BEYOND THE BEAM: CONSTRUCTIVE REPAIR AND CANDIDATE COMPLETION FOR GENERATIVE RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative recommenders retrieve items by generating identifiers, but a valid identifier can remain outside the beam after catalog expansion. This raises two connected questions: which failures can identifier assignment repair, and how should retrieval proceed beyond the initial beam? We characterize assignment repair with a fixed generator and retained old identifiers. Output-invariance certificates identify failures shared by all admissible assignments. Under a common effective prefix, coupled support and ranking constraints give the exact feasible interval of new-item counts for target recovery. Building on this characterization, Beyond the Beam (BB) obtains minimum-replacement repairs through an integral flow formulation, selects a shared map and adapts the generator. At inference, generative likelihood and collaborative evidence define one score for ranking, candidate priority and stopping. Retained prefix bounds guide candidate completion and certify its global Top-K when the stopping condition is met. Exhaustive finite-catalog evaluation confirms construction in every feasible case. Across three Amazon Reviews categories and three random seeds, the full T5 procedure improves mean Recall@10 by 15.5–46.3% and NDCG@10 by 15.2–44.4% over the best-performing evaluated generative baseline for each dataset and metric. Matched controls show that shared construction and adaptation improve new-target ranking and certification efficiency on Beauty and Toys. Combined scoring and candidate completion improve NDCG@10 across all three datasets with both T5 and decoder-only LC-Rec.

1 INTRODUCTION

Generative recommenders retrieve items by generating compact identifiers (Rajput et al., 2023). After catalog expansion, a new item can have a valid identifier and strong collaborative support yet be omitted by beam search. Reranking cannot return an item outside its candidate pool. This failure motivates two questions: when can changing identifiers repair retrieval while preserving old codes, and how can inference evaluate promising items beyond the initial beam? Appendix B.1 illustrates both questions with a concrete example.

Existing approaches update the index, tokenizer or model. Joint tree/model learning optimizes item-to-leaf matching (Zhu et al., 2019); DREAM uses multi-context evidence to choose candidate identifiers (Guan et al., 2026); DACT incorporates collaborative information into tokenization (Feng et al., 2026). These procedures choose an assignment and measure its performance. The complementary question concerns the entire admissible family: which failures persist across every assignment, and which admit a repair while retaining old identifiers and recommendation eligibility?

A target's identifier score alone does not determine repair feasibility. An occupied leaf activates prefixes that compete for a finite beam. Supporting the target's path can therefore displace other paths, while filling the catalog can introduce higher-ranked terminal competitors. The challenge is to characterize which catalogs jointly satisfy these support and ranking requirements while returning the target.

After a common effective prefix, we characterize exactly which catalogs recover a specified target. Prefix support sets a lower bound on new-item counts; ranking competition and catalog capacity set

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.

an upper bound. Every feasible count admits an explicit repair. A complementary output-invariance certificate identifies queries whose ordered predictions remain unchanged across admissible assignments, yielding bounds on population improvement.

Building on this feasible family, we develop Beyond the Beam (BB), connecting constructive repair with bound-guided candidate completion. For each feasible state and target, minimum-replacement repair retains as much of the current leaf occupancy as the constraints allow. Disjoint support groups give an integral minimum-cost-flow solution. We turn local repairs into shared-map proposals, compare them across re-encoded training histories, and adapt the generator to the selected map.

The shared map is learned across training contexts, while candidate access remains query-dependent. At inference, collaborative evidence can favor items outside the generator's initial beam. We therefore combine generator likelihood with an item-level correction and evaluate additional candidates under this common score. Adding the known correction to retained prefix scores yields bounds for candidate priority and global Top-K certification.

Our contributions connect repair feasibility and minimum leaf replacement with candidate completion under the combined score. Exhaustive finite catalogs and direct model decoding test the constructive guarantees. Three-seed T5 controls show that shared construction and adaptation improve new-target ranking and certification efficiency on Beauty and Toys. Combined scoring and candidate completion improve NDCG@10 across all three datasets with both T5 and decoder-only LC-Rec.

2 Problem Formulation

Given a user's history h, a generative recommender predicts an item through its semantic identifier, a length-L path in $\mathcal{Z} = \Sigma_1 \times \cdots \times \Sigma_L$. An injective assignment $c : \mathcal{I} \to \mathcal{Z}$ maps the current items to the occupied catalog $A = c(\mathcal{I})$. Beam search maintains W live prefixes and returns a ranked item list. We study when assigning identifiers to new items can recover a target in this list while preserving existing item identifiers.

Assignment repair. Let c_O be the retained assignment of old items $\mathcal{I}_{\text{old}}$, with occupied paths $O = c_O(\mathcal{I}_{\text{old}})$. A legal expansion preserves their identities and recommendation eligibility:

$$O \subseteq A \subseteq \mathcal{Z}, \qquad c(i) = c_O(i) \quad (i \in \mathcal{I}_{\text{old}}). \tag{1}$$

The m new items $\mathcal{I}_{\text{new}} = \mathcal{I} \setminus \mathcal{I}_{\text{old}}$ receive distinct paths in $A \setminus O$. A repair is an admissible assignment under which a specified target appears in Top-K. The theoretical question is whether such an assignment exists. For recommendation, we use repairs from training contexts to propose a shared map and evaluate it over held-out queries. The map used for a test query does not depend on its target.

To characterize assignment repair, fix the generator, encoded history, vocabulary, beam width and termination rule. Old-only histories retain the same encoding as new assignments vary; mixed histories require re-encoding when evaluating the shared map. At depth d, the effective decoder state S_{d-1} contains its live prefixes, accumulated scores, inputs and caches. Before masking invalid catalog children, its full-vocabulary expansions are

$$\mathcal{E}_d(S_{d-1}) = \{(p, a) : p \in S_{d-1}^{\text{live}}, a \in \Sigma_d\}, \qquad q(p, a) = s(p) + r_\theta(h, S_{d-1}; p, a). \tag{2}$$

Legal children keep these scores. Deterministic state transitions and a fixed tie order make comparisons across catalogs well defined. The assignment-family results assume query-separable execution and verify relevant numerical gaps (Appendix D.7).

Scoring and candidate completion. After the shared map and adapted model are fixed, let $\text{enc}_c(h)$ denote the encoded history. A standard autoregressive generator assigns complete-code score $s_\theta(i \mid h) = \sum_{t=1}^{L} \log p_\theta(c_t(i) \mid \text{enc}_c(h), c_{<t}(i))$. We augment this score with an item-level predictor. The inference task is to retrieve the Top-K under the combined score. Collaborative evidence may favor items outside the initial beam, so candidate completion evaluates additional items while retaining the chosen map. Its inputs are the history, current catalog and learned scores; the target is used only for evaluation.

We measure Recall and NDCG on the full test population. A target that has not yet entered the current catalog contributes a miss. The candidate-completion budget controls the number of additional complete-item likelihoods evaluated, separately from the initial beam width and the cost of the item-level predictor.

3 Assignment Limits and Constructive Guarantees

We first establish when identifier assignment cannot change retrieval, then characterize when it can recover a target. Both results hold the generator and encoded input fixed, isolating the effect of catalog occupancy. We study Top-K retrieval within a width-W beam, with $K \leq W$.

3.1 Limits of assignment-only repair

Suppose the unmasked Top-W extensions at every layer of a reference run are finite, live, old-supported prefixes with strict score gaps. Under the scoring and termination premises of Section 2, every admissible catalog retains these extensions and their order. The effective state is therefore preserved layer by layer, and all assignments return the same ordered old-item list (Proposition 1). A failed check leaves repair unresolved.

This certificate bounds population metrics normalized to the interval $[0, 1]$. If C is a certified subset of an N-query population and query $u \in C$ has metric upper bound c_u, then, over legal assignments c, $\sup_c N^{-1} \sum_u m_u(c) \leq (\sum_{u \in C} c_u + N - |C|)/N$. Unresolved queries receive the optimistic metric value one. An observed update exceeding this bound on the same population cannot be matched by any legal assignment of the fixed parent model. The bound also applies to a single shared assignment, even when different queries require incompatible repairs (Appendix A).

3.2 Exact feasibility of assignment repair

Assume all admissible catalogs share a valid effective beam of W old-supported prefixes at depth $L-2$. Successful old-prefix checks establish this condition. At the next depth, old-supported prefixes are mandatory, while optional prefixes can be enabled by occupying a descendant. Enumerate the ordered Top-W states S induced by the optional prefixes that can reach the mandatory cutoff. For a state S, exclude descendants of unselected optional prefixes above its cutoff, leaving maximal catalog Z_S. Each selected nonold prefix needs one occupied descendant; these disjoint support groups form $\mathcal{G}_S$. Write $U_S = Z_S \setminus O$, $C_S = |U_S|$ and $r_S = |\mathcal{G}_S|$.

Theorem 1 (Catalog feasibility and constructive repair)**.** *A catalog reaches S exactly when $O \subseteq A \subseteq Z_S$ and $A \cap G \neq \varnothing$ for every $G \in \mathcal{G}_S$. For a new terminal child y of S, let $R_S(y)$ be its full one-based rank. Let $a_S(y)$ count its old outrankers and $g_S(y)$ count the support groups whose every leaf precedes y. Let $\delta_S(y)$ be zero if y belongs to a support group and one otherwise. An m-item repair returning y in Top-K exists exactly when*

$$a_S(y) + g_S(y) < K, \qquad r_S + \delta_S(y) \leq m \leq C_S - \max\{0, R_S(y) - K\}. \tag{3}$$

A satisfying case yields an explicit legal assignment.

The interval exposes two competing requirements. Enough new leaves must be occupied to support the selected prefixes and the target. Enough higher-ranked leaves must remain unoccupied to keep the target in Top-K. Thus catalog capacity and ranking competition impose an upper count bound alongside the lower bound from prefix support.

The characterization is constructive: select the target, support uncovered groups with non-outranking leaves whenever possible, and fill remaining capacity with non-outrankers before using higher-ranked leaves. Enumerating the candidate states and target leaves decides repair feasibility within the common-prefix setting. Section 4 uses this feasible family as the constraint set for minimizing leaf replacement while recovering a specified target in a chosen state. Appendix A gives the proofs, critical-state search and the worst-case lower bound for complete-state score-oracle access.

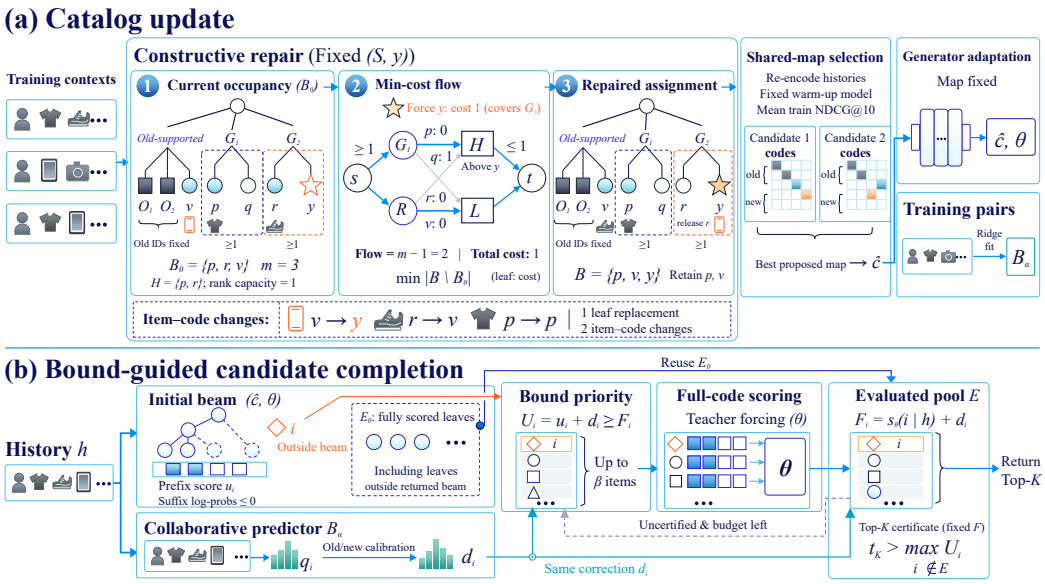


Figure 1: Overview of BB. (a) Minimum-cost flow repairs fixed (S, y) while preserving old identifiers; trees omit unchanged branches and additional leaves. Replacing r with y changes one occupied leaf and two item codes. Re-encoded proposals are selected by training NDCG@10 under the fixed warm-up model; the generator then adapts with the map fixed. (b) The correction d_i enters both U_i and F_i. Filled/outlined tokens denote scored prefixes/unevaluated suffixes. Reuse includes all scored leaves E_0, even outside the returned beam; the highlighted item initially lies outside E_0. Priority evaluates only unscored items. With $|E| \geq K$, (11) certifies global Top-K under fixed F; budget exhaustion can leave it uncertified.

4 Constructive Repair and Bound-Guided Candidate Completion

BB connects catalog updating with retrieval for each query (Figure 1). We solve the repair constraints of Theorem 1 with minimum leaf replacement, then evaluate shared-map proposals on re-encoded histories and adapt the generator. With that map fixed, the combined score guides candidate completion and stopping.

4.1 Constructive Assignment and Generator Adaptation

Repair jointly constrains prefix support, ranking competition and catalog size. Among feasible occupancies, the objective preserves as much of the current occupancy as possible. Let B_0 be the current m-leaf new-item occupancy and fix a feasible state–target pair (S, y). Let $H_S(y)$ contain optional leaves ranked above y and $a_S(y)$ count its mandatory old outrankers. Within the compatible universe U_S of Theorem 1, we solve

$$\begin{aligned} \min_{B \subseteq U_S} \quad & |B \setminus B_0| \\ \text{s.t.} \quad & |B| = m, \quad y \in B, \quad B \cap G \neq \varnothing \ (G \in \mathcal{G}_S), \\ & |B \cap H_S(y)| \leq K - 1 - a_S(y). \end{aligned} \tag{4}$$

The universe excludes leaves that would displace the selected beam state. The count constraint preserves catalog size, the support constraints activate every required prefix, and the final constraint reserves a Top-K position for the target. The solution specifies occupied leaves; the next step assigns item identities and evaluates the resulting shared map.

Reduction to an integral flow. Disjoint support groups let us aggregate leaves by support membership, rank relative to y, and retention cost. The resulting minimum-cost-flow network enforces all three constraints jointly and minimizes leaf replacement for the specified state and target, avoiding

enumeration of the m-leaf subsets of U_S. Its integral solution and reconstruction into a repair are detailed in Appendix B.

From local solutions to a shared map. A local solution specifies leaf occupancy for one decoder state and target. To obtain item mappings, we pin each proposal's target identity and retain compatible original identities. We combine compatible proposals into a finite set $\mathcal{C}_{\text{prop}}$ of modified candidate maps c. Each map re-encodes histories, so we assess its effect across training contexts under the fixed warm-up generator. Let $r_c(h, i)$ be the target's one-based rank in beam output after re-encoding history h under c. We evaluate candidates on a subset $\mathcal{D}_{\text{sel}}$ of update-training examples disjoint from the proposal-generation contexts. For a nonempty candidate set, we select the shared assignment by mean NDCG@10:

$$\widehat{c} \in \arg\max_{c \in \mathcal{C}_{\text{prop}}} \frac{1}{|\mathcal{D}_{\text{sel}}|} \sum_{(h,i) \in \mathcal{D}_{\text{sel}}} \frac{\mathbf{1}\{r_c(h, i) \le 10\}}{\log_2(1 + r_c(h, i))}. \tag{5}$$

Unreturned targets have rank ∞ and contribute zero. Ties use Recall@10, fewer changed identities, then deterministic mapping order. If the candidate set is empty, we retain the original map. Candidate budgets and composition details are given in Appendix B.

Learning under the constructed assignment. The selected map $\widehat{c}$ fixes history encodings and target identifiers throughout generator adaptation. For update-training pairs $(h, i) \in \mathcal{D}_{\text{upd}}$, write $z = \widehat{c}(i)$ and minimize

$$\mathcal{L}_{\text{adapt}}(\theta; \widehat{c}) = -\mathbb{E}_{(h,i) \sim \mathcal{D}_{\text{upd}}} \left[\frac{w(h, i)}{L} \sum_{\ell=1}^{L} \log p_\theta(z_\ell \mid \text{enc}_{\widehat{c}}(h), z_{<\ell}) \right]. \tag{6}$$

Here $w(h, i)$ is the example weight, normalized to mean one. Adaptation starts from the common warm-up model; Appendices B and F specify the weights, trainable parameters and validation checkpoint selection.

4.2 Catalog-wide scoring for candidate completion

With the map and adapted generator fixed, the candidate beam depends on the query history. Collaborative evidence can favor items outside this likelihood-based beam. A catalog-wide predictor provides item scores before identifier evaluation, supplying the correction used for ranking and candidate completion.

We fit a supervised ridge predictor B_α from normalized, recency-weighted history vectors to one-hot next items, using only available training examples. With these inputs and targets stacked as X and Y, the fit minimizes $\|Y - XB\|_F^2 + \alpha\|B\|_F^2$ over B. Linear item-to-item prediction is an established component of recommendation (Steck, 2019); the closed-form solution and parameter grids are given in Appendix B.4. For history h, write $q_i = \max\{(x_h B_\alpha)_i, 10^{-8}\}$.

The amount of training evidence differs between old and newly admitted items. We therefore calibrate the new-item contribution around the uniform reference $\mu = -\log|\mathcal{I}|$. For an item with identifier $z = \widehat{c}(i)$, the correction and final score are

$$d_i = \lambda \log q_i + \mathbf{1}\{i \in \mathcal{I}_{\text{new}}\}\big[\lambda(\gamma - 1)(\log q_i - \mu) + b\big], \tag{7}$$

$$F(i \mid h) = \sum_{\ell=1}^{L} \log p_\theta(z_\ell \mid \text{enc}_{\widehat{c}}(h), z_{<\ell}) + d_i. \tag{8}$$

The summed term is the adapted generator score $s_\theta(i \mid h)$. Here λ weights collaborative evidence, γ rescales variation among new-item corrections around the reference, and b shifts new-item scores relative to old items. The choices $(\gamma, b) = (1, 0)$ and $(0, 0)$ recover the global correction and the uniform new-item reference, respectively. Validation selects the parameters for each dataset and retriever run. Because d_i is known before full identifier evaluation, it can also enter an upper bound on $F(i \mid h)$. Candidate completion uses this bound to prioritize evaluations under the same score used for final ranking.

4.3 BOUND-GUIDED CANDIDATE COMPLETION

During catalog-constrained beam search, retain each item's longest evaluated prefix, with length ℓ_i and cumulative log probability u_i. For $z = \widehat{c}(i)$, full-vocabulary log probabilities without length normalization give

$$\begin{aligned} U_i &:= u_i + d_i = \sum_{j=1}^{\ell_i} \log p_\theta(z_j \mid \mathrm{enc}_{\widehat{c}}(h), z_{<j}) + d_i, \\ U_i - F(i \mid h) &= -\sum_{j=\ell_i+1}^{L} \log p_\theta(z_j \mid \mathrm{enc}_{\widehat{c}}(h), z_{<j}) \geq 0. \end{aligned} \tag{9}$$

The unevaluated suffix contributes only nonpositive terms. Thus U_i bounds the final score even after the path leaves the beam, allowing pruned identifiers to remain eligible for completion. An empty prefix has $\ell_i = 0$ and $u_i = 0$.

Budgeted candidate selection. The initial evaluated set E_0 contains every complete catalog leaf scored in the terminal beam expansion, including leaves outside the returned beam. At round t, let n_t count additional full evaluations, with $n_0 = 0$. For batch size β and total budget $B_{\max}$, set $b_t = \min\{\beta, B_{\max} - n_t, |\mathcal{I} \setminus E_t|\}$. We select the batch with the largest total upper bound, evaluate its complete identifier likelihoods by teacher forcing, and update

$$\begin{aligned} Q_t &\in \operatorname*{arg\,max}_{\substack{Q \subseteq \mathcal{I} \setminus E_t \\ |Q| = b_t}} \sum_{i \in Q} U_i, \\ E_{t+1} &= E_t \cup Q_t, \qquad n_{t+1} = n_t + |Q_t|. \end{aligned} \tag{10}$$

Sorting the remaining bounds solves this selection problem, with item order breaking ties.

Certified stopping. For an evaluated set E containing at least K items, let t_K be its Kth largest score. Completion stops with a global Top-K certificate when

$$t_K > \max_{i \notin E} U_i. \tag{11}$$

Every unevaluated item then scores below the Kth evaluated item; the maximum over an empty remaining set is $-\infty$. If the budget ends first, the procedure returns the highest-scoring evaluated items without that certificate. The rule applies optimal-stopping principles for monotone prefix scores (Huang et al., 2017; Meister et al., 2020) to the corrected catalog score. Appendix B.6 gives the certificate and the nonincreasing certification-gap argument. Algorithm 1 summarizes training and inference.

Algorithm 1 Minimum-replacement repair and bound-guided completion

Require: Training pairs, catalog, warm-up model; width W, cutoff K, batch β, budget $B_{\max}$

Solve (4) for sampled state–target proposals by integral flow
Pin target identities and compose compatible proposals into shared maps
Select $\widehat{c}$ by (5), retaining the original map if $\mathcal{C}_{\mathrm{prop}} = \varnothing$
Adapt the generator with $\widehat{c}$ fixed using (6)
Fit the next-item predictor; select scoring parameters on validation
for each recommendation history h **do**
    Run catalog beam search; retain prefix bounds u_i and complete leaves E_0
    Compute d_i, U_i for all items and F_i for $i \in E_0$; set $n_0 \leftarrow 0$, $t \leftarrow 0$
    while (11) fails for E_t and $n_t < B_{\max}$ **do**
        Select Q_t by (10) and evaluate F_i for $i \in Q_t$
        Update E_{t+1}, n_{t+1} by (10); set $t \leftarrow t + 1$
    Return the K highest-scoring items in E_t and whether (11) holds

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Data and task. We use Amazon 2014 Beauty, Tools, and Toys and Games (He & McAuley, 2016). Beauty and Toys expand their catalogs from the 60th to the 80th event-time percentile; Tools uses DACT's public period-0.8 sequences. All methods within a domain share the examples and available catalog. Beauty/Tools/Toys provide 7,335/5,360/6,266 test queries. Future-catalog targets remain misses; *Primary* denotes current-catalog new targets with entirely old-item histories. Appendix C gives preprocessing and training coverage.

Comparisons. We adapt JTM matching (Zhu et al., 2019) and DREAM voting (Guan et al., 2026) to the shared T5/candidate framework. DACT (Feng et al., 2026), Reformer (Shi et al., 2025) and five tokenizer/generative recommendation model (GRM) update strategies share a tokenizer-compatible parent. All methods are validation-tuned from original-paper or official settings; BB selects adaptation and scoring settings per dataset and seed. Appendix C specifies initialization and training budgets. We also evaluate decoder-only LC-Rec (Zheng et al., 2024; Feng et al., 2026) on the same datasets (Appendix F).

Evaluation and selection. T5 uses four-token identifiers and beam width 40. Completion reuses fully scored leaves, returns 20 items and evaluates at most 80 additional items in batches of 20. Validation Recall and NDCG at 10 and 20 jointly guide tuning, with one configuration per run for all four metrics. The main comparison reports means and sample standard deviations over seeds 17, 42 and 2027 for every method on all three datasets. Matched ablations inherit each BB run's settings. *Without construction* uses the paired static map and adapted checkpoint, distinct from the separately tuned *Static assignment* baseline. The linear predictor is shared across retriever seeds. Table 3 lists the nine main-result configurations.

