ICR: SCALING IN-CONTEXT LEARNING FOR SEQUENTIAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential recommendation in new domains typically requires extensive per-dataset training data and compute. We present ICR, a zero-shot sequential recommender in the Prior-Fitted Network framework that adapts to new domains at inference time through in-context learning. Given a set of support sequences from the target, ICR produces next-item predictions in a single forward pass without requiring dataset-specific fine-tuning. Our design is motivated by three observations: (1) sequential recommendation dynamics are well-characterized by a hierarchy of aggregation functions, enabling a synthetic prior with provable full support over finite item sequences; (2) dependence on absolute embedding coordinates is not justified for a foundational ICL model—we prove this formally—motivating an ID-based architecture with a pointer mechanism that selects items directly from context; and (3) identity-restricted cross-attention scales sublinearly with context size, enabling performance to improve with more context rather than degrade. On the majority of 11 benchmark datasets, ICR outperforms every zero-shot baseline and, more remarkably, matches or exceeds standard supervised baselines—to the best of our knowledge the first zero-shot recommender to do so, allowing for lower end-to-end cost than supervised training for realistic user populations.

1 INTRODUCTION

The strongest sequential recommenders—based on self-attention and deep architectures—require substantial per-dataset training data and compute (Kang & McAuley, 2018; Sun et al., 2019; Zhou et al., 2020; Zhai et al., 2024). Foundation models have enabled zero-shot generalization in NLP and vision (Achiam et al., 2023; Liu et al., 2024a), but analogous success in recommendation remains elusive (Huang et al., 2025): user preferences are inherently domain-specific, making transferable representations hard to learn from aggregated interaction data.

Existing zero-shot approaches take one of two routes, each limited by this domain-specificity. LLM-prompting methods (Lyu et al., 2024; Wang & Lim, 2024; Liu et al., 2025) see only the textual side of items, surfacing semantic associations but missing the collaborative signal carried by user interactions—and they incur high latency at every query. Pre-training on diverse interaction datasets (DING et al., 2022; Jiang et al., 2025; Sheng et al., 2025) aims to learn behavioural patterns directly, but the patterns that emerge from one mixture of datasets transfer poorly to a new one.

We instead take an *in-context learning* (ICL) approach (Dong et al., 2024): a model learns a general-purpose sequential pattern-matching algorithm from diverse synthetic environments, and then adapts to any target domain by conditioning on a small set of support sequences at inference time without weight updates, following the Prior-Fitted Networks framework first introduced by Hollmann et al. (2022) for tabular tasks. Tan et al. (2026) and Kang et al. (2026) extend this to sequential recommendation. The former scales poorly with context size and matches supervised baselines only in low-compute or low-data regimes, while the latter is competitive only on smaller datasets.

To address these limitations, we introduce ICR (In-Context Recommender), a zero-shot sequential recommender with three contributions.

(1) A theoretically-grounded synthetic prior. We formalize sequential recommenders as projected kernel models and construct a generator spanning linear and content-dependent aggregation, with full support over finite item sequences (Theorem 3).

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.

(2) An ID-based architecture with pointer-based selection. We employ an ID-based architecture with pointer-based selection, eliminating the embedding output bottleneck. To avoid the dilution that degrades standard long-context attention (Liu et al., 2024b), we further introduce an identity-restricted cross-attention (IRCA) mechanism that constrains each query to a small, structurally-relevant subset of context positions.

(3) Strong zero-shot performance with monotonic context scaling. Across 11 benchmarks, ICR outperforms all zero-shot baselines on 10 datasets and supervised baselines on 8, while improving monotonically with context size—without per-dataset training.

2 Related Work

Sequential and foundation recommenders. Sequential recommendation has progressed from recurrent (Hidasi et al., 2015; Tan et al., 2016) and convolutional (Tang & Wang, 2018) architectures to self-attention models (Kang & McAuley, 2018; Sun et al., 2019; Zhou et al., 2020), all of which require per-dataset training data and substantial compute. More recent foundation-model approaches aim to remove this requirement (Huang et al., 2025): LLM-prompting methods (Lyu et al., 2024; Wang & Lim, 2024; Liu et al., 2025) face high latency and miss the collaborative signal carried by user interactions, while pre-training recommendation architectures on aggregated datasets (Wang et al., 2023; Jiang et al., 2025; Sheng et al., 2025; Hou et al., 2022; 2023) typically still requires fine-tuning and transfers poorly to unseen domains.

In-context learning for structured data. Prior-Fitted Networks approximate Bayesian posterior prediction by training on synthetic tasks sampled from a prior (Hollmann et al., 2022; 2025), with strong results in tabular (Qu et al., 2025; Ma et al., 2024; Arazi et al., 2025), graph (Choi et al., 2026), and time-series (Hoo et al., 2025) settings. For sequential recommendation, two concurrent works apply the PFN recipe. RecPFN (Tan et al., 2026) use embedding-space prediction and full cross-attention over the support context, but their performance degrades as context grows. SRPFN (Kang et al., 2026) uses a graph-structured prior and conditions on low-rank co-occurrence summaries from the target domain, but produces user-dependent item representations that must be recomputed per query, making it practical only on small catalogs. Furthermore, both approaches design their priors ad hoc, without formal coverage guarantees. ICR addresses the remaining limitations: a prior with provable full support (Theorem 3), identity-restricted attention that scales sublinearly with context size, and direct inference from raw interaction sequences—a design further motivated by Bayesian scaling laws (Arora et al., 2024), which tie ICL efficiency to the quality of the task prior.

Architectural building blocks. Our pointer mechanism builds on Pointer Networks (Vinyals et al., 2015) and copy mechanisms (Gu et al., 2016), which enable attention-based selection from input sequences; unlike generative-retrieval approaches (Rajput et al., 2023; Ju et al., 2026) that decode into a fixed pre-trained codebook, our pointer selects from a dynamically-constructed support context that changes with every inference call. We further address the dilution and positional bias that standard long-context attention exhibits (Liu et al., 2024b)—usually mitigated through locality-sensitive hashing (Han et al., 2024) or differential softmax maps (Ye et al., 2025)—through exact sparsity, restricting each query to same-item context positions and yielding cost proportional to average item frequency rather than total context length.

3 A Synthetic Prior for Sequential Recommendation

ICR adapts at inference time by approximating Bayesian posterior prediction across a broad distribution of synthetic environments. We first recall the Prior-Fitted Network framework, then develop a prior grounded in the structure of existing sequential recommenders.

3.1 Prior-Fitted Networks

Let the item catalog be $\mathcal{I} = \{1, \ldots, N\}$ and a user history $s_{1:t} = (s_1, \ldots, s_t)$ with $s_\tau \in \mathcal{I}$. We write $\pi_t(j) := P(s_{t+1} = j \mid s_{\leq t})$ for the next-item distribution. An environment $h \in \mathcal{H}$ specifies its evolution law $\pi_t(j \mid h)$, and a prior $p(h)$ encodes beliefs about plausible recommendation dynamics.

Prior-Fitted Networks (Hollmann et al., 2022; 2025) approximate the Bayesian posterior predictive distribution (PPD) $\pi_t(j \mid \mathcal{D}) = \int_{\mathcal{H}} \pi_t(j \mid h)\, p(h \mid \mathcal{D})\, dh$ in a single forward pass, where $\mathcal{D}$ is a set of observed sequences from the target domain. A network f_θ approximates this PPD by minimizing $\mathcal{L}(\theta) = \mathbb{E}_{h,\, \mathcal{D},\, (s_{1:t},\, y)}\big[-\log f_\theta(y \mid s_{1:t}, \mathcal{D})\big]$, where $h \sim p(h)$ and data are drawn from h. The design of the prior $p(h)$ is the central modeling choice here: it must be broad enough to cover plausible real-world dynamics, yet structured enough to admit useful inductive biases.

3.2 Prior Design

Rather than designing the prior from scratch, we take guidance from the computational structure of existing sequential recommenders (Kang & McAuley, 2018; Sun et al., 2019; Jannach & Ludewig, 2017). Despite their architectural diversity, we show that these models share a common scoring template, which we use to identify the sequential patterns the prior must generate, design a generator that spans them systematically, and prove coverage guarantees on what the PFN trains on.

3.2.1 Projected Kernel Models

Across nonparametric methods, factorized models, and neural architectures, sequential recommenders share a common scoring structure, which we formalize as the *projected kernel model*:

Definition 1 (Projected Kernel Model)**.** *A sequential recommender is a projected kernel model if it scores candidate items via:*

$$\pi_t(j) = \mathrm{softmax}_j\big(\phi(s_{\le t})^\top \psi(j) + b_j\big) \tag{1}$$

where $\phi : \mathcal{I}^{\le T_{\max}} \to \mathbb{R}^d$ encodes the interaction history, $\psi : \mathcal{I} \to \mathbb{R}^d$ maps each candidate item to the same space, and $b_j \in \mathbb{R}$ captures item-level bias.

For any such model, ϕ can be decomposed as a weighted sum of per-position contributions:

$$\phi(s_{\le t}) = \sum\nolimits_{\tau=1}^{t} w_\tau(s_{\le t}) \cdot g_\tau(s_{\le t}) \tag{2}$$

where $w_\tau(s_{\le t}) \in \mathbb{R}$ is a weight on position τ and $g_\tau(s_{\le t}) \in \mathbb{R}^d$ its contribution. We denote the full class of projected kernel models $\mathcal{F}_{\mathrm{full}}$.

Linear projected kernel models ($\mathcal{F}_{\mathrm{lin}}$). A natural subclass $\mathcal{F}_{\mathrm{lin}} \subseteq \mathcal{F}_{\mathrm{full}}$ arises when the aggregation weights depend only on position, not item content:

$$\phi(s_{\le t}) = \sum\nolimits_{\tau=1}^{t} w_{t-\tau} \cdot M \cdot \mathbf{e}_{s_\tau} \tag{3}$$

where $w_{t-\tau}$ depends only on the lag and M is a shared projection. This subclass includes nonparametric and factorized methods, such as item-based collaborative filtering and Markov-chain models (Jannach & Ludewig, 2017; Rendle et al., 2010); neural sequence models such as SASRec (Kang & McAuley, 2018) and GRU4Rec (Hidasi et al., 2015) fall outside it, with weights or contributions that depend on sequence content (explicit decompositions in Appendix J.1). We first show that the SCM scoring family subsumes $\mathcal{F}_{\mathrm{lin}}$, then extend coverage to the rest of $\mathcal{F}_{\mathrm{full}}$.

3.2.2 Covering $\mathcal{F}_{\mathrm{lin}}$

Each environment in our prior is a structural causal model (SCM; Pearl, 2009) specifying item properties and a sequence-generation process. Building on the random-graph prior of Tan et al. (2026), the SCM scores the next item as a lag-weighted sum of item-to-item transition scores:

$$\pi_t(j) = \sigma_j\Big(\sum\nolimits_{\tau=t-W+1}^{t} w_{t-\tau} \cdot T_{s_\tau, j} \;+\; b_j\Big) \tag{4}$$

where W is the window size, $T \in \mathbb{R}^{N \times N}$ a transition matrix, $w_{t-\tau} \in \mathbb{R}$ a per-lag weight, b_j a popularity bias, and σ a monotone normalization. This is exactly Eq. 3 with $g_\tau = T_{s_\tau, :}$ and $\psi(j) = \mathbf{e}_j$, establishing the family's representational expressiveness:

Proposition 2 (Expressiveness)**.** *Some (T, w, W, b) makes the generator's scoring function—and hence its item ranking—identical to any given $f \in \mathcal{F}_{\mathrm{lin}}$. (Proof in Appendix J.2.)*

Proposition 2 establishes that the unconstrained SCM scoring family subsumes $\mathcal{F}_{\mathrm{lin}}$. We instantiate a structured prior within this family using sparse random transitions and geometric decay, concentrating mass on plausible recommendation dynamics (Appendix G.1).

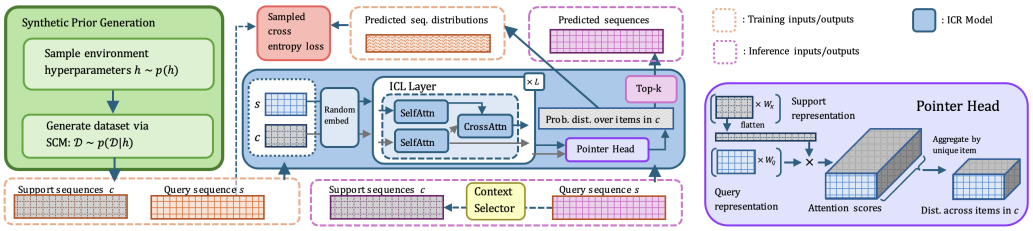


Figure 1: Overall framework of ICR.

3.2.3 Extending Beyond $\mathcal{F}_{\text{lin}}$

Comparing Eqs. 2 and 3, $\mathcal{F}_{\text{full}}$ differs from $\mathcal{F}_{\text{lin}}$ along two axes: (1) *content-dependent weighting*—$w_\tau(s_{\leq t})$ depends on item identity rather than position alone (e.g., attention weights in SASRec); and (2) *content-dependent contributions*—$g_\tau(s_{\leq t})$ depends on neighbouring items (e.g., multi-layer value blending). We close each gap with a targeted extension to the SCM: a routing matrix $R \in \mathbb{R}^{N\times N}$ for (1) and an order-sensitive pattern hash for (2) (full specification in Appendix G.2):

$$\pi_t(j) = \sigma_j\Big(\sum_{\tau=1}^{t} \gamma_{s_t}^{t-\tau} \cdot \exp(R_{s_\tau,s_t}) \cdot T_{s_\tau,j} \;+\; \alpha \cdot a_{h(s_{t-k+1:t}),\,j} \;+\; b_j\Big) \tag{5}$$

The routing term $\exp(R_{s_\tau,s_t})$ modulates how strongly the current item attends to each past item, analogous to content-content attention. The pattern term hashes the recent k-gram to a bucket and reads additive score contributions from a sparse affinity matrix a, letting transitions depend on specific sequential patterns rather than individual items—the effect of multi-layer value blending in neural models. Setting $R = 0$, $w_{t-\tau} = \gamma^{\,t-\tau}$, and $\alpha = 0$ recovers the linear prior (Eq. 4). Theorem 3 gives an insurance guarantee for the general sequence prior: no finite item sequence is excluded. Its structured design—controlled sparsity, popularity, recency, and pattern modulation—then concentrates probability mass on dynamics intended to resemble real recommendation environments:

Theorem 3 (Full Support of the General SCM Family)**.** *Let $P^{(L)}_{\text{prior}}$ be the distribution induced directly by the general SCM in Eq.* (5) *for a fixed finite length L, with $\sigma_{\text{noise}} > 0$. For every finite catalog $\mathcal{I}$ and every sequence $s_{1:L} \in \mathcal{I}^L$, $P^{(L)}_{\text{prior}}(s_{1:L}) > 0$. (Proof in Appendix J.3.)*

4 Model

ICR is a decoder-only transformer that produces next-item recommendations (Figure 1) by conditioning on a *support context* from the target domain: a set of example user sequences drawn from the same dataset. The backbone processes query and support sequences through L layers of interleaved causal self- and cross-attention, and then selects a next-item directly from the support context with a pointer head, bypassing fixed-vocabulary scoring and embedding-space retrieval.

4.1 Input Formulation

On each model call, every item i is assigned a fresh, non-trainable random vector $\mathbf{e}_i \sim \mathcal{N}(\mathbf{0}, d^{-1}\mathbf{I})$, shared across query and support sequences within the call. These vectors carry no semantic content and serve only as identity tokens.

This design is formally motivated. Suppose no direction in embedding space is systematically privileged across environments—that is, the environment distribution is invariant to rotations of the embedding coordinate system. Rotations preserve all inner products, so this assumption retains semantic similarity while excluding only dependence on absolute coordinates (Appendix K). It is also consistent with our empirical observation that no existing foundational recommender reliably outperforms EmbKNN-R, a non-parametric baseline using random embeddings and recency-weighted similarity (Table 3). The following result shows that access to in-domain context does not make absolute coordinates generally useful:

Theorem 4 (ICL Does Not Privilege Absolute Coordinates)**.** *Let f be any foundational ICL recommender that uses a single fixed set of weights across all environments, and assume that the envi-*

ronment distribution is invariant to the embedding coordinate system. Then there exists a rotation-invariant model $\bar{f}$ with $\mathcal{L}(\bar{f};\phi) \le \mathcal{L}(f;\phi)$. (Formal assumptions and proof in Appendix K.)

