Semantic Trimming and Auxiliary Multi-step Prediction for Generative Recommendation

Tianyu Zhan*
Zhejiang University
Hangzhou, Zhejiang, China
yuzt@zju.edu.cn

Kairui Fu*
Zhejiang University
Hangzhou, Zhejiang, China
fukairui.fkr@zju.edu.cn

Chengfei Lv
Alibaba
Hangzhou, Zhejiang, China
chengfei.lcf@alibaba-inc.com

Zheqi Lv†
Zhejiang University
Hangzhou, Zhejiang, China
zheqilv@zju.edu.cn

Shengyu Zhang†
Zhejiang University
Hangzhou, Zhejiang, China
sy_zhang@zju.edu.cn

arXiv:2604.05329v3 [cs.IR] 28 Aug 2026

Abstract

Generative Recommendation (GR) has recently transitioned from atomic item-indexing to Semantic ID (SID)-based frameworks to capture intrinsic item relationships and enhance generalization. However, the adoption of high-granularity SIDs leads to two critical challenges: prohibitive training overhead due to sequence expansion and unstable performance reliability characterized by non-monotonic accuracy fluctuations. We identify that these seemingly independent issues are fundamentally rooted in the Semantic Dilution Effect, where redundant tokens consume massive computation and dilute the already sparse learning signals in recommendation. To counteract this, we propose STAMP (Semantic Trimming and Auxiliary Multi-step Prediction), a framework utilizing a dual-end optimization strategy. We argue that effective SID learning requires simultaneously addressing low input information density and sparse output supervision. On the input side, Semantic Adaptive Pruning (SAP) dynamically filters redundancy during the forward pass, converting noise-laden sequences into compact, information-rich representations. On the output side, Multi-step Auxiliary Prediction (MAP) employs a multi-token objective to densify feedback, strengthening long-range dependency capture and ensuring robust learning signals despite compressed inputs. By coupling input purification with signal amplification, STAMP enhances both training efficiency and representation capability. Experiments on public Amazon and large-scale industrial datasets show that, unlike more aggressive compression baselines that trade substantial accuracy for maximal speedup, STAMP achieves 1.23–1.38× speedup and 17.2%–54.7% VRAM reduction while maintaining or improving performance across multiple architectures.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Large Language Models; Generative Recommendation; Training Acceleration

ACM Reference Format:
Tianyu Zhan, Kairui Fu, Chengfei Lv, Zheqi Lv, and Shengyu Zhang. 2018. Semantic Trimming and Auxiliary Multi-step Prediction for Generative Recommendation. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Generative Recommendation (GR) has emerged as a paradigm shift in the field of recommender systems, reformulating the task as an end-to-end sequence generation problem [19, 28, 38, 48]. However, early frameworks relying on atomic item identifiers were limited by the independence of unique IDs, which hinders information sharing. This limitation often leads to vocabulary explosion and cold-start problem for new items. To mitigate these issues, Semantic Identifier (SID)-based Generative Recommendation (SID-GR) [12, 33, 38, 65] decomposes each item ID into a sequence of SIDs. The shared SID vocabulary across different items enhances the generalization ability of such methods (e.g., cross-domain recommendation), which is why it has gained attention recently.

Although SID-GR demonstrates a higher potential compared to item ID-based GR, it also exposes two critical limitations that hinder its practical deployment: (1) High resource consumption and slow training speed [12]. Expanding an item ID into multiple SIDs significantly increases the sequence length, leading to quadratic growth in memory (VRAM) consumption and training time. (2) Large performance fluctuations and low stability [18, 23, 34, 64]. The relationship between the granularity of SIDs and recommendation accuracy is non-monotonic and exhibits inconsistent patterns across different models and datasets. Currently, research on SID-GR remains in its early stages. Efforts have prioritized improving SID informativeness [12, 45, 47, 50, 51, 54] and generation speed [13, 65], while optimizations of the "GR" backbone aimed at mitigating these specific inefficiencies remain relatively under-explored.

Intriguingly, we observe that these two issues—training inefficiency and performance instability—share a common structural

*Both authors contributed equally to this research.
†Corresponding authors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference acronym 'XX, Woodstock, NY
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

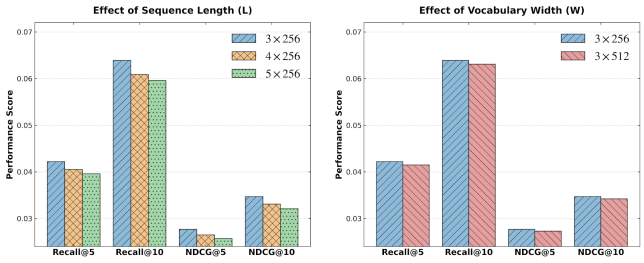


Figure 1: The adoption of finer-grained SID representations, entailing longer sequences and larger vocabularies, results in performance degradation.

source rather than being coincidental: both arise from the same L-fold sequence expansion inherent to SID tokenization, which we term the Semantic Dilution Effect. User intent is typically driven by a sparse subset of critical tokens within SIDs, meaning much of the sequence length consists of non-essential redundancy [1, 8, 30, 55]. This redundancy not only incurs massive computation (Information Non-uniformity) [36, 42] but also dilutes the already sparse learning signal in recommendation, making it difficult for the model to distinguish pivotal features from noise (Supervision Sparsity) [2, 3]. Unlike general language modeling, the sparse feedback loop in recommendation is insufficient to regularize such high-granularity, high-redundancy sequences.

Viewed through the lens of Semantic Dilution, existing optimization strategies from adjacent domains reveal inherent limitations. (1) *KV caching* and *Efficient attention kernels* [9, 16, 25, 35, 62] mitigate the computational burden of long-sequence decoding; however, these are primarily optimized for the long-horizon inference of complex reasoning tasks, whereas SID-GR is severely constrained by the massive retraining overhead required to capture volatile user interests. (2) *Token pruning* [1, 4, 8, 20, 39, 56, 60] aims to skip redundant inputs, but unlike NLP where non-informative particles (e.g., stop words) are distinct, every token in a high-granularity SID is a semantically dense identifier with complex inter-dependencies, making it difficult to distinguish "salient" tokens from noise without compromising representation quality [55]. Consequently, effective pruning mechanisms tailored for SID-GR remain unexplored.

Guided by this shared structural source, we propose **STAMP** (**S**emantic **T**rimming and **A**uxiliary **M**ulti-step **P**rediction), a framework designed to counteract Semantic Dilution Effect through a dual-end optimization strategy. We argue that effective SID learning requires simultaneously tackling low information density at the input and sparse supervision signals at the output. On the input side, we introduce Semantic Adaptive Pruning (SAP) to address Information Non-uniformity. SAP functions as a dynamic filter, identifying token utility via semantic features and attention centrality to remove redundancy during the forward pass [16, 62]. This directly mitigates computational overhead by converting noise-laden sequences into compact, information-rich representations. On the output side, we incorporate Multi-step Auxiliary Prediction (MAP) to address Supervision Sparsity. Beyond standard next-token prediction, MAP employs a multi-token objective [14] that densifies the feedback loop. This mechanism strengthens the model's ability to capture long-range dependencies, ensuring robust learning signals even with compressed inputs [14, 32]. By coupling input purification with signal amplification, STAMP jointly targets training efficiency and representation robustness. Our ablation study examines the extent to which the two modules act independently versus reinforce one another. We implemented STAMP on two representative architectures: Encoder-Decoder (T5) and Decoder-Only (Qwen). Extensive experiments on Amazon datasets and a large-scale industrial dataset (AL-GR-Tiny) demonstrate the effectiveness of our approach. Results show that STAMP achieves a 1.23–1.38× speedup and reduces VRAM usage by 17.2%–54.7% while maintaining recommendation performance.

The main contributions of this work are summarized as follows,

- We identify the Semantic Dilution Effect as a shared structural source of inefficiency and instability in SID-GR, representing an early effort focused on dynamically restructuring the SID learning process.
- We propose STAMP, a dual-end framework that couples input purification with signal amplification to reconcile training efficiency with model robustness.
- Comprehensive experiments across multiple architectures and large-scale datasets validate STAMP as a viable solution for industrial generative recommendation.

