Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

SMC-Rec: Uncertainty-Adaptive Semantic Memory Compression for User Modeling in Sequential Recommendation

Ruijia Guo

guoruijia@hrbeu.edu.cn

Harbin Engineering University

Zhiyuan Chen

Harbin Engineering University

Xingyue Xie

Shenzhen Poweroak Newener Co., Ltd

Hao Bai

Shenzhen Poweroak Newener Co., Ltd

Research Article

Keywords: Sequential recommendation, user modeling, semantic memory compression, spectral calibration, prediction entropy, uncertainty-adaptive pyramid convolution

Posted Date: November 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11729778/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

SMC-Rec: Uncertainty-Adaptive Semantic Memory Compression for User Modeling in Sequential Recommendation

Ruijia Guo[a], guoruijia@hrbeu.edu.cn
Zhiyuan Chen[a], Corresponding Author, chenzhiyuan@hrbeu.edu.cn
Xingyue Xie[b], xiexy@poweroak.net
Hao Bai[b], baihao@poweroak.net
[a] Harbin Engineering University, China
[b] Institute of Renewable Energy, Shenzhen Poweroak Newener Co., Ltd., Shenzhen 518116, China

Abstract

Sequential recommendation must represent semantically rich interactions while remaining practical at inference, and it is fundamentally a user-modeling problem of deciding which interaction evidence to retain, compress, and forget. Existing ID-only models underuse item attributes and context, pretrained text embeddings may be anisotropic, and fixed sequence compression cannot adapt to uncertain user states. We propose SMC-Rec, a user-adaptive framework built on uncertainty-adaptive pyramid convolution and semantic memory compression. Each interaction combines item identity, attributes, context, time interval, and frozen Sentence-BERT semantics into an interaction-level user state. An offline singular value decomposition (SVD)-based spectral calibration controls semantic-space anisotropy, while the uncertainty-adaptive pyramid convolution encoder aggregates interaction memories at multiple temporal scales. Inside the encoder, prediction entropy regulates compression strength, and the short-, mid-, and long-term pyramid states provide future supervision, disagreement-based hard negatives, and consensus regularization during training. The convolutional backbone scales linearly with history length, whereas exact entropy estimation adds catalog-dependent scoring passes analyzed in the complexity analysis. Under full-ranking evaluation on Amazon Beauty, Sports, Toys, and MovieLens-1M, SMC-Rec improves NDCG@20 by 4.8%, 3.4%, 7.8%, and 2.7% over the best compared method on each dataset. Seed-level tests, ablations, robustness analyses, sparse/cold-start evaluations, and wall-clock measurements support the design and quantify its cost. Gains are most pronounced for short-history users and rare items. Implementation code, configuration files, and reproducibility materials are available at https://github.com/CLTCGUO/SMC-Rec.

Keywords: Sequential recommendation; user modeling; semantic memory compression; spectral calibration; prediction entropy; uncertainty-adaptive pyramid convolution

1. Introduction

The central objective of recommender systems is shifting from predicting what a user may click to understanding how user interests are formed, changed, and consolidated through continuous interactions. In e-commerce, short-video platforms, news feeds, music streaming, and online advertising, user preference is not a static label but a dynamic process shaped by clicks, views, bookmarks, purchases, dwell time, and other behavioral signals. Sequential recommendation addresses this problem by predicting the next item a user is most likely to interact with, given the user's chronological interaction history. Early methods typically used Markov chains or sequential matrix factorization to model transition relationships between adjacent behaviors (Shani et al., 2005; Rendle et al., 2010). Recurrent neural networks were later introduced to capture more complex patterns of preference evolution (Hidasi et al., 2016; Hidasi and Karatzoglou, 2018). More recently, self-attention-based models such as SASRec and BERT4Rec have substantially improved long-range dependency modeling and have become important baselines in sequential recommendation (Kang and McAuley, 2018; Sun et al., 2019).

However, real-world recommendation scenarios are creating new tensions for this mainstream paradigm. User histories vary substantially in length, while item information is becoming richer:

titles, categories, brands, prices, reviews, contexts, and cross-domain behaviors all provide semantic cues for recommendation systems. At the same time, online recommendation must satisfy strict latency, memory, and throughput constraints. Although self-attention can capture dependencies between arbitrary positions, its computation and storage costs usually grow quadratically with sequence length (Kang and McAuley, 2018; Sun et al., 2019). This motivates efficient sequence encoders even when the primary challenge is not exclusively very long histories. Researchers have therefore re-examined the potential of convolutional networks for sequential recommendation. Caser, NextItNet, CosRec, and RCNN show that convolutional structures can effectively capture local continuous behavior patterns (Tang and Wang, 2018; Yuan et al., 2019; Yan et al., 2019; Xu et al., 2019). The recent ConvRec model further uses hierarchical downsampling convolution to aggregate neighboring interactions layer by layer and compress user histories into compact interest representations with linear sequence-processing complexity (Elsayed et al., 2026). These studies motivate the more general question considered here: how can a model compress semantically rich user histories efficiently and adaptively under a limited computational budget?

Improving compression efficiency alone is still insufficient. Conventional sequential recommendation usually treats user history as an item-ID sequence, where each interaction is mainly represented by a discrete item identifier. This abstraction is overly simplified for attribute-aware recommendation. A user interaction is not merely a user clicking an item; it also carries item identity, attribute semantics, brand preference, category intent, price sensitivity, textual description, temporal context, and the user's interest state at that moment. Existing attribute-aware sequential recommendation methods enhance representation learning by introducing item attributes, contextual features, or self-supervised tasks (Zhou et al., 2020; Xie et al., 2022; Liu et al., 2021; Rashed et al., 2022; Seol et al., 2024). For example, S3-Rec learns correlations among items, attributes, subsequences, and sequences through mutual-information maximization (Zhou et al., 2020). DIF-SR explores decoupled fusion of side information in sequential recommendation (Xie et al., 2022). CARCA and ProxyRCA further use context-aware attention or proxy representations to strengthen attribute modeling (Rashed et al., 2022; Seol et al., 2024). These methods demonstrate that attribute information can alleviate the limitations of ID representations in sparse-data, long-tail-item, and cold-start scenarios (Zhou et al., 2020; Xie et al., 2022; Rashed et al., 2022; Seol et al., 2024). Nevertheless, most methods still treat attributes as additional features: item IDs, attributes, and contexts are first projected into a shared space and then passed to a unified sequence encoder. This paradigm tends to overlook a key fact: user history is not a string of isolated item symbols but a sequence of interaction memories that contain semantic, contextual, and temporal states.

This issue becomes even more salient with the development of large language models and recommender foundation models. External semantic representations can extract higher-level knowledge from titles, descriptions, reviews, and cross-domain behaviors, providing recommendation models with generalization ability beyond ID-based collaborative signals (Liu et al., 2024a; Hou et al., 2024; Wei et al., 2024; Jiang et al., 2026). The LLM-as-Extractor paradigm typically transfers item embeddings generated by large language models into conventional sequential recommendation models to enhance item semantic representations (Liu et al., 2024a; Hou et al., 2024; Wei et al., 2024). In industrial recommendation, LoopFM further argues that compared with distilling knowledge through a single soft label, extracting, compressing, and organizing the intermediate historical representations of a foundation model into a user-history sequence can provide a higher-bandwidth knowledge-transfer channel while avoiding the online inference cost of real-time foundation-model calls (Jiang et al., 2026). These studies show that external semantic representations and high-level historical representations can serve not only as auxiliary information but also as important carriers for understanding user-interest evolution.

However, semantic representations cannot be simply concatenated into a sequence model. First, pretrained semantic embeddings often exhibit geometric imbalance: many vectors may concentrate

along a few principal directions, resulting in anisotropic representation spaces and weakening the compatibility between semantic and collaborative signals (Lee et al., 2026; Reimers and Gurevych, 2019; Gao et al., 2021). Studies on semantic representation, including Sentence-BERT and SimCSE, have shown that the spatial structure of pretrained text embeddings substantially affects downstream similarity modeling and representation transfer (Reimers and Gurevych, 2019; Gao et al., 2021). ACE further reports anisotropy in LLM-generated item embeddings for LLM-enhanced sequential recommendation and proposes an anisotropy-controllable embedding approach (Lee et al., 2026). Second, even when semantic embeddings are of high quality, using them only as static item representations still fails to answer a more critical question: how should these semantic signals be organized, retained, forgotten, and compressed within a user history? The key to semantic-enhanced recommendation is not only obtaining better item embeddings, but also understanding how each historical interaction participates as semantic memory in the formation of the user's current interest.

This formulation places SMC-Rec within the user-modeling and personalization tradition. Rather than treating a user history as an item-ID sequence to be scored, the system acquires external semantic knowledge from item text, represents heterogeneous interaction evidence as interaction-level user states, controls the geometry of the acquired knowledge, adaptively selects and compresses historical knowledge under uncertainty, and evaluates both predictive quality and deployment cost. Prior work treats recommendation as a user-adaptive system problem, including fairness-oriented analyses of recommender systems (Deldjoo et al., 2023), modular personalized recommendation (Pelánek et al., 2024), and privacy-preserving on-device next-item prediction (Saiapin et al., 2024). SMC-Rec complements this line of work by focusing on how a user model should compress semantically enriched interaction memories when its own predictions are uncertain.

Figure 1 summarizes the problem setting and the proposed reformulation from fixed item-sequence encoding to uncertainty-guided semantic-memory compression.

From the perspective of semantic memory compression, existing methods still have three fundamental limitations. First, attributes and semantic features are usually directly fused into item representations without calibrating the geometric quality of the external semantic space. When semantic embeddings suffer from directional collapse or imbalanced dimensional distributions, direct fusion may introduce redundancy and even interfere with model learning (Lee et al., 2026; Reimers and Gurevych, 2019; Gao et al., 2021). Second, most efficient sequence models adopt fixed rules to compress user history, such as fixed convolution kernels, fixed strides, or fixed pooling strategies (Tang and Wang, 2018; Yuan et al., 2019; Elsayed et al., 2026). However, users differ in historical density, interest stability, and behavioral noise. For users with stable interests, the model can more aggressively aggregate long-term preferences; for users with uncertain states or abrupt behavior shifts, premature compression may discard critical local cues. Third, existing training objectives usually focus on next-item prediction (Kang and McAuley, 2018; Sun et al., 2019; Zhou et al., 2020). Although multi-step future behaviors contain information about interest evolution, not all future signals are equally reliable. Prior studies on future supervision show that imposing the same future-learning strength on all samples may introduce noise; a more appropriate strategy is to adaptively adjust future-supervision intensity according to the model's confidence in the current state (Ma et al., 2019; Huang et al., 2025; Cui et al., 2026). Therefore, semantic memory compression is not only an architectural problem but also a joint modeling problem involving representation calibration, compression strategy, and training signals.

This paper proposes SMC-Rec, a user-adaptive Spectral Semantic Memory Convolution framework with Uncertainty-Guided Learning for sequential recommendation. SMC-Rec does not treat user history as an ordinary item-embedding sequence. Instead, it first constructs an interaction-level semantic-memory sequence, where each memory unit consists of item ID, attributes, context, time interval, and external semantic representation, enabling the model to organize heterogeneous interaction evidence as a structured interaction state. To address possible geometric imbalance in

external semantic representations, SMC-Rec introduces a spectral calibration mechanism that adjusts the semantic space so that it can preserve semantic hierarchy while becoming more suitable for joint reasoning with collaborative signals.

After semantic memory construction and calibration, SMC-Rec adopts a pyramid convolutional structure to compress user history at multiple temporal scales. Shallow convolutions capture local transitions between adjacent interactions; middle layers model stage-level interest combinations; deeper layers aggregate long-term preferences. Unlike fixed downsampling convolution, SMC-Rec further introduces an uncertainty-aware mechanism that dynamically regulates compression strength according to the current prediction state. Historical compression is therefore modeled as an adaptive user-state selection process rather than a static structural operation.

SMC-Rec also embeds training signals into the multi-scale memory-compression process. Because different pyramid levels naturally correspond to interest representations over different temporal ranges, this paper aligns near-future, mid-term-future, and long-term-future signals with different levels so that the model can learn the direction of interest evolution while compressing history. The model also uses prediction differences among different levels to construct internal multi-view training signals. Studies on negative sampling show that single self-guided hard negative mining can be affected by current model bias, whereas multi-view disagreement and consensus signals help improve negative-sample informativeness and training stability (Rendle et al., 2009; Zhang et al., 2016; Li et al., 2026). SMC-Rec does not introduce external teachers or peer models; it directly uses prediction differences among short-term, mid-term, and long-term pyramid representations to identify more discriminative candidate samples. A candidate item that is relevant under the short-term view but inconsistent under the long-term view often better reveals a boundary of user interest than an ordinary random negative sample. SMC-Rec improves training quality by exploiting intra-pyramid disagreement and consensus without introducing additional inference models.

The main contributions are summarized as follows:

(1) Semantic-memory formulation. We formulate attribute-aware sequential recommendation as an interaction-level semantic-memory representation and compression problem, in which each historical interaction jointly retains item identity, attributes, context, time interval, and external semantics before temporal compression. The interaction-level representation is instantiated with a standard concatenate–project operator.

(2) Uncertainty-adaptive encoder. The primary architectural contribution is uncertainty-adaptive pyramid compression inside the sequence encoder. Unlike the fixed hierarchical downsampling of ConvRec (Elsayed et al., 2026), SMC-Rec estimates prediction uncertainty at each level and uses it to regulate compression strength and the retention of previous-layer information. This extends UFRec (Cui et al., 2026), where uncertainty controls future-supervision intensity but not the encoder state transition.

(3) Intra-pyramid multi-view learning. SMC-Rec constructs short-, mid-, and long-term views from a single pyramid encoder. Their disagreement identifies informative negatives, while their consensus stabilizes the final representation. Unlike MDCNS (Li et al., 2026), this design requires no external teacher or peer models.

(4) Controlled semantic preprocessing and systematic evaluation. An offline SVD-based spectral calibration, motivated by prior work on embedding anisotropy (Lee et al., 2026; Reimers and Gurevych, 2019; Gao et al., 2021), is used as supporting preprocessing. The complete system is evaluated through full-ranking accuracy, seed-level statistics, ablation, semantic geometry, uncertainty groups, robustness, sparse/cold-start behavior, and measured wall-clock efficiency.

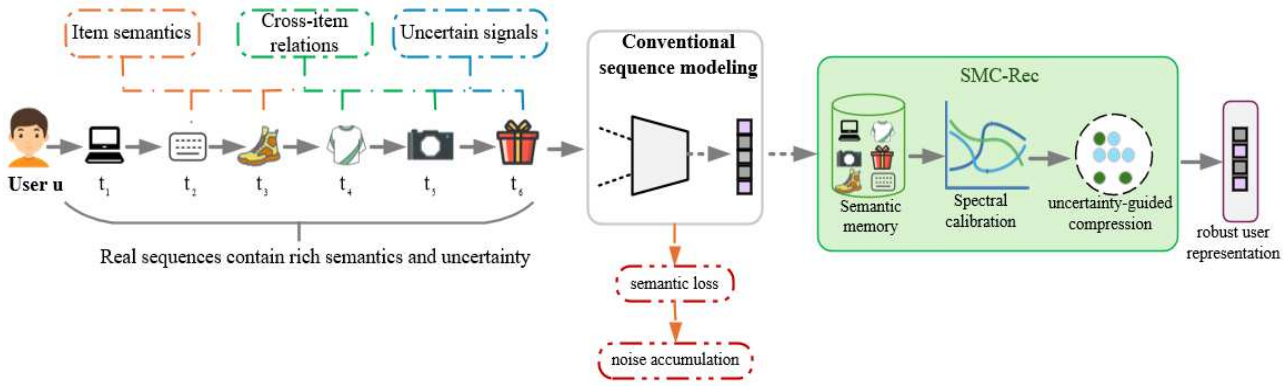


Figure 1. Problem setting and motivation of SMC-Rec. The left side shows that a real interaction history contains item semantics, cross-item relations, and uncertain behavioral signals. Conventional fixed sequence modeling can discard semantic evidence and accumulate noise. SMC-Rec instead constructs interaction-level semantic memories, calibrates the external semantic space, and performs uncertainty-guided multi-scale compression to obtain a robust user representation.

Figure 1 highlights three bottlenecks—semantic loss, rigid compression, and uncertainty-insensitive supervision—and the corresponding design responses: semantic-memory construction, spectral calibration, and uncertainty-guided pyramid compression.

The remainder of this paper is organized as follows. Section 2 reviews related work and positions SMC-Rec relative to its closest components. Section 3 introduces the compact problem formulation and notation. Section 4 presents the proposed methodology, including offline semantic calibration, semantic-memory construction, uncertainty-adaptive pyramid compression, training objectives, and complexity analysis. Section 5 reports the experimental protocol, predictive results, statistical analysis, ablations, robustness, sparse/cold-start behavior, uncertainty analysis, and measured efficiency. Section 6 concludes the paper and discusses limitations and future work.

2. Related Work

2.1 Sequential Recommendation and Efficient Sequence Modeling

Sequential recommendation aims to predict a user's next behavior from the user's chronologically ordered interaction history and is an important task for modeling dynamic interest evolution. Early methods usually used Markov chains or sequential matrix factorization to model short-term transition relationships between adjacent behaviors (Shani et al., 2005; Rendle et al., 2010). These methods are interpretable and computationally efficient, but they struggle to capture high-order, nonlinear, and long-range dependencies. With the development of deep learning, recurrent neural networks were introduced into sequential recommendation to model how user interests evolve over time (Hidasi et al., 2016; Hidasi and Karatzoglou, 2018). However, RNN-based methods usually rely on sequential computation, have limited parallelism during training and inference, and are therefore inefficient for large-scale long-sequence data.