The intuition is symmetry: with no systematically privileged direction, rotational averaging cannot increase expected loss, even when the model conditions on context. The result preserves all Gram-matrix geometry, including semantic similarity; it rules out only absolute-coordinate dependence. We make the stronger practical choice of fresh random assignments as an architectural regularizer, forcing the model to infer collaborative structure from context rather than persistent embedding geometry. Semantic similarity remains available externally through embedding-space expansion (Section 6.4).

4.2 Architecture

The backbone consists of L layers, each alternating a self-attention block with a cross-attention block. Both use standard multi-head attention $\mathrm{MHA}(X, Z, M)$ with query input X, key/value input Z, and additive mask M, followed by a feed-forward network with residual connections (full definitions in Appendix F). Let $\mathbf{h}^{(\ell)}_{q,i}, \mathbf{h}^{(\ell)}_{c,i} \in \mathbb{R}^{T\times d}$ denote the representations of the i-th query sequence and the i-th support sequence (of length T) at layer ℓ, where the subscript c marks support (context) positions. Self-attention operates independently per sequence; cross-attention attends from each query sequence to the concatenation of all support sequences $\bar{H}^{(\ell)}_c = [\mathbf{h}^{(\ell)}_{c,1}; \ldots; \mathbf{h}^{(\ell)}_{c,N_c}] \in \mathbb{R}^{C\times d}$ with $C = N_c T$:

$$\underbrace{\tilde{\mathbf{h}}^{(\ell)}_{q/c,i} = \mathrm{MHA}(\mathbf{h}^{(\ell-1)}_{q/c,i}, \mathbf{h}^{(\ell-1)}_{q/c,i}, M_{q/c})}_{\text{self-attention}}, \qquad \underbrace{\mathbf{h}^{(\ell)}_{q,i} = \mathrm{MHA}(\tilde{\mathbf{h}}^{(\ell)}_{q,i}, \bar{H}^{(\ell)}_c, M_{\text{cross}})}_{\text{cross-attention}} \tag{6}$$

For self-attention, M_q combines causal masking with hard-ALiBi (Press et al., 2022) (a non-learned positional bias), while M_c is unrestricted (no causal constraint on context). The cross-attention mask $M_{\text{cross}} \in \mathbb{R}^{T\times C}$ implements identity-restricted routing, described in Section 4.3.

4.3 Identity-Restricted Cross-Attention

Standard cross-attention over all C context positions scales linearly with context size and suffers from attention dilution as C grows (Liu et al., 2024b), posing the primary context scaling bottleneck. Since non-causal self-attention already gives each occurrence of an item a distinct context-dependent representation, we observe that restricting cross-attention to same-item positions is sufficient: the query compares all contextualised occurrences of a given item and attends to the most relevant, removing both linear dependence on C and dilution from irrelevant positions. Formally, $M_{\text{cross}}[t,\tau] = 0$ if $s_t = c_\tau$ and $-\infty$ otherwise, where s_t is the item at query position t and c_τ is the item at context position τ. Each query position attends only to context positions sharing its item identity; positions whose item does not appear in context receive no update (identity residual).

Implementation. We implement this via gather-attend-scatter: group positions by item identity, compute attention within each group, and scatter results back (implementation details in Appendix I). The expected cost per query position is $O(C/U)$ where U is the number of unique items in context, compared to $O(C)$ for standard cross-attention (derivation in Appendix I).

4.4 Pointer Head

Rather than scoring all possible items via an embedding projection, ICR predicts directly over items observed in context via a pointer mechanism (Vinyals et al., 2015). The pointer head computes multi-head attention from query to all C context positions, then aggregates weights per unique item:

$$P(j \mid s_{1:t}, \mathcal{D}) \propto \sum_{\tau:\, c_\tau = j} \alpha_{t,\tau}, \qquad \alpha_{t,\tau} = \mathrm{softmax}_\tau\left(q_t^\top k_\tau / \sqrt{d}\right) \tag{7}$$

where $q_t = W_Q h_t$ and $k_\tau = W_K \bar{h}_\tau$ are learned projections; there is no value projection, since the attention weights themselves are the prediction. Unlike the backbone's restricted cross-attention, the pointer softmax normalises over all C context positions. Scatter-summing weights over positions containing each item yields per-item probabilities, computed independently across H_p heads and

Table 1: Per-baseline summary against ICR across 11 datasets. "Wins" = number of datasets on which the baseline strictly beats ICR on the given metric (out of 11). "Mean Δ" = mean percent difference relative to ICR (negative = baseline worse). ^ = LLM-embedding variant.

Zero-shot baselines (11 datasets)																
	RecPFN		SRPFN		RecFormer		RecGPT		UniSRec		VQ-Rec		EmbKNN		EmbKNN-R	
@3	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR
Wins / 11	0	1	0	0	0	0	0	1	0	0	0	1	0	0	1	1
Mean Δ (%)	-42	-42	-43	-52	-62	-62	-60	-52	-75	-76	-56	-50	-65	-70	-41	-41
Supervised baselines trained for 50 epochs (11 datasets)																
	BERT		BERT^		FDSA		GRU		GRU^		SAS		SAS^		UniSRec		VQ-Rec	
@3	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR
Wins / 11	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1
Mean Δ (%)	-26	-27	-83	-85	-67	-71	-70	-74	-68	-68	-54	-59	-68	-69	-32	-36	-22	-23
ICR	strictly beats all 8 zero-shot baselines on 20/22 cells; all 9 supervised baselines on 17/22 cells															

summed to form the final distribution. At inference, we replace the dense $Q \cdot K^\top$ with exact top-k' inner-product search per query per head, computing softmax only over the k' retrieved positions before scatter-aggregation; this reduces cost from $O(C)$ to $O(k')$ per query position.

4.5 Training and Inference

Each training step samples a fresh SCM by drawing $(T, R, \gamma, a, b, \alpha)$ from randomized ranges (Appendix F.3); we adopt a two-stage curriculum (Bengio et al., 2009) where Stage 1 trains on the linear SCM only (Eq. 4) and Stage 2 activates the full extended SCM (Eq. 5). Generated sequences are partitioned into a support context and queries, and the pointer head is trained with a cross-entropy loss on the next item. At inference, support sequences are retrieved from the target domain's training set by recency-weighted item overlap with the query (Appendix H); the pointer head then produces a distribution over context items, from which top-k recommendations are drawn.

5 Experiments

5.1 Setup and Implementation

Setup. We evaluate on 11 public datasets spanning diverse platforms and domains, with catalog sizes ranging from ∼22K to ∼340K items (Appendix E.1). On each, we compare ICR against a comprehensive set of supervised and zero-shot baselines: ID-based models, embedding-based models, pre-trained recommenders with dataset-specific fine-tuning, and non-parametric embedding baselines (full list and per-method configurations in Appendix E.2). We report HR@3 and MRR@3 under leave-one-out evaluation with full-catalog ranking; all methods use identical 70/10/20 train/validation/test splits, and supervised baselines are averaged over 4 random seeds.

Implementation. ICR is trained on a single A10 GPU for 16 hours. Best-model selection normalises validation HR@10 against a non-parametric EmbKNN baseline to control for environment difficulty. Reported metrics are averaged over 4 training runs. Full details in Appendix E and F.

5.2 Main Results

We summarise the per-dataset benchmark in Table 1; full per-dataset HR@3 and MRR@3 numbers for every baseline can be found in Appendix A. For each baseline we report, across the 11 evaluation datasets, the number of (dataset, metric) cells on which the baseline beats ICR, and the mean percent difference in HR@3 and MRR@3 relative to ICR.

Zero-shot comparison. ICR attains the best HR@3 and best MRR@3 on 10 of 11 datasets, the sole exception being Yelp—the sparsest, most weakly-sequential benchmark—where EmbKNN-R edges ahead. The closest competitor overall is SRPFN, which ranks second on five datasets but still trails

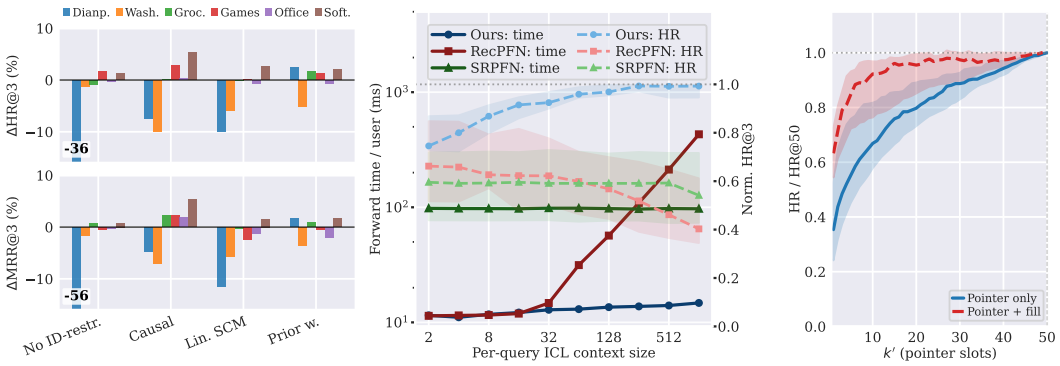


Figure 2: Analysis figures. **Left:** Ablation deltas; prior-width family aggregated as max $|\Delta|$ over $0.5\times/2\times/3\times$. **Middle:** Per-user forward-pass time (solid, left log axis) and normalized HR@3 (dashed, right axis) vs. per-query context size n_c; median across datasets, bands are 25th–75th percentile. **Right:** Normalized HR vs. pointer slots k'; bands are 25th–75th percentile.

ICR significantly on most benchmarks. RecPFN and EmbKNN-R are competitive on embedding-rich Amazon categories but fall well behind on ID-heavy or dense datasets (Dianping, ML-20M). Notably, EmbKNN-R—which uses random embeddings and no learned parameters—outperforms every zero-shot baseline that does not exploit a support context, providing empirical support for the prominence of $\mathcal{F}_{\text{lin}}$ in our projected-kernel hierarchy (Section 3.2.1).

Supervised comparison. Against supervised baselines trained per dataset with early stopping (up to 50 epochs), ICR (no per-dataset training) is the outright winner on 8 of 11 datasets (17 of 22 cells) and remains competitive on the remaining 3; the closest supervised competitors, BERT4Rec and VQ-Rec, trail by $\sim$20% on average. To the best of our knowledge, this is the first zero-shot recommender to match standard supervised baselines across a diverse suite of sequential recommendation benchmarks.

6 Analysis

6.1 Ablation Studies

Figure 2 shows HR@3 and MRR@3 deltas vs. the full model; more ablations can be found in Appendix C.

Architecture. Ablating identity-restricted cross-attention is catastrophic on Dianping (-36% HR@3, -56% MRR@3) while leaving the other five datasets nearly unchanged, exposing the attention dilution it is designed to prevent on dense catalogs. Removing non-causal context encoding degrades two of six datasets significantly, indicating that successor information in context is genuinely useful even if its value is not uniform across domains.

Synthetic prior. Dropping the extended SCM and training on the linear SCM alone significantly degrades two of six datasets and produces a small but consistent decline on three of the remaining four—indicating the extended SCM drives competitive performance on harder, denser domains. Within the extended SCM, the model is robust to hyperparameter misspecification: scaling all prior parameter ranges by $0.5\times$, $2\times$, or $3\times$ produces only small changes on every dataset.

6.2 Context Scaling

Sweeping the per-query context size n_c from 2 to 1024 (Figure 2; HR@3 normalized per dataset by the best score across all methods and n_c; latency measured on Amazon Games as a medium-sized representative dataset), ICR climbs monotonically with diminishing returns and plateaus around $n_c = 256$, while RecPFN (Tan et al., 2026) peaks early and then *decays*—the classic attention-dilution signature. On compute, ICR's per-user latency is essentially flat in n_c thanks to the identity-

Table 2: Efficiency comparison. **Bold**/underlined = best/second-best on the accuracy rows only; time and breakeven rows are not ranked (their relative "best" depends on serving scale).

	Arts					Dianping				
	ICR	RecPFN	SRPFN	VQ-Rec	BERT	ICR	RecPFN	SRPFN	VQ-Rec	BERT
Preproc. + Train (s)	109	113	71	11,391	11,946	29.6	35.6	18	4047	60,834
Inference (ms / user)	1.91	5.12	148	1.38	0.08	6.10	7.85	117	1.09	0.10
HR@3	**.2035**	.1723	.0407	.1861	.0585	**.0599**	.0115	.0073	.0038	.0157
MRR@3	**.1822**	.1412	.0325	.1717	.0502	**.0490**	.0082	.0044	.0019	.0096
Breakeven users		–	–	21M	6.5M		–	–	802k	10M

restricted cross-attention's bounded per-query cost, whereas RecPFN scales approximately linearly and reaches $\mathbf{29}\times$ slower at $n_c = 1024$. SRPFN (Kang et al., 2026) likewise shows flat latency across n_c, but both its latency and accuracy remain largely constant—providing more context yields no quality gain.

6.3 Efficiency

Table 2 extends the comparison to one representative method per class on two datasets of contrasting scale (methodology in Appendix E.3). ICR attains the best HR@3 and MRR@3 on both datasets while paying *no per-dataset training cost*, dominating the zero-shot baselines on both latency and quality. SRPFN's per-user inference is up to $75\times$ slower than ICR's, a consequence of its cross-encoder scoring that re-encodes every candidate item per query. The supervised baselines have lower per-user inference, but their large per-dataset training cost leads to high breakeven user counts (802k–21M). Below those crossovers ICR is strictly cheaper end-to-end on top of being more accurate; supervised models are only more efficient on a single domain served beyond that scale.

6.4 Embedding-Space Expansion

A natural extension of the pointer head is to use its output as a semantic anchor for retrieval beyond the support context: given the top-k' pointer predictions, we form a query by averaging the corresponding LLM embeddings (weighted by pointer probability) and retrieve additional items via cosine similarity over the full catalog, filling positions $k'+1$ through k. This lets the model produce recommendations of arbitrary length and surface items that the retriever may not have placed in the support context, without changing the model itself.

Figure 2 shows HR@50 normalized per dataset by the ceiling at $k' = k$ (no expansion, all k slots filled by the pointer head). Each additional pointer slot captures more of the ceiling than an embedding-filled slot, which is why we use the no-expansion setting as our primary evaluation mode. The embedding-fill curve still recovers a substantial fraction of the ceiling at low k', confirming that the embedding-space expansion is a viable mode of operation when out-of-context recommendations are desirable—for example, when the recommendation list is longer than the support context or when catalog coverage in the support context is limited.

6.5 Mechanistic Analysis

We probe the trained model to verify the mechanisms behind ICR's design; full probe definitions are in Appendix D.

Identity-restricted cross-attention bounds attention dilution. Figure 3 (a) measures the participation count of the pointer-head distribution—the effective number of context positions over which mass spreads—with and without the identity-restricted backbone. The pointer itself attends globally in both variants; nevertheless, the gap grows nearly linearly in n_c, showing that IRCA's upstream restriction yields representations that keep the final unrestricted attention focused as context grows.

Pointer-head K vectors encode neighbourhood geometry. The pointer head's keys score context positions by dot product with a learned query. Figure 3 (b) reports, on pairs of query and context

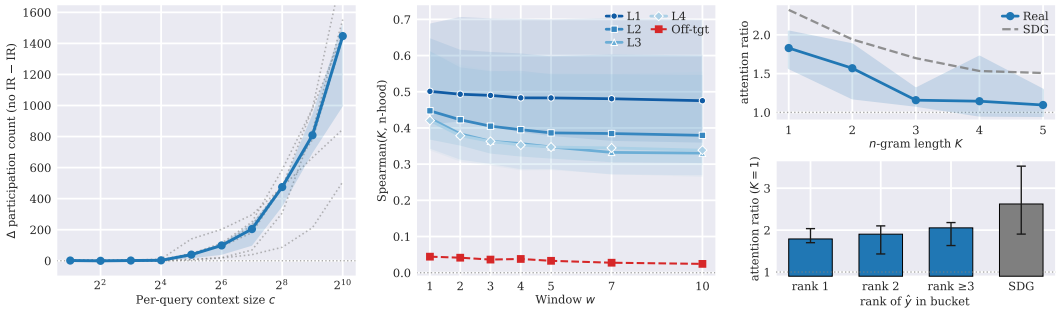


Figure 3: Mechanistic probes; bands are 25th–75th percentile across datasets throughout. **Left:** Participation-count gap (no-IR - IR) vs. per-query context size n_c. **Middle:** Spearman correlation between pointer-head K-cosine and neighbourhood overlap, vs. window w for each encoder layer (dashed: permuted-identity null). **Top right:** Layer-2 attention ratio (match / non-match) on real and synthetic data. **Bottom right:** The same $K=1$ ratio stratified by rank of $\hat{y}$.

positions that the query attends to, the Spearman correlation between pairwise K-cosine and pairwise recency-weighted Jaccard overlap of preceding-item windows. Trained K vectors correlate at $\sim$0.4–0.5 across window widths $w \in \{1, \ldots, 10\}$ while controls collapse to zero, showing that K encodes local neighbourhood structure on top of item identity—this is what lets the pointer pick *which occurrence* of a recurring item to attend to. The window-width dependence also sharpens with depth: shallower layers correlate uniformly across w, while deeper layers' correlation peaks at small w and falls off faster, consistent with progressively more local context tuning.

