng Yu, Shijie Li, Shijie Yi, Shuang Yang, Shugui Liu, Shuni Chen, Tinghai Zhang, Tingting Gao, Xiang Wang, Xiangyu Wu, Xiangyu Zhao, Xiao Lv, Xiaoyou Zhou, Xuming Wang, Yong Du, Zejian Zhang, Zhaojie Liu, Zhiyang Zhang, Zhuang Zhuang, Ziqi Wang, and Ziyi Zhao. Onereason technical report, June 2026.

Ting-Chun Wang, Shang-Yu Su, and Yun-Nung Chen. BARCOR: Towards A unified framework for conversational recommendation systems, 2022a.

Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. Diffusion recommender model. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 832–841. Association for Computing Machinery (ACM), July 2023. doi: 10.1145/3539618.3591663.

Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1929–1937. Association for Computing Machinery (ACM), August 2022b. doi: 10.1145/3534678.3539382.

Yibiao Wei, Jie Zou, Weikang Guo, Guoqing Wang, Xing Xu, and Yang Yang. Mscrs: Multi-modal semantic graph prompt learning framework for conversational recommender systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 42–52, 2025. doi: 10.1145/3726302.3730040.

Yue Wu, Mingyu Zhao, Songming Zhang, Ziyu Lyu, Dayong Peng, Wanji Zheng, and Hua Chai. A semi-supervised co-training algorithm for robust recommendation. *ACM Transactions on Knowledge Discovery from Data*, 20(1):1–25, December 2025. doi: 10.1145/3773988.

Zhengyi Yang, Jiancan Wu, Zhicai Wang, Xiang Wang, Yancheng Yuan, and Xiangnan He. Generate what you prefer: Reshaping sequential recommendation via guided diffusion. *Advances in Neural Information Processing Systems*, 36:24247–24261, 2023.

Wencai Ye, Mingjie Sun, Shuhang Chen, Wenjin Wu, and Peng Jiang. Align³gr: Unified multi-level alignment for llm-based generative recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16154–16162, March 2026. doi: 10.1609/aaai.v40i19.38651.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, volume 00, pp. 3813–3824. Institute of Electrical and Electronics Engineers (IEEE), October 2023. doi: 10.1109/iccv51070.2023.00355.

Yiming Zhang, Lingfei Wu, Qi Shen, Yitong Pang, Zhihua Wei, Fangli Xu, Bo Long, and Jian Pei. Multiple choice questions based Multi-Interest policy learning for conversational recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 2153–2162. Association for Computing Machinery, 2022. doi: 10.1145/3485447.3512088.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1006–1014. Association for Computing Machinery (ACM), August 2020a. doi: 10.1145/3394486.3403143.

Kun Zhou, Wayne Xin Zhao, Hui Wang, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. Leveraging historical interaction data for improving conversational recommender system. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 2349–2352. Association for Computing Machinery, 2020b. doi: 10.1145/3340531.3412098.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced mlp is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 2388–2399. Association for Computing Machinery (ACM), April 2022. doi: 10.1145/3485447.3512111.

A Detailed Methodology

This section describes the implementation details of semantic alignment, intent initialization, editable state updates, conditional recommendation, and inference, together with the experimental protocols.

A.1 Semantic Alignment Pretraining

We organize alignment pretraining into six tasks across three groups: semantic alignment, masked generation, and matching and ranking. Table 4 specifies the visible condition and prediction target for each task. Together, these tasks link language to item codes, retain behavioral context, and provide candidate-level supervision.

Table 4: Six Pretraining Tasks by Learning Objective

Task group	Task	Condition	Target
Semantic alignment	Text2SID	title, category path, attributes	complete four-token SID
	SID2Attr	complete SID and output-view prompt	category/object/title tokens
Masked generation	MaskItem	history with one item masked in place	masked item's SID
	NextItem	behavioral prefix	next item's SID
Matching and ranking	Match	history and one candidate	binary compatibility label
	Rank	history and a positive–negative pair	position of the positive item

Shared input construction. Each example contains a task identifier, a visible condition block, and a prediction target. Text and SID tokens are distinguished by modality and block embeddings, with depth embeddings identifying SID positions. Generative tasks predict masked target tokens while keeping the condition visible; Match and Rank use a task-specific decision head.

Semantic alignment. Text2SID predicts an item's complete four-token SID from its title, category path, and attributes. SID2Attr learns the reverse correspondence by predicting category, object, or title tokens from the complete SID and an output-view prompt specifying the requested view. Both tasks use token-level cross-entropy:

$$\mathcal{L}_{\text{sem}} = -\mathbb{E}\left[\frac{1}{D}\sum_{d=1}^{D}\log p(s_d \mid x) + \frac{1}{L}\sum_{\ell=1}^{L}\log p(x_\ell \mid \mathbf{s}, v)\right]. \tag{11}$$

Here, $\mathbf{s} = (s_1, \ldots, s_D)$ is a SID with $D = 4$ tokens, x denotes item text, and v specifies the output view for SID2Attr. In the second term, x is the requested text target of length L. The target modality is masked, and p denotes the corresponding prediction head's softmax probability. These bidirectional prediction losses ground linguistic attributes in discrete item identities.