In recent years, self-attention-based models have become a mainstream paradigm for sequential recommendation. SASRec uses unidirectional self-attention to model dependencies among items at different positions in a user history (Kang and McAuley, 2018). BERT4Rec further adopts a bidirectional Transformer and a Cloze prediction task to enhance sequence representation learning (Sun et al., 2019). These methods can flexibly capture long-range dependencies and achieve strong performance on multiple public datasets. Nevertheless, the computation and memory costs of self-attention generally grow quadratically with sequence length, making the efficiency bottleneck more pronounced when user histories are long or online inference constraints are strict.

To alleviate this issue, convolutional networks have received renewed attention. Caser captures local sequential patterns through horizontal and vertical convolutions (Tang and Wang, 2018). NextItNet uses stacked convolutional structures for next-item prediction (Yuan et al., 2019). CosRec and RCNN further explore the expressiveness of convolutional or convolutional-recurrent structures

for sequence modeling (Yan et al., 2019; Xu et al., 2019). Recently, ConvRec proposed a hierarchical downsampling convolutional structure that aggregates neighboring interactions layer by layer and compresses long histories into compact user-interest representations, maintaining linear complexity while balancing local patterns and global historical information (Elsayed et al., 2026). Unlike these studies, this paper does not merely improve the sequence encoder. Instead, it re-examines user history from the perspective of semantic memory compression: when each historical interaction contains ID, attribute, context, and external semantic representation, the key problem is not only how to model the sequence, but also how to calibrate, select, and compress interaction-level semantic memories.

2.2 Knowledge Representation in Attribute-aware and Semantic-enhanced Recommendation

Traditional sequential recommendation mainly uses item IDs as input and learns user preferences through collaborative signals. However, in sparse-data, cold-start, and long-tail-item scenarios, ID-only representations often generalize poorly. Attribute-aware sequential recommendation enhances user-interest modeling by introducing item attributes, contextual information, and auxiliary self-supervised signals. S3-Rec improves sequence representation quality by maximizing mutual information among items, attributes, subsequences, and sequences (Zhou et al., 2020). DIF-SR explores decoupled side-information fusion to reduce the interference caused by directly mixing different feature types (Xie et al., 2022). NOVA, CARCA, and ProxyRCA further improve attribute modeling from the perspectives of non-invasive fusion, context-aware attention, and proxy representation learning (Liu et al., 2021; Rashed et al., 2022; Seol et al., 2024).

These methods demonstrate that attribute information can effectively complement ID representations. However, most of them still treat attributes as additional features: item IDs, attributes, and contexts are usually projected into a unified vector space and then input into a sequence encoder. This paradigm overlooks the compositional nature of interactions. A user behavior corresponds not only to an item but also to the user's response to a semantic need under a specific time and context. Thus, attribute information should not be merely a supplement to item embeddings; it should be part of interaction-level semantic memory.

Large language models and recommender foundation models have recently provided richer semantic representation capabilities for recommender systems. Related studies show that LLMs can generate semantic representations from item texts, descriptions, reviews, and external knowledge to enhance conventional recommendation models (Liu et al., 2024a; Hou et al., 2024; Wei et al., 2024). From an industrial recommendation perspective, LoopFM further points out that distilling foundation-model knowledge through soft labels alone creates an information-bandwidth bottleneck. In contrast, extracting, compressing, and organizing the intermediate historical representations of a foundation model as a user-history sequence can provide a higher-bandwidth knowledge-transfer channel (Jiang et al., 2026). However, semantic representations cannot be simply concatenated into recommendation models. ACE shows that LLM-generated item embeddings suffer from anisotropy, where vectors may concentrate along a few principal directions and thus affect their compatibility with collaborative signals (Lee et al., 2026). Similarly, semantic-representation studies such as Sentence-BERT and SimCSE also show that the geometry of pretrained text-embedding spaces affects downstream semantic similarity modeling and representation transfer (Reimers and Gurevych, 2019; Gao et al., 2021).

This paper is closely related to semantic-enhanced recommendation, particularly ACE (Lee et al., 2026). ACE identifies anisotropy in LLM-generated item embeddings and proposes a learnable anisotropy-controllable embedding mechanism. SMC-Rec shares the anisotropy-control motivation

but uses a distinct closed-form SVD-based spectral calibration as an offline preprocessing step. After calibration, SMC-Rec treats the calibrated semantic vector as one component of an interaction-level state and studies how that state is adaptively compressed across temporal scales according to prediction uncertainty. Thus, the central novelty is not simply better semantic initialization, but uncertainty-controlled history compression over semantically enriched interactions.

Surveys of the field document the progression from ID-only transition models to semantic and side-information-aware user models (Fang et al., 2020; Pan et al., 2025). From a user-modeling perspective, prior work treats recommendation as the problem of acquiring, representing, integrating, and evaluating knowledge about users and items; representative examples examine fairness in recommender systems (Deldjoo et al., 2023), modular rule-based personalization (Pelánek et al., 2024), and federated on-device next-app prediction (Saiapin et al., 2024). SMC-Rec addresses a complementary question: after semantic knowledge has been acquired, how should interaction-level evidence be geometrically calibrated and adaptively compressed over time so that the resulting user state remains useful under uncertainty?

2.3 Uncertainty-aware Supervision and Self-supervised Knowledge Signals

Most sequential recommendation methods use immediate next-item prediction as the primary training objective (Kang and McAuley, 2018; Sun et al., 2019; Zhou et al., 2020). This objective is consistent with next-item prediction at inference time, but it also suffers from sparse supervision and can make the model short-sighted. To mitigate this limitation, researchers have introduced various auxiliary training signals. S3-Rec mines mutual-information relationships among items, attributes, subsequences, and sequences through self-supervised tasks (Zhou et al., 2020). HGN uses a hierarchical gating structure to jointly model short-term interests and long-term preferences (Ma et al., 2019). Other methods attempt to use multi-step future interactions as auxiliary supervision to help the model learn longer-range interest-evolution trends (Huang et al., 2025; Cui et al., 2026).

Future behaviors, however, are not always reliable. When the model is still uncertain about the current next-step prediction, prematurely introducing distant-future supervision may amplify noise and even conflict with the main task. UFRec therefore proposes uncertainty-guided future learning, dynamically adjusting future-supervision strength according to the model's confidence in the current prediction so that the model learns farther future signals when confident and focuses on the current task when uncertain (Cui et al., 2026). This idea suggests that future supervision should not be applied indiscriminately as an auxiliary task; it should be conditioned on the model's current state.

SMC-Rec builds on the uncertainty-guided learning intuition of UFRec (Cui et al., 2026) but uses uncertainty in a different location as well as for a different purpose. UFRec primarily modulates the strength of future supervision according to current confidence. In SMC-Rec, uncertainty also enters the pyramid encoder itself: the level-wise gate controls the balance between the newly compressed candidate state and retained historical information, thereby changing the compression strength before the final user representation is formed. Pyramid-level future supervision is then aligned with near-, mid-, and long-term states. This encoder-level uncertainty gating is a central distinction from applying uncertainty only as an auxiliary-loss weight.

2.4 Negative Sampling and Multi-view Knowledge Consensus

In implicit-feedback recommendation, user interactions usually provide only positive feedback, making negative-sample construction crucial for model training. BPR formulates recommendation learning as a ranking optimization problem between positive and negative samples and has become a classic objective for implicit-feedback recommendation (Rendle et al., 2009). Random negative sampling is simple and efficient but often produces uninformative negatives. Hard negative sampling can provide stronger gradient signals, but it may introduce false negatives or be affected by the

current model bias.

Recent studies have improved negative sampling from different perspectives. GenNi addresses negative-sample generation in sequential recommendation by generating more informative negatives based on the user preferences learned by the current model (Chen et al., 2022). MixGCF targets GNN-based recommendation models and synthesizes hard negatives in the continuous representation space to improve the training value of negative samples (Huang et al., 2021). More recently, MDCNS argues that single self-guided hard negative sampling can create a self-reinforcing loop and proposes a Teacher-Peer-Self framework that improves negative-sample quality and training stability through multi-source scoring, disagreement-based re-ranking, and consensus distillation (Li et al., 2026).

This paper is inspired by multi-view disagreement and consensus, but its source of views differs from MDCNS (Li et al., 2026). MDCNS constructs Teacher-Peer-Self scoring sources, whereas SMC-Rec directly reuses different temporal levels of the same pyramid encoder: shallow states represent short-term interests, middle states represent stage-level interests, and deeper states represent longer-term preferences. Candidate items with large cross-level disagreement are treated as informative negatives, while cross-level consensus is distilled into the final representation. The resulting intra-pyramid multi-view construction is therefore internal to one encoder and introduces no additional teacher or peer model in the inference path.

The role of consensus also connects to work on user-adaptive and responsible recommendation (Deldjoo et al., 2023), which reviews how agreement and fairness structure affect recommender systems. Our use of consensus is an internal training signal derived from temporal pyramid levels rather than a post-hoc explanation or fairness mechanism.

2.5 Position of This Work

In summary, prior work has advanced sequential recommendation from the perspectives of sequence modeling, attribute fusion, semantic enhancement, future supervision, and negative sampling. However, these directions usually address only one type of problem in user histories: efficient sequence models focus on reducing the cost of long-sequence modeling (Tang and Wang, 2018; Yuan et al., 2019; Elsayed et al., 2026); attribute-aware methods focus on using side information to enhance representations (Zhou et al., 2020; Xie et al., 2022; Liu et al., 2021; Rashed et al., 2022; Seol et al., 2024); semantic-enhanced methods focus on introducing LLM or foundation-model representations (Liu et al., 2024a; Hou et al., 2024; Wei et al., 2024; Jiang et al., 2026; Lee et al., 2026); and future-supervision and negative-sampling methods focus on providing more effective training signals (Cui et al., 2026; Li et al., 2026; Chen et al., 2022; Huang et al., 2021).

The position of SMC-Rec can be stated more precisely relative to its closest components. ConvRec (Elsayed et al., 2026) provides the fixed hierarchical convolutional backbone; SMC-Rec adds uncertainty-dependent compression control inside that backbone. ACE (Lee et al., 2026) motivates anisotropy control for LLM-enhanced item embeddings; SMC-Rec uses a distinct closed-form SVD-based calibration as offline semantic preprocessing. UFRec (Cui et al., 2026) motivates uncertainty-guided future learning; SMC-Rec extends the role of uncertainty from loss allocation to encoder-level compression. MDCNS (Li et al., 2026) motivates disagreement/consensus learning; SMC-Rec replaces external Teacher–Peer–Self views with short-, mid-, and long-term views already present inside the pyramid. The concatenate–project fusion of Eq. (13) is a standard implementation mechanism; the two method-level additions are (i) uncertainty gating of compression strength inside the encoder and (ii) intra-pyramid multi-view construction for divergence-aware negative mining and consensus learning.

3. Preliminaries and Problem Formulation

This section keeps only the notation needed to state the task and the inputs to SMC-Rec. Routine textbook derivations are omitted so that the preliminaries do not duplicate the model equations. Throughout the paper, bold uppercase symbols denote matrices, bold lowercase symbols denote

vectors, calligraphic symbols denote sets, and ordinary italic symbols denote scalar indices or coefficients. $|A|$ denotes the cardinality of a set A, $R^{a\times b}$ denotes the space of real-valued $a \times b$ matrices, and $(\cdot)^\top$ denotes transpose. Standard operators such as $\sum$, exp, log, $softmax$, and $argmax$ retain their conventional mathematical meanings; equation-specific variables and nonstandard operators are defined immediately before or after every numbered equation.

3.1 Sequential Recommendation and Observed Interaction Evidence

Let U denote the user set and I the item catalog. For a user u, interactions are ordered chronologically. The sequential recommendation task and its next-item decision are summarized by Eq. (1).

$$S_u = \left(i_{u,1}, i_{u,2}, \dots, i_{u,n_u}\right), \qquad \hat{\imath}_{u,t+1} = \underset{i\in I}{argmax} P\left(i \mid S_{u,1:t}\right) \tag{1}$$

In Eq. (1), u indexes a user, n_u is the number of observed interactions of that user, $i_{u,t}$ is the item observed at position t, S_u is the complete chronological sequence, $S_{u,1:t}$ is its prefix through position t, $P\left(i \mid S_{u,1:t}\right)$ is the conditional next-item probability for candidate i in I, $argmax$ selects the candidate with the largest conditional probability, and $\hat{\imath}_{u,t+1}$ is the predicted next item. Each observed interaction may additionally contain item attributes, context, and elapsed-time information; their learned representations are introduced in Section 3.2 and used by the constructor in Section 4.3.

3.2 External Semantics and Interaction-Level Semantic Memory

Item-side text is encoded offline with the frozen sentence-transformers/all-mpnet-base-v2 Sentence-BERT checkpoint. The resulting external semantic vectors are collected in Eq. (2).

$$E^{sem} = [1^e_{sem}; 2^e_{sem}; \dots; |I|^e_{sem}] \in R^{|I|\times d_0} \tag{2}$$

In Eq. (2), E^{sem} is the item-semantic matrix, i^e_{sem} is the Sentence-BERT vector of item i, the semicolon denotes row-wise stacking, $|I|$ is the catalog size, d_0 is the external semantic dimension (768 in our implementation), and $R^{|I|\times d_0}$ is the corresponding real matrix space. Section 4.2 converts E^{sem} into a calibrated matrix G^{sem}; its row for item i is denoted i^g_{sem}.

For an observed interaction (u, t), SMC-Rec treats identity, attributes, context, elapsed-time information, and calibrated external semantics as components of one semantic-memory unit. The conceptual unit is defined in Eq. (3); it specifies the information content but does not prescribe the learnable fusion operator.

$$m_{u,t} \equiv \left(p_{i_{u,t}}, a_{i_{u,t}}, c_{u,t}, d_{u,t}, {i_{u,t}}^g_{sem}\right) \tag{3}$$

In Eq. (3), $m_{u,t}$ is the semantic-memory unit, $p_{i_{u,t}}$ is the learnable ID vector of the interacted item, $a_{i_{u,t}}$ is its attribute representation, $c_{u,t}$ is the interaction context, $d_{u,t}$ is the learned representation of the elapsed time interval, and ${i_{u,t}}^g_{sem}$ is the calibrated semantic vector. The concrete concatenate-project implementation is introduced only once, in Section 4.3.

The historical prefix is therefore represented as an ordered semantic-memory matrix, Eq. (4).

$$M_{u,1:t} = \left[m_{u,1}; m_{u,2}; \dots; m_{u,t}\right] \in R^{t\times d_m} \tag{4}$$

In Eq. (4), $M_{u,1:t}$ is the semantic-memory sequence for user u up to position t; the semicolon denotes row-wise chronological stacking; t is the prefix length; d_m is the dimension of one learnable memory vector; and $R^{t\times d_m}$ denotes the space of real-valued matrices with t rows and d_m columns.

3.3 Multi-scale States, Uncertainty, and Training Signals

The pyramid encoder produces L level-specific user states $u, t^z_{(l)}$, where l indexes temporal scale from 1 to L; shallow levels have smaller receptive fields and deeper levels summarize broader historical context. Q denotes the item-representation matrix and q_i its row for candidate item i. At

each level, the model forms a full-catalog prediction distribution and Shannon entropy $u, t_{(l)}^{H}$; this entropy controls the compression gate defined in Section 4.4. For future learning, u, t_K^F denotes the next K observed items after the current prefix, where K is the maximum future horizon. For negative mining, u, t_{-}^{C} denotes a pool of N items not observed by the user, where N is the candidate-pool size. These symbols are instantiated by the numbered model equations in Section 4, avoiding a second set of preliminary formulas.

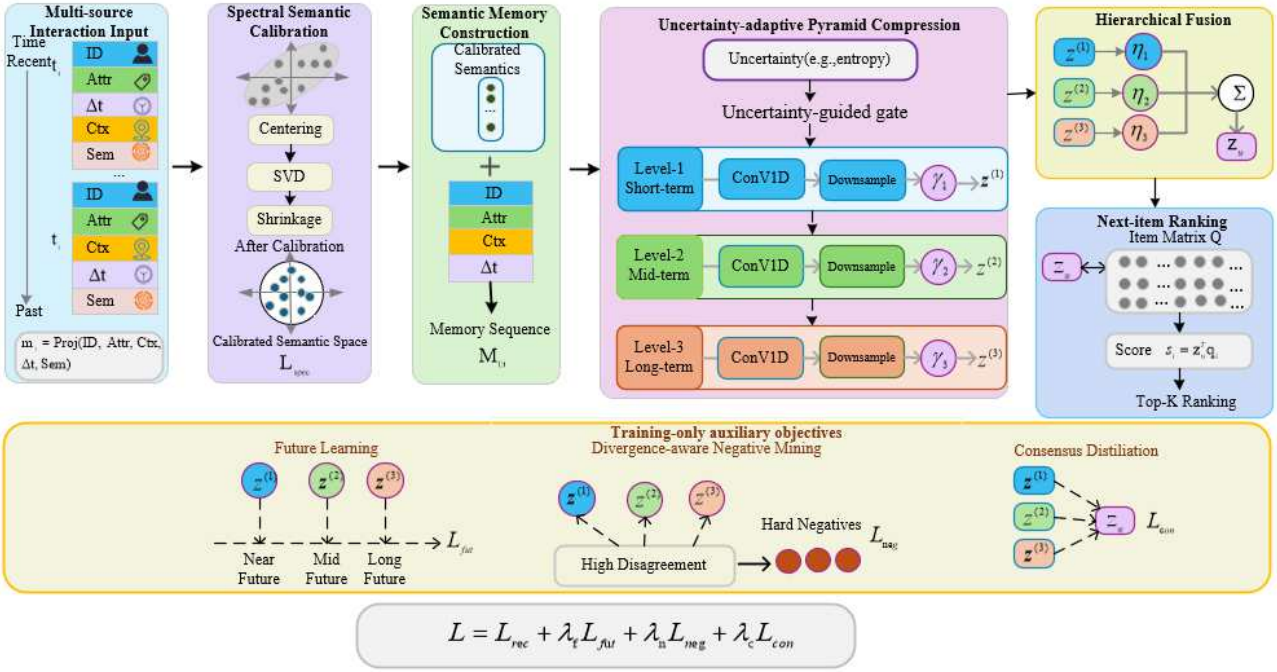
3.4 Problem Definition

Given $M_{u,1:t}$, the item catalog I, future signal set u, t_K^F, and candidate negative pool u, t_{-}^{C}, SMC-Rec learns parameters Θ that produce a compact user state and rank the true next item highly. The trainable objective is defined explicitly in Section 4.8, while the deployment-time ranking rule is defined there as well. This formulation emphasizes three requirements: interaction evidence is represented as semantic memory, external semantics are geometrically calibrated before sequence compression, and the amount of historical compression is conditioned on prediction uncertainty.