Cross-attention layer 2 implements context-aligned induction. The four cross-attention layers are indexed 0–3; we focus on layer 2, where the induction signal is strongest. For each query, Figure 3 (c) computes the ratio of layer-2 attention on bucket positions whose preceding K-item window matches the query's, to attention on non-matching positions. The ratio is 1.83 at $K=1$ on real data (cross-dataset median) and decays smoothly through $K=5$, with the training-distribution reference exhibiting the same shape at uniformly stronger magnitude. Stratifying further by the rank of the predicted item $\hat{y}$ in the bucket's successor-frequency ordering (Figure 3 d), the $K=1$ ratio is 1.79 when $\hat{y}$ is the modal successor and grows to 2.05 when $\hat{y}$ is a minority successor (rank ≥ 3, where bucket voting would actively pick a different item). The amplification rules out a "modal-successor wins" explanation: layer 2's selectivity is largest precisely when context-aligned override is needed.

7 Conclusion

We introduced ICR, a zero-shot sequential recommender built around a theoretically-grounded synthetic prior, an ID-based architecture with identity-restricted cross-attention, and a pointer head that predicts directly over the support context. Together these design choices let a single model adapt to a new domain in one forward pass while scaling monotonically with context size—producing, to the best of our knowledge, the first zero-shot recommender that matches standard supervised baselines across a diverse suite of sequential recommendation benchmarks. The result demonstrates that inference-time adaptation can match, and often outperform, dataset-specific fine-tuning on sequential recommendation.

Several directions remain open. ICR's prior is grounded in the projected-kernel structure of existing sequential recommenders; extending it to capture richer side-information (item attributes, temporal context, social signal) would broaden the class of dynamics the model can recover from a support context. The identity-restricted attention mechanism likewise generalises beyond next-item prediction—we expect identity-keyed sparsity to be useful in any in-context setting where the same token recurs across long contexts.

AI USE STATEMENT

In this work, we used generative AI tools for discovery of related work, implementing the ICR codebase (but excluding core components of the evaluation code), mathematical work related to connection netween existing sequential recommender methods and the projected kernel model and assistance in formalizing the proofs of mathematical claims and ideation for some of the mechanistic probes.

We have not used generative AI tools in conceiving or developing the core components of the model or synthetic data generation model, formulating the hypotheses, designing the evaluation methodology and experimental protocol, formatting or cleaning of evaluation datasets or generating synthetic datasets.

Additionally, we used generative AI tools for paraphrasing sections of the paper to improve readability and conciseness, summarizing and analyzing existing literature and brainstorming at various stages in the research process.

We have reviewed all AI-assisted work. In particular, we have verified the correctness of the LLM-generated code with particular focus on integration of the evaluation code into the LLM-generated codebase to ensure that the generated results are correct. We have further replicated the results from a benchmark using the LLM integrated evaluation logic to ensure that the code is accurate. We have carefully read the sections edited using generative AI tools to ensure that the text still carries our intended meaning. We have also checked that the proofs generated by generative AI tools according to our explicit and specific instructions are correct and represent our intentions We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Code for reproducing the results in this paper is available at `https://osf.io/knfsu/overview?view_only=d0d3a67327294fbda212251437e96161`. This includes the synthetic data generation code which will allow for training a model from scratch, a trained model checkpoint, and evalaution code. It also includes a single test dataset for which the results in the paper can be reproduced.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Alan Arazi, Eilam Shapira, and Roi Reichart. Tabstar: A tabular foundation model for tabular data with text fields. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Aryaman Arora, Dan Jurafsky, Christopher Potts, and Noah D. Goodman. Bayesian scaling laws for in-context learning. *arXiv preprint arXiv:2410.16531*, 2024.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Jeongwhan Choi, Jongwoo Kim, Woosung Kang, and Noseong Park. Learning posterior predictive distributions for node classification from synthetic graph priors. In *International Conference on Learning Representations (ICLR)*, 2026.

HAO DING, Anoop Deoras, Bernie Wang, and Hao Wang. Zero-shot recommender systems. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, et al. A survey on in-context learning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 1107–1128, 2024.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David P. Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. In *International Conference on Learning Representations (ICLR)*, 2024.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):1–19, 2015.

Ruining He and Julian McAuley. Fusing similarity models with Markov chains for sparse sequential recommendation. In *2016 IEEE 16th international conference on data mining (ICDM)*, pp. 191–200, 2016.

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

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *arXiv preprint arXiv:2501.02945*, 2025.

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 585–593, 2022.

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.

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

Chengkai Huang, Hongtao Huang, Tong Yu, Kaige Xie, Junda Wu, Shuai Zhang, Julian McAuley, Dietmar Jannach, and Lina Yao. A survey of foundation model-powered recommender systems: From feature-based, generative to agentic paradigms. *arXiv preprint arXiv:2504.16420*, 2025.

Dietmar Jannach and Malte Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, pp. 306–310, 2017.

Yangqin Jiang, Xubin Ren, Lianghao Xia, Da Luo, Kangyi Lin, and Chao Huang. Recgpt: A foundation model for sequential recommendation. *arXiv preprint arXiv:2506.06270*, 2025.

Clark Mingxuan Ju, Tong Zhao, Leonardo Neves, Liam Collins, Bhuvesh Kumar, Jiwen Ren, Lili Zhang, Wenfeng Zhuo, Vincent Zhang, Xiao Bai, Jinchao Li, Karthik Iyer, Zihao Fan, Yilun Xu, Yiwen Chen, Peicheng Yu, Manish Malik, and Neil Shah. Semantic ids for recommender systems at snapchat: Use cases, technical challenges, and design choices. In *arXiv preprint arXiv:2604.03949*, 2026.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pp. 197–206. IEEE, 2018.

Woosung Kang, Jiwon Jeong, Jonghyeok Shin, Jeongwhan Choi, and Noseong Park. One sequential recommendation model pretrained from synthetic priors predicts multiple datasets. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026.

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. Text is all you need: Learning language representations for sequential recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1258–1267, 2023.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pp. 1419–1428, 2017.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Jiahao Liu, Xueshuo Yan, Dongsheng Li, Guangping Zhang, Hansu Gu, Peng Zhang, Tun Lu, Li Shang, and Ning Gu. Improving llm-powered recommendations with personalized information. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2560–2565, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024b.

Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. STAMP: Short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1831–1839, 2018.

Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 2017.

Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. Llm-rec: Personalized recommendation via prompting large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 583–612, 2024.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Hamidreza Kamkari, Alex Labach, Jesse C Cresswell, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony L Caterini. Tabdpt: Scaling tabular foundation models. *arXiv preprint arXiv:2410.18164*, 2024.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. *arXiv preprint arXiv:2502.05564*, 2025.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan 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 (NeurIPS)*, 2023.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World Wide Web*, pp. 811–820, 2010.

Leheng Sheng, An Zhang, Yi Zhang, Yuxin Chen, Xiang Wang, and Tat-Seng Chua. Language representations can be what recommenders need: Findings and potentials. In *ICLR*, 2025.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 1441–1450, 2019.

En Zhi Tan, Jia Xiang Lim, Bryan Lijie Chew, Tze Minh Ng, and Benjamin Yan Han Yap. Recpfn: Prior-fitted networks for in-context-based recommendations. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1731–1742, 2026.

Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pp. 17–22, 2016.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pp. 565–573, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

Junting Wang, Adit Krishnan, Hari Sundaram, and Yunzhe Li. Pre-trained neural recommenders: A transferable zero-shot framework for recommendation systems. *arXiv preprint arXiv:2309.01188*, 2023.

Lei Wang and Ee-Peng Lim. The whole is better than the sum: Using aggregated demonstrations in in-context learning for sequential recommendation. *arXiv preprint arXiv:2403.10135*, 2024.

Lanling Xu, Zhen Tian, Gaowei Zhang, Junjie Zhang, Lei Wang, Bowen Zheng, Yifan Li, Jiakai Tang, Zeyu Zhang, Yupeng Hou, et al. Towards a more user-friendly and easy-to-use benchmark library for recommender systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2837–2847, 2023.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. In *International Conference on Learning Representations (ICLR)*, 2025.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. *arXiv preprint arXiv:2402.17152*, 2024.

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, pp. 4320–4326, 2019.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pp. 1893–1902, 2020.

APPENDIX CONTENTS

A Full Benchmark Results **17**

A.1 Zero-Shot and Supervised Benchmarks 17

A.2 Non-Parametric Baselines 18

B Context Coverage Analysis **18**

C Full Ablation Breakdown **20**

D Mechanistic Probes **21**

D.1 Attention Dilution (Figure 3a) 21

D.2 Pointer-Head Key Geometry (Figure 3b) 21

D.3 Layer-2 Context-Aligned Induction (Figure 3c) 21

D.4 Induction Amplification by $\hat{y}$ Rank (Figure 3d) 22

E Experimental Details **22**

E.1 Datasets 22

E.2 Baselines 23

E.3 Efficiency measurement conventions 24

F Implementation Details **25**

F.1 Multi-Head Attention and Feed-Forward Block 25

F.2 Training Details 25

F.3 Synthetic Prior Hyperparameter Ranges 26

G Synthetic Prior: Full Specification **26**

G.1 Linear SCM ($\mathcal{F}_{\text{lin}}$) 26

G.2 Extended SCM ($\mathcal{F}_{\text{full}}$) 27

G.3 Full Generation Procedure 27

H Context Selection **27**

H.1 Inverted-Index Matcher 27

H.2 Computational Cost 29

I Identity-Restricted Cross-Attention: Implementation **29**

J Projected Kernel Models: Instantiations and Proofs **30**

J.1 Sequential Recommenders as Projected Kernel Models 30

J.2 Proof of Proposition 2 33

J.3 Proof of Theorem 3 33

K Proof: Absolute Coordinate Dependence Is Not Justified **34**

K.1 Setup and Notation 34
K.2 Core Assumption 35
K.3 Key Definitions 36
K.4 Loss 36
K.5 Proof via Symmetrization 36
K.6 Implications 38

A Full Benchmark Results

A.1 Zero-Shot and Supervised Benchmarks

Table 3: Zero-shot comparison at HR@3 and MRR@3. ICR results are averaged over 4 independent training runs.

Dataset	@3	ICR	RecPFN	SRPFN	RecFormer	RecGPT	UniSRec	VQ-Rec	EmbKNN	EmbKNN-R	Improv.	Rank
Appl.	HR	**.2589**	.0993	.2340	.0764	.1064	.0426	.1064	.0638	.1277	+10.6%	1
	MRR	**.2125**	.0839	.1608	.0729	.0993	.0319	.0981	.0414	.1182	+32.2%	1
Arts	HR	**.2029**	.1723	.0407	.1533	.0384	.1247	.1228	.1530	.1670	+17.8%	1
	MRR	**.1812**	.1412	.0325	.1331	.0370	.1076	.1171	.1230	.1368	+28.3%	1
Games	HR	**.0737**	.0482	.0472	.0326	.0336	.0140	.0320	.0202	.0480	+52.8%	1
	MRR	**.0595**	.0394	.0306	.0240	.0326	.0101	.0298	.0116	.0373	+51.1%	1
Groc.	HR	**.1078**	.0769	.0525	.0434	.0402	.0237	.0422	.0368	.0625	+40.2%	1
	MRR	**.0920**	.0605	.0363	.0320	.0383	.0162	.0389	.0236	.0486	+51.9%	1
Office	HR	**.1204**	.0781	.0566	.0517	.0479	.0473	.0480	.0346	.0767	+54.2%	1
	MRR	**.1035**	.0567	.0432	.0389	.0437	.0338	.0426	.0202	.0566	+82.7%	1
Sci.	HR	**.0971**	.0824	.0785	.0582	.0626	.0254	.0660	.0569	.0831	+16.9%	1
	MRR	**.0775**	.0687	.0531	.0500	.0590	.0196	.0497	.0434	.0569	+12.8%	1
Soft.	HR	**.1840**	.1182	.1592	.0977	.0988	.0977	.0929	.0700	.1158	+15.5%	1
	MRR	**.1557**	.0921	.1084	.0854	.0892	.0820	.0824	.0294	.0923	+43.7%	1
Dianp.	HR	**.0607**	.0115	.0073	.0029	.0002	.0006	.0001	.0011	.0044	+428.0%	1
	MRR	**.0490**	.0082	.0044	.0024	.0002	.0003	.0000	.0005	.0020	+497.0%	1
ML-20M	HR	**.0805**	.0066	.0336	.0014	.0005	.0031	.0007	.0042	.0133	+139.6%	1
	MRR	**.0542**	.0036	.0192	.0011	.0002	.0019	.0003	.0026	.0058	+182.7%	1
Wash.	HR	**.0253**	.0121	.0170	.0054	.0156	.0032	.0148	.0019	.0144	+48.7%	1
	MRR	**.0200**	.0114	.0117	.0043	.0152	.0026	.0146	.0013	.0144	+31.4%	1
Yelp	HR	.0214	.0195	.0144	.0103	.0174	.0047	.0180	.0182	**.0219**	-2.4%	2
	MRR	.0155	.0165	.0091	.0083	.0170	.0039	.0171	.0133	**.0191**	-18.6%	5

Bold = best, underline = 2nd, dotted = 3rd. Baseline details in Appendix E.2.

Table 4: ICR (zero-shot) vs. supervised baselines trained for 50 epochs per dataset. ICR results are averaged over 4 independent training runs. $^\wedge$ = LLM-embedding variant. * = statistically significant improvement over 2nd-best ($p < 0.05$, one-sided Welch's t-test against the baseline's seed distribution).

Dataset	@3	ICR	BERT	BERT$^\wedge$	FDSA	GRU	GRU$^\wedge$	SAS	SAS$^\wedge$	UniSRec	VQ-Rec	Improv.	Rank
Appl.	HR	**.2589***	.0213	.1285	.0514	.0213	.0602	.0674	.1505	.1117	.1153	+72.0%	1
	MRR	**.2125***	.0097	.1001	.0236	.0115	.0407	.0289	.1051	.0919	.1070	+98.6%	1
Arts	HR	**.2029***	.0585	.0094	.0215	.0082	.0218	.0344	.0218	.1734	.1861	+9.0%	1
	MRR	**.1812***	.0502	.0063	.0144	.0056	.0154	.0302	.0156	.1568	.1717	+5.5%	1
Games	HR	**.0737***	.0644	.0067	.0302	.0176	.0296	.0495	.0330	.0556	.0643	+14.4%	1
	MRR	**.0595***	.0531	.0044	.0219	.0110	.0263	.0376	.0284	.0445	.0507	+12.0%	1
Groc.	HR	**.1078***	.0985	.0140	.0261	.0314	.0766	.0427	.0270	.0836	.0996	+8.2%	1
	MRR	**.0920***	.0853	.0094	.0160	.0201	.0663	.0342	.0198	.0661	.0851	+7.8%	1
Office	HR	**.1204***	.1100	.0074	.0371	.0301	.0141	.0465	.0905	.0920	.1131	+6.5%	1
	MRR	**.1035***	.0989	.0046	.0279	.0220	.0117	.0378	.0784	.0771	.1008	+2.7%	1
Sci.	HR	**.0971***	.0701	.0328	.0560	.0337	.0332	.0665	.0784	.0687	.0673	+23.9%	1
	MRR	**.0775***	.0629	.0244	.0444	.0274	.0272	.0461	.0649	.0520	.0521	+19.4%	1
Soft.	HR	**.1840***	.1218	.0750	.1022	.0504	.0739	.0928	.0981	.1632	.1324	+12.8%	1
	MRR	**.1557***	.1024	.0553	.0844	.0393	.0622	.0750	.0756	.1198	.0997	+29.9%	1
Dianp.	HR	**.0607***	.0157	.0001	.0025	.0049	.0000	.0045	.0000	.0037	.0038	+288.0%	1
	MRR	**.0490***	.0096	.0001	.0016	.0030	.0000	.0025	.0000	.0022	.0019	+408.6%	1
ML-20M	HR	.0805	**.1123**	.0107	.0289	.0507	.0743	.0435	.0031	.0444	.0614	-28.4%	2
	MRR	.0542	**.0778**	.0064	.0185	.0323	.0512	.0278	.0010	.0289	.0388	-30.4%	2
Wash.	HR	.0253	**.0269**	.0024	.0097	.0160	.0051	.0191	.0003	.0210	.0263	-6.1%	3
	MRR	**.0200***	.0175	.0014	.0060	.0103	.0039	.0126	.0002	.0144	.0186	+7.5%	1
Yelp	HR	.0214	.0216	.0026	.0090	.0102	.0027	.0124	.0008	.0177	**.0264**	-19.1%	3
	MRR	.0155	.0140	.0017	.0057	.0064	.0019	.0081	.0004	.0129	**.0192**	-19.0%	2

Bold = best, underline = 2nd, dotted = 3rd. BERT = BERT4Rec, SAS = SASRec, GRU = GRU4Rec. Baseline details in Appendix E.2.