Masked generation. MaskItem reconstructs the SID of an item masked at its original position in the history. NextItem predicts the next item's SID in a masked target block following the visible history. Both tasks use cross-entropy over the target SID tokens:

$$\mathcal{L}_{\text{mask}} = -\mathbb{E}\left[\frac{1}{D}\sum_{d=1}^{D}\log p(s_d \mid \widetilde{\mathcal{H}}) + \frac{1}{D}\sum_{d=1}^{D}\log p(s_d^{+} \mid \mathcal{H})\right]. \tag{12}$$

Here, $\mathcal{H}$ is the visible history, $\widetilde{\mathcal{H}}$ replaces one history item with a masked SID block, $\mathbf{s}$ is that item's SID, and $\mathbf{s}^{+}$ is the next item's SID. Only target SID positions contribute to the loss. Together, these tasks preserve contextual reconstruction and behavioral prediction alongside semantic alignment.

Matching and ranking. Match receives a history and one candidate and predicts a binary compatibility label. Rank receives a history and a positive–negative candidate pair and predicts the positive candidate's position. We use binary cross-entropy for Match and two-class cross-entropy for Rank:

$$\mathcal{L}_{\text{rank}} = -\mathbb{E}\left[y\log q + (1-y)\log(1-q)\right] - \mathbb{E}\left[\log\frac{\exp(a_r)}{\exp(a_1)+\exp(a_2)}\right]. \tag{13}$$

Here, y is the binary compatibility label and q is its predicted positive probability. For Rank, a_1 and a_2 are the decision logits, and $r \in \{1, 2\}$ is the positive candidate's position. The two expectations are over the respective task examples. These decision losses provide candidate-level discrimination. The positive item is an observed training interaction; sampled negatives represent alternatives rather than explicit user dislikes.

The combined pretraining objective is

$$\mathcal{L}_{\text{align}} = \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}} + \lambda_{\text{rank}}\mathcal{L}_{\text{rank}}. \tag{14}$$

The coefficients balance the three groups, and only tasks enabled in a given training stage contribute to its objective. Behavioral examples are constructed from training interactions, excluding held-out validation and test events.

A.2 Preference-Prototype Codebook and Intent Initialization

The preference-prototype codebook represents shared behavioral preference patterns and provides a prior for initializing user intents. Let $\mathcal{P} = \{\mathbf{p}_m\}_{m=1}^{M}$ denote the learned preference prototypes. The prototypes are learned with next-item prediction supervision, together with assignment regularization that encourages concentrated assignments for individual users and balanced prototype usage across users. The learned codebook and collaborative item representations are kept fixed when constructing the history-based intent condition.

Given a user's observed history $\mathcal{H}_u = (i_{u,1}, \ldots, i_{u,L_u})$, we first compute a soft assignment of each historical item to the preference prototypes:

$$a_{\ell,m} = \frac{\exp\big(\cos(\mathbf{e}_{i_{u,\ell}}, \mathbf{p}_m)/\tau_c\big)}{\sum_{j=1}^{M} \exp\big(\cos(\mathbf{e}_{i_{u,\ell}}, \mathbf{p}_j)/\tau_c\big)}, \tag{15}$$

where $\mathbf{e}_{i_{u,\ell}}$ is the collaborative representation of item $i_{u,\ell}$ and τ_c controls assignment sharpness. We aggregate these assignments with recency weights:

$$\rho_\ell = \frac{\exp(\gamma\ell)}{\sum_{j=1}^{L_u} \exp(\gamma j)}, \qquad q_{u,m} = \sum_{\ell=1}^{L_u} \rho_\ell a_{\ell,m}. \tag{16}$$

Padding positions are excluded. The resulting distribution q_u expresses the user's historical support for each shared preference prototype.

We retain the $K_0 = 4$ prototypes with the largest probability mass. Let $m_{u,k}$ denote their indices. The initial intent representations and weights are

$$\bar{\mathbf{p}}^0_{u,k} = \text{norm}(\mathbf{p}_{m_{u,k}}), \qquad \pi^0_{u,k} = \frac{q_{u,m_{u,k}}}{\sum_{j=1}^{K_0} q_{u,m_{u,j}}}. \tag{17}$$

These representations are projected into the decoder's hidden space following Section 4.2:

$$\mathbf{z}^0_{u,k} = \text{Proj}\big(\bar{\mathbf{p}}^0_{u,k}\big) + \mathbf{r}_k. \tag{18}$$

This construction preserves several historically supported preference directions instead of averaging all prototypes into a single condition. It uses only the observed history and learned representations, without accessing the target item. The recorded configuration uses $\tau_c = 0.1$ and $\gamma = 0.2$.