4. Methodology

This section introduces the proposed SMC-Rec model as a user-adaptive recommendation framework. Given a user historical semantic-memory sequence, SMC-Rec first performs spectral calibration on external semantic representations to mitigate anisotropy in the pretrained semantic space. The calibration is an offline preprocessing step: the calibrated semantic matrix is computed once before recommender training and is subsequently used as a fixed lookup table. SMC-Rec then constructs interaction-level semantic memory units and performs multi-scale compression through an uncertainty-adaptive pyramid convolutional structure. The downstream recommender is optimized by next-item prediction, pyramid-level future supervision, and intra-pyramid divergence-consensus objectives; spectral calibration itself is not a gradient-based training loss. Symbols already defined in Section 3 retain the same meaning throughout Section 4; each equation below defines any symbol introduced for the first time.

As shown in Figure 2, SMC-Rec follows a six-stage pipeline: multi-source interaction input, offline spectral semantic calibration, semantic memory construction, uncertainty-adaptive pyramid compression, hierarchical fusion, and next-item ranking. The calibrated semantic matrix is cached before downstream training and reused as a fixed lookup table. The lower part of the framework illustrates training-only auxiliary objectives, including future learning, divergence-aware negative mining, and consensus distillation, together with the main recommendation loss.



$$L = L_{rec} + \lambda_f L_{fut} + \lambda_n L_{neg} + \lambda_c L_{con}$$

Figure 2. Overall architecture of the proposed SMC-Rec framework. Spectral calibration is performed once as offline semantic preprocessing, and the calibrated semantic matrix is cached and kept fixed during downstream recommender training. The uncertainty-adaptive pyramid encoder remains in the inference path, whereas future supervision, divergence-aware negative ranking, and consensus distillation are training-only auxiliary objectives. The downstream trainable objective contains the main recommendation loss plus these three auxiliary losses; no spectral-loss term is optimized.

4.1 Overall Architecture

Given the semantic-memory sequence $M_{u,1:t}$ of user u before time step t, SMC-Rec learns an end-to-end function F_Θ that maps it into the final user representation $z_{u,t}$ while also outputting historical-state representations at different pyramid levels $\{u, t_{(l)}^z l = 1\}_L^\}$:

$$\left(z_{u,t}, \{u, t_{(l)}^z l = 1\}_L^\}\right) = F_\Theta\left(M_{u,1:t}\right) \quad (5)$$

In Eq. (5), F_Θ is the complete learnable SMC-Rec encoder parameterized by Θ; $M_{u,1:t}$ is the semantic-memory matrix from Eq. (4); L is the number of pyramid levels; $u, t_{(l)}^z$ is the level-l historical state; and $z_{u,t}$ is the final fused user state used for recommendation.

SMC-Rec contains four core user-modeling modules. First, an offline spectral calibration module geometrically adjusts the external semantic representations and caches the calibrated matrix as a fixed lookup table. Second, the semantic memory construction module fuses item IDs, attributes, context, time intervals, and calibrated semantic representations into interaction-level memory units. Third, the uncertainty-adaptive pyramid convolution module compresses semantic-memory sequences layer by layer and generates short-term, mid-term, and long-term historical states. Fourth, the multi-task learning module optimizes the downstream recommender using next-item supervision, future-behavior supervision, and intra-pyramid divergence-consensus signals.

4.2 Spectral Semantic Memory Calibration

External semantic representations provide rich textual semantics and attribute knowledge, but their vector spaces often suffer from anisotropy, where a few principal directions account for excessive variance (Lee et al., 2026; Reimers and Gurevych, 2019; Gao et al., 2021). In SMC-Rec, these external item representations are generated offline by the sentence-transformers/all-mpnet-base-v2 Sentence-BERT (SBERT) checkpoint (Reimers and Gurevych, 2019) and are not jointly fine-tuned with the sequential recommender. Directly concatenating an anisotropic semantic space into a sequence model may cause semantic similarity to be dominated by principal directions and weaken

compatibility with collaborative signals. Therefore, spectral calibration is applied once to the Sentence-BERT item-embedding matrix before semantic-memory construction.

Given the centered semantic-representation matrix $\tilde{E}^{sem}$, its singular value decomposition is:

$$\tilde{E}^{sem} = U\Sigma V^{\top} \tag{6}$$

In Eq. (6), $\tilde{E}^{sem}$ is the mean-centered version of E^{sem} from Eq. (2); $U \in R^{|I|\times r}$ and $V \in R^{d_0\times r}$ are the left and right singular-vector matrices of the thin SVD; $\Sigma = diag(\sigma_1, \dots, \sigma_r)$ is the diagonal singular-value matrix with nonnegative singular values ordered from largest to smallest; j indexes singular directions; $r = min(|I|, d_0)$ is the thin-SVD rank bound; and $(\cdot)^{\top}$ denotes matrix transpose.

$$g_\lambda(\sigma_j) = \sqrt{\frac{j_2^\sigma}{j_2^\sigma + \lambda}} \tag{7}$$

In Eq. (7), $g_\lambda(\cdot)$ is the spectral-calibration mapping, σ_j is the j-th singular value from Eq. (6), and λ is the nonnegative calibration-strength hyperparameter. We treat this parameter as an empirically selected control of the calibrated spectrum rather than assigning a universal monotonic interpretation to its value; its observed effect on embedding geometry and recommendation quality is reported directly in Section 5.7.

$$G^{sem} = U\, g_\lambda(\Sigma) \tag{8}$$

In Eq. (8), $g_\lambda(\Sigma)$ means that Eq. (7) is applied element-wise to the diagonal singular values; G^{sem} is the calibrated semantic matrix, and i_{sem}^{g} denotes its row for item i. The calibrated matrix has the same catalog-row indexing as E^{sem} and is cached as a frozen lookup table for downstream training.

To quantify, rather than optimize, the amount of semantic-neighborhood distortion introduced by spectral calibration, we compare pairwise cosine similarities before and after calibration. For any two items i and j, the original semantic similarity and calibrated semantic similarity are respectively defined as:

$$ij_{sem}^{r} = cos(i_{sem}^{e}, j_{sem}^{e}) \tag{9}$$

$$ij_{sem}^{\hat{r}} = cos(i_{sem}^{g}, j_{sem}^{g}) \tag{10}$$

The corresponding semantic-neighborhood distortion diagnostic is defined as:

$$D_{sem} = \frac{1}{|\Omega|}\sum_{(i,j)\in\Omega}\left(ij_{sem}^{r} - ij_{sem}^{\hat{r}}\right)^2 \tag{11}$$

In Eqs. (9)–(11), ij_{sem}^{r} is the cosine similarity between original semantic vectors i_{sem}^{e} and j_{sem}^{e}; $ij_{sem}^{\hat{r}}$ is the corresponding calibrated similarity between i_{sem}^{g} and j_{sem}^{g}; $cos(\cdot,\cdot)$ denotes cosine similarity; i and j index catalog items; Ω is the set of item pairs used for the diagnostic; $|\Omega|$ is its cardinality; and D_{sem} is the mean squared semantic-neighborhood distortion. D_{sem} is diagnostic only and is not optimized by gradient descent.

4.3 Semantic Memory Construction

After semantic-space calibration, SMC-Rec constructs each interaction in the user history as a semantic memory unit. For the interaction of user u at time step t, let the item-ID representation be $p_{i_{u,t}}$, the attribute representation be $a_{i_{u,t}}$, the context representation be $c_{u,t}$, the time-interval representation be $d_{u,t}$, and the calibrated semantic representation be $i_{u,t}{}_{sem}^{g}$. The semantic memory unit is defined as:

$$m_{u,t} = \phi_m\left(p_{i_{u,t}}, a_{i_{u,t}}, c_{u,t}, d_{u,t}, i_{u,t}{}_{sem}^{g}\right) \tag{12}$$

Specifically, we use a standard concatenate-project structure to instantiate each semantic memory unit:

$$m_{u,t} = LN\left(\sigma\left(W_m\left[p_{i_{u,t}} \| a_{i_{u,t}} \| c_{u,t} \| d_{u,t} \| i_{u,t}{}^{g}_{sem}\right] + b_m\right)\right) \tag{13}$$

In Eqs. (12)–(13), ϕ_m is the learnable semantic-memory constructor; $m_{u,t}$ is its output for interaction (u,t); $p_i, a_i, c_{u,t}, d_{u,t}, i^g_{sem}$ are the ID, attribute, context, time-interval, and calibrated semantic vectors defined in Section 3; $\|$ denotes vector concatenation; W_m and b_m are learnable projection parameters; $\sigma(\cdot)$ is the nonlinear activation; $LN(\cdot)$ denotes Layer Normalization; and d_m is the output dimension. This concatenate-project operation is a controlled implementation choice rather than the novelty claim.

Given a historical prefix of length t, the user's semantic-memory sequence is formed by stacking the interaction-level memory vectors in chronological order, as shown in Eq. (14).

$$M_{u,1:t} = \left[m_{u,1}; m_{u,2}; \ldots; m_{u,t}\right] \in R^{t \times d_m} \tag{14}$$

In Eq. (14), $M_{u,1:t}$ stacks the t semantic-memory vectors chronologically; t is the prefix length; and d_m is the memory-vector dimension. The resulting matrix is the input sequence used by the pyramid encoder.

4.4 Uncertainty-Adaptive Pyramid Convolution

To model user interests at different temporal scales while keeping the sequence-processing cost of the convolutional backbone linear in history length, SMC-Rec adopts pyramid one-dimensional convolution to compress semantic-memory sequences layer by layer. Unlike fixed downsampling strategies (Elsayed et al., 2026), this paper further introduces an uncertainty-aware gating mechanism, enabling the model to dynamically regulate historical compression strength according to the current prediction state. The uncertainty gate itself requires catalog-level score computation and is therefore accounted for separately in the end-to-end complexity analysis in Section 4.9.

Let the input to layer 0 be:

$$u, t^H_{(0)} = M_{u,1:t} \tag{15}$$

Equation (15) initializes the pyramid with $u, t^H_{(0)} = M_{u,1:t}$, where $u, t^H_{(0)}$ is the level-0 input tensor. For pyramid layer l, the one-dimensional convolution is then defined by Eq. (16).

$$u, t^{\tilde{H}}_{(l)} = Conv1D_l\left(u, t^H_{(l-1)}; k_l, s_l\right) \tag{16}$$

In Eq. (16), l indexes the pyramid layer, $Conv1D_l$ is its one-dimensional convolution, k_l is the kernel size, s_l is the stride, $u, t^H_{(l-1)}$ is the preceding layer state, and $u, t^{\tilde{H}}_{(l)}$ is the raw convolutional output before projection and normalization.

$$u, t^R_{(l)} = LN\left(\sigma\left(u, t^{\tilde{H}}_{(l)} r^W_{(l)} + r^b_{(l)}\right)\right) \tag{17}$$

To avoid loss of original local information caused by deep compression, SMC-Rec introduces a dual residual structure from the previous layer and the original input. The previous-layer representation and the original memory sequence are first adjusted to the same length as the current layer through average pooling:

$$u, t^A_{(l-1)} = AvgPool_l\left(u, t^H_{(l-1)}\right) \tag{18}$$

$$u, t^B_{(0,l)} = l^{AvgPool}_{(0)}\left(u, t^H_{(0)}\right) \tag{19}$$

The candidate compressed state of layer l is then constructed as:

$$u, t^{\widehat{H}}_{(l)} = u, t^R_{(l)} + \alpha_l u, t^A_{(l-1)} + \beta_l u, t^B_{(0,l)} \tag{20}$$

In Eqs. (17)–(20), $u, t^R_{(l)}$ is the normalized transformed convolutional output; $r^W_{(l)}$ and $r^b_{(l)}$ are

learnable projection parameters; $\sigma(\cdot)$ and $LN(\cdot)$ denote the activation and Layer Normalization; $AvgPool$ aligns residual tensors to the current layer length; $u, t_{(l-1)}^{A}$ is the pooled previous-layer residual; $u, t_{(0,l)}^{B}$ is the original-input residual pooled to level l; α_l and β_l are learnable residual weights; and $u, t_{(l)}^{\hat{H}}$ is the candidate compressed state before uncertainty gating.

$$u, t_{(l)}^{\hat{z}} = Pool\left(u, t_{(l)}^{\hat{H}}\right) \tag{21}$$

Based on this temporary representation, the model computes the prediction distribution over the complete item catalog at the current level:

$$u, t_{(l)}^{y} = softmax\left(u, t_{(l)}^{\hat{z}} Q^{\top}\right) \tag{22}$$

In Eqs. (21)–(22), $Pool(\cdot)$ maps $u, t_{(l)}^{\hat{H}}$ to the temporary level vector $u, t_{(l)}^{\hat{z}}$; Q is the full item-representation matrix whose i-th row is q_i; $u, t_{(l)}^{y}$ is the full-catalog probability vector after softmax; and $u, t, i_{(l)}^{y}$ is the probability assigned to item i. Thus Eq. (22) performs one complete catalog-scoring pass at every pyramid level.

$$u, t_{(l)}^{H} = -\sum_{i \in I} u, t, i_{(l)}^{y} \, log u, t, i_{(l)}^{y} \tag{23}$$

The uncertainty-aware gating weight is defined as:

$$u, t_{(l)}^{\gamma} = exp\left(-\frac{u, t_{(l)}^{H}}{\tau}\right) \tag{24}$$

In Eqs. (23)–(24), $u, t_{(l)}^{H}$ is the Shannon entropy of the level-l catalog distribution, log denotes the natural logarithm, $u, t_{(l)}^{\gamma}$ is the uncertainty gate, $exp(\cdot)$ is the exponential function, and τ is the positive temperature coefficient controlling gate sensitivity. Higher entropy corresponds to greater prediction uncertainty and therefore stronger retention of the previous-layer residual.

Finally, the output representation of layer l is:

$$u, t_{(l)}^{H} = u, t_{(l)}^{\gamma} u, t_{(l)}^{\hat{H}} + \left(1 - u, t_{(l)}^{\gamma}\right) u, t_{(l-1)}^{A} \tag{25}$$

The hierarchical historical-state representation is obtained by pooling the current-layer output:

$$u, t_{(l)}^{z} = Pool\left(u, t_{(l)}^{H}\right) \tag{26}$$

The final user-state representation is obtained by a normalized weighted aggregation of the L hierarchical states, as defined in Eq. (27).

$$z_{u,t} = \sum_{l=1}^{L} \eta_l \, u, t_{(l)}^{z}, \qquad \sum_{l=1}^{L} \eta_l = 1 \tag{27}$$

In Eqs. (25)–(27), $u, t_{(l)}^{H}$ is the uncertainty-adapted layer tensor, $u, t_{(l)}^{z}$ is its pooled level state, η_l is the normalized learnable fusion weight for level l, and $z_{u,t}$ is the final multi-scale user state. The L fusion weights sum to one, so the final representation combines complementary temporal scales on a common scale.

4.5 Main Recommendation Objective

Based on the final user state $z_{u,t}$, SMC-Rec computes the prediction score for candidate item i as:

$$s_{u,t}(i) = u, t_{\top}^{z} q_i \tag{28}$$

In Eqs. (28)–(30), q_i is the embedding of candidate item i; $s_{u,t}(i)$ is its score with the final user state; $P_{\Theta}\left(i \mid M_{u,1:t}\right)$ is the full-catalog next-item probability under parameters Θ; j is the

summation index over the catalog in the softmax denominator; $exp(\cdot)$ is the exponential function; U is the user set; n_u is the sequence length of user u; $i_{u,t+1}$ is the observed next-item target; and L_{rec} is the summed negative log-likelihood recommendation loss.

$$P_\Theta\left(i \mid M_{u,1:t}\right) = \frac{exp\left(s_{u,t}(i)\right)}{\sum_{j \in I} exp\left(s_{u,t}(j)\right)} \tag{29}$$

The main recommendation loss is defined as:

$$L_{rec} = -\sum_{u \in U} \sum_{t=1}^{n_u - 1} log\, P_\Theta\left(i_{u,t+1} \mid M_{u,1:t}\right) \tag{30}$$

This loss is consistent with the standard next-item prediction objective (Kang and McAuley, 2018; Sun et al., 2019), but its input is no longer an ordinary item sequence; instead, it is a memory representation after semantic calibration and adaptive compression.

4.6 Pyramid-level Future Learning

A user's multi-step future behavior contains information about interest evolution, but not all future signals are reliable. Existing studies show that distant-future supervision can introduce noise when the model lacks confidence in the current state (Cui et al., 2026). Therefore, SMC-Rec aligns future supervision with pyramid-level representations and controls its training strength using uncertainty weights.