Tables 3 and 4 contain the per-dataset HR@3 and MRR@3 numbers summarised in Section 5.2; datasets, splits, evaluation protocol, and baseline configurations are described in Appendix E.1 and Appendix E.2.

A.2 Non-Parametric Baselines

Table 5: Comparison against non-parametric baselines at HR@3 and MRR@3. ICR results are averaged over 4 independent training runs. * = statistically significant improvement over 2nd-best ($p < 0.05$, one-sided Welch's t-test; only where both methods have ≥ 2 seeds).

Dataset	@3	ICR	Popular	Co-occur (last)	Co-occur (all)	Successor	Context Succ.	EmbKNN	EmbKNN-R	Improv.	Rank
Appl.	HR	**.2589***	.0213	.0745	.0691	.2110	.2163	.0638	.1277	+19.7%	1
	MRR	**.2125***	.0071	.0482	.0508	.1622	.1652	.0414	.1182	+28.6%	1
Arts	HR	**.2029***	.0026	.0418	.0387	.1762	.1775	.1530	.1670	+14.3%	1
	MRR	**.1812***	.0013	.0297	.0271	.1461	.1540	.1230	.1368	+17.7%	1
Games	HR	**.0737***	.0027	.0100	.0061	.0507	.0502	.0202	.0480	+45.4%	1
	MRR	**.0595***	.0022	.0065	.0037	.0372	.0373	.0116	.0373	+59.4%	1
Groc.	HR	**.1078***	.0006	.0147	.0103	.0827	.0845	.0368	.0625	+27.5%	1
	MRR	**.0920***	.0003	.0091	.0065	.0637	.0667	.0236	.0486	+37.9%	1
Office	HR	**.1204***	.0004	.0242	.0129	.0907	.0930	.0346	.0767	+29.4%	1
	MRR	**.1035***	.0002	.0159	.0081	.0720	.0750	.0202	.0566	+38.1%	1
Sci.	HR	**.0971**	.0351	.0267	.0211	.0790	.0796	.0569	.0831	+16.9%	1
	MRR	**.0775***	.0167	.0178	.0145	.0604	.0615	.0434	.0569	+26.0%	1
Soft.	HR	**.1840***	.0507	.0332	.0277	.1448	.1420	.0700	.1158	+27.1%	1
	MRR	**.1557***	.0462	.0207	.0182	.1097	.1098	.0294	.0923	+41.8%	1
Dianp.	HR	**.0607***	.0002	.0019	.0008	.0077	.0074	.0011	.0044	+686.1%	1
	MRR	**.0490***	.0001	.0013	.0004	.0048	.0047	.0005	.0020	+916.3%	1
ML-20M	HR	**.0805***	.0099	.0025	.0022	.0203	.0454	.0042	.0133	+77.0%	1
	MRR	**.0542***	.0058	.0015	.0014	.0128	.0291	.0026	.0058	+86.4%	1
Wash.	HR	**.0253**	.0017	.0017	.0014	.0086	.0089	.0019	.0144	+75.5%	1
	MRR	**.0200**	.0008	.0010	.0007	.0057	.0059	.0013	.0144	+38.7%	1
Yelp	HR	.0214	.0033	.0020	.0013	.0049	.0054	.0182	**.0219**	-2.4%	2
	MRR	.0155	.0020	.0013	.0008	.0031	.0034	.0133	**.0191**	-18.6%	2

Bold = best, underline = 2nd, dotted = 3rd. Baseline details in Appendix E.2.

Table 5 compares ICR against the non-parametric baselines defined in Appendix E.2. The co-occurrence variants, which count items appearing anywhere in the same sequence rather than directly after the query's last item, are the weakest. *Successor* is stronger: it scores items by global first-order transition counts across all training sequences. *Context Succ.* holds retrieval fixed — it applies the same successor vote but restricted to the support sequences retrieved for each query — making it the most direct baseline for isolating the model's contribution from the retrieval procedure.

Successor and *Context Succ.* perform quite close to each other across most datasets, suggesting the retrieved context is a reasonable proxy for the global transition distribution. ICR outperforms all count-based baselines on all 11 datasets. The significant performance gap between *Context Succ.* and ICR on most datasets suggests the transformer/pointer model contributes beyond what retrieval alone provides — a plausible reading is that ICR estimates the successor distribution from the full in-context neighbourhood rather than counting a single transition, letting it specialise where the useful signal is local. This is consistent with the context-aligned induction mechanism characterised in Appendix D.3.

B Context Coverage Analysis

The pointer head predicts exclusively over items present in the retrieved support context, so the fraction of test targets appearing in context is a hard upper bound on pointer-only HR@k. Figure 4 plots four quantities against per-query context size n_c: *context coverage* (fraction of targets anywhere in the retrieved sequences), *context successor coverage* (fraction of targets that are a direct successor of the query's last item in the retrieved sequences), *Context Succ. HR@3* (count-based successor ranking over the same context), and ICR HR@3. The flat line shows Global Succ. HR@3 (same ranking but over the global training transition matrix, independent of n_c).

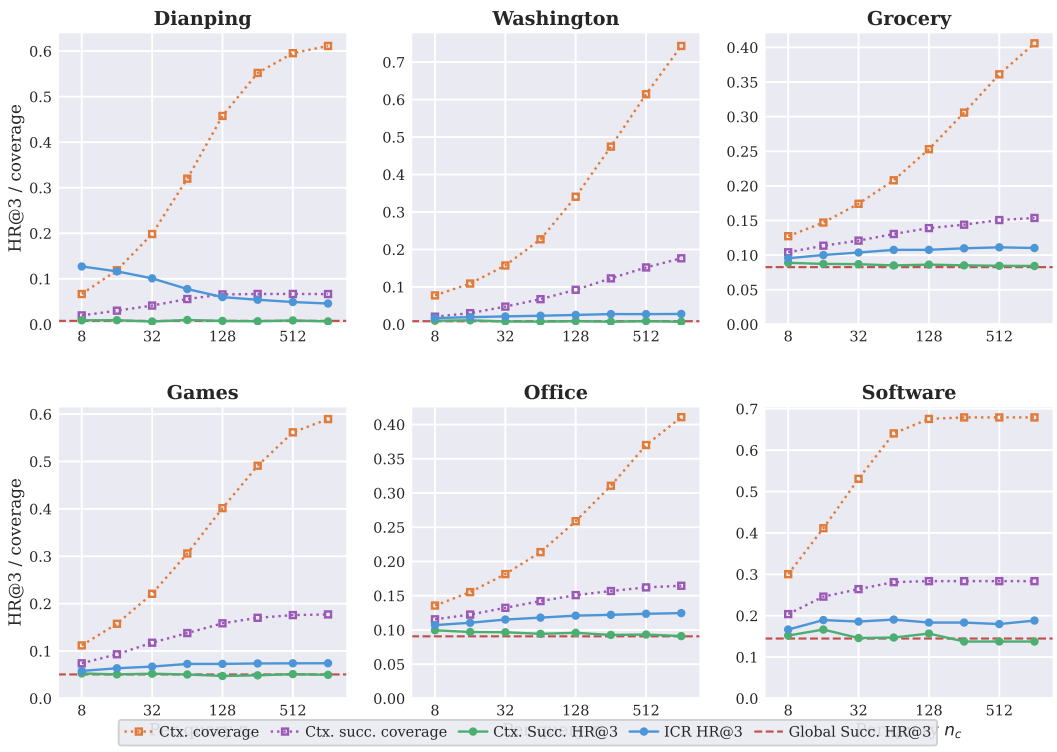


Figure 4: Coverage and successor performance vs. per-query context size n_c for six datasets. Varying lines: ICR HR@3 (blue), context coverage (orange), context successor coverage (purple), and Context Succ. HR@3 (green). Flat dashed line: Global Succ. HR@3 (red). See text for definitions.

Coverage ceilings. Context coverage (orange) rises with n_c as expected, reaching 25–67% at the default n_c=128 and diminishing returns by n_c≈512; it constitutes the hard ceiling for pointer-only prediction, since the pointer head cannot retrieve items absent from context. Context successor coverage (purple)—the fraction of targets that are a *direct* successor of the query's last item in the retrieved sequences—is substantially lower, sitting well below context coverage on all datasets. This suggests that the pointer ceiling is less limiting in practice than it may appear: even when the target is present in context, it is often not supported by direct successor transitions, which is arguably a more informative measure of how well the context structurally enables the correct prediction. That said, ICR is not restricted to successor transitions and can exploit broader in-context patterns, so the relationship between context successor coverage and model performance is indirect. Items absent from context entirely can be recovered via embedding-space expansion (Section 6.4).

Retrieved context as a proxy for global transitions. Comparing Context Succ. HR@3 (green) to Global Succ. HR@3 (red dashed) shows that the two are close across all six datasets. Even at small n_c, the retrieved neighbourhood appears to be a reasonably representative sample of the global first-order transition distribution, so count-based ranking over the context captures most of the signal available globally. At the same time, the additional specificity of retrieval does not translate into a meaningful performance gain for count-based successor ranking beyond what global statistics provide. Washington illustrates a particular failure mode of count-based ranking under retrieval: context successor coverage rises with n_c, but HR@3 remains flat—the target becomes reachable as a successor in more retrieved sequences, but too many competing successors dilute the count-based signal.

Model contribution beyond retrieval. ICR (blue) consistently exceeds Context Succ. HR@3 (green) on all six datasets, showing that the transformer adds value beyond local count statistics. Notably, on five of six datasets the blue line increases with n_c while the green line remains flat and

close to the global successor baseline. This suggests the model is better at extracting signal from larger context—possibly by being more robust to the noise introduced by less-relevant sequences at large n_c—whereas count-based ranking appears to saturate early. The gap between the two therefore likely reflects the model's ability to leverage context structure beyond simple successor frequency.

C Full Ablation Breakdown

Section 6.1 reports HR@3 and MRR@3 deltas aggregated over six evaluation datasets spanning a range of catalog densities: Dianping, Washington, Amazon Grocery, Amazon Games, Amazon Office, and Amazon Software. This appendix gives the per-dataset breakdown.

All entries in Table 6 are signed percent changes against the base model ($d = 256$, $L = 4$, $n_c = 128$), whose absolute HR@3 and MRR@3 are given in the top row. The no-identity-restricted-cross-attention variant exceeds memory at the baseline batch size and is therefore evaluated at a smaller batch but the same per-query context size $n_c = 128$, keeping its row comparable to the rest.

Table 6: Relative performance of various ablation experiments (expressed as percentage change).

Experiment	Dianp.		Wash.		Groc.		Games		Office		Soft.	
@3	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR	HR	MRR
ICR (base)	0.0599	0.0490	0.0252	0.0201	0.1077	0.0921	0.0729	0.0598	0.1209	0.1048	0.1834	0.1552
Architecture — Model Size (base d=256, L=4)												
$d = 64$	-3.8	-7.1	-7.5	-6.1	-4.4	-6.8	-10.4	-12.1	-4.1	-9.6	-13.2	-11.1
$d = 128$	-3.8	-3.2	-4.8	-3.5	-1.9	-3.9	-4.8	-7.9	-3.8	-8.3	-7.9	-9.3
$d = 512$	+2.2	+1.7	-4.0	-3.9	+2.0	+1.7	+1.4	-1.5	+0.7	-2.4	-3.9	-0.9
$d = 1024$	+3.8	+0.5	-1.6	-5.3	-2.7	-4.5	-4.9	-9.2	-2.8	-7.0	+1.3	-1.2
$L = 1$ layer	-3.8	-1.4	-28.2	-24.3	-13.9	-19.4	-26.1	-26.4	-13.4	-21.5	-19.1	-19.8
$L = 2$ layers	-13.0	-17.9	-6.0	-4.1	-3.4	-6.9	-5.6	-7.9	-3.9	-9.7	-11.2	-12.3
$L = 6$ layers	-8.8	-13.7	-1.2	-2.6	+0.6	+1.7	0.0	-1.9	-0.2	-0.5	+1.3	-0.4
Architecture — Design Choices												
No identity-restricted xattn	-36.4	-56.2	-1.2	-1.6	-0.9	+0.8	+1.6	-0.5	-0.3	-0.3	+1.3	+0.8
Causal context encoding	-7.3	-4.7	-9.9	-7.1	+0.1	+2.2	+2.9	+2.3	+0.2	+1.8	+5.3	+5.3
Synthetic Prior												
Only linear SCM	-10.0	-11.5	-6.0	-5.7	0.0	-0.3	+0.1	-2.4	-0.7	-1.3	+2.6	+1.6
Prior width $0.5\times$	-2.2	-2.0	-2.8	-2.1	+1.7	+0.9	-0.7	-2.6	-0.7	-2.0	-0.7	+0.4
Prior width $2\times$	+1.0	-0.4	-5.2	-3.6	-0.3	+0.6	+0.8	-1.4	-0.1	0.0	-1.3	+0.9
Prior width $3\times$	+2.5	+1.7	0.0	-1.2	-0.2	-0.1	+1.2	-0.4	-0.4	-0.5	+2.0	+1.7
Context-Selector Breadth (default α=10)												
$\alpha = 1$	+39.2	+37.2	-7.5	-4.8	-1.4	+0.3	+0.4	+0.6	-1.8	0.0	-4.6	-2.3
$\alpha = 100$	-42.6	-44.2	+2.4	+1.2	+1.2	+1.7	+0.8	-0.3	-0.3	+0.4	0.0	0.0
$\alpha = 10^6$	-44.1	-43.8	-1.2	-1.0	+1.4	+2.0	+1.8	+0.9	-0.2	0.0	0.0	0.0

Architecture. Removing the identity-restricted cross-attention is mild on the sparse Amazon datasets but collapses on Dianping (-36% HR@3, -56% MRR@3), so the mechanism matters most on dense catalogs where many candidates compete per context. Halving the depth is the most damaging size change, with double-digit losses on five of the six datasets; halving the width is milder. Larger widths and depths produce mixed results across datasets, so the base configuration is already close to the best size we tested.

Synthetic prior. Varying the prior width by $0.5\times$, $2\times$, or $3\times$ leaves performance close to the base on every dataset, indicating low sensitivity to this hyperparameter. Restricting the generator to the linear SCM regime is more visible: a 10–12% drop on Dianping and a smaller but consistent loss on Washington, with little movement on the Amazon datasets. The non-linear regime therefore contributes most where the catalog dynamics are furthest from simple co-occurrence.

Context-selector breadth. The inverted-index context selector has one tunable knob, the breadth multiplier α, which controls how widely the selector searches before stopping. The sparse Amazon datasets are nearly insensitive to α, but the dense Dianping dataset reacts strongly: wider search ($\alpha \geq 100$) loses around 40%, while the tightest setting ($\alpha = 1$) *adds* more than 30%. The production default $\alpha = 10$ is the only choice that is never far from the per-dataset best, and we use it throughout the main results.

D MECHANISTIC PROBES

This appendix describes the four mechanistic probes summarised in Section 6.5 and reports the numbers behind Figure 3. All probes are evaluated on the trained ICR model at the paper-default inference configuration, aggregated across six public datasets (Amazon Games, Amazon Office, Amazon Software, Amazon Grocery, Dianping, Washington). Synthetic-data references use ten batches drawn from the synthetic prior used at training time.

D.1 ATTENTION DILUTION (FIGURE 3A)

We measure how the pointer head's attention distribution responds to growing per-query context size n_c. For each query the pointer head emits a softmax distribution α over *all* context positions: IRCA restricts only the preceding backbone layers, not the pointer itself. We summarise pointer concentration by the participation count $\exp\big(H(\alpha)\big)$, the effective number of positions over which its mass is spread. The headline quantity is the cross-dataset median participation count of an ablation trained without IRCA minus that of the full model, plotted against n_c. A growing positive gap shows that restricted routing in the backbone produces representations that keep the subsequent unrestricted pointer focused as context grows.

The per-query context size is varied by holding the total support pool fixed and sweeping the number of queries per batch, which produces $n_c \in \{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024\}$. Within each $(n_c, \text{dataset}, \text{model})$ cell the per-query participation count is summarised by its median across 512 queries.