A.3 Editable Intent State and Operation Rules

We distinguish the updater's persistent constraint state from the decoder-facing state S^t_u. Let

$$\Xi^t_u = (P^t_u, N^t_u, \Omega^t_u), \qquad o^t_u = (a^t_u, c^t_u, c^t_{u,\text{src}}, w^t_u), \quad w^t_u \in [0, 1], \tag{19}$$

where P^t_u and N^t_u map catalog category IDs to positive and negative strengths, respectively, and Ω^t_u records the valid operations applied through turn t. The operation fields specify an action, a target category, an optional source category, and a strength. Each category must belong to the validated catalog candidate set, and a source is required for `Refine`. Starting from copies of the previous maps, the deterministic state updater uses the following rules (user and turn indices are omitted inside map operations for readability):

- **Enhance:** set $P[c] = w$ and remove c from N.
- **Suppress:** set $N[c] = w$ and remove c from P.
- **Undo:** remove c from both maps. This deletes the current constraint on c; it neither restores an earlier value nor rolls back a decoder state.
- **Refine:** remove c_{src} from the positive map, set $P[c] = \max\big(\mathrm{get}(P_u^{t-1}, c_{\mathrm{src}}; w), w\big)$, and remove c from the negative map. Other entries remain unchanged.

An invalid or missing parsed operation preserves the prior state in the rollout. Repeated `Enhance` or `Suppress` operations overwrite the strength of the same category rather than accumulating it across turns. Constraints on different categories remain simultaneously active, and conflicts on the same category are resolved by the new operation. The updater does not impose a general ancestor–descendant conflict-resolution policy or a four-entry capacity limit.

The condition builder converts Ξ_u^t and the history-derived initialization S_u^0 into the decoder-facing state in Eq. 7. Each current positive entry produces an active intent condition and branch weight, while all current negative entries are combined into the shared suppression condition $\mathbf{B}_u^{-,t}$. Operations therefore edit preference constraints upstream of decoding and never modify intent slots or denoising states in place.

Table 5: Example of multi-turn state updating. A is a broad category, B an excluded category, and C a refinement of A. Symbolic category IDs are used for illustration; this is not an experimental recommendation case.

Turn	Operation	Positive map	Negative map
0	Initialize	$\varnothing$	$\varnothing$
1	enhance$(A, 0.7)$	$\{A : 0.7\}$	$\varnothing$
2	suppress$(B, 0.8)$	$\{A : 0.7\}$	$\{B : 0.8\}$
3	refine$(A \to C, 0.8)$	$\{C : 0.8\}$	$\{B : 0.8\}$
4	undo(B)	$\{C : 0.8\}$	$\varnothing$

A.4 Conditional Recommendation Training

Following Section 4.3, the conditional recommendation model is trained with the LLM and item SID codebook fixed. User history and language feedback form the input condition, and the next-item SID is the denoising target. The primary loss reconstructs masked SID tokens, the auxiliary intent objective combines semantic reconstruction and diversity:

$$\mathcal{L}_{\mathrm{intent}} = \mathcal{L}_{\mathrm{recon}} + \beta_d \mathcal{L}_{\mathrm{diversity}}, \tag{20}$$

where β_d controls the strength of diversity regularization. The reconstruction term aligns an aggregate of the semantically decoded intent slots $\mathbf{z}_{u,k}^t$ with the target-item representation, grounding the generation conditions in observed preferences. The aggregation uses a training-only soft assignment based on the intent weights $\pi_{u,k}^t$ and slot–target semantic similarity. The diversity term penalizes squared cosine similarity between distinct decoded slot representations to discourage redundant intents. At inference, the model uses the current intent state and its weights $\pi_{u,k}^t$, without access to the target item.

A.5 Inference and List Aggregation

Intent-conditioned branches. At inference, each branch is conditioned on its own positive intent slot through cross-attention at the currently masked target positions, while sharing the same behavioral history and decoder parameters. The negative SID guidance $\mathbf{B}_u^{-,t}$ is shared across all branches and subtracted from their SID logits. Each branch maintains a separate target SID state, and all branches are evaluated in parallel along the batch dimension. The branch budget is configurable according to the number of active positive intents and the available computational resources. In our experiments, we use up to four branches, with four decoding steps.