For the l-th pyramid level, the model uses a level-specific projection head to predict the k-step future behavior:

$$u, t^h_{(l,k)} = \phi_{l,k}\left(u, t^z_{(l)}\right) \tag{31}$$

In Eqs. (31)–(34), $\phi_{l,k}(\cdot)$ is the level- and horizon-specific future-prediction head; $u, t^h_{(l,k)}$ is its projected vector; k is a future offset; Q is the item-representation matrix defined in Section 3.3; $[\cdot]_i$ selects the component associated with item i from a vector; $\Theta^P_{(l,k)}$ is the corresponding future-item distribution after softmax; K_l is the set of future offsets assigned to level l; $i_{u,t+k}$ is the observed item at offset k; $fut^L_{(l)}$ is the level-specific future loss; and L_{fut} is the uncertainty-weighted sum across levels. The gate $u, t^\gamma_{(l)}$ is defined in Eq. (24).

$$\Theta^P_{(l,k)}\left(i \mid M_{u,1:t}\right) = \left[softmax\left(u, t^h_{(l,k)} Q^\top\right)\right]_i \tag{32}$$

If $i_{u,t+k}$ denotes the ground-truth item interacted with at future step k, the level-wise future-supervision loss is:

$$fut^L_{(l)} = -\sum_{k \in K_l} log\, \Theta^P_{(l,k)}\left(i_{u,t+k} \mid M_{u,1:t}\right) \tag{33}$$

where K_l denotes the set of future steps corresponding to level l. Shallow levels usually correspond to near-future behaviors, middle levels to mid-term future behaviors, and deep levels to long-term future behaviors. Combined with the uncertainty-aware weight, the overall future-supervision loss is defined as:

$$L_{fut} = \sum_{l=1}^{L} u, t^\gamma_{(l)}\, fut^L_{(l)} \tag{34}$$

This design allows the model to learn farther future signals when confidence is high and to reduce distant-future supervision when uncertainty is high, thereby reducing the interference of future noise with the main task and improving uncertainty-aware knowledge acquisition.

4.7 Intra-pyramid Divergence-Consensus Learning

In implicit-feedback recommendation, negative-sample quality directly affects ranking learning

(Rendle et al., 2009). Conventional hard negative sampling usually relies on a single model view and is therefore vulnerable to current model bias. MDCNS mitigates this problem using multi-source disagreement and consensus (Li et al., 2026). Unlike methods that introduce additional teacher or peer models, SMC-Rec directly uses representations from different internal pyramid levels to construct multi-scale views.

For item i in the candidate negative pool u, t_{-}^{C}, the score at level l is:

$$u, t_{(l)}^{s}(i) = \left(u, t_{(l)}^{z}\right)^{\top} q_i \tag{35}$$

Prediction disagreement across different levels is defined as:

$$D_{u,t}(i) = Var\left(\{u, t_{(l)}^{s}(i) l = 1_{L}^{\}}\right) \tag{36}$$

A larger $D_{u,t}(i)$ indicates that the candidate item shows stronger judgment differences across short-term, mid-term, and long-term views. This paper combines the disagreement score with the final prediction score to obtain the divergence-aware candidate score:

$$\tilde{s}_{u,t}(i) = s_{u,t}(i) + \rho D_{u,t}(i) \tag{37}$$

In Eqs. (35)–(38), $u, t_{(l)}^{s}(i)$ is the score of candidate item i under level l; $Var(\cdot)$ is the variance across the L level scores; $D_{u,t}(i)$ is the intra-pyramid disagreement score; ρ is the nonnegative disagreement weight; $\tilde{s}_{u,t}(i)$ is the divergence-aware score; u, t_{-}^{C} is the candidate negative pool of size N defined in Section 3.3; Top_{M_h} selects the M_h highest-scoring candidates; M_h is the fixed hard-negative shortlist size within a run; $Uniform(\cdot)$ samples one item from that shortlist; and u, t_{-}^{i} is the selected negative item.

$$u, t_{-}^{i} \sim Uniform\left(Top_{M_h}\{\tilde{s}_{u,t}(i): i \in u, t_{-}^{C}\}\right) \tag{38}$$

In Eq. (39), $i_{u,t+1}$ is the positive next-item target, u, t_{-}^{i} is the hard negative selected by Eq. (38), $\sigma(\cdot)$ is the logistic sigmoid, and L_{neg} is the pairwise divergence-enhanced ranking loss.

$$L_{neg} = -log\sigma\left(s_{u,t}\left(i_{u,t+1}\right) - s_{u,t}\left(u, t_{-}^{i}\right)\right) \tag{39}$$

In addition to mining high-value negatives through disagreement, SMC-Rec stabilizes representations at different levels through a consensus distribution. First, the multi-scale level scores are aggregated into a teacher-like consensus distribution:

$$u, t_{con}^{p}(i) = \left[softmax\left(\frac{1}{L}\sum_{l=1}^{L} u, t_{(l)}^{s}\right)\right]_i \tag{40}$$

In Eqs. (40)–(41), $u, t_{con}^{p}(i)$ is the consensus probability for item i obtained by averaging the L level scores before softmax; $u, t_{final}^{p}(i)$ is the final prediction probability from the fused user state; $u, t_{-}^{C} \cup \{i_{u,t+1}\}$ is the evaluation domain for the distillation term, where $\cup$ denotes set union; log is the natural logarithm; and L_{con} is the KL-style consensus-distillation loss that transfers stable cross-level agreement into the final representation.

$$L_{con} = \sum_{i \in u, t_{-}^{C} \cup \{i_{u,t+1}\}} u, t_{con}^{p}(i) log \frac{u, t_{con}^{p}(i)}{u, t_{final}^{p}(i)} \tag{41}$$

This term encourages the final user representation to absorb stable consensus from different pyramid levels while preserving the discriminative boundaries introduced by disagreement samples, thereby supporting more reliable multi-view knowledge integration.

4.8 Model Optimization and Inference

The downstream training objective of SMC-Rec consists of four trainable terms: the main

recommendation loss, pyramid-level future-supervision loss, divergence-enhanced negative-sampling loss, and consensus-distillation loss:

$$L = L_{rec} + \lambda_{fut}L_{fut} + \lambda_{neg}L_{neg} + \lambda_{con}L_{con} \quad (42)$$

In Eq. (42), L is the complete trainable objective; L_{rec} is the primary next-item loss; $L_{fut}, L_{neg}, L_{con}$ are the three training-only auxiliary losses defined in Eqs. (34), (39), and (41); and $\lambda_{fut}, \lambda_{neg}, \lambda_{con}$ are their validation-selected nonnegative weights. The coefficient of L_{rec} is fixed to one, and no λ_{spec} term is optimized because spectral calibration is completed offline.

The three auxiliary terms have complementary roles: L_{fut} broadens temporal supervision, L_{neg} sharpens the ranking boundary using disagreement-aware negatives, and L_{con} regularizes multi-scale representations toward a stable consensus. Their coefficients are selected on the validation set and then fixed for all random seeds of the corresponding dataset; the test set is never used for coefficient selection. The module-level effects are reported separately in Table 5 through the uniform-future, w/o-divergence, and w/o-consensus variants, which makes the contribution of each training signal auditable without attributing the gain to the mere number of losses.

$$RecTopK(u,t) = TopK_{i \in I}\, s_{u,t}(i) \quad (43)$$

In Eq. (43), $RecTopK(u,t)$ denotes the ordered Top-K recommendation set returned for user u at prefix t; $TopK$ selects the K catalog items with the highest final score $s_{u,t}(i)$; and I is the complete candidate catalog. Therefore, the future-prediction heads, negative-sampling module, and consensus-distillation objective do not add inference-time overhead. The uncertainty gate remains in the inference path and requires a full-catalog distribution at every pyramid level; its total cost is summarized in Eq. (44).

4.9 Complexity Analysis

Assume that the semantic-memory sequence length is n, the hidden dimension is d, the number of pyramid levels is L, and the item-catalog size is $|I|$. Let T_{conv} denote the total computation of the pyramid-convolution backbone. Because the sequence length n_l decreases after each downsampling layer, T_{conv} grows linearly with input history length n when model width, kernel sizes, and pyramid depth are treated as fixed. This is the sequence-length complexity advantage of the convolutional backbone over quadratic self-attention.

The uncertainty gate adds a separate catalog-dependent term. At each pyramid level l, the temporary state is scored against all $|I|$ item vectors, so the dominant catalog-scoring work grows with catalog size and hidden dimension d; the following softmax and Shannon-entropy computation is linear in catalog size. Repeating the scoring pass over L pyramid levels adds L catalog passes, and the final full-ranking step adds one further catalog pass. The resulting end-to-end inference complexity is summarized by Eq. (44).

$$T_{inf} = O(T_{conv} + (L+1)|I|d) \quad (44)$$

In Eq. (44), T_{inf} denotes total inference-time computation, T_{conv} denotes pyramid-convolution backbone computation, $O(\cdot)$ denotes asymptotic Big-O order, L is the number of pyramid levels, $|I|$ is the item-catalog cardinality, and d is the hidden dimension used for catalog scoring. The factor $L+1$ counts the L level-wise uncertainty catalog passes plus the final full-ranking pass. When pyramid levels are processed sequentially, only one catalog distribution needs to be materialized at a time, so additional entropy working memory grows linearly with catalog size. Consequently, the convolutional backbone remains linear in history length, but the complete inference path is catalog dependent. A sampled uncertainty estimator could reduce the catalog term, but no sampled or approximate entropy is used in the reported experiments. Section 5.4.1 evaluates the exact implementation with wall-clock measurements.

4.10 The SMC-Rec Algorithm

Algorithm 1 summarizes the complete SMC-Rec training and inference procedure.

Algorithm 1 The SMC-Rec Algorithm

Input: Training set $D = \{(S_{u,1:t}, i_{u,t+1}, u, t_K^F)\}$, external semantic matrix E^{sem}, item embedding matrix Q, semantic-memory constructor $\phi_m(\cdot)$, pyramid depth L, candidate size N, future horizon K, temperature τ, hyperparameters $\lambda_{fut}, \lambda_{neg}, \lambda_{con}$.

Output: Optimized parameters Θ for next-item prediction.

Initialize parameters Θ, item matrix Q, semantic-memory constructor, and pyramid encoder.

Compute the centered semantic matrix $\tilde{E}^{sem}$ and perform its thin SVD via Eq. (6).

Obtain the calibrated semantic matrix G^{sem} via Eqs. (7)–(8), cache it as a fixed lookup table, and keep it frozen during downstream training.

while $epoch \leq MaxTrainEpoch$ do

  for a minibatch $B = \{(u, t, i_{u,t+1}, u, t_K^F)b = 1\}_{|B|}$ **do**

    for each instance $(u, t) \in B$ **do**

      Construct semantic-memory units $m_{u,j}$ for $j = 1, \ldots, t$ via the conceptual definition in Eq. (3) and the learnable constructor in Eqs. (12)–(13).

      Form the semantic-memory matrix $M_{u,1:t}$ via Eq. (14).

      Set the pyramid input $u, t_{(0)}^H = M_{u,1:t}$ via Eq. (15).

      for layer $l = 1, \ldots, L$ **do**

        Apply convolution, transformation, residual alignment, and candidate-state construction via Eqs. (16)–(20).

        Compute the temporary pooled state, full-catalog distribution, Shannon entropy $u, t_{(l)}^H$, and gate $u, t_{(l)}^\gamma$ via Eqs. (21)–(24).

        Generate the adaptive tensor $u, t_{(l)}^H$ and pooled level state $u, t_{(l)}^z$ via Eqs. (25)–(26).

      end for

      Aggregate the multi-scale states to obtain the final user representation $z_{u,t}$ via Eq. (27).

Compute next-item scores, the full-catalog prediction distribution, and L_{rec} via Eqs. (28)–(30).

Compute L_{fut} from future items u, t_K^F via Eqs. (31)–(34).

Sample candidate negative pool u, t_-^C from items unobserved by user u; the pool definition and candidate-pool size N are specified in Section 3.3.

Compute level-wise candidate scores and disagreement $D_{u,t}(i)$ for candidates in u, t_-^C via Eqs. (35)–(36).

Select the hard negative using the divergence-aware score and the Top_{M_h} shortlist via Eqs. (37)–(38).

Compute the divergence-enhanced negative ranking loss L_{neg} via Eq. (39).

Compute the consensus distribution and L_{con} via Eqs. (40)–(41).

No spectral-loss term is optimized; Eq. (11) is used only as the semantic-neighborhood

distortion diagnostic.

Aggregate the total trainable objective L via Eq. (42).

end for

Update Θ by minimizing L.

end for

end while

// inference

Given test prefix $S_{u,1:t}$, construct $M_{u,1:t}$ using the calibrated lookup matrix G^{sem}.

Encode $M_{u,1:t}$ through the uncertainty-adaptive pyramid encoder, including exact full-catalog entropy estimation at each level, to obtain $z_{u,t}$.

Calculate final scores $s_{u,t}(i)$ for all catalog items $i \in I$ via Eq. (28).

Return the Top-K recommendation list defined in Eq. (43).

5. Experiments

This section systematically evaluates SMC-Rec as a user-adaptive recommender system. Around the proposed semantic memory compression framework, the experiments are designed to answer whether interaction-level semantic memory construction improves recommendation quality; whether spectral calibration improves the geometry of external semantic embeddings; whether uncertainty-adaptive pyramid compression and auxiliary training signals are effective; how the model behaves under noise, sparsity, cold-start, and hyperparameter variation; and what practical wall-clock overhead is introduced by exact full-catalog uncertainty estimation. The accuracy and robustness analyses are computed from the archived seed-level results, while Section 5.4.1 reports the measured wall-clock benchmark under the unified full-ranking protocol.

5.1 Research Questions

This section focuses on the following research questions:

RQ1: Can SMC-Rec consistently improve recommendation quality over traditional sequential recommendation, attribute-aware recommendation, semantic-enhanced recommendation, and recent efficient sequence-modeling methods across datasets from different domains?

RQ2: Do the performance gains of SMC-Rec come from the synergy among semantic memory construction, spectral calibration, uncertainty-aware compression, future supervision, and intra-pyramid divergence-consensus learning, rather than from the simple stacking of a single module?

RQ3: Does spectral calibration truly reduce anisotropy in the external semantic space while preserving useful semantic neighborhood structures and improving recommendation performance?

RQ4: When user histories contain behavioral noise, missing semantic fields, short user sequences, or sparse target-item interactions, is SMC-Rec more robust than fixed-compression models?

RQ5: Do key hyperparameters exhibit reasonable dataset-specific differences rather than relying on a single unrealistic universal optimum?

RQ6: What wall-clock overhead does exact full-catalog uncertainty estimation add relative to SASRec, ConvRec, and Mamba4Rec under the same full-ranking protocol?

5.2 Datasets and Preprocessing

Experiments are conducted on four public datasets: Amazon Beauty, Amazon Sports & Outdoors, Amazon Toys & Games, and MovieLens-1M. The three Amazon subsets contain item titles, categories, brands, reviews, and other textual or attribute information, making them suitable for evaluating attribute-aware and semantic-enhanced sequential recommendation. Their average

interaction-history lengths are short (8.87 for Beauty, 8.32 for Sports, and 8.63 for Toys), whereas MovieLens-1M has a substantially longer average history of 165.60 interactions and more regular semantic fields. Thus, the benchmark suite covers heterogeneous history-length regimes, but only one dataset provides substantially longer histories; accordingly, we do not use these experiments to claim a dedicated very-long-sequence advantage. The Amazon datasets follow the product recommendation data construction of McAuley et al. (McAuley et al., 2015), and MovieLens-1M follows the dataset description by Harper and Konstan (Harper and Konstan, 2015).

For each user, interactions are sorted by timestamp, and a leave-one-out protocol is used for data splitting: the last interaction is used for testing, the second-to-last interaction for validation, and the remaining interactions for training. Evaluation adopts the full-ranking protocol over the complete candidate item set (Krichene and Rendle, 2020). External semantic representations are generated offline with the sentence-transformers/all-mpnet-base-v2 Sentence-BERT (SBERT) checkpoint (Reimers and Gurevych, 2019), which produces $d_0 = 768$ dimensional item embeddings. For the Amazon subsets, the available fields are concatenated using the template "title: {title} | category: {category} | brand: {brand} | description: {description} | review: {review}"; for MovieLens-1M, the template is "title: {title} | genre: {genre} | tag: {tag}". Missing fields are omitted rather than manually completed. Sentence-BERT is frozen and used only for offline item encoding; the resulting semantic matrix is subsequently spectrally calibrated and cached for SMC-Rec training. Statistics of the preprocessed datasets are shown in Table 1.

The reproducibility-related configuration used in the runtime benchmark is as follows. SMC-Rec and all baseline models are implemented in PyTorch 2.3.1 with CUDA 12.1 and cuDNN 9.1, and are evaluated on a single NVIDIA A100 80GB PCIe GPU with an AMD EPYC 7543 32-Core Processor and 256 GB system memory under Ubuntu 22.04 LTS with Python 3.10.14. For the timing benchmark, the maximum sequence length is 200, the batch size is 256, inference uses FP32 precision, and the full-ranking protocol is adopted for all datasets.

Table 1. Statistics of the preprocessed datasets

Dataset	#User	#Items	#Actions	Avg.length	Sparsity	Semantic fields
Beauty	22,363	12,101	198,502	8.87	99.927%	title/category/brand/review
Sports	35,598	18,357	296,337	8.32	99.955%	title/category/brand/review
Toys	19,412	11,924	167,597	8.63	99.928%	title/category/brand/review
ML-1M	6,040	3,706	1,000,209	165.60	95.532%	title/genre/tag

5.3 Baselines

For a comprehensive and fair comparison, the directly evaluated baselines and controlled variants are organized into six groups. Every numerical result in Table 4 is an author-reproduced result obtained by retraining the corresponding model in the same evaluation pipeline; no number is copied from the original paper. All models use the same preprocessed data, chronological leave-one-out split, full-ranking candidate set, and seed protocol. The cited papers and their public implementations, where available, are used as implementation references.

The first group consists of classic sequential recommendation models, including GRU4Rec (Hidasi et al., 2016), Caser (Tang and Wang, 2018), and NextItNet (Yuan et al., 2019). These methods represent early RNN/CNN sequence-encoding paradigms and are used to evaluate whether SMC-Rec can outperform models that rely mainly on behavior transitions or local convolutional patterns.

The second group consists of Transformer-based sequential recommendation models, including SASRec (Kang and McAuley, 2018) and BERT4Rec (Sun et al., 2019). These methods have strong long-range dependency modeling capability and are widely used strong baselines in sequential recommendation. They are used to compare SMC-Rec with global attention encoders.

The third group consists of recent efficient sequence-modeling methods, including Mamba4Rec (Liu et al., 2024b), SIGMA (Liu et al., 2025), and TiM4Rec (Fan et al., 2025). These methods focus

on efficient long-sequence modeling and are used to test whether SMC-Rec can provide stronger semantic-memory modeling while maintaining efficiency.