D.2 POINTER-HEAD KEY GEOMETRY (FIGURE 3B)

The pointer head's attention is a softmax of dot products between a single query vector and per-context-position key vectors $k_\tau = W_K \bar{h}_\tau$, where $\bar{h}_\tau$ is the post-encoder representation of context position τ (averaged across the four pointer-head subspaces). We test whether the geometry these keys induce on context positions reflects local sequential context.

For each dataset we draw $50{,}000$ unordered position pairs (τ_1, τ_2), restricted to pairs whose source sequences differ. For every pair we compute (i) the key-space cosine $\rho_K = \cos(k_{\tau_1}, k_{\tau_2})$ and (ii) the recency-weighted Jaccard overlap J_w of the two positions' preceding w-item windows, excluding the central item so that shared identity at τ_1 and τ_2 does not mechanically inflate overlap. The reported quantity is the Spearman rank correlation between ρ_K and J_w, swept over window half-widths $w \in \{1, \ldots, 10\}$ and reported per encoder layer.

Two null curves accompany the trained-key correlation: (i) a permuted-identity null, in which item ids are shuffled before J_w is computed, and (ii) a random-key null, in which K is replaced by matched-scale Gaussian vectors with the identity-restricted mask re-applied. Both null curves collapse to zero, calibrating the magnitude of the observed correlation. Queries are restricted to cross-sequence position pairs because this is the comparison the pointer head performs at inference, when the same item appears in multiple support sequences.

D.3 LAYER-2 CONTEXT-ALIGNED INDUCTION (FIGURE 3C)

For each query we identify the bucket: the set of context positions whose item id equals the query's last seen item M — exactly the positions the identity-restricted mask exposes. Let $(s^c_1, s^c_2, \ldots)$ denote the source sequence of context position p (so $s^c_p = M$), and let $(s^q_1, \ldots, s^q_T)$ denote the query sequence (with $s^q_T = M$). For each bucket position p and each $K \in \{1, \ldots, 5\}$ we set $\mathrm{match}_K(p) = 1$ if

$$(s^c_{p-K}, \ldots, s^c_{p-1}) \;=\; (s^q_{T-K}, \ldots, s^q_{T-1}),$$

i.e. the K items immediately preceding p in its source sequence equal the K items immediately preceding M in the query, and $\mathrm{match}_K(p) = 0$ otherwise. At each cross-attention layer ℓ we then compute the within-bucket attention-mass ratio

$$r^{(K)}_\ell \;=\; \frac{\mathrm{mean}\big(\alpha_\ell \mid \mathrm{match}_K = 1\big)}{\mathrm{mean}\big(\alpha_\ell \mid \mathrm{match}_K = 0\big)}.$$

A ratio above 1 means the layer directs more attention to bucket positions whose preceding context resembles the query's preceding context — the textbook signature of an induction head. The per-query ratio is computed only when both match groups are non-empty, the per-dataset summary is the median across queries, and the figure reports the median across the six datasets with the inter-quartile range shaded.

We restrict the population to queries with bucket size at least 5 and non-self-loop predictions ($\hat{y} \neq M$), so that the K-gram-match counting is well-defined and not dominated by the trivial repeat-prediction case. The same filter is applied to the synthetic reference.

Preceding context vs. successor identity. An earlier formulation matched on the *successor* of each bucket position rather than its preceding context. That variant collapsed to ratio ≈ 1 on real data. The encoder is non-causal, so each position's representation aggregates both preceding and following items; the natural axis on which Layer-2 discriminates is therefore preceding-context similarity to the query, not successor identity. Changing the match definition surfaces the signal.

D.4 Induction Amplification by $\hat{y}$ Rank (Figure 3d)

This probe asks whether the Layer-2 induction signal is doing real work or merely tracks bucket majority voting. For each query we rank the items in the bucket by their frequency as the successor of M and stratify queries by the rank of the predicted item $\hat{y}$: rank 1 is the modal successor of M within the bucket, rank 2 the runner-up, and rank ≥ 3 a minority successor — in the last stratum, voting on bucket-successor counts alone would actively select a different item. Within each stratum we report the cross-dataset median of the Layer-2 $K{=}1$ ratio defined above, with the synthetic-distribution reference shown for comparison.

The stratification distinguishes two hypotheses. If the Layer-2 ratio reflects passive bucket-majority voting, it should be highest at rank 1 and decrease for rare predictions. If it implements a genuine context-aligned override — routing attention more sharply when bucket voting alone would be wrong — it should *increase* with rank. Observed ratios are 1.79 at rank 1, 1.90 at rank 2, and 2.05 at rank ≥ 3, monotonically increasing as predicted by the latter hypothesis. The synthetic reference sits higher still, at 2.62, indicating that the same mechanism operates more cleanly on the training distribution.

E Experimental Details

E.1 Datasets

We evaluate on 11 public datasets drawn from four distinct sources, covering e-commerce, entertainment, restaurants, and local services across different geographies and platforms. This diversity ensures that results are not specific to any one interaction pattern or item domain.

- **Amazon Product Reviews** (Hou et al., 2024): Seven product categories from the 2023 Amazon review dataset—Appliances, Arts & Crafts, Video Games, Grocery, Office Products, Scientific & Industrial, and Software. Full catalog sizes (all items eligible for recommendation) range from ∼22K (Software) to ∼307K (Office) items, with per-user sequence lengths between 3.7 and 9.8 interactions on average.
- **MovieLens-20M** (Harper & Konstan, 2015): 20 million movie ratings from the MovieLens platform (∼27K items, ∼138K users). One of the densest benchmarks in our suite, with users averaging ∼144 interactions.
- **Dianping**: Restaurant and service reviews from a major Chinese O2O (online-to-offline) platform (∼338K items). Represents a linguistically and culturally distinct domain from the English-language datasets.
- **Washington**: Local business interactions from the Washington, D.C. metropolitan area (∼120K items). Large, dense dataset with ∼20 interactions per user on average.
- **Yelp**: Business reviews (restaurants, retail, services) from the Yelp open dataset (∼150K items). Broad geographic and category coverage, similar in structure to Dianping but in an English-language context.

Table 7 reports statistics after loading and filtering (sequences of length 1 are excluded as the final item cannot be used as a prediction target). All datasets use a 70/10/20 user split into train/validation/test, with validation and test capped at 10,000 users each (surplus assigned to training). Within each split, the final item in each user's sequence is the prediction target and the remainder forms the input. Ranking is performed over the full item catalog without negative sampling. Supervised baselines are repeated across up to 4 random seeds and averaged. ICR results are likewise averaged over 4 independent training runs. Statistical significance markers (*) in the supervised comparison table use a one-sided Welch's two-sample t-test ($\alpha = 0.05$, H_1: ICR mean $>$ baseline mean) applied to the respective seed distributions.

Table 7: Dataset statistics.

Dataset	Users	Items	Interactions	Avg. len.
Appliances	702	30,239	6,875	9.8
Arts	477,916	302,380	1,774,603	3.7
Games	100,955	71,911	733,447	7.3
Grocery	248,193	283,354	1,841,452	7.4
Office	186,762	306,617	1,290,349	6.9
Scientific	28,095	165,687	199,798	7.1
Software	4,144	21,639	26,766	6.5
Dianping	208,596	338,483	3,906,295	18.7
ML-20M	138,493	27,278	20,000,263	144.4
Washington	562,930	120,080	11,065,695	19.7
Yelp	371,242	150,346	4,728,381	12.7

Notes: Users and interactions filtered to the evaluation split; sequences of length 1 excluded. Items = full catalog size (all items eligible for ranking).

E.2 Baselines

Zero-shot methods. These require no per-dataset training:

- **RecPFN** (Tan et al., 2026): Prior-Fitted Network with LLM embeddings and full cross-attention. Evaluated with its original paper configuration (8 support sequences, $k = 2$ nearest neighbors).
- **SRPFN** (Kang et al., 2026): Prior-Fitted Network for sequential recommendation that designs a hierarchical stochastic block model prior and conditions predictions on low-rank SVD decompositions of user–item co-occurrence and item–item transition matrices. Unlike embedding-based methods, SRPFN derives its input representations directly from the target domain's interaction statistics without requiring auxiliary text or pre-trained embeddings. We use the publicly released checkpoint evaluated without any target-domain gradient updates.
- **RecFormer** (Li et al., 2023): Text-based sequential recommender pre-trained on item descriptions across domains.
- **RecGPT** (Jiang et al., 2025): Generative pre-training on aggregated user sequences for zero-shot transfer.
- **UniSRec** (Hou et al., 2022): Universal sequence representations via contrastive pre-training on item text. Evaluated at epoch 0 (pre-trained weights, no fine-tuning).
- **VQ-Rec** (Hou et al., 2023): Vector-quantized item representations for cross-domain transfer. Evaluated at epoch 0.
- **EmbKNN**: Non-parametric baseline that averages LLM item embeddings in the user's history and retrieves by cosine similarity. No learned parameters.
- **EmbKNN-R**: Same as EmbKNN but using random embeddings (matching ICR's input regime), isolating the contribution of in-context learning from the embedding choice.

Non-parametric methods. A third class, distinct from both zero-shot and supervised methods, uses only training-set interaction counts: no gradient training, no learned or pre-trained embeddings, and no neural forward pass. They establish how far simple frequency statistics alone carry next-item prediction. Full per-dataset results are in Appendix A.2 (Table 5).

- **Popular**: Ranks all items by their global frequency in the training interactions; the same top-k list is predicted for every query. We use a deterministic top-count ranking.
- **Co-occur (last)**: Ranks candidate items by how often they co-occur with the query's last item in the same training sequence (co-occurrence anywhere in the sequence, not restricted to adjacency).
- **Co-occur (all)**: Scores candidates by summed co-occurrence counts over every item in the query history (uniform weight per history item), excluding items already seen.
- **Successor**: Context-free first-order transition count. Ranks candidates by how often they *immediately follow* the query's last item across all training sequences. This is the strongest non-parametric baseline and the natural count-based analogue of the pointer head's in-context successor selection.
- **Context Succ.**: Applies the same successor vote as *Successor*, but restricts scoring to the support sequences retrieved for that query by the inverted-index context selector, rather than all training sequences. Holds retrieval fixed with ICR's own retrieval configuration, isolating the model's contribution to performance.

Co-occur, Successor, and Context Succ. sample proportional to counts and are averaged over 4 seeds; Popular is deterministic. EmbKNN and EmbKNN-R (above) are also non-parametric but are reported with the zero-shot methods since they operate in the embedding regime.

Supervised methods. Trained per-dataset with early stopping on validation MRR@10 and a ceiling of 50 epochs; each baseline is reported at its best validation epoch. Results are averaged across up to 4 random seeds. We do not perform per-dataset hyperparameter search, matching common practice for large multi-dataset benchmarks. The baselines are:

- **SASRec** (Kang & McAuley, 2018): Self-attentive sequential model with learned ID embeddings.
- **BERT4Rec** (Sun et al., 2019): Bidirectional encoder with masked item prediction.
- **GRU4Rec** (Hidasi et al., 2015): GRU-based session recommender.
- **FDSA** (Zhang et al., 2019): Feature-level deeper self-attention network.
- **UniSRec / VQ-Rec** (fine-tuned): Same architectures as above, fine-tuned per dataset under the same early-stopping protocol (up to 50 epochs).

LLM-embedding variants (^). For SASRec, BERT4Rec, and GRU4Rec, we additionally evaluate variants that replace learned ID embeddings with frozen Qwen-1B text embeddings as input. These provide a controlled comparison of the embedding regime: same architecture and training procedure, but with pre-trained semantic embeddings instead of learned IDs. This isolates whether access to LLM representations helps supervised models close the gap with ICR.

Implementations. SASRec, GRU4Rec, BERT4Rec, and FDSA are implemented through RecBole (Xu et al., 2023) with its recommended hyperparameters. All other baselines use their official repositories and default hyperparameters for fine-tuning (where applicable) and inference. SRPFN required three modifications: (i) a clean split evaluation mode supporting disjoint train/test users and full-catalog ranking (the original assumes leave-one-out with candidate sampling); (ii) fold-in of test-user embeddings via item CF vectors to avoid a second SVD; and (iii) lazy batch generation in the scoring pipeline, replacing upfront materialisation of all candidate batches with on-demand generation to support full-catalog ranking without exhausting memory.

E.3 Efficiency measurement conventions

This subsection documents how every cell of Table 2 is sourced.

Hardware. All wall-clock times are measured on a single NVIDIA A10 GPU (24 GB), so all timings reported in this paper are produced under identical hardware and software conditions.

Pretrain. For ICR and RecPFN, "pretrain" is the per-dataset preprocessing cost (building the inverted index used for ICL retrieval); neither model sees per-dataset training. For SRPFN, "pretrain" is the per-dataset co-occurrence matrix construction and factorization. For VQ-Rec and BERT4Rec, "pretrain" is preprocessing plus 50 epochs of per-dataset training.

Per-user inference. Per-user inference is the total inference wall-clock divided by the number of test users, measured at each method's default deployment configuration: ICR at batch size 64 with per-query context size 128, RecPFN at batch size 8 with per-query context size 8, VQ-Rec and BERT4Rec at batch size 1024, and SRPFN at batch size 64.

Breakeven users. The breakeven row reports $N^* = (P_b - P_o)/(I_o - I_b)$, where P_b, P_o are the baseline's and ICR's pretrain seconds and I_o, I_b are their per-user inference times in seconds. Below N^*, ICR's total cost ($P_o + N \cdot I_o$) is lower than the baseline's. We render "–" in two cases: (i) the baseline is at least as slow per user as ICR *and* pays at least as much in pretrain (so no N^* exists where the baseline overtakes ICR), and (ii) the baseline is cheaper from $N = 1$ (which does not occur for any of the methods in Table 2).

F Implementation Details

F.1 Multi-Head Attention and Feed-Forward Block

We adopt the standard multi-head attention (MHA) block of Vaswani et al. (2017), exposed in the body as $\mathrm{MHA}(X, Z, M)$ with query input X, key/value input Z, and additive mask M:

$$\mathrm{MHA}(X, Z, M) = \mathrm{LayerNorm}\big(\mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_H)W^O + X\big), \tag{8}$$

with $\mathrm{head}_m = \mathrm{softmax}\Big(\frac{Q_m K_m^\top}{\sqrt{d_m}} + M\Big) V_m$ and $Q_m = XW_m^Q$, $K_m = ZW_m^K$, $V_m = ZW_m^V$. Each MHA block is followed by a feed-forward network $\mathrm{FFN}(x) = W_2\,\mathrm{ReLU}(W_1 x)$ with a residual connection.

F.2 Training Details

Optimizer and learning rate schedule. We use AdamW (Loshchilov et al., 2017) with peak learning rate 2×10^{-4}, weight decay 10^{-4}, and $(\beta_1, \beta_2) = (0.9, 0.999)$. The schedule consists of 6 linear warmup epochs followed by cosine annealing down to a minimum of $0.01\times$ the peak LR over the remaining epochs. Training proceeds in two stages: Stage 1 trains on the linear SCM prior alone, and Stage 2 warm-starts from the Stage 1 checkpoint and trains on the full extended SCM prior, which adds routing, item-dependent decay, and pattern affinity (Appendix F.3). Each stage runs for up to 120 epochs. All forward and loss passes use bfloat16 mixed precision with gradient checkpointing to reduce memory.

Early stopping and best-model selection. Model selection is based on HR@10 evaluated on a synthetic validation set that is held fixed between checkpoints. At the start of training a validation set is sampled and the model's HR@10 on it becomes the initial bar to beat. Each subsequent epoch is evaluated on the same fixed set; if HR@10 improves, the checkpoint is saved, a fresh validation set is drawn, and the new checkpoint's HR@10 on that set becomes the new bar. If no improvement is seen for 20 consecutive epochs, training stops. Holding the validation set fixed between checkpoints ensures that epoch-to-epoch comparisons are made on identical environments, so measured improvements reflect genuine learning rather than sampling variance in the synthetic tasks.

Evaluation configuration. At inference, the model evaluates in batches of 64 query sequences with 128 support sequences retrieved per query (8192 support sequences per batch). Retrieval uses the inverted-index matcher with $k = 30$ recency layers and random padding enabled (Appendix H).

F.3 Synthetic Prior Hyperparameter Ranges

Table 8 reports the full parameter ranges for both training stages. These ranges define the prior over environments from which synthetic tasks are sampled during training.

Table 8: SCM hyperparameter ranges by training stage. Stage 1 uses only the linear SCM ($\sigma_R = \sigma_\gamma = \alpha = 0$). Stage 2 activates all extensions.