Candidate scoring and aggregation. Let $\mathcal{C}^t_{u,k}$ be the item candidates mapped from the SIDs generated by branch k, and let $g^t_{u,k}(i)$ be the returned generation score. SIDs without a catalog mapping are discarded. The intent-parallel decoder merges candidates using

$$\begin{aligned} \mathrm{score}^t_u(i) &= \max_{k:\, i \in \mathcal{C}^t_{u,k}} \left[g^t_{u,k}(i) + \log \max(\pi^t_{u,k}, 10^{-8}) \right], \\ \mathcal{R}^t_u &= \mathrm{TopN}_{i \in \cup_k \mathcal{C}^t_{u,k}} \mathrm{score}^t_u(i). \end{aligned} \tag{21}$$

The intent weight adjusts each branch's contribution. If an item appears in multiple branches, only its highest adjusted score is retained. The unique candidate union is then sorted to produce the final list; fewer than N items are returned if that union is too small. Thus, aggregation combines candidate lists after generation and does not average the intent representations before decoding.

Inference procedure. At inference time, ControlRec generates a recommendation list from the current editable intent state in four steps:

1. **Read the current intent state.** We obtain the active positive intents and their normalized weights $\{(\mathbf{z}^t_{u,k}, \pi^t_{u,k})\}_{k=1}^{K_t}$ from the current state. Inactive or suppressed intents are not used as positive generation conditions.
2. **Generate candidates for each active positive intent.** For each active positive intent $\mathbf{z}^t_{u,k}$, the Intent-Guided Diffusion Decoder generates a SID candidate list conditioned on the shared behavioral history and the corresponding intent representation. All active branches use the same decoder parameters and are decoded in parallel.
3. **Map and aggregate candidates.** The generated SIDs are mapped back to catalog items. Unmapped outputs are discarded, and duplicate items produced by different intent branches are combined using the aggregation rule in Eq. 21, with each branch weighted by its intent weight $\pi^t_{u,k}$.
4. **Rank and return the recommendation list.** The resulting unique candidates are sorted by their adjusted scores, and the top N items are returned as the final recommendation list.

A.6 Dialogue Construction

A.6.1 Amazon: Simulated Category Refinement

Evaluating conversational intent control requires explicit feedback about how users want recommendations to change, whereas most datasets, including Amazon datasets, do not record behaviors with the corresponding corrective exchanges. We therefore construct simulated dialogues to supply this missing interaction signal. An LLM uses the ground-truth target's category metadata to express a consistent preference goal in coarse-grained natural language. The target informs the offline simulation, while the recommender receives the resulting feedback without the target's identity. At inference, ControlRec accepts user-authored free-form feedback without access to target metadata; the target-derived trajectories are an evaluation protocol rather than a restriction on its conversational interface. This benchmark therefore measures execution of progressively revealed, target-relevant preferences rather than intent elicitation from real users. Such feedback intentionally provides information about current needs beyond behavioral history, as a user's request would in an interactive setting. All feedback-aware methods receive the same simulated utterances, allowing a controlled comparison of their ability to translate expressed preferences into recommendations.

Feedback construction. Amazon Reviews provides item interaction sequences but does not record users' conversational feedback after receiving recommendations. To evaluate whether a recommender can respond to progressively clarified needs, we construct a standardized three-turn feedback trajectory for each held-out next-item prediction case. The held-out target is used by the offline feedback simulator to identify a target-consistent category path; it is not supplied as an item identity to the evaluated recommenders. The trajectory consists of:

1. **T1: Broad preference.** The simulated user expresses a current need associated with a broad category containing the target item. The utterance translates category information into what the user wants to accomplish. For example: "I currently need something for measurement and inspection."

2. **T2: Category suppression.** The simulated user expresses an exclusion corresponding to an over-exposed sibling category that does not contain the target, while retaining the positive preference established at T1. For example: "I do not want an electronic device."
3. **T3: Preference refinement.** The simulated user clarifies the intended use or usage context, narrowing the broad preference to a finer category containing the target while preserving the exclusion at T2. For example: "More specifically, I need it for household use."

The edits are cumulative, so each turn evaluates recommendation under the preference state established by the preceding turns. The resulting utterances do not disclose the target title, item ID, semantic ID (SID), future interaction, or item-specific attributes. This protocol measures how well a model uses informative, category-level feedback to revise its recommendations; it is not intended to model how users formulate feedback in an unconstrained conversation.

Shared inputs and method-specific interfaces. For each test case, we fix the behavioral history and the complete feedback trajectory before evaluating any method. All dialogue-aware methods receive the same sequence of natural-language utterances at the same turns, with the same held-out target used only for evaluation. In particular, the target item is not used to select, score, or filter candidates at inference time. The tag-reranking baseline explicitly tests how much can be gained from applying the category information as a rule, helping distinguish category-based candidate reduction from the additional benefit of personalized intent-conditioned generation. We quantify how many catalog items remain compatible with the category information expressed at each stage. As shown in Table 6, a T1 broad category contains 2,565 items on average. Even after T3 refinement, the specified fine category contains 1,032 items on average. Applying the T2 suppression constraint as well leaves an average of 984 items (median: 144). Thus, the feedback narrows the recommendation space but generally does not identify a unique product: the model must still rank among many category-compatible items using the user's behavioral history.