2 Related Work

Generative Retrieval. Generative recommendation has recently emerged as a transformative paradigm [19, 28, 38, 48, 59]. Prior to this shift, researchers primarily used LLMs as auxiliary components to enhance conventional discriminative models [5, 40, 44]. In contrast, the generative retrieval framework serializes item interaction sequences into discrete tokens, allowing the model to directly predict target items in an autoregressive manner [12, 18, 55]. Building on this foundation, subsequent studies have focused on refining the learning process by developing optimized training frameworks [24, 33] to handle large-scale data, while strategies such as reinforcement learning [46, 63] have been integrated to better align generated outputs with complex user preferences.

Semantic Identifier. The efficacy of generative recommendation depends critically on tokenization. The SID paradigm, pioneered by TIGER [38], represents items as hierarchical codeword sequences that enable knowledge transfer among semantically similar entities. While early methods derived codes purely from content modalities, subsequent work [22, 31, 33, 49] argues content alone is insufficient and incorporates collaborative signals. Regarding encoding efficiency [27], OneRec [65] streamlines quantization via Residual Quantized K-means, whereas RPG [17] replaces residual quantization with Optimized Product Quantization [13] to enable parallel SID prediction. However, it is structurally incompatible with the RQ-based SID construction used in many scenarios. GRID [18] introduced a unified, open-source SID-GR framework, while FORGE [12] targeted industrial deployment with a large-scale benchmark.

Acceleration of LLMRec. Extensive efforts have addressed the efficiency challenges of generative recommendation from different perspectives. Regarding model complexity, techniques such

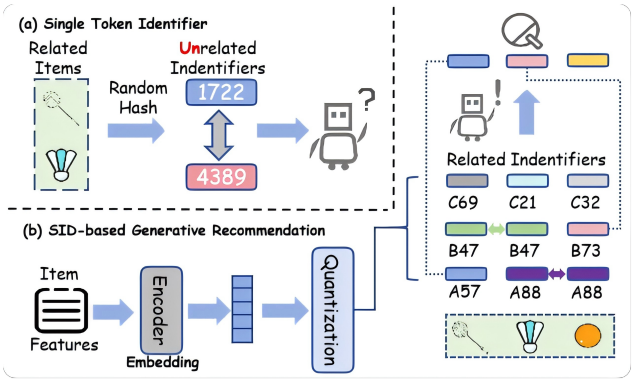


Figure 2: Comparison between (a) the single-token identifier mechanism and (b) the SID-based generative recommendation framework.

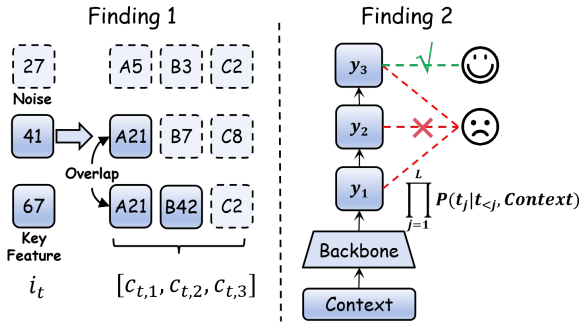


Figure 3: Redundancy is characterized by feature-level redundancy and sequence-level semantic overlaps (left). The joint probability constraint imposes higher demands on the model's long-range dependency capabilities (right).

as quantization [11, 26] and knowledge distillation [7, 43] effectively reduce memory usage, but these approaches usually impose additional post-training phases and often incur a loss in model performance. To target latency reduction, researchers have explored speculative sampling [29, 52] and efficient attention mechanisms [9, 25, 35, 57]. More recently, token pruning has been investigated as a promising direction to skip redundant computation in NLP and CV [1, 8, 55, 56, 58, 60, 61]. Despite their potential, speculative sampling and existing token pruning methods remain primarily inference-centric, leaving training-time efficiency comparatively under-explored.

3 Preliminaries

3.1 SID-based Generative Recommendation

Generative Recommendation. GR reformulates recommendation as an end-to-end sequence generation problem rather than a ranking task. Given a user $u \in \mathcal{U}$ with chronological history $H_u = [i_1, \ldots, i_t]$, GR learns a model P_θ that directly generates the identifier of the target item i_{t+1} autoregressively, aligning recommendation with language modeling and its associated LLM capabilities.

Semantic ID Tokenization. Independent atomic IDs suffer from vocabulary explosion and the cold-start problem. SIDs address this via a two-stage encoding-then-quantization process:

$$s_i = \mathcal{T}(i) = \mathrm{RQ}(h_i) = [c_{i,1}, c_{i,2}, \ldots, c_{i,L}], \tag{1}$$

where h_i is the semantic embedding of item i (e.g., from Flan-T5-XL [6]) and $\mathrm{RQ}(\cdot)$ is a hierarchical residual quantizer (RQ-VAE [21] or RQ-Kmeans [65]) mapping it to a length-L code over a shared, compact codebook $\mathcal{V}$. This indexes massive item spaces without vocabulary expansion while letting semantically similar items share tokens, mitigating cold-start [27].

SID-based Generative Recommendation. The model serializes the user's history into $X = [\mathcal{T}(i_1), \ldots, \mathcal{T}(i_t)]$, an input of length $N = t \times L$, and is trained to autoregressively predict the target's SID sequence $Y = [y_1, \ldots, y_L]$ by minimizing:

$$\mathcal{L}_{\mathrm{NTP}} = -\sum_{j=1}^{L} \log P(y_j \mid X, y_{<j}; \theta). \tag{2}$$

Crucially, the two challenges below are not independent artifacts of SID-GR but direct consequences of this same L-fold expansion: decomposing an item into L SIDs simultaneously (i) inflates X with tokens of low marginal information density, and (ii) requires satisfying L consecutive predictions against a per-example supervisory budget that does not scale with L. Both trace back to sequence length growing faster than either the information content of the input or the density of the supervision it receives.

3.2 Finding 1: Information Non-uniformity

The observation that input data is dominated by non-critical, redundant information is a pervasive pattern across machine learning: the Lottery Ticket Hypothesis [10] demonstrates that most parameters in dense networks are functionally redundant; token pruning and KV-cache eviction methods [1, 8, 53, 55, 56, 60] routinely discard a large fraction of tokens without damaging performance; near-duplicate training samples yield rapidly diminishing returns [42]; and LLMs struggle to uniformly exploit information dispersed across long, redundant contexts [36]. The common thread is that model input information density is highly non-uniform: a small subset of elements carries the bulk of the predictive signal, while the rest is redundant or distracting.

This effect is structurally amplified in SID-GR. Decomposing one item into L tokens boosts zero-shot semantic expressiveness but proportionally expands the input with tokens of low marginal information density, manifesting as two interconnected redundancy types: (1) *Feature-level Redundancy*, where user intent is often driven by a few key features (e.g., Brand) rather than the full hierarchy Category → Brand → Attribute, so fine-grained attributes act as comparatively irrelevant noise; (2) *Sequence-level Redundancy*, where historical items frequently share identical SIDs (e.g., brand, category), so repeated encoding yields diminishing marginal information across the temporal dimension.

This does not mean repeated tokens are individually uninformative, as a recurring brand token is a valid preference signal. However, it is primarily the *first* occurrence that carries substantial information gain, while subsequent repetitions largely reinforce a signal already encoded elsewhere. Indiscriminately retaining every repeated SID dilutes the limited attention budget across near-duplicates rather than sharpening it on the genuinely distinctive tokens that drive the prediction.

Notably, this redundancy also carries an optimization opportunity. Because SID explicitly decouples an item's previously entangled attributes into independent, discrete tokens, the underlying attention mechanism can evaluate each token's contribution in isolation, rather than having to disentangle redundancy from a single monolithic item embedding. This structural separation gives pruning a natural handle: low-utility tokens can be located and removed at the granularity of individual SIDs, enabling aggressive compression while preserving the specific tokens that carry the user's core intent.

3.3 Finding 2: Supervision Sparsity

Unlike general generative tasks where every generated token is an immediate supervision signal, GR derives valid training signal solely from the ground-truth item at the sequence's end, forcing gradients to backpropagate across long contexts without intermediate guidance. Although SID extends the prediction target from a single atomic ID to L tokens, the input sequence expands by the exact same factor L, so the *relative* density of supervision per input token does not improve at all—the model still receives feedback from only one training example's worth of ground truth, now spread across a proportionally longer context. This makes it substantially harder for the model to attribute the final purchase decision to the specific, sparse set of key features embedded within an otherwise noisy interaction history.

Moreover, correctly predicting an item requires getting all L SID tokens right consecutively: since each token's prediction conditions on the correctness of all preceding tokens, a single early mistake cascades through the remaining steps and irrecoverably shifts the trajectory away from the intended item:

$$P(\text{Item}) = \prod_{j=1}^{L} P(y_j \mid y_{<j}, \text{Context}) \tag{3}$$

so early-token errors compound via the chain rule and shift the entire subsequent trajectory, causing degradation as L grows [2, 12, 18]. Unlike open-ended language generation, where numerous plausible continuations can absorb this effect, SID-GR admits only one exact target sequence per interaction, leaving the sparse, single-point terminal supervision especially fragile and motivating auxiliary supervision to stabilize training and strengthen long-range dependency induction.

4 Methodology

4.1 Overview

Motivated by these observations, we propose **STAMP** (Semantic Trimming and Auxiliary Multi-step Prediction). As illustrated in Figure 4, STAMP integrates two modules into the standard Transformer backbone to reconcile the trade-off between training efficiency and representation robustness. **SAP** acts as an adaptive information filter inserted at a designated layer l_{prune}: it dynamically evaluates token importance through intrinsic information and extrinsic structural centrality, removing redundant tokens to accelerate computation in all subsequent layers. **MAP** functions as a semantic amplifier at the learning stage, introducing a dense supervision signal by predicting future tokens, thereby enhancing the model's capability to capture long-range dependencies. Both modules are lightweight: SAP reuses the attention weights already computed by the backbone, and MAP consists of a single MLP that shares the LM head, so neither introduces a substantial number of additional parameters.

4.2 Semantic Adaptive Pruning

We design SAP by capitalizing on the Transformer's intrinsic self-attention mechanism. Since attention weights naturally quantify the information flow between tokens, they serve as an effective, computationally efficient proxy for token utility [16, 62], enabling SAP to dynamically identify and retain discriminative SIDs without incurring significant additional computational overhead.

4.2.1 ***Token Importance Evaluation.*** Let $\mathbf{H}^{(l)} \in \mathbb{R}^{B\times N\times d}$ be the hidden states at pruning layer l. We score each token along two complementary dimensions. **Semantic Saliency** uses the ℓ_1-norm, since high-magnitude features typically indicate significant semantic activation [39]:

$$S_{\text{sem}}(i) = \text{Norm}\left(|\mathbf{h}_i^{(l)}|_1\right), \tag{4}$$

with $\text{Norm}(\cdot)$ a stabilizing normalization. **Attention Centrality** [41] captures token i's structural importance by summing the attention it receives from all other tokens and heads:

$$S_{\text{attn}}(i) = \sum_{h=1}^{\mathcal{H}} \sum_{j=1}^{N} \mathbf{A}_{h,j,i}, \tag{5}$$

where a higher score marks i as a frequently-attended information hub. The final importance score multiplies both signals:

$$\mathcal{I}_i = S_{\text{sem}}(i) \times S_{\text{attn}}(i), \tag{6}$$

so a high $\mathcal{I}_i$ requires a token to be both internally rich and externally pivotal.

4.2.2 ***Architecture-Adaptive Pruning Execution.*** To translate importance scores into practical acceleration, we adopt a unified Order-Preserving Sequence Compression strategy. This mechanism explicitly reduces the sequence length for subsequent layers while rigorously maintaining the relative temporal order of tokens.

Token Selection. The execution begins by identifying the set of indices, $\mathcal{K}$, intended for retention. We retain the top-scoring indices with $|\mathcal{K}| = \lfloor \alpha \cdot N \rfloor$ under a retention ratio $\alpha \in (0, 1]$ that determines the total budget of preserved tokens. For decoder-only architectures (e.g., Qwen), where input history and the generation target share one token stream, we additionally protect a window $\mathcal{W}_{\text{recent}} \subseteq \mathcal{K}$ spanning the target segment and its immediate boundary, retained regardless of importance score to preserve gradient flow to the

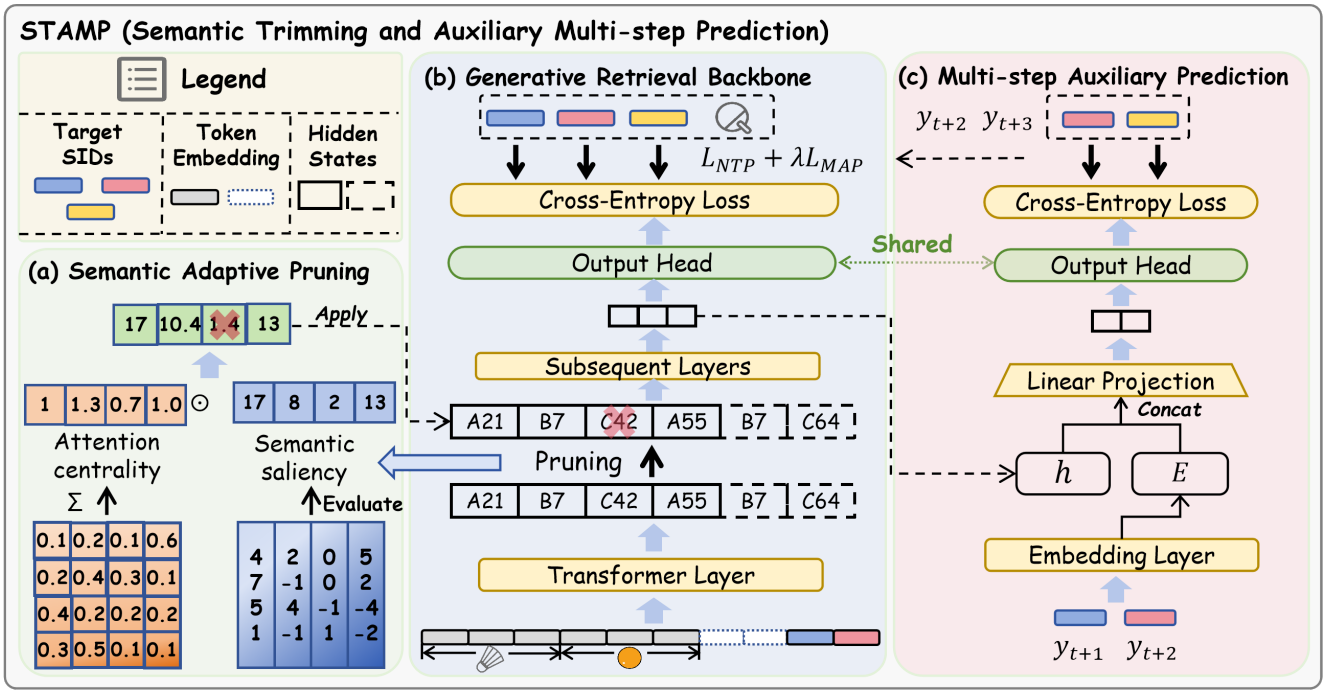


Figure 4: Overview of the STAMP framework. (b) STAMP accelerates training by compressing sequences via pruning while ensuring performance through multi-token prediction. (a) SAP computes token importance to retain high-utility information and remove redundancy. (c) MAP predicts multiple future items simultaneously to provide richer supervision signals.

target; the remaining budget in $\mathcal{K}$ is filled by the highest-scoring tokens elsewhere.