Parameter	**Stage 1**	**Stage 2**
Transition matrix		
k (out-degree)	3	$\{1, 3, 5, 7\}$
p_{self} (self-loop prob.)	0.2	$\{0.1, 0.2, 0.3\}$
Sequence generation		
W (window size)	$\{1, 2, 3\}$	$\{1, 2, 3\}$
γ_{base} (base decay)	$\{0.5, 0.8\}$	$\{0.5, 0.8\}$
β (popularity weight)	$\{0, 0.1, 0.2, 0.3, 0.5\}$	$\{0, 0.1, 0.2, 0.3, 0.5\}$
σ_{noise}	0.01	0.01
σ_{pop} (popularity skew)	4.0	4.0
Content-dependent extensions (Stage 2 only)		
σ_R (routing scale)	0	$\mathcal{U}[0, 2.0]$
σ_γ (decay spread)	0	$\mathcal{U}[0, 0.5]$
α (pattern strength)	0	$\mathcal{U}[0, 1.0]$
H (hash buckets)	—	$\{20, 50, 100, 200\}$
k (pattern order)	—	$\{2, 3\}$
m (nonzeros per bucket)	—	$\{3, 5, 7, 10, 15\}$
Sequence structure		
$L_{\max}$ (max length)	30	30
$L_{\min}$ (min length)	5	5
Repeat suppression	enabled	enabled

G Synthetic Prior: Full Specification

This appendix provides the complete generative specification of the structural causal model (SCM) used to sample synthetic environments for training. The structure mirrors Section 3: we first specify the linear SCM covering $\mathcal{F}_{\text{lin}}$ (Section G.1), then the extensions covering $\mathcal{F}_{\text{full}}$ (Section G.2), followed by the full generation procedure (Section G.3).

Let N denote the item catalog size. All matrices and vectors are indexed from 1 to N; index 0 is reserved for padding and carries zero probability mass throughout.

G.1 Linear SCM ($\mathcal{F}_{\text{lin}}$)

The linear SCM generates next-item distributions via Eq. 4. Its components are:

Transition matrix. $T \in \mathbb{R}^{N \times N}$ is constructed via a sparse random directed graph:

1. For each item i, sample k target items uniformly from $\{1, \ldots, N\}$ (with replacement). Set $T_{i,j} = 1$ for each sampled target j.
2. Disable self-loops: $T_{i,i} = -\infty$ for all i.
3. Re-enable self-loops stochastically: for each i, with probability p_{self}, set $T_{i,i} = 1$.

Each row has approximately k nonzero entries (plus potentially a self-loop).

Popularity distribution. Item popularities are sampled as:

$$b_j = \text{softmax}(z_1, \ldots, z_N)_j, \quad z_j \sim \mathcal{N}(0, \sigma_{\text{pop}}^2) \tag{9}$$

where σ_{pop} controls skewness (higher values produce more concentrated, Zipf-like distributions).

Sequence generation. Given a prefix $s_{\leq t}$, the next item is sampled from:

$$P(s_{t+1} = j \mid s_{\leq t}) = \sigma_j \left(\sum_{\tau=\max(1,\, t-W+1)}^{t} \gamma^{\,t-\tau} \cdot T_{s_\tau, j} \;+\; \beta \cdot b_j \;+\; \sigma_{\text{noise}} \cdot \epsilon_j \right) \tag{10}$$

where W is the window size, $\gamma \in (0, 1)$ a decay coefficient, β the popularity mixing weight, and $\epsilon_j \sim \mathcal{N}(0, 1)$ additive noise. The normalization σ is a clamp-normalize map: negative scores are clamped to zero, then normalized to a distribution. As noted in Appendix J.2, σ is monotone in the raw scores, so it preserves the item ranking induced by the projected-kernel score; the geometric weights $\gamma^{\,t-\tau}$ are the recency-biased instantiation of the general per-lag schedule $w_{t-\tau}$ in Eq. 4.

G.2 Extended SCM ($\mathcal{F}_{\text{full}}$)

The extended SCM augments the linear SCM with content-dependent mechanisms, producing next-item distributions via Eq. 5. Setting all extension parameters to zero recovers the linear SCM.

Routing matrix. $R \in \mathbb{R}^{N \times N}$ introduces content-dependent weighting:

$$R_{i,j} \sim \mathcal{N}(0, \sigma_R^2), \quad i, j \in \{1, \ldots, N\} \tag{11}$$

The effective weight on position τ containing item s_τ when predicting from current item s_t becomes $\gamma_{s_t}^{t-\tau} \cdot \exp(R_{s_\tau, s_t})$. When $\sigma_R = 0$, the routing term reduces to 1.

Item-dependent decay. Per-item decay rates replace the global coefficient:

$$\gamma_i = \text{clamp}(\gamma_{\text{base}} + \epsilon_i,\ 0.01,\ 1.0)\,, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_\gamma^2) \tag{12}$$

When $\sigma_\gamma = 0$, all items share the global decay γ_{base}.

Pattern affinity. An order-sensitive hash and sparse affinity matrix introduce content-dependent contributions:

- *Hash function.* For a k-gram $(s_{t-k+1}, \ldots, s_t)$: $f_t = \left(\sum_{i=1}^{k} s_{t-k+i} \cdot 31^{k-i} \right) \bmod H$, where H is the number of hash buckets.
- *Affinity matrix.* $a \in \mathbb{R}^{H \times N}$ is sparse: each row has exactly m nonzero entries at uniformly random column positions, with values drawn from $\mathcal{N}(0, 1)$.
- *Application.* At step t (if $t \geq k$), the pattern contribution to candidate j is $\alpha \cdot a_{f_t, j}$, where α controls overall strength.

G.3 Full Generation Procedure

Algorithm 1 describes the complete sequence generation procedure given a fully instantiated SCM (linear or extended).

For the linear SCM, the routing term $\exp(R_{s_\tau, s_t})$ reduces to 1, item-dependent decay reduces to γ_{base}, and the pattern contribution is absent. Repeat suppression is an optional post-normalization sampling policy, not part of the general SCM distribution in Theorem 3.

H Context Selection

At inference time, ICR requires a pool of support sequences from the target domain. We describe the retrieval procedure used to select these sequences given a query.

H.1 Inverted-Index Matcher

Given a training pool of P sequences $\mathcal{P} = \{s^{(1)}, \ldots, s^{(P)}\}$ from the target domain, we retrieve the N_c most relevant sequences for each test query $q = (q_1, \ldots, q_T)$.

Algorithm 1 SCM Sequence Generation

Require: Instantiated SCM parameters $(T, R, \gamma, a, b, \alpha, \beta, \sigma_{\text{noise}}, W, k)$; number of sequences M; sequence length L

Ensure: Generated sequences $\{(s_1, \ldots, s_L)^{(m)}\}_{m=1}^{M}$

for $m = 1, \ldots, M$ **do**
  Sample $s_1 \sim \text{Uniform}(\{1, \ldots, N\})$
  Sample $L_m \sim \text{Uniform}(\{L_{\min}, \ldots, L_{\max}\})$
  for $t = 1, \ldots, L_m - 1$ **do**
    // Transition scoring with routing and decay
    for $j = 1, \ldots, N$ **do**
      $r_j \leftarrow \sum_{\tau=\max(1,\, t-W+1)}^{t} \gamma_{s_t}^{t-\tau} \cdot \exp(R_{s_\tau, s_t}) \cdot T_{s_\tau, j}$
    end for
    if $t \geq k$ and $\alpha > 0$ **then**
      // Pattern contribution
      $f_t \leftarrow \text{hash}(s_{t-k+1}, \ldots, s_t) \bmod H$
      $r_j \leftarrow r_j + \alpha \cdot a_{f_t, j}$ for all j
    end if
    // Popularity mixing and noise
    $\tilde{r}_j \leftarrow (1 - \beta)\, r_j + \beta\, b_j + \sigma_{\text{noise}} \cdot \epsilon_j$ where $\epsilon_j \sim \mathcal{N}(0, 1)$
    // Clamp and normalize
    $\tilde{r}_j \leftarrow \max(\tilde{r}_j, 0)$ for all j
    $P(j) \leftarrow \tilde{r}_j \,/\, \sum_{j'} \tilde{r}_{j'}$
    // Optional repeat suppression
    Set $P(j) = 0$ for $j \in \{s_1, \ldots, s_t\}$; re-normalize
    Sample $s_{t+1} \sim \text{Categorical}(P)$
  end for
end for

Index construction. We build an inverted index $\mathcal{J} : \mathcal{I} \to 2^{[P]}$ mapping each item $v \in \mathcal{I}$ to the set of sequence indices containing v (excluding each sequence's final item, which serves as the prediction target):

$$\mathcal{J}(v) = \{i \mid v \in s^{(i)}_{1:T_i - 1}\} \tag{13}$$

Long sequences exceeding the maximum length $T_{\max}$ are split into overlapping chunks with stride $T_{\max}/2$, each treated as an independent sequence in the pool.

Candidate retrieval. Given a query q, we retrieve candidates using a layered strategy that prioritizes recent items. Starting from the most recent item q_T and proceeding backward through $q_{T-1}, q_{T-2}, \ldots$, we accumulate the union of posting lists $\mathcal{J}(q_{T-d})$ for $d = 0, 1, \ldots, k-1$, stopping once the number of distinct candidate sequences exceeds $\alpha \cdot N_c$ (with at least two priority items always processed) or once k layers have been examined. The parameter k controls the maximum depth into the query history, and α is the breadth multiplier ablated in Table 6; in our experiments $k = 30$ and $\alpha = 10$.

Similarity scoring. Candidates are ranked by weighted Jaccard similarity with exponential recency decay. Each item q_t at position t in the query receives weight $w_t = \exp(-\lambda(T - 1 - t))$, where λ is the decay rate. The weighted Jaccard similarity between query q and candidate sequence s is:

$$J_w(q, s) = \frac{\sum_{v \in q \cap s} w_v}{\sum_{v \in q} w_v - \sum_{v \in q \cap s} w_v + |s|} \tag{14}$$

where $q \cap s$ denotes the set of items appearing in both sequences (excluding target items), and w_v is the weight of item v from its position in q. This prioritizes sequences that share *recent* items with the query, reflecting the intuition that recent interactions are more predictive of the next item.

The top N_c candidates by J_w are returned as the support context. We use $\lambda = 1.0$ in all experiments.

Random padding. When fewer than N_c candidates have nonzero similarity (e.g., when the query contains items not seen in the training pool), the remaining slots are filled with randomly sampled sequences from $\mathcal{P}$. This ensures the model always receives a full pool of N_c support sequences, even for cold-start queries.

H.2 Computational Cost

The inverted index is built once per dataset in $O(P\,\bar{T})$ time, where $\bar{T}$ is the average sequence length. At query time, candidate retrieval cost is bounded by the breadth cap $\alpha \cdot N_c$, and scoring is vectorised and linear in the number of candidates. In practice, context selection adds negligible overhead compared to the forward pass.

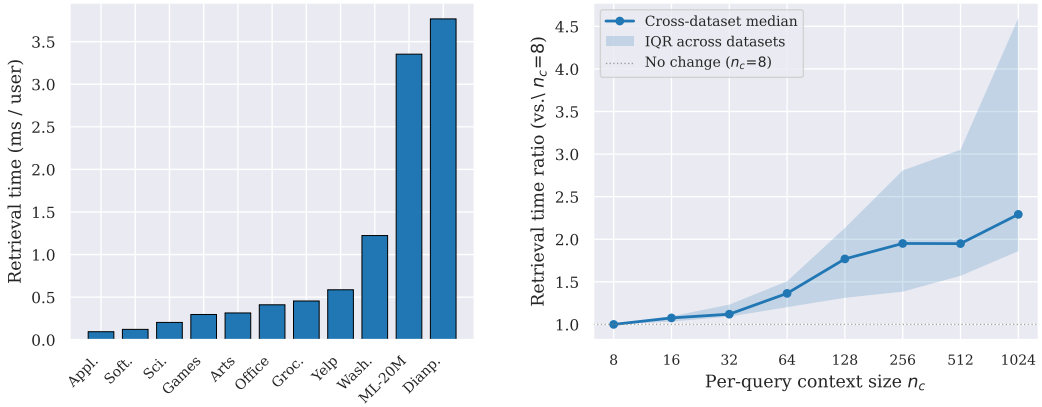


Figure 5: Inverted-index retrieval cost. **Left:** per-user retrieval time at the paper-default inference configuration. **Right:** retrieval time relative to $n_c = 8$ as a function of per-query context size, aggregated across all eleven datasets.

Figure 5 (left) reports per-user retrieval time across the eleven evaluation datasets at the paper-default configuration. Even on the slowest dataset (Dianping) retrieval takes under 4 ms per user, well below the per-user forward-pass cost. The dense catalogs (Dianping, ML-20M) are slower than the sparse Amazon ones because more posting-list mass passes through the breadth cap before it trips. Figure 5 (right) sweeps the per-query context size n_c at fixed total budget and normalises to the smallest $n_c = 8$; the cross-dataset median grows sub-linearly with n_c across the entire range, since the breadth cap is reached after the same number of priority items in expectation regardless of n_c.

I Identity-Restricted Cross-Attention: Implementation

Algorithm 2 details the gather-attend-scatter implementation of identity-restricted cross-attention. The key challenge is that each query position attends to a variable-length set of context positions (those sharing its item identity), which precludes a single batched attention call. We address this via frequency bucketing.

Frequency bucketing. Within each bucket, all query positions are padded to the maximum context count in that bucket. Sorting by frequency before partitioning ensures that positions with similar context counts are grouped together, minimising wasted computation on padding. With $B = 3$ buckets, items with few context occurrences (rare items appearing in few sequences) are separated from items with many occurrences (popular items appearing in many sequences), reducing the average padding overhead.

Optimised FFN. The output projection (W^O) and feed-forward network are applied only to positions that received a cross-attention update (i.e., positions in $\mathcal{S}$). Positions whose item does not

Algorithm 2 Identity-Restricted Cross-Attention

Require: Query representations $\mathbf{h}_{q,i} \in \mathbb{R}^{T\times d}$, flattened context $\bar{H}_c \in \mathbb{R}^{C\times d}$, query item IDs $\mathbf{s} \in \mathcal{I}^T$, context item IDs $\mathbf{c} \in \mathcal{I}^C$, number of buckets B
Ensure: Updated query representations
// Index construction
Compute unique context items and their positions: for each unique item j, store the set of context positions $\mathcal{P}_j = \{\tau : c_\tau = j\}$ and count $n_j = |\mathcal{P}_j|$
Identify shared positions: $\mathcal{S} = \{t : s_t \in \{c_1, \ldots, c_C\}\}$
// Frequency bucketing
Sort shared query positions by their item's context count n_{s_t}
Partition into B equal-sized buckets $\mathcal{B}_1, \ldots, \mathcal{B}_B$
// Bucketed attention
for bucket $b = 1, \ldots, B$ **do**
  $C_b \leftarrow \max_{t\in\mathcal{B}_b} n_{s_t}$ (max context count in bucket)
  Gather: For each $t \in \mathcal{B}_b$, retrieve context keys/values at positions $\mathcal{P}_{s_t}$, padded to length C_b
  Attend: Compute scaled dot-product attention with padding mask: each query attends to its n_{s_t} valid context positions
  Scatter: Write attention outputs back to positions $\mathcal{B}_b$
end for
// Output projection and FFN
Apply output projection and feed-forward network only to positions in $\mathcal{S}$ (positions not in $\mathcal{S}$ retain their input via identity residual)
return Updated query representations

appear in context skip both operations entirely, as their output would be overwritten by the identity residual regardless.

Complexity. A query position $t \in \mathcal{S}$ attends only to the n_{s_t} context positions sharing its item, so its attend cost is $O(n_{s_t})$; positions outside $\mathcal{S}$ are skipped entirely (Optimised FFN, above). The exact total attend cost is therefore $\sum_{t\in\mathcal{S}} n_{s_t}$. Writing $\bar{c} = C/U$ for the average context occurrences per unique item (U unique items in context), this total equals $|\mathcal{S}| \cdot \bar{c}$ when query items are distributed uniformly over the unique context items; under heavy-tailed popularity—query items drawn in proportion to their context frequency—it grows to $\frac{1}{C}\sum_j n_j^2 = \bar{c}\,(1+\mathrm{CV}^2)$, where CV is the coefficient of variation of the counts $\{n_j\}$. Either way the cost is $O(|\mathcal{S}| \cdot \bar{c})$ up to the dispersion factor, against $O(T \cdot C)$ for dense cross-attention, which scores all T query positions against all C context positions; IRCA saves on both factors, since $|\mathcal{S}| \le T$ and $\bar{c} \le C$. Bucketing adds a padding overhead bounded by the variance of context counts within each bucket, which decreases with more buckets at the cost of additional kernel launches.