5.2 VALIDATION OF CONSTRUCTIVE REPAIR AND GUARANTEES

Across 96 scorers and 221,184 exhaustively decoded catalogs, full construction solves all 1,816 feasible count/cutoff conditions, versus 1,708 for reference-state construction. High-score allocation solves all 882 prefix-local cases but only 733 of 934 complete-state cases. Removing support or either count bound produces false feasibility decisions; the full criterion makes none. Thus each constraint is necessary in this finite design, and searching beyond the reference state recovers additional cases.

Direct Beauty decoding verifies 27 repairs, including 22 requiring new-prefix support; simpler allocations match these counts. Population-bound experiments establish two Beauty separations across three parents; all six Tools comparisons remain unresolved. Appendix D gives the full feasibility and population accounting.

5.3 MAIN RECOMMENDATION COMPARISON

With T5 and decoder-only LC-Rec, BB leads Recall@10 and NDCG@10 on all three datasets (Table 1; Appendix F). On T5, changing the tokenizer alone reduces accuracy; adapting the retriever recovers much of it. Among standard strategies, retraining is strongest on Beauty; retaining the tokenizer and fine-tuning the retriever is strongest on Tools and Toys.

5.4 COMPONENT EFFECTS ON RANKING AND CERTIFICATION

Table 2 and Figure 2 report matched ranking, cohort and certification effects.

Generative and collaborative evidence. Combining generative and collaborative evidence improves NDCG@10 over either alone on all datasets. Calibration further improves these means by adjusting collaborative evidence for old and new items (Figure 2a).

Shared construction and adaptation. Relative to the paired without-construction branch, shared construction and adaptation improve new-target NDCG@10 by 3.82/9.56% on Beauty/Toys, with

Table 1: T5 recommendation under catalog updates: Recall and NDCG at 10 (%), mean ± sample s.d. over seeds 17/42/2027 for all methods; bold marks the highest mean. In tokenizer/GRM pairs, FT denotes fine-tuning and RT retraining.

	Beauty		Tools		Toys	
Method	R@10	N@10	R@10	N@10	R@10	N@10
Frozen model	1.636±0.098	0.775±0.084	2.077±0.075	1.074±0.043	1.059±0.066	0.532±0.003
Static assignment	2.122±0.127	1.006±0.105	3.464±0.127	1.913±0.065	1.612±0.188	0.747±0.090
Frozen/Frozen	1.445±0.024	0.734±0.038	2.475±0.106	1.390±0.056	1.107±0.176	0.546±0.068
FT/Frozen	0.345±0.008	0.156±0.004	0.336±0.000	0.166±0.021	0.681±0.082	0.373±0.035
Frozen/FT	1.822±0.048	0.914±0.021	4.291±0.226	2.325±0.128	1.468±0.120	0.743±0.059
FT/FT	2.086±0.076	0.998±0.067	4.080±0.118	2.163±0.085	1.426±0.088	0.704±0.052
FT/RT	2.200±0.180	1.058±0.062	2.730±0.324	1.467±0.127	1.468±0.100	0.714±0.064
JTM matching (adapted)	2.059±0.165	0.984±0.116	3.302±0.202	1.799±0.190	1.623±0.060	0.735±0.024
DREAM voting (adapted)	2.154±0.165	1.022±0.122	3.470±0.135	1.915±0.067	1.325±0.032	0.614±0.012
DACT (2026)	2.095±0.200	1.063±0.068	4.142±0.391	2.260±0.214	1.505±0.066	0.762±0.047
Reformer (2025)	2.063±0.143	1.015±0.072	4.210±0.159	2.436±0.140	1.623±0.202	0.809±0.109
BB (Ours)	**3.217±0.072**	**1.535±0.047**	**4.956±0.138**	**2.806±0.053**	**2.154±0.064**	**1.027±0.063**

Table 2: Matched T5 component controls at cutoff 10: Recall and NDCG (%), mean ± sample s.d. over seeds 17/42/2027. Each removal inherits its corresponding full-method settings; BB uses the runs in Table 1. Without completion reranks the returned beam; initial-pool reranking uses all fully scored leaves E_0 with no additional evaluations.

	Beauty		Tools		Toys	
Variant	R@10	N@10	R@10	N@10	R@10	N@10
Generator only	2.336±0.202	1.115±0.105	3.408±0.151	1.850±0.129	1.649±0.103	0.790±0.072
Without collaborative score	2.336±0.202	1.115±0.105	3.414±0.153	1.852±0.129	1.649±0.103	0.790±0.071
Without completion	2.872±0.163	1.407±0.076	4.291±0.067	2.558±0.016	2.016±0.040	0.975±0.049
Initial-pool reranking	2.881±0.142	1.416±0.070	4.291±0.067	2.558±0.016	2.080±0.024	1.003±0.032
Without construction	3.231±0.189	1.518±0.081	4.963±0.131	2.810±0.054	2.096±0.109	1.023±0.083
Item correction only	2.449±0.042	1.156±0.026	3.197±0.298	1.790±0.281	1.282±0.009	0.639±0.003
Collaborative priority	3.222±0.075	1.537±0.048	4.820±0.163	2.760±0.079	2.197±0.079	1.042±0.060
Without calibration	2.568±0.028	1.206±0.016	4.434±0.152	2.406±0.142	2.032±0.024	0.968±0.062
BB (Ours)	3.217±0.072	1.535±0.047	4.956±0.138	2.806±0.053	2.154±0.064	1.027±0.063

larger Primary gains of 13.95/19.42%. They also raise numerical Top-20 certification by 3.54/9.87 percentage points and reduce additional evaluations by 2.86/7.90% (Figure 2; Appendix E.3). Old-target losses partly offset the ranking gains, leaving overall NDCG@10 improvements of 1.13/0.37%; Beauty Recall@10 is slightly lower.

Tools's 133 new-target training contexts are far fewer than Beauty's 2,993 or Toys' 4,950, limiting shared-map training support. Two maps remain unchanged; the third changes two identities, with near-zero matched construction effects (Appendix E.3).

5.5 Candidate access, certification and inference cost

Separating candidate reuse and additional scoring. Reranking E_0 reuses fully scored leaves without additional evaluations, improving mean NDCG@10 over returned-beam reranking on Beauty/Toys and tying Tools. Full completion further improves Recall@10 and NDCG@10 in all nine runs, recovering a net 3.36/6.65/0.74 Top-10 hits per 1,000 queries (Table 2; Appendix E.4).

Certification efficiency. At the same score, stopping test and 80-item maximum, bound priority reduces Beauty's additional evaluations from 60.69 to 51.08 (15.83%) at nearly identical Recall@10 and NDCG@10. Numerical Top-20 certification reaches 77.64/4.45/61.53% on Beauty/Tools/Toys, versus 45.40/0.71/31.95% under collaborative priority. Tools nearly exhausts both budgets. Certification follows Appendix B.5's numerical allowance and requires (11); budget exhaustion is insufficient.

Measured inference cost. GPU wall time covers beam search, item scoring and completion. Batched means for bound/collaborative priority are 10.16/10.50, 10.64/10.63 and 10.96/11.98 ms/query on fixed Beauty, Tools and Toys subsets (Table 11).

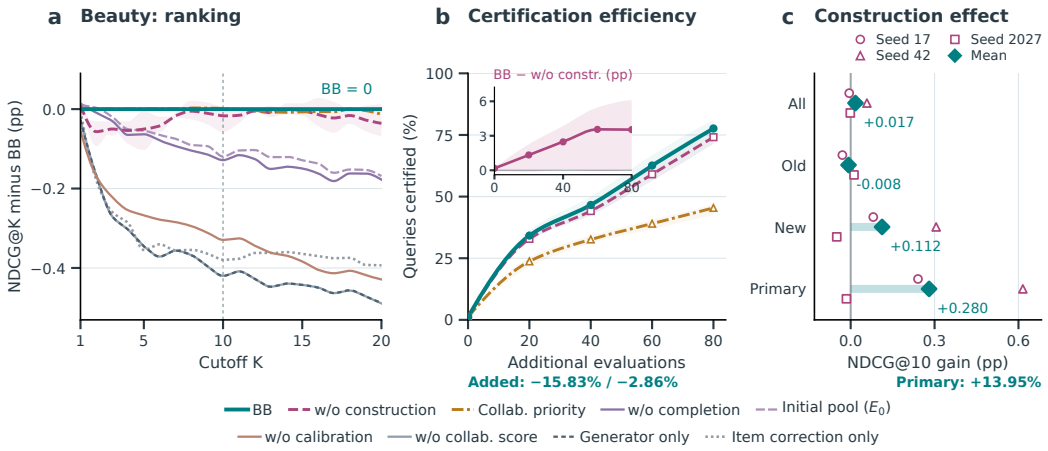


Figure 2: Beauty T5 ablations, seeds 17/42/2027. (a) Paired NDCG@K differences, variant minus BB (pp). (b) Certification efficiency: numerical Top-20 certification versus additional evaluations; inset: paired BB-minus-without-construction gap (pp). The two reductions compare BB with collaborative priority and without construction, respectively. (c) NDCG@10 differences, BB minus without construction; labelled open symbols identify seeds and diamonds denote means. Primary denotes current-catalog new targets with entirely old-item histories, a subset of New. All/Old/New/Primary contain 7,335/5,334/1,486/734 queries per seed; All includes future-target misses. Curves interpolate measured seed means, with bands showing the range across seeds. The Primary annotation reports the relative gain.

6 Related Work

Identifier and model updates. JTM matches items to leaves (Zhu et al., 2019); Reformer, DACT and SID-Staleness update tokenization (Shi et al., 2025; Feng et al., 2026; Baikalov et al., 2026); DREAM refines assignments and preserves multiple decoding paths (Guan et al., 2026). BB characterizes repair feasibility while preserving old identifiers, then minimizes leaf replacement for a specified state and target.

Reachability and scoring. Prefix support, training rollouts and item resolution affect retrieval (Peng et al., 2026; Zhang et al., 2026; Ding et al., 2026a); beam-aware training addresses pruning during learning (Zhuo et al., 2020; Yang et al., 2026). BB connects assignment repair with collaborative scoring and access to items outside the initial beam.

Candidate access and search. SpecGR proposes and verifies candidates (Ding et al., 2026b). Building on monotone search bounds (Huang et al., 2017; Meister et al., 2020), BB uses a known item correction to bound the combined score, including pruned paths, and certify catalog-wide Top-K when the stopping condition holds. Appendix G develops these connections.

7 Discussion and Conclusion

Shared construction and adaptation improve new-target ranking and certification with fewer evaluations on Beauty and Toys. Ranking gains are largest for entirely old-item histories; old-target losses partly offset them in the population mean.

Combined scoring and completion improve NDCG@10 across all three datasets with both T5 and decoder-only LC-Rec, linking candidate access to the final ranking objective across these backbone families. Extending repair beyond late branching after a common effective prefix, optimizing shared assignments across contexts and tightening completion bounds are natural next steps.

AI USE STATEMENT

Generative AI tools were used solely to assist with language editing and polishing during manuscript preparation. All other aspects of this work were completed by the authors without generative AI assistance. The authors reviewed and approved the final manuscript and take full responsibility for its content.

REPRODUCIBILITY STATEMENT

Experimental settings and evaluation protocols are provided in Appendices C and F for T5 and LC-Rec, respectively. The selected T5 configurations are listed in Table 3. These details are provided to facilitate reproduction of the reported results. The code is available at `https://anonymous.4open.science/r/bb_code-809D/`.

REFERENCES

Fabio Aiolli. Efficient top-N recommendation for very large scale binary rated datasets. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pp. 273–280, 2013. doi: 10.1145/2507157.2507189.

Vladimir Baikalov, Iskander Bagautdinov, and Sergey Muravyov. Mitigating collaborative semantic ID staleness in generative retrieval. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 3602–3608, 2026. doi: 10.1145/3805712.3809877.

Dimitri P. Bertsekas. *Linear Network Optimization: Algorithms and Codes*. MIT Press, 1991. URL `https://web.mit.edu/dimitrib/www/LNets_Full_Book.pdf`.

Jiandong Ding, Huijie Qin, Tiandeng Wu, and Yi Cao. SIDScope: A diagnostic resource for semantic-ID interfaces in generative recommendation. *arXiv preprint arXiv:2608.18779*, 2026a. doi: 10.48550/arXiv.2608.18779.

Yijie Ding, Jiacheng Li, Julian McAuley, and Yupeng Hou. Inductive generative recommendation via retrieval-based speculation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40 (17):14675–14683, 2026b. doi: 10.1609/aaai.v40i17.38486.

Su Feng, Boris Glavic, and Oliver Kennedy. Efficient approximation of certain and possible answers for ranking and window queries over uncertain data. *Proceedings of the VLDB Endowment*, 16 (6):1346–1358, 2023. doi: 10.14778/3583140.3583151.

Yuebo Feng, Jiahao Liu, Mingzhe Han, Dongsheng Li, Hansu Gu, Peng Zhang, Tun Lu, and Ning Gu. Drift-aware incremental token adaptation with collaborative semantics for generative recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 336–346, 2026. doi: 10.1145/3805712.3809645.

Liwei Guan, Huanjie Wang, Hongwei Zhang, Linxun Chen, and Zhaojie Liu. DREAM: Dynamic refinement of early assignment mappings. *arXiv preprint arXiv:2606.06947*, 2026. doi: 10.48550/arXiv.2606.06947.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pp. 507–517, 2016. doi: 10.1145/2872427.2883037.

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. doi: 10.1145/3397271.3401063.

Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*, pp. 1162–1171, 2023. doi: 10.1145/3543507.3583434.

Yupeng Hou, Haven Kim, Clark Mingxuan Ju, Eduardo Escoto, Neil Shah, and Julian McAuley. Expressiveness limits of autoregressive semantic ID generation in generative recommendation. *arXiv preprint arXiv:2605.06331*, 2026. doi: 10.48550/arXiv.2605.06331.

Liang Huang, Kai Zhao, and Mingbo Ma. When to finish? Optimal beam search for neural text generation (modulo beam size). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2134–2139, 2017. doi: 10.18653/v1/D17-1227.

Bowen Jin, Hansi Zeng, Guoyin Wang, Xiusi Chen, Tianxin Wei, Ruirui Li, Zhengyang Wang, Zheng Li, Yang Li, Hanqing Lu, Suhang Wang, Jiawei Han, and Xianfeng Tang. Language models as semantic indexers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22244–22259, 2024. URL https://proceedings.mlr.press/v235/jin24h.html.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining*, pp. 197–206, 2018. doi: 10.1109/ICDM.2018.00035.

Clara Meister, Tim Vieira, and Ryan Cotterell. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809, 2020. doi: 10.1162/tacl_a_00346.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, 2022. doi: 10.18653/v1/2022.findings-acl.146.

Jie Peng, Yanping Zheng, Zhewei Wei, Bin Tong, Guan Wang, and Bo Zheng. Can generative recommendation reach cold items? A temporal perspective on semantic-ID generation. *arXiv preprint arXiv:2607.21101*, 2026. doi: 10.48550/arXiv.2607.21101.

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

Vivek Ramanujan, Pavan Kumar Anasosalu Vasu, Ali Farhadi, Oncel Tuzel, and Hadi Pouransari. Forward compatible training for large-scale embedding retrieval systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19364–19373, 2022. doi: 10.1109/CVPR52688.2022.01878.

Chenglei Shen, Teng Shi, Weijie Yu, Xiao Zhang, and Jun Xu. GenRecEdit: Adapting model editing for generative recommendation with cold-start items. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1601–1611, 2026. doi: 10.1145/3805712.3809662.

Yantao Shen, Yuanjun Xiong, Wei Xia, and Stefano Soatto. Towards backward-compatible representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6367–6376, 2020. doi: 10.1109/CVPR42600.2020.00640.

Haihan Shi, Xinyu Lin, Wenjie Wang, Wentao Shi, Junwei Pan, Jie Jiang, and Fuli Feng. Incremental learning for LLM-based tokenization and recommendation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 2643–2652, 2025. doi: 10.1145/3746252.3761385.

Tianhao Shi, Yang Zhang, Zhijian Xu, Chong Chen, Fuli Feng, Xiangnan He, and Qi Tian. Preliminary study on incremental learning for large language model-based recommender systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 4051–4055, 2024. doi: 10.1145/3627673.3679922.

Harald Steck. Embarrassingly shallow autoencoders for sparse data. In *The World Wide Web Conference*, pp. 3251–3257, 2019. doi: 10.1145/3308558.3313710.

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2400–2409, 2024. doi: 10.1145/3627673.3679569.

Weiqin Yang, Bohao Wang, Zhenxiang Xu, Jiawei Chen, Shengjia Zhang, Jingbang Chen, Canghong Jin, and Can Wang. BEAR: Towards beam-search-aware optimization for recommendation with large language models. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2196–2207, 2026. doi: 10.1145/3805712.3809533.

Hyunsik Yoo, Ting-Wei Li, SeongKu Kang, Zhining Liu, Charlie Xu, Qilin Qi, and Hanghang Tong. Continual low-rank adapters for LLM-based generative recommender systems. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=DBCNTM7mot.

Kangning Zhang, Haotian Fang, Xukun Luo, Hao Yin, Yang Gao, Peng Yan, Weiwen Liu, Weinan Zhang, and Yong Yu. Learning from unreachable rewards: Hint-conditioned reinforcement learning for generative recommendation. *arXiv preprint arXiv:2608.11980*, 2026. doi: 10.48550/arXiv.2608.11980. Accepted at the 35th ACM International Conference on Information and Knowledge Management.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering*, pp. 1435–1448, 2024. doi: 10.1109/ICDE60146.2024.00118.

Han Zhu, Daqing Chang, Ziru Xu, Pengye Zhang, Xiang Li, Jie He, Han Li, Jian Xu, and Kun Gai. Joint optimization of tree-based index and deep model for recommender systems. In *Advances in Neural Information Processing Systems*, volume 32, pp. 3971–3980, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/1c6a0198177bfcc9bd93f6aab94aad3c-Abstract.html.

Jingwei Zhuo, Ziru Xu, Wei Dai, Han Zhu, Han Li, Jian Xu, and Kun Gai. Learning optimal tree models under beam search. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11650–11659, 2020. URL https://proceedings.mlr.press/v119/zhuo20a.html.

A Theoretical Results, Proofs and Counterexamples

We first certify when assignment leaves the output unchanged and convert this certificate into a cohort bound. We then characterize exact repair after a common effective prefix, using the catalogs that realize each candidate state.

A.1 Certifying output invariance

Write $z_{d,1}, \dots, z_{d,W+1}$ for the highest-scoring unmasked extensions, with scores $q_{d,1} \geq \cdots \geq q_{d,W+1}$, and O_d for the depth-d prefixes of O. Define

$$z_{d,j} \in O_d \ (1 \leq j \leq W), \qquad \gamma_d = \min_{1 \leq j \leq W} (q_{d,j} - q_{d,j+1}) > 0. \tag{12}$$

All selected hypotheses must be finite and live. Inactive or finished hypotheses must not displace them before the common termination.

Proposition 1 (All-assignment output invariance)**.** *Under the stated state, scoring, and termination premises, if* (12) *holds at every layer of one admissible run, every catalog in* (1) *returns the same ordered W old paths and the same item identities.*

The detailed proof is given in Appendix A.5.

The pass/fail flag is itself allocation-independent: two reference runs share all states through their first failed layer, including that layer's test. The checker observes the decoder's unmasked scores and records one flag/margin per layer; it requires no target labels, candidate pool, or extra model forward pass. The result is a sufficient certificate: a failed check leaves repair unresolved. Numerical ties or unverified execution premises return unknown (Appendix D.7).

A.2 Bounding improvement over a query cohort

For a fixed N-query cohort $\mathcal{Q}$, let C be queries with certified per-query metric upper bounds $m_u(c) \le c_u \in [0,1]$ over legal assignments c. Then

$$\sup_c \frac{1}{N} \sum_{u \in \mathcal{Q}} m_u(c) \le \frac{\sum_{u \in C} c_u + N - |C|}{N} =: U_C. \tag{13}$$

Ordered-output invariance permits $c_u = m_u^0$, its fixed metric value. For a metric on new targets, any sound no-new Top-K certificate permits $c_u = 0$, even when old outputs differ across states, giving $U_C = 1 - |C|/N$. Old-target contributions use their own certified bounds. Unknown and mixed-input queries contribute the optimistic value one. One shared allocation obeys

$$\underbrace{\sup_c \frac{1}{N} \sum_u m_u(c)}_{\mathrm{OPT}_{\mathrm{shared}}} \le \underbrace{\frac{1}{N} \sum_u \sup_c m_u(c)}_{\mathrm{OPT}_{\mathrm{separate}}} \le U_C. \tag{14}$$

An observed update exceeding U_C establishes a separation from every legal allocation of the parent. A value below U_C leaves the comparison unresolved. Equation (14) also locates a source of slack: different queries can require incompatible assignments. Thus individual witnesses and cohort exclusion answer distinct questions about the same allocation family.

A.3 Exact catalogs and fixed-count reachability

The sufficient certificate requires old prefixes to occupy the whole beam. We now allow the beam to branch in the final two layers and solve reachability through the catalogs that induce each state.

Common-prefix setting. Assume a common, valid effective beam b of W old-supported prefixes at depth $L-2$ for every containing catalog, independently of its new-item count. Successful prefix checks in Section A establish this premise. When the premise is known for a single m, the result applies at that count; reuse across counts requires a common state over the count range. Let $K \le W$. A fixed total order extends scores and is preserved when candidates are deleted. Decoder evaluations verify strict relevant gaps and leave ambiguous ties unknown.

Let E extend b to depth $L-1$, and let $M = E \cap O_{L-1}$ be its mandatory candidates. At least W exist. If τ is their Wth score, optional candidates strictly below τ cannot survive. Enumerating subsets of $Q = \{e \in E \setminus M : q(e) \ge \tau\}$ and merging identical ordered Top-W indices therefore covers every possible penultimate beam S; put $q = |Q|$.

For one such S, let F_S be the unselected optional prefixes preceding its last member. Write $D(e)$ for the full leaf descendants of e and set

$$Z_S = \mathcal{Z} \setminus \bigcup_{e \in F_S} D(e), \quad U_S = Z_S \setminus O, \quad C_S = |U_S|. \tag{15}$$

The selected nonold prefixes require support groups $\mathcal{G}_S = \{D(s) : s \in S \setminus O_{L-1}\}$, with $r_S = |\mathcal{G}_S|$. Each group needs an occupied descendant to keep its prefix in the beam. Together, forbidden subtrees and required support determine the complete catalog family for S.

Theorem 2 (Exact catalog family)**.** *A containing catalog reaches S if and only if*

$$O \subseteq A \subseteq Z_S, \qquad A \cap G \neq \varnothing \quad (G \in \mathcal{G}_S). \tag{16}$$

It does so with exactly m new leaves if and only if $r_S \le m \le C_S$.

The detailed proof is given in Appendix A.6.

For final alphabet size B_L, $C_S = |\mathcal{Z}| - |O| - B_L|F_S|$ is symbolic. It includes compatible leaves outside the current beam, which can supply padding.

Which admission counts permit a repair? After fixing S, a repair must both support the beam and keep enough higher-ranked leaves out of the catalog. Score all terminal children of S. For a new terminal leaf y, let $R_S(y)$ be its full one-based rank. Let $a_S(y)$ count the old leaves preceding it and $g_S(y)$ count the required groups whose *every* leaf precedes it. Put $\delta_S(y) = 0$ if y belongs to a required group, and 1 otherwise.