Index Re-ordering and Compression. Naively gathering selected tokens by importance would disrupt the temporal integrity of the user interaction sequence. Therefore, we first sort the selected indices to restore their original relative positions:

$$\mathcal{K}_{\text{sorted}} = \text{Sort}(\mathcal{K}, \text{order} = \text{ascending}). \tag{7}$$

Subsequently, we apply compression by gathering the hidden states corresponding to these sorted indices:

$$H^{(l)}_{\text{compressed}} = \text{Gather}\left(H^{(l)}, \mathcal{K}_{\text{sorted}}\right). \tag{8}$$

This operation reduces the sequence length from N to $\alpha \cdot N$ for all subsequent layers. Surviving tokens retain their *original* absolute position identifiers rather than being renumbered, so relative or rotary positional encodings (e.g., Qwen) still perceive the true temporal spacing between interactions rather than misreading pruned-adjacent tokens as consecutive.

4.3 Multi-step Auxiliary Prediction

MAP addresses supervision dilution via Multi-Token Prediction (MTP) [14]. Given SIDs are typically only 3 tokens, we extend the supervision horizon with a single auxiliary head.

4.3.1 ***Dense Supervision Formulation.*** We add an objective forecasting y_{t+2} alongside the standard target y_{t+1}. Given backbone state $\mathbf{h}_t \in \mathbb{R}^d$, we fuse it with the ground-truth embedding $E(y_{t+1})$ into a *foresight representation*:

$$\mathbf{h}^{mtp}_t = \text{MLP}\left(\text{Concat}(\mathbf{h}_t, E(y_{t+1}))\right), \tag{9}$$

which simulates the step-$(t+1)$ hidden state and forces the backbone to encode features transferable to future contexts. Conditioning on the ground-truth y_{t+1}—rather than a possibly-erroneous self-prediction—mirrors how DeepSeek-V3's MTP module keeps its prediction chain causally consistent across depths [32], unlike the fully independent, parallel heads of earlier MTP formulations [14]; this keeps the auxiliary objective cheap (one MLP, shared LM head) while still probing $\mathbf{h}_t$'s ability to encode generalizable, forward-looking features.

4.3.2 ***Shared Projection Strategy.*** We map $\mathbf{h}^{mtp}_t$ to the vocabulary space using the backbone's own unembedding matrix rather than a new head:

$$P(y_{t+2} \mid \mathbf{h}_t, y_{t+1}) = \text{Softmax}(\text{LM_Head}(\mathbf{h}^{mtp}_t)), \tag{10}$$

aligning the auxiliary representation with the primary semantic space at no extra parameter cost.

4.4 Optimization Objective

STAMP is trained end-to-end with the primary NTP loss (Eq. 2) plus the MAP objective, which minimizes the negative log-likelihood of y_{j+1} given the foresight context at step j (summed to $L-1$, since predicting y_{j+1} requires ground-truth y_j as input):

$$\mathcal{L}_{\text{MAP}} = -\sum_{j=1}^{L-1} \log P(y_{j+1} \mid \mathbf{h}_j, y_j). \tag{11}$$

The overall training objective is then a weighted sum of the primary and auxiliary losses:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{NTP}} + \lambda \cdot \mathcal{L}_{\text{MAP}}, \tag{12}$$

where λ balances the auxiliary signal. As a training-only objective, MAP is discarded at inference, incurring zero latency overhead.

5 Experiments

To comprehensively evaluate the versatility and robustness of STAMP across different architectures and data scales, we conduct experiments using a dual-track evaluation setting: the **GRID** [18] framework implements the Encoder-Decoder (T5) architecture on three public Amazon datasets, while the **FORGE** [12] framework instantiates the Decoder-Only (Qwen) architecture on the industrial dataset AL-GR-Tiny. Through this setup, we aim to answer the following research questions:

- **RQ1:** How does STAMP perform in performance and efficiency on different architectures of SID-based generative recommendation?
- **RQ2:** What roles do STAMP's two modules, SAP and MAP, play in its performance?
- **RQ3:** How do the design configurations of SAP (*i.e.*, pruning strategy, pruning layer, and retention ratio) affect the model performance?
- **RQ4:** What mechanism enables STAMP to safely prune the majority of tokens?

5.1 Experimental Settings

5.1.1 ***Datasets.*** We employ four datasets divided into two categories, summarized in Table 1.

Table 1: Statistics of the datasets.

Dataset	#Users	#Items	#Interactions
Sports	35.6K	18.4K	296.3K
Beauty	22.4K	12.1K	198.5K
Toys	19.4K	11.9K	167.6K
AL-GR	131M	251M	14B

- **Amazon**[1]**:** We utilize three widely adopted subsets of the Amazon review dataset: *Beauty*, *Toys*, and *Sports*, which capture extensive user interactions within specific e-commerce categories. Consistent with established protocols, we apply a 5-core filter to retain only users and items with at least five interactions, and adopt a leave-one-out evaluation strategy: for each user sequence, the final interaction is reserved for testing, the penultimate for validation, and the remaining interactions serve as the training set.
- **AL-GR-Tiny**[2]**:** To evaluate scalability in a massive industrial context, we use AL-GR-Tiny, a subset of the AL-GR benchmark originating from a large-scale e-commerce platform in China—the first industrial-scale benchmark designed for forming SIDs in GR, integrating multimodal (text and image) features across over 250 million items. We use the benchmark's pre-processed open-source Stage 1 (S1) split (four days of training behaviors, with a test set from the following day) to ensure reproducibility.

[1] https://jmcauley.ucsd.edu/data/amazon/
[2] https://huggingface.co/datasets/AL-GR/AL-GR-Tiny

5.1.2 ***Baselines.***

- **EARN[55]:** An inference acceleration framework that introduces learnable prefix and suffix register tokens to compress the historical interactions of the user. Optimally, at 1/4 of the total network depth, it discards the original prompt tokens and computes attention exclusively using the register tokens for the subsequent layers, reducing both the computational overhead and the size of the KV cache. Notably, this framework is instantiated during the model training phase to serve as a baseline in our comparison.

5.1.3 ***Evaluation Metrics.***

- **Recommendation Effectiveness:** For GRID, we report **Recall@K** and **NDCG@K** ($K \in \{5, 10\}$). For FORGE, we report **HitRate@K** ($K \in \{20, 100, 500, 1000\}$), the percentage of cases where the ground-truth item appears in the top-K candidates.
- **Training Efficiency: Training Speedup** (time relative to the baseline over 1,600 steps) and **VRAM Reduction** (percentage decrease in peak GPU memory).

5.1.4 ***Implementation Details.*** We align our implementation with the standard configurations of the respective frameworks, adjusting only the framework-specific modules introduced by STAMP.

For GRID, we use Flan-T5-XL for semantic encoding and RK-Means tokenization with three codebooks. The backbone is a lightweight T5 Encoder-Decoder (8 layers: 4 encoder, 4 decoder; hidden dim 128; 6 heads; MLP dim 1,024). We set SAP's retention ratio $\alpha = 1/3$ and MAP's fusion coefficient $\lambda = 0.3$, optimizing with Adam (lr 1×10^{-3}, weight decay 1×10^{-4}, dropout 0.15) on a single A100-40GB GPU (batch size 32, sequence length 120), with early stopping (patience 10, validated every 1,600 steps). Because GRID's streaming training relies on early stopping, the total step count varies across methods and runs, which in turn causes final performance to fluctuate considerably across different random seeds; we therefore report results averaged over five seeds ($\{1, 42, 999, 1024, 2025\}$) for a statistically robust comparison.

For FORGE, we use the pre-processed AL-GR-Tiny dataset and a Qwen2.5-0.5B-Instruct backbone with a 1,280-token input (1,024 source, 256 target). We set $\alpha = 0.6$ and $\lambda = 0.3$, fine-tuning in bfloat16 on two A100-80GB GPUs with AdamW (β_1=0.9, β_2=0.999, ϵ=1e-8, per-device batch size 80) for a single epoch, totaling 100,000 steps. Given the large scale of this industrial dataset, training for a single epoch is sufficient for the model to reach convergence, and this fixed-epoch setup further ensures that the total number of training steps is strictly identical across all compared methods. Consequently, unlike the GRID experiments, seed-averaging is unnecessary here, and we report results under FORGE's default single-run configuration.