J Projected Kernel Models: Instantiations and Proofs

This appendix provides two contributions. First, we show that many of the most widely used sequential recommenders—from classical methods (Markov chains, k-NN) to modern neural architectures (SASRec, GRU4Rec)—are instances of the projected kernel model framework (Definition 1, restated in Section J.1), demonstrating that it unifies a diverse range of landmark models and motivating engineering a synthetic prior around it. Second, we prove expressiveness of the generator's scoring family (Proposition 2) and full support of its general sequence prior over finite item sequences (Theorem 3).

J.1 Sequential Recommenders as Projected Kernel Models

Recall from Definition 1 that a sequential recommender is a *projected kernel model* if it scores candidates via

$$\mathrm{score}(j \mid s_{\le t}) = \phi(s_{\le t})^\top \psi(j) + b_j, \qquad \phi(s_{\le t}) = \sum_\tau w_\tau(s_{\le t}) \cdot g_\tau(s_{\le t}) \tag{15}$$

where ϕ encodes the history, ψ maps each candidate to the same space, and b_j is an item bias (Eq. 2). The defining requirement is that the history representation and the candidate enter the score through a bilinear (dot-product) interaction; models whose scoring head is instead a nonlinear function of the history–candidate pair fall outside the framework. Within it, we identify w_τ, g_τ, and ψ for each model below. The key distinction is between models in $\mathcal{F}_{\text{lin}}$ (where weights depend only on position: $w_\tau = w_{t-\tau}$) and models in $\mathcal{F}_{\text{full}}$ (where weights depend on the content of the sequence).

J.1.1 Models in $\mathcal{F}_{\text{lin}}$

These models aggregate past items with position-dependent weights that do not depend on which items appear at which positions. The linear SCM directly generates data from this class.

First-order Markov chain. The bigram model defines $P(j \mid s_t = i) \propto c(i,j)$ where $c(i,j)$ counts observed transitions:

$$\begin{aligned} g_\tau(s_{\le t}) &= \delta_{s_\tau} \in \mathbb{R}^N \quad \text{(one-hot for item at position } \tau) \\ w_\tau &= \mathbb{1}[\tau = t] \\ \psi(j) &= \mathbf{T}_{:,j} \in \mathbb{R}^N \quad \text{(column } j \text{ of row-normalized bigram matrix)}, \quad b_j = 0 \end{aligned}$$

This yields $\phi^\top \psi = T_{s_t,j}$: the simplest instance of $\mathcal{F}_{\text{lin}}$ with window size 1.

ItemKNN. Scores via average cosine similarity over history: $\text{score}(j) = \frac{1}{t}\sum_\tau \cos(\mathbf{r}_{s_\tau}, \mathbf{r}_j)$ where $\mathbf{r}_i$ is item i's co-occurrence vector:

$$\begin{aligned} g_\tau(s_{\le t}) &= \hat{\mathbf{r}}_{s_\tau} \quad (\ell_2\text{-normalized co-occurrence vector}) \\ w_\tau &= 1/t \\ \psi(j) &= \hat{\mathbf{r}}_j, \quad b_j = 0 \end{aligned}$$

Sequential variants use recency-weighted $w_\tau = \lambda^{t-\tau}$, mapping directly to the prior's exponential decay.

SKNN (Jannach & Ludewig, 2017). Session-based k-NN with position-weighted session vectors:

$$\begin{aligned} g_\tau(s_{\le t}) &= \mathbf{b}_{s_\tau} \in \{0,1\}^{|\mathcal{S}|} \quad \text{(session incidence vector)} \\ w_\tau &= \text{pos}(\tau)/|q| \quad \text{(position-weighted, linear recency)} \\ \psi(j) &= \mathbf{D}\,\mathbf{b}_j \quad \text{where } \mathbf{D} = \text{diag}(1/|s_1|, \ldots), \quad b_j = 0 \end{aligned}$$

The effective transition matrix $\mathbf{T} \propto \mathbf{B}^\top \mathbf{D}\,\mathbf{B}$ is a weighted Gram matrix.

FPMC (Rendle et al., 2010). Factorized personalized Markov chains score via $\hat{x}_{u,j,t} = \mathbf{p}_u^\top \mathbf{q}_j + \mathbf{v}_{s_t}^{L\top} \mathbf{v}_j^R$:

$$\begin{aligned} g_\tau(s_{\le t}) &= [\mathbf{p}_u;\ \mathbf{v}^L_{s_\tau}] \in \mathbb{R}^{d_{\text{MF}}+d_{\text{MC}}} \\ w_\tau &= \mathbb{1}[\tau = t] \\ \psi(j) &= [\mathbf{q}_j;\ \mathbf{v}^R_j], \quad b_j = 0 \end{aligned}$$

The sequential component $\mathbf{V}^L(\mathbf{V}^R)^\top$ has rank d_{MC}; this is $\mathcal{F}_{\text{lin}}$ with window size 1.

Fossil (He & McAuley, 2016). Extends FPMC with per-lag weights: $\hat{x}_{u,j} = \mathbf{p}_u^\top \mathbf{q}_j + (\sum_\ell \eta_u^{(\ell)} \mathbf{v}^L_{s_{t-\ell+1}})^\top \mathbf{v}_j^R$:

$$\begin{aligned} g_\tau(s_{\le t}) &= [\mathbf{p}_u;\ \mathbf{v}^L_{s_\tau}] \\ w_\tau &= \eta_u^{(t-\tau)} \quad \text{(per-user, per-lag)} \\ \psi(j) &= [\mathbf{q}_j;\ \mathbf{v}^R_j], \quad b_j = 0 \end{aligned}$$

The weights $\eta_u^{(\ell)}$ are per-user and per-lag but *content-independent*: they do not depend on which item occupies position ℓ. This is $\mathcal{F}_{\text{lin}}$ with arbitrary (non-exponential) position-dependent weights.

J.1.2 MODELS IN $\mathcal{F}_{\text{full}}$

These models have content-dependent aggregation weights: how much attention is paid to position τ depends on what item is there and/or what the current item is. This is the structural gap that separates simple aggregation from learned attention mechanisms.

SASRec (Kang & McAuley, 2018). A single-head, single-layer causal transformer computes:

$$\text{score}(j) = \left(\sum_{\tau=1}^{t} \alpha_{t\tau} \cdot \mathbf{e}_{s_\tau} W_V W_O + \mathbf{e}_{s_t} \right)^\top \mathbf{e}_j \tag{16}$$

where the attention weight decomposes as:

$$\alpha_{t\tau} \propto \exp\left(\frac{1}{\sqrt{d_k}} \left[\underbrace{\mathbf{p}_t^\top W_Q^\top W_K \mathbf{p}_\tau}_{\text{pos-pos}} + \underbrace{\mathbf{e}_{s_t}^\top W_Q^\top W_K \mathbf{e}_{s_\tau}}_{\text{content-content}} + \underbrace{\text{cross terms}}_{\text{content-pos}} \right] \right) \tag{17}$$

In the notation of Eq. 2:

$$\begin{aligned} w_\tau(s_{\le t}) &= \alpha_{t\tau} \quad \text{(content-dependent via } \mathbf{e}_{s_t}^\top W_Q^\top W_K \mathbf{e}_{s_\tau}\text{)} \\ g_\tau(s_{\le t}) &= \mathbf{e}_{s_\tau} W_V W_O + \mathbb{1}[\tau = t] \cdot \mathbf{e}_{s_t} \\ \psi(j) &= \mathbf{e}_j, \quad b_j = 0 \end{aligned}$$

The content-content term makes w_τ depend on the identity of both s_t and s_τ, placing SASRec in $\mathcal{F}_{\text{full}} \setminus \mathcal{F}_{\text{lin}}$. If the content-dependent terms are zeroed out, the model reduces to $\mathcal{F}_{\text{lin}}$ with position-only weights.

GRU4Rec (Hidasi et al., 2015). The hidden state evolves as $\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t$ with update gate $\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{e}_{s_t} + \mathbf{U}_z \mathbf{h}_{t-1})$. Scoring is $\text{score}(j) = \mathbf{h}_t^\top \mathbf{e}_j$. Unrolling the recurrence:

$$\begin{aligned} w_\tau(s_{\le t}) &= \prod_{k=\tau+1}^{t} (1 - \mathbf{z}_k) \cdot \mathbf{z}_\tau \quad \text{(product of forget gates)} \\ g_\tau(s_{\le t}) &= \tilde{\mathbf{h}}_\tau = \tanh(\mathbf{W} \mathbf{e}_{s_\tau} + \mathbf{U}(\mathbf{r}_\tau \odot \mathbf{h}_{\tau-1})) \\ \psi(j) &= \mathbf{e}_j, \quad b_j = 0 \end{aligned}$$

When the gate is approximately constant ($\mathbf{z}_t \approx \bar{z}$), this reduces to an exponential average:

$$\mathbf{h}_t \approx \sum_{\tau=1}^{t} \bar{z}(1 - \bar{z})^{t-\tau} \, \tilde{\mathbf{h}}_\tau \tag{18}$$

which is $\mathcal{F}_{\text{lin}}$ with $\gamma = 1 - \bar{z}$. The full model is in $\mathcal{F}_{\text{full}} \setminus \mathcal{F}_{\text{lin}}$ because: (i) $\mathbf{W}_z \mathbf{e}_{s_t}$ produces item-dependent gating (different items cause different retention rates), and (ii) $\mathbf{U}_z \mathbf{h}_{t-1}$ makes gating depend on accumulated history content.

BERT4Rec (Sun et al., 2019). Structurally identical to SASRec at inference time: $\phi(s_{\le t}) = f_{\text{BERT}}(s_1, \ldots, s_t, [\text{MASK}])$, scoring via $\text{softmax}(\phi^\top \mathbf{e}_j)$. The same decomposition as SASRec applies ($w_\tau = \alpha_{t\tau}$, $g_\tau = \mathbf{e}_{s_\tau} W_V W_O$, $\psi(j) = \mathbf{e}_j$); content-dependent attention weights place it in $\mathcal{F}_{\text{full}} \setminus \mathcal{F}_{\text{lin}}$.

NARM (Li et al., 2017). Combines GRU with attention: $\phi(s_{\le t}) = [\mathbf{h}_t;\ \sum_\tau \bar{\alpha}_\tau \mathbf{h}_\tau]$ where $\bar{\alpha}_\tau = \text{softmax}_\tau(\mathbf{v}^\top \sigma(\mathbf{A}_1 \mathbf{h}_\tau + \mathbf{A}_2 \mathbf{h}_t))$. In the notation of Eq. 2:

$$\begin{aligned} w_\tau(s_{\le t}) &= \bar{\alpha}_\tau \quad \text{(content-dependent via } \mathbf{A}_2 \mathbf{h}_t\text{)} \\ g_\tau(s_{\le t}) &= [(\textstyle\prod_{k=\tau+1}^{t} (1 - \mathbf{z}_k) \cdot \mathbf{z}_\tau) \, \tilde{\mathbf{h}}_\tau;\ \mathbf{h}_\tau] \\ \psi(j) &= [\mathbf{e}_j;\ \mathbf{e}_j], \quad b_j = 0 \end{aligned}$$

The attention weights depend on $\mathbf{h}_t$ (which encodes the full history content), placing NARM in $\mathcal{F}_{\text{full}} \setminus \mathcal{F}_{\text{lin}}$.

STAMP (Liu et al., 2018). $\phi(s_{\le t}) = \sigma(\mathbf{W}_s \sum_\tau \bar{\alpha}_\tau \mathbf{e}_{s_\tau}) + \sigma(\mathbf{W}_t \mathbf{e}_{s_t})$ where $\bar{\alpha}_\tau$ depends on both $\mathbf{e}_{s_\tau}$ and $\mathbf{e}_{s_t}$. In the notation of Eq. 2:

$$\begin{aligned} w_\tau(s_{\le t}) &= \bar{\alpha}_\tau \quad \text{(content-dependent via } \mathbf{e}_{s_t}\text{)} \\ g_\tau(s_{\le t}) &= \mathbf{W}_s \mathbf{e}_{s_\tau} + \mathbb{1}[\tau = t] \cdot \mathbf{W}_t \mathbf{e}_{s_t} \\ \psi(j) &= \mathbf{e}_j, \quad b_j = 0 \end{aligned}$$

Content-dependent attention weights place STAMP in $\mathcal{F}_{\text{full}} \setminus \mathcal{F}_{\text{lin}}$; the simplified variant STMP (uniform $\bar{\alpha}_\tau$) is in $\mathcal{F}_{\text{lin}}$.

J.2 Proof of Proposition 2

This proposition establishes that the generator's scoring function can reproduce *any* model in the linear class $\mathcal{F}_{\text{lin}}$ exactly. The intuition is straightforward: a model in $\mathcal{F}_{\text{lin}}$ scores items as a lag-weighted sum of projected transition scores (Eq. 3), which is precisely the form of the generator's raw score (Eq. 4) once we treat the transition matrix T and the per-lag weights w as free parameters.

Proposition (Restated: Proposition 2)**.** *For any linear projected kernel model with N items, there exists a parameterization (T, w, W, b) under which the generator's scoring function—and hence its induced item ranking—is identical.*

Proof. Let $f \in \mathcal{F}_{\text{lin}}$ be an arbitrary linear projected kernel model with scoring function:

$$\text{score}_f(j \mid s_{\le t}) = \left(\sum_{\tau=1}^{t} w^f_{t-\tau} \cdot M \cdot \mathbf{e}_{s_\tau} \right)^\top \psi(j) + b^f_j = \sum_{\tau=1}^{t} w^f_{t-\tau} \cdot \mathbf{e}_{s_\tau}^\top M^\top \psi(j) + b^f_j \tag{19}$$

The generator's raw scoring function (Eq. 4, before normalization) is:

$$\text{score}_{\text{prior}}(j \mid s_{\le t}) = \sum_{\tau=\max(1,t-W+1)}^{t} w_{t-\tau} \cdot T_{s_\tau, j} + b_j \tag{20}$$

Set $W = t$ (full history window), and choose parameters:

$$T_{ij} = \mathbf{e}_i^\top M^\top \psi(j), \qquad w_{t-\tau} = w^f_{t-\tau}, \qquad b_j = b^f_j. \tag{21}$$

Substituting into Eq. (20) gives $\text{score}_{\text{prior}}(j \mid s_{\le t}) = \sum_{\tau=1}^{t} w^f_{t-\tau} \cdot \mathbf{e}_{s_\tau}^\top M^\top \psi(j) + b^f_j = \text{score}_f(j \mid s_{\le t})$ for every candidate j and every prefix $s_{\le t}$. The two scoring functions coincide exactly. Since ranking over candidates is determined by the raw scores through a monotone normalization (which preserves order), the two models induce the same ranking, and hence identical HR@k and MRR@k. □

Implementation note (output map). $\mathcal{F}_{\text{lin}}$ maps scores to probabilities via $\mathrm{softmax}$ (Def. 1), whereas our generator uses a clamp-normalize map σ (clamp negatives to zero, then ℓ_2-normalize; Appendix G.1) for computational convenience. Both are strictly monotone in the raw scores, so they induce the *same* item ranking from identical scores. Because the model is trained and evaluated on rank-based objectives (HR@k, MRR@k), the coverage claim above is stated at the level of scores/ranking and is unaffected by the choice of normalization; matching the exact softmax distribution would additionally require $\sigma = \mathrm{softmax}$, which we do not rely on.

J.3 Proof of Theorem 3

The theorem formalizes the role of the general prior as an insurance mechanism: every finite item sequence has positive probability, while the prior's structure and hyperparameter distributions determine where most of its mass lies.

Theorem (Restated: Theorem 3)**.** *Let $P^{(L)}_{\text{prior}}$ be the distribution induced directly by the general SCM in Eq.* (5) *for a fixed finite length L, with $\sigma_{\text{noise}} > 0$. For every finite catalog $\mathcal{I}$ and every sequence $s_{1:L} \in \mathcal{I}^L$, $P^{(L)}_{\text{prior}}(s_{1:L}) > 0$.*

Proof. Let $\mathcal{E}$ be the event that every self-loop is enabled in the sampled transition graph. Since each of the finitely many self-loops is enabled independently with probability $p_{\text{self}} > 0$, $P(\mathcal{E}) = p_{\text{self}}^N > 0$. Conditional on $\mathcal{E}$, every transition score used by the generator is finite.