Theorem 3 (Exact fixed-count reachability)**.** *There exists an admissible allocation of exactly m new items reaching S and returning y in Top-K if and only if*

$$\boxed{a_S(y) + g_S(y) < K, \qquad r_S + \delta_S(y) \le m \le C_S - \max\{0, R_S(y) - K\}.} \tag{17}$$

A satisfying case constructively yields a catalog and an assignment of a specified new target to y.

The detailed proof is given in Appendix A.6.

Equation (17) separates three requirements. Old outrankers and unavoidable support must leave room in Top-K. The lower endpoint supplies enough new leaves to support the state and target. The upper endpoint reserves enough unoccupied paths to remove optional outrankers. These requirements connect catalog size directly to retrieval, beyond checking whether a target path exists.

A.4 Constructing repairs and accounting for search

Exact decision and witnesses. Enumerate S, discard states with $m \notin [r_S, C_S]$, obtain their complete terminal scores, and apply (17) to every new leaf. Within the common-prefix setting, the absence of a feasible pair certifies that no m-admission catalog returns a new item in Top-K. A feasible pair gives one query's witness; different queries need not share that allocation. With unrestricted admission count, universal no-new retrieval requires every state's full Top-K to contain only old leaves: if a new leaf is in a state's full Top-K, Z_S realizes that state and outcome. Such a leaf has a witness using at most $r_S + 1 \le W + 1$ new paths. Computation caps and numerical ambiguity still mean unknown, not an exact negative.

Algorithm and cost boundary. One terminal sort plus cumulative old counts and required-group endpoints gives all intervals in $O(WB_L \log(WB_L))$ time and $O(WB_L)$ storage after scoring. There are at most 2^q distinct states, each with its own decoder continuation. Algorithm 2 summarizes the construction. Supporting the selected state and filling with non-outrankers first attains the rank bound. The cost depends on critical branches and terminal scoring, in addition to writing the allocation. Appendix A.6 gives the full procedure and complexity accounting. Unverified execution premises or ambiguous numerical ties return unknown before an affected decision is made.

Theorem 4 (Complete-state oracle lower bound)**.** *For every $q \ge 1$, a positive normalized three-layer score family with $W = q + 1$, $K = 1$, and $m = q + 1$ has 2^q reachable penultimate states and q critical optional prefixes. Any deterministic exact algorithm deciding universal no-new Top-1 through arbitrary complete-state terminal-score oracle access requires 2^q state queries in the worst case.*

A query returns the entire terminal-score vector for one ordered effective state. The proof places the successful repair in one hidden state (Appendix A.6). It identifies the worst-case information cost of exact decisions when complete-state scores are accessed through this oracle. Prefix-local scoring can supply additional structure beyond that interface.

A.5 Proof Details and Diagnostic Counterexamples

A.5.1 Effective-state induction

Proposition 1 concerns the state consumed by the next scoring step, and the final ordered predictions. It does not require every temporary decoder variable to agree. Let S_d^A contain the ordered live hypotheses, their accumulated scores, model inputs/positions, and the cache state after the layer-d transition under catalog A. All random choices and numerical execution conditions are fixed. Query separability means that changing another query's values, without changing this query's numerical execution shape, does not change its scoring function or transition. This is a stated numerical premise.

For an arbitrary A satisfying (1), assume $S^A_{d-1} = S^{A_0}_{d-1}$ for the reference A_0. The unmasked score vector is identical. Every reference top-W expansion belongs to O_d, hence to the prefix closure of A. The positive minimum adjacent gap fixes its rank among all codebook extensions. An admissible mask can delete only lower-ranked competing extensions, so the same ordered top W is selected. Identical predecessor indices and tokens produce identical next inputs and cache reorderings through the decoder transition. Thus $S^A_d = S^{A_0}_d$ for every layer whose state is consumed by a subsequent score computation. At the common terminal layer, the selected paths and scores agree; all paths lie in O and resolve to the same old items.

Catalog independence of the *certificate* follows from the same induction until the first failed condition. Before that layer the states coincide, so the failure agrees too. No claim is made about later flags on an already failed trace. An old-only catalog can therefore serve as the reference if it meets the decoder's validity assumptions. Same-length new aliases do not help: any such catalog is still included in (1).

For (13), fix any legal assignment c and split the cohort sum into C and $\mathcal{Q} \setminus C$. By assumption, the first sum is at most $\sum_{u \in C} c_u$; each remaining contribution is at most one. Dividing by N and taking the supremum over c proves the bound. Under ordered-output invariance, the certified contributions equal their fixed reference values. The supremum need not attain the bound because all queries share one global assignment. For a new-target metric the certified contribution is zero. For an old-target metric it is its fixed reference contribution, not necessarily zero.

A.5.2 Final-only and static-support checks are insufficient

Consider two constructed scorers, a two-level code space with four symbols per level, $W = 2$, and old paths $O = \{a0, a1, c1, c2, c3\}$. In the reference allocation the single new item occupies $a2$. Both its atomic symbols and its first prefix occur in old paths. The two scorers share all second-level probabilities:

Prefix	0	1	2	3
a	.60	.38	.01	.01
b	.26	.25	.25	.24
c	.97	.01	.01	.01
d	.25	.25	.25	.25

Their root probabilities over a, b, c, d are $(.75, .04, .20, .01)$ and $(.49, .30, .20, .01)$, respectively. All values are evaluated as exact rational numbers.

For the first scorer, full-codebook search keeps a, c and then selects old paths $a0 = .45$ and $a1 = .285$, ahead of $c0 = .194$. Every layer satisfies the strict old-prefix condition. Enumerating all 11 unoccupied paths as the new item's allocation gives the same old prediction list and no new hit.

For the second scorer, full-codebook search keeps a, b, then also returns old paths $a0 = .294$ and $a1 = .1862$. The published catalog permits only root prefixes a, c; new $a2 = .0049$ still loses. However, allocating the new item to $c0$ instead gives $c0 = .194 > a1 = .1862$, so it enters at rank two. All old paths and identities stay fixed. Full-codebook search is not a monotone envelope of returned paths in its subcatalogs: branch b removes c early but produces no competitive final leaf. The two scorers therefore agree on static support and both simple final-only observations, yet differ in whether allocation repair is possible. The complete example uses 26 exact beam calls: two scorers, each with the current catalog, full codebook, and all 11 legal allocations. The construction isolates the effect of intermediate beam competition.

A.5.3 Why masking and old eligibility matter

Local renormalization changes the score function with the catalog. Let $W = L = 2$, $O = \{00, 01, 10\}$, and use the following probabilities over four symbols:

$$p_{\text{root}} = (.59, .40, .005, .005), \qquad p_0 = (.65, .34, .005, .005), \qquad p_1 = (.01, .45, .27, .27).$$

The unrenormalized full-codebook top two are old paths $00 = .3835$ and $01 = .2006$. Renormalizing over valid children and then adding new path 11 instead yields $P(11) = 100/253 \approx .395257$,

ahead of $P(00) = 3835/9801 \approx .391287$. The original mask-only condition cannot be transferred to that decoder.

Old lookup validity is also insufficient. With width one and fixed scores .6 for an old path and .4 for a new path, hiding the old candidate returns the new item even if a historical resolver still maps the old path correctly. Our $O \subseteq A$ requirement includes eligibility. Reserved tombstones, versioned identifiers, vocabulary expansion, and different input encodings must therefore be analyzed under their actual interfaces.

A.6 Exact-repair proofs and decision procedure

A.6.1 Catalog family and fixed-count endpoints

The common-prefix premise covers every containing catalog $O \subseteq A \subseteq \mathcal{Z}$, irrespective of $|A \setminus O|$. Consequently it also covers the maximal catalog Z_S and all padding choices. If that premise is established only for a fixed count, the theorem is pointwise at that count; terminal scores cannot automatically be reused at other counts. Effective next state is determined by the common prior state and ordered selected candidate indices, including the corresponding decoder cache update.

All mandatory prefixes are enabled. For an obtainable ordered beam S, no unselected mandatory candidate can precede its final member. Every selected optional prefix must have an occupied descendant, and every unselected optional prefix above the cutoff must have none. Conversely those conditions leave all selected prefixes eligible and all other eligible candidates below the cutoff. This proves the necessity and sufficiency of (16). The sets $D(s)$ for selected optional prefixes are disjoint, nonempty subsets of U_S; selecting one leaf per group realizes r_S admissions. Any remaining member of U_S can be added without changing the beam, realizing every count through C_S. Forbidden subtrees are also disjoint and contain no old leaf, giving the symbolic capacity $C_S = |\mathcal{Z}| - |O| - B_L|F_S|$.

For a specified new leaf y, partition $U_S \setminus \{y\}$ into the h optional terminal outrankers H and the $t = C_S - 1 - h$ remaining compatible leaves T. The latter include all compatible leaves outside the terminal expansion. A required group wholly in H forces one selected outranker. Every other group contains y or a member of T. The disjointness of groups makes the minimum support count $r_S + \delta_S(y)$ exact. Any m-element new set containing y requires at least $m - 1 - t$ leaves from H, and at least $g_S(y)$ for support. Thus its best possible terminal rank is

$$1 + a_S(y) + \max\{g_S(y), m - 1 - t\}.$$

Choose y, one T member of each unsupported group whenever possible, and one H member otherwise. Fill unused T capacity before selecting further H members. This attains the lower bound for every $r_S + \delta_S(y) \le m \le C_S$. Using $R_S(y) - 1 = a_S(y) + h$ gives the upper endpoint $C_S - \max\{0, R_S(y) - K\}$ and the separate obstruction $a_S(y) + g_S(y) < K$. Equality at either endpoint is permitted. Too few admissions cannot support both the state and target; near a full catalog, too many can force outrankers back in. The interval describes the reachability of the specified leaf as m varies.

If the unrestricted full terminal Top-K contains a new leaf, its full rank is at most K and Z_S realizes that outcome. Supporting each optional selected prefix and y needs at most $r_S + 1 \le W + 1$ new leaves; deleting the other optional leaves cannot worsen y while those supports and forbidden-prefix constraints preserve S. Conversely, K mandatory old leaves preceding every new leaf remain present in every compatible catalog. This gives exact unrestricted no-new certification over the complete reachable state set. The branch experiment in Appendix D evaluates the sufficient no-new predicate used for its coverage comparisons.

A.6.2 Constructive decision procedure and complexity

Every unverified numerical tie, execution premise, or exhausted computation cap returns unknown; Algorithm 2's final negative requires a complete valid enumeration. A fixed total secondary order handles abstract ties. Applications to trained models retain the tie order of the implemented decoder.

A group's contribution to $g_S(y)$ starts just after its worst-ranked leaf. After one sort, cumulative old flags and these group endpoints give all count intervals in linear additional time. If $n \le 2^q$ distinct

Algorithm 2 Fixed-count repair after a certified common prefix

Require: Fixed input/model/decoder, mandatory old paths O, admission count m, $K \le W$

if common depth-$(L-2)$ effective state is not established **then**
  return unknown
Obtain its global expansion scores; form mandatory M and critical optional Q
Enumerate subsets of Q with M retained; merge identical ordered Top-W states
for each state S **do**
  Form forbidden prefixes F_S, support groups $\mathcal{G}_S$, and capacity C_S
  if $r_S \le m \le C_S$ **then**
    Obtain terminal scores at the full effective state S
    Sort once; compute ranks, old-before counts, and required-group endpoints
    for each new terminal leaf y satisfying (17) **do**
      Select y and support every uncovered group using a non-outranker when possible
      Pad to m new leaves with compatible non-outrankers before outrankers
      Assign the designated target to y and biject the other new items to remaining leaves
      return reachable and the complete allocation witness
return unreachable

states survive the count filter, a straightforward implementation reads WB_{L-1} expansion scores, spends $O(2^q(|M|+q)\log(|M|+q))$ time on subset planning, requests n terminal-state evaluations, and spends $O(nWB_L \log(WB_L))$ on terminal ranking. Intervals do not require rescoring for each m within the common-state range. Explicitly writing a catalog of m new identifiers still costs at least m outputs; symbolic capacity does not remove that cost.

A.6.3 Proof of the state-oracle lower bound

For each $q \ge 1$, use three layers with root alphabet size $2q+1$ and binary second/final alphabets. Every root has one old leaf at suffix $(0,0)$. Set $W = q+1$. The anchor root has probability $3/4$. Distribute total mass $249/1000$ among q lower roots in proportion to $1 + i/(100q)$, $i = 1, \dots, q$. Each of q off-beam roots has mass $1/(1000q)$. All roots are old-supported, so the ordered beam after layer one is common: the anchor and all lower roots. The padding roots are always excluded.

Split the anchor into old/optional probabilities $1 - 1/(10000q)$ and $1/(10000q)$; split each lower root into $2/5$ old and $3/5$ optional. Lower root masses vary by less than a factor 1.01, so all lower optional children precede all lower old children. The anchor old child dominates them, and the anchor optional child is below the lowest mandatory score. Exactly q optional children are critical. Every subset survives entirely, with mandatory lower old children filling unused positions; all 2^q states are distinct.

The anchor's old terminal leaf wins under the good oracle, with old/new terminal probabilities $.9/.1$. Other prefixes can use $.6/.4$; even their entire accumulated mass is smaller than the winning anchor leaf. For any designated state S^*, change only the anchor terminal distribution there to $.01/.99$. Its new leaf y then wins. Every probability remains positive and normalized, and all other state answers agree with the good oracle. Arbitrary dependence on the complete ordered state permits this family.

Each off-beam root has three optional leaves, providing $3q$ padding leaves that cannot alter the first beam. A fixed-count witness for any state includes y, one leaf for each selected optional prefix, and enough such padding to reach $m = q+1$. It introduces no forbidden prefix and reaches the designated state. Thus every hidden bad state has a witness at the same admission count.

One oracle query supplies the complete terminal vector for one specified ordered state. On a run receiving only good answers, any unqueried state might be S^*. The good oracle requires a universal-no-new answer, whereas the alternative requires the opposite answer. Their observed responses are identical, so a deterministic exact algorithm cannot stop before all 2^q state queries. This lower bound concerns complete-state terminal-score oracle access. Network weights, symbolic or prefix-local score structure, and richer batched oracles provide information beyond this model.

B Construction, Adaptation and Completion Details

This appendix develops the steps from assignment repair to candidate completion. A worked example first shows how changing catalog occupancy and evaluating additional candidates each affect retrieval. We then specify shared-map construction, generator adaptation and completion, with the final T5 configurations listed in Table 3.

B.1 A worked example of repair and completion

Figure 3 follows an old item a and a new item x under a fixed generator and old-item history, with two-token codes and $W = K = 1$. Their initial codes are P0 and Q0. Empty leaves are masked without renormalizing token probabilities. The beam selects prefix P and returns only a, giving the initial fully scored catalog set $E_0 = \{a\}$.

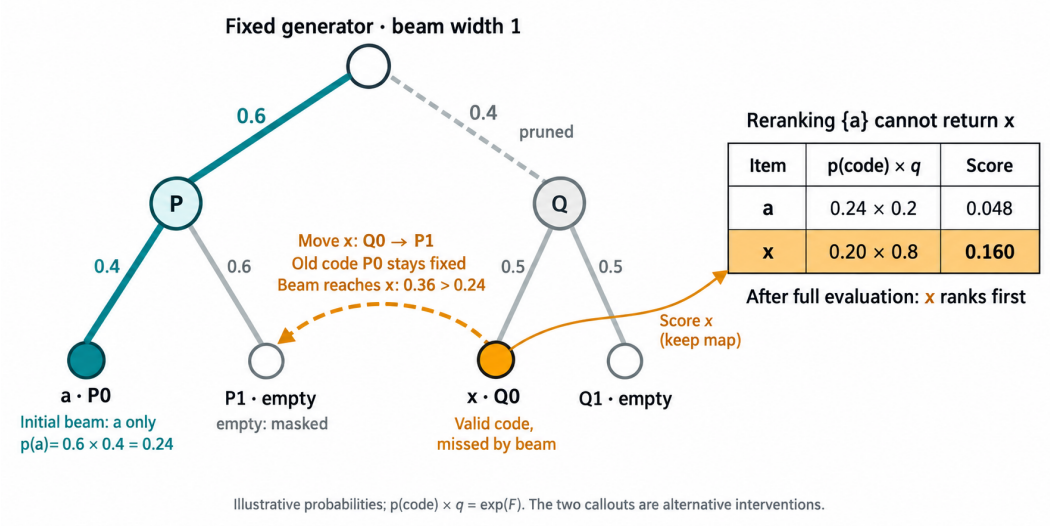


Figure 3: Repair and completion in a two-item catalog with illustrative probabilities. The teal path is the initial beam; hollow leaves are unoccupied. The dashed arrow changes the assignment, whereas the solid arrow keeps it fixed and adds a complete likelihood evaluation. These interventions are applied separately to the initial miss. Suffix probabilities below the pruned Q branch and the completed score of x are shown retrospectively.

The dashed arrow moves x from Q0 to P1 while retaining the old code P0. The beam then returns x, whose code probability 0.36 exceeds a's 0.24. The previously occupied Q0 becomes empty, completing a repair with one occupied-leaf replacement.

The solid arrow retains the original map. With known collaborative item scores $q_a = 0.2$ and $q_x = 0.8$, choose $(\lambda, \gamma, b) = (1, 1, 0)$ in (7). The figure displays $\exp(F_i) = p_\theta(c(i) \mid h)q_i$, preserving the ranking of (8). Reranking E_0 still cannot return x. Evaluating Q0 expands the scored set to $\{a, x\}$; x then ranks first with $0.160 > 0.048$, while both identifiers remain fixed.

In the full method, construction and adaptation determine the shared map and generator on which candidate completion operates. The following subsections detail these stages in that order.

B.2 From a repair witness to a shared assignment

The example repairs one query by replacing a single occupied leaf. General repairs must also satisfy prefix-support and rank constraints while preserving catalog size. For a fixed state and target leaf y, let U be the admissible new leaves, B_0 the current m-leaf occupancy, and G_j the disjoint support groups. Let H contain optional leaves ranked above y, and let a count mandatory old outrankers. The exact-state and rank constraints give the optimization

$$\min_{B \subseteq U} |B \setminus B_0| \quad \text{s.t.} \quad |B| = m, \quad y \in B, \quad B \cap G_j \neq \varnothing \ (\forall j), \quad |B \cap H| \leq K - 1 - a.$$

In the experiments, $K = 10$ matches the primary recommendation cutoff. Removing the forced target reduces demand to $m - 1$ and discharges its support group. The remaining leaves are partitioned by support group, membership in H, and membership in B_0. Each support group and the ungrouped remainder has at most four such buckets. A bucket has its integer leaf capacity and unit cost zero for retained leaves or one for replacements.

This formulation is an integral minimum-cost-flow problem. Source arcs supply group nodes, with lower bound one for each uncovered support group. Bucket arcs connect group nodes to high- or low-ranked nodes; the high node has sink capacity $K - 1 - a$. Total flow is $m - 1$. Integer capacities and network-flow integrality yield an optimal integral occupancy (Bertsekas, 1991, Section 2.3). The forced target's replacement cost is constant and is added to the objective. This minimizes leaf replacement for the specified state and target; identity edits and the search over states are separate decisions.

Equivalence and optimality. After removing y, let $V_j = G_j \setminus \{y\}$ for each support group, and let V_0 contain the remaining ungrouped leaves of $U \setminus \{y\}$. These sets partition the residual universe. The source-to-group arc has upper capacity $|V_j|$ and lower capacity one exactly when $j \neq 0$ and $y \notin G_j$; all other lower capacities are zero. Split V_j into buckets according to membership in H and B_0. A bucket arc has capacity equal to its size and cost zero or one according to retention, and ends at the corresponding high- or low-ranked node. Their sink capacities are $K - 1 - a$ and $m - 1$, respectively; the required total flow is $m - 1$.

Every feasible occupancy sends one unit along the bucket route of each leaf other than y. Its support constraints meet the lower capacities, and its outranker count meets the high-ranked capacity. Conversely, an integral feasible flow selects its prescribed number of distinct leaves from each bucket. Bucket capacities and disjointness make this possible; adding y then satisfies every occupancy constraint. Flow cost plus $\mathbf{1}\{y \notin B_0\}$ equals $|B \setminus B_0|$ in both directions. Integral capacities, lower bounds and total demand admit an integral minimum-cost flow whenever feasible, proving equality of the two optima. Negative high-ranked capacity or an unsupported empty group is infeasible. The network has at most four bucket arcs per group, so its size is $O(r_S + 1)$ once bucket capacities are available; finding those capacities and writing the selected leaves remain separate costs.

We pin the target identity, retain compatible original identities, and assign displaced identities deterministically. Complete mappings are checked after history re-encoding.

Candidate generation samples up to 256 training contexts and retains at most 12 proposals. It considers up to eight feasible states per context, with at most eight critical branches, 256 enumerated states, and four target leaves per state. Every proposal preserves old identities and changes at most 16 identities. A disjoint subset of up to 1,024 update-training examples ranks proposals by re-encoded NDCG@10, Recall@10, and fewer identity changes. It balances old and new targets where available and draws from the training examples used in the common warm-up. Starting from the original map, greedy composition accepts at most four proposals. Each proposal must change identities disjoint from earlier accepted proposals, preserve injectivity, and keep the total within 16 identity changes. At each step, we re-encode and evaluate every compatible merge on the same training subset. We accept the largest strict improvement in the ordered criteria above, retaining proposal order on ties, and stop when none improves the current map. Each accepted intermediate map joins the candidate pool. One nonoriginal mapping is selected from single proposals and these compositions; the lexicographically largest SHA-256 digest of the serialized mapping resolves remaining ties. When no feasible proposal exists, the mapping remains static.

B.3 TRAINING AND CHECKPOINT SELECTION

For the selected assignment, both history items and target items are encoded with the same fixed map. Equation (6) averages cross-entropy over the four identifier tokens using example weights $w(h, i)$ normalized to mean one over the training set. Uniform weighting sets $w(h, i) = 1$. The main T5 results use cross-day weighting for Beauty seed 2027 and Toys seed 42, and uniform weighting for the other runs. LC-Rec uses uniform weights. T5 updates all generator parameters. The LC-Rec adaptation uses the same identifier-token objective with the trainable modules described in Appendix F.

Within each dataset and seed, the constructed and static branches start from the same five-epoch warm-up. We treat the adaptation setting as a categorical hyperparameter with three candidate values. *Constant* uses a fixed learning rate of 3×10^{-4} throughout adaptation. *Cosine+EMA* uses cosine learning-rate decay from 10^{-4} to 10^{-5} and an exponential moving average (EMA) of decay 0.99. *Weighted EMA* adds cross-day training weight two and same-day weight one half to the cosine+EMA setting. For this weighted setting, $w(h, i) = \widetilde{w}(h, i)/\overline{w}$, where $\widetilde{w}$ is the stated raw weight and $\overline{w}$ is its mean over the whole training set. EMA is initialized from the warm-up weights, updated after every optimizer step, and used for validation and testing. AdamW, batch size 256, weight decay 0.01, and gradient clipping at one remain fixed. Each run initializes one optimizer and retains its state throughout adaptation.