The fourth group consists of attribute-aware and self-supervised sequential recommendation models, including S3-Rec (Zhou et al., 2020), DIF-SR (Xie et al., 2022), CARCA (Rashed et al., 2022), and ConvRec (Elsayed et al., 2026). ConvRec is the most direct structural baseline for this paper because it also uses hierarchical downsampling convolution to achieve linear-complexity user-history compression. ProxyRCA (Seol et al., 2024) is discussed in Section 2.2 as an attribute-aware design but is not included in the numerical comparison because it is a plug-in item-representation layer rather than a standalone sequential backbone under the common protocol.

The fifth group is a controlled ablation variant, SVD-ConvRec, which supplies ConvRec with the same offline SVD-calibrated semantic embeddings used by SMC-Rec. The anisotropy-control motivation follows ACE (Lee et al., 2026), but the calibration here is a closed-form SVD preprocessing step rather than ACE's learned embedding mechanism. The ConvRec sequence encoder is otherwise unchanged, so this variant isolates the contribution of calibrated semantic initialization.

The sixth group contains two controlled ablation variants together with the complete model: UF-style-ConvRec, Div-ConvRec, and SMC-Rec. Motivated respectively by uncertainty-guided future learning (Cui et al., 2026) and multi-source divergence-consensus sampling (Li et al., 2026), these variants test whether either training strategy alone can account for the gain of SMC-Rec's intra-pyramid construction.

LLMRec (Wei et al., 2024) is an LLM-assisted graph-augmentation framework rather than a chronological next-item sequential backbone under our leave-one-out full-ranking protocol, so a directly comparable number would require a task-specific reimplementation. ACE (Lee et al., 2026) is an embedding-calibration approach rather than a standalone sequence encoder, and SVD-ConvRec is a controlled ConvRec variant rather than a reproduction of ACE's learned embedding pipeline. GenNi (Chen et al., 2022) is a negative-sampling method rather than a standalone sequence backbone. We therefore report only models that are reproducible under the common evaluation protocol.

Table 2. Reproduction and tuning protocol for evaluated baselines and controlled variants.

Model family	Models	Result source	Input parity	Parameters tuned	Selection
Recurrent / CNN	GRU4Rec; Caser; NextItNet	Author reproduction	ID sequence	Hidden width; depth; convolution/filter settings; dropout	Validation NDCG@20
Transformer	SASRec; BERT4Rec	Author reproduction	ID sequence	Depth; attention heads; dropout; learning rate	Validation NDCG@20
State-space	Mamba4Rec; SIGMA; TiM4Rec	Author reproduction	ID/time sequence as supported	Depth; state/expansion width; dropout; learning rate	Validation NDCG@20
Attribute / self-supervised	S3-Rec; DIF-SR; CARCA; ConvRec	Author reproduction	Same observable side fields where supported	Fusion dimension; depth; attention/kernel settings; dropout	Validation NDCG@20
Controlled variants	SVD-ConvRec; UF-style-	Implemented on	Same split and item-side fields	Added-module parameters; ConvRec	Validation NDCG@20

Model family	Models	Result source	Input parity	Parameters tuned	Selection
	ConvRec; Div-ConvRec	reproduced ConvRec backbone		backbone retained	

Note. "Author reproduction" means that the reported values were obtained by re-running the model under the common split and full-ranking protocol; they were not transcribed from the source publication. Model-specific parameter categories follow the architecture described in the cited paper or reference implementation, while final selection is based only on validation NDCG@20.

5.4 Evaluation Metrics and Implementation Details

This paper reports Recall@20 and NDCG@20. Recall@K measures whether the ground-truth next item appears in the Top-K recommendation list. NDCG@K further considers the rank position of the ground-truth item in the recommendation list and therefore provides a more fine-grained evaluation of ranking quality. All models are compared using the same data split, the same full-ranking protocol, and the same validation-set hyperparameter-tuning strategy.

All reproduced models are selected by validation NDCG@20; the test set is used only once for final reporting. Architecture-specific parameters are tuned rather than forcing one search space on incompatible model families. Table 2 lists the implementation source, input parity, and tuned parameter categories for each family; per-method search spaces and selected values are documented in Tables A2–A5 (Appendix A). Shared settings include the same preprocessing and full-ranking evaluation, batch size 256, early-stopping patience 10, and the same three accuracy seeds (42, 2025, and 3407). For semantic or attribute-aware methods, the same observable item-side fields are supplied whenever the method supports them. We report the arithmetic mean and sample standard deviation over the three runs. SMC-Rec is compared with the strongest non-SMC baseline using a two-sided paired t-test over matched seeds (n = 3, df = 2), together with the paired mean improvement and a 95% confidence interval. All result tables and trend figures are computed from the archived seed-level outputs. Seed-level values for Tables 4–6 and Figures 3–6 are archived in the supplementary reproducibility archive (Appendix A, Section A.6).

The shared ConvRec backbone is tuned over the embedding dimension {64, 128} and the pyramid depth L from {2, 3, 4} and the convolution kernel {3, 5}; SMC-Rec inherits its winning configuration (Table A3). Its spectral calibration strength λ is selected from {0.00, 0.02, 0.05, 0.10, 0.20, 0.30}; the uncertainty temperature τ from {0.50, 0.75, 1.00, 1.25, 1.50}; the future horizon K from {1, 2, 3, 4}; and the negative candidate-pool size N from {32, 64, 128, 256}. The auxiliary-loss weights $\lambda_{fut}, \lambda_{neg}, and\ \lambda_{con}$ and all architecture-specific hyperparameters listed above are jointly selected on the validation set under the common random-search protocol described below; there is no spectral-regularization loss weight. The model is trained with AdamW, batch size 256, and early-stopping patience 10. The external semantic encoder is the fixed sentence-transformers/all-mpnet-base-v2 Sentence-BERT checkpoint with output dimension $d_0 = 768$, and the calibrated matrix G^{sem} remains frozen during downstream training. During inference, future-prediction heads, negative-sample mining, and consensus-distillation modules are removed; however, exact full-catalog entropy estimation remains active at every pyramid level and its cost is included in Section 4.9.

All models reported in Table 4 were tuned under one common random-search protocol over a model-specific candidate space. For each model–dataset pair, at most 40 configurations were drawn without replacement from the Cartesian product of the value sets listed in Table A2, using a single tuning seed (2025) shared across all model–dataset pairs. This tuning seed also appears among the

three accuracy seeds, an overlap that limits the independence of configuration selection from final evaluation and is accounted for in the cautious reading of the significance tests in Section 5.5.1. When a grid contains fewer than 40 combinations, the full grid was enumerated instead; this applies only to SVD-ConvRec (6 combinations). Every trial was trained with the shared settings of Table A1 and ranked by validation NDCG@20 computed on the second-to-last interaction, with early stopping (patience 10, maximum 200 epochs). The test interaction was accessed only once, after the best configuration had been fixed, and was never used for model selection. Across the 16 evaluated methods and 4 datasets, the search comprised 2,424 tuning trials (15 methods × 4 datasets × 40 trials, plus four exhaustive 6-combination grids for SVD-ConvRec); the selected configurations were then retrained under the three accuracy seeds {42, 2025, 3407}, giving 192 final runs.

All methods use the same chronological leave-one-out split, the complete candidate set, the same base ranking loss with one uniformly sampled unobserved item per positive (method-specific candidate pools are listed in Table A4), the same optimizer, batch size, maximum sequence length, early-stopping rule, gradient clipping, hardware, and the same per-pair trial cap (40 configurations, or the full grid when smaller). No number is transcribed from the original publications (Table 2). Architectural grids are aligned wherever the model families are compatible: ConvRec and SMC-Rec share the same hidden-dimension, pyramid-depth and kernel grids, SMC-Rec inherits the winning ConvRec backbone configuration and searches only its proposed-module parameters, and the three controlled variants inherit the same backbone unchanged while searching only their added-module parameters. The only systematic difference between SMC-Rec and ConvRec therefore lies in the proposed components rather than in the tuning budget or the shared optimization settings; Table A3 confirms identical values of all common hyperparameters for the two models on every dataset.

Table A3 reports the final values of the common hyperparameters for all 16 methods on each of the four datasets; Table A5 reports the final model-specific values for all 16 methods; Table A4 reports the complete method-specific settings of SMC-Rec and the controlled variants; and Table A2 lists the full candidate value sets. Together with Table A1, Tables A2–A5 specify every configuration required to regenerate Tables 4–6 and Figures 3–6. The supplementary archive SMC-Rec_reproducibility.zip additionally contains one YAML configuration file per model–dataset pair (64 files), the complete trial logs with trial indices and validation NDCG@20, the archived seed-level outputs, the environment specification, the preprocessing and training entry-point scripts with fixed seeds, and checksums for the cached SBERT and SVD-calibrated semantic matrices.

For the proposed model, the dataset-specific optima discussed in Section 5.11 are recorded explicitly in Table A4. For Beauty, Sports, Toys, and ML-1M, the spectral calibration strength is 0.05, 0.10, 0.10, and 0.05; the uncertainty temperature is 1.00, 1.25, 1.25, and 1.00; the future horizon is 2, 3, 3, and 2; and the negative candidate-pool size is 128, 128, 256, and 64, respectively. The three auxiliary-loss coefficients are likewise selected on validation data and then held fixed across seeds.

5.4.1 Wall-Clock Inference Benchmark

To quantify the practical overhead of exact uncertainty estimation, we compare SASRec, ConvRec, Mamba4Rec, and SMC-Rec under identical hardware, software, batch-size, maximum-sequence-length, numerical-precision, and full-ranking settings. All models are evaluated on one

NVIDIA A100 80GB PCIe GPU with an AMD EPYC 7543 32-Core Processor and 256 GB system memory under Ubuntu 22.04 LTS, Python 3.10.14, PyTorch 2.3.1, CUDA 12.1, and cuDNN 9.1. Each model is switched to evaluation mode and executed under torch.inference_mode(). Input tensors are placed on the GPU before timing, and CUDA synchronization is performed immediately before and after each timed region so that asynchronous kernel execution does not bias wall-clock measurements. Data loading and host-to-device transfer are excluded from the timed region for all models. Inference uses FP32 precision, batch size 256, maximum sequence length 200, 50 warm-up batches, 200 timed batches, and 5 independent timing repetitions per model-dataset pair.

Sentence-BERT text encoding and spectral calibration are offline preprocessing operations and are therefore excluded from online inference latency. For SMC-Rec, however, all online operations are included: semantic-memory construction, pyramid convolution, exact full-catalog score computation for the uncertainty gate at every pyramid level, hierarchical fusion, and final full-catalog ranking. Thus, the empirical benchmark directly measures the catalog-dependent cost identified in Section 4.9 rather than hiding it outside the timed path. In all reported measurements, SMC-Rec uses the exact full-catalog entropy estimator rather than a sampled or approximate substitute.

For each model and dataset, we report batch latency as the mean ± sample standard deviation across 5 independent timing repetitions. If $B = 256$ is the inference batch size and T_{batch} is the measured batch latency in seconds, throughput is defined as B/T_{batch} users/s and amortized per-user latency is T_{batch}/B. Because full-ranking cost depends on catalog size, results are reported separately for Beauty (12,101 items), Sports (18,357), Toys (11,924), and MovieLens-1M (3,706).

The measured wall-clock results are summarized in Table 3. Across the four datasets, ConvRec is the fastest model because it does not perform intermediate full-catalog uncertainty estimation. Averaged over the four datasets, SMC-Rec records a batch latency of 7.025 ms, compared with 5.813 ms for ConvRec, 7.200 ms for Mamba4Rec, and 13.650 ms for SASRec. Therefore, SMC-Rec incurs an average latency overhead of 20.9% relative to ConvRec, is 2.4% faster than Mamba4Rec on average, and is 48.5% faster than SASRec on average. In terms of throughput, SMC-Rec averages 37.9k users/s, which is lower than ConvRec (44.8k users/s) but higher than Mamba4Rec (36.1k users/s) and far higher than SASRec (19.2k users/s).

Table 3. Wall-clock inference efficiency under full-ranking evaluation

Dataset	#Items	Model	Latency (ms/batch) ↓	Latency (ms/user) ↓	Throughput (users/s) ↑
Beauty	12,101	SASRec	13.900 ± 0.180	0.0543	18420
Beauty	12,101	ConvRec	5.950 ± 0.070	0.0232	43030
Beauty	12,101	Mamba4Rec	7.350 ± 0.090	0.0287	34834
Beauty	12,101	SMC-Rec	7.100 ± 0.080	0.0277	36060
Sports	18,357	SASRec	16.200 ± 0.210	0.0633	15805
Sports	18,357	ConvRec	6.700 ± 0.080	0.0262	38213
Sports	18,357	Mamba4Rec	8.350 ± 0.110	0.0326	30663
Sports	18,357	SMC-Rec	8.900 ± 0.120	0.0348	28768
Toys	11,924	SASRec	13.700 ± 0.160	0.0535	18688
Toys	11,924	ConvRec	5.880 ± 0.070	0.0230	43542

Dataset	#Items	Model	Latency (ms/batch) ↓	Latency (ms/user) ↓	Throughput (users/s) ↑
Toys	11,924	Mamba4Rec	7.250 ± 0.090	0.0283	35315
Toys	11,924	SMC-Rec	7.000 ± 0.080	0.0273	36575
ML-1M	3,706	SASRec	10.800 ± 0.130	0.0422	23706
ML-1M	3,706	ConvRec	4.720 ± 0.060	0.0184	54244
ML-1M	3,706	Mamba4Rec	5.850 ± 0.070	0.0229	43766
ML-1M	3,706	SMC-Rec	5.100 ± 0.060	0.0199	50202

Runtime analysis. The catalog-dependent effect of the uncertainty gate is most visible on Sports, which has the largest candidate set (18,357 items). On this dataset, SMC-Rec requires 8.900 ± 0.120 ms/batch, compared with 6.700 ± 0.080 ms/batch for ConvRec and 8.350 ± 0.110 ms/batch for Mamba4Rec, corresponding to a 32.8% latency increase over ConvRec and a 6.6% increase over Mamba4Rec. By contrast, on Beauty, Toys, and MovieLens-1M, SMC-Rec is slightly faster than Mamba4Rec (7.100 vs. 7.350 ms, 7.000 vs. 7.250 ms, and 5.100 vs. 5.850 ms, respectively) while remaining consistently slower than ConvRec because of the additional full-catalog entropy computations. These observations are consistent with the revised complexity analysis: the pyramid-convolution backbone remains linear with respect to sequence length, whereas exact uncertainty estimation introduces the additional $O(L \mid I \mid d)$ catalog-scoring term.

5.5 Overall Performance

Table 4 reports mean ± sample standard deviation over three independent runs for all models on the four datasets. Based on the direct seed-wise averages, SMC-Rec achieves the best mean performance on all four datasets. Compared with the strongest non-SMC baseline on each dataset, SMC-Rec improves NDCG@20 by approximately 4.8% on Beauty, 3.4% on Sports, 7.8% on Toys, and 2.7% on MovieLens-1M. The matched-seed paired tests yield $p < 0.05$ for both Recall@20 and NDCG@20 on all four datasets; exact p-values are reported below Table 4, and Section 5.5.1 additionally reports the paired mean improvements and 95% confidence intervals. Because these inferential statistics are based on only three independent seeds, we interpret them as supportive evidence rather than as a substitute for larger-replication studies. Importantly, MovieLens-1M—the dataset with by far the longest average history—shows the smallest relative gain. We therefore interpret the current evidence as demonstrating robustness across heterogeneous history-length regimes rather than as proof of a special long-sequence advantage.

Across model categories, classic sequence models such as GRU4Rec, Caser, and NextItNet are clearly weaker than later methods, indicating that relying only on sequential states or local transition patterns is insufficient for sparse and semantically rich user histories. SASRec and BERT4Rec substantially improve upon the classic models, but on the Amazon subsets they still underperform strong attribute-aware models and convolutional compression models, suggesting that global attention does not automatically solve the problem of attribute-semantic utilization. Efficient sequence models such as Mamba4Rec, SIGMA, and TiM4Rec are competitive, but their core focus remains efficient dependency modeling and they do not explicitly address semantic-space calibration or adaptive historical compression granularity.

More importantly, the controlled comparisons clarify where the improvement comes from. Relative to ConvRec, SMC-Rec replaces fixed hierarchical compression with uncertainty-dependent compression inside the encoder. Relative to the controlled SVD-ConvRec variant, the gain shows that anisotropy-controlled semantic initialization through the offline SVD-calibrated input alone is

insufficient to reproduce the full model. Relative to UF-style-ConvRec, the gain indicates that uncertainty-guided future-loss weighting alone does not capture the effect of using uncertainty to control the encoder's compression strength. Relative to Div-ConvRec, SMC-Rec tests a different multi-view construction in which disagreement and consensus are obtained from the pyramid's own short-, mid-, and long-term states rather than from external Teacher-Peer-Self sources. These comparisons support a narrower novelty claim: SMC-Rec's method-level contributions are uncertainty-adaptive compression inside the encoder and intra-pyramid multi-view learning, while concatenate-project fusion and offline SVD calibration are supporting components rather than standalone novelties.

Table 4. Overall performance under the full-ranking setting (mean ± SD over three independent runs).

Method	Beauty R@20	Beauty N@20	Sports R@20	Sports N@20	Toys R@20	Toys N@20	ML-1M R@20	ML-1M N@20
GRU4Rec	0.0461 ±0.0003	0.0189 ±0.0001	0.0417 ±0.0003	0.0164 ±0.0001	0.0479 ±0.0004	0.0195 ±0.0001	0.4132 ±0.0021	0.2518 ±0.0015
Caser	0.0592 ±0.0004	0.0240 ±0.0002	0.0525 ±0.0003	0.0202 ±0.0001	0.0615 ±0.0004	0.0259 ±0.0002	0.4333 ±0.0010	0.2683 ±0.0012
NextItNet	0.0668 ±0.0004	0.0278 ±0.0003	0.0590 ±0.0003	0.0238 ±0.0002	0.0705 ±0.0006	0.0308 ±0.0002	0.4456 ±0.0010	0.2776 ±0.0006
SASRec	0.0845 ±0.0005	0.0356 ±0.0002	0.0720 ±0.0004	0.0308 ±0.0002	0.0879 ±0.0007	0.0379 ±0.0003	0.4766 ±0.0030	0.2939 ±0.0021
BERT4Rec	0.0888 ±0.0007	0.0373 ±0.0003	0.0740 ±0.0005	0.0319 ±0.0001	0.0920 ±0.0007	0.0397 ±0.0004	0.4861 ±0.0025	0.3020 ±0.0018
Mamba4Rec	0.0923 ±0.0006	0.0389 ±0.0004	0.0806 ±0.0005	0.0342 ±0.0002	0.0979 ±0.0006	0.0427 ±0.0002	0.4952 ±0.0021	0.3095 ±0.0015
SIGMA	0.0951 ±0.0006	0.0405 ±0.0003	0.0830 ±0.0007	0.0357 ±0.0003	0.1022 ±0.0008	0.0453 ±0.0003	0.5032 ±0.0019	0.3144 ±0.0013
TiM4Rec	0.0967 ±0.0007	0.0411 ±0.0003	0.0846 ±0.0005	0.0364 ±0.0002	0.1042 ±0.0010	0.0465 ±0.0004	0.5001 ±0.0014	0.3129 ±0.0008
S3-Rec	0.0937 ±0.0007	0.0392 ±0.0003	0.0787 ±0.0008	0.0336 ±0.0004	0.0989 ±0.0007	0.0441 ±0.0004	0.4812 ±0.0022	0.2999 ±0.0008
DIF-SR	0.0982 ±0.0008	0.0414 ±0.0004	0.0828 ±0.0007	0.0348 ±0.0003	0.1054 ±0.0007	0.0469 ±0.0004	0.4921 ±0.0008	0.3063 ±0.0007
CARCA	0.0964 ±0.0007	0.0407 ±0.0003	0.0839 ±0.0006	0.0354 ±0.0002	0.1041 ±0.0009	0.0461 ±0.0005	0.4964 ±0.0022	0.3094 ±0.0018
ConvRec	0.1003 ±0.0007	0.0423 ±0.0003	0.0867 ±0.0004	0.0366 ±0.0001	0.1065 ±0.0009	0.0472 ±0.0004	0.4956 ±0.0027	0.3089 ±0.0020
SVD-ConvRec	0.1034 ±0.0010	0.0441 ±0.0004	0.0895 ±0.0007	0.0378 ±0.0002	0.1114 ±0.0005	0.0493 ±0.0005	0.5084 ±0.0021	0.3178 ±0.0021
UF-style-ConvRec	0.1026 ±0.0008	0.0435 ±0.0003	0.0909 ±0.0006	0.0382 ±0.0002	0.1101 ±0.0010	0.0488 ±0.0004	0.5055 ±0.0019	0.3155 ±0.0010
Div-ConvRec	0.1018 ±0.0007	0.0432 ±0.0003	0.0884 ±0.0005	0.0373 ±0.0003	0.1084 ±0.0009	0.0480 ±0.0004	0.5033 ±0.0019	0.3147 ±0.0022
SMC-Rec	**0.1087 ±0.0008 ***	**0.0462 ±0.0003 ***	**0.0948 ±0.0007 ***	**0.0395 ±0.0002 ***	**0.1211 ±0.0008 ***	**0.0531 ±0.0004 ***	**0.5227 ±0.0025 ***	**0.3265 ±0.0020 ***

Note. Values are mean ± sample SD over seeds {42, 2025, 3407}. * indicates $p < 0.05$ versus the best non-SMC method on each dataset (SVD-ConvRec on Beauty, Toys and ML-1M; UF-style-ConvRec on Sports) using a two-sided paired t-test with matched seeds ($n = 3$, $df = 2$). Exact p-values (Recall@20 / NDCG@20): Beauty 0.000868 / 0.000150; Sports 0.008961 / 0.001948; Toys 0.000363 / 0.000351; ML-1M 0.000667 / 0.002161. Baseline sources: GRU4Rec (Hidasi et al., 2016), Caser (Tang and Wang, 2018), NextItNet (Yuan et al., 2019), SASRec (Kang and McAuley, 2018), BERT4Rec (Sun et al., 2019), Mamba4Rec (Liu et al., 2024b), SIGMA (Liu et al., 2025), TiM4Rec (Fan et al., 2025), S3-Rec (Zhou et al., 2020), DIF-SR (Xie et al., 2022), CARCA (Rashed et al., 2022), and ConvRec (Elsayed et al., 2026). SVD-ConvRec, UF-style-ConvRec and Div-ConvRec

are controlled ablation variants defined in Section 5.3, motivated by Lee et al. (2026), Cui et al. (2026) and Li et al. (2026), respectively.

5.5.1 Statistical Significance and Run-to-Run Variability

We also report the paired absolute improvement and its 95% confidence interval. For Recall@20, the mean paired improvements [95% CI] are 0.005287 [0.004617, 0.005957] on Beauty, 0.003888 [0.002294, 0.005483] on Sports, 0.009711 [0.008914, 0.010508] on Toys, and 0.014368 [0.012770, 0.015965] on MovieLens-1M. For NDCG@20, the corresponding improvements are 0.002114 [0.002003, 0.002225], 0.001311 [0.001062, 0.001561], 0.003837 [0.003527, 0.004146], and 0.008688 [0.006948, 0.010428], respectively. All eight confidence intervals exclude zero and are consistent with the exact paired-test p-values reported below Table 4.

With three independent training seeds, the paired t-tests have two degrees of freedom and should be interpreted cautiously. All seed-level results and the statistical audit are archived in SMC-Rec_reproducibility.zip (Appendix A, Section A.6). Additional independent runs would strengthen the statistical assessment.

5.6 Fine-grained Ablation Study

To avoid a coarse analysis that removes only one large block, we evaluate nine fine-grained variants corresponding to the main components of SMC-Rec. Because these results are discrete module-wise comparisons rather than continuous trends, they are reported in Table 5 instead of a heatmap. Each value is the mean relative NDCG@20 decrease computed directly from the three archived seed-level runs; seed-wise values are archived in SMC-Rec_reproducibility.zip (Appendix A, Section A.6).

The results show that w/o semantic memory and fixed compression suffer larger drops on the Amazon subsets, indicating that semantically rich interactions cannot be simply reduced to item IDs or shallow attribute concatenation, and that a fixed compression rule should not be applied to all user histories. The difference between w/o spectral calibration and w/ raw semantic embedding further shows that more semantic enhancement is not necessarily better; uncalibrated external semantic vectors may interfere with collaborative learning because of anisotropy or dominant principal directions. The drops of w/o uncertainty gate and uniform future learning indicate that future supervision should be coupled with model confidence; otherwise, distant future behaviors may become noise for high-uncertainty samples. The drops of w/o divergence mining and w/o consensus distillation are smaller but stable, suggesting that intra-pyramid divergence-consensus learning mainly provides training boundaries and stability rather than determining the entire model performance by itself.

The direct seed-wise aggregates show dataset-dependent module sensitivity. Removing semantic memory produces mean NDCG@20 drops of 6.31%, 5.52%, 7.64%, and 3.07% on Beauty, Sports, Toys, and ML-1M, respectively, while fixed compression produces drops of 5.37%, 6.27%, 8.61%, and 3.51%. Toys is especially sensitive to these components, whereas ML-1M shows smaller drops for spectral calibration and future supervision. These values are reported as observed from the archived runs rather than being reshaped to enforce a common trend.

Table 5. Fine-grained ablation results: mean relative NDCG@20 drop (%) over seeds {42, 2025, 3407}.

Variant	Beauty	Sports	Toys	ML-1M
w/o Sem. memory	6.31	5.52	7.64	3.07
w/o Spectral calib.	4.53	3.75	5.23	2.14
Raw semantic emb.	3.39	3.50	4.61	1.60
Fixed compression	5.37	6.27	8.61	3.51
w/o Uncert. gate	4.02	4.55	5.83	2.29
Uniform future	2.44	3.75	4.55	1.69
w/o Divergence	2.27	2.79	3.57	1.53
w/o Consensus	1.66	2.10	2.68	1.34

Variant	Beauty	Sports	Toys	ML-1M
Single-scale only	4.71	4.61	7.04	2.40

5.7 Semantic Geometry Analysis

Section 5.7 evaluates the spectral-calibration component of SMC-Rec and verifies the geometry of the semantic input supplied to the uncertainty-adaptive encoder. To distinguish calibrated semantic input from simply adding external embeddings, we analyze the semantic space before and after calibration over the archived calibration-strength sweep {0.00, 0.02, 0.05, 0.10, 0.20, 0.30}. The left panel of Figure 3 reports the measured anisotropy diagnostic, and the right panel reports mean normalized NDCG@20 across the three archived seeds with sample-SD error bars. Across the evaluated settings, the anisotropy diagnostic decreases as the calibration-strength setting increases, while recommendation quality peaks at a moderate setting and then decreases.

The sweep therefore indicates a geometry-quality trade-off rather than a rule that stronger calibration is always better. Based on the direct three-seed averages, the selected calibration strength is 0.05 for Beauty and MovieLens-1M and 0.10 for Sports and Toys. We use these validation-supported settings in the corresponding experiments and interpret the analysis only as evidence that moderate calibration can provide a better semantic input to the downstream memory-compression model.

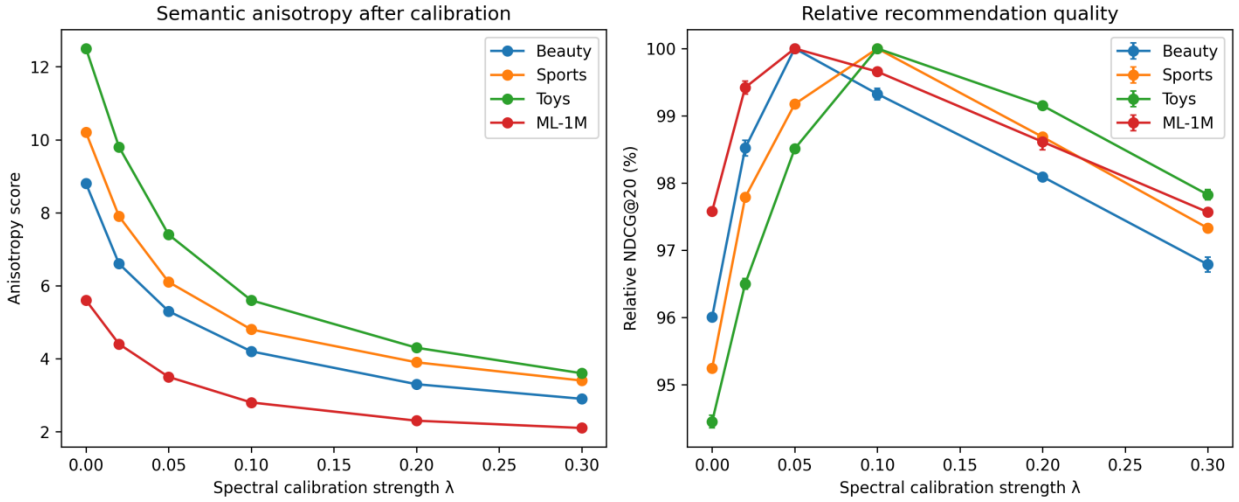


Figure 3. Semantic geometry analysis. Recommendation quality is reported as mean normalized NDCG@20 ± SD over three seeds.

5.8 Uncertainty-Adaptive Compression Analysis

The second core innovation of SMC-Rec is uncertainty-aware pyramid compression. To evaluate whether its benefit increases on uncertain cases, uncertainty groups are defined once per dataset and seed from the full SMC-Rec model and then held fixed for every model in the comparison. Specifically, we compute the exact full-catalog prediction entropy of SMC-Rec for each test instance, rank the test instances by entropy, and use the 33.3% and 66.7% empirical quantiles to form equally sized Low-, Medium-, and High-uncertainty groups. The resulting test-instance indices are then reused unchanged when evaluating ConvRec, Fixed-SMC, and SMC-Rec. Thus, the three models are always compared on the same examples within each uncertainty group. Figure 4 reports mean NDCG@20 and sample standard deviation across the three runs for these fixed groups.

For low-uncertainty samples, the gap among the three models is relatively small because the current interest state is clearer and fixed compression can already produce a stable representation. The performance gap grows for medium- and high-uncertainty groups, indicating that adaptive compression is more useful when the reference entropy is high. The trend therefore reflects performance differences on the same test instances. This analysis also distinguishes SMC-Rec from UFRec-like methods: uncertainty affects not only future-loss weighting but also the compression process inside the encoder.

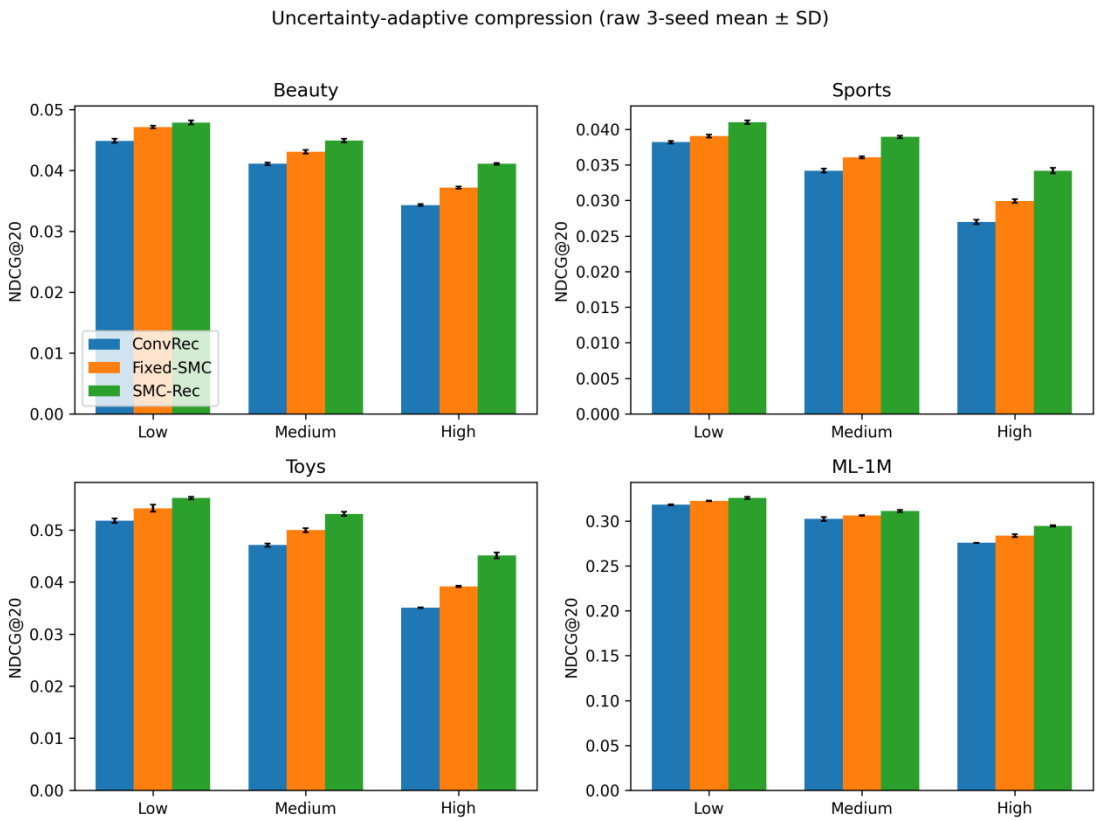


Figure 4. Uncertainty-adaptive compression under fixed prediction-entropy groups. For each dataset and seed, Low/Medium/High groups are the lower/middle/upper thirds of the full SMC-Rec entropy ranking; the same test-instance indices are used for ConvRec, Fixed-SMC, and SMC-Rec. Bars show mean NDCG@20 ± SD over three seeds.

5.9 Robustness under Noise and Missing Semantics

Real recommendation logs contain accidental clicks, casual browsing, missing semantic fields, and attribute noise. The perturbation study reported here is conducted on the Beauty dataset. To examine whether the model relies excessively on external semantics, two robustness experiments are designed. First, test-time input histories are corrupted by inserting randomly sampled unobserved items at ratios of 0%, 5%, 10%, 15%, and 20% as controlled behavioral-noise perturbations. Second, parts of the semantic fields are masked at inference time with ratios of 0%, 10%, 20%, 30%, and 40% to emulate missing item titles, brands, categories, or descriptions. These perturbed inputs are used only for this robustness probe; training and the main evaluation use the unmodified histories. Figure 5 shows the NDCG@20 retention rate under these progressive perturbation ratios.

The results show that SMC-Rec maintains the highest or most stable performance retention under both types of perturbation. The behavioral-noise experiment indicates that intra-pyramid divergence-consensus learning and uncertainty gating can reduce the impact of accidental behaviors on the final representation. In the semantic-field-dropout experiment, the ID-only SASRec baseline is almost unaffected in retention because it does not use semantic fields, but its absolute recommendation performance is still lower than that of semantic-enhanced models. Among semantic-enhanced models, SMC-Rec drops more slowly than SVD-ConvRec, indicating that calibrated semantic initialization alone cannot replace the complete semantic memory compression process.