Fix any SCM realization $h \in \mathcal{E}$. The first item is sampled uniformly, so $P(s_1 \mid h) = 1/N > 0$. Now consider a prefix $s_{1:t}$ and desired next item $s_{t+1} = j$. Write the finite, noise-free part of candidate j's score in Eq. (5) as $r_j(s_{1:t}; h)$. The generator adds $\sigma_{\text{noise}}\epsilon_j$, where $\epsilon_j \sim \mathcal{N}(0, 1)$ and $\sigma_{\text{noise}} > 0$. Because the Gaussian distribution has full support on $\mathbb{R}$, the event

$$\epsilon_j > \frac{1 - r_j(s_{1:t}; h)}{\sigma_{\text{noise}}} \tag{22}$$

has strictly positive probability. On this event, candidate j has a positive score after clamping, the normalization denominator is positive, and hence its categorical sampling probability is strictly positive. Integrating over the fresh noise draw therefore gives

$$P(s_{t+1} = j \mid s_{1:t}, h) > 0. \tag{23}$$

Applying the chain rule conditional on h,

$$P(s_{1:L} \mid h) = P(s_1 \mid h) \prod_{t=1}^{L-1} P(s_{t+1} \mid s_{1:t}, h) > 0, \tag{24}$$

because it is a finite product of strictly positive terms. Finally,

$$P_{\text{prior}}^{(L)}(s_{1:L}) \geq P(\mathcal{E})\, \mathbb{E}[P(s_{1:L} \mid h) \mid \mathcal{E}] > 0. \tag{25}$$

□

Remark (support vs. concentration). Full support over finite sequences does not imply that the prior is uniform or that it closely matches every possible sequence distribution. The structured transition scores and the distributions placed over sparsity, popularity, recency, routing, and pattern parameters concentrate probability mass on dynamics intended to resemble real recommendation environments. The theorem supplies the complementary insurance statement that the general sequence prior does not exclude any finite sequential behavior.

K Proof: Absolute Coordinate Dependence Is Not Justified

We restate the theorem for convenience and then provide the full proof.

Theorem (Restated: Theorem 4)**.** *Let f be any foundational ICL recommender that uses a single fixed set of weights across all environments, and assume that the environment distribution is invariant to the embedding coordinate system (Eq. 27). Then there exists a rotation-invariant model $\bar{f}$ with $\mathcal{L}(\bar{f}; \phi) \leq \mathcal{L}(f; \phi)$.*

The claim is conditional on Eq. 27 and on A1–A2, and concerns the loss-minimising *foundational* model. It does not assert that embedding geometry is useless: rotation-invariant models may exploit the complete Gram matrix, including semantic similarity. Its force is that access to in-domain context does not make absolute coordinate structure systematically useful across environments, motivating random embeddings as a stronger practical regularizer.

The argument proceeds by constructing, for any model that depends on absolute embedding coordinates, a rotation-invariant model that achieves equal or lower loss. We first establish the formal framework, then prove the result via symmetrization over the orthogonal group.

K.1 Setup and Notation

Let $\mathcal{I} = \{1, \ldots, N\}$ be a finite item catalog and $\mathcal{I}^* = \bigcup_{t=0}^{\infty} \mathcal{I}^t$ the set of all finite sequences over $\mathcal{I}$.

Definition 5 (Environment)**.** *An environment h is a collection of conditional distributions $h = \{P_h(\cdot \mid s) : s \in \mathcal{I}^*\}$, where each $P_h(\cdot \mid s) \in \Delta(\mathcal{I})$ specifies the probability of each next item given a history $s = (s_1, \ldots, s_t)$. An environment fully characterises the sequential dynamics of a single recommendation domain.*

Definition 6 (Environment distribution)**.** *Let $\mathcal{H}$ denote the set of all environments. The environment distribution μ is a probability measure over $\mathcal{H}$ representing the space of domains that a foundational model must handle. A foundational model is evaluated in expectation over environments drawn from μ.*

Definition 7 (Context)**.** *A support context $\mathcal{D} = (s^{(1)}, \ldots, s^{(N_c)})$ consists of N_c support sequences from the target environment, each generated by sampling sequentially from P_h:*

$$s^{(m)} = (s_1^{(m)}, \ldots, s_{L_m}^{(m)}), \quad s_{t+1}^{(m)} \sim P_h(\cdot \mid s_{1:t}^{(m)})$$

The support context provides the in-domain signal that an ICL model uses to adapt to a specific environment at inference time.

Definition 8 (Embedding)**.** *An embedding is an injective function $\phi : \mathcal{I} \to \mathbb{R}^d$, mapping each item to a d-dimensional vector. We extend ϕ elementwise to sequences: $\phi(s_1, \ldots, s_t) = (\phi(s_1), \ldots, \phi(s_t))$, and to support contexts by applying to each sequence: $\phi(\mathcal{D}) = (\phi(s^{(1)}), \ldots, \phi(s^{(N_c)}))$.*

Definition 9 (Foundational ICL Recommender)**.** *A foundational ICL recommender is a measurable function f that takes an embedded support context $\phi(\mathcal{D})$ and an embedded query $\phi(s_{1:t})$ and produces a distribution over items. The predicted probability for item j is:*

$$\hat{P}_f(j \mid \mathcal{D}, s_{1:t}; \phi) = f(\phi(\mathcal{D}), \phi(s_{1:t}))(j) \tag{26}$$

The model satisfies two axioms:

- ***A1 (Foundational.)*** *f is a single fixed function applied across all environments—it does not depend on h.*
- ***A2 (In-context learning.)*** *The only information about h available to f is contained in its inputs $(\phi(\mathcal{D}), \phi(s_{1:t}))$.*

Together, A1 and A2 define the class of models that must generalise across environments using only context-provided signal, without per-environment weight updates.

K.2 Core Assumption

The proof relies on a single assumption about the relationship between the embedding and the environment distribution. We first define what embedding rotation invariance means, then state the assumption.

Definition 10 (Embedding rotation invariance)**.** *Let $O(d) = \{Q \in \mathbb{R}^{d \times d} : Q^\top Q = QQ^\top = I\}$ denote the orthogonal group. For any $Q \in O(d)$, define the rotated embedding $Q\phi$ by $(Q\phi)(i) = Q\phi(i)$ for all $i \in \mathcal{I}$.*

The environment distribution μ satisfies embedding rotation invariance *if the expected loss of any foundational model (Definition 9) is unchanged when all item embeddings are simultaneously transformed by any orthogonal matrix:*

$$\mathcal{L}(f; Q\phi) = \mathcal{L}(f; \phi) \quad \text{for all } Q \in O(d) \text{ and all } f \text{ satisfying A1–A2} \tag{27}$$

The quantifier ranges over all such f deliberately: Eq. 27 states that no *foundational observer can distinguish ϕ from $Q\phi$ in expectation over μ, which is precisely what it means for the environment distribution to privilege no coordinate system. It is a property of μ, not a restriction on the model.*

Assumption. *The environment distribution μ satisfies embedding rotation invariance (Eq. 27).*

Interpretation. This assumption says that no direction in embedding space is privileged across the distribution of environments. If a particular coordinate pattern in the embedding (e.g., item i having a large first coordinate) is informative about transitions in some environment, there is an equally likely environment where the same behavioral structure is associated with a different coordinate pattern. The coordinate system of the embedding is arbitrary; only relative geometric relationships (inner products between embedding vectors) carry potential information.

When does this hold? The assumption is satisfied whenever the environment distribution μ is defined independently of the embedding *coordinate system*. This is the natural setting for a foundational model: the environments (recommendation domains) exist independently of whatever embedding function we choose to represent items. The dynamics of a music streaming service do not depend on the coordinate system used to embed song titles.

What rotation invariance does and does not exclude. It is tempting to think the assumption fails for informative embeddings such as pre-trained LLM embeddings, where semantically similar items are geometrically close and this closeness correlates with transition dynamics. It does not. Because $Q \in O(d)$ preserves all inner products, $(Q\phi(i))^\top (Q\phi(j)) = \phi(i)^\top \phi(j)$, a rotation leaves the entire Gram matrix—hence all cosine similarities, Euclidean distances, and norms—unchanged. Consequently, *any dynamics driven by semantic similarity between items automatically satisfies Eq. 27*: LLM embeddings encode similarity through inner products, and that structure is rotation-invariant. Informative embeddings are therefore not an exception to the assumption; they are the central case it covers.

The assumption is violated *only* when the dynamics depend on structure that is *not* a function of the Gram matrix. Two such cases exist: (i) dependence on absolute coordinate semantics, e.g. a fixed axis meaning "price" with transitions keyed to that coordinate as a coordinate rather than via item-to-item similarity; and (ii) a fixed external anchor $u \in \mathbb{R}^d$ that is *not* rotated with ϕ, so that $u^\top \phi(i)$ changes under Q (the proof rotates $\phi(\mathcal{D})$ and $\phi(s_{1:t})$ jointly by the same Q). Our empirical observation that EmbKNN-R outperforms foundational baselines built on pre-trained embeddings (Table 3) supports using a stronger geometry-free representation in practice.

K.3 Key Definitions

Definition 11 (Rotation-invariant model)**.** *A model f is* rotation-invariant *if its predictions are unchanged under orthogonal transformations of the embedding. That is, for all $Q \in O(d)$, all $(\mathcal{D}, s_{1:t})$, and all $j \in \mathcal{I}$:*

$$\hat{P}_f(j \mid \mathcal{D}, s_{1:t}; Q\phi) = \hat{P}_f(j \mid \mathcal{D}, s_{1:t}; \phi) \tag{28}$$

Interpretation. A rotation-invariant model depends on the embeddings only through quantities that are preserved under orthogonal transformations—namely, pairwise inner products $\phi(i)^\top \phi(j)$. It does not extract information from absolute coordinate values or directions in embedding space. Its predictions are determined entirely by the relational structure among items (which items are similar, which co-occur in context) rather than by the specific vectors assigned to them.

Definition 12 (Absolute-coordinate-dependent model)**.** *A model f* depends on absolute coordinates *if it is not rotation-invariant: there exist $Q \in O(d)$, an input $(\mathcal{D}, s_{1:t})$, and an item j such that:*

$$\hat{P}_f(j \mid \mathcal{D}, s_{1:t}; Q\phi) \neq \hat{P}_f(j \mid \mathcal{D}, s_{1:t}; \phi) \tag{29}$$

K.4 Loss

The expected loss of a model f with embedding ϕ under the environment distribution μ is:

$$\mathcal{L}(f; \phi) = \mathbb{E}_{h \sim \mu}\, \mathbb{E}_{\mathcal{D} \sim P_h^{N_c}}\, \mathbb{E}_{(s_{1:t}, j) \sim P_h} \left[-\log \hat{P}_f(j \mid \mathcal{D}, s_{1:t}; \phi) \right] \tag{30}$$

This is the standard cross-entropy loss averaged over environments, contexts drawn from each environment, and prediction targets.

K.5 Proof via Symmetrization

The proof strategy is constructive: given any model f, we build a rotation-invariant model $\bar{f}$ by averaging f's predictions over all orthogonal transformations. We then show that this averaging cannot increase the loss.

Definition 13 (Symmetrized model)**.** *For any model f and embedding ϕ, define the symmetrized model $\bar{f}$ by averaging over the orthogonal group:*

$$\hat{P}_{\bar{f}}(j \mid \mathcal{D}, s_{1:t}; \phi) = \int_{O(d)} \hat{P}_f(j \mid \mathcal{D}, s_{1:t}; Q\phi)\, d\mu_{\mathrm{Haar}}(Q) \tag{31}$$

where μ_{Haar} is the normalized Haar measure on $O(d)$—the unique probability measure that is invariant under left and right multiplication by group elements.

Well-definedness. Since $O(d)$ is a compact group, the Haar measure is finite. For each Q and each j, $\hat{P}_f(j \mid \cdot; Q\phi) \in [0,1]$. The integral is therefore well-defined. Moreover, $\sum_j \hat{P}_f(j \mid \cdot; Q\phi) = 1$ for every Q, so integrating and exchanging sum with integral (by dominated convergence) gives $\sum_j \hat{P}_{\bar{f}}(j \mid \cdot; \phi) = 1$. Thus $\hat{P}_{\bar{f}}$ is a valid probability distribution over $\mathcal{I}$.

Proposition 14 (Symmetrization preserves or improves loss)**.** *Let μ satisfy Eq. 27. For any model f satisfying A1–A2:*

$$\mathcal{L}(\bar{f}; \phi) \leq \mathcal{L}(f; \phi)$$

Proof. Expanding the loss of $\bar{f}$ and substituting Eq. 31:

$$\mathcal{L}(\bar{f}; \phi) = \mathbb{E}_{h,\mathcal{D},s,j}\left[-\log \int_{O(d)} \hat{P}_f(j \mid \mathcal{D}, s; Q\phi)\, d\mu_{\text{Haar}}(Q)\right] \tag{32}$$

By Jensen's inequality (since $-\log$ is convex and μ_{Haar} is a probability measure):

$$\leq \int_{O(d)} \mathbb{E}_{h,\mathcal{D},s,j}\left[-\log \hat{P}_f(j \mid \mathcal{D}, s; Q\phi)\right] d\mu_{\text{Haar}}(Q) \tag{33}$$

For each fixed $Q \in O(d)$, the inner expectation is $\mathcal{L}(f; Q\phi)$. By the rotation invariance assumption (Eq. 27):

$$\mathcal{L}(f; Q\phi) = \mathcal{L}(f; \phi) \quad \text{for all } Q \in O(d)$$

Substituting:

$$\mathcal{L}(\bar{f}; \phi) \leq \int_{O(d)} \mathcal{L}(f; \phi)\, d\mu_{\text{Haar}}(Q) = \mathcal{L}(f; \phi) \quad \square$$

Proposition 15 ($\bar{f}$ is rotation-invariant)**.** *The symmetrized model $\bar{f}$ defined by Eq. 31 satisfies Eq. 28.*

Proof. For any $R \in O(d)$:

$$\hat{P}_{\bar{f}}(j \mid \mathcal{D}, s; R\phi) = \int_{O(d)} \hat{P}_f(j \mid \mathcal{D}, s; QR\phi)\, d\mu_{\text{Haar}}(Q) \tag{34}$$

Substituting $Q' = QR$ (since $R \in O(d)$, this is a bijection on $O(d)$, and the Haar measure is invariant under right multiplication):

$$= \int_{O(d)} \hat{P}_f(j \mid \mathcal{D}, s; Q'\phi)\, d\mu_{\text{Haar}}(Q') = \hat{P}_{\bar{f}}(j \mid \mathcal{D}, s; \phi) \quad \square$$

Theorem (Restated: Theorem 4)**.** *Let f be any foundational ICL recommender satisfying A1–A2, and let the environment distribution satisfy embedding rotation invariance (Eq. 27). Then there exists a rotation-invariant model $\bar{f}$ such that:*

$$\mathcal{L}(\bar{f}; \phi) \leq \mathcal{L}(f; \phi)$$

with equality if f is already rotation-invariant.

Proof. Existence and inequality: Propositions 14 and 15.

Equality ⇒ rotation-invariant. If f satisfies Eq. 28, then the integrand in Eq. 31 is constant in Q:

$$\hat{P}_{\bar{f}}(j \mid \mathcal{D}, s; \phi) = \int_{O(d)} \hat{P}_f(j \mid \mathcal{D}, s; \phi)\, d\mu_{\text{Haar}}(Q) = \hat{P}_f(j \mid \mathcal{D}, s; \phi)$$

So $\bar{f} = f$ and the losses are equal.

$\square$

K.6 IMPLICATIONS

Rotation-invariant models use only pairwise inner products. A rotation-invariant model satisfying Eq. 28 depends on embeddings only through the Gram matrix $G_{ij} = \phi(i)^\top \phi(j)$. This is because the Gram matrix is the complete invariant of a point configuration under orthogonal transformations: two embeddings ϕ and ϕ' satisfy $\phi' = Q\phi$ for some $Q \in O(d)$ if and only if they share the same Gram matrix.

Why random embeddings are preferred in practice. The theorem establishes that some optimal foundational ICL recommender is rotation-invariant and may therefore retain the full Gram matrix. Practical transformers, however, are not rotation-invariant and can learn persistent coordinate shortcuts that the assumption says do not generalize across environments.

Freshly resampled random embeddings $\phi(i) \sim \mathcal{N}(0, d^{-1}I)$ are a stronger, empirically motivated regularizer: they remove both persistent coordinates and semantic geometry from the backbone, forcing adaptation through relational structure in the context (co-occurrences, transitions, ordering). This is a design choice motivated by, rather than required by, the theorem.

Semantic similarity is preserved externally. LLM-embedding similarity remains available through embedding-space expansion (Section 6.4), which retrieves out-of-context items around pointer predictions. This separates context-derived collaborative inference in the backbone from an optional semantic retrieval stage.

Why supervised models are not affected. A supervised model violates A1: it is optimised for a specific environment h^*, not evaluated in expectation over μ. Assumption 27 does not apply because the evaluation distribution is P_{h^*}—a single environment that is not rotation-invariant. Through gradient-based optimisation on h^*-specific data, a supervised model can discover environment-specific correlations between embedding coordinates and transitions. The symmetrisation $\bar{f}$ would *hurt* such a model by averaging out precisely the coordinate-level information that training has extracted.