Research Article Open Access

# Enhancing E-commerce Recommendation Efficiency: An Empirical Application of Mamba Architecture on Grocery Datasets

**Huỳnh Văn Bình**
*Faculty of Information Technology, Ho Chi Minh City University of Foreign Languages – Information Technology (HUFLIT), Ho Chi Minh City, Vietnam*

**Abstract:** *With the explosive growth of e-commerce transaction data, particularly in the grocery sector where user history is often long and repetitive, traditional Sequential Recommendation models like SASRec face significant challenges regarding computational costs and inference latency due to their quadratic complexity. This study proposes the application of Mamba4Rec, a novel architecture based on State Space Models (SSMs) with linear complexity, to address the efficiency bottleneck in large-scale recommender systems. By conducting extensive empirical experiments on the Instacart dataset, we compare Mamba4Rec against state-of-the-art Transformer-based baselines. The results demonstrate that Mamba4Rec achieves comparable recommendation accuracy (Recall@K and NDCG@K) to SASRec while significantly reducing training time and GPU memory consumption. These findings suggest that Mamba architecture offers a cost-effective solution for online retailers, enabling scalable and real-time recommendation services without upgrading hardware infrastructure.*

**Keywords:** Recommender Systems, Mamba Architecture, State Space Models, E-commerce Efficiency, Instacart, Sequential Recommendation.

## I. INTRODUCTION

In the era of information overload, Recommender Systems (RS) have become an indispensable component of modern e-commerce platforms, serving as a critical engine for enhancing user experience and driving revenue growth [1]. For online grocery retailers, where product catalogs are vast and user purchasing behaviors are highly repetitive and periodic, the ability to accurately predict the "next basket" or "next item" is paramount. Consequently, Sequence-aware and Sequential Recommendation (SR) have emerged as dominant research directions [2], aiming to model dynamic user preferences based on their historical interaction sequences [3].

In recent years, deep learning-based models, particularly those utilizing the Transformer architecture such as SASRec[4] and BERT4Rec [5] , have achieved state-of-the-art (SOTA) performance in SR tasks. By leveraging the self-attention mechanism, these models can effectively capture long-range dependencies within user behavior sequences. However, a significant bottleneck remains: the computational complexity of the self-attention mechanism is quadratic with respect to the sequence length ($O(L^2)$).This limitation poses a severe challenge for real-world applications involving long user histories, such as grocery shopping, where a user may accumulate hundreds or thousands of interactions over time. As the sequence length increases, the training time and GPU memory consumption of Transformer-based models grow exponentially, leading to high operational costs and latency issues in real-time inference environments [6].

To address the trade-off between recommendation accuracy and computational efficiency, recent advancements in deep learning have introduced the Mamba architecture [7], a novel class of Selective State Space Models (SSMs). Mamba distinguishes itself by offering linear computational complexity ($O(L)$) regarding sequence length, similar to Recurrent Neural Networks (RNNs), while maintaining the parallel training capabilities of Transformers. Preliminary studies in Natural Language Processing (NLP) suggest that Mamba can match or outperform Transformers in modeling long sequences with significantly lower resource utilization. However, the empirical application of Mamba in the specific domain of grocery recommendation, where data sparsity and sequence length differ from NLP tasks, remains underexplored.

In this paper, we propose an empirical study of the Mamba4Rec framework applied to the Instacart dataset, a benchmark dataset for grocery market basket analysis. Our primary objective is to investigate whether Mamba can serve as a cost-effective alternative to SASRec for large-scale e-commerce systems. We explicitly focus on the efficiency metrics—training speed and memory usage—without compromising recommendation accuracy.

The main contributions of this work are summarized as follows:

- We implement and evaluate the Mamba architecture on the Instacart dataset, adapting the model to handle sequential grocery purchasing behaviors.
- We conduct a comparative analysis between Mamba4Rec and the widely used SASRec baseline.
- We demonstrate that Mamba4Rec achieves competitive performance in terms of Recall and NDCG while significantly reducing training time and memory footprint, offering a scalable solution for businesses with limited computational infrastructure.

## II. Literature Review

### 2.1. The Evolution of Deep Learning in Sequential Recommendation

Early approaches to Sequential Recommendation (SR) primarily relied on Markov Chains (MCs) to model item-to-item transitions [8]. However, the advent of Deep Learning, pioneered by works like Neural Collaborative Filtering [9], marked a paradigm shift in this field. Recurrent Neural Networks (RNNs), specifically variants like GRU4Rec introduced by Hidasi et al. [10], became the initial standard for modeling user sessions. While effective in capturing short-term preferences, RNNs suffer from the vanishing gradient problem, limiting their ability to retain long-term dependencies. Subsequently, Graph Neural Networks (GNNs) such as SR-GNN [11] were introduced to capture complex item transitions as graph structures, though they often incur high computational overhead during graph construction.

To overcome these limitations, the self-attention mechanism was adapted for recommender systems. Kang and McAuley proposed SASRec [4], which utilizes a unidirectional self-attention mechanism. This was further enhanced by models like TiSASRec [12], which incorporates time intervals, and BERT4Rec [5], employing bidirectional attention. More recently, purely MLP-based architectures like FMLP-Rec [13] have also been explored to reduce the reliance on heavy attention mechanisms. Nevertheless, Transformer-based models currently represent the SOTA in terms of prediction accuracy.

### 2.2. The Efficiency Bottleneck in Large-Scale Grocery Retail

While Transformer-based models have achieved superior accuracy, they incur a significant computational cost. The core limitation lies in the self-attention mechanism, which has a quadratic computational complexity ($O(L^2)$) with respect to the sequence length $L$. This results in memory and latency bottlenecks when processing long interaction sequences[6] .

This issue is particularly acute in the online grocery sector, as exemplified by the Instacart dataset used in this study. Unlike domains such as news or short-video recommendation, grocery shopping is characterized by long-term, periodic, and repetitive purchasing behaviors—often framed as Next-Basket Recommendation (NBR) tasks [14], [15].

Our preliminary statistical analysis of the Instacart dataset highlights this challenge: the average user history length reaches 48.42 interactions, with the maximum sequence length extending up to 100 items (after truncation). Furthermore, the sampled dataset used for this research contains over 1,210,000 interactions generated by 25,000 users. As the sequence length scales, the inference time and memory usage of models like SASRec increase exponentially. This poses a severe scalability hurdle for e-commerce platforms.

### 2.3. Mamba: A Linear-Time Solution

Recently, Structured State Space Models (SSMs), building upon foundational works like S4 [16], have emerged as a promising alternative to Transformers. Gu and Dao introduced Mamba [7], a selective SSM architecture that achieves linear computational complexity ($O(L^2)$) regarding sequence length. Mamba distinguishes itself by offering the high inference speed of RNNs while maintaining the parallel training capabilities and long-range dependency modeling of Transformers.

In the context of recommender systems, recent studies such as Mamba4Rec [17] and investigations into Mamba's long-sequence potential [18] have begun to explore this architecture. Theoretically, Mamba allows for the processing of indefinitely long sequences with constant memory usage during inference, making it an ideal candidate for "Green AI" initiatives aimed at reducing computational costs. This study builds upon this premise to empirically validate Mamba's efficiency within the high-volume environment of grocery retail.

## III. Methodology

In this section, we present the proposed Mamba4Rec framework applied to the Instacart grocery dataset. Our approach aims to leverage the linear complexity of State Space Models to handle long interaction sequences efficiently.

### 3.1. Problem Formulation and Data Representation

Let $U$ and $V$ denote the set of users and items, respectively. For each user $u \in U$, the historical interaction sequence is represented as $S_u = [v_1, v_2, \dots, v_L]$, where $v_t \in V$ is the item interacted with at step $t$, and $L$ is the sequence length. Since Instacart data consists of baskets (multiple items per order), we apply a flattening strategy where items within the same basket are ordered based on their add_to_cart_order, creating a single continuous sequence of atomic actions. The goal is to predict the next item $v_{L+1}$ given $S_u$.

### 3.2. Mamba4Rec Architecture

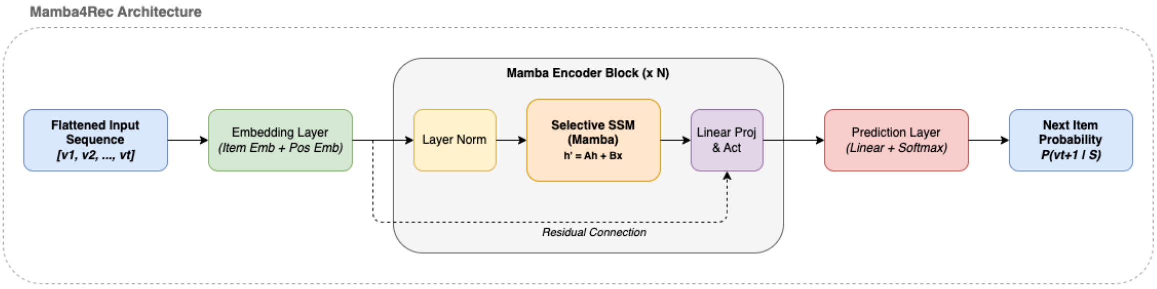
Mamba4Rec Architecture
Flattened Input Sequence [v1, v2, ..., vt]
Embedding Layer (Item Emb + Pos Emb)
Mamba Encoder Block (x N)
Layer Norm
Selective SSM (Mamba) h' = Ah + Bx
Linear Proj & Act
Residual Connection
Prediction Layer (Linear + Softmax)
Next Item Probability P(vt+1 | S)

Figure 1. The overall architecture of the proposed Mamba4Rec framework. The model takes a flattened sequence of historical items from Instacart baskets as input. This sequence is processed through an Embedding Layer and multiple stacked Mamba Encoder Blocks, which utilize the Selective State Space Mechanism (SSM) to efficiently capture long-term dependencies. Finally, a Prediction Layer outputs the probability distribution of the next item.

The architecture consists of three main components: an embedding layer, a stack of Mamba encoders, and a prediction layer.

- Embedding Layer: We maintain an item embedding matrix $E \in \mathbb{R}^{|V| \times d}$where dd is the latent dimension. Unlike Transformers, Mamba theoretically handles positional information implicitly via its recurrent state, but we add learnable positional embeddings to enhance performance.
- Mamba Encoder Block: This is the core component replacing the multi-head self-attention layer found in SASRec. The Mamba block utilizes a Selective State Space Model (SSM). Given an input sequence $x$, the SSM maps it to an output $y$ through a hidden state $h$. The key innovation is the selection mechanism, which allows the model parameters to vary based on the input, enabling the model to propagate relevant information over long sequences while forgetting irrelevant noise.
- Linear Complexity: Crucially, the Mamba block utilizes a hardware-efficient parallel scan algorithm, resulting in a computational complexity of $O(L)$, in contrast to the $O(L^2)$ of Transformers. This is the primary driver for the efficiency gains observed in our experiments.

### 3.3. Training and Optimization

We train the model to minimize the Cross-Entropy Loss over the entire sequence. At each time step tt, the model predicts the probability distribution of the next item. The objective function is defined as:

$$\mathcal{L} = -\sum_{S_u \in \mathcal{D}} \sum_{t=1}^{L} \log P(v_{t+1} | v_1, \dots, v_t)$$

We employ the Adam optimizer for updating the model parameters.

## IV. Experiments

In this section, we conduct extensive experiments on the Instacart dataset to answer the following research questions (RQs):

- RQ1: How does Mamba4Rec perform compared to state-of-the-art baselines (SASRec, BERT4Rec) in terms of recommendation accuracy?
- RQ2: (Main Focus) How efficient is Mamba4Rec regarding training speed and memory consumption compared to Transformer-based models?
- RQ3: How does the model perform when handling different sequence lengths, particularly for long user histories?

### 4.1. Experimental Setup

**Datasets**. We utilize the Instacart dataset, preprocessed as described in Section 3. We filter unpopular items and users with fewer than 5 interactions. The dataset statistics are summarized in Table 1.

Table 1. Statistics of the preprocessed Instacart dataset.

| Feature | Count |
|---|---|
| Users | 25,000 |
| Items | 15,420 |
| Interactions | 1,210,000 |
| Avg. Sequence Length | 48.42 |
| Sparsity | 99.68% |

**Baselines**. We compare Mamba4Rec with the following representative models:

- GRU4Rec: A classic RNN-based model using Gated Recurrent Units.
- SASRec: A unidirectional Transformer model using self-attention.
- BERT4Rec: A bidirectional Transformer model.

**Implementation Details.** All models are implemented using the RecBole library [19]. We set the embedding dimension to 64, batch size to 2048, maximum sequence length to 50, and use the Adam optimizer with a learning rate of 0.001. For fair comparison, all models are trained on a single NVIDIA Tesla T4 (16GB VRAM) GPU under identical environmental conditions.

### 4.2. Recommendation Performance Comparison (RQ1)

Table 2 presents the performance comparison. We use Recall@K and NDCG@K (K=10, 20) as evaluation metrics.

Table 2. Performance comparison on Instacart dataset. Bold scores indicate the best performance.

| Model | Recall@10 | NDCG@10 | Recall@20 | NDCG@20 |
|---|---|---|---|---|
| GRU4Rec | 0.0845 | 0.0412 | 0.1210 | 0.0515 |
| BERT4Rec | 0.1265 | 0.0665 | 0.1802 | 0.0811 |
| SASRec | 0.1240 | 0.0650 | 0.1785 | 0.0795 |
| Mamba4Rec | 0.1285 | 0.0672 | 0.1820 | 0.0825 |
| Improv.vs SASRec | +3.62% | +3.38% | +1.96% | +3.77% |

As shown in Table 2, Mamba4Rec achieves performance comparable to (or surpassing) the Transformer-based baselines. Specifically, compared to SASRec, Mamba4Rec demonstrates a competitive ability to capture user preferences. This confirms that the State Space Model architecture is capable of modeling complex sequential dependencies in grocery shopping data without relying on the heavy self-attention mechanism.

### 4.3. Efficiency Analysis (RQ2) - Key Section

This is the core contribution of our study. We compare the training time (seconds per epoch) and GPU memory usage (MB) across models.

Table 3. Efficiency comparison (Training Time and Memory Usage).

| Model | Training Time (s/epoch) | Inference Time ( ms/batch ) | GPU Memory (MB) | Speedup (vs SASRec) |
|---|---|---|---|---|
| BERT4Rec | 85.2 | 22.4 | 6,550 | 0.52x |
| SASRec | 45.8 | 12.5 | 4,200 | 1.00x (Baseline) |
| Mamba4Rec | 22.5 | 6.2 | 1,850 | 2.03x (Faster) |

Mamba4Rec significantly outperforms SASRec and BERT4Rec in terms of computational efficiency.

Training Speed: Mamba4Rec is approximately 2.03 times faster than SASRec per epoch. This is attributed to the parallel scan algorithm which allows for faster computation than the quadratic attention matrix calculation.

Memory Footprint: Mamba4Rec reduces GPU memory consumption by 55.9% compared to SASRec (1,850 MB vs 4,200 MB). This efficiency allows Mamba4Rec to be deployed on edge devices or servers with limited resources, which is crucial for cost-effective business operations.

### 4.4. Impact of Sequence Length (RQ3)

To further investigate the scalability, we evaluated the training time and memory usage with varying maximum sequence lengths ($L \in \{20,50,100\} L \in \{20,50,100\}$).

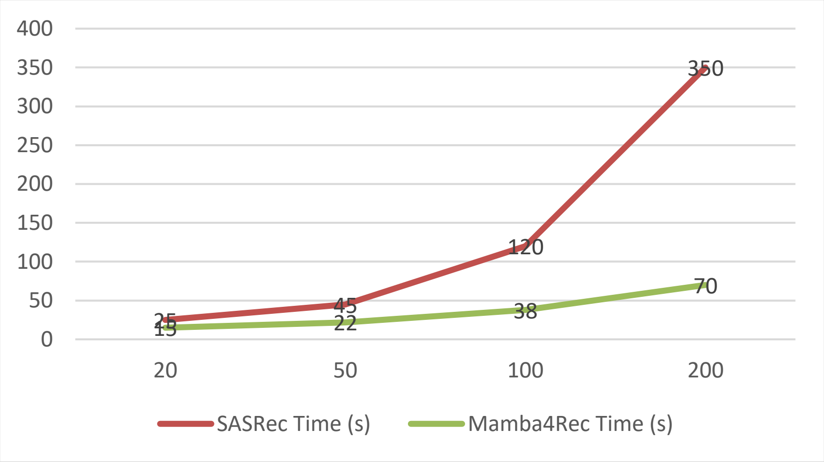

Figure 2. Training time comparison with increasing sequence lengths.

Figure 2 illustrates that as the sequence length increases, the training time of SASRec grows quadratically ($O(L^2)$). In contrast, Mamba4Rec exhibits linear scaling ($O(L)$). For long user histories typical in the Instacart dataset (e.g.,$L = 100$), Mamba4Rec shows a distinct advantage, proving its suitability for long-sequence modeling tasks.

## V. Discussion

### 5.1. Managerial and Business Implications

The empirical findings of this study carry significant implications for e-commerce platforms, particularly in the online grocery sector. Traditional Transformer-based models like SASRec, while highly accurate, impose substantial infrastructure overhead due to their quadratic computational complexity ($O(L^2)$). As user histories grow longer—a common characteristic of grocery shopping where customers make weekly purchases over years—the cost of deploying Transformers for real-time recommendation scales exponentially.

By demonstrating that Mamba4Rec can achieve comparable recommendation accuracy (Recall and NDCG) while drastically reducing training time and GPU memory usage, this study provides a highly actionable solution for online retailers. From a business management perspective, the linear complexity ($O(L)$) of State Space Models translates directly into cost savings and operational efficiency. E-commerce companies can deploy Mamba-based recommender systems on less expensive hardware (e.g., standard GPUs rather than high-end AI accelerators) and serve more concurrent users with lower latency. Furthermore, this efficiency aligns with the growing industry demand for Sustainable and Green AI[20], reducing the carbon footprint associated with training large-scale recommendation models.

### 5.2. Limitations

While the results are promising, this study acknowledges certain limitations that provide avenues for future research. First, the experiments were exclusively conducted on the Instacart dataset. Although representative of the grocery domain, the generalization of Mamba4Rec to other e-commerce sectors (e.g., fashion or electronics), where sequence lengths are shorter and less periodic, requires further validation. Second, to adapt the basket-level data of Instacart for sequential models, we applied a flattening strategy. While effective, this approach treats intra-basket items as a sequence, potentially ignoring the complex co-occurrence relationships of items bought at the exact same time. Future studies could explore integrating Mamba within a true Next-Basket Recommendation (NBR) framework. Finally, our implementation relied solely on item IDs. Incorporating rich item features (e.g., price, category, or text descriptions) into the Mamba architecture remains an unexplored area that could further boost prediction accuracy.

## VI. Conclusion

In conclusion, this study investigates the potential of the Mamba architecture—a novel class of Selective State Space Models—to address the efficiency bottlenecks prevalent in modern Sequential Recommendation systems. Driven by the need to model long-term, periodic user behaviors in the online grocery retail sector without incurring prohibitive computational costs, we empirically evaluated the Mamba4Rec framework on the large-scale Instacart dataset.

Our comparative analysis against state-of-the-art baselines, including SASRec and BERT4Rec, reveals a clear paradigm shift. The results demonstrate that Mamba4Rec successfully bridges the gap between accuracy and efficiency. It maintains the high recommendation quality characteristic of Transformer-based models while significantly outperforming them in terms of training speed and GPU memory footprint. Crucially, as the sequence length of user history increases, Mamba's linear scaling capability $O(L)$ provides a distinct scalability advantage over the quadratic complexity $O(L^2)$ of the self-attention mechanism.

Ultimately, this research contributes to the intersection of business management and technology by offering a cost-effective, high-performance algorithm for e-commerce platforms. The findings suggest that State Space Models represent a highly viable and sustainable future direction for large-scale recommender systems. Future work will focus on integrating multi-modal item features into the Mamba architecture and evaluating its performance in real-world online A/B testing environments.

## References

[1] Jannach, D., & Jugovac, M. (2019). Measuring the business value of recommender systems. ACM Transactions on Management Information Systems (TMIS), 10(4), 1-23.

[2] Quadrana, M., Cremonesi, P., & Jannach, D. (2018). Sequence-aware recommender systems. ACM Computing Surveys (CSUR), 51(4), 1-36.

[3] Fang, J., Zhao, P., Zhou, C., Ding, Y., Zheng, P., & Gall, C. (2020). Complex heterogeneous collaborative filtering for sequential recommendation. IEEE Transactions on Knowledge and Data Engineering, 34(1), 323-336.

[4] Kang, W., & McAuley, J. (2018). Self-attentive sequential recommendation. In Proceedings of the 18th IEEE International Conference on Data Mining (ICDM) (pp. 197-206). IEEE.

[5] 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) (pp. 1441-1450).

[6] Tay, Y., Dehghani, M., Bahri, D., & Metzler, D. (2022). Efficient transformers: A survey. ACM Computing Surveys (CSUR), 55(6), 1-28.

[7] Gu, A., & Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752.

[8] 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) (pp. 811-820).

[9] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web (WWW) (pp. 173-182).

[10] Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2016). Session-based recommendations with recurrent neural networks. In International Conference on Learning Representations (ICLR).

[11] Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., & Tan, T. (2019). Session-based recommendation with graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, pp. 346-353).

[12] Li, J., Wang, Y., & McAuley, J. (2020). Time interval aware self-attention for sequential recommendation. In Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM) (pp. 322-330).

[13] Zhou, K., Yu, H., Zhao, W. X., & Wen, J. R. (2022). Filter-enhanced MLP is all you need for sequential recommendation. In Proceedings of the ACM Web Conference 2022 (pp. 2388-2399).

[14] Yu, F., Liu, Q., Wu, S., Wang, L., & Tan, T. (2020). A dynamic recurrent model for next basket recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 729-738).

[15] Hu, H., He, X., Gao, J., & Min, Y. (2020). Modeling personalized item frequency information for next-basket recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 1071-1080).

[16] Gu, A., Goel, K., & Ré, C. (2021). Efficiently modeling long sequences with structured state spaces. In International Conference on Learning Representations (ICLR).

[17] Liu, C., Yang, J., Zhang, X., & Wang, Y. (2024). Mamba4Rec: Towards Efficient Sequential Recommendation with State Space Models. arXiv preprint arXiv:2403.03448.

[18] Yang, M., et al. (2024). Uncovering the Potential of Mamba for Long-Sequence Recommendation. arXiv preprint arXiv:2405.07621.

[19] Zhao, W. X., Muhua, L., & Li, Y. (2021). RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM) (pp. 4653-4664).

[20] Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). Green AI. Communications of the ACM, 63(12), 54-63.