5.2 Overall Performance (RQ1)

We compare STAMP against EARN across both paradigms. The notation $L = n$ denotes pruning/compression after the n-th Transformer layer; following EARN's own layer-ratio convention, we fix

Table 2: Overall performance on GRID (T5) over Beauty, Sports and Toys. Results are averaged over five random seeds to mitigate performance variance caused by streaming training and early stopping. We report the training time and VRAM usage on a single NVIDIA A100 GPU. ↑ indicates higher is better, and ↓ indicates lower is better.

Backbone	Dataset	Method	Performance Metrics				Efficiency Metrics			
			Recall@5	Recall@10	NDCG@5	NDCG@10	Time ↓	Speedup ↑	VRAM ↓	Reduction ↑
GRID (T5)	Beauty	Base	0.0426 ± 0.0013	0.0645 ± 0.0019	0.0282 ± 0.0014	0.0353 ± 0.0013	212s	–	22234MiB	–
		EARN (L=2)	0.0404 ± 0.0013	0.0619 ± 0.0010	0.0263 ± 0.0012	0.0333 ± 0.0010	116s	1.83×	13544MiB	39.1%
		STAMP (L=2)	**0.0444** ± 0.0012	0.0655 ± 0.0015	**0.0295** ± 0.0011	**0.0363** ± 0.0012	170s	1.25×	13862MiB	37.7%
		EARN (L=1)	0.0379 ± 0.0010	0.0591 ± 0.0009	0.0246 ± 0.0006	0.0315 ± 0.0005	**98s**	**2.16×**	**9116MiB**	**59.0%**
		STAMP (L=1)	0.0441 ± 0.0010	**0.0657** ± 0.0022	0.0293 ± 0.0006	**0.0363** ± 0.0009	154s	1.38×	10378MiB	53.3%
	Sports	Base	**0.0234** ± 0.0004	**0.0354** ± 0.0004	**0.0154** ± 0.0002	**0.0193** ± 0.0001	205s	–	29370MiB	–
		EARN (L=2)	0.0208 ± 0.0015	0.0322 ± 0.0018	0.0135 ± 0.0012	0.0172 ± 0.0013	114s	1.80×	17834MiB	39.3%
		STAMP (L=2)	0.0228 ± 0.0015	0.0352 ± 0.0015	0.0153 ± 0.0008	**0.0193** ± 0.0008	166s	1.23×	18318MiB	37.6%
		EARN (L=1)	0.0197 ± 0.0009	0.0311 ± 0.0014	0.0126 ± 0.0006	0.0163 ± 0.0007	**95s**	**2.16×**	**12074MiB**	**58.9%**
		STAMP (L=1)	0.0232 ± 0.0009	**0.0354** ± 0.0019	0.0153 ± 0.0009	0.0192 ± 0.0012	151s	1.36×	13302MiB	54.7%
	Toys	Base	0.0345 ± 0.0010	0.0499 ± 0.0008	0.0238 ± 0.0009	0.0287 ± 0.0006	216s	–	22268MiB	–
		EARN (L=2)	0.0312 ± 0.0010	0.0471 ± 0.0017	0.0205 ± 0.0010	0.0256 ± 0.0011	117s	1.85×	11772MiB	47.1%
		STAMP (L=2)	**0.0356** ± 0.0009	0.0503 ± 0.0010	**0.0246** ± 0.0006	**0.0293** ± 0.0007	174s	1.24×	14256MiB	36.0%
		EARN (L=1)	0.0274 ± 0.0007	0.0415 ± 0.0013	0.0179 ± 0.0007	0.0225 ± 0.0009	**98s**	**2.20×**	**8428MiB**	**62.2%**
		STAMP (L=1)	0.0344 ± 0.0012	**0.0509** ± 0.0015	0.0235 ± 0.0009	0.0286 ± 0.0013	159s	1.36×	10658MiB	52.1%

Table 3: Overall performance on FORGE (Qwen2.5-Instruct-0.5B) over AL-GR-Tiny. We report the training time and VRAM usage on two NVIDIA A100 GPUs. ↑ indicates higher is better, and ↓ indicates lower is better.

Backbone	Dataset	Method	Performance Metrics				Efficiency Metrics			
			Hit@20	Hit@100	Hit@500	Hit@1000	Time ↓	Speedup ↑	VRAM ↓	Reduction ↑
FORGE (Qwen)	AL-GR-Tiny	Base	**0.0209**	**0.0508**	**0.1155**	0.1400	1139s	–	2*64232MiB	–
		EARN (L=6)	0.0152	0.0354	0.0839	0.1021	**434s**	**2.62×**	**2*29452MiB**	**54.1%**
		STAMP (L=6)	0.0207	0.0506	0.1149	**0.1405**	851s	1.34×	2*53196MiB	17.2%

L a priori at roughly one-quarter and one-half of backbone depth for the aggressive and conservative configurations ($L \in \{1, 2\}$ of 4 T5 encoder layers; $L = 6$ of 24 Qwen layers).

5.2.1 ***Performance on GRID (T5).*** Table 2 compares the base model, baselines, and STAMP.

- **Efficiency:** SAP's early pruning yields substantial gains: the aggressive configuration ($L = 1$) achieves a 1.36–1.38× speedup and over 52% VRAM reduction across all datasets. EARN is faster still, but STAMP's acceleration remains highly practical for resource-constrained deployment.
- **Effectiveness:** EARN's aggressive compression—encoding the entire user history into only two register tokens—creates an information bottleneck that causes severe performance degradation. STAMP, in contrast, closely tracks base-model accuracy on all three datasets, with the $L = 2$ configuration surpassing it outright on *Beauty* and *Toys*—suggesting SAP acts as a denoising filter that sharpens focus on core preferences. On *Sports*, the base model remains marginally ahead on most metrics; we note this as a case where pruning is not fully cost-free, though STAMP still substantially narrows the gap that SAP-only pruning would otherwise incur, and by a far smaller margin than EARN opens. MAP's densified supervision further improves training stability and helps offset this residual gap, yielding a favorable speed-accuracy balance overall.

5.2.2 ***Performance on FORGE (Qwen).*** Table 3 shows dynamics broadly consistent with GRID, with nuances from structural and data differences.

- **Efficiency:** STAMP ($L = 6$) reduces VRAM by 17.2%—more modest than on T5 because (i) we use a conservative $\alpha = 0.6$ to preserve dense collaborative signals, and (ii) Qwen2.5-0.5B's 24 layers mean more static VRAM is occupied by model parameters rather than data-dependent activations. The absolute memory savings, on the order of several gigabytes per GPU, remain highly impactful for industrial-scale training where GPU memory is a binding constraint.
- **Effectiveness:** AL-GR-Tiny's multimodal, collaborative-signal-enriched SIDs [12] are exceptionally information-dense, so EARN's bottleneck causes even sharper degradation here, and SAP-only pruning (Section 5.3.1) similarly loses accuracy to irrecoverable information loss. STAMP substantially narrows this gap: MAP's densified supervision recovers most of the lost accuracy, landing within 0.001 of base on Hit@20/100/500 and slightly exceeding it on Hit@1000.

5.3 In-depth Analysis

5.3.1 Ablation Study (RQ2). To isolate the contributions of **SAP** and **MAP**, Figure 5 reports accuracy (T5 on three Amazon datasets, Qwen on AL-GR-Tiny) and Table 4 reports efficiency. Three observations emerge consistently:

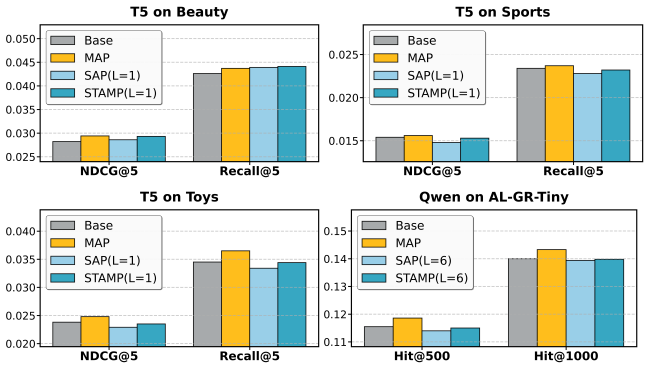


Figure 5: Ablation study of T5 on Amazon datasets and Qwen on AL-GR-Tiny.

