INTEGRATING HIGH-UTILITY AND PERIODIC PATTERNS WITH DEEP LEARNING FOR NEXT BASKET RECOMMENDATION

Le Bao Minh Duy*, **Tran Minh Thai**

Faculty of Information Technology,
Ho Chi Minh City University of Foreign Languages – Information Technology, Vietnam

22dh110529@st.huflit.edu.vn, thaitm@huflit.edu.vn

ABSTRACT— Next Basket Recommendation (NBR) implies the task of predicting users' future purchases based on analyzing historical transaction sequences. Current deep learning approaches have achieved variety of significant progress, yet they still have limitations in handling sequences with various lengths, or baskets with dense items, as well as struggling to integrate explicit signals such as item utility and purchase periodicity. This paper proposes HybridSPMF, a sequential pattern mining "SPM-First" framework, which orients neuralizing classical SPM concepts into a learnable architecture. To optimize processing of high-density transactions (on a scale of 9 to 91 items per shopping cart), the framework applied a Hierarchical Memory Pool and Adaptive Basket Compression. Experimental results on four retail datasets demonstrate that HybridSPMF achieves improved performance. Specifically, on the Ta-Feng dataset, the model recorded a 56.6% improvement in UtilityRecall@20 and a 24.1% increase in NDCG@20. These findings confirm that neural network modeling of data mining patterns works efficiently in both recommendation systems and business-oriented values.

Keywords— Next Basket Recommendation, Neuralizing, High-Utility Sequential Pattern Mining, Periodic Pattern Mining, Gated Refinement Mechanism, Hierarchical Memory Networks, Adaptive Basket Compression, Deep Learning.

I. INTRODUCTION

The continuous increase in transaction data on e-commerce platforms demands advancements in recommendation systems. Next Basket Recommendation (NBR) plays an essential role in predicting future shopping behavior by researching data from past shopping carts. Compared to conventional paradigms, NBR presents greater methodological complexity because it necessitates the simultaneous modeling of multiple factors, such as transaction chains, preferred behaviors, time characteristics, and the utilities of each product.

A. LIMITATIONS OF EXISTING APPROACHES

Current approaches to NBR fall into two categories.

Deep learning approaches apply recurrent neural networks (RNNs) and Transformer architectures to sequential recommendation problems. A typical example is the DREAM model [1] which uses an RNN structure to model dependency relationships at the shopping cart level. However, these methods face implementation challenges when dealing with diverse datasets. In this paper, we experiment on diverse datasets to simulate different scenarios of transaction chain size and cart information density. For instance, the Dunnhumby dataset has an average of 91 shopping carts per user, while the Ta-Feng dataset has only 9.6. Traditional RNN models frequently suffer from gradient vanishing when processing extended histories, which causes the degradation of critical sequential patterns from early transactions. Another significant problem is shopping cart density, which varies significantly between datasets (Online Retail averages 23 items per cart compared to 6 for Ta-Feng). In particular, the attention mechanism becomes computationally expensive when processing high-density shopping carts, and risks diluting focus on important items. Furthermore, most neural network methods treat all items equally, ignoring explicit useful information – such as the product of price and quantity – which reflects product importance.

Pattern mining approaches, including Sequential Pattern Mining (SPM) and its extended formats, High Utility Sequential Pattern Mining (HUSPM), and Periodic Pattern Mining (PPM), provide foundational and theoretical frameworks for discovering valuable patterns and enriching rules. Typical algorithms such as PrefixSpan [2] and USpan [3] allow for the extraction of highly interpretable rules. However, this group of methods reveals inherent limitations, including rigid pattern matching mechanisms that lack flexibility for previously unseen patterns, scalability issues on large-scale databases, and barriers to direct integration into neural network architectures.

B. PROPOSED APPROACH: SPM-FIRST WITH ADAPTIVE ENHANCEMENT

We propose HybridSPMF based on the fundamental assumption that sequential pattern modeling should serve as the primary module, while utility and periodic signals function as enhancements of the predictions, not competitors with the main forecasts. This approach differs from traditional hybrid methods that treat components as independent scoring branches, such as multimodal architecture. In our architecture, the Core Sequential Encoder remains active; a hierarchical memory-augmented GRU exploits dependency relationships

* Coressponding Author

across the baskets as a priority signal. The combination of Utility Refinement and Periodic Refinement modules adjusts signals to optimize sequential forecast results, rather than establishing separate scoring scales. Finally, the Learnable Integration Mechanism performs synchronous integration of components through trainable gates, as shown in Equation (1)

$$final\_score = sequence\_score + \alpha \cdot utility\_boost + \beta \cdot periodic\_boost \tag{1}$$

where α and β are learnable parameters initialized at 0.3.

C. CONTRIBUTIONS

This research advances the field through four core contributions. First, we propose the SPM-First Architecture. In this model, sequential pattern modeling plays a core role, while utility and periodicity elements are integrated as learning-capable refinement components. Second, we design a Hierarchical Memory Pool with a three-level storage system. This solution allows for efficient processing of sequences from 9 to 91 baskets without being affected by gradient degradation during training. Third, we introduce an Adaptive Basket Compression mechanism based on the top-k attention technique. This technique optimizes the processing of high-density baskets (up to 23 items) while preserving essential information. Finally, we conduct a comprehensive evaluation on four diverse datasets. Accompanying analysis (ablation studies) also clearly confirm the contributing role of each component in the system.

D. PAPER ORGANIZATION

The study is structured as follows. Part II introduces the fundamental concepts and formal definitions of sequential pattern mining (SPM), high-utility sequential pattern mining (HUSPM), top-k mining, and periodic pattern mining (PPM); these definitions form the crucial theoretical foundation of the proposed method. Part III reviews related works on sequential pattern mining, deep learning applications in sequential recommendation systems, and hybrid approaches. Part IV specifies the architecture of the HybridSPMF model, states the problem, details its constituent components, and analyzes the functionality of these components. Part V presents experimental setup, a control system, evaluation indices, in-depth analysis of results, and ablation studies. Finally, Part VI summarizes the contributions of the paper and giving out suggestions for future research.

II. PRELIMINARIES

This section introduces the fundamental concepts and formal definitions that form the basis for the proposed architectural framework. We clarify the definitions of sequential pattern mining, high-utility sequential pattern mining, top-k mining, and periodic pattern mining. These definitions provide the necessary theoretical foundation for the neuralized module featured in the HybridSPMF model.

A. SEQUENTIAL PATTERN MINING

*Definition 1 (**Sequence Database**)*: Let $\mathfrak{I} = \{i_1, i_2, \dots, i_m\}$ be a set of distinct items. An *itemset* $X \subseteq \mathfrak{I}$ is a non-empty subset of items. A *sequence* $s = \langle X_1, X_2, \dots, X_n \rangle$ is an ordered list of itemsets. A *sequence database* $\mathcal{D} = \{s_1, s_2, \dots, s_n\}$ is a collection of sequences.

*Definition 2 (**Subsequence and Support**)*: A sequence $\alpha = \langle A_1, A_2, \dots, A_k \rangle$ is a *subsequence* of sequence $\beta = \langle B_1, B_2, \dots, B_l \rangle$, denoted $\alpha \sqsubseteq \beta$, if there exist integers $1 \le j_1 < j_2 < \cdots < j_k \le l$ such that $A_1 \subseteq B_{j1}, A_2 \subseteq B_{j2}, \dots, A_k \subseteq B_{jk}$. The *support* of a sequence α in database $\mathcal{D}$ is defined as follows.

$$sup(\alpha) = |\{s \in \mathcal{D} : \alpha \sqsubseteq s\}| \tag{2}$$

*Definition 3 (**Sequential Pattern Mining**)*: Given a sequence database $\mathcal{D}$ and a minimum support threshold min_sup, the Sequential Pattern Mining (SPM) problem is to discover all sequences α such that $sup(\alpha) \ge min\_sup$. Such sequences are called *frequent sequential patterns* [4].

The seminal PrefixSpan algorithm [2] addresses this problem using pattern-growth methodology, which projects the database by frequent prefixes to avoid expensive candidate generation. This approach has become the foundation for many subsequent SPM algorithms.

B. HIGH-UTILITY SEQUENTIAL PATTERN MINING

Traditional SPM methods usually treats all items equally based on frequency, regardless their actual business value. To overcome this drawback, High Utility Sequential Pattern Mining (HUSPM) intergrate utility value to reflect the weight of each item in the transaction chain [3],[5].

*Definition 4 (**Utility of an Item**)*: Each item $i \in \mathfrak{I}$ is associated with an external utility $p(i)$ representing its unit profit or importance. For an item i appearing in itemset X_j of sequence s, its internal utility $q(i, X_j, s)$ represents the quantity purchased. The utility of item i in X_j is computed as follows.

$$u(i, X_j, s) = p(i) \times q(i, X_j, s) \tag{3}$$

*Definition 5 (**Utility of a Sequence**)*: The utility of an itemset X_j in sequence s is $u(X_j, s) = \sum_{i \in X} u(i, X_j, s)$. The utility of sequence s is $u(s) = \sum_{X_j \in s} u(X_j, s)$. For a sequential pattern α occurring in s, its utility $u(\alpha, s)$ is the sum of utilities of itemsets in s that match α.