Adaptation lasts at least 20 and at most 60 epochs after the common warm-up. Checkpoint selection uses the 1,040 Beauty and 774 Toys positive-day-gap validation examples; Tools uses all 3,007 validation examples because timestamps are unavailable. We evaluate at the warm-up checkpoint and every five adaptation epochs. Four evaluations without a strict NDCG@10 improvement on this selection subset trigger early stopping after the minimum duration. Checkpoint ordering uses NDCG@10 and Recall@10 on that subset, then full-validation NDCG@10 and earlier epoch.

The main comparison selects an adaptation setting for each dataset and seed using validation results. Table 3 gives the selected setting and checkpoint epochs for both paired branches, counting from the start of their common warm-up. Epoch 5 therefore denotes the trained warm-up checkpoint before any further adaptation. The without-construction branch uses the same adaptation setting and randomization with the original mapping. JTM and DREAM retain their 20-epoch update schedule; DACT follows Appendix C.5.

B.4 TRAINING DATA AND PARAMETER SELECTION

For a history of length T, its item at position t receives weight $\exp[-\rho(T-t)]$; repeated occurrences are summed and the vector is normalized to unit sum. Training histories form X and their one-hot next items form Y. The predictor minimizes a supervised ridge objective:

$$B_\alpha = \arg\min_B \|Y - XB\|_F^2 + \alpha\|B\|_F^2 = (X^\top X + \alpha I)^{-1} X^\top Y. \tag{18}$$

Its next-item target differs from reconstruction of the history vector. The linear predictor fits available initial and update training examples, deduplicated by user, history and target. Beauty, Tools and Toys contain 55,909, 49,661 and 46,964 distinct examples. Histories are normalized recency weighted item counts; the target is a one-hot next-item vector. Validation and test examples are excluded from this supervised fit. Repeated items remain eligible under the shared recommendation protocol.

We evaluate six predictor configurations per domain: the Cartesian product of decay $\rho \in \{0, 0.2\}$ and ridge $\alpha \in \{1, 10, 100\}$. Validation NDCG@10 selects a configuration, followed by Recall@10 and fixed grid order to resolve ties. The selected ridge is 100 in all domains; decay is 0.2 for Beauty and Tools and zero for Toys. The normal equations are solved by double-precision Cholesky factorization; sampled-column relative residuals are below 10^{-8}. Selected coefficients are stored in single precision.

The collaborative score is clipped at 10^{-8} before taking its logarithm. Scoring parameters (λ, γ, b) are selected using the validation procedure in Appendix C.4. Table 3 reports the configuration used for each test run and its matched ablations. Figure 9 compares calibrated and global corrections under these configurations.

B.5 Inference and numerical verification

The decoder computes full-vocabulary log probabilities before masking invalid catalog paths. Prefix accumulation uses their sum, without beam length normalization. The longest scored prefix of each catalog item provides its bound. All catalog leaves already scored in the terminal beam expansion enter the initial evaluated set, including leaves outside the final width-40 output. We record this set size separately from additional teacher-forced evaluations.

Additional candidates are scored in groups of 20, with an 80-item maximum per query. The final output contains 20 items. For the numerical stopping check, the twentieth known score minus 10^{-3} must exceed every remaining bound plus 10^{-3}. The mathematical guarantee in Proposition 2 concerns exact scores; this allowance and numerical checks do not constitute a formal floating-point error bound. A run reaching its item budget can still produce a useful list without satisfying the stopping check.

We verify prefix bounds and complete Top-20 rankings by exhaustive scoring of 64-leaf catalogs on three training histories, at weights 0, 0.5 and 2 with both candidate priorities. Complete scores are also checked against independent teacher forcing. The initial beam predictions match the unchanged generator outputs. Calibration is additionally checked in 144 exhaustive small-catalog comparisons covering the grid and both priorities. All test metrics are independently recomputed and candidate budgets verified across domains and seeds. Fresh evaluation on the timing subsets reproduces the saved rankings and added-item counts.

B.6 Completion guarantee

Proposition 2 (Completion certificate)**.** *For fixed history, model, map and collaborative scores, let E be the set of items with complete likelihoods, and let U_i be defined by* (9)*. If the Kth largest score in E satisfies* (11)*, the returned list is the global Top-K under $F(\cdot \mid h)$.*

Proof. Every suffix contributes a sum of nonpositive log probabilities, so each unevaluated item's complete score is at most U_i. Under the stopping condition, all such items score below the Kth evaluated item. Hence none can enter the returned Top-K set. □

Progress toward certification. With at least K evaluated items, define the certification gap

$$\Delta(E) = \max_{i \notin E} U_i - t_K(E), \qquad \max \varnothing = -\infty.$$

For fixed prefix bounds and item corrections, the update in (10) enlarges E, so its Kth largest score cannot decrease and the largest remaining bound cannot increase. Hence $\Delta(E_{t+1}) \leq \Delta(E_t)$; a negative gap gives the stopping certificate. This monotonicity does not guarantee certification within the item budget or a gain in recommendation quality.

B.7 Ablation definitions and computational accounting

The *generator only* control retains the constructed map and adapted model with ordinary beam decoding. *Without completion* applies the selected combined score to the original 40 candidates. *Without the collaborative score* sets $d_i = 0$ and retains completion. *Without construction* runs the full inference procedure on the paired static map and adapted checkpoint; it measures this complete assignment-and-adaptation choice. *Item correction only* ranks the current catalog by the calibrated item correction, with ties ordered by item index. *Collaborative priority* replaces the bound-based candidate order with this same correction, keeping the final score, bound checks, batch size and budget fixed. *Without calibration* sets $(\gamma, b) = (1, 0)$, retaining the selected λ and applying the global correction to all items.

For each query, the candidate budget counts newly evaluated complete items outside its initial fully scored set E_0. Initial terminal expansions, encoder work and the linear predictor also incur cost. Candidate counts and stopping fractions use every test query. Fresh runtime measurements use the same first 128 test histories per domain and retriever seed (17/42/2027), with neural inference on one NVIDIA RTX 3090 and two CPU threads for host-side work. All timed runs use the configurations in Table 3. Each seed has three timed repetitions, with CUDA synchronization around

each neural stage. The common width-40 beam and linear predictor are recomputed once per repetition and included in both priority costs. Each priority separately recomputes its encoder and candidate likelihoods; no likelihood cache is shared between them or reused from earlier experiments. Model/input loading and a four-query beam-and-completion warmup for both priorities are excluded. Beam batches contain 64 histories, completion batches 16, and teacher-forcing calls at most 256 sequences. Priority order alternates by repetition and seed. We average repetitions within each seed before computing seed means and sample standard deviations. All timed rankings, candidate counts and stopping flags match their full-test cached outputs. Table 11 separates full-population search counts from amortized wall time per query on this fixed, batched runtime sample.

For I items, the dense predictor stores $O(I^2)$ coefficients. A direct Cholesky fit costs $O(I^3)$; prediction from a sparse length-T history costs $O(TI)$. Prefix-bound arrays also scale with the current catalog.

C Data, Baselines and Evaluation Protocols

C.1 Task, data, and common training procedure

We evaluate shared assignments together with ordinary retriever updating on Amazon 2014 Beauty, Tools, and Toys and Games (He & McAuley, 2016). Beauty retains the chronological split described in Appendix C.2; Toys uses the same preprocessing rule with its own timestamps and content embeddings. Their initial catalogs contain items observed by the 60th event-time percentile, and the update catalogs include items observed by the 80th percentile. Training uses up to the last four eligible next-event prefixes per training user, with histories truncated to 20 items. Validation users are disjoint from training users. Test targets are the first subsequent interaction, with the history ending at the update cutoff. Boundary timestamp ties are included consistently. Tools uses the public period-0.8 training, validation, and test sequences supplied with DACT (Feng et al., 2026). Thus all methods within a dataset share the same examples and information; the two raw-review splits and the supplied Tools split retain their respective validation conventions.

We use all available update examples: 49,784/1,870 training/validation examples for Beauty, 4,227/3,007 for Tools, and 41,230/1,559 for Toys. The assignment objectives draw from all available new-target training contexts, including histories containing new items: 2,993, 133, and 4,950, respectively. These cover 532/595 Beauty, 74/158 Tools, and 944/1,025 Toys new items. The remaining identities stay in the candidate catalog and in evaluation. The complete test sets contain 7,335, 5,360, and 6,266 queries. Targets outside the available catalog remain misses in the full-stream metrics; current-catalog metrics and all denominators are also retained.

The T5 retriever (Rajput et al., 2023) architecture and content encodings are shared within each dataset. The main comparison uses parent seeds 17/42/2027, with checkpoint selection by validation loss. Beauty and Toys parents train for 40 epochs from random initialization, using AdamW at 3×10^{-4}, weight decay 0.01, batch size 256 and gradient clipping at one. Tools parents start from the public period-0.6 checkpoint and train on period-0.7 data with AdamW at 10^{-4}, weight decay 0.01 and batch size 256. Examples targeting items newly admitted in period 0.7 receive weight two. Tools parent training stops after 15 epochs without lower validation loss or at 200 epochs. In both procedures, validation is evaluated every epoch, including epoch zero, and the earliest minimum-loss checkpoint is retained.

JTM and DREAM use a common five-epoch warm-up before assignment selection, followed by 15 further training epochs. Training uses AdamW with learning rate 3×10^{-4}, weight decay 0.01, batch size 256, and gradient norm clipping at one. The optimizer is restarted after assignment. Within each compatible-parent comparison group and seed, methods receive the same initial weights and minibatch randomization. BB and its paired without-construction control use the validation-selected adaptation in Appendix B; DACT and Reformer use the selected settings specified below.

C.2 CHRONOLOGICAL SPLIT AND CONTENT ENCODING

Beauty uses Amazon Reviews 2014 5-core reviews and content-only sentence-T5-base embeddings (He & McAuley, 2016; Ni et al., 2022). Toys follows the same preprocessing rule with its own timestamps and embeddings.

Time and user boundaries. The 60th and 80th percentiles of event time, using the inverted empirical CDF, are Unix times 1380326400 and 1394150400. All ties at each boundary are included. The old catalog contains 11,183 items observed by the first cutoff; the current catalog contains 11,778 items observed by the second, admitting 595. At each cutoff, training retains the last four eligible next-event prefixes per user, with at least three history items and at most 20 inputs. Users whose SHA-256 account-ID integer is zero modulo ten are excluded from training and supply one last eligible validation prefix. No raw account IDs are released. This gives 34,836/1,416 parent train/validation examples and 49,784/1,870 update examples.

Each test query contains the last 20 events at or before the second cutoff and predicts the first event strictly after it. All 7,335 eligible users are retained; 5,732 queries have pure-old inputs, 1,486 have an admitted new target, and their intersection contains the 734 Primary queries. Mixed histories remain unknown in an allocation bound. Future targets are retained and score zero under the current catalog. Metrics decode predicted paths to eligible item identities: a future item's unallocated placeholder code must never count as a hit by accidentally matching an eligible item's code.

Fixed content tokenizer and learning. We normalize each 768-dimensional content vector and fit three residual KMeans levels, each with 64 centers, on old items only. Each level uses one initialization, at most 100 iterations, and seed $20260910 + d$ for zero-based level d. Each item receives its three center indices and a unique suffix from 0–255. Old items are assigned first, then new items, each in stable numeric item order; choose the first free suffix. Old paths are retained exactly. Four position-specific blocks of 256 tokens, with offsets 1/257/513/769, give a 1,025-token vocabulary including pad/start token zero.

Each parent is a fresh T5 with four encoder and decoder layers, model dimension 128, feed-forward dimension 1,024, six heads of key/value dimension 64, ReLU and dropout 0.1.

Temporal validation populations. Beauty and Toys timestamps have day-level granularity. Validation predicts the last eligible interaction before the update cutoff; testing predicts the first interaction after it using pre-cutoff history. Same-day targets constitute 44.39% and 50.35% of their validation sets, and none of their test sets. The corresponding median prediction gaps are 7/0 days in validation and 134/150 days in testing. For adaptation and scoring selection, the positive-day-gap subsets contain 1,040 Beauty and 774 Toys queries. Tools uses all 3,007 validation queries because timestamps are unavailable.

C.3 JTM AND DREAM ASSIGNMENT OBJECTIVES

JTM and DREAM share a score-based candidate bank and retriever procedure. A seed-controlled sample first selects one new-target context per observed item in random order, then additional contexts if required, up to 256. The bank contains the original catalog and up to eight catalogs formed from the highest-scoring free leaves on these sampled contexts.

For JTM (Zhu et al., 2019), we adapt its assignment objective by maximizing summed training sequence log probability. For DREAM, we use its UC3 module with temperature-one, entropy-weighted multi-context probabilities, at most eight candidates per item, minimum support three, and vote margin 0.05 (Guan et al., 2026). Its candidate union includes every item's original code. An item failing either confidence gate retains that code; rectangular maximum-weight matching resolves competing assignments among the remaining items.

A shared assignment can change codes in histories as well as targets. Every complete candidate mapping is therefore re-encoded and decoded on the full validation set before selection. The original mapping remains available as a fallback.

C.4 PARAMETER SELECTION AND EVALUATION

For JTM and DREAM, full-validation NDCG@10 selects the shared mapping, followed by Recall@10, fewer changed identities, and candidate order. We evaluate checkpoints after 5, 10, 15, and 20 total update epochs and select by validation NDCG@10, Recall@10, and earlier epoch. Test evaluation uses beam width 40, with Recall and NDCG at 10 and 20. The main comparison uses complete test populations; Appendix E reports component effects and old/new-target analyses, with subset definitions and denominators given alongside each result.

Main T5 results and matched ablations. We evaluate every method in the main T5 comparison on Beauty, Tools and Toys using seeds 17, 42 and 2027. For every seed, validation Recall@10, NDCG@10, Recall@20 and NDCG@20 jointly guide parameter tuning. We select one configuration per dataset and seed and use it to report all four test metrics. Table 3 specifies the selected adaptation setting, paired checkpoint epochs, mapping changes and scoring parameters. Table 1 reports means and sample standard deviations across these three seeds. This dispersion includes differences in the adaptation settings and scoring parameters selected for individual runs.

Initialization and validation populations. Each seed has a parent model and a five-epoch update warm-up. Beauty and Toys use seed-specific T5 parent training on their initial-period data; Tools uses seed-specific training from a shared public initialization. Constructed and static branches share the warm-up weights within each dataset and seed. The selected mapping is fixed before their adaptation trajectories. For adaptation and scoring selection, Beauty uses 1,040 positive-day-gap validation queries, Toys uses 774, and Tools uses all 3,007 validation queries. The training-subset shared-map selector is specified separately in (5).

Adaptation and scoring search. The categorical adaptation hyperparameter and checkpoint rules are given in Appendix B. Scoring search combines coarse comparisons and local refinement of (λ, γ, b). The coarse Tools comparison uses $\lambda \in \{0.125, 0.25, 0.5, 1, 2\}$, $\gamma \in \{0, 0.25, 0.5, 1\}$ and $b \in \{0.5, 1, 1.5, 2\}$. Additional local comparisons use multipliers 0.75/1 for λ, increments 0/0.125 for γ, and $b \in \{1.5, 2.25\}$. Refinement multiplies the reference λ by 0.75/1/1.25 and adds $-0.125/0/0.125$ to γ and $-0.25/0/0.25$ to b. Repeated configurations are evaluated once. These comparisons use the same four validation metrics; the reported parameter values identify the configuration used for each test run.

Table 3: Configurations for the nine T5 runs in Table 1. Adaptation setting is a categorical hyperparameter with three values defined in Appendix B. Selected update epochs include the common five-epoch warm-up and exclude parent training; epoch 5 denotes the trained warm-up checkpoint. They locate the selected checkpoints, not the stopping epochs. IDs count changed item codes; leaves count newly occupied paths, both relative to the paired static mapping. Validation uses the cross-day subset for Beauty/Toys and the complete split for Tools.

Dataset	Seed	Adaptation setting	Selected update epoch		IDs	Leaves	λ	γ	b
			BB	W/o construction					
Beauty	17	Constant	25	15	5	1	1.125	1.25	2.5
Beauty	42	Constant	15	5	2	0	1.5	1.25	2.5
Beauty	2027	Weighted EMA	10	10	2	0	1.125	1.125	2.5
Tools	17	Cosine+EMA	10	10	0	0	0.625	0.25	1.75
Tools	42	Cosine+EMA	5	5	2	0	0.75	0.25	2.25
Tools	2027	Cosine+EMA	5	5	0	0	0.25	0.625	2.25
Toys	17	Constant	45	10	2	0	0.25	1	1
Toys	42	Weighted EMA	60	60	1	1	0.5	1	1.5
Toys	2027	Constant	25	25	2	0	0.25	1	1

Matched component controls. Component removals inherit the corresponding BB scoring parameters. Without construction uses the static map and paired adaptation setting, with its selected

checkpoint shown in Table 3. The comparison therefore measures assignment together with subsequent adaptation. Score and candidate-access controls retain the BB map and checkpoint and change only the specified inference component.

C.5 DACT Comparison Across Three Domains

C.5.1 Data and compatible initialization

DACT (Feng et al., 2026) is evaluated on the same Beauty, Tools, and Toys queries and candidate catalogs as Table 1. Each domain has a collaboration-aware tokenizer and a retriever trained on its initial codes. Adaptation therefore starts from a model that has learned the preceding DACT encoding. Hyperparameters are tuned on validation data starting from the published configuration. Beauty and Toys use the existing initial-period training examples. Tools uses the deduplicated training examples from its first two observed periods: 41,576 plus 3,858 examples, with 2,993 validation examples at the second origin. This cumulative initialization retains the earlier interactions available to the other Tools parents. The update training, validation, and test examples are unchanged.

C.5.2 Collaborative tokenization

We use the public DACT encoder, residual quantizer, memory gate, and differentiated update objective. Item content is represented by the existing 768-dimensional embeddings. The encoder has widths 512, 256, 128, 64, and 32; the decoder reverses these widths. Three codebooks each contain 256 vectors of dimension 32. Initial training minimizes reconstruction, quantization, and collaborative alignment losses, with commitment weight 0.25 and alignment weight 0.02. It runs for 20,000 AdamW steps at learning rate 10^{-3}. DACT updating runs for 5,000 steps at 10^{-4}, with batch size 1,024, weight decay 10^{-4}, linear decay after 500 warm-up steps, and gradient clipping at one. These are optimizer-step budgets, following the paper's implementation description.

The update uses a 0.3 drift-selection fraction, gate temperature 0.1, old-branch weight 0.1, latent-stability weight one, assignment-divergence weight five, and gate penalty 0.001. The public implementation applies the assignment-divergence term to the first codebook. Code reassignment retains the entire previous code when its first token remains unchanged. Retained codes are reserved before assigning collision suffixes to changed and new items, giving an injective four-token map without altering those retained identities.

Collaborative features come from a 32-dimensional, three-layer LightGCN (He et al., 2020). Its graph contains deduplicated user–item pairs from training histories and targets. The update inherits the initial embedding parameters; sampled negatives exclude all known training positives from both periods. Full-batch Bayesian personalized ranking uses Adam at 10^{-3} and regularization 10^{-4}. Validation uses the mean history-item embedding to represent each query, including held-out users. Validation is checked every 20 epochs, with patience five and a 300-epoch cap in every domain. These fixed collaborative-feature preparation settings are shared across domains; the public DACT release does not specify a collaborative-feature training configuration for these splits.

C.5.3 Retriever training and selection

The retriever follows the public DACT training defaults: AdamW at learning rate 10^{-4}, batch size 256, weight decay 0.01, and at most 200 epochs. Full-validation cross entropy is evaluated every epoch in batches of 128, using the arithmetic mean of batch losses. Training stops after 15 consecutive epochs without a strictly lower validation loss; the minimum-loss checkpoint is selected, with earlier epochs retained on ties. Each phase uses one optimizer, without a learning-rate scheduler or gradient clipping.

The same fixed recipe trains the encoding-compatible parent from random initialization and adapts its selected checkpoint to the updated codes, with a fresh optimizer for adaptation. The public release supplies the adaptation recipe; we also apply it to parent training on these shared benchmark splits. The four-layer T5 dimensions follow the public model; the per-domain vocabulary and data are those of the common benchmark. The random seed is set before model initialization, and the best checkpoint is saved separately from the current training state.

All nine selected checkpoints are fixed before testing. Decoding uses beam width 40 and the complete current catalog; unavailable future targets remain misses. Three retriever seeds share one tokenizer and collaborative-feature fit per domain, so the reported standard deviations measure retriever-seed variation conditional on those fits. DACT's own parent and adaptation budget differ from the common-warm-up comparisons described in Appendix C.

C.6 Standard Update Strategies and Reformer

This comparison evaluates how tokenizer and retriever updates affect recommendation under the same catalog expansion. Table 1 includes all five standard strategies and Reformer; Tables 16 and 17 give cutoff-20 and old/new-target results. All methods rank the same current catalog for the same test histories, using a width-40 beam and seeds 17, 42, and 2027.

Standard update strategies. Following the tokenizer/GRM comparison in DACT (Feng et al., 2026), Frozen/Frozen retains both the initial tokenizer and retriever; FT/Frozen updates only the tokenizer; Frozen/FT updates only the retriever; FT/FT updates both; and FT/RT updates the tokenizer and trains a new retriever from random initialization. These strategies share DACT's initial residual quantizer, collaborative features, and corresponding initial T5 models. The frozen tokenizer preserves existing codes and assigns new items using its unchanged quantizer. Tokenizer fine-tuning re-encodes all items with the updated quantizer. A fourth token resolves complete-code collisions. The frozen-model and static-assignment rows in the main table instead use the initial models of the construction comparison (Appendix C) and serve as separate controls.

Plain tokenizer fine-tuning minimizes reconstruction, quantization, and collaborative-alignment losses, with alignment coefficient 0.02. It uses 5,000 AdamW steps, learning rate 10^{-4}, batch size 1,024, weight decay 10^{-4}, 500 warmup steps, and gradient clipping at 1. The final tokenizer is used in each strategy. Retriever fine-tuning uses update training examples. Retraining uses the deduplicated union of initial and update training examples: 55,909 for Beauty, 49,661 for Tools, and 46,964 for Toys. Both use AdamW at 10^{-4}, batch size 256, weight decay 0.01, and a 200-epoch maximum. Full update-validation cross-entropy is evaluated each epoch with batch size 128; 15 consecutive epochs without a strict improvement stop training, with earlier ties retained. Frozen retrievers require no additional fitting.

Reformer. We evaluate Reformer's incremental quantizer and identifier assignment (Shi et al., 2025) with the same initial tokenizer and T5 models as the standard strategies. Its three semantic codebooks each gain 16 entries, while existing item identifiers remain fixed. New items train the quantizer with the published reconstruction and quantization losses; warm items establish the frequency counts. We use the released frequency exponents $(0.25, 0, 0)$, power scaling, and Sinkhorn configuration $(0, 0, 0.003)$ with 50 iterations. New-item assignment uses unit distance scales and up to 11 collision-resolution passes; the common fourth-token suffix resolves remaining catalog collisions. The retriever gains 48 semantic-token embeddings while retaining all existing embedding rows. Tokenizer updates use the common 5,000-step configuration above, and retriever adaptation uses the same optimizer and stopping rule as the standard strategies. Thus the comparison preserves Reformer's update mechanisms under the common initialization and training protocol. Hyperparameters are tuned on validation data starting from the published settings.

C.7 Conventional recommendation references

Shared information and evaluation. SASRec, EASE and ItemKNN use the same test histories, target items and current catalog as Table 1. Training combines the available initial and update examples, removing duplicate user–history–target records. This yields 55,909, 49,661 and 46,964 training examples for Beauty, Tools and Toys. Current validation and test examples are not used as supervised training rows. The original chronological splits and maximum 20-item histories are retained. All current-catalog items are ranked, including previously observed items; future-catalog targets remain misses. Equal scores are ordered by item identity. These baselines fit accumulated training data, whereas the generative comparisons use the update procedures specified in their respective appendices.

Selected baseline settings. We tune hyperparameters on validation data, using the RecBole v1.2.1 model and training defaults as starting points.[1] SASRec uses two layers, two attention heads, hidden width 64, feed-forward width 256, GELU, hidden and attention dropout 0.5, layer-normalization tolerance 10^{-12}, and normal initialization with standard deviation 0.02. Its position table supports 50 positions; actual input histories retain the common 20-item maximum. The reference cross-entropy variant predicts the next item from the last observed position. Adam uses learning rate 10^{-3}, batch size 2,048 and no weight decay, scheduler or gradient clipping. Validation MRR@10 is evaluated after every epoch and rounded to four decimal places, following the reference evaluation precision. Ties replace the current best checkpoint and reset patience; training stops after more than ten consecutive strictly worse epochs or at the fixed 300-epoch cap. All nine runs stop by this rule. Seeds are 17, 42 and 2027.

EASE fits its zero-diagonal linear reconstruction model with regularization 250. ItemKNN uses cosine similarity, 100 neighbors per output item, zero shrinkage and the reference 10^{-6} denominator offset. Both use a binary user–item matrix formed only from the available training histories and targets. The matrices contain 132,687, 65,788 and 112,002 observed user–item pairs, respectively. At prediction, each method multiplies a query's binary observed history by the fitted item-weight matrix. This history fold-in uses the same query information as the other recommenders and does not fit evaluation targets. Each deterministic model is fitted once per domain.

The complete-population and cohort results are reported in Appendix E.

D Feasibility Validation and Certificate Coverage

This appendix evaluates assignment feasibility and certificate coverage under fixed scoring models. The exhaustive construction comparison uses 96 deterministic scorers; the independent Beauty parent/update comparison uses seeds 17, 29 and 43. Recommendation quality for the final T5 method is evaluated separately in Appendix E.

D.1 Exhaustive Evaluation of Assignment Invariance

The experiment uses $\mathcal{Z} = \{0, 1\}^3$, widths one and two, 16 seeds, and every nonempty proper old-path subset: the integer masks 1 through 254, inclusive. For each query/scorer, positive integer weights derived from a deterministic SHA-256 key are normalized to exact rational probabilities. The prefix family keys on seed and prefix; the state family additionally keys on the entire ordered beam and its exact accumulated probabilities. Ties in decoding use lexical prefix order. Certification uses strict adjacent gaps and therefore rejects relevant ties rather than relying on this tie-breaking order.

We enumerate all $A \supseteq O$, including $A = O$ and the full codebook. Ground-truth invariance means equality of the complete ordered final list across this family. At width two, 992 prefix-scoring cases and 987 state-scoring cases never return a new path. Only 976 and 965 cases, respectively, preserve the complete ordered output list. A query can have an invariant new-target contribution without an invariant ordered old list. The certificate establishes the stronger list-level claim. Relevant top-rank ties occur in one coverage query/scorer case and nine cohort query/scorer cases; exact arithmetic retains them.

For cohort calibration, use eight seeds, four fixed queries, two new item identities, and old-catalog sizes two, four, or six. Query target identities are $(0, 1, 0, 1)$. Every ordered injection of the two new identities into unused paths defines one shared allocation. The separate optimum allows a different legal injection for each query, whereas the shared optimum maximizes the four-query hit count with a single injection. The bound treats all uncertified queries as potentially correct. All three objectives use the same fixed item labels; changing an occupied path set without tracking identity would not define this test.

An independent implementation shares only the deterministic probability specification with the primary implementation. It separately computes catalog decoding, full-codebook certification, ground-truth invariance, and both allocation optima; all 16,256 coverage rows and 4,032 cohort rows match exactly. The CPU evaluation takes approximately three seconds for this complete finite design.

[1] https://github.com/RUCAIBox/RecBole/tree/v1.2.1

Experimental specifications, per-instance results, and explicit examples of missed invariance and conflicting query optima are included in the supplementary material.

Table 4: Exhaustive calibration of assignment invariance. Each row contains 4,064 single-query cases and 1,008 four-query cohorts. C/I is certified / all-allocation invariant; the final three columns are cohort means in percent. $U_C \geq \mathrm{OPT}_{separate} \geq \mathrm{OPT}_{shared}$ throughout. The width-one equality of C and I is an observation in this finite design.

Scoring	W	C/I	U_C	Separate optimum	Shared optimum
Prefix	1	2,032/2,032	51.56	50.00	39.51
Prefix	2	910/976	78.94	75.40	65.30
Beam state	1	2,032/2,032	54.69	50.00	41.10
Beam state	2	866/965	80.51	75.97	62.85

D.2 Can allocation alone match a model update?

The output-invariance certificate fixes the returned old items on 6,253 of 6,398 VK-LSVD (VK) Primary queries. Thus every legal allocation of that parent has new-item Recall@20 at most $145/6{,}398 = 2.27\%$, compared with the update's 8.82%. The gap excludes allocation alone as an explanation for the update's performance. On the prospective Beauty cohort, two of three updates also exceed their width-20 bounds (Table 5).

Table 5: Can an allocation alone match the updated retriever? Each row evaluates the complete pure-old-input, new-target cohort. $|C|$ queries are certified to contribute zero under every legal allocation of the fixed parent. $U = 1 - |C|/N$ bounds new-item Recall@20. Updated-model recall uses the indicated candidate mapping and the same cohort; all six Tools comparisons and Beauty seed 17 remain below the bound. Beauty uses newly trained models for the chronological comparison. Percent units.

Data/seed	Candidates	$\lvert C\rvert/N$	U	Updated	Gap (pp)	$R > U$
VK/42	Hungarian	6,253/6,398	2.27	8.82	+6.55	Yes
Tools/17	Frozen	172/368	53.26	13.59	-39.67	No
Tools/17	Refreshed	172/368	53.26	9.51	-43.75	No
Tools/29	Frozen	224/368	39.13	14.67	-24.46	No
Tools/29	Refreshed	224/368	39.13	11.96	-27.17	No
Tools/43	Frozen	173/368	52.99	12.50	-40.49	No
Tools/43	Refreshed	173/368	52.99	12.23	-40.76	No
Beauty/17	Fixed content	712/734	3.00	2.72	-0.27	No
Beauty/29	Fixed content	709/734	3.41	5.04	+1.63	Yes
Beauty/43	Fixed content	717/734	2.32	4.63	+2.32	Yes

Tools shows the complementary outcome. All six updates remain below their 39.13–53.26% bounds, leaving allocation-only repair unresolved despite updated-model recovery on parent-certified subsets. Adding mixed-input queries to the VK and Beauty new-target cohorts also makes their bounds inconclusive. On VK, the denominator increases to 13,676 while 6,253 queries remain certified, giving a 54.28% bound versus 14.21% updated Recall@20. On Beauty, the denominator increases from 734 to 1,486. The certified counts remain 712/709/717 for seeds 17/29/43, giving bounds of 52.09/52.29/51.75% versus updated recall of 3.23/5.65/5.59%. Mixed histories contribute the maximum possible value one to these bounds because their inputs can change with the assignment. These comparisons measure the coverage of the certificate as well as the size of the established separation.

D.3 How much does branch analysis improve coverage?

Increasing the beam width exposes a limitation of the original certificate. At width 40, its Beauty bounds rise to 7.49–10.22%, exceeding all three updated Recall@20 values. At width 80, all first-layer checks fail because 64 old root symbols cannot fill 80 strictly ordered old positions. The resulting bounds leave every width-80 comparison unresolved (Appendix D.4).

Final-layer certification and penultimate branching recover useful width-40 bounds. Final-layer certification permits changes in the old ordering while checking that Top-20 contains only old items. Branch analysis additionally enumerates penultimate alternatives after the first two layers are certified. It adds 31/29/23 certified query–seed configurations to the final-layer test, raising the combined counts to 708/705/714 out of 734 queries. The corresponding bounds fall to 3.54/3.95/2.72%, and two updates again exceed them (Table 6). The remaining 26/29/20 queries contribute one each to the bounds.

Table 6: Stronger certificates on the same Beauty cohort ($W = 40$, Recall@20; percent units). All-layer, final-layer, and branch certificates use the same parent checkpoints. Combined counts are over all 734 queries. Updated-model outcomes are reused. Branch analysis is a post-hoc extension of the original width experiment.

Seed	All-layer U	Final-layer U	Added	Combined $\lvert C\rvert$	New U	Updated	$R > U$
17	9.81	7.77	31	708	3.54	3.00	No
29	10.22	7.90	29	705	3.95	5.18	Yes
43	7.49	5.86	23	714	2.72	4.36	Yes

The added configurations comprise 69 distinct test rows across three seeds. Their critical sets contain $q = 1$–6 optional branches: 324 eligibility subsets merge into 317 ordered states. Of these, 83 reference states are reused and 234 require terminal batch calls. This post-hoc extension of the Beauty width experiment uses the existing parents and updates. Tools' fixed-condition applicability analysis is given in Appendix D.5; computation for the broader constructive comparison is reported in Appendix D.6.

D.4 Beam-Width Sensitivity

Widths 20, 40 and 80 were fixed for all three Beauty parent/update pairs before observing the chronological-update test outcomes. All widths use the same checkpoints, full test cohort and seeds. Recall@20 uses the first 20 returned items.

Width 20 reuses the baseline predictions; widths 40 and 80 run new current-catalog/full-codebook parent and current-catalog update passes. Each width is certified under its own scoring and state transitions. Independent reconstruction reproduces the certificate decisions and metrics for the six additional width/seed conditions. Repeated checkpoint inference also checks predictions and scores at both wider widths for seed 17.

The width-80 failure follows from the old codebook's at most 64 distinct first-level symbols, fewer than the 80 strict old extensions required by the condition. This setting measures the resulting structural boundary of the certificate.

D.5 Fixed-count and branch applicability

The Beauty branch study reuses 83 query–seed score arrays containing 317 states and 3,246,080 terminal candidates. At $m = 595$, every repair interval is empty. Capacities range from 4,294,954,577 to 4,294,956,113 and support-group counts from zero to six; the count endpoints are not tight. The exact count criterion adds no certified queries on these states. Its endpoint behavior is tested in the exhaustive finite designs. In the predeclared Tools condition, 172 of 368 queries are certified, 121 fail the common-prefix premise and 75 have a new leaf in the full-reference Top-20. No query en-

Table 7: Complete prespecified beam-width comparison. Recall@20 always uses the first 20 returned items on the same 734 Primary queries. Width 20 uses the baseline predictions. A different width requires a new certificate. All values except counts/widths are percentages or percentage-point gaps.

Seed	W	$\lvert C\rvert/N$	Bound	Updated R@20	Gap (pp)
17	20	712/734	3.00	2.72	-0.27
17	40	662/734	9.81	3.00	-6.81
17	80	0/734	100.00	3.00	-97.00
29	20	709/734	3.41	5.04	+1.63
29	40	659/734	10.22	5.18	-5.04
29	80	0/734	100.00	5.18	-94.82
43	20	717/734	2.32	4.63	+2.32
43	40	679/734	7.49	4.36	-3.13
43	80	0/734	100.00	4.36	-95.64

ters additional branch enumeration. The bound remains 53.26%, above the update's 13.59%, with no exclusion. The complete coverage partition and original score arrays are retained.

D.6 Constructive repair comparisons

D.6.1 Methods and common evaluation rules

The comparison concerns per-query existence: a method receives the designated target identity and may assign it any free leaf while retaining every old item and admitting exactly m new identities. The output is one complete assignment for that query. This task measures constructive reachability for a supplied target identity. The comparisons reuse previously evaluated checkpoints. Method choices, finite designs, query selection rules and computation limits were fixed before the comparative executions.

The initial finite allocation uses the first m unoccupied leaves in numerical order and assigns the target to the first one. Beauty uses its reference assignment. Random search draws four legal m-leaf catalogs, independently assigns the target to a selected new leaf, and reports whether any trial retrieves it. All four trials are evaluated, including after success.

High-score allocation selects the highest-scoring new leaf in the full-codebook reference expansion and fills the remaining slots in numerical order within the reference-compatible universe: optional prefixes above its beam cutoff that are absent from that state remain disabled. It does not enforce support for selected optional prefixes, so decoding the catalog may change the beam. Reference-state exact construction applies the complete support and count criterion while retaining only the full-reference penultimate state. Full construction examines that state first and then the remaining states in deterministic order, returning the first verified witness or certifying failure after complete enumeration. The lexical-padding ablation retains exact state and target selection and support choices but fills remaining slots in numerical order instead of preferring non-outrankers. Every returned mapping is evaluated by the complete decoder.

D.6.2 Finite end-to-end comparisons and condition ablations

We use three-layer alphabets $(3, 2, 2)$ and $(4, 2, 2)$ with widths two and three. Every root has one mandatory old leaf at suffix $(0, 0)$, so the root beam is common to every containing catalog. Each shape has 24 deterministic scoring seeds for each of two families: prefix-local scoring and scoring that depends on the complete ordered penultimate state. Hash-derived positive integer weights are normalized to exact rational probabilities; ties use lexical path order. There are 96 scorers, with 9 or 12 optional leaves per scorer.

Independent full beam decoding exhausts all 221,184 containing catalogs. For every positive admission count and $K \in \{1, 2\}$, it determines whether any assignment can retrieve the target. This yields 2,016 conditions, 1,816 feasible. The finite columns of Table 8 use these feasible denominators, 882

Table 8: Constructive comparisons. Finite columns report successful target retrieval among all feasible conditions; Beauty columns report repairs among all remaining common-prefix queries (40/44/27), each initially missed. The initial allocation is lexical append in finite problems and the reference mapping on Beauty. Random search reports success in any of four trials. Each repair uses its own assignment.

	Finite scoring family		Beauty seed		
Method	Prefix-local	Complete-state	17	29	43
Initial allocation	170/882	107/934	0/40	0/44	0/27
Random, four trials	488/882	494/934	0/40	0/44	0/27
High-score allocation	882/882	733/934	9/40	15/44	3/27
Reference-state exact	850/882	858/934	9/40	15/44	3/27
Full construction	882/882	934/934	9/40	15/44	3/27
Lexical padding	882/882	934/934	9/40	15/44	3/27

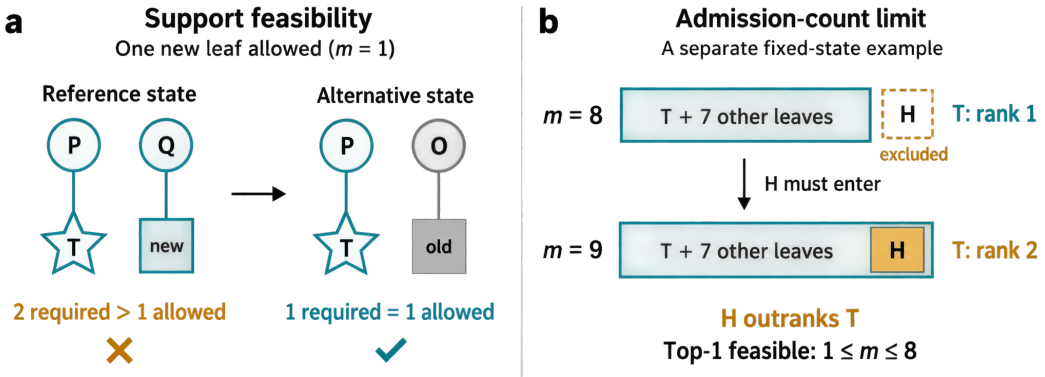


Figure 4: Support and admission-count mechanisms in two exact finite examples. **a**, At $m = 1$, reference prefixes $P = (1, 1)$ and $Q = (2, 1)$ each require a new supporting leaf. The alternative replaces Q by the old-supported prefix $O = (1, 0)$, allowing target leaf $T = 6$ to be retrieved with one admission. The left drawing shows required supports, not a feasible one-admission catalog. **b**, In a separate fixed state, the nine optional leaves include target $T = 11$ and its higher-scoring competitor $H = 10$. At $m = 8$, excluding H gives target rank one; admitting all nine leaves forces H into the catalog and lowers the target to rank two. Exhaustive decoding verifies Top-1 feasibility exactly for $1 \leq m \leq 8$; at $m = 0$ the target is absent. Mandatory old leaves remain fixed and are omitted in (b).

for prefix-local and 934 for complete-state scoring. Full construction also correctly rejects all 200 infeasible conditions. High-score allocation solves all feasible prefix-local conditions in this finite design; its 201 misses all occur under complete-state scoring. Reference-state exact construction misses 32 prefix-local and 76 complete-state conditions because another beam is needed. Lexical padding has no observed success penalty in this comparison.

Condition ablations test a different endpoint: feasibility of a specified leaf at a specified state and count. Removing support groups also removes their contribution to required counts and forced outrankers. The other two ablations remove only the lower or upper count endpoint, retaining the compatible-state capacity for the latter. Across 24,630 decisions, these removals produce 1,586, 659, and 802 false positives, respectively, with no false negatives. The full criterion has neither error. These deliberately incomplete predicates serve as diagnostic checks of the necessary conditions. An independent local ranking experiment checks 193,408 decisions and 24,252 constructed witnesses. It enumerates 67,392 catalogs across 2,400 ranking instances with local universes of six or eight leaves, disjoint two-leaf support groups, every old mask and eligible group subset, four ranking seeds, and $K \in \{1, 2, 3, 4\}$. Two leaves are outside the terminal expansion. It tests forced support, insufficient count and off-beam padding with no false-positive or false-negative decisions.

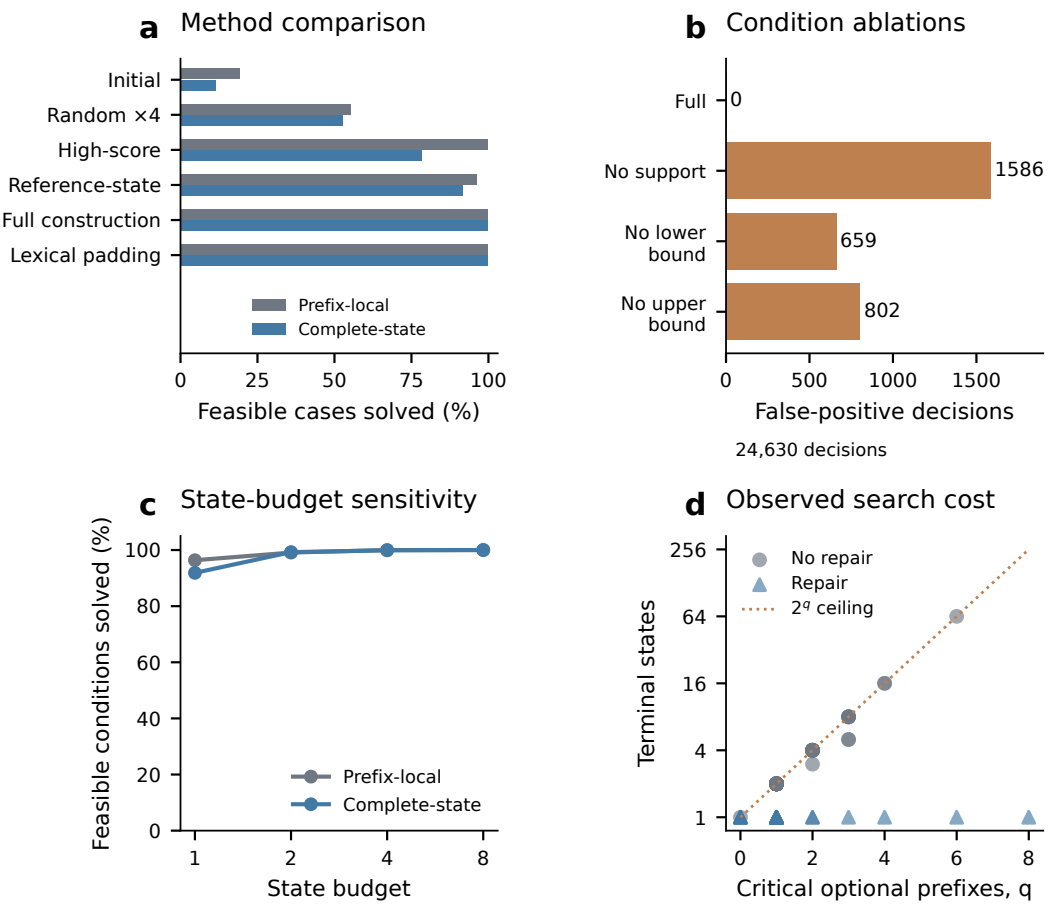


Figure 5: Method comparisons, component removals, and search cost. **a**, Success among feasible conditions in each finite scoring family (882 and 934); bars are exact descriptive proportions. **b**, False feasibility decisions among 24,630 state–target–count conditions after removing support or count endpoints. **c**, Feasible conditions solved with a fixed terminal-state budget; unfinished enumeration leaves unresolved cases rather than certifying failure. **d**, States evaluated for all 111 trained-model comparisons, with the 2^q ceiling. Successful construction may stop at the first state, whereas failure requires enumeration. Repeated conditions and query–seed pairs are within-design measurements.

Figure 4 uses two deterministically selected examples from this enumeration. In panel a, $m = 1$ cannot support both optional prefixes in the reference state; replacing one with an old-supported prefix allows target leaf 6 to be retrieved. In panel b, the specified target is leaf 11, $K = 1$, and the required group is $\{10, 11\}$. Its predicted interval is $[1, 8]$. At $m = 9$, leaf 10 must be included and outranks the target. At $m = 0$, the target cannot be admitted, so no rank is plotted.

D.6.3 Paired evaluation with trained Beauty models

We reuse the three Beauty parents at width 40 and $K = 20$, with 11,183 old items and 595 new items. Each seed retains all 734 Primary queries in the accounting. Before new comparisons, final-layer certification resolves 677/676/691 queries, while 17/14/16 fail the common first-two-layer condition. All remaining 40/44/27 queries enter the comparison, including cases with and without a new leaf in the reference Top-20. The selected query set is retained after successful repairs. Every selected query is initially missed under its reference assignment.

Each parent runs on an NVIDIA RTX 3090 with PyTorch 2.6.0+cu124 and Transformers 5.9.0, deterministic float32 scoring, disabled TF32, and the original 64-query batches and within-batch slots. CuBLAS uses the deterministic workspace configuration `:4096:8`. The complete pre-branch ob-

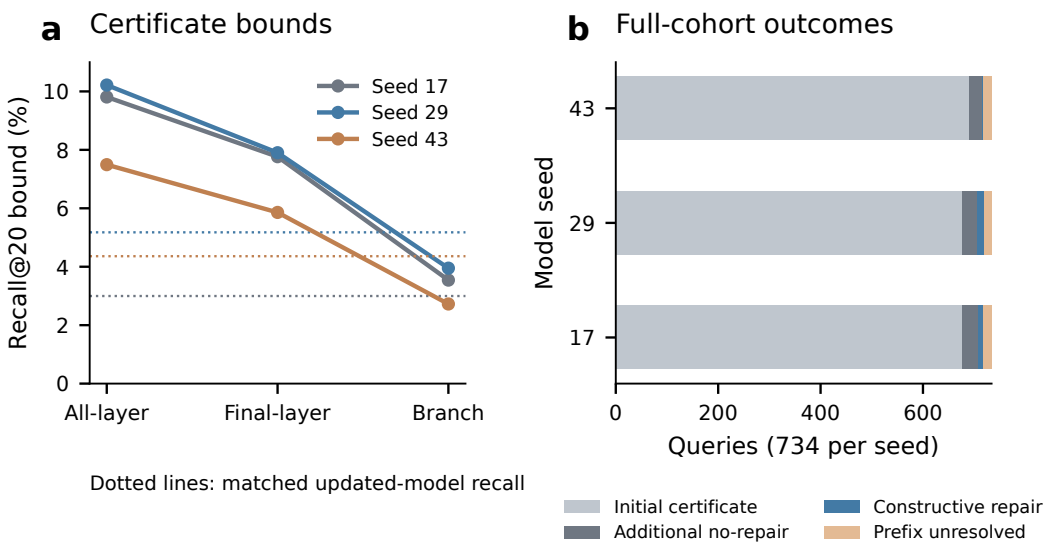


Figure 6: Certificate strength and complete-cohort accounting. **a**, Original all-layer, final-layer, and branch bounds at Beauty width 40; dotted lines are the matched updated-model recalls. **b**, Every Primary query is retained after constructive comparison: initial certification, additional no-repair decisions, query-specific successful constructions, or unresolved common prefixes. The seed-43 partition includes the additional tied-old-score case described in the text. A different assignment may realize each repair; the blue portion aggregates per-query repairs.

servations and reference terminal scores agree exactly with the corresponding recorded GPU evaluations. Branch continuations retain the actual caches. Enumeration allows at most eight critical optional prefixes and 256 states per query; no selected query exceeds these limits. Every method uses the same model, input, catalog size, old identity mapping, and numerical execution shape.

Independent checks verify 636 complete legal assignments across all methods and random trials. For every full-construction witness, direct decoding reproduces the constructed state, terminal scores and predicted target rank.

Full construction repairs 9/15/3 queries, matching both high-score allocation and reference-state exact construction. With four trials per query, random search retrieves none of the 111 targets. All 27 successful repairs occur in the reference state. Five need no new-prefix support; the other 22 require one to seven groups (the counts for 1/2/3/4/6/7 groups are 16/2/1/1/1/1).

The remaining 84 compared queries have no repair in the enumerated family. All 318 of their terminal states have at least 20 strictly higher-scoring mandatory old leaves than the best new leaf, so internal old-score ties do not change these negative decisions. The rank-based comparison includes one seed-43 query outside the certificate coverage in Table 6: its old scores contain an internal tie, but 49 old leaves strictly outrank its best new leaf.

D.6.4 Budgets, computation, and statistical units

The finite budget comparison limits the deterministic state order to 1, 2, 4, or 8 terminal evaluations and retains unknown outcomes when the budget does not complete a negative decision. The trained-model comparison evaluates 345 states across 111 queries, including 111 reference states and 234 additional terminal continuations. Critical-set sizes range from zero to eight. Positive cases stop at their verified first-state construction; negative cases require all reachable states. Figure 5 shows these search outcomes; the sufficient certificate's single-pass cost is measured separately.

The three complete paired studies take 42.88/52.66/28.58 seconds, including model loading, all comparison methods, validation, and output recording, but excluding interpreter and library startup. Peak allocated GPU memory is 5.67 GiB per study. The shared reference reconstruction uses four decoder steps per compared query. Together, the methods use 1,283/1,547/836 decoder batch steps

in the three studies. Per-method synchronized execution times, shared-reference times, search and recording times, and forward counts are supplied with the individual results. The methods have different computation costs: random search uses four full decodings, while exact search depends on the state count.

For the finite designs, summaries distinguish the 96 scorers, count/cutoff conditions and state-specific decisions. Trained-model summaries distinguish the three parents and their query–seed pairs. Random trials and repeated states are within-study measurements. All summaries are descriptive; no significance test is applied.

D.7 Decoder Implementation and Experimental Details

D.7.1 Parent/update models and data provenance

The Beauty models use the chronological split and tokenizer in Appendix C.2. Each parent is trained from scratch. AdamW uses learning rate 3×10^{-4}, weight decay 0.01, gradient clipping at one, and batch size 256. All 40 parent and 20 update epochs run; minimum validation teacher-forced loss selects each checkpoint. The update starts at its selected parent, uses a new optimizer, and includes eligible old training examples. Checkpoint selection includes epoch zero for both parent and updated models. Validation loss selects parent epochs 30/23/24 and update epochs 11/17/10.

The VK and Tools certificate experiments use fixed parent/update model pairs. The VK training universe contains 65,659 IDs, including future IDs used as negatives in dense training; decoding permits 62,398 current items. Its released validation data reuse gap-training interactions, so the epoch-28 endpoint is fixed independently of those data. Certificate metrics use explicit query means and capped labels. Multi-target old/new slices can overlap. Each Tools parent supplies frozen- and refreshed-candidate updates; both are evaluated against that parent's certificate on the same Primary cohort.

D.7.2 Decoder configuration

The VK GPT and Tools T5 certificate experiments use four generated SID tokens, width 20, and return 20 paths. These are distinct from the width-40 T5/LC-Rec recommendation comparisons. The GPT codebook blocks have 512 tokens per position with zero bias; the T5 blocks have 256 tokens with bias one. Log-softmax is computed over the full model vocabulary before the sole catalog mask. No allowed-child renormalization, sampling, repetition penalty, or variable-length valid EOS completion is introduced. The effective generation settings have length penalty one.

The first beam has score zero and other initial copies have score -10^9. Valid top-20 expansions are checked to remain above inactive sentinels. At layers 1–3, finite selected expansions do not finish, and the inactive completed pool cannot change the live beam or trigger stopping. A catalog with few legal backup extensions may fill some top-40 positions with invalid/EOS sentinels. These cannot enter the finite top 20 or its effective completed pool. All paths terminate at layer four and share the same length normalization. The subsequently unused running state can differ because of sentinel rounding ties; the proof concerns effective pre-score states and final outputs, not those dead temporaries.

The T5 evaluation additionally tracks the encoder output and the self-/cross-attention caches. Every T5 generation call performs one encoder and four decoder forwards. Both models run in evaluation mode without batch normalization. Fixed-shape query separability and deterministic scoring are premises; matching repeated cache fingerprints is finite implementation evidence.

D.7.3 Numerical validation

The implementation is checked separately from the logical criterion. An independent exhaustive decoder check covers 12,240 binary three-level cases; its 2,975 certificates agree across all 41,552 containing catalogs. Reference/full-codebook comparisons, repeated executions and independently reconstructed layer decisions check the saved VK and Tools trajectories.

Certificates use scores and effective states from the actual cached decoding pass. A separate full-forward evaluation can differ beyond the fixed 2×10^{-5} tolerance, so it is not substituted for the cached score function.

D.7.4 METRIC AND COHORT DEFINITIONS

VK histories take the last 20 items from time parts $[0, 9)$; targets are in $[9, 10)$. Old items occur in $[0, 8)$, admitted new items in $[0, 9)$ but not the old catalog. The test set also has targets not yet admitted; these remain in whole-catalog evaluation with zero contribution. Recall counts unique predicted hits against the label set and divides by the count of label occurrences capped at 20. New-target cohorts retain only their new-label component for that metric. Multi-target queries can belong to more than one target-kind cohort. Tools uses the local 0.7-to-0.8 catalog transition and one target per query.

The primary bound comparison uses every pure-old query with an admitted new target. Parent-certified groups are secondary subsets defined from the parent before examining updated-model results. Tools' three parent certificates agree across the frozen and refreshed mappings, as implied by reference independence for those inputs. They certify 4,274/5,091, 4,459/5,091, and 3,951/5,091 pure-old queries, respectively. None of the six updates exceeds the bound on the complete Primary cohort.

Within each certificate comparison, parent and updated checkpoints use identical numerical settings.

D.8 COST OF INVARIANCE CHECKING

Streaming evaluation reproduces every ordered prediction, layer decision and margin for 39,219 VK and 5,360 Tools queries without an additional model forward pass. The first and last batches of each architecture receive one A/B warmup and a fixed ABBAABBA timing sequence: 32 timed calls, four repeats per mode and shape. Per-call wall time includes score checks and synchronization, but excludes input transfer, output copying and file writing. Both modes retain the same old-prefix index; peak-memory differences exclude its shared resident cost. Index storage is 1,074,560 bytes on VK and 225,688 bytes on Tools. Table 9 reports the four shapes; branch enumeration incurs separate terminal continuations and does not share this same-pass cost guarantee.

Table 9: Same-pass checker cost on two NVIDIA RTX 3090 GPUs. A is uninstrumented decoding and B adds the checker. Each fixed shape has four timed A/B calls after one warmup each; mean wall time is shown. Both modes retain the same old-prefix index. Δpeak is the difference in peak allocated memory in that environment, not total temporary or index memory. Negative timing change is not evidence of speedup.

Architecture	Batch	A (ms)	B (ms)	Change	Δpeak (KiB)
GPT / VK	256	1336.52	1337.75	+0.09%	20
GPT / VK	51	271.23	271.97	+0.27%	4
T5 / Tools	128	1878.40	1882.54	+0.22%	10
T5 / Tools	112	1649.98	1643.02	-0.42%	9

E EXTENDED T5 RESULTS AND COMPONENT ANALYSIS

This appendix evaluates the final T5 models for seeds 17, 42 and 2027. Parameter and budget sweeps center on the configurations in Table 3; matched component controls inherit each full-method run's adaptation and scoring settings.

E.1 SENSITIVITY TO SCORING PARAMETERS AND EVALUATION BUDGET

We hold the nine trained models, assignments and collaborative predictors fixed and vary scoring parameters around the configurations in Table 3. For each dataset and seed, we multiply one of λ, γ or b by $\{0.5, 0.75, 1, 1.25, 1.5\}$ while holding the other two fixed. Beam width is 40, the additional-item budget is 80, and the completion batch size is 20. Evaluation uses the same 1,040 Beauty, 3,007 Tools and 774 Toys validation queries as parameter selection. Figure 7 shows paired NDCG@10 changes from each seed's adopted configuration; all four ranking metrics are retained in the source data.

The response differs across parameters and datasets. Halving λ changes mean NDCG@10 by -0.129, -0.145 and -0.013 percentage points on Beauty, Tools and Toys. On Beauty, halving b increases it by 0.100 points, whereas multiplying b by 1.5 decreases it by 0.166 points. Across the tested one-parameter perturbations, mean changes range from -0.166 to $+0.100$ on Beauty, -0.145 to $+0.029$ on Tools, and -0.055 to $+0.035$ on Toys. Individual seeds can respond in opposite directions, as seen for Toys at half the fusion weight.

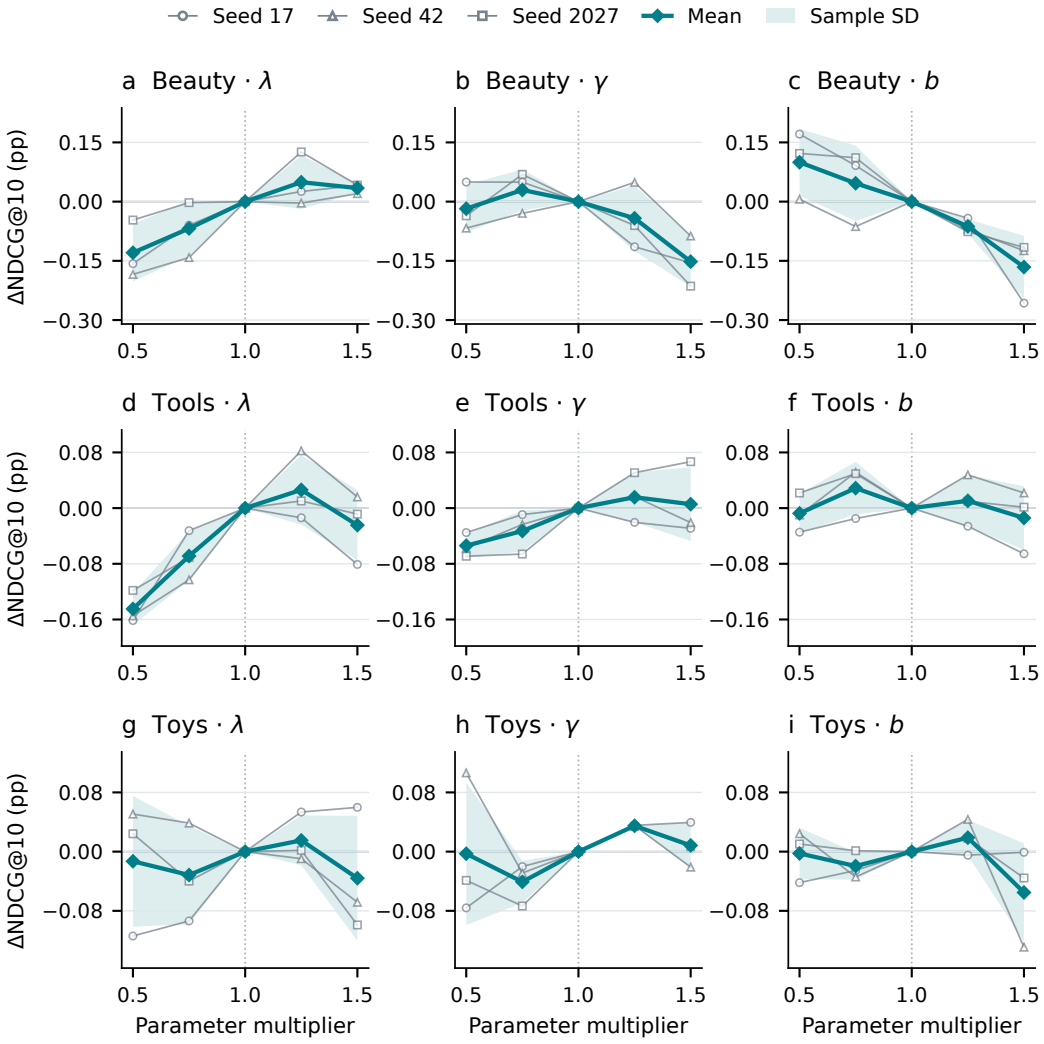


Figure 7: Scoring sensitivity around the nine final T5 configurations. Rows show Beauty, Tools and Toys; columns vary λ, γ and b one at a time. The horizontal coordinate multiplies each seed's own selected parameter. Vertical coordinates are paired validation NDCG@10 changes in percentage points from multiplier one. Gray paths show seeds 17/42/2027; teal diamonds and bands show their mean and sample standard deviation. Straight segments join the five measured multipliers. Other parameters, models, mappings and the 80-item completion budget remain fixed.

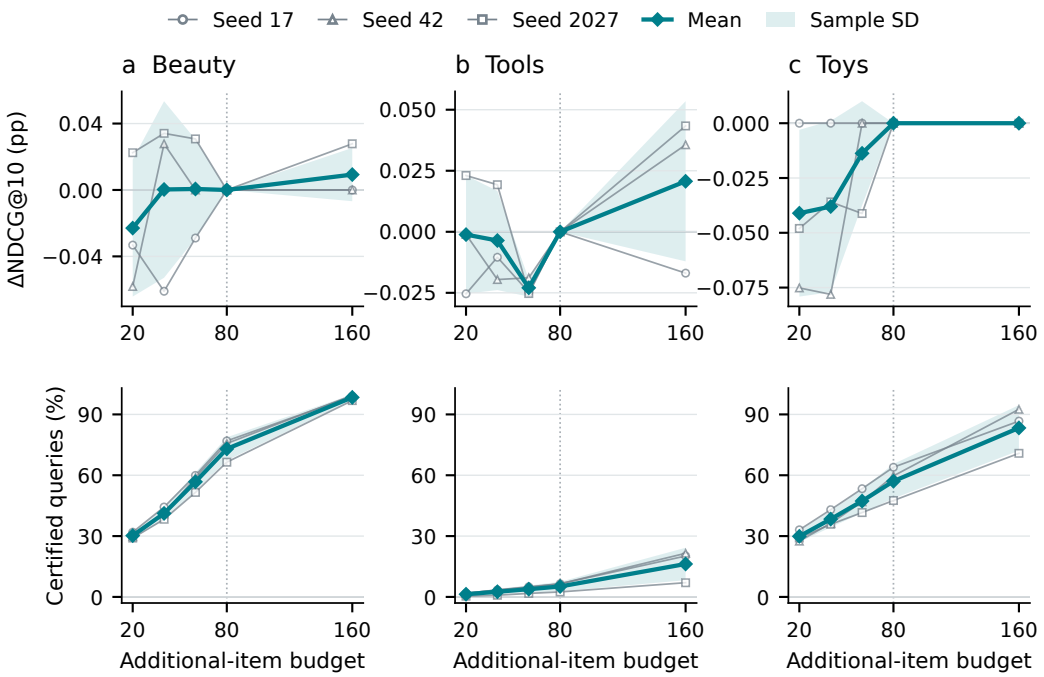


Figure 8: Validation sensitivity to the additional-item budget at fixed final scoring parameters. Top: paired NDCG@10 change from budget 80. Bottom: the fraction satisfying the numerical Top-20 certificate. Gray paths show seeds 17/42/2027; teal diamonds and bands give means and sample standard deviations. Points correspond to budgets 20/40/60/80/160; straight segments connect observations. The budget caps additional item evaluations; actual counts and all four ranking metrics appear in Table 10.

Table 10: Validation ranking and work as the additional-item budget varies. Means over seeds 17/42/2027 with fixed final models, mappings and scoring parameters. R/N denote Recall/NDCG (%); Added counts actual additional likelihood evaluations, and Cert. is numerical Top-20 certification (%). Per-seed variation is shown in Figure 8.

Dataset	Budget	R@10	N@10	R@20	N@20	Added	Cert.
Beauty	20	5.641	3.071	8.910	3.901	19.79	30.22
	40	5.705	3.095	9.103	3.956	33.74	41.19
	60	5.705	3.095	9.167	3.972	45.51	56.76
	80	5.705	3.094	9.167	3.971	54.15	73.01
	160	5.737	3.104	9.199	3.981	64.71	98.40
Tools	20	5.421	3.082	7.205	3.530	19.99	1.33
	40	5.399	3.079	7.250	3.547	39.72	2.55
	60	5.332	3.060	7.250	3.542	59.21	3.81
	80	5.410	3.083	7.383	3.576	78.45	5.14
	160	5.476	3.104	7.394	3.584	152.46	16.24
Toys	20	4.651	2.376	7.278	3.037	18.11	29.89
	40	4.651	2.379	7.364	3.063	32.13	38.37
	60	4.737	2.404	7.450	3.084	44.45	47.29
	80	4.780	2.417	7.407	3.076	55.00	57.06
	160	4.780	2.417	7.321	3.058	79.84	83.33

Budget and certification. At the adopted scoring parameters, we vary only the maximum number of additional likelihood evaluations over $\{20, 40, 60, 80, 160\}$ (Figure 8, Table 10). Raising the budget from 80 to 160 increases mean numerical Top-20 certification from 73.01/5.14/57.06% to 98.40/16.24/83.33% on Beauty/Tools/Toys. Mean additional evaluations rise from 54.15/78.45/55.00 to 64.71/152.46/79.84. The corresponding NDCG@10 changes are $+0.009$, $+0.021$ and zero percentage points. Toys Recall@20/NDCG@20 decrease from 7.407/3.076% to 7.321/3.058%. Thus the larger budget certifies more rankings, while its recommendation effect remains dataset- and seed-dependent. All models and the main-result configurations remain fixed throughout these measurements.

E.2 Calibration tradeoffs and candidate effort

Figure 9 shows the old/new-item tradeoffs between global correction and calibrated scoring across all datasets and three retriever seeds. Figure 10 reports their candidate effort and measured GPU inference costs.

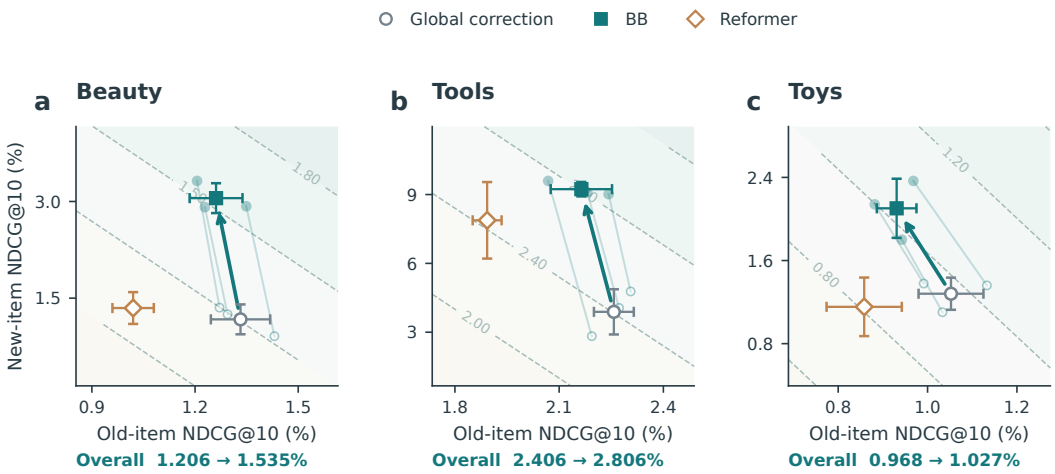


Figure 9: Calibration redistributes old- and new-item ranking quality under the main-result configurations with seeds 17/42/2027. Thin paths connect global correction to BB for each of three retriever seeds; large markers and error bars show means and sample standard deviations. Dashed contours give overall test NDCG@10 (%) using the complete query population. Arrows connect matched global/calibrated settings; all methods use seeds 17/42/2027.

Table 11: Candidate accounting and measured GPU inference cost, averaged across seeds 17/42/2027. Initial/added leaves and numerical Top-20 certification rates use complete test sets. Wall time per query uses the same first 128 histories per domain and seed, three repetitions, RTX 3090 neural inference and two CPU threads; it includes fresh beam, item scoring, encoder and completion. Timing excludes loading and warmup (Appendix B.4).