Table 6: Number of catalog items compatible with category-level feedback in the Amazon intent-refinement evaluation. The final row additionally excludes items from the suppressed category.

Feedback	Minimum	Median	Mean
T1: Broad	34	599	2,565
T3: Fine	32	151	1,032
T3: Fine + Suppression	2	144	984

A.6.2 ReDial: Natural and Constructed Preference Edits

We construct the evaluation instances from real ReDial conversations, retaining the dialogue context and the associated accepted recommendation targets. Positive preferences are taken directly from the original seeker utterances. The positive task uses original seeker utterances with explicit genre preferences and the preceding dialogue context. Preference cues are identified from the utterance; eligible targets are subsequently recommended movies marked as suggested and liked, whose genres match an expressed preference. The comparison uses 175 common target-level instances. The original request is retained rather than rewritten from the target's metadata.

For negative editing, an LLM generates contextually appropriate exclusion requests that remain compatible with the accepted target. We construct 150 instances from the positive common set by appending a request such as "I do not want comedy movies right now." This creates a fixed test of removing an exposed, unwanted attribute while retaining a compatible target.

We construct 150 multi-intent evaluation instances comprising naturally occurring requests from ReDial, and additional requests generated by a large language model using dialogue context and movie metadata. Each request specifies multiple genre preferences to be satisfied collectively by the recommendation list, without requiring individual items to satisfy every preference. The construction procedure is independent of the evaluated recommenders, and all methods receive identical dialogue contexts, requests, and evaluation targets.

The constructed negative and multi-intent requests are frozen and shared across all compared methods, together with the same dialogue prefixes and evaluation targets. Future recommendation utter-

ances and acceptance labels are not provided as dialogue conditions. Genre-based satisfaction and coverage measure execution of explicit requirements; recommendation accuracy separately measures target recovery.

Control metrics. Let L_e denote the returned top-10 list for evaluation instance e, $G(i)$ the genre set of item i, and Q_e and F_e the requested and forbidden genre sets specified by the benchmark. Let $\mathcal{D}_+$, $\mathcal{D}_-$, and $\mathcal{D}_{\text{multi}}$ denote the positive, negative, and multi-intent evaluation sets, respectively.

Sat.@10 measures the fraction of recommended items matching at least one requested genre, averaged over positive-editing instances:

$$\text{Sat}\,@10 = \frac{1}{|\mathcal{D}_+|} \sum_{e \in \mathcal{D}_+} \frac{1}{|L_e|} \sum_{i \in L_e} \mathbf{1}[G(i) \cap Q_e \neq \varnothing]\,. \tag{22}$$

Viol.@10 measures the fraction of returned items matching at least one forbidden genre, pooled across negative-editing instances:

$$\text{Viol}\,@10 = \frac{\sum_{e \in \mathcal{D}_-} \sum_{i \in L_e} \mathbf{1}[G(i) \cap F_e \neq \varnothing]}{\sum_{e \in \mathcal{D}_-} |L_e|}. \tag{23}$$

All-Req@10 measures the fraction of multi-intent instances for which the recommendation list collectively covers every requested genre:

$$\text{All-Req}\,@10 = \frac{1}{|\mathcal{D}_{\text{multi}}|} \sum_{e \in \mathcal{D}_{\text{multi}}} \mathbf{1}\left[Q_e \subseteq \bigcup_{i \in L_e} G(i)\right]. \tag{24}$$

Different items may satisfy different requests; an individual item need not satisfy all requested genres. Higher Sat.@10 and All-Req@10 and lower Viol.@10 indicate better intent control.

A.7 Implementation details

We optimize the alignment and recommendation modules using AdamW. Semantic alignment follows task-specific training schedules, with cosine learning rates of 1×10^{-5}–3×10^{-4}, 12 epochs per stage with a batch size of 256. The final item-level alignment stage updates the SID-to-text adapter and its conditioning modules for 6 epochs at 5×10^{-5}, keeping the recommendation backbone frozen, we set $\lambda_{\text{sem}} = 0.4$, $\lambda_{\text{mask}} = 0.4$, and $\lambda_{\text{rank}} = 0.2$ in Eq. 14. For Amazon, the intent module is trained for 10 epochs with a learning rate of 8×10^{-4}, and a batch size of 1024. For ReDial, we jointly optimize the intent-conditioning modules and the four-layer diffusion backbone (hidden dimension 256) with a 20-epoch budget, with a learning rate of 2×10^{-4} and a batch size of 512. The item SID tokenizer remains fixed during recommendation training, and we set $\lambda_s = 0.05$ and $\beta_d = 0.05$.

For intent interpretation, we use frozen Qwen2.5-7B-Instruct with greedy decoding. The prompts require the model to extract preferences grounded in the visible dialogue, preserve their positive or negative polarity, and resolve revisions using the latest explicit feedback. Predictions are returned as structured JSON under the prescribed schema, without additional explanations or unsupported preference inference. Reported ControlRec results in the main tables are averaged over three runs with random seeds 2024, 2025, and 2026. Because the reported baselines do not provide matched per-seed outputs, we do not apply cross-method hypothesis tests against their point estimates. Paired-bootstrap confidence intervals are reported only for analyses with paired instance-level outputs, as in Appendix B.3.

B Additional Experiments

B.1 Inference Efficiency

We measure inference time on a single NVIDIA A100 80GB GPU with batch size one and warmed-up resident models. Table 7 reports recommendation accuracy and measured inference time in

Table 7: Recommendation accuracy and inference time on an NVIDIA A100 80GB GPU. For ControlRec in the conversational setting, the reported time is intent generation/decoding and merging.

Dialogue-free recommendation

Method	R@10	N@10	Time (ms)
HSTU	.0450	.0253	1.22
LLaDA-Rec	.0475	.0257	25.36
TIGER	.0446	.0236	94.21
DiffuRec	.0310	.0158	103.19
DreamRec	.0299	.0155	130.26
ControlRec	**.0483**	**.0270**	25.98

Conversational recommendation

Method	Time (ms)
Qwen Ranking Agent	743.97
Chat-REC	1085.11
ChatCRS	1041.08
InteRecAgent	1462.35
ControlRec	578.82 / 27.97

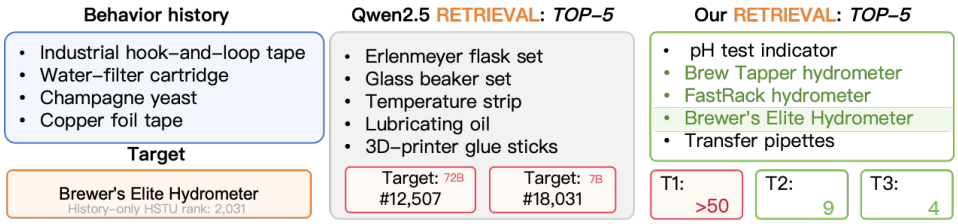

Figure 4: A hard-case example comparing LLM-based agent retrieval and CONTROLREC.

dialogue-free and conversational settings. For conversational agents, we use prepared Top-100 candidates following their respective evaluation protocols. For ControlRec, the conversational time is reported separately for intent generation and intent-conditioned decoding with candidate merging.

In the dialogue-free setting, ControlRec achieves the highest Recall@10 and NDCG@10 among the generative recommenders, with 25.98 ms per recommendation. This is only 0.62 ms (2.4%) slower than LLaDA-Rec while providing higher accuracy.

With conversational feedback, intent generation accounts for 578.82 ms, whereas intent-conditioned decoding and merging take only 27.97 ms. Thus, the main inference cost comes from processing the natural-language feedback rather than from multi-intent SID generation. The measured ControlRec components total 606.79 ms, while the reported conversational agents require 743.97–1462.35 ms under their respective evaluation settings.

B.2 EFFECT OF FEEDBACK ALIGNMENT

To test whether ControlRec benefits from feedback that is relevant to the current user, we compare correctly assigned and shuffled feedback conditions on the same 43,577-user hard subset. For the shuffled setting, we cyclically shift the feedback-derived conditions across users while keeping each user's behavioral history and evaluation target unchanged, using the same shift at every turn to preserve the donor trajectory across turns.

Table 8 shows that correctly assigned feedback yields a consistent gain across turns, with R@10 increasing from 0.01960 at T1 to 0.03438 at T3. In contrast, shuffled conditions remain near the history-only level, reaching only 0.01095 R@10 at T3. At T3, correct feedback improves R@10 and N@10 by 214.0% and 243.3% relative to shuffled feedback, respectively. This result shows that the gains depend on feedback being aligned with the current user's preference state rather than simply adding an extra conditioning signal.

We also provide a test case in Figure 4 that illustrates this hard-case recovery mechanism. History-only HSTU ranks the target hydrometer at 2,031 because the recent interactions mix brewing tools with unrelated industrial items. Qwen 2.5-VL 7B/72B retrievers latch onto laboratory surface cues and push the target to 18,031/12,507. After T1–T3, CONTROLREC returns a hydrometer-dominated slate and lifts the target to rank 4. This example illustrates how successive intent edits can redirect personalized generation toward the user's expressed needs.

Table 8: Effect of correctly assigned and shuffled feedback conditions on the Amazon hard-case subset.

Condition	T1		T2		T3	
	R@10	N@10	R@10	N@10	R@10	N@10
Correct	**.01960**	**.00959**	**.02761**	**.01392**	**.03438**	**.01726**
Shuffled	.01177	.00563	.01102	.00502	.01095	.00503

B.3 Effect of Multi-Intent Decoding

Experimental setup. We evaluate multi-intent decoding on 768 controlled examples, balanced across two, three, and four intents (256 each) and across Chinese and English. Future-window categories and frequencies are verbalized as feedback without exposing item IDs, titles, SIDs, or rankings to the parser. Each example is evaluated on a 50-item candidate pool. This controlled setting isolates the effect of intent decoding and aggregation on multi-preference recommendation.

Decoding strategies. Figure 5 compares six configurations. Top-1 retains only the highest-weight intent, while Joint conditions a single generation process on all active intents. Score Sum and Prob. Slate generate separate branches and aggregate their outputs using score-based and probability-aware strategies, respectively. Prob. Slate uses the branch aggregation rule in Eq. 21. Oracle and Shuffled provide reference conditions with correct and mismatched intent conditions. We report R@10 and N@10 for the target item, Set-N@10 for all future relevant items, and All-Intent@10 for intent coverage.

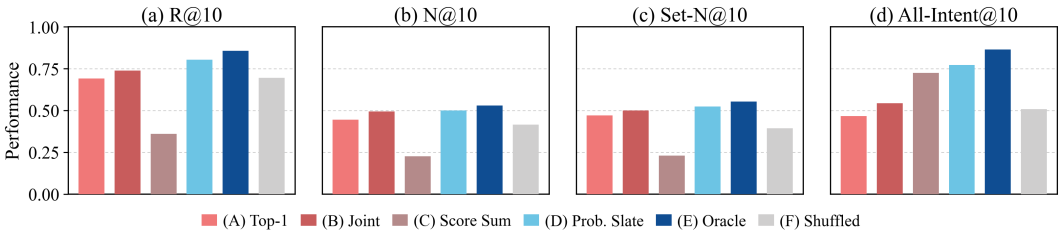

Figure 5: Comparison of multi-intent decoding and aggregation strategies on the controlled candidate-pool evaluation.

Results and analysis. Prob. Slate improves R@10 by 0.0651 and Set-N@10 by 0.0232 over Joint, with paired-bootstrap 95% confidence intervals of [0.0416, 0.0885] and [0.0133, 0.0328], respectively. Its higher All-Intent@10 further indicates better preservation of concurrent preferences. In contrast, Top-1 provides limited intent coverage, while simple branch score summation yields weaker target ranking. These results highlight the importance of intent-factorized generation and the subsequent aggregation strategy.

The coverage advantage becomes more pronounced as requests contain more intents. With two, three, and four intents, Joint achieves All-Intent@10 of 0.7656, 0.4922, and 0.3750, whereas Prob. Slate achieves 0.9297, 0.6914, and 0.6992, respectively. Shuffled conditions also yield weaker performance, supporting the importance of correctly matched intent signals. The single-target N@10 difference between Prob. Slate and Joint is not statistically significant; the clearest benefits are in relevant-item recovery and simultaneous intent coverage.

To examine how performance changes with the number of intents, Table 9 further breaks down All-Intent@10 by intent count. Prob. Slate outperforms joint conditioning across all three settings. As the number of intents increases from two to four, joint conditioning declines from 76.56% to 37.50%, whereas branch-based decoding maintains 69.92% coverage with four intents. This yields the largest absolute improvement, 32.42 percentage points, suggesting that separate generation branches help preserve concurrent preferences as requests become more complex. Oracle

Table 9: All-Intent@10 (%) by the number of intents. Prob. Slate uses intent-factorized decoding and probability-aware aggregation, while Oracle uses ground-truth intent conditions as a reference.

# Intents	Joint	Prob. Slate	Oracle
2	76.56	**92.97**	94.14
3	49.22	**69.14**	86.33
4	37.50	**69.92**	78.91

branch decoding achieves higher coverage throughout, indicating additional room for improvement in the quality of the intent conditions.

B.4 Reliability of Language-to-Intent Parsing

We evaluate the language-to-intent interface on 1,506 unique bilingual feedback trajectories comprising 4,518 turns. Feedback uses natural leaf-level category expressions, and evaluation follows a stateful rollout in which each turn uses the state produced by previous predictions rather than ground-truth states.

We compare contextual parsing with exact-name anchoring, which overwrites the predicted category with a matched leaf-name anchor when available. Table 10 reports category accuracy, refinement-turn accuracy, and whole-trajectory accuracy. Contextual parsing achieves 97.26% category accuracy and 93.29% trajectory accuracy, compared with 93.12% and 81.21% for exact-name anchoring.