Table 4: Efficiency Analysis.

Backbone	Dataset	Method	Time (s)	Speedup	VRAM (MiB)	Red.
T5	Beauty	Base	212	-	22234	-
		MAP	216	0.98×	22238	~0%
		SAP (L=2)	165	1.28×	13858	37.7%
		STAMP (L=2)	170	1.25×	13862	37.7%
		SAP (L=1)	146	1.45×	10374	**53.3%**
		STAMP (L=1)	154	1.38×	10378	53.3%
	Sports	Base	205	-	29370	-
		MAP	208	0.99×	29374	~0%
		SAP (L=2)	160	1.28×	18314	37.6%
		STAMP (L=2)	166	1.23×	18318	37.6%
		SAP (L=1)	145	1.41×	13298	54.7%
		STAMP (L=1)	151	1.36×	13302	54.7%
	Toys	Base	216	-	22268	-
		MAP	219	0.99×	22272	~0%
		SAP (L=2)	169	1.28×	14252	36.0%
		STAMP (L=2)	174	1.24×	14256	36.0%
		SAP (L=1)	153	1.41×	10654	52.2%
		STAMP (L=1)	159	1.36×	10658	52.1%
Qwen	AL-GR-Tiny	Base	1139	-	2*64232	-
		MAP	1152	0.99×	2*65786	-2.42%
		SAP (L=6)	838	1.36×	2*51632	19.6%
		STAMP (L=6)	851	1.34×	2*53196	17.2%

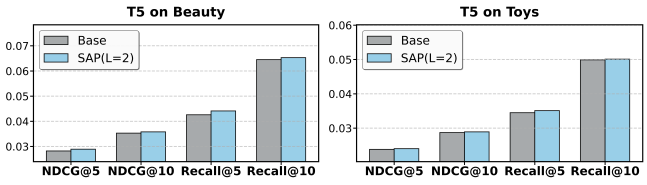


Figure 6: Base vs. SAP ($L = 2$) of T5 on Beauty and Toys.

- **MAP** alone consistently outperforms Base, indicating it is not merely a compensatory patch for SAP's information loss but an independent enhancement that improves shared-representation optimization via auxiliary supervision—justifying its marginal compute cost. This overhead scales with hidden dimension (the auxiliary MLP operates on Concat($\mathbf{h}_t, E(y_{t+1})$)): negligible for T5 ($d = 128$, ~0% VRAM change) but more visible for Qwen ($d = 896$, +2.42%), an acceptable trade-off given MAP is discarded at inference.
- **SAP** substantially cuts latency and VRAM via pruning, typically with a minor accuracy trade-off—except on T5 *Beauty*/*Toys*, where pruning at the second encoder layer actually improves accuracy over the base model (Figure 6). One possible explanation, which we leave to future work to verify, is that once the initial layers have already aggregated information across the sequence, removing the remaining low-utility tokens serves as a denoising step: real-world interaction data is known to often contain label or interaction noise [37], and discarding these tokens would let the model focus its remaining capacity on more reliable signals.
- **STAMP** combines both modules' effects, which the ablation shows are largely additive rather than mutually dependent—each contributes independently without one compensating for artifacts in the other. This additive relationship is itself useful: STAMP achieves SAP's efficiency and MAP's robustness simultaneously, and the fact that the two modules do not require tight coordination to be effective means that each can be tuned independently for a given deployment scenario, offering a practical solution for industrial settings that require frequent, efficient retraining.

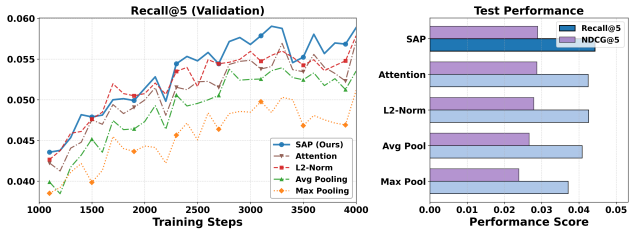


Figure 7: Comparison of Different Pruning Strategies (T5 on Beauty). (Left) Validation curves during the training phase. (Right) Performance on the test set.

5.3.2 Impact of Pruning Strategies (RQ3). We compare SAP against (1) aggregation baselines (Max/Average Pooling) and (2) single-signal baselines that rely only on magnitude (ℓ_2-norm) or only on attention centrality, all within the GRID framework. Figure 7 shows validation curves during training (left) and final test results (right). Two insights emerge. First, aggregation methods perform markedly worse: as seen in their lower training curves, blending token features together obscures the fine-grained semantic detail needed to capture precise user preferences, rather than merely discarding irrelevant tokens. Second, while the ℓ_2-norm method performs reasonably by favoring large-magnitude tokens, it only inspects a token's internal value in isolation and ignores its contextual importance, so it may discard tokens with small individual magnitude but high incoming attention from others; the attention-only method has a complementary blind spot, favoring tokens that actively broadcast information to the rest of the sequence while overlooking the semantic aggregators that become important later by gradually absorbing information from other tokens. SAP consistently outperforms both single-signal baselines, with the gap

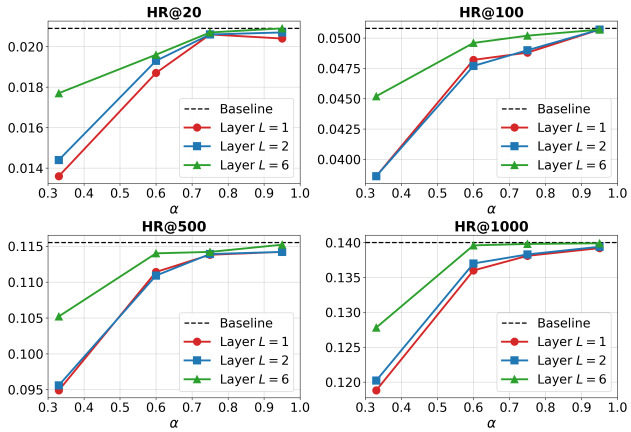


Figure 8: Performance of SAP with different Pruning Layer and Retention Ratio α (Qwen on AL-GR-Tiny).

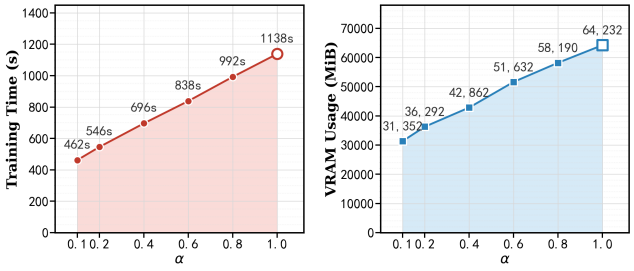


Figure 9: Efficiency of SAP (L=6) with different Retention Ratio α (Qwen on AL-GR-Tiny).

widening in later training stages, confirming that accurately estimating token importance requires jointly considering semantic saliency and attention centrality rather than either alone.

5.3.3 Hyperparameter Sensitivity: Pruning Layer and Retention Ratio (RQ3). Using Qwen2.5-0.5B-Instruct with SAP, we analyze the fundamental trade-off between training efficiency and downstream accuracy across the pruning layer (L) and retention ratio (α). In general, more aggressive pruning strategies—intervening at earlier layers or enforcing lower retention ratios—offer larger gains in training speed and VRAM reduction by minimizing forward token propagation, but correspondingly raise the risk of severe performance degradation. As shown in Figure 8, the model exhibits remarkable tolerance for moderate compression, validating the premise that a substantial proportion of input tokens is functionally redundant for the core recommendation task; however, predictive accuracy deteriorates sharply once compression surpasses a critical threshold, indicating that excessive pruning begins removing fine-grained semantic information indispensable for accurate user profiling. We further observe that the optimal pruning depth is intrinsically coupled with the retention ratio: the specific choice of pruning layer yields negligible impact under conservative settings, but delaying pruning to deeper layers becomes substantially more effective under aggressive compression. Consistent with prior work in inference acceleration [55], this suggests shallow layers are fundamentally crucial for encoding basic features and remain highly

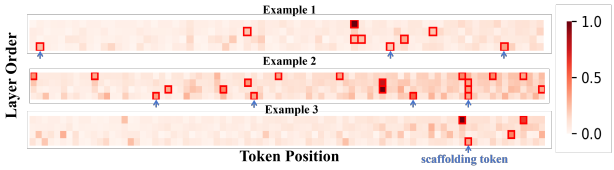


Figure 10: The attention distributions on different layers of three examples (T5 on Beauty).

sensitive to premature information loss, whereas deeper layers are more robust to structural token removal, having already gathered sufficient context.

5.3.4 Why is Pruning Safe? (RQ4). To understand why we can compress the majority of tokens without incurring performance loss, Figure 10 visualizes T5 encoder attention heatmaps (Layers 1–4) for three randomly sampled sequences from the Beauty test set. Consistent with the information-aggregation phenomenon observed in LLMs [15, 53], the attention patterns reveal a clear hierarchical information flow characterized by two key observations. First, a large portion of item tokens show consistently low attention scores throughout the entire encoding process; these likely represent noisy interactions or redundant information already overlapped by other dominant tokens. Since they rarely participate in the information exchange required for user profiling, they act as "dead weight" that can be safely pruned to save computation. Second, a distinct subset of tokens exhibits a dynamic pattern, receiving high attention weights in early layers but fading rapidly in deeper ones. This suggests early layers use these tokens as scaffolding to build basic features, which are later absorbed by a smaller set of key tokens that then carry the aggregated information forward. Crucially, this explains why pruning should not occur too early: removing these scaffolding tokens before their information has been absorbed disrupts the initial aggregation process, whereas pruning them later, once their semantic value has already been digested by the remaining tokens, is both safe and efficient. These findings align with the earlier quantitative observation that shallow-layer pruning is substantially more sensitive than deep-layer pruning.

6 Conclusion

In this work, we identified the Semantic Dilution Effect as the shared structural source of inefficiency and instability in SID-based Generative Recommendation, characterized by Information Non-uniformity at the input and Supervision Sparsity at the output. To address these interlinked challenges, we proposed STAMP, a dual-end framework reconciling training efficiency with model robustness. Its Semantic Adaptive Pruning (SAP) module dynamically filters redundant tokens to substantially cut computational and memory overhead, while its Multi-step Auxiliary Prediction (MAP) module acts as an independent, proactive enhancement—rather than a mere remedy for SAP's information loss—that densifies supervision throughout the sequence, strengthening long-range dependency capture and enabling robust representation learning from sparse behavioral trajectories. Extensive experiments across diverse datasets and backbone architectures (Encoder-Decoder and

Decoder-Only) show that STAMP achieves significant training acceleration and memory savings while consistently maintaining, and in some cases surpassing, the recommendation accuracy of full-sequence baselines. An empirical analysis further reveals that the underlying attention mechanism naturally concentrates on a sparse subset of discriminative tokens, providing a mechanistic rationale for why aggressive pruning remains safe.

7 GenAI Usage Disclosure

Portions of the code development for this work, specifically for debugging and refactoring, were assisted by Generative AI tools and subsequently reviewed by the authors to ensure accuracy. Additionally, Generative AI models, such as Gemini and ChatGPT, were utilized to polish the writing of this paper.

8 Ethical Considerations

All datasets used in this work are publicly available and have been anonymized prior to release, containing no personally identifiable information. No private user data was collected or used at any stage of this research. Therefore, we do not foresee significant ethical concerns regarding privacy, fairness, or potential misuse arising from this work.

References

[1] Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. 2025. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 9392–9401.

[2] Gregor Bachmann and Vaishnavh Nagarajan. 2024. The pitfalls of next-token prediction. *arXiv preprint arXiv:2403.06963* (2024).

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, Vol. 28.

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2022. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461* (2022).

[5] Lei Chen, Chen Gao, Xiaoyi Du, Hengliang Luo, Depeng Jin, Yong Li, and Meng Wang. 2025. Enhancing ID-based recommendation with large language models. *ACM Transactions on Information Systems* 43, 5 (2025), 1–30.

[6] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of machine learning research* 25, 70 (2024), 1–53.

[7] Yu Cui, Feng Liu, Pengbo Wang, Bohao Wang, Heng Tang, Yi Wan, Jun Wang, and Jiawei Chen. 2024. Distillation matters: empowering sequential recommenders to match the performance of large language models. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 507–517.

[8] Shuangrui Ding, Peisen Zhao, Xiaopeng Zhang, Rui Qian, Hongkai Xiong, and Qi Tian. 2023. Prune spatio-temporal tokens by semantic-aware temporal accumulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 16945–16956.

[9] Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. 2021. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 1733–1737.

[10] Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* (2018).

[11] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323* (2022).

[12] Kairui Fu, Tao Zhang, Shuwen Xiao, Ziyang Wang, Xinming Zhang, Chenchi Zhang, Yuliang Yan, Junjun Zheng, Xiangheng Kong, Shengyu Zhang, et al. 2026. Forge: Forming semantic identifiers for generative retrieval in industrial datasets. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 8858–8869.

[13] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. 2013. Optimized product quantization. *IEEE transactions on pattern analysis and machine intelligence* 36, 4 (2013), 744–755.

[14] Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*. 15706–15734.

[15] Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. 2024. When attention sink emerges in language models: An empirical view. *arXiv preprint arXiv:2410.10781* (2024).

[16] Zhiyu Guo, Hidetaka Kamigaito, and Taro Watanabe. 2024. Attention Score is not All You Need for Token Importance Indicator in KV Cache Reduction: Value Also Matters. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 21158–21166.

[17] Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian McAuley. 2025. Generating long semantic ids in parallel for recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 956–966.

[18] Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. 2025. Generative Recommendation with Semantic IDs: A Practitioner's Handbook. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM)*.

[19] Sein Kim, Hongseok Kang, Seungyoon Choi, Donghyun Kim, Minchul Yang, and Chanyoung Park. 2024. Large language models meet collaborative filtering: An efficient all-round llm-based recommender system. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1395–1406.

[20] Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. 2022. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 784–794.

[21] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11523–11532.

[22] Kaiyuan Li, Rui Xiang, Yong Bai, Yongxiang Tang, Yanhua Cheng, Xialong Liu, Peng Jiang, and Kun Gai. 2025. Bbqrec: Behavior-bind quantization for multi-modal sequential recommendation. *arXiv preprint arXiv:2504.06636* (2025).

[23] Xiaopeng Li, Bo Chen, Junda She, Shiteng Cao, You Wang, Qinlin Jia, Haiying He, Zheli Zhou, Zhao Liu, Ji Liu, Zhiyang Zhang, Yu Zhou, Guoping Tang, Yiqing Yang, Chengcheng Guo, Si Dong, Kuo Cai, Pengyue Jia, Maolin Wang, Wanyu Wang, Shiyao Wang, Xinchen Luo, Qigen Hu, Qiang Luo, Xiao Lv, Chaoyi Ma, Ruiming Tang, Kun Gai, Guorui Zhou, and Xiangyu Zhao. 2025. A Survey of Generative Recommendation from a Tri-Decoupled Perspective: Tokenization, Architecture, and Optimization. *Preprints* (December 2025). doi:10.20944/preprints202512.0203.v1

[24] Zida Liang, Changfa Wu, Dunxian Huang, Weiqiang Sun, Ziyang Wang, Yuliang Yan, Jian Wu, Yuning Jiang, Bo Zheng, Ke Chen, et al. 2025. Tbgrecall: A generative retrieval model for e-commerce recommendation scenarios. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*. 5863–5870.

[25] Jianghao Lin, Xinyi Dai, Rong Shan, Bo Chen, Ruiming Tang, Yong Yu, and Weinan Zhang. 2025. Large language models make sample-efficient recommender systems. *Frontiers of Computer Science* 19, 4 (2025), 194328.

[26] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems* 6 (2024), 87–100.

[27] Xinyu Lin, Haihan Shi, Wenjie Wang, Fuli Feng, Qifan Wang, See-Kiong Ng, and Tat-Seng Chua. 2025. Order-agnostic identifier for large language model-based generative recommendation. In *Proceedings of the 48th international ACM SIGIR conference on research and development in information retrieval*. 1923–1933.

[28] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Bridging items and language: A transition paradigm for large language model-based recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1816–1826.

[29] Xinyu Lin, Chaoqun Yang, Wenjie Wang, Yongqi Li, Cunxiao Du, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Efficient inference for large language model-based generative recommendation. *arXiv preprint arXiv:2410.05165* (2024).

[30] Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, Weizhu Chen, et al. 2024. Not all tokens are what you need for pretraining. *Advances in Neural Information Processing Systems* 37 (2024), 29029–29063.

[31] Gui Ling, Zhihong Chen, Yu Li, Tong Xiong, Kunhai Lin, Kaixuan Zhang, Yuliang Yan, Dan Ou, Haihong Tang, and Bo Zheng. 2026. Beyond Semantic IDs: Encoding Business-Value Ranking into Document Identifiers for Generative Retrieval. *arXiv preprint arXiv:2607.11392* (2026).

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

[33] Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. 2025. Generative recommender with end-to-end learnable item tokenization. In *Proceedings of the 48th International ACM SIGIR Conference on Research and*

Development in Information Retrieval. 729–739.

[34] Jingzhe Liu, Liam Collins, Jiliang Tang, Tong Zhao, Neil Shah, and Clark Mingxuan Ju. 2025. Understanding Generative Recommendation with Semantic IDs from a Model-scaling View. arXiv:2509.25522 [cs.AI] https://arxiv.org/abs/2509.25522

[35] Langming Liu, Liu Cai, Chi Zhang, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Yifu Lv, Wenqi Fan, Yiqi Wang, Ming He, et al. 2023. Linrec: Linear attention mechanism for long-term sequential recommender systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 289–299.

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

[37] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. 2021. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749* (2021).

[38] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems* 36 (2023), 10299–10315.

[39] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. 2021. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems* 34 (2021), 13937–13949.

[40] Yankun Ren, Zhongde Chen, Xinxing Yang, Longfei Li, Cong Jiang, Lei Cheng, Bo Zhang, Linjian Mo, and Jun Zhou. 2024. Enhancing sequential recommenders with augmented knowledge from aligned large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 345–354.

[41] Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. 2025. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 22857–22867.

[42] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems*, Vol. 35. 19523–19536.

[43] Wenqi Sun, Ruobing Xie, Junjie Zhang, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024. Distillation is all you need for practically using different pre-trained recommendation models. *CoRR* (2024).

[44] Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. 2024. Idgenrec: Llm-recsys alignment with textual id learning. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*. 355–364.

[45] Dongsheng Wang, Yuxi Huang, Shen Gao, Yifan Wang, Chengrui Huang, and Shuo Shang. 2025. Generative next poi recommendation with semantic id. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 2904–2914.

[46] Jie Wang, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. 2024. Reinforcement learning-based recommender systems with large language models for state reward and action modeling. In *Proceedings of the 47th International ACM SIGIR conference on research and development in information retrieval*. 375–385.

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

[48] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, and Tat-Seng Chua. 2023. Generative recommendation: Towards next-generation recommender paradigm. *arXiv preprint arXiv:2304.03516* (2023).

[49] Yidan Wang, Zhaochun Ren, Weiwei Sun, Jiyuan Yang, Zhixiang Liang, Xin Chen, Ruobing Xie, Su Yan, Xu Zhang, Pengjie Ren, et al. 2024. Content-based collaborative generation for recommender systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 2420–2430.

[50] Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, et al. 2024. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3245–3254.

[51] Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, Yang Zeng, Qiang Luo, Wencong Zeng, Ruiming Tang, Kun Gai, et al. 2025. Oneloc: Geo-aware generative recommender systems for local life service. *arXiv preprint arXiv:2508.14646* (2025).

[52] Yunjia Xi, Hangyu Wang, Bo Chen, Jianghao Lin, Menghui Zhu, Weiwen Liu, Ruiming Tang, Zhewei Wei, Weinan Zhang, and Yong Yu. 2025. Efficiency unleashed: Inference acceleration for LLM-based recommender systems with speculative decoding. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1891–1901.

[53] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. EFFICIENT STREAMING LANGUAGE MODELS WITH ATTENTION SINKS. The Twelfth International Conference on Learning Representations.

[54] Longtao Xiao, Haozhao Wang, Cheng Wang, Linfei Ji, Yifan Wang, Jieming Zhu, Zhenhua Dong, Rui Zhang, and Ruixuan Li. 2025. Unger: Generative recommendation with a unified code via semantic and collaborative integration. *ACM Transactions on Information Systems* 44, 2 (2025), 1–31.

[55] Chaoqun Yang, Xinyu Lin, Wenjie Wang, Yongqi Li, Teng Sun, Xianjing Han, and Tat-Seng Chua. 2025. EARN: Efficient Inference Acceleration for LLM-based Generative Recommendation by Register Tokens. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 3483–3494.

[56] Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. 2025. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 22128–22136.

[57] Yufei Ye, Wei Guo, Hao Wang, Hong Zhu, Yuyang Ye, Yong Liu, Huifeng Guo, Ruiming Tang, Defu Lian, and Enhong Chen. 2025. Fuxi-\beta: Towards a lightweight and fast large-scale generative recommendation model. *arXiv preprint arXiv:2508.10615* (2025).

[58] Tianyu Zhan, Kairui Fu, Zheqi Lv, and Shengyu Zhang. 2026. RASTP: Representation-Aware Semantic Token Pruning for Generative Recommendation with Semantic Identifiers. In *Proceedings of the ACM Web Conference 2026*. 8309–8312.

[59] Tianyu Zhan, Gui Ling, Tong Xiong, Kunhai Lin, Yang Wang, Kaixuan Zhang, Zhihong Chen, Yuliang Yan, Dan Ou, Shengyu Zhang, et al. 2026. TSGR: Taobao Search Generative Retrieval. *arXiv preprint arXiv:2607.18796* (2026).

[60] Zheng Zhan, Zhenglun Kong, Yifan Gong, Yushu Wu, Zichong Meng, Hangyu Zheng, Xuan Shen, Stratis Ioannidis, Wei Niu, Pu Zhao, et al. 2024. Exploring token pruning in vision state space models. *Advances in Neural Information Processing Systems* 37 (2024), 50952–50971.

[61] Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2025. Long context compression with activation beacon. In *International Conference on Learning Representations*, Vol. 2025. 101932–101948.

[62] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems* 36 (2023), 34661–34710.

[63] Zuowu Zheng, Ze Wang, Fan Yang, Jiangke Fan, Teng Zhang, Yongkang Wang, and Xingxing Wang. 2025. EGA-V2: An End-to-end Generative Framework for Industrial Advertising. arXiv:2505.17549 [cs.IR] https://arxiv.org/abs/2505.17549

[64] Qiyong Zhong, Jiajie Su, Yunshan Ma, Julian McAuley, and Yupeng Hou. 2025. Pctx: Tokenizing Personalized Context for Generative Recommendation. arXiv:2510.21276 [cs.IR] https://arxiv.org/abs/2510.21276

[65] Guorui Zhou, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Shiyao Wang, et al. 2025. OneRec Technical Report. *arXiv preprint arXiv:2506.13695* (2025).