Domain	Priority	Initial	Added	Certified (%)	ms/query
Beauty	BB (Ours)	126.37	51.08	77.64	10.16
Beauty	Collaborative priority	126.37	60.69	45.40	10.50
Tools	BB (Ours)	40.17	78.70	4.45	10.64
Tools	Collaborative priority	40.17	79.77	0.71	10.63
Toys	BB (Ours)	96.08	52.75	61.53	10.96
Toys	Collaborative priority	96.08	63.65	31.95	11.98

Runtime and candidate counts. On Beauty, mean total wall time is 10.16 versus 10.50 ms/query for bound and collaborative priority; the completion stage alone is 4.10 versus 4.45 ms/query. On

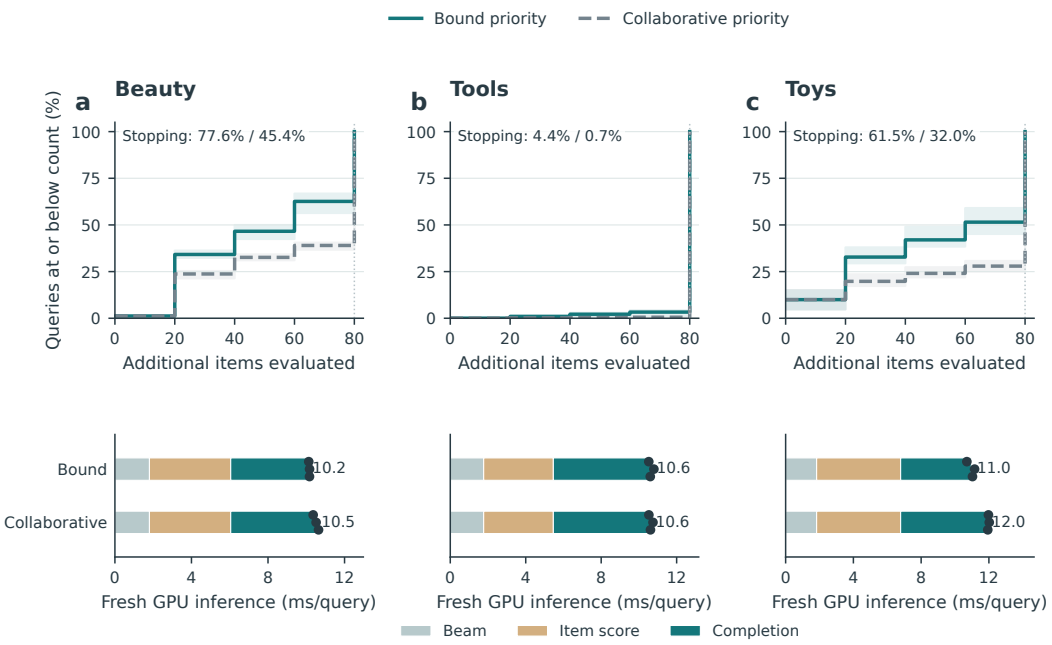


Figure 10: Candidate effort and measured inference cost for seeds 17/42/2027 under the same calibrated score and 80-item maximum. Top: cumulative evaluated-item counts over complete test sets, with seed means and three-seed ranges. Both curves reach 100% at the budget; this includes budget exhaustion. Annotations give the separate numerical Top-20 certification rates in bound/collaborative order. Bottom: fresh GPU inference wall time on the same 128 histories per domain and seed, decomposed into beam, item scoring and completion costs. Bars show seed means and dots show per-seed totals after averaging three repetitions. Neural inference uses an RTX 3090; item correction and bookkeeping use two CPU threads.

Toys, total time is 10.96 versus 11.98 ms/query, while Tools is nearly unchanged (10.64 versus 10.63 ms/query). Total latency also reflects beam decoding, item scoring, batch utilization and host-side work. The reported times are batched means on the fixed timing subsets.

Matched component controls. Tables 2, 12 and 13 use seeds 17/42/2027 and the corresponding settings in Table 3. These matched controls share the BB runs in the main comparison, as do the GPU timings above.

Table 12: Matched T5 component controls at cutoff 20: Recall and NDCG (%), mean ± sample s.d. over seeds 17/42/2027. Each removal inherits its corresponding full-method settings; BB uses the runs in Table 1. Initial-pool reranking uses E_0 without additional evaluations, whereas without completion uses only the returned beam.

	Beauty		Tools		Toys	
Variant	R@20	N@20	R@20	N@20	R@20	N@20
Generator only	4.045±0.089	1.547±0.073	4.932±0.163	2.236±0.123	2.979±0.148	1.124±0.086
Without collaborative score	4.040±0.082	1.546±0.073	4.956±0.173	2.242±0.124	2.979±0.148	1.123±0.085
Without completion	4.663±0.250	1.858±0.091	5.585±0.235	2.883±0.050	3.240±0.115	1.282±0.060
Initial-pool reranking	4.663±0.263	1.867±0.100	5.585±0.235	2.883±0.050	3.383±0.042	1.329±0.043
Without construction	5.135±0.322	1.999±0.116	6.430±0.121	3.180±0.010	3.394±0.101	1.348±0.046
Item correction only	4.372±0.034	1.642±0.025	4.558±0.814	2.130±0.410	2.043±0.028	0.831±0.007
Collaborative priority	5.153±0.157	2.022±0.072	6.437±0.049	3.168±0.039	3.426±0.051	1.349±0.036
Without calibration	4.158±0.180	1.606±0.061	5.808±0.138	2.752±0.109	3.346±0.064	1.296±0.060
BB (Ours)	5.208±0.189	2.035±0.079	6.437±0.112	3.179±0.009	3.458±0.109	1.352±0.030

Table 13: Subgroup NDCG@10 (%), averaged over seeds 17/42/2027 using the full-method and matched-control settings in Table 3. New denotes current-catalog new targets; Primary additionally requires old-only histories. Current combines old and new targets. Future-catalog targets remain misses in the main-table denominator.

Domain	Variant	Old	New	Current	Primary
Beauty	Generator only	1.271	0.943	1.200	0.674
Beauty	Without construction	1.269	2.940	1.633	2.010
Beauty	Item correction only	0.458	4.063	1.243	3.463
Beauty	Without calibration	1.332	1.171	1.297	0.383
Beauty	BB (Ours)	1.261	3.052	1.651	2.290
Tools	Generator only	1.598	4.367	1.850	4.457
Tools	Without construction	2.164	9.277	2.810	9.372
Tools	Item correction only	1.113	8.566	1.790	8.776
Tools	Without calibration	2.258	3.888	2.406	4.221
Tools	BB (Ours)	2.164	9.234	2.806	9.355
Toys	Generator only	0.848	1.089	0.896	1.184
Toys	Without construction	0.971	1.919	1.159	1.596
Toys	Item correction only	0.080	3.321	0.725	3.279
Toys	Without calibration	1.052	1.281	1.098	1.092
Toys	BB (Ours)	0.931	2.103	1.164	1.907

E.3 New-target ranking, certification and query effects of construction

This comparison evaluates the selected shared assignment and its subsequent adaptation against the paired static branch. Both use the adaptation setting, selected checkpoints and scoring parameters in Table 3. Queries are paired by history and target; the Beauty/Tools/Toys denominators are 7,335/5,360/6,266, including unavailable future targets.

New-target ranking. On Beauty and Toys, new-target NDCG@10 increases by 3.82% and 9.56%, respectively (Table 13). The gains are larger for Primary queries, whose new targets follow entirely old-item histories: 2.010% to 2.290% on Beauty and 1.596% to 1.907% on Toys, or 13.95% and 19.42% relative improvements. These queries account for 734/7,335 and 571/6,266 of the full test populations. Old-target NDCG@10 decreases by 0.60% and 4.14%, leaving overall improvements of 1.13% and 0.37%. For Beauty, the new- and old-target contributions to the overall NDCG@10 difference are $+0.02273$ and -0.00557 percentage points; for Toys they are $+0.03216$ and -0.02838. These contributions use the full-query denominator and explain how the group effects combine.

Certification and evaluation effort. On Beauty, the constructed/adapted branch increases numerical Top-20 certification from 74.10% to 77.64%, while reducing additional evaluations from 52.59 to 51.08 per query (2.86%). On Toys, certification increases from 51.66% to 61.53%, with evaluations falling from 57.28 to 52.75 (7.90%). Tools averages 78.70 additional evaluations in both branches, with certification rates of 4.43% and 4.45%. These full-test measurements use the same 80-item maximum and numerical stopping rule. Figure 2b shows the Beauty curves and their paired certification differences.

Training support and selected assignments. Tools supplies 133 new-target training contexts covering 74 of 158 new items, compared with 2,993 covering 532 of 595 on Beauty and 4,950 covering 944 of 1,025 on Toys (Appendix C). Its selected assignment remains unchanged in seeds 17 and 2027 and changes two identities in seed 42. In the first two seeds, constructed and static predictions coincide; seed 42 loses one Top-10 hit. This combination of sparse new-target supervision and little assignment change distinguishes Tools from the two datasets with larger cohort and certification gains.

Paired query gains and losses. Table 14 resolves the aggregate effects by seed. Beauty seed 42 recovers 43 Top-10 hits and loses 34, and its NDCG@10 gain outweighs the small negative overall differences in seeds 17 and 2027. Seed 17 recovers 34 hits and loses 42, with positive and negative NDCG@10 contributions of $+0.3591$ and -0.3643 percentage points. Toys seed 17 recovers 51 hits and loses 46; its positive NDCG change exceeds the negative changes in seeds 42 and 2027. Seed 42 has two recovered and two lost hits, with a negative NDCG change. Primary effects also vary: the three paired NDCG@10 differences are $+0.24070$, $+0.61568$ and -0.01528 percentage points on Beauty, and $+0.83901$, 0 and $+0.09132$ on Toys.

The selected maps directly recode a history or target in 61/32/88 Beauty queries and 24/5/29 Toys queries. Ranking changes extend beyond these sets: Beauty seed 17 recovers 33 and loses 40 hits among queries with unchanged history and target codes, whose competing candidate assignments and adapted scores can still differ.

Table 14: Paired query effects of final construction and adaptation. Recovered/lost count Top-10 misses becoming hits and hits becoming misses, relative to without construction. ΔN^+ and ΔN^- sum positive and negative query NDCG@10 changes and divide by all test queries, in percentage points; their sum is Net. Recoded counts queries whose history or target contains an item with a changed code.

Dataset	Seed	Recoded	Recovered	Lost	ΔN^+	ΔN^-	Net
Beauty	17	61	34	42	+0.3591	-0.3643	-0.0052
	42	32	43	34	+0.3398	-0.2817	+0.0581
	2027	88	4	8	+0.0447	-0.0461	-0.0014
Tools	17	0	0	0	0.0000	0.0000	0.0000
	42	33	0	1	+0.0113	-0.0232	-0.0118
	2027	0	0	0	0.0000	0.0000	0.0000
Toys	17	24	51	46	+0.4449	-0.4142	+0.0307
	42	5	2	2	+0.0133	-0.0319	-0.0186
	2027	29	27	21	+0.2142	-0.2149	-0.0007

E.4 Query recovery under the combined score

Reuse and additional scoring. Initial-pool reranking orders all fully scored leaves E_0 using the same model, map and combined score as BB, with no additional likelihood evaluations. Tables 2 and 12 place this control between returned-beam reranking and full completion. Table 15 reports the two paired increments. Reuse raises mean NDCG@10 by 0.009/0.000/0.028 percentage points on Beauty/Tools/Toys; additional scoring contributes 0.119/0.249/0.023 percentage points. Tools has identical four-metric results under returned-beam and initial-pool reranking for each seed. Its mean initial pool contains 40.17 items, close to the 40 returned candidates; Beauty and Toys contain 126.37 and 96.08. The contribution of reuse is largest on Toys, where both stages improve the dataset means.

Full completion exceeds initial-pool reranking on all four dataset-mean metrics and on Recall@10 and NDCG@10 in all nine runs. The paired NDCG@10 increases on Toys seeds 17 and 2027 are small, at 0.006913 and 0.003187 percentage points. At cutoff 20, Toys seed 42 has four fewer hits (213 to 209; -0.063837 percentage points), while its NDCG@20 increases by 0.004283 percentage points. These paired contrasts capture both recovered and displaced hits under the fixed score.

Candidate access across cutoffs. Figure 11 compares the generator, returned-beam reranking, initial-pool reranking and full completion at every cutoff from 1 to 20. Its lower panels separate hit turnover from initial-pool reuse and additional scoring. At cutoff 10, reuse adds a net 0.09/0.00/0.64 hits per 1,000 queries on Beauty/Tools/Toys; additional scoring contributes 3.36/6.65/0.74. Each increment includes recovered and displaced hits under the same model, mapping and score.

A fixed-score recovery case. Figure 12 shows full-range and zoomed views of one query under the same score. For Beauty seed 17, we select among the 16 new-target queries whose targets are

Table 15: Paired decomposition of candidate-access gains, in percentage points: mean ± sample s.d. across seeds 17/42/2027. Reuse is initial-pool reranking minus returned-beam reranking; additional scoring is full completion minus initial-pool reranking. The model, map, combined score and queries are fixed within each seed.

Dataset	Contrast	ΔR@10	ΔN@10	ΔR@20	ΔN@20
Beauty	Reuse	0.009±0.021	0.009±0.012	0.000±0.047	0.009±0.010
Beauty	Additional scoring	0.336±0.119	0.119±0.050	0.545±0.076	0.169±0.027
Tools	Reuse	0.000±0.000	0.000±0.000	0.000±0.000	0.000±0.000
Tools	Additional scoring	0.665±0.088	0.249±0.037	0.852±0.191	0.296±0.045
Toys	Reuse	0.064±0.058	0.028±0.018	0.144±0.073	0.047±0.019
Toys	Additional scoring	0.074±0.060	0.023±0.032	0.074±0.121	0.022±0.016

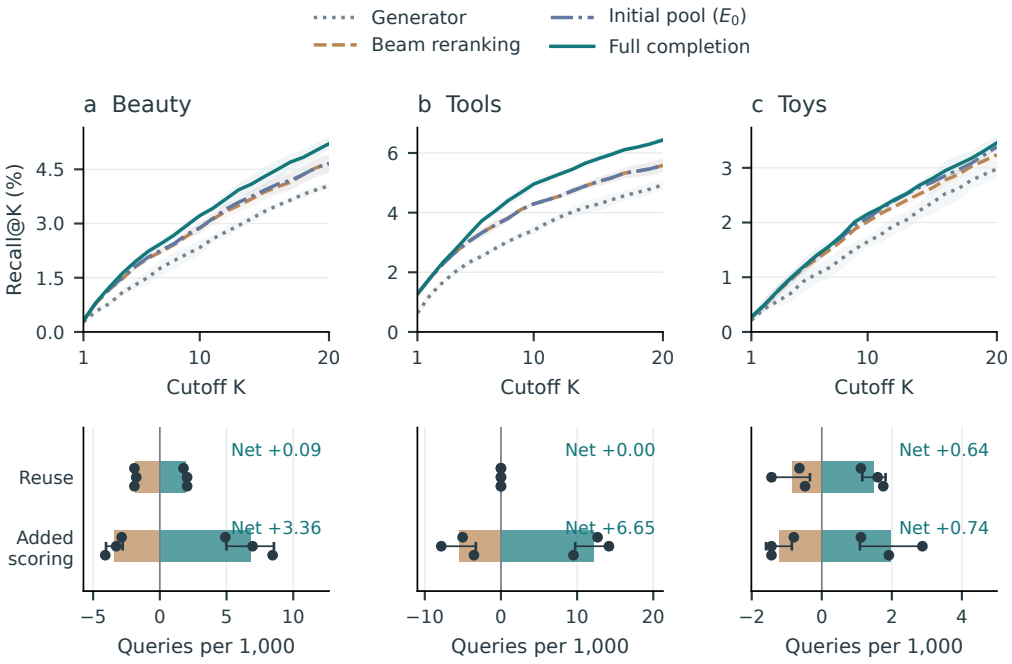


Figure 11: Candidate access under the final T5 configurations, seeds 17/42/2027. Top: Recall at all integer cutoffs 1–20; lines and bands show means and sample standard deviations. Bottom: Top-10 turnover for returned beam → E_0 (reuse) and E_0 → full completion (added scoring). Teal bars count recovered hits and ochre bars displaced hits per 1,000 queries. Bars and error bars give means and sample standard deviations; dots show each seed and annotations give the mean net change.

actually added by completion and change from a Top-10 miss under beam reranking to a hit. We choose the final target rank closest to this subset's median (rank 7), breaking ties by test-query order; the chosen target finishes at rank 8. The case illustrates a successful recovery; the complete-population gains and losses in Figure 11 characterize aggregate effects.

The initial search fully evaluates 82 items but returns only 40 beam candidates. Completion adds 20 full-likelihood evaluations, yielding 102 scored items. The target is fifth in priority order within that added batch and finishes at rank 8; 5 added items enter the final Top-10.

The initial-beam control ranks only the returned 40 items, whereas the full method also reuses scored terminal expansions outside the beam. Both panels display the same final score coordinates. In particular, scores obtained during completion are plotted retrospectively in the left panel; they

were not available before those evaluations. No model update or identifier change occurs between the two panels.

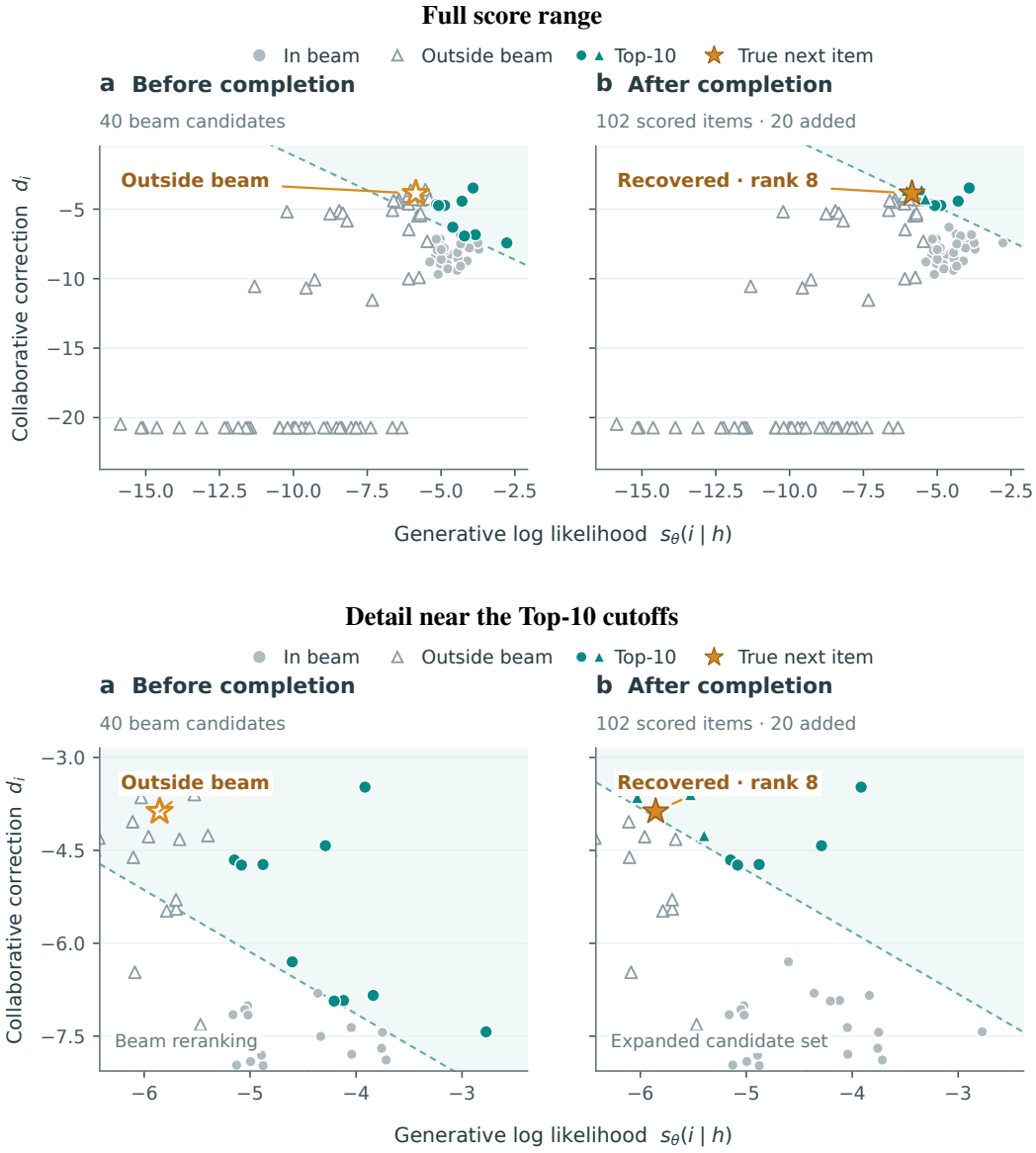


Figure 12: Candidate completion for a Beauty seed-17 query under the fixed score $F_i = s_\theta(i \mid h) + d_i$. Each row compares (a) beam reranking and (b) ranking after completion. The upper row shows all 102 scored items; the lower row enlarges the 41 items near the decision boundaries. Circles/triangles denote items inside/outside the returned beam; teal marks returned Top-10 items and the gold star marks the target. Dashed lines mark stage-specific Top-10 cutoffs. Initial search scores 82 items and returns 40; completion scores 20 more and places the target at rank 8. Coordinates stay fixed within each row; later-computed scores are shown retrospectively in (a). The case-selection rule is stated above.

E.5 Where the Tools ranking difference arises

The Tools test set contains 4,873 old-target and 487 new-target queries. BB achieves Recall@10 of 4.956%, compared with Reformer's 4.210%, and NDCG@10 of 2.806%, compared with 2.436%. Old targets contribute $+0.24652$ percentage points to BB minus Reformer NDCG@10 and new targets contribute $+0.12336$ points. Their sum is the aggregate difference; the subgroup orderings are distinct (Table 17).

Calibration and the generator. Generator-only new-target NDCG@10 is 4.367%. Applying the learned correction to every item yields 3.888%; the calibrated method reaches 9.234%, while old-target NDCG changes from 2.258% to 2.164%. New-target Recall@10 changes from 6.982% to 13.689%. This isolates a scoring effect without changing the generator or its identifiers. Reformer's new-target NDCG@10 is 7.876%; the complete-system difference also includes representation and adaptation choices.

Training support and concentration. New denotes addition since the old catalog, not absence of training interactions. Of the 487 new-target test queries, 308 have target items observed in update training and 179 have items absent from the available training interactions. Neither the calibrated BB nor Reformer hits any of the latter targets in Top-10. Reformer's new-target gain is concentrated: one item, appearing as a training target 12 times and as a test target 40 times, contributes 75.7% of its new-target NDCG. The same item contributes 76.8% of BB's new-target NDCG. BB hits eight, seven and eight distinct new items across seeds 17/42/2027; each Reformer seed hits four. These are query-weighted recommendation results rather than uniform averages over item identities.

E.6 Update strategies and conventional recommenders

Tables 16 and 17 extend the standard update and Reformer comparisons to cutoff 20 and target cohorts. The conventional references use the same test histories and catalog; their training and scoring protocols are in Appendix C.7. With validation-tuned settings, BB has the highest column means in the conventional comparison at cutoff 10.

Table 16: Cutoff-20 results for standard update strategies, Reformer and BB (%). Mean ± sample s.d. over seeds 17/42/2027 for all methods.

	Beauty		Tools		Toys	
Method	R@20	N@20	R@20	N@20	R@20	N@20
Frozen/Frozen	2.259 ± 0.052	0.937 ± 0.043	3.601 ± 0.263	1.674 ± 0.078	1.793 ± 0.189	0.720 ± 0.071
FT/Frozen	0.673 ± 0.052	0.239 ± 0.014	0.609 ± 0.103	0.233 ± 0.044	1.005 ± 0.138	0.454 ± 0.049
Frozen/FT	3.149 ± 0.121	1.247 ± 0.035	5.896 ± 0.233	2.728 ± 0.130	2.474 ± 0.169	0.995 ± 0.077
FT/FT	3.536 ± 0.057	1.363 ± 0.034	5.721 ± 0.106	2.577 ± 0.077	2.522 ± 0.152	0.981 ± 0.027
FT/RT	3.913 ± 0.125	1.489 ± 0.048	4.049 ± 0.153	1.796 ± 0.100	2.676 ± 0.088	1.016 ± 0.046
Reformer (2025)	3.499 ± 0.311	1.376 ± 0.115	5.808 ± 0.109	2.837 ± 0.142	2.553 ± 0.257	1.044 ± 0.123
BB (Ours)	$\mathbf{5.208 \pm 0.189}$	$\mathbf{2.035 \pm 0.079}$	$\mathbf{6.437 \pm 0.112}$	$\mathbf{3.179 \pm 0.009}$	$\mathbf{3.458 \pm 0.109}$	$\mathbf{1.352 \pm 0.030}$

E.7 Old/new utility across recommendation cutoffs

Figure 13 compares old- and new-target Recall at $K \in \{5, 10, 15, 20\}$ while keeping each model, map and scoring configuration fixed. The three datasets retain the same query populations at every cutoff. Their old/new/all query counts are 5,334/1,486/7,335 (Beauty), 4,873/487/5,360 (Tools) and 4,428/1,099/6,266 (Toys). Future-catalog targets contribute zero to the all-query metric, which satisfies

$$R_{\text{all}}(K) = \frac{n_{\text{old}} R_{\text{old}}(K) + n_{\text{new}} R_{\text{new}}(K)}{n_{\text{all}}}.$$

On Tools at cutoff 10, BB raises old-target Recall from Reformer's 3.441% to 4.084%, and new-target Recall from 11.910% to 13.689%. BB also exceeds Reformer on both cohorts at cutoff 20. Newly admitted items can have update-training interactions; these cohorts distinguish catalog age rather than interaction-free cold start. The paired component effects are reported in Table 2.

Table 17: Old- and new-target results for the expanded update comparison (%). Mean ± sample s.d. over seeds 17/42/2027 for all methods.

Dataset	Method	Old R@10	Old N@10	New R@10	New N@10
Beauty	Frozen/Frozen	1.987±0.032	1.009±0.052	0.000±0.000	0.000±0.000
	FT/Frozen	0.462±0.011	0.211±0.012	0.045±0.078	0.014±0.024
	Frozen/FT	1.937±0.066	1.000±0.008	2.041±0.236	0.924±0.135
	FT/FT	2.025±0.037	1.005±0.043	3.028±0.308	1.315±0.178
	FT/RT	2.100±0.172	1.032±0.053	3.320±0.383	1.520±0.166
	Reformer	1.994±0.103	1.020±0.060	3.028±0.583	1.347±0.247
	BB (Ours)	2.618±0.076	1.261±0.077	6.483±0.371	3.052±0.232
Tools	Frozen/Frozen	2.722±0.117	1.529±0.061	0.000±0.000	0.000±0.000
	FT/Frozen	0.349±0.036	0.175±0.017	0.205±0.356	0.067±0.115
	Frozen/FT	3.879±0.160	2.067±0.069	8.419±0.895	4.902±0.961
	FT/FT	3.147±0.140	1.642±0.086	13.415±0.314	7.375±0.584
	FT/RT	3.003±0.357	1.614±0.139	0.000±0.000	0.000±0.000
	Reformer	3.441±0.031	1.893±0.042	11.910±1.550	7.876±1.668
	BB (Ours)	4.084±0.142	2.164±0.088	13.689±0.314	9.234±0.314
Toys	Frozen/Frozen	1.566±0.249	0.773±0.097	0.000±0.000	0.000±0.000
	FT/Frozen	0.964±0.116	0.527±0.050	0.000±0.000	0.000±0.000
	Frozen/FT	1.814±0.079	0.926±0.062	1.062±0.379	0.502±0.193
	FT/FT	1.716±0.158	0.857±0.090	1.213±0.139	0.560±0.067
	FT/RT	1.724±0.192	0.840±0.109	1.426±0.344	0.686±0.123
	Reformer	1.716±0.197	0.858±0.084	2.335±0.430	1.154±0.282
	BB (Ours)	1.972±0.138	0.931±0.044	4.337±0.556	2.103±0.285

Table 18: Additional comparison with conventional recommenders on complete test sets: Recall and NDCG at 10 (%). Neural methods, including BB, use seeds 17/42/2027 and report mean ± sample s.d. ($n = 3$); ItemKNN and EASE are deterministic fits. All methods use validation-tuned settings, the same test queries and the same catalog. Training protocols are given in Appendix C.7. Bold marks the highest column means.

	Beauty		Tools		Toys	
Method	R@10	N@10	R@10	N@10	R@10	N@10
Frozen model	1.636±0.098	0.775±0.084	2.077±0.075	1.074±0.043	1.059±0.066	0.532±0.003
Static assignment	2.122±0.127	1.006±0.105	3.464±0.127	1.913±0.065	1.612±0.188	0.747±0.090
ItemKNN (2013)	1.391	0.656	1.325	0.738	1.404	0.719
EASE (2019)	1.963	1.010	3.265	1.936	1.835	0.968
SASRec (2018)	2.677±0.088	1.176±0.030	2.935±0.188	1.519±0.032	1.654±0.124	0.756±0.030
JTM matching (adapted)	2.059±0.165	0.984±0.116	3.302±0.202	1.799±0.190	1.623±0.060	0.735±0.024
DREAM voting (adapted)	2.154±0.165	1.022±0.122	3.470±0.135	1.915±0.067	1.325±0.032	0.614±0.012
DACT (2026)	2.095±0.200	1.063±0.068	4.142±0.391	2.260±0.214	1.505±0.066	0.762±0.047
BB (Ours)	**3.217±0.072**	**1.535±0.047**	**4.956±0.138**	**2.806±0.053**	**2.154±0.064**	**1.027±0.063**

Table 19: Validation-tuned conventional baselines at cutoff 20 (%). SASRec reports mean ± sample s.d. over seeds 17/42/2027; the other methods are deterministic.

	Beauty		Tools		Toys	
Method	R@20	N@20	R@20	N@20	R@20	N@20
ItemKNN	2.290	0.883	2.220	0.964	2.474	0.991
EASE	3.367	1.363	4.795	2.321	2.777	1.207
SASRec	4.394±0.028	1.607±0.023	4.683±0.275	1.963±0.073	2.718±0.037	1.024±0.011

Table 20: Old- and new-target NDCG@10 for validation-tuned conventional baselines (%). SASRec reports mean ± sample s.d. over seeds 17/42/2027; the other methods are deterministic.

	Beauty		Tools		Toys	
Method	Old	New	Old	New	Old	New
ItemKNN	0.758	0.516	0.792	0.198	0.758	1.043
EASE	1.315	0.262	2.103	0.267	1.222	0.596
SASRec	0.861±0.032	2.713±0.187	1.588±0.034	0.827±0.394	0.761±0.028	1.247±0.246

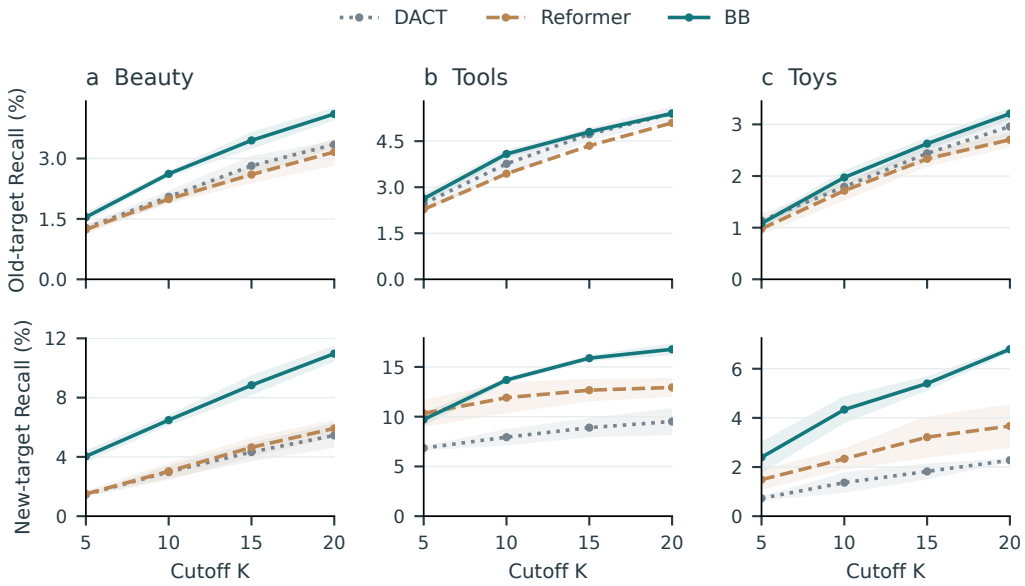


Figure 13: T5 cohort utility across cutoffs on Beauty, Tools and Toys. Top: old-target Recall. Bottom: new-target Recall. All methods use seeds 17/42/2027; curves and bands show means and sample standard deviations. Markers give the four evaluated cutoffs, joined by straight segments. Models, maps, scores and cohort membership are held fixed as the cutoff changes.

F LC-Rec Backbone: Comparison, Components and Cost

This extension examines whether the scoring and completion procedure also works with a decoder-only recommendation backbone. The full comparison, component profiles and computational costs are collected here. The study uses seed 17 and measures scoring and completion on a second backbone; the selected maps remain unchanged in all three domains.

F.1 Complete recommendation comparison

Table 21 compares seven methods under the common catalog update. BB leads the six Recall/NDCG@10 cells, with gains of 15.7–32.9% and 10.5–35.6% over the strongest baseline per dataset and metric. All values use seed 17. The component comparisons below attribute these gains to scoring and completion; the selected construction maps are unchanged in all three domains.

F.2 Shared model and update protocol

The decoder-only comparison uses Qwen2.5-1.5B-Instruct with semantic item identifiers, following the LC-Rec backbone configuration of Feng et al. (2026). It uses the same Beauty, Tools and Toys examples, maximum history length and available catalogs as Appendix C. All methods start from a shared initial model for each domain and evaluate the catalog update specified there. The three available catalogs contain 11,778, 10,043 and 11,639 items. Every reported run uses seed 17.

Table 21: Recommendation with the LC-Rec backbone (%). Single seed 17; the same test queries and available catalogs as the T5 comparison. Bold marks the highest value in each column.

	Beauty		Tools		Toys	
Method	R@10	N@10	R@10	N@10	R@10	N@10
Frozen	1.159	0.631	1.474	0.812	0.958	0.414
Fine-tuning	1.704	0.872	3.284	1.737	1.133	0.537
LSAT	1.377	0.723	1.847	1.085	1.037	0.442
PESO	1.745	0.916	3.396	1.772	1.357	0.651
DACT	2.154	1.022	3.563	1.925	1.468	0.741
Reformer	1.909	0.926	3.526	2.049	1.053	0.501
BB (Ours)	**2.863**	**1.386**	**4.123**	**2.264**	**1.899**	**0.858**

Initial models use rank-8 LoRA with scale 32 and dropout 0.05, applied to query, value, attention-output and feed-forward up/down projections; input embeddings and the output head are also trainable. Training uses an 8-bit base model with bfloat16 computation and full-precision trainable parameters. AdamW uses learning rate 2×10^{-5}, effective batch size 128, zero weight decay, 200 warmup steps and a cosine schedule. The maximum is 50 epochs. Full validation response cross-entropy is checked approximately five times per epoch after the first epoch; ten checks without improvement stop training, and the earliest minimum selects the model. These settings follow the public DACT recommender implementation; the same optimization recipe is used to fit the initial models. The initial fits stop after 18.40, 19.00 and 21.81 epochs on Beauty, Tools and Toys, respectively.

Frozen retains the initial model and identifiers. Fine-tuning adapts the recommender with the shared update mapping. LSAT (Shi et al., 2024) interpolates the historical adapter and the adapted adapter, with weight 0.5. PESO (Yoo et al., 2026) uses the released latest-adapter proximal objective with coefficient 2. DACT uses its updated tokenizer mapping, including changes to 1,363/1,829/1,110 old-item identifiers. Reformer adds 16 entries per semantic codebook while retaining existing identifiers. It uses rank-64 LoRA with scale 128 and an initialization preserving the shared parent's adaptation; DACT retains rank 8 and scale 32. Both reuse the tokenizer preparation described in Appendices C.5 and C.6. Recommender settings are tuned on validation from the public implementations; tokenizer preparation follows the protocols above.

BB first adapts the shared parent for five fixed epochs, then evaluates construction candidates on training examples and continues adaptation. The paired without-construction model uses the same warmup weights and update recipe. The selected maps remain unchanged in all three domains, so the paired models and predictions coincide. Continued adaptation selects its best validation loss at epochs 5.54/3.00/5.95 and stops at 7.52/4.79/7.93, measured after the five-epoch warm-up.

F.3 Scoring selection and evaluation

BB reuses the training-only linear predictor and the correction in (7). Parameters are selected separately for this backbone. The joint grid uses $\lambda \in \{2, 4, 6, 8\}$ on Beauty/Toys and $\lambda \in \{0.25, 0.5, 0.75, 1\}$ on Tools, with $\gamma \in \{0.5, 1, 1.5\}$ and $b \in \{0, 0.5, 1, 1.5, 2\}$. Each domain evaluates 60 settings with an 80-candidate completion budget, maximizing validation NDCG@10 followed by Recall@10. Exact ties retain the previous configuration when available, then minimize parameter distance from it, followed by smaller parameters. Holding the selected score fixed, validation compares budgets 80, 160 and 320, preferring the smaller budget on ties. The selected $(\lambda, \gamma, b, \text{budget})$ are $(4, 1.5, 0, 80)$, $(0.5, 1.5, 2, 160)$ and $(6, 1.5, 0, 80)$ for Beauty, Tools and Toys.

Scoring selection uses 1,040 Beauty and 774 Toys cross-day validation queries and all 3,007 Tools validation queries; model selection by response loss uses the full validation splits. All three scoring configurations are fixed before their test evaluation. Inference uses the original full-precision base weights and learned adapters, a width-40 beam and full-vocabulary log probabilities summed over four identifier tokens, excluding EOS. Each method re-encodes the same histories with its own mapping; repeated items remain eligible. The combined score is used consistently for ranking, candidate bounds and stopping. As in (10), E_0 contains all terminal leaves fully scored by that query's initial

search. Completion selects only items outside the evaluated set, and each selected item counts once toward the additional budget. Offline likelihood reuse across parameter configurations leaves this per-query accounting unchanged. The selected budgets are 80/160/80 on Beauty/Tools/Toys.

F.4 Component effects and additional metrics

Table 22 reports every paired control. Collaborative scoring, new-item calibration and candidate completion each improve NDCG@10 over their corresponding removal in all three domains. These component gains come from scoring and completion, as the without-construction control uses the same maps and models. Section 5.2 separately evaluates constructive feasibility. Collaborative priority slightly exceeds bound priority on Tools (2.270% versus 2.264% NDCG@10) and ties it on the other two domains. Both use the same final score and candidate budget.

Table 22: LC-Rec component controls: Recall and NDCG at 10 (%), seed 17. Each removal retains the selected parameters and its domain's completion budget; the without-construction control uses the paired map and model.

	Beauty		Tools		Toys	
Variant	R@10	N@10	R@10	N@10	R@10	N@10
BB (Ours)	2.863	1.386	4.123	2.264	1.899	0.858
Without construction	2.863	1.386	4.123	2.264	1.899	0.858
Without calibration	2.468	1.201	3.731	1.936	1.835	0.793
Without collaborative score	1.922	1.010	2.761	1.465	1.293	0.618
Without completion	2.399	1.184	3.489	1.910	1.580	0.774
Item correction only	2.590	1.267	3.563	1.949	1.787	0.804
Collaborative priority	2.863	1.386	4.142	2.270	1.899	0.858
Generator only	1.922	1.010	2.761	1.465	1.293	0.618

The generator-only Toys NDCG@10 is 0.618%; a diagnostic ranking by current update-training target frequency obtains 0.620% on the same test queries. The complete procedure reaches 0.858%, above the generator-only score and frequency diagnostic on these Toys queries. Tables 23 and 24 give the full cutoff-20 comparison and BB target-cohort results for seed 17.

Table 23: LC-Rec Recall and NDCG at 20 (%), seed 17.

	Beauty		Tools		Toys	
Method	R@20	N@20	R@20	N@20	R@20	N@20
Frozen	1.977	0.838	2.295	1.015	1.548	0.563
Fine-tuning	3.095	1.217	4.776	2.115	2.139	0.787
LSAT	2.209	0.929	3.396	1.475	1.756	0.620
PESO	3.067	1.248	4.925	2.154	2.426	0.916
DACT	3.517	1.365	4.981	2.283	2.218	0.930
Reformer	3.599	1.355	4.944	2.409	2.043	0.749
BB (Ours)	4.513	1.802	5.690	2.662	2.968	1.128

F.5 Inference cost

Table 25 separates full-test candidate counts from fresh inference latency on validation queries. The stopping fraction is the fraction satisfying the score-bound stopping test at the prescribed numerical tolerance of 10^{-3}. Queries exhausting their candidate budget without that condition remain empirical rankings. Latency uses fresh inference on validation queries; the grid reuses likelihoods across configurations.

F.6 Old/new utility across cutoffs

Figure 14 separates old- and new-target Recall using the cohort definitions in Appendix E.7. On Tools, BB improves old-target and all-target utility while new-target Recall at cutoff 10 remains

Table 24: BB with the LC-Rec backbone by target cohort (%), seed 17. Future targets lie outside the available catalog and remain misses in the complete-population metrics.

Dataset	Target cohort	Queries	R@10	N@10
Beauty	Old	5334	2.381	1.136
Beauty	New	1486	5.585	2.762
Beauty	Future	515	0.000	0.000
Tools	Old	4873	3.817	2.018
Tools	New	487	7.187	4.733
Tools	Future	0	–	–
Toys	Old	4428	1.649	0.669
Toys	New	1099	4.186	2.197
Toys	Future	739	0.000	0.000

Table 25: Inference costs of BB with the LC-Rec backbone. Additional candidates and stopping fractions use the full test sets. Latency uses fresh beam decoding and completion for the first 32 validation queries on one RTX 3090, excluding model loading and common input/predictor preparation.

Dataset	Mean additional candidates	Stopping (%)	Seconds/query
Beauty	45.5	96.4	0.260
Tools	108.7	76.1	0.638
Toys	44.9	98.7	0.249

below DACT; the new-target ordering changes at cutoff 20. On Toys, BB and DACT tie on old-target Recall at cutoff 10, while BB has higher new-target Recall. All comparisons use seed 17.



Figure 14: LC-Rec cohort utility on Beauty, Tools and Toys, seed 17. Top: old-target Recall. Bottom: new-target Recall. Models, mappings and scoring settings stay fixed while the cutoff changes. Markers give the four evaluated cutoffs, joined by straight segments; no seed variability interval is estimated. Absolute cutoff-10/20 and component comparisons are given in Tables 21–24.

G Extended Related Work

This section expands Section 6 around the connection between identifier assignment, candidate access and final ranking that motivates BB.

G.1 Identifier updates and constrained repair

Semantic identifiers make the item mapping part of the retrieval mechanism. VQ-Rec learns transferable item representations from discrete codes (Hou et al., 2023). LMIndexer learns semantic identifiers through document reconstruction and progressive training (Jin et al., 2024). TIGER learns residual-quantized identifiers and generates them autoregressively (Rajput et al., 2023). LC-Rec aligns language and collaborative semantics (Zheng et al., 2024), while LETTER incorporates semantic, collaborative and diversity objectives into tokenization (Wang et al., 2024). These approaches establish the importance of how items are represented and organized before decoding begins.

Catalog evolution adds the problem of updating this organization while retaining useful learned structure. Reformer studies incremental tokenization and recommendation (Shi et al., 2025). DACT identifies collaborative drift and differentiates tokenizer updates, followed by hierarchical code reassignment (Feng et al., 2026). SID-Staleness aligns refreshed codebooks with the previous token space to support warm-start retriever adaptation (Baikalov et al., 2026). GenRecEdit adapts the generator through token-level editing for cold-start items, with position-dependent triggering (Shen et al., 2026). These different interventions motivate separating the effects of identifier assignment from changes to the generator.

Score-based assignment provides a particularly close connection. JTM jointly learns a tree index and preference model, formulating item-to-leaf assignment through weighted matching (Zhu et al., 2019). DREAM constructs collaborative candidate identifiers, uses confidence-weighted multi-context evidence for commitment, and preserves alternative paths for inference (Guan et al., 2026). Such objectives provide evidence for choosing identifiers. In a finite beam, however, a target's score interacts with the occupied paths that support or compete with it.

BB makes this interaction explicit through support, terminal-rank and catalog-count constraints. After a common effective prefix, these constraints characterize the feasible catalog family for a fixed generator and encoded history. Minimum-replacement repair then retains the largest feasible overlap with current leaf occupancy for a specified state and target. This local construction supplies shared-map proposals, which are evaluated across re-encoded histories before generator adaptation (Sections 3 and 4).

The retained-catalog constraint also connects to representation compatibility. Backward- and forward-compatible embedding methods support retrieval across model versions (Shen et al., 2020; Ramanujan et al., 2022); here, compatibility fixes old item-to-identifier assignments and preserves their recommendation eligibility. Possible-worlds ranking studies answers over families of admissible data instances (Feng et al., 2023). Our family varies catalog occupancy, which changes prefix support and hence the beam-search trajectory itself.

G.2 Reachability and item-level scoring

Several studies identify why a valid identifier may still fail to retrieve its item. Temporal cold-item analysis relates retrieval to observed token and prefix support (Peng et al., 2026). SIDScope distinguishes mapping properties, generated-path survival and item resolution (Ding et al., 2026a). HCGRec addresses unreachable reward groups during post-training by supplying target-prefix hints to hard training examples (Zhang et al., 2026). Together, these studies motivate examining the intermediate decisions between identifier assignment and item-level ranking.

Training and architectural interventions address different parts of this process. Beam-aware tree learning studies calibration under the deployed search procedure (Zhuo et al., 2020). BEAR regularizes token ranks to reduce premature beam pruning (Yang et al., 2026), while Latte introduces a latent token to relax structural coupling between generation paths (Hou et al., 2026). BB studies how legal assignment changes alter the support and competition encountered by a fixed decoder, then uses the resulting repairs in shared-map construction and adaptation.

The inference stage addresses the remaining query-dependent candidate access problem. Linear collaborative models provide item-level evidence from interactions (Steck, 2019). BB fits a supervised history-to-next-item ridge predictor and combines its calibrated correction with identifier likelihood. The correction is available before full identifier evaluation, so it can guide both final ranking and the evaluation of omitted items. This connects the ranking objective to candidate completion: reranking alone cannot recover an item outside the evaluated pool.

G.3 Candidate access and search guarantees

Inference can expand access without changing the item mapping. SpecGR uses an inductive drafter to propose items, verifies them with generative likelihoods, and guides subsequent drafts using generated prefixes (Ding et al., 2026b). Its adaptive exit is triggered when enough candidates pass a verification threshold; accepted items are ranked by verifier scores. BB links candidate priority and stopping to the combined generator–collaborative score. The remaining question is whether an unevaluated catalog item can still displace the current Kth item.

This question connects to established search principles. Huang et al. (2017) derive optimal stopping modulo beam size for neural generation, including bounded length rewards. Meister et al. (2020) use score monotonicity for best-first beam search and early pruning. BB applies the prefix upper-bound principle to fixed catalog identifiers under the final ranking score. Adding the known item correction to an evaluated prefix score bounds the complete score because the remaining log-probability terms are nonpositive.

The bound covers every unevaluated catalog item, including identifiers whose paths left the initial beam. Additional evaluations are ordered by these bounds, and the strict stopping condition certifies the global Top-K under the combined score (Appendix B.6). An exhausted evaluation budget can instead return an uncertified ranking. Thus constructive repair organizes the shared catalog, while completion uses query-specific evidence to recover promising omitted candidates and determine when further evaluation cannot change the result.