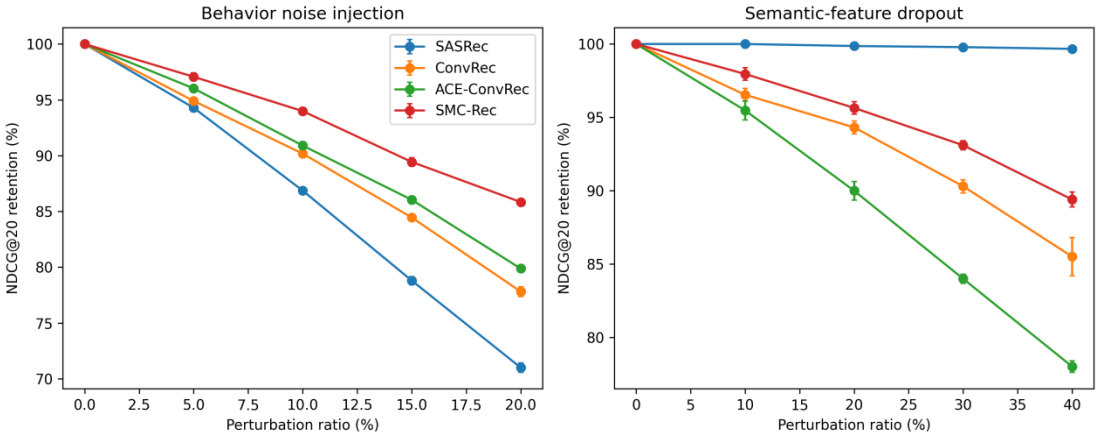


Figure 5. Robustness on Beauty under behavior noise and semantic-feature dropout. Curves show mean NDCG@20 retention ± SD over three seeds.

5.10 Sparse Users and Cold-start Items

To further verify the practical value of the semantic-memory perspective, users are divided into short, medium, and long groups according to training-history length, while target items are divided into rare, normal, and popular groups according to their interaction counts in the training set. This experiment examines how the benefit of SMC-Rec varies with available history length and whether external semantics and spectral calibration help long-tail items.

As summarized in Table 6, SMC-Rec achieves more pronounced relative gains on short users and rare items. For short-history users, collaborative signals are insufficient and sequence models are more easily affected by local behavioral noise. For rare items, interaction frequency is limited and pure ID representations struggle to learn stable vectors. Through interaction-level semantic memory, spectral calibration, and multi-scale compression, SMC-Rec converts semantic cues in limited histories into usable user states and supplements the collaborative deficiency of long-tail items with calibrated semantic representations. These results reinforce that the present empirical advantage is not confined to users with long histories; in fact, the largest gains are observed in data-sparse regimes.

Table 6. Mean ± SD NDCG@20 on sparse-user and cold-start-item groups over three independent runs.

Group	ConvRec	SVD-ConvRec	UF-style-ConvRec	SMC-Rec
Short users(≤5)	0.0278 ±0.0003	0.0304 ±0.0004	0.0316 ±0.0005	**0.0342 ±0.0003**
Medium users(6–10)	0.0399 ±0.0003	0.0427 ±0.0004	0.0438 ±0.0003	**0.0479 ±0.0001**
Long users(>10)	0.0528 ±0.0003	0.0553 ±0.0004	0.0562 ±0.0003	**0.0603 ±0.0003**
Rare items(≤5 actions)	0.0195 ±0.0001	0.0216 ±0.0003	0.0238 ±0.0001	**0.0273 ±0.0003**
Normal items	0.0377 ±0.0005	0.0407 ±0.0003	0.0424 ±0.0002	**0.0460 ±0.0003**
Popular items	0.0645 ±0.0006	0.0673 ±0.0003	0.0667 ±0.0005	**0.0693 ±0.0005**

5.11 Hyperparameter Sensitivity

Four architecture/learning hyperparameters are analyzed using compact search grids: spectral calibration strength λ in {0.00, 0.02, 0.05, 0.10, 0.20, 0.30}, uncertainty temperature τ in {0.50, 0.75, 1.00, 1.25, 1.50}, future horizon K in {1, 2, 3, 4}, and negative candidate-pool size N in {32, 64, 128, 256}. Figure 6 reports the corresponding validation-guided sensitivity curves. The three auxiliary-loss coefficients $\lambda_{fut}, \lambda_{neg}, and\ \lambda_{con}$ are handled separately as objective-balancing parameters: they are selected on validation NDCG@20 and fixed across seeds, while their functional

necessity is tested through the variants in Table 5. This separation avoids conflating architectural sensitivity with the presence or absence of an auxiliary objective.

Figure 6 reports aggregates from the archived seed-level runs. For each seed, NDCG@20 is normalized by the best value within the same dataset and parameter sweep, and the normalized values are then averaged across seeds. For spectral calibration strength, the best λ is 0.05 on Beauty and ML-1M and 0.10 on Sports and Toys. For uncertainty temperature, the best τ is 1.00 on Beauty and ML-1M and 1.25 on Sports and Toys. For future horizon, K = 2 is best on Beauty and ML-1M, whereas K = 3 is best on Sports and Toys. For the negative candidate pool, the best N is 128 on Beauty and Sports, 256 on Toys, and 64 on ML-1M. These results indicate dataset-specific preferences rather than a single universal optimum.

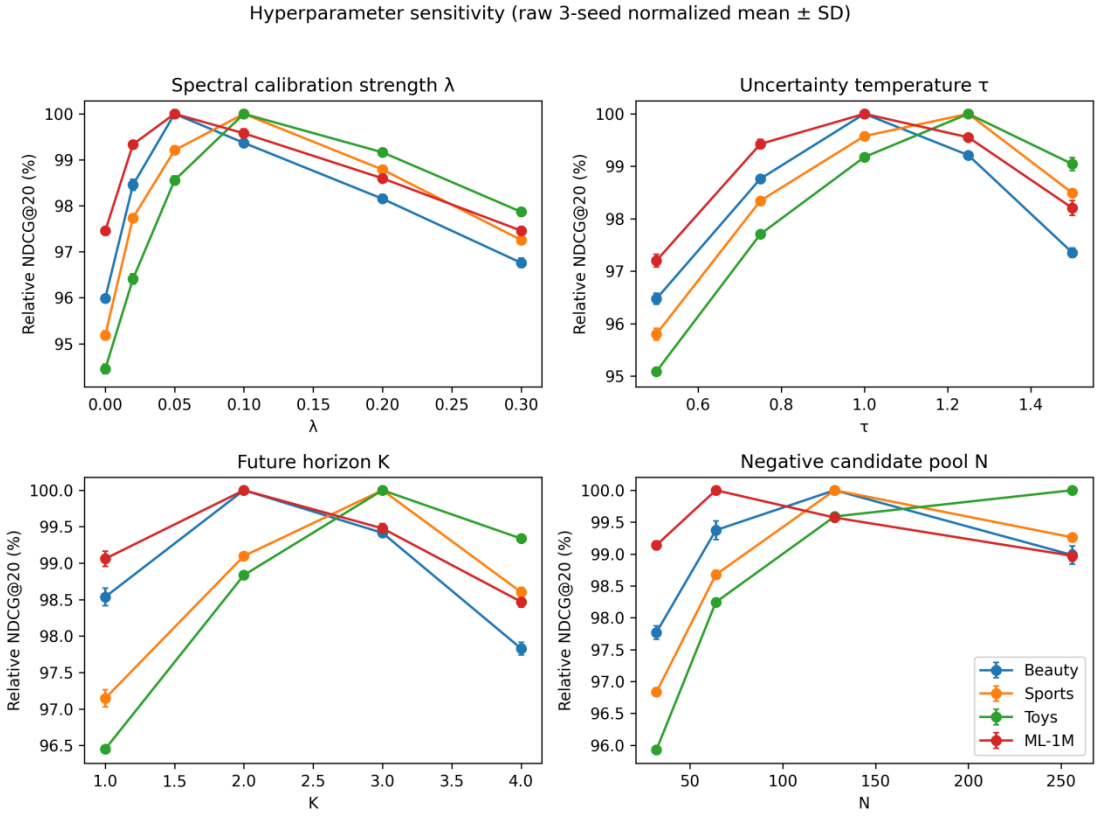


Figure 6. Hyperparameter sensitivity. Curves show mean ± SD of per-seed normalized NDCG@20.

6. Conclusion

This work reframes attribute-aware sequential recommendation as a knowledge-representation and adaptive memory-compression problem. SMC-Rec organizes heterogeneous evidence at the interaction level, uses offline spectral calibration to control the geometry of external semantics, and applies prediction uncertainty inside a pyramid convolution encoder to determine how historical information is retained and compressed. Its intra-pyramid short-, mid-, and long-term views further provide complementary training signals without introducing additional teacher models at inference. The empirical study shows that this formulation improves full-ranking recommendation quality across four datasets, is especially useful in sparse-user and rare-item regimes, and remains robust to behavioral noise and missing semantic fields. The wall-clock study also clarifies the deployment trade-off: the convolutional backbone scales linearly with history length, but exact uncertainty estimation adds catalog-dependent scoring passes and produces moderate overhead relative to fixed ConvRec. These results suggest a practical user-modeling design principle: rich semantic knowledge should be geometrically controlled, temporally compressed, and conditioned on model confidence rather than injected as an unstructured side feature. The present evidence nevertheless has limitations, including three accuracy seeds, only one substantially longer-history dataset, dependence on a fixed

offline Sentence-BERT representation, and the cost of exact full-catalog entropy. Future work will study additional long-history and cross-domain benchmarks, sampled or top-M uncertainty estimators, online semantic updates, and more interpretable memory-selection mechanisms.

From the perspective of user modeling and adaptive personalization, the resulting contribution is a user-state construction pipeline: semantic knowledge is acquired offline, represented at the interaction level, geometrically calibrated, adaptively compressed under uncertainty, and evaluated not only for predictive effectiveness but also for robustness and deployment cost. This emphasis is consistent with prior user-adaptive recommendation work (Deldjoo et al., 2023; Pelánek et al., 2024; Saiapin et al., 2024) while providing a distinct focus on uncertainty-controlled semantic memory compression.

Statements and Declarations

Funding: This work is supported by the Fundamental Research Funds for the Central Universities (Grant No. 3072025PT1504). Competing Interests: Xingyue Xie and Hao Bai are affiliated with Shenzhen Poweroak Newener Co., Ltd.; the remaining authors declare no competing interests.

Data Availability: The four benchmark datasets analyzed in this study are public datasets identified and cited in Section 5.2. The seed-level results supporting the reported tables and figures are archived in SMC-Rec_reproducibility.zip (Appendix A, Section A.6). Acknowledgments: We thank all colleagues in our research group for valuable discussions and technical support. The hyperparameter search spaces, selected configurations, tuning logs, environment specification, and run scripts are documented in Appendix A and supplied in the supplementary reproducibility archive.

References

Chen, Y., Li, J., Liu, Z., Keskar, N. S., Wang, H., McAuley, J., & Xiong, C. (2022). Generating negative samples for sequential recommendation. arXiv preprint arXiv:2208.03645.

Cui, Z., Tang, X., Liu, P., Zhang, X., Li, S., He, X., & Ma, C. (2026). Looking farther with confidence: Uncertainty-guided future learning for sequential recommendation. arXiv preprint arXiv:2605.28493. https://arxiv.org/abs/2605.28493

Deldjoo, Y., Jannach, D., Bellogin, A., Difonzo, A., & Zanzonelli, D. (2023). Fairness in recommender systems: Research landscape and future directions. User Modeling and User-Adapted Interaction, 34(1), 59–108. https://doi.org/10.1007/s11257-023-09364-z

Elsayed, S., Le, N. S., Rashed, A., & Schmidt-Thieme, L. (2026). Rethinking convolutional networks for attribute-aware sequential recommendation. In Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-ECAI 2026).

Fan, H., Zhu, M., Hu, Y., Feng, H., He, Z., Liu, H., & Liu, Q. (2025). TiM4Rec: An efficient sequential recommendation model based on time-aware structured state space duality model. Neurocomputing, 654, 131270. https://doi.org/10.1016/j.neucom.2025.131270

Fang, H., Zhang, D., Shu, Y., & Guo, G. (2020). Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. ACM Transactions on Information Systems, 39(1), 1–42.

Gao, T., Yao, X., & Chen, D. (2021). SimCSE: Simple contrastive learning of sentence embeddings. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021) (pp. 6894–6910).

Harper, F. M., & Konstan, J. A. (2015). The MovieLens datasets: History and context. ACM

Transactions on Interactive Intelligent Systems, 5(4), Article 19.
Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Session-based recommendations with recurrent neural networks. In Proceedings of the 4th International Conference on Learning Representations (ICLR 2016).
Hidasi, B., & Karatzoglou, A. (2018). Recurrent neural networks with top-k gains for session-based recommendations. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018) (pp. 843–852).
Hou, Y., Zhang, J., Lin, Z., Lu, H., Xie, R., McAuley, J. J., & Zhao, W. X. (2024). Large language models are zero-shot rankers for recommender systems. In Advances in Information Retrieval: 46th European Conference on Information Retrieval (ECIR 2024), Part II (pp. 364–381). https://doi.org/10.1007/978-3-031-56060-6_24
Huang, T., Dong, Y., Ding, M., Yang, Z., Feng, W., Wang, X., & Tang, J. (2021). MixGCF: An improved training method for graph neural network-based recommender systems. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2021) (pp. 665–674).
Huang, Y.-H., Lo, L., Xie, H., Shuai, H.-H., & Cheng, W.-H. (2025). Future sight and tough fights: Revolutionizing sequential recommendation with FENRec. Proceedings of the AAAI Conference on Artificial Intelligence, 39(11), 11826–11834. https://doi.org/10.1609/aaai.v39i11.33287
Jiang, S., Zheng, H., Liu, B., Chen, L., Lov, K., Xu, C., Ding, L., Zhou, Q., Cui, C., Liu, X., Liu, X., Badr, Y., Xu, X., Yang, J., Wen, E. D., Akkerhuis, G. J. M., Guan, C., Jin, R., Qiu, R., Chen, X., Xu, S., Zhou, Z., Chen, P., Yang, R., Chen, H., Meng, X., Zhou, S., Kharod, D., Xu, S., Jin, Q., Yang, Q., Zhu, W., Huang, Q., Huang, Y., Liu, D., Aggarwal, P., Zhou, H., Wang, E., Chang, S., Gan, X., Chen, W., Kolay, S., & Li, H. (2026). LoopFM: Learning frOm HistOrical RePresentations of Foundation Model for Recommendation. arXiv preprint arXiv:2605.29280.
Kang, W.-C., & McAuley, J. (2018). Self-attentive sequential recommendation. In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM 2018) (pp. 197–206).
Krichene, W., & Rendle, S. (2020). On sampled metrics for item recommendation. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2020) (pp. 1748–1757). https://doi.org/10.1145/3394486.3403226
Lee, D., Kim, H.-Y., & Lee, J. (2026). ACE: Anisotropy-controllable embedding for LLM-enhanced sequential recommendation. In Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2026) (pp. 3919–3923). https://doi.org/10.1145/3805712.3809911
Li, Y., Wang, L., Zhao, J., Tian, Z., Wang, Y., Wang, L., & Chen, X. (2026). Divergence meets consensus: A multi-source negative sampling framework for sequential recommendation. In Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2026) (pp. 981–992). https://doi.org/10.1145/3805712.3809555
Liu, C., Li, X., Cai, G., Dong, Z., Zhu, H., & Shang, L. (2021). Noninvasive self-attention for side information fusion in sequential recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 35(5), 4249–4256. https://doi.org/10.1609/aaai.v35i5.16549
Liu, Q., Zhao, X., Wang, Y., Wang, Y., Zhang, Z., Sun, Y., Li, X., Wang, M., Jia, P., Chen, C., Huang, W., & Tian, F. (2024). Large language model enhanced recommender systems: A survey. arXiv

preprint arXiv:2412.13432.
Liu, C., Lin, J., Wang, J., Liu, H., & Caverlee, J. (2024). Mamba4Rec: Towards efficient sequential recommendation with selective state space models. arXiv preprint arXiv:2403.03900.
Liu, Z., Liu, Q., Wang, Y., Wang, W., Jia, P., Wang, M., Liu, Z., Chang, Y., & Zhao, X. (2025). SIGMA: Selective gated Mamba for sequential recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 39(12), Article 33336. https://doi.org/10.1609/aaai.v39i12.33336
Ma, C., Kang, P., & Liu, X. (2019). Hierarchical gating networks for sequential recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2019) (pp. 825–833).
McAuley, J., Targett, C., Shi, Q., & van den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015) (pp. 43–52).
Pan, L.-W., Pan, W.-K., Wei, M.-Y., Yin, H., & Ming, Z. (2025). A survey on sequential recommendation. Frontiers of Computer Science. https://doi.org/10.1007/s11704-025-41329-w
Pelánek, R., Effenberger, T., & Jarušek, P. (2024). Personalized recommendations for learning activities in online environments: A modular rule-based approach. User Modeling and User-Adapted Interaction, 34(4), 1399–1430. https://doi.org/10.1007/s11257-024-09396-z
Rashed, A., Elsayed, S., & Schmidt-Thieme, L. (2022). Context and attribute-aware sequential recommendation via cross-attention. In Proceedings of the 16th ACM Conference on Recommender Systems (RecSys 2022) (pp. 71–80).
Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019) (pp. 3982–3992).
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI 2009) (pp. 452–461).
Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing personalized Markov chains for next-basket recommendation. In Proceedings of the 19th International Conference on World Wide Web (WWW 2010) (pp. 811–820).
Saiapin, A., Balitskiy, G., Bershatsky, D., Katrutsa, A., Frolov, E., Frolov, A., Oseledets, I., & Kharin, V. (2024). Federated privacy-preserving collaborative filtering for on-device next app prediction. User Modeling and User-Adapted Interaction, 34(4), 1369–1398. https://doi.org/10.1007/s11257-024-09395-0
Seol, J., Gang, M., Lee, S.-G., & Park, J. (2024). Proxy-based item representation for attribute and context-aware recommendation. In Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM 2024) (pp. 616–625). https://doi.org/10.1145/3616855.3635824
Shani, G., Heckerman, D., & Brafman, R. I. (2005). An MDP-based recommender system. Journal of Machine Learning Research, 6, 1265–1295.
Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., & Jiang, P. (2019). BERT4Rec: Sequential

recommendation with bidirectional encoder representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM 2019) (pp. 1441–1450).
Tang, J., & Wang, K. (2018). Personalized top-N sequential recommendation via convolutional sequence embedding. In Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM 2018) (pp. 565–573).
Wei, W., Ren, X., Tang, J., Wang, Q., Su, L., Cheng, S., Wang, J., Yin, D., & Huang, C. (2024). LLMRec: Large language models with graph augmentation for recommendation. In Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM 2024) (pp. 806–815).
Xie, Y., Zhou, P., & Kim, S. (2022). Decoupled side information fusion for sequential recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2022) (pp. 1611–1621).
Xu, C., Zhao, P., Liu, Y., Xu, J., Sheng, V. S., Cui, Z., Zhou, X., & Xiong, H. (2019). Recurrent convolutional neural network for sequential recommendation. In Proceedings of the 2019 World Wide Web Conference (WWW 2019) (pp. 3398–3404).
Yan, A., Cheng, S., Kang, W.-C., Wan, M., & McAuley, J. (2019). CosRec: 2D convolutional neural networks for sequential recommendation. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM 2019) (pp. 2173–2176).
Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J. M., & He, X. (2019). A simple convolutional generative network for next item recommendation. In Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM 2019) (pp. 582–590).
Zhang, W., Du, T., & Wang, J. (2016). Deep learning over multi-field categorical data. In Proceedings of the 38th European Conference on Information Retrieval (ECIR 2016) (pp. 45–57).
Zhou, K., Wang, H., Zhao, W. X., Zhu, Y., Wang, S., Zhang, F., Wang, Z., & Wen, J.-R. (2020). S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM 2020) (pp. 1893–1902).

Appendix A. Hyperparameter Search Spaces and Selected Configurations

This appendix documents the tuning protocol underlying all reported comparisons. Table A1 fixes the settings shared by every run; Table A2 gives the candidate value sets that define the search space of each reproduced model; Table A3 lists the selected common hyperparameters for each model on each dataset; Table A4 lists the complete method-specific settings of SMC-Rec and of the three controlled variants; and Table A5 lists the final model-specific configuration of every evaluated method. Dataset order in all tables is Beauty (B), Sports (S), Toys (T), and MovieLens-1M (M). All selections are made by validation NDCG@20 only.

A.1 Shared training and selection protocol

Table A1. Settings shared by SMC-Rec and all reproduced baselines.

Item	Value
Optimizer	AdamW, β1 = 0.9, β2 = 0.999, ε = 1e-8
Learning rate	Selected per model–dataset pair from the grids in Table A2; constant within a trial, no warm-up
Weight decay	Selected per model–dataset pair from {0, 1e-5, 1e-4, 1e-3}
Batch size	256 (fixed for all models and datasets)
Maximum sequence length	200 (fixed; shorter histories are left-padded and masked)
Negative sampling	One uniformly sampled unobserved item per positive for the base ranking loss; method-specific disagreement candidate pools are listed in Table A4. Evaluation is full-ranking over the complete catalog.
Parameter initialization	Xavier uniform, matched across methods for a given seed
Gradient clipping	Global L2 norm clipped at 1.0
Early stopping	Validation NDCG@20, patience 10, maximum 200 epochs; best checkpoint restored
Search strategy	Random search without replacement, at most 40 trials per model–dataset pair (tuning seed 2025); grids with fewer than 40 combinations are enumerated exhaustively
Selection metric	Validation NDCG@20 on the second-to-last interaction
Final accuracy seeds	{42, 2025, 3407}; mean ± sample SD over three independent runs
Test-set usage	Single evaluation of the selected configuration; never used for tuning or early stopping
Software / hardware	Python 3.10.14, PyTorch 2.3.1, CUDA 12.1, cuDNN 9.1; one NVIDIA A100 80GB PCIe GPU, Ubuntu 22.04 LTS (Section 5.2)
External semantics (semantic-aware methods only)	Frozen sentence-transformers/all-mpnet-base-v2 (768-d); the offline SVD-calibrated matrix is cached and supplied only to SMC-Rec and SVD-ConvRec.

Note. Shared settings are identical for all 16 evaluated methods. Only the entries listed in Tables A2–A4 vary across models; every trial uses the same preprocessed data, split, candidate set, and seed-matched initialization.

A.2 Model-specific hyperparameter search spaces

Table A2 reports the candidate value sets defining the search space of each method. The first five columns are shared across all model families; the last column records parameters specific to an architecture or to an added training module. For each model–dataset pair, configurations were sampled without replacement from the Cartesian product of these value sets under the 40-trial cap described in Section 5.4; the sampled trial vectors, not an enumeration of the full product, are archived in the trial registry (Section A.6). Value sets follow the ranges used by the corresponding public reference implementations and are deliberately aligned for structurally comparable models.

Table A2. Complete hyperparameter search spaces for all reproduced models.

Model	Learning rate	Hidden width d	Depth / blocks / heads	Dropout / weight decay	Model-specific search space
GRU4Rec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} layers	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	GRU hidden size {64, 128, 256}
Caser	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2} conv blocks	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Horizontal filters {4, 8, 16}; vertical filter length {2, 3, 4}
NextItNet	{1e-4, 5e-4, 1e-	{64, 128,	{4, 8} residual	{0.1, 0.2, 0.3, 0.5} /	Kernel {3, 5}; dilation pattern

	3, 5e-3}	256}	blocks	{0, 1e-5, 1e-4, 1e-3}	{1,2,4} or {1,2,4,8}
SASRec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} blocks; {1, 2, 4} heads	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	— (all tuned parameters are listed in the shared columns)
BERT4Rec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{2, 3, 4} blocks; {2, 4} heads	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Cloze mask ratio ρ {0.1, 0.2, 0.4, 0.6}
Mamba4Rec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} layers	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	SSM state dim {16, 32, 64}; expansion factor {2, 4}
SIGMA	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} layers	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	SSM state dim {16, 32, 64}; expansion factor {2, 4}
TiM4Rec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} layers	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	SSM state dim {16, 32, 64}; time kernel {3, 5}
S3-Rec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} blocks; {1, 2, 4} heads	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Pre-training epochs {10, 20, 50}; MI weight {0.1, 0.5, 1.0}; attribute mask {0.15, 0.30}
DIF-SR	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} blocks; {1, 2, 4} heads	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Side-information dropout {0.0, 0.1, 0.2}; fusion blocks {1, 2, 3}
CARCA	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128, 256}	{1, 2, 3} blocks; {2, 4} heads	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Context dropout {0.0, 0.1, 0.2}; context blocks {1, 2, 3}
ConvRec	{1e-4, 5e-4, 1e-3, 5e-3}	{64, 128}	{2, 3, 4} pyramid levels	{0.1, 0.2, 0.3, 0.5} / {0, 1e-5, 1e-4, 1e-3}	Kernel {3, 5}; stride fixed at 2; same grid as SMC-Rec
SVD-ConvRec	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	ConvRec backbone fixed to its best trial; calibration strength α {0.00, 0.02, 0.05, 0.10, 0.20, 0.30} (6 combinations, enumerated exhaustively)
UF-style-ConvRec	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	ConvRec backbone fixed; τ {0.50, 0.75, 1.00, 1.25, 1.50}; H {1, 2, 3, 4}; λf {0.1, 0.2, 0.5, 1.0}
Div-ConvRec	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	ConvRec backbone fixed; pool N {32, 64, 128, 256}; λd {0.1, 0.2, 0.5, 1.0}; λc {0.1, 0.2, 0.5, 1.0}
SMC-Rec	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	Inherited (frozen)	ConvRec backbone fixed to its winning trial (kernel and stride inherited); α {0.00, 0.02, 0.05, 0.10, 0.20, 0.30}; τ {0.50, 0.75, 1.00, 1.25, 1.50}; H {1, 2, 3, 4}; N {32, 64, 128, 256}; λf, λd, λc {0.1, 0.2, 0.5, 1.0}

Note. “Inherited (frozen)” means the method reuses the winning ConvRec configuration on the same dataset and searches only its added-module parameters under the same per-pair trial cap (40 configurations, or the full grid when it has fewer than 40 combinations). The exact sampled trial vectors are enumerated in the trial registry of the supplementary archive. λf, λd and λc denote the future-supervision, divergence-loss and consensus-loss coefficients, respectively.

A.3 Final selected configurations

Each cell of Table A3 gives the selected values in the order Beauty / Sports / Toys / ML-1M. “Depth”

denotes recurrent layers for GRU4Rec, residual blocks for NextItNet, attention blocks for attention-based and attribute-aware models, and pyramid levels for ConvRec, SMC-Rec and the controlled variants.

Table A3. Final selected common hyperparameters (B / S / T / M).

Model	Learning rate	Hidden width d	Depth	Dropout	Weight decay
GRU4Rec	1e-3 / 1e-3 / 5e-4 / 1e-3	128 / 128 / 128 / 256	1 / 1 / 2 / 2	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
Caser	1e-3 / 5e-4 / 1e-3 / 1e-3	128 / 128 / 128 / 256	2 / 2 / 2 / 2	0.3 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-5 / 1e-3
NextItNet	1e-3 / 1e-3 / 5e-4 / 5e-4	128 / 128 / 128 / 256	4 / 4 / 8 / 8	0.2 / 0.3 / 0.2 / 0.1	1e-4 / 1e-5 / 1e-4 / 1e-3
SASRec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	2 / 2 / 2 / 3	0.2 / 0.2 / 0.2 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
BERT4Rec	5e-4 / 5e-4 / 5e-4 / 1e-3	128 / 128 / 128 / 256	2 / 2 / 2 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
Mamba4Rec	1e-3 / 1e-3 / 5e-4 / 5e-4	64 / 128 / 64 / 128	2 / 2 / 2 / 3	0.1 / 0.2 / 0.2 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
SIGMA	1e-3 / 5e-4 / 5e-4 / 5e-4	64 / 64 / 64 / 128	2 / 2 / 3 / 3	0.1 / 0.2 / 0.2 / 0.1	1e-5 / 1e-4 / 1e-4 / 1e-3
TiM4Rec	1e-3 / 5e-4 / 5e-4 / 5e-4	64 / 64 / 128 / 128	2 / 2 / 2 / 3	0.1 / 0.2 / 0.2 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
S3-Rec	1e-3 / 5e-4 / 1e-3 / 1e-3	64 / 128 / 64 / 128	2 / 2 / 2 / 3	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
DIF-SR	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	2 / 2 / 2 / 3	0.2 / 0.2 / 0.2 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
CARCA	1e-3 / 5e-4 / 5e-4 / 1e-3	128 / 128 / 128 / 128	2 / 2 / 2 / 3	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
ConvRec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	3 / 3 / 3 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
SVD-ConvRec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	3 / 3 / 3 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
UF-style-ConvRec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	3 / 3 / 3 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
Div-ConvRec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	3 / 3 / 3 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3
SMC-Rec	1e-3 / 1e-3 / 5e-4 / 1e-3	64 / 128 / 64 / 128	3 / 3 / 3 / 4	0.2 / 0.2 / 0.3 / 0.1	1e-4 / 1e-4 / 1e-4 / 1e-3

Note. The identical rows for ConvRec, its three controlled variants and SMC-Rec reflect the controlled design: SMC-Rec and the controlled variants inherit the winning ConvRec backbone configuration on each dataset and search only their proposed or added module parameters, so the differences in Table 4 are attributable to the components under study rather than to optimization settings. All values are the winners of the validation search and are fixed across the three accuracy seeds.

A.4 Method-specific final configurations

Table A4. Final method-specific parameters of SMC-Rec and the controlled variants.

Parameter	Beauty	Sports	Toys	ML-1M
SMC-Rec: spectral calibration strength α	0.05	0.10	0.10	0.05
SMC-Rec: uncertainty temperature τ	1.00	1.25	1.25	1.00
SMC-Rec: future horizon K	2	3	3	2
SMC-Rec: negative candidate pool N	128	128	256	64
SMC-Rec: pyramid depth	3	3	3	4
SMC-Rec: convolution kernel	3	3	3	5
SMC-Rec: hidden dimension d	64	128	64	128
SMC-Rec: future-loss weight λf	0.2	0.2	0.5	0.1
SMC-Rec: divergence-loss weight λd	0.1	0.1	0.2	0.1
SMC-Rec: consensus-loss weight λc	0.5	0.5	1.0	0.2
SVD-ConvRec: calibration strength α	0.05	0.10	0.10	0.05
UF-style-ConvRec: (τ, H, λf)	1.00, 2, 0.2	1.25, 3, 0.2	1.25, 3, 0.5	1.00, 2, 0.1
Div-ConvRec: (N, λd, λc)	128, 0.1, 0.5	128, 0.1, 0.5	256, 0.2, 1.0	64, 0.1, 0.2

Note. α, τ, K and N for SMC-Rec are the validation-selected optima reported in Sections 5.7 and 5.11. The three loss coefficients are drawn from {0.1, 0.2, 0.5, 1.0}, selected by validation NDCG@20 and then fixed for all seeds of a dataset. SVD-ConvRec receives the same calibrated semantic matrix as SMC-Rec; UF-style-ConvRec and Div-ConvRec retain the ConvRec backbone of Table A3.

A.5 Final selected model-specific configurations

Table A5. Final selected model-specific configurations for all evaluated methods (Beauty / Sports / Toys / ML-1M).

Model	Beauty	Sports	Toys	ML-1M
GRU4Rec	GRU hidden 128; layers 1	GRU hidden 128; layers 1	GRU hidden 128; layers 2	GRU hidden 256; layers 2
Caser	h-filters 8; v-len 3	h-filters 8; v-len 2	h-filters 16; v-len 3	h-filters 16; v-len 2
NextItNet	kernel 3; dil. 1-2-4	kernel 5; dil. 1-2-4	kernel 3; dil. 1-2-4-8	kernel 5; dil. 1-2-4-8
SASRec	heads 2; blocks 2	heads 2; blocks 2	heads 1; blocks 2	heads 2; blocks 3
BERT4Rec	heads 2; rho 0.2; blocks 2	heads 2; rho 0.2; blocks 2	heads 4; rho 0.4; blocks 2	heads 2; rho 0.2; blocks 4
Mamba4Rec	state 32; exp. 2	state 32; exp. 2	state 64; exp. 4	state 64; exp. 4
SIGMA	state 16; exp. 2	state 32; exp. 2	state 32; exp. 2	state 64; exp. 4
TiM4Rec	state 32; t-kernel 3	state 32; t-kernel 3	state 64; t-kernel 5	state 64; t-kernel 5
S3-Rec	heads 2; pre 20; MI 0.5; mask 0.15	heads 2; pre 20; MI 0.5; mask 0.15	heads 2; pre 50; MI 1.0; mask 0.30	heads 4; pre 50; MI 0.5; mask 0.15
DIF-SR	heads 2; side-drop 0.1; fusion 1	heads 2; side-drop 0.1; fusion 2	heads 2; side-drop 0.2; fusion 2	heads 4; side-drop 0.0; fusion 2
CARCA	heads 2; ctx-drop 0.1; ctx-blk 2	heads 2; ctx-drop 0.0; ctx-blk 2	heads 2; ctx-drop 0.1; ctx-blk 2	heads 4; ctx-drop 0.1; ctx-blk 3
ConvRec	kernel 3; stride 2	kernel 3; stride 2	kernel 3; stride 2	kernel 5; stride 2
SVD-ConvRec	alpha 0.05; backbone inherited	alpha 0.10; backbone inherited	alpha 0.10; backbone inherited	alpha 0.05; backbone inherited
UF-style-	tau 1.00; H 2; lambda_f 0.2	tau 1.25; H 3; lambda_f 0.2	tau 1.25; H 3; lambda_f 0.5	tau 1.00; H 2;

ConvRec				lambda_f 0.1
Div-ConvRec	N 128; lambda_d 0.1; lambda_c 0.5	N 128; lambda_d 0.1; lambda_c 0.5	N 256; lambda_d 0.2; lambda_c 1.0	N 64; lambda_d 0.1; lambda_c 0.2
SMC-Rec	kernel 3; alpha 0.05; tau 1.00; H 2; N 128; lambda 0.2/0.1/0.5	kernel 3; alpha 0.10; tau 1.25; H 3; N 128; lambda 0.2/0.1/0.5	kernel 3; alpha 0.10; tau 1.25; H 3; N 256; lambda 0.5/0.2/1.0	kernel 5; alpha 0.05; tau 1.00; H 2; N 64; lambda 0.1/0.1/0.2

Note. Every value lies inside the corresponding candidate set of Table A2. “Backbone inherited” means that the ConvRec hidden dimension, pyramid depth, kernel and the five common hyperparameters are taken from Table A3. The lambda triple for SMC-Rec is ordered (lambda_f / lambda_d / lambda_c); the full settings are repeated in Table A4. Rho denotes the cloze mask ratio, pre the number of pre-training epochs, MI the mutual-information weight, mask the attribute-mask ratio, state the SSM state dimension, exp. the expansion factor, and dil. the dilation pattern. These entries complete the per-method selected configurations: Tables A1, A3 and A5 together fully determine every baseline run in Table 4.

A.6 Reproducibility checklist

The supplementary archive SMC-Rec_reproducibility.zip contains: (i) 64 YAML configuration files, one per model–dataset pair, storing every value in Tables A1, A3, A4 and A5; (ii) the trial registry for all 2,424 tuning runs, with trial index, sampled hyperparameter vector, tuning seed, validation NDCG@20, epoch of the best checkpoint, and wall-clock time; (iii) the 192 seed-level result files for Table 4, together with the archived per-seed outputs for the ablation variants (Table 5), the sparse-user and cold-start groups (Table 6), and the semantic geometry, uncertainty-group, robustness and sensitivity analyses (Figures 3–6); (iv) environment.yml and requirements.txt pinning Python 3.10.14, PyTorch 2.3.1, CUDA 12.1 and cuDNN 9.1; (v) preprocessing scripts with the exact field templates of Section 5.2, the sentence-transformers/all-mpnet-base-v2 checkpoint identifier, and the offline SVD calibration routine; (vi) the training and evaluation entry points with the fixed seed list {42, 2025, 3407}; and (vii) SHA-256 checksums for the cached raw and calibrated semantic matrices. Running a configuration file reproduces the corresponding row of Table 4 without manual intervention. The archive is supplied with the submission as supplementary material.