The largest gap occurs on refinement turns, where contextual parsing reaches 95.55% accuracy versus 82.40% for exact-name anchoring. This difference arises in part from overlapping hierarchical names, such as "Tile Epoxy Adhesives" and its parent category "Epoxy Adhesives," where literal anchoring can replace a valid fine-grained prediction with its parent. These results show that contextual parsing better preserves category-level intent during multi-turn state updates.

Table 10: Language-to-intent parsing accuracy (%) on 1,506 bilingual trajectories. Refine denotes category accuracy on refinement turns.

Parser	Category	Refine	Trajectory
Contextual parsing	**97.26**	**95.55**	**93.29**
Exact-name anchoring	93.12	82.40	81.21

B.5 Comparison with Matched Tag Reranking

We compare history-only candidates from LLaDA-Rec with feedback-conditioned candidates from ControlRec under identical tag-reranking rules, feedback constraints, and SID-matching criteria on the same 43,577 test users. Table 11 reports results before and after applying tag reranking to both methods, examining whether ControlRec's advantage persists under shared rule-based postprocessing.

ControlRec incorporates user intent directly into candidate generation. Its LLM-based intent interpreter converts conversational feedback into explicit preference updates, while language–SID semantic pretraining connects these updates to the item-generation space. Together with behavioral history, these conditions guide generation toward items consistent with personalized preferences and current needs. After identical tag reranking, ControlRec still exceeds LLaDA-Rec by 167.16% in R@10 and 264.24% in N@10 at T3. This advantage supports the benefit of feedback-conditioned generation, rather than gains attributable solely to tag-based postprocessing. Rule-based reranking can reorder an existing candidate pool but cannot introduce relevant items absent from it; conditioning generation on user intent allows ControlRec to change the candidate pool itself.

Table 11: Recommendation results with and without matched tag reranking.

Method	T1		T2		T3	
	R@10	N@10	R@10	N@10	R@10	N@10
LLaDA-Rec	.01157	.00550	.01157	.00550	.01157	.00550
+ Tag Reranking	.01928	.00931	.02139	.01046	.02302	.01141
ControlRec	.01960	.00959	.02761	.01392	.03438	.01726
+ Tag Reranking	**.03575**	**.02225**	**.05083**	**.03320**	**.06150**	**.04156**

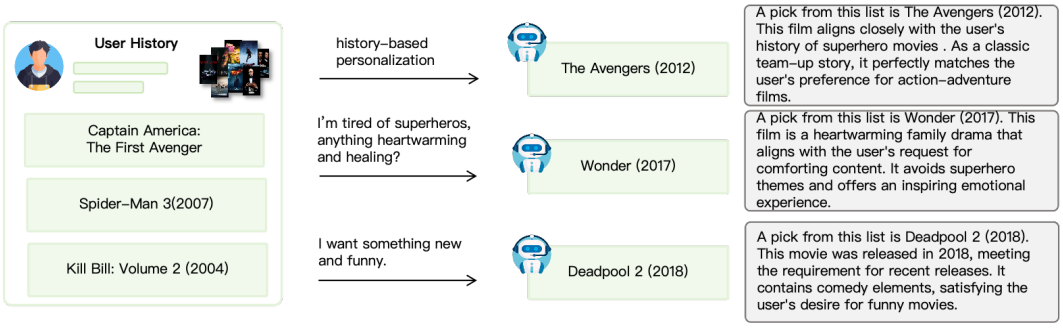

Figure 6: Free-form semantic guidance under the same user history. Independent requests lead to different featured recommendations.

B.6 Qualitative Case Study

We conducted a real case exploratory user study , using the user history and dialogue intents. Figure 6 compares three independent recommendation requests under the same viewing history: *Captain America: The First Avenger*, *Spider-Man 3*, and *Kill Bill: Volume 2*. Without explicit feedback, the system recommends *The Avengers*, consistent with the action and superhero themes in the history. When the user expresses fatigue with superhero movies and requests something "heartwarming and comforting," the generated list includes *Wonder* as a featured recommendation. Under a separate request for "newer, funny movies," the system instead features *Deadpool 2*. These examples illustrate how current needs can redirect recommendation while behavioral history remains available for personalization.

The advantage of ControlRec extends beyond matching requests to predefined category labels. Language–SID semantic pretraining connects item identities with textual content and metadata, providing a semantic basis for intent-guided generation. In our demo, free-form preferences are matched against item descriptions, including titles and plot summaries, and the resulting semantic relevance weights are aggregated into SID-level logit guidance. A request for a "heartwarming" movie can thus favor items with semantically relevant descriptions even when no corresponding category label exists. This mechanism supports preferences that cannot be expressed through predefined label matching alone and injects their semantic relevance into the generation process. We provide an interactive demo at `https://control-demo-mimphzrsxk8bygcajafwxm.streamlit.app/`, where users can specify their own viewing histories and freely express their preferences to explore intent-guided recommendation.