*Definition 6 (**High-Utility Sequential Pattern Mining**)*: Given a sequence database $\mathcal{D}$ with utility values and a minimum utility threshold minUtil, the High-Utility Sequential Pattern Mining (HUSPM) problem is to discover all sequences α such that the following condition holds.

$$u(\alpha) = \sum_{s \in \mathcal{D} \,:\, \alpha \sqsubseteq s} u(\alpha, s) \geq \text{minUtil} \tag{4}$$

Such sequences are called high-utility sequential patterns [3].

Unlike frequency-based support, utility does not satisfy the downward closure property, making HUSPM more challenging. Algorithms like USpan [3] and HUS-Span [6] address this through upper-bound pruning strategies.

C. TOP-K MINING

Setting appropriate thresholds for SPM and HUSPM is often difficult in practice. Top-k mining provides an alternative formulation that avoids explicit threshold specification [7].

*Definition 7 (**Top-k Sequential Pattern Mining**)*: Given a sequence database $\mathcal{D}$ and an integer k, top-k sequential pattern mining discovers the k patterns with the highest support or utility, without requiring a predefined threshold.

Top-k approaches dynamically raise the threshold during mining, starting from zero and progressively increasing as patterns are discovered. This strategy is particularly useful when users do not have prior knowledge about appropriate threshold values. Within the HybridSPMF framework, the Adaptive Basket Compression module (Section IV-D) applies a top-k mechanism to select the k most important items from high-density carts. This process is based on important scores and specified in Equation (11).

D. PERIODIC PATTERN MINING

Periodic Pattern Mining (PPM) not only help to capture temporal regularity in sequential data, but it is also a key element in modeling habitual shopping behaviors [8], [9].

*Definition 8 (**Period and Periodicity**)*: Given a sequence s with timestamps, the period τ of a pattern α represents the regular time interval between consecutive occurrences of α in s. The periodicity measures how consistently α occurs at interval τ.

*Definition 9 (**Periodic Sequential Pattern Mining**)*: Given a sequence database $\mathcal{D}$ with timestamps, a maximum period threshold *max_per*, and a minimum support threshold *min_sup*, Periodic Pattern Mining (PPM) discovers all patterns α that satisfy both the support constraint and have a period not exceeding *max_per* [8].

PHM [10] combines periodic and high-utility constraints to mine patterns that are both temporally regular and economically valuable. In HybridSPMF, we neuralize periodic concepts through the Periodic Refinement module described in Section IV, computing period alignment scores as defined in Equation (20).

E. RELATIONSHIP TO NEURAL APPROACHES

Traditional pattern mining algorithms rely on explicitly enumerating patterns using both combinatorial search and pruning mechanisms. In contrast, our neuralized approach learns implicit pattern representations using gradient based optimization. Table 1 summarizes the key correspondences relationship between classical mining concepts and their neural components in HybridSPMF architecture. This approach supports end-to-end learning while maintaining the interpretable structure of classical mining concepts.

Table 1. Correspondence Between Mining Concepts And Neural Components

Mining Concept	Neural Component	Reference
Sequential patterns	GRU hidden states	Sec. IV
Support counting	Frequency embeddings	Eq. (14)
Utility weighting	Utility refinement gate	Eq. (18)
Periodic alignment	Cosine-based scoring	Eq. (20)
Top-k selection	Adaptive compression	Eq. (11)

III. RELATED WORK

A. SEQUENTIAL PATTERN MINING

Sequential Pattern Mining (SPM) performs the task of detecting frequently occurring sub-sequential sequences in transaction databases. The pioneering research of Agrawal and Srikant [4] proposed the AprioriAll algorithm as the first baseline for this problem. PrefixSpan [2] , with its pattern-growth method, created a breakthrough for SPM in optimizing mining performance. This technique projects the database according to frequently occurring prefixes to eliminate the costly candidate generation process. . Although achieving high performance on long sequences with low memory consumption, the classic limitation of SPM is always focusing only on frequency and ignoring the weight of each item. Comprehensive evaluations of the algorithm and applications of SPM can be found in the articles [7], [11].

High Utility Sequential Pattern Mining (HUSPM) extends the SPM algorithm by adding utility values, such as profit and quantity, to each item [5]. Unlike frequency-based mining, which relies on counting methods, HUSPM identifies patterns whose total utility exceeds a defined threshold (details in Definition 6). A pioneering study in this field, USpan [3] used lexicographic sequence trees combined with utility-based pruning. HUS-Span [6] optimized performance through tighter upper bounds. These methods help to extract patterns with practical business value. However, setting an inappropriate utility thresholds can easily lead to explosion in number of patterns or failure in extract any patterns.

Periodic pattern mining (PPM) helps to detect patterns that appear at regular, cyclical intervals. This method is particularly effective in simulating repetitive shopping behaviors, such as customers' weekly grocery shopping habits (Definition 9). Algorithms like PHM [10] perform highly useful periodic pattern mining by combining both time patterns with item weight constraints. Although PPM effectively describes cyclical consumer behavior, it is still depended on the accuracy of cycle estimation and struggle to deal with irregular shopping breaks. Instead of listing explicitly, we will embed utility and periodic signals into learnable neural components, as discussed in Section II-E.

B. DEEP LEARNING FOR SEQUENTIAL RECOMMENDATION

Neural network techniques have significantly improved sequential recommendation system by their ability to extract complex nonlinear rules.

PersonalPop (Personalized Popularity) [12] is a simple but effective baseline, especially in cases where users repurchase old items frequently; the model's idea is to rank items based on their past purchase frequency. TIFU-KNN (Time-Frequency User-KNN) [13] leverages the preferences of similar users, while prioritizing recent shopping behaviors through a time-based weighting reduction mechanism. FPMC (Factorizing Personalized Markov Chains) [14] combines the ability to analyze long-term preferences and short-term behavioral switching habits of users through coordinated matrix and Markov chain decomposition.

HRM (Hierarchical Representation Model) [15] employs a two-level architecture to aggregate item information in the basket and applies GRU to the entire sequence chain. DREAM (Dynamic REcurrent bAsket Model) [1] improves the RNNs by updating continously user characteristics after finishing each transaction, effectively capturing repeative consumption patterns. NARM (Neural Attentive Recommendation Machine) [16] intergrates attention into the session-based recommendation model to adjust between overall purchase intent and the most recent specific behaviors. RepeatNet (A Repeat-Aware Neural Recommendation Machine) [17] trades off the conflict between repurchasing old items and discovering new items (repeat-explore) by using a flexible switching between these two modes.

SASRec (Self-Attentive Sequential Recommendation) [18] applies the unidirectional Transformer architecture and self-attention to capture long-term dependencies without the need of recursive structures. Based on this foundation, we adopt the BasketTransformer architecture proposed in BTBR (Transformer-based model for Next Novel Basket Recommendation) [19] to focus on exploiting features at the basket level; we keep this name for consistency in the baseline system. Finally, contrastive learning method is represented by BCL (Behavior-aware Contrastive Learning) [20] through the method of constructing behavior-aware positive and negative views; within our experiments, we denote this method as BEACON, a naming choice used solely for consistency and directly corresponding to the original BCL formulation.

C. HYBRID AND KNOWLEDGE-ENHANCED APPROACHES

Recent works bridge pattern mining with deep learning through rule-enhanced methods and knowledge graph approaches. However, most existing hybrid methods treat utility and periodic signals as parallel scoring branches that compete with sequential predictions. We adopt an alternative approach where sequential modeling serves as the primary module, while mining-inspired signals function as refinements that adjust predictions for already-relevant items rather than independently scoring candidates.

IV. METHODOLOGY

A. PROBLEM DEFINITION

Let $\mathcal{U}$ denote the set of users and $\mathfrak{I}$ the set of items. For each user $u \in \mathcal{U}$, we observe a sequence of historical baskets given by Equation (5).

$$\mathcal{H}_u = (B_1, B_2, \dots, B_{K_u}) \tag{5}$$

In this formulation, K_u is the number of baskets for user u, and each basket $B_k \subseteq \mathfrak{I}$ is a set of items purchased together. This corresponds to the sequence database definition in Definition 1, where each user's history forms a sequence and each basket corresponds to an itemset. Associated with each basket are utility values $U_k = \{u_{k,i} : i \in B_k\}$ where $u_{k,i}$ represents the utility of item i in basket k, computed as the product of price and quantity following Definition 4, and timestamp τ_k indicating when basket k was purchased. The time interval between consecutive baskets is $\Delta t_k = \tau_k - \tau_{k-1}$. The task is formulated as follows: given $\mathcal{H}_u$, predict a ranked list of items for the next basket B_{K_u+1}.

Table 2 summarizes the key notation used throughout this paper.

Table 2. Notation Summary

Symbol	Description	Typical Values
B	Batch size	64
L	Maximum sequence length (baskets)	50
S	Maximum basket size (items)	30
D	Embedding dimension	128
H	Hidden dimension	256
α, β	Learnable refinement gates	Init: 0.3
K	Top-K for evaluation	{10, 20, 50}
k_{max}	Top-k for basket compression	15

B. ARCHITECTURE OVERVIEW

Based on the challenges pointed out in Section I, we propose HybridSPMF, a framework with four main components to address long sequence degradation, dense basket overload, and missing utility signals. Fig. 1 illustrates the overview of the system's architecture.

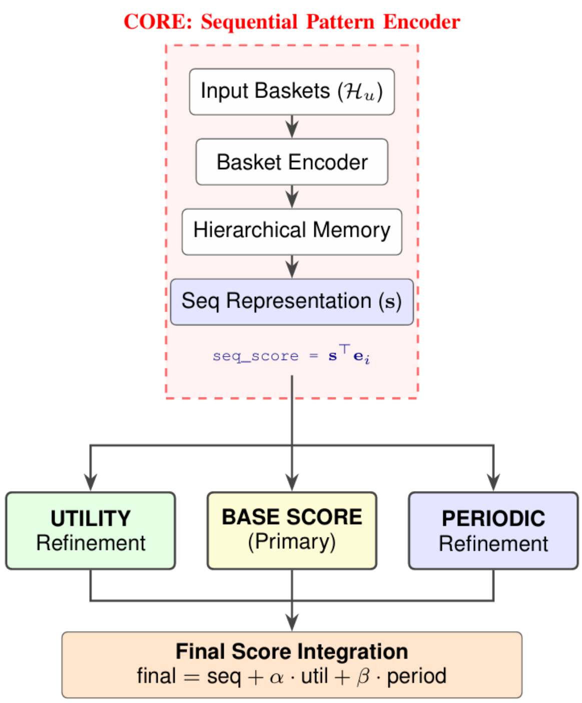

Figure 1. HybridSPMF architecture overview. The sequential encoder processes cart chains through hierarchical memory to generate the primary signal *seq_score*. Utility and periodic modules generate refinement signals that adjust predictions through learnable gates α and β.

The inference pipeline proceeds through five stages. First, raw baskets are encoded via the Hierarchical Memory Pool to handle variable sequence lengths. Second, dense baskets are compressed through the Adaptive Basket Compression module using the top-k selection mechanism defined in Section II-C. Third, the SPM Sequence Encoder processes the compressed representations to produce sequential scores, capturing patterns analogous to those discovered by traditional SPM as described in Definition 3. Fourth, Utility and Periodic Refinement modules generate adjustment signals inspired by HUSPM as in Definition 6 and PPM as in Definition 9, respectively. Finally, all components are combined through learnable gates to produce the final score.

C. HIERARCHICAL MEMORY POOL

Real-world transaction histories vary significantly in length, and a fixed-length encoder either wastes capacity on short sequences or truncates important signals from long ones. As shown in Table 3, Dunnhumby users average 91 baskets while Ta-Feng users average only 9.6 baskets. Standard RNN architectures struggle with such long sequences due to gradient vanishing, causing important patterns from early purchase history to be lost.

The Hierarchical Memory Pool addresses this challenge through three memory levels. The Recent Memory with window size $W_R = 10$ provides full attention to the 10 latest baskets as formulated in Equation (6).

$$m_{recent} = \frac{1}{W_R} \sum_{t=L-W_R}^{L} b_t \tag{6}$$

The Medium-Term Memory with window size $W_M = 20$ compresses baskets from positions $L - W_R - W_M$ to $L - W_R$ as given by Equation (7).

$$m_{medium} = f_{compress}\left(\frac{1}{W_M} \sum_{t=L-W_R-W_M}^{L-W_R} b_t \right) \tag{7}$$

The Long-Term Prototype Memory with $P = 8$ prototypes employs prototype-based summarization using multi-head attention as expressed in Equation (8).

$$m_{long} = MHA\big(Q_{proto}, B_{early}, B_{early}\big) \tag{8}$$

The three levels are combined through learned gates according to Equation (9).

$$m_u = g_1 \cdot m_{recent} + g_2 \cdot m_{medium} + g_3 \cdot m_{long} \tag{9}$$

D. ADAPTIVE BASKET COMPRESSION

Similar to sequence length variability, the density of each basket varies dramatically between 4 datasets. Online Retail has an averages of 23 items per cart, almost four times higher than Ta-Feng's 6 items. Processing all items equally can easily lead to information overload and unnecessary computation costs. To overcome this problem, the Adaptive Basket Compression module uses a selective attention method with time complexity $O(|B_t| \log k_{max})$ for screening step and $O(k_{max}^2 D)$ for attention aggregation step.

Specifically, each item is assigned an importance score derived from its latent embedding vector, as formalized in Equation (10).

$$s_i = W_2 \cdot GELU(W_1 e_i) \tag{10}$$

For baskets with more than $k_{max} = 15$ items, we select the top-k most important items according to Equation (11), implementing the top-k mining concept from Definition 7 in a neural context.

$$\mathfrak{T}_{selected} = \mathrm{TopK}\big(\{s_i\}_{i \in B_t}, k_{\max}\big) \tag{11}$$

Selected items are then aggregated through multi-head attention with a learnable query as formulated in Equation (12).

$$b_t = MHA(q_{basket}, E_{selected}, E_{selected}) \tag{12}$$

For baskets with fewer than k_{max} items, we simply use mean pooling as given by Equation (13).

$$b_t = \frac{1}{|B_t|} \sum_{e \in B_t} e_i \tag{13}$$

E. SPM SEQUENCE ENCODER

The SPM Sequence Encoder serves as the primary module of HybridSPMF, always active regardless of ablation configuration. This module captures order-dependent co-occurrence patterns across the basket sequence through three components, effectively learning representations analogous to sequential patterns discovered by traditional SPM algorithms as described in Definition 3.

Beyond standard learned embeddings $e_i \in \mathbb{R}^D$, we incorporate explicit frequency features as expressed in Equation (14). This corresponds to incorporating support information from Equation (2) into the neural representation.

$$e_i' = e_i + f_{freq}(freq_i) \tag{14}$$

In this equation, $f_{freq} : \mathbb{R} \to \mathbb{R}^D$ is a two-layer MLP that projects the item's global purchase frequency into embedding space.

We employ a single-layer GRU [21] to model temporal dependencies as given by Equation (15).

$$h_t = GRU(b_t, h_{t-1}), \qquad h_0 = 0 \tag{15}$$

In this formulation, $b_t \in \mathbb{R}^D$ is the compressed basket representation and $h_t \in \mathbb{R}^H$ is the hidden state.

The final GRU hidden state combines with hierarchical memory through a linear projection as formulated in Equation (16).

$$s = W_{fuse}\left[h_L \parallel m_{pool}\right] + h_L \tag{16}$$

Here $W_{fuse} \in \mathbb{R}^{H\times(H+D)}$ and $[\cdot\|\cdot]$ denotes concatenation. The final representation produces item scores via dot product as given by Equation (17).

$$seq\_score_i = s^{\top} e_i \tag{17}$$

F. UTILITY AND PERIODIC REFINEMENT

In HybridSPMF, utility and periodic signals are treated as refinement modules rather than independent scoring branches. This design ensures that sequential patterns remain the primary prediction signal, that utility and periodic modules only adjust items already considered relevant, and that ablation can clearly isolate each module's contribution.

The Utility Refinement Module captures high-utility patterns inspired by HUSPM as defined in Definition 6 [5]. For each item i, we compute historical utility following Definition 4 as the cumulative transaction value $u_i^{hist} = \sum_{k\,:\,i\in B_k}\left(price_i \times quantity_{k,i}\right)$. The utility adjustment uses a gated sigmoid activation with learned parameters as formulated in Equation (18).

$$util\_adjust_i = \alpha \cdot \sigma(W_g s) \cdot s^{\top} f_{util}\left(\left[e_i \| \log\left(1 + u_i^{hist}\right)\right]\right) \tag{18}$$

In this equation, α is a learnable scalar initialized at 0.3, and $\sigma(W_g s) \in (0,1)$ is a context-dependent gate, and $f_{util} : \mathbb{R}^{D+1} \to \mathbb{R}^H$ is a linear projection.

The Periodic Refinement Module captures regular purchase cycles inspired by PPM as defined in Definition 9 [8]. For each item i, we estimate the purchase period following Definition 8 using the average inter-purchase interval as given by Equation (19).

$$\bar{\tau}_i = \frac{1}{n_i - 1}\sum_{j=1}^{n_i - 1}\left(t_{j+1} - t_j\right) \tag{19}$$

In this formulation, $t_1, \ldots, t_{ni}$ are the timestamps of the user's purchases of item i. Items with fewer than 2 purchases have undefined periodicity and receive zero adjustment.

Items receive an adjustment when due for repurchase. We use cosine-based alignment to measure how close the current time is to the expected repurchase time as expressed in Equation (20).

$$alignment_i = \frac{1 + \cos\left(2\pi\left(\frac{\Delta t_i}{\bar{\tau}_i} - \left\lfloor\frac{\Delta t_i}{\bar{\tau}_i}\right\rceil\right)\right)}{2} \tag{20}$$

Here Δt_i is days since last purchase and $\lfloor\cdot\rceil$ denotes rounding to the nearest integer. This produces values in [0, 1], with 1 indicating perfect alignment with the expected cycle. The periodic adjustment is formulated in Equation (21).

$$period\_adjust_i = \beta \cdot \sigma(W_p s) \cdot alignment_i \tag{21}$$

In this equation, β is a learnable scalar initialized at 0.3 and $\sigma(W_p s)$ is a context-dependent gate.

The final score combines sequential predictions with refinement adjustments according to Equation (22).

$$score_i = seq\_score_i + util\_adjust_i + period\_adjust_i \tag{22}$$

When a refinement module is disabled during ablation, its adjustment becomes zero, allowing isolation of each module's contribution.

G. TRAINING OBJECTIVE

We use Binary Cross-Entropy (BCE) loss with negative sampling. This loss function is fitted for next basket recommendation because it can handle the multi-label nature of basket prediction and allow efficientive training without computing scores for all items. The loss function is defined in Equation (23).

$$\mathcal{L} = -\frac{1}{|\mathcal{B}|} \sum_{(u,\, i^+,\, \mathfrak{X}^-)} \left[log\, \sigma(score_{i^+}) + \sum_{j \in \mathfrak{X}^-} log\left(1 - \sigma\left(score_j\right)\right) \right] \tag{23}$$

In this formulation, i^+ denotes positive items in the ground-truth basket, $\mathfrak{X}^-$ denotes sampled negative items not in the basket, and σ is the sigmoid function.

We use popularity-biased negative sampling with sampling [22] probability $P(i) \propto freq_i^{0.5}$, where $freq_i$ is the global purchase frequency of item i. This sampling strategy provides harder negatives than uniform sampling, as popular items are more likely to be confused with true positives. We sample 5 negatives per positive item.

V. EXPERIMENTS

In this section, we describe the setup for experiment and the method of how we evaluate the results. We conducted experiments to answer two research questions: (RQ1) How does HybridSPMF compare against competitive baselines? (RQ2) What is the contribution of each component?

A. DATASETS

We evaluated on four retail datasets with diverse characteristics summarized in Table 3.

Table 3. Dataset Statistics

Dataset	Users	Items	Baskets	Bskt/Usr	Itm/Bskt	Rep(%)
Ta-Feng	7,619	21,509	73,114	9.59	6.00	18.2
Online Retail	2,060	4,589	31,374	15.23	23.34	42.7
Dunnhumby	2,471	91,967	225,237	91.16	8.52	67.3
Instacart	115,909	49,480	2,117,386	18.27	10.13	60.8

Ta-Feng contains Taiwan grocery data from 2000 to 2001, representing short purchase histories where the Hierarchical Memory Pool described in Section IV-C is designed to maximize limited signals from users with few transactions. Online Retail contains UK e-commerce data from 2010 to 2011 with high basket density, where the Adaptive Basket Compression described in Section IV-D is designed to handle baskets with many items. Dunnhumby contains US retail loyalty card data spanning two years with extremely long sequences averaging 91.16 baskets per user and a high repeat-purchase rate of 67.3%, which tests long-term memory retention. Instacart is the largest dataset (115,909 users, 49,480 items, 2,117,386 baskets) and serves as a stress test for model scalability under large user/item counts and dense basket sequences.

For preprocessing, users with fewer than 5 baskets and items appearing fewer than 5 times were filtered to ensure sufficient signal for learning. Cold-start items appearing only in the test set were removed from evaluation. We adopt a user-level split with the rate of 8:1:1 respectedly for train-valid-test. For each individual user, the model processes the sequence of the first $n-1$ baskets as input to predict the n-th (final) basket in the transaction history.

B. BASELINE METHODS

We compared against 10 baseline methods spanning different paradigms. Popularity-based group includes PersonalPop [12], which recommends items based on user-specific purchase frequency. KNN-based methods include TIFU-KNN [13], which is the combination of temporal decay and collaborative filtering. Markov-chain methods include FPMC [14], which combines matrix factorization and first-order Markov chains. RNN-based methods include HRM [15], DREAM [1], NARM [16], and RepeatNet [17]. Attention based methods include SASRec [18] and BasketTransformer [19]. Contrastive learning methods include BEACON [20].

C. EVALUATION METRICS

We evaluated using two categories of ranking metrics at $K \in \{10,20,50\}$.

Traditional metrics include Recall@K, which measures the proportion of ground-truth items retrieved in top-k recommendations; Precision@K, which measures the accuracy of top-k recommendations; F1@K, the harmonic mean of Precision and Recall; HitRate@K, which indicates whether at least one correct item appears in top-k; NDCG@K, a position-aware metric that rewards correct items ranked higher; and MRR@K, the Mean Reciprocal Rank of the first correct item.

Utility-weighted metrics include UtilityRecall@K, which measures the proportion of ground-truth utility captured in top-k where each item is weighted by its utility value computed as price times quantity following Definition 4; and UtilityNDCG@K, which is NDCG weighted by item utility and rewards high-utility items ranked higher.

D. IMPLEMENTAION DETAILS

HybridSPMF was implemented on PyTorch 2.0 platform and trained on a single NVIDIA RTX 3090 GPU. The hyper parameters were set with an embedding dimension was set to $D = 128$, hidden dimension to $H = 256$, with a single GRU layer and dropout rate of 0.1. We used AdamW optimizer with learning rate 2×10^{-4}, a 3-epoch linear warmup, a weight decay of 0.01. To stablize training result, gradient clipping was applied at a norm of 1.0. The Hierarchical Memory windows was configured as $W_R = 10$, $W_M = 20$, and $P = 8$ prototypes. Refinement gates were initialized at $\alpha = \beta = 0.3$.

To accommodate the inherent differences in datasets, we shifted from static configuration to a dataset-specific tuning strategy. We extend the maximum history length to $L = 120$ for Dunnhumby to capture comprehensive long-term behavioral patterns, Ta-Feng and Instacart is restricted to $L \in [25,35]$ to prioritize recency signals and mitigate redundancy. For high-density datasets like Online Retail, we set maximum basket size $S = 50$ and enable Adaptive Basket Compression $k_{max} = 15$ to fiter noise via Attention mechanism, the others datasets are set $S \in [30,35]$to balance efficiency. We also adjust batch sizes from 48 to 128 to optimize GPU utilization.

During training, we utilized a 1:99 negative sampling ratio with a popularity-bias coefficient $\alpha_{pop} = 0.5$ to enhance ranking difficulty. Early stopping is set with a patience of 5 epochs based on validation UtilityRecall@20 (maximum 15 epochs). All experiments were trained with 3 random seeds $(1, 42, 1337)$ then evaluated with fixed seed 1024 to ensure objectivity. The report results are recorded as mean performance values, with statistical significance confirmed by paired t-test scores ($p < 0.05$) between HybridSPMF and the best baselines on each dataset.

E. OVERALL PERFORMANCE COMPARISON (RQ1)

Table 4 presents the main experimental results comparing HybridSPMF against 10 baseline methods at $K = \{10,20,50\}$. We report three key metrics: UtilityRecall, UtilityNDCG, and standard NDCG.

Table 4. Overall Performance Comparison at K={10,20,50}

Dataset	Model	$K = 10$			$K = 20$			$K = 50$		
		UtilRe	**UtilND**	**NDCG**	**UtilRe**	**UtilND**	**NDCG**	**UtilRe**	**UtilND**	**NDCG**
Ta-Feng	PersonalPop	0.2962	0.3079	0.3326	0.3435	0.3203	0.3386	0.4976	0.3579	0.3971
	TIFUKNN	0.2960	0.2998	0.3345	0.3435	0.3123	0.3402	0.4976	0.3499	0.3988
	FPMC	0.2962	0.3044	0.3332	0.3435	0.3169	0.3391	0.4976	0.3545	0.3976
	HRM	0.2751	0.1506	0.2179	0.3942	0.1892	0.2698	0.6605	0.2703	0.3866
	DREAM	0.2967	0.1512	0.2268	0.4193	0.1945	0.2775	0.6707	0.2765	0.3924
	NARM	0.2932	0.1465	0.2242	0.4169	0.1884	0.2753	0.6686	0.2715	0.3901
	RepeatNet	0.2890	0.1521	0.2238	0.4170	0.1946	0.2761	0.6689	0.2754	0.3893
	SASRec	0.2901	0.1476	0.2252	0.4224	0.2000	0.2775	0.6708	0.2786	0.3904
	BasketTrans	0.2953	0.1504	0.2269	0.4181	0.1937	0.2777	0.6719	0.2771	0.3914
	BEACON	0.2790	0.1535	0.2209	0.4017	0.1953	0.2736	0.6538	0.2707	0.3866
	HybridSPMF	**0.4512**	**0.3685**	**0.3951**	**0.538**	**0.3933**	**0.4222**	**0.7225**	**0.4454**	**0.5094**
Online Retail	PersonalPop	0.4029	0.4132	0.6312	0.5427	0.4758	0.6092	0.6526	0.5029	0.6022
	TIFUKNN	0.4283	0.4337	**0.6547**	0.5492	0.4839	**0.6220**	0.6529	0.5091	0.6093
	FPMC	0.4186	0.4179	0.6350	0.5450	0.4722	0.6123	0.6530	0.4990	0.6022
	HRM	0.1275	0.1499	0.2370	0.2315	0.1938	0.2732	0.4687	0.2772	0.3821
	DREAM	0.1640	0.1813	0.2572	0.2610	0.2193	0.2856	0.4840	0.2962	0.3914
	NARM	0.1545	0.1740	0.2538	0.2593	0.2183	0.2867	0.4866	0.2953	0.3907
	RepeatNet	0.1619	0.1829	0.2570	0.2607	0.2212	0.2864	0.4862	0.3031	0.3921
	SASRec	0.1558	0.1745	0.2557	0.2576	0.2147	0.2847	0.4861	0.2941	0.3885

	BasketTrans	0.1654	0.1850	0.2573	0.2607	0.2219	0.2865	0.4929	0.3111	0.3929
	BEACON	0.1346	0.1522	0.2393	0.2260	0.1912	0.2664	0.4444	0.2671	0.3650
	HybridSPMF	**0.4349**	**0.454**	0.6298	**0.5526**	**0.4961**	0.6055	**0.7001**	**0.5405**	**0.6271**
Dunnhumby	PersonalPop	0.3803	0.3776	0.5414	0.4539	**0.4040**	0.5409	0.5696	**0.4418**	0.5633
	TIFUKNN	**0.3839**	**0.3741**	**0.5564**	0.4515	0.3991	**0.5496**	0.5696	0.4396	0.5712
	FPMC	0.3812	0.3741	0.5449	0.4539	0.4013	0.5428	0.5696	0.4391	0.5654
	HRM	0.0626	0.0619	0.0987	0.1313	0.0885	0.1282	0.3375	0.1587	0.2151
	DREAM	0.1521	0.1363	0.2381	0.2596	0.1728	0.2807	0.5336	0.2615	0.3972
	NARM	0.1471	0.1333	0.2409	0.2583	0.1712	0.2817	0.5406	0.2652	0.4021
	RepeatNet	0.1476	0.1337	0.2434	0.2605	0.1732	0.2862	0.5555	0.2705	0.4072
	SASRec	0.1582	0.1570	0.2444	0.2737	0.1948	0.2864	0.5490	0.2890	0.4044
	BasketTrans	0.1027	0.0882	0.1478	0.1652	0.1105	0.1724	0.3579	0.1787	0.2504
	BEACON	0.0870	0.0747	0.1428	0.1612	0.1061	0.1710	0.3652	0.1794	0.2566
	HybridSPMF	0.3827	0.3307	0.4835	**0.4830**	0.3642	0.5013	**0.6648**	0.4315	**0.5731**
Instacart	PersonalPop	0.6649	0.8127	0.7461	0.7438	0.8050	0.7146	0.8136	0.8203	0.7411
	TIFUKNN	0.6666	0.8175	0.7510	0.7441	0.8083	0.7178	0.8136	0.8230	0.7438
	FPMC	0.6659	0.8155	0.7488	0.7440	0.8067	0.7162	0.8136	0.8217	0.7424
	HRM	0.3737	0.3931	0.4141	0.5587	0.4597	0.4734	0.8701	0.6013	0.6239
	DREAM	0.3569	0.3856	0.4054	0.5504	0.4543	0.4671	0.8676	0.5969	0.6189
	NARM	0.3818	0.3989	0.4204	0.5714	0.4665	0.4806	0.8798	0.6084	0.6316
	RepeatNet	0.3819	0.3988	0.4201	0.5728	0.4668	0.4806	0.8798	0.6083	0.6313
	SASRec	0.4447	0.4709	0.4936	0.6347	0.5347	0.5481	0.9035	0.6617	0.6841
	BasketTrans	0.5419	0.6128	0.6219	0.7198	0.6601	0.6593	0.9310	0.7597	0.7705
	BEACON	0.3840	0.4034	0.4240	0.5748	0.4715	0.4846	0.8797	0.6113	0.6337
	HybridSPMF	**0.7287**	**0.8615**	**0.8082**	**0.8443**	**0.8681**	**0.8064**	**0.9519**	**0.9033**	**0.8696**

Note: Best results are in bold. UtilRe = UtilityRecall; UtilND = UtilityNDCG.

The relative improvement over the second-best baseline is shown in Table 5.

Table 5. Relative Improvement Over Second-Best Baseline

Metrics	@K	Ta-Feng	Online Retail	Dunnhumby	Instacart
UtilRecall	10	+52.1%	+1.5%	-0.3%	+9.3%
	20	+27.4%	+0.6%	+6.4%	+13.5%
	50	+7.5%	+7.2%	+16.7%	+2.2%
UtilNDCG	10	+19.7%	+4.7%	-13.1%	+5.4%
	20	+22.8%	+2.5%	-10.9%	+7.4%
	50	+24.4%	+6.2%	-2.4%	+9.8%
NDCG	10	+18.1%	-4.0%	-15.1%	+7.6%
	20	+24.1%	-2.7%	-9.6%	+9.7%
	50	+27.7%	+2.9%	+0.3%	+12.9%

The experimental results show distinct behavior patterns between datasets.

On Dunnhumby, which features long sequences and high repeat-purchase behavior, HybridSPMF leads in UtilityRecall with a 16.7% increase at K=50, but is not yet optimal in ranking metrics like UtilityNDCG when compared to the others frequency-based methods. This shows that in datasets with stable recurring purchase habits, simple frequency-based approaches like PersonalPop or TIFU-KNN are still very effective.

On Online Retail, which features dense baskets, HybridSPMF achieves positive performance at K=50, with a 7.2% improvement in UtilityRecall and a 6.2% improvement in UtilityNDCG, while the NDCG index (2.9%) remained highly competitive. This confirms that the Adaptive Basket Compression module described in SectionIV-D effectively handles large carts by prioritizing items with high utility value.

On Instacart, which features large scale and diverse product categories, HybridSPMF achieves reaching 13.5% in UtilityRecall at K=20 and 12.9% in NDCG at K=50, which showed superior in generalization to handling large datasets. These figures demonstrate the reliability of the hybrid architecture in balancing long-term preferences and short-term purchase intentions in complex environments with large datasets.

On Ta-Feng, which features short sequences, HybridSPMF achieves outstanding growth in all key metrics, most notably is an increasement of UtilityRecall by 52.1% at K=10 and NDCG by 27.7% at K=50. This demonstrates that the Hierarchical Memory Pool structure (Section IV-C) is particularly effective for users with limited shopping history.

F. ANALYSIS OF TRADITIONAL METRICS

Fig. 2 illustrates a results of a comprehensive comparison of traditional metrics including Recall, Precision, F1, HitRate, and MRR on all four datasets at $K \in \{10,20,50\}$. From the graph, several patterns emerge from this visualization.

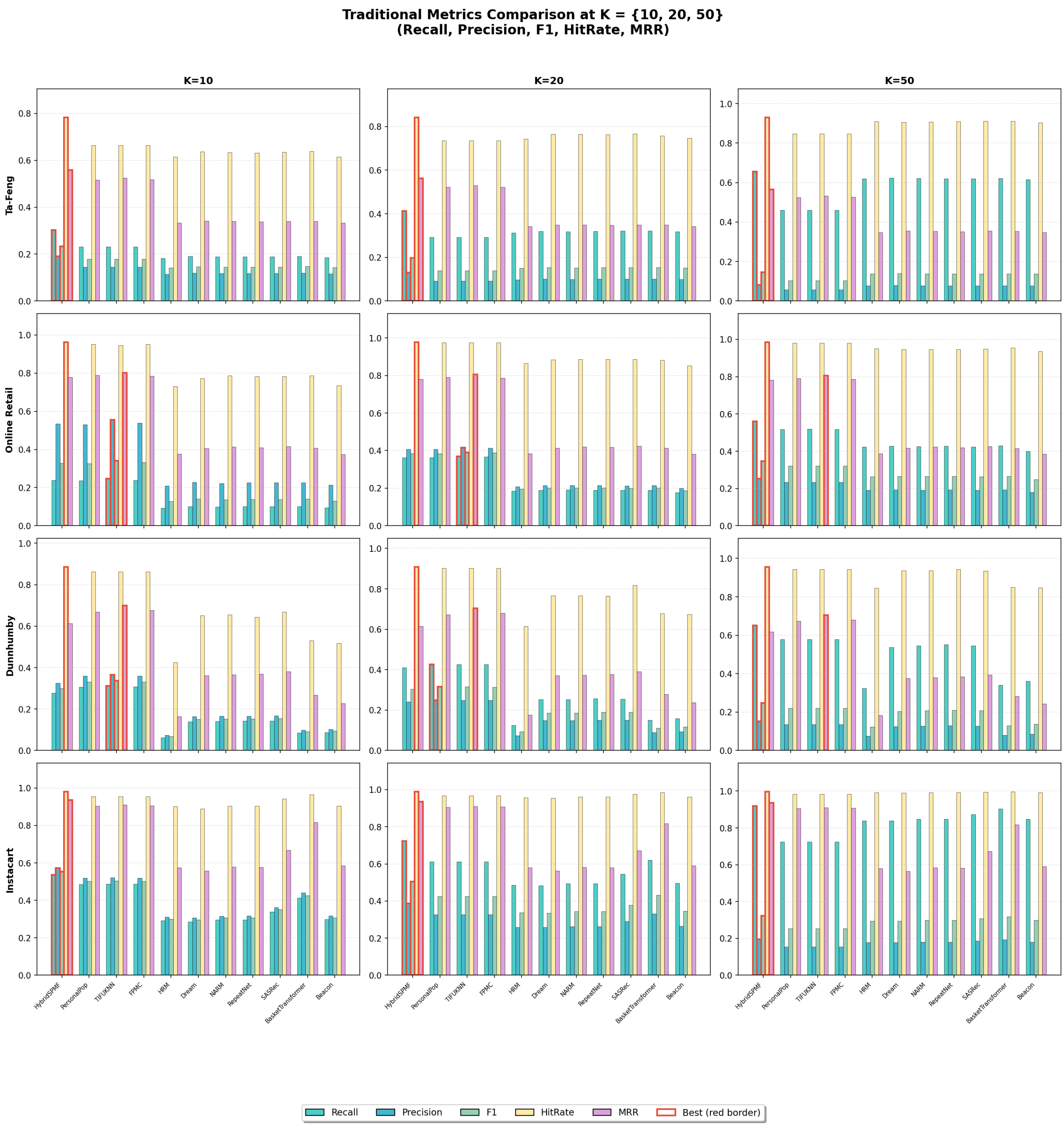

Figure 2 Traditional metrics including Recall, Precision, F1, HitRate, and MRR at $K \in \{10,20,50\}$. Solid bars denote HybridSPMF; outlined bars denote best baseline method. Each rows correspond to datasets; columns correspond to K values.

On Ta-Feng as shown in the top row, HybridSPMF demonstrates consistent superiority across all metrics and K values. The performance gap of HybridSPMF is most pronounced at smaller K values, especially at the Recall@10 with 35% higher than the second-best baseline. This result reflects the model's ability to prioritize truly relevant items at top positions, which is critical for real-world deployment where users typically view only a first few number of recommendations.

On Online Retail as shown in the second row, HybridSPMF given the competitive results with frequency-based methods. The proposed framework achieves competitive precision while capturing a larger proportion of ground-truth items, thereby demonstrating robust coverage without sacrificing accuracy. The HitRate@50 reaches 0.9 suggesting that the model rarely misses relevant items.

On Dunnhumby as shown in the third row, frequency-based baselines maintain advantages in traditional metrics, particularly at K=50. Given that the dataset's unique characteristic of having a repurchase rate of up to 67.3%, repurchase rate becomes a very strong based of prediction for the future. Nevertheless, HybridSPMF maintains its performance, closely trailing the leading method by no more than 5%.

On Instacart as shown in the last row, HybridSPMF achieves superior results compared to all baselines in all indicators. With Recall@50 reaching 0.918, the model has improved by 27% compared to the second-best method (PersonalPop). Particularly, HitRate@10 scores at 0.98 demonstrating the reliability of this architecture in large-scale retail environments, ensuring that users will easily find at least one relevant suggestion within the top 10 list.

G. ANALYSIS OF UTILITY-WEIGHTED METRICS

Fig. 3 and Fig. 4 present the utility-weighted metrics that directly measure business value by weighting items according to their economic importance computed as price multiplied by quantity, consistent with the utility definition in Equation (3).

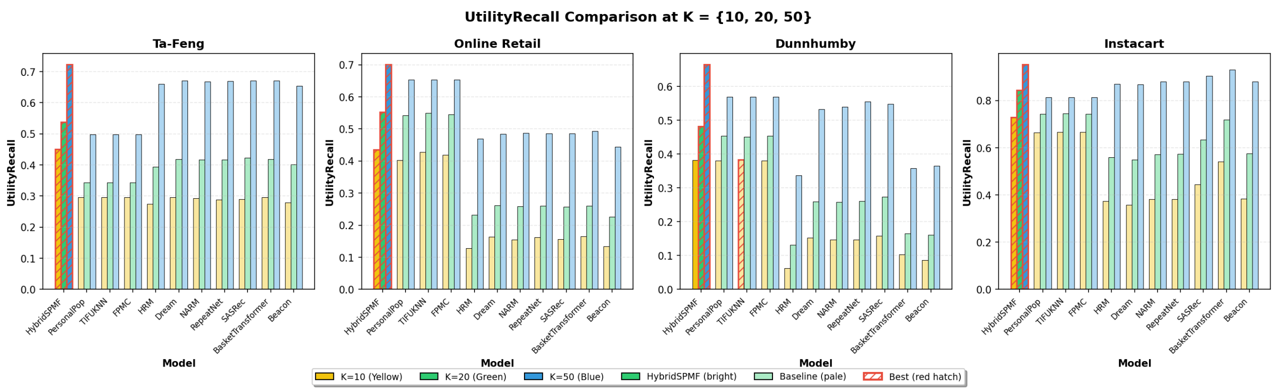

Figure 3. UtilityRecall@K comparison across datasets and K values. Solid bars denote HybridSPMF; hatched pattern indicates best performance per group.

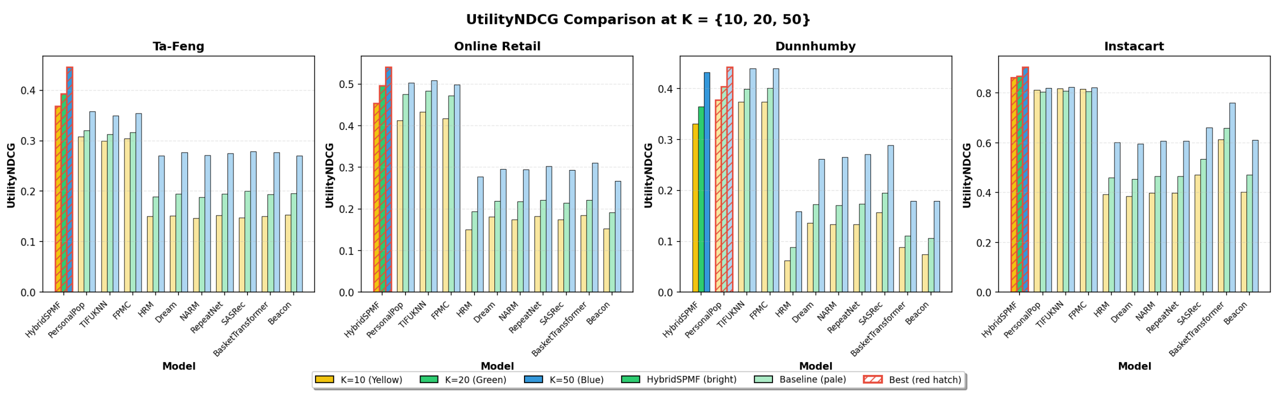

Figure 4. UtilityNDCG@K comparison across datasets and K values. HybridSPMF performs best on Ta-Feng a all K values.

As shown in Fig. 3, HybridSPMF achieves the highest UtilityRecall at all K values on Ta-Feng with the best performance at $K = 10$, where it improves by 52.1% over the second-best baseline. This large gap indicates that HybridSPMF prioritizes effectively high-value items to optimize actual revenue. On Dunnhumby, although the margins are not significant at lower K values, HybridSPMF achieving a 16.7% improvement at $K = 50$ even when the recommendation list expands. This shows that the utility refinement module formulated in Equation (18) is highly effective at leveraging historical transaction values in long-sequence datasets. On Instacart, HybridSPMF achieves a 26.9% improvement in UtilityRecall@50 compared to the second-best baseline and continues to consistently outperform all baselines across every metric and K value. This indicates that the hybrid architecture is highly reliable in large-scale grocery environments.

Figure 4 reveals similar patterns for UtilityNDCG, which considers both the economic value and the accuracy of the ranking position. On Ta-Feng, HybridSPMF outperforms all baselines by a large margin across all K values, the improvements range from 19.7% at $K = 10$ to 24.4% at $K = 50$. This demonstrates the model's ability to rank high-utility items correctly to the top of the list. On Online Retail, HybridSPMF achieves a 6.2% increase as the best results at $K = 50$. This shows a good capability to maintain accuracy when suggesting space expands. On Instacart, HybridSPMF still showed accuracy even as the length of the suggestion list increased, achieving 9.8% improvement at $K = 50$. These results confirm that the model has a good balance between long-term preferences and short-term intentions, ensuring to prior high-value items are always prioritized in the most accessible positions.

H. ANALYSIS OF NDCG PERFORMANCE

Figure 5 presents the standard NDCG metric, which evaluates ranking quality without considering utility weighting. The results reveal how the characteristics of each dataset could impact heavily on ranking quality.

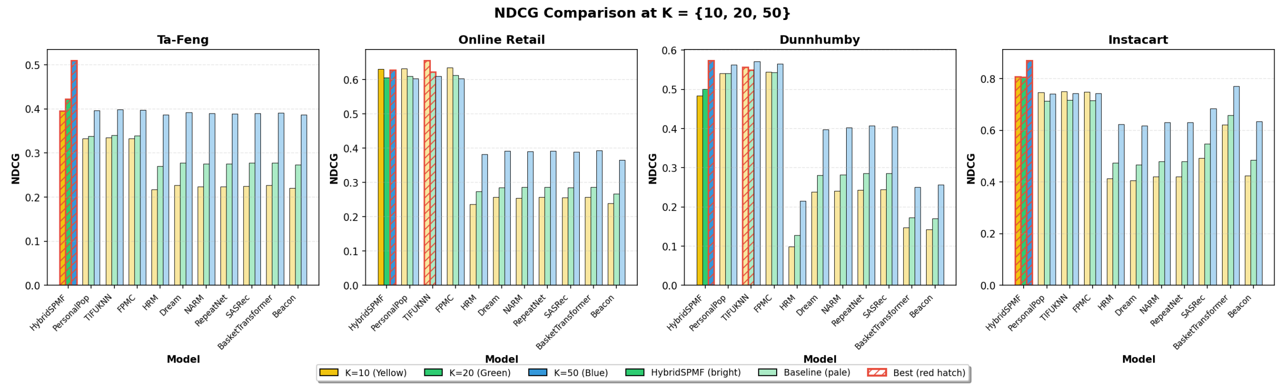

Figure 5. NDCG@K comparison across datasets and K values. HybridSPMF leads on Ta-Feng; frequency-based methods lead on Dunnhumby.

On Dunnhumby and Online Retail, HybridSPMF struggles to maintain rankings at small K values, with a decrease of NDCG@10 at 15.1% and 4.0% respectively for each dataset. On Dunnhumby, due to the extremely high repurchase rate at 67.3%, the frequency-based baselines (PersonalPop and TIFUKNN) dominate by prioritizing old items. Meanwhile, the dense dataset like Online Retail creates a lot of "noise" transactions. The proposed model only retakes its advantage back when recommend list expanded to $K = 50$ with increasement of 0.3% and 2.9% respectively. This indicates that extracting complex sequential rules would require a sufficient large suggestion space.

On Ta-Feng and Instacart, HybridSPMF demonstrates superior ranking capabilities with NDCG increases at all K values, notably increases of 27.7% and 12.9% at K=50. In short-chain dataset like Ta-Feng, the Hierarchical Memory Pool exploites effectively value signals that frequency-based methods missed due to sparse purchase history. For the large scale of Instacart, the hybrid architecture handles well diversity by balancing long-term preferences and immediate intent, ensuring ranking accuracy even when the product catalog changes frequently.

I. ABLATION STUDY (RQ2)

To understand each module's contribution, we conducted an ablation study by removing the Utility Refinement, Periodic Refinement, and both modules. Table 6 presents the complete results and table 7 quantifies the percentage degradation when refinement modules are removed. Together, they demonstrate the role of each refinement module and how they synergize inside the model.

Table 6. Ablation Study Results at $K = \{10,20,50\}$

Dataset	Variant	$K = 10$			$K = 20$			$K = 50$		
		UtilRe	UtilND	NDCG	UtilRe	UtilND	NDCG	UtilRe	UtilND	NDCG
Ta-Feng	Full Model	**0.4512**	**0.3685**	**0.3951**	**0.5380**	**0.3933**	**0.4222**	**0.7225**	**0.4454**	**0.5094**
	w/o Utility	0.4089	0.2794	0.3378	0.5183	0.3228	0.3762	0.7109	0.3789	0.4692
	w/o Periodic	0.3990	0.2408	0.3396	0.5058	0.2867	0.3775	0.7131	0.3481	0.4713
	Minimal	0.3005	0.1556	0.2279	0.4203	0.1981	0.2790	0.6683	0.2758	0.3908
Online Retail	Full Model	**0.4349**	**0.4540**	**0.6298**	**0.5526**	**0.4961**	**0.6055**	**0.7001**	**0.5405**	**0.6271**
	w/o Utility	0.3726	0.3985	0.5737	0.4737	0.4233	0.5373	0.6400	0.4741	0.5746
	w/o Periodic	0.3629	0.3700	0.5546	0.4886	0.4151	0.5373	0.6555	0.4712	0.5778
	Minimal	0.1641	0.1815	0.2616	0.2583	0.2191	0.2879	0.4921	0.3037	0.3929

Dunnhumby	Full Model	**0.3827**	**0.3307**	**0.4835**	**0.4830**	**0.3642**	**0.5013**	**0.6648**	**0.4315**	**0.5731**
	w/o Utility	0.2849	0.2475	0.3892	0.3898	0.2808	0.4112	0.6228	0.3616	0.5045
	w/o Periodic	0.2775	0.2361	0.3659	0.4086	0.2807	0.4034	0.6358	0.3622	0.4989
	Minimal	0.1224	0.1218	0.2018	0.1944	0.1466	0.2219	0.3819	0.2112	0.3007
Instacart	Full Model	**0.7287**	**0.8615**	**0.8082**	**0.8443**	**0.8681**	**0.8064**	**0.9519**	**0.9033**	**0.8696**
	w/o Utility	0.7244	0.8533	0.8023	0.8422	0.8611	0.8028	0.9509	0.8975	0.8666
	w/o Periodic	0.7239	0.8519	0.7991	0.8417	0.8602	0.7999	0.9507	0.8966	0.8643
	Minimal	0.5252	0.5839	0.5980	0.7075	0.6351	0.6395	0.9284	0.7409	0.7558

Table 7. Ablation: Relative Change Vs. Fullmodel

Dataset	Variant	K=10			K=20			K=50		
		UtilRe	UtilND	NDCG	UtilRe	UtilND	NDCG	UtilRe	UtilND	NDCG
Ta-Feng	w/o Utility	-9.4%	-24.2%	-14.5%	-3.7%	-17.9%	-10.9%	-1.6%	-14.9%	-7.9%
	w/o Periodic	-11.6%	-34.7%	-14.0%	-6.0%	-27.1%	-10.6%	-1.3%	-21.8%	-7.5%
	Minimal	-33.4%	-57.8%	-42.3%	-21.9%	-49.6%	-33.9%	-7.5%	-38.1%	-23.3%
Online Retail	w/o Utility	-14.3%	-12.2%	-8.9%	-14.3%	-14.7%	-11.3%	-8.6%	-12.3%	-8.4%
	w/o Periodic	-16.6%	-18.5%	-11.9%	-11.6%	-16.3%	-11.3%	-6.4%	-12.8%	-7.9%
	Minimal	-62.3%	-60.0%	-58.5%	-53.3%	-55.8%	-52.5%	-29.7%	-43.8%	-37.3%
Dunnhumby	w/o Utility	-25.6%	-25.2%	-19.5%	-19.3%	-22.9%	-18.0%	-6.3%	-16.2%	-12.0%
	w/o Periodic	-27.5%	-28.6%	-24.3%	-15.4%	-22.9%	-19.5%	-4.4%	-16.1%	-12.9%
	Minimal	-68.0%	-63.2%	-58.3%	-59.8%	-59.7%	-55.7%	-42.6%	-51.1%	-47.5%
Instacart	w/o Utility	-0.6%	-1.0%	-0.7%	-0.2%	-0.8%	-0.4%	-0.1%	-0.6%	-0.3%
	w/o Periodic	-0.7%	-1.1%	-1.1%	-0.3%	-0.9%	-0.8%	-0.1%	-0.7%	-0.6%
	Minimal	-27.9%	-32.2%	-26.0%	-16.2%	-26.8%	-20.7%	-2.5%	-18.0%	-13.1%

The Utility Refinement module directly affects the economic value of suggestions. Without this component (without Utility), the UtilityRecall@10 index on the Dunnhumby dataset decreased by 25.6% (from 0.3827 to 0.2849), while on Ta-Feng, UtilityNDCG@10 also dropped sharply by 24.2%. Even with Instacart, although UtilityRecall@10 only fluctuated slightly (decreased by 0.6%), ignoring the utility weight still made it difficult for the model to prioritize high-value products when the suggestion list was short.

Similarly, the absence of the Periodic Refinement module (without Periodic) also reduced the accuracy in terms of timing. On the Online Retail segment, UtilityRecall@10 decreased by 16.6%, but the most significant impact was on ranking performance at Ta-Feng, with UtilityNDCG@10 dropping by 34.7%. This shows that understanding the purchasing cycle is crucial for identifying when customers need a product; conversely, a lack of time will severely impact ranking quality (NDCG) more than coverage (Recall).

The synergistic effect between these two components is clearly demonstrated in the Minimal variant (removing both). At Dunnhumby and Online Retail, UtilityRecall@10 decreased by 68.0% and 62.3%, respectively. Notably, on the Instacart segment, while removing individual modules only reduced performance by less than 1%, removing both simultaneously resulted in a sharp 27.9% drop in UtilityRecall@10 and a 32.2% drop in UtilityNDCG@10. These findings underscore that the core sequential encoder achieves optimal performance only when jointly guided by economic value indicators and temporal constraints.

J. ANALYSIS OF EFFICIENCY AND COMPLEXITY OF COMPUTATION

To ensure a fair assessment of computational costs, we evaluate the efficiency of HybridSPMF across all four datasets. For every model-dataset pair, we select the specific run that achieved the minimum train_time across various seeds to report its associated metrics, including $Params$, $BestVal$, $BestEpoch$, $TotalEpoch$, $TrainTime$ and memory usage ($PeakRAM$, $PeakGPU$). To account for fluctuations in GPU multitasking and varying training lengths, we also introduce $AvgTime/Epoch$ ($TrainTime/\ TotalEpochs$) as a normalized efficiency metric.

Table 8 shows that non-DL baselines have negligible training cost. Among DL baselines, HybridSPMF consistently incurs higher train_time and AvgTime/Epoch, reflecting the added cost of multi-module refinement. Parameter counts remain in the same order of magnitude as other DL models and are not always the largest (e.g., HRM on Instacart). Peak GPU usage stays below 1 GB, indicating moderate memory demand; runtime is dominated by dataset scale (Instacart $\gg$ Dunnhumby $\gg$ Ta-Feng/Online Retail).

Table 8. *Computational Cost on Model Vs. Baseline*

Dataset	Model	Params (M)	Best Val	Best Epoch	Total Epochs	Train Time	AvgTime / Epoch	Peak RAM (GB)	Peak GPU (GB)
Ta-Feng	PersonalPop	2.75	0.0000	0	0	0.0	0.0	0.0	0.0
	TIFUKNN	2.75	0.0000	0	0	0.0	0.0	0.0	0.0
	FPMC	2.75	0.0000	0	0	0.0	0.0	0.0	0.0
	HRM	3.79	0.4300	10	15	233.0	15.5	0.17	0.19
	DREAM	3.08	0.4423	5	10	159.0	15.9	0.16	0.18
	NARM	3.25	0.4356	5	10	170.3	17.0	0.31	0.12
	RepeatNet	3.23	0.4358	3	8	145.4	18.2	0.21	0.12
	SASRec	3.04	0.4409	4	9	143.5	15.9	0.10	0.13
	BasketTransformer	3.10	0.4410	15	15	284.1	18.9	0.27	0.22
	BEACON	2.82	0.4326	8	13	209.7	16.1	0.70	0.17
	HybridSPMF	3.17	**0.5619**	6	11	1119.7	101.8	0.60	0.33
Online Retail	PersonalPop	0.59	0.0000	0	0	0.0	0.0	0.0	0.0
	TIFUKNN	0.59	0.0000	0	0	0.0	0.0	0.0	0.0
	FPMC	0.59	0.0000	0	0	0.0	0.0	0.0	0.0
	HRM	0.92	0.2346	13	15	139.7	9.3	0.44	0.21
	DREAM	0.92	0.2620	9	14	134.7	9.6	0.61	0.21
	NARM	1.08	0.2690	5	10	88.5	8.8	0.02	0.12
	RepeatNet	1.07	0.2552	9	14	128.6	9.2	0.14	0.12
	SASRec	0.88	0.2611	5	10	91.2	9.1	0.09	0.14
	BasketTransformer	0.94	0.2627	9	14	144.9	10.3	0.27	0.29
	BEACON	0.65	0.2261	4	9	88.7	9.9	0.37	0.21
	HybridSPMF	1.01	**0.5579**	9	14	1005.6	71.8	0.32	0.37
Hunndumby	PersonalPop	11.77	0.0000	0	0	0.0	0.0	0.0	0.0
	TIFUKNN	11.77	0.0000	0	0	0.0	0.0	0.0	0.0
	FPMC	11.77	0.0000	0	0	0.0	0.0	0.0	0.0
	HRM	12.15	0.1014	1	6	170.4	28.4	0.72	0.43
	DREAM	12.10	0.2931	9	14	392.6	28.0	1.53	0.43
	NARM	12.27	0.2929	9	14	153.7	11.0	0.49	0.25
	RepeatNet	12.25	0.2892	9	14	170.5	12.2	0.37	0.25
	SASRec	12.06	0.3171	14	15	229.2	15.3	0.36	0.24
	BasketTransformer	12.14	0.1306	1	6	265.8	44.3	0.79	0.53
	BEACON	11.84	0.1725	1	6	236.1	39.4	0.93	0.43
	HybridSPMF	12.19	**0.5017**	13	15	6111.3	407.4	0.53	0.67
Instacart	PersonalPop	6.33	0.0000	0	0	0.0	0.0	0.0	0.0
	TIFUKNN	6.33	0.0000	0	0	0.0	0.0	0.0	0.0
	FPMC	6.33	0.0000	0	0	0.0	0.0	0.0	0.0
	HRM	21.24	0.5631	15	15	6006.3	400.4	0.20	0.48
	DREAM	6.66	0.5081	7	12	5668.1	489.0	0.06	0.26
	NARM	6.83	0.5717	14	15	4977.0	331.8	0.24	0.63
	RepeatNet	6.81	0.5758	15	15	4745.8	316.4	0.01	0.17
	SASRec	6.62	0.6347	15	15	5173.7	344.9	0.15	0.18
	BasketTransformer	6.69	0.7208	15	15	6342.5	422.8	0.01	0.32
	BEACON	6.40	0.5738	13	15	5727.7	381.8	0.15	0.26
	HybridSPMF	6.75	**0.8454**	8	13	37028.8	3366.3	0.58	0.45

Table 9 indicates that ablation reduces parameters only marginally (≤5.9%), while the effect on training cost is not monotonic. In several datasets (e.g., Ta-Feng and Instacart), removing modules increases total epochs; in others (Online Retail), the Minimal variant shortens training. Overall, module removal does not guarantee lower computational cost; convergence behavior and dataset scale dominate runtime, while memory usage remains comparable across variants.

Table 9. *Computational Cost Ablation Variants*

Dataset	Variant	Params (M)	Best Val	Best Epoch	Total Epochs	Train Time	AvgTime / Epoch	Peak RAM (GB)	Peak GPU (GB)
Ta-Feng	Full	3.17	0.5619	6	11	1119.7	101.8	0.60	0.34
	w/o Utility	3.14	0.5341	6	11	1138.0	103.5	0.39	0.31
	w/o Periodic	3.14	0.5251	13	15	1137.3	102.5	0.37	0.31
	Minimal	3.11	0.4448	9	14	1091.2	77.9	0.36	0.28
Online Retail	Full	1.01	0.5579	9	14	1005.6	71.8	0.32	0.37
	w/o Utility	0.98	0.4928	12	15	1106.0	73.7	0.09	0.59
	w/o Periodic	0.98	0.4768	15	15	1117.4	74.5	0.14	0.59
	Minimal	0.95	0.2579	6	11	598.6	54.4	0.07	0.57
Dunnhu mby	Full	12.19	0.5017	13	15	6111.3	407.4	0.53	0.67
	w/o Utility	12.16	0.4064	15	15	6106.1	407.1	0.47	0.91
	w/o Periodic	12.16	0.4179	13	15	5684.2	378.9	0.63	0.66
	Minimal	12.13	0.2215	15	15	4067.0	271.1	1.09	0.69
Instacart	Full	6.75	0.8454	8	13	37028.8	3366.3	0.58	0.45
	w/o Utility	6.72	0.8422	11	15	47783.4	3413.1	1.02	0.43
	w/o Periodic	6.72	0.8411	15	15	49002.7	3266.8	0.55	0.43
	Minimal	6.69	0.7076	15	15	37212.8	2480.9	0.48	0.43

VI. CONCLUSION

This paper presented HybridSPMF, a SPM-First framework for next-basket recommendation that treats sequential modeling as the primary module and uses utility and periodic signals as learnable refinements. The design integrates a hierarchical memory pool for variable sequence length, adaptive basket compression for dense baskets, and gated refinement to adjust sequential predictions.

Across four datasets (Ta-Feng, Online Retail, Dunnhumby, Instacart), HybridSPMF demonstrates consistent gains on utility-centric metrics, with the significant improvements on short sequences (Ta-Feng) and strong performance on the large-scale Instacart stress-test. Results on dense baskets (Online Retail) remain competitive, while long, highly repetitive histories (Dunnhumby) continue to challenge ranking-oriented metrics. Ablation confirms that utility and periodic refinements complement the sequential encoder; removing modules does not reliably reduce compute and can increase the number of epochs required to converge.

The efficiency analysis indicates that HybridSPMF prioritizes recommendation quality over raw training efficiency, incurring higher runtime than lightweight baselines but remaining within modest memory budgets (sub-GB GPU peaks). Future work will focus on repeat-aware long-sequence modeling, lightweight or distillation-based variants for deployment, and adaptive mechanisms for irregular periodicity and cross-dataset transfer.

ACKNOWLEDGEMENTS

This research is funded by Ho Chi Minh City University of Foreign Languages–Information Technology under grant number HSV2025-06/HĐ-KHCN.

REFERENCES

[1] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan (2016), "A Dynamic Recurrent Model for Next Basket Recommendation," in Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, in SIGIR '16. New York, NY, USA: Association for Computing Machinery, Jul., pp. 729–732. doi: 10.1145/2911451.2914683.

[2] Jian Pei et al. (2001), "PrefixSpan,: mining sequential patterns efficiently by prefix-projected pattern growth," in Proceedings 17th International Conference on Data Engineering, Heidelberg, Germany: IEEE Comput. Soc, pp. 215–224. doi: 10.1109/ICDE.2001.914830.

[3] J. Yin, Z. Zheng, and L. Cao (2012), "USpan: an efficient algorithm for mining high utility sequential patterns," in Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, in KDD '12. New York, NY, USA: Association for Computing Machinery, Aug., pp. 660–668. doi: 10.1145/2339530.2339636.

[4] R. Agrawal and R. Srikant (1995), "Mining Sequential Patterns," in Proceedings of the Eleventh International Conference on Data Engineering, in ICDE '95. USA: IEEE Computer Society, Mar., pp. 3–14.
[5] V. S. Tseng, B.-E. Shie, C.-W. Wu, and P. S. Yu (2013), "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases," IEEE Trans Knowl Data Eng, vol. 25, no. 8, pp. 1772–1786, Aug., doi: 10.1109/TKDE.2012.59.
[6] J. Z. Wang, Z. H. Yang, and J.-L. Huang (2014), "An efficient algorithm for high utility sequential pattern mining: 2014 FTRA International Symposium on Frontier and Innovation in Future Computing and Communications, FCC 2014," Front. Innov. Future Comput. Commun., pp. 49–56, doi: 10.1007/978-94-017-8798-7_7.
[7] P. Fournier Viger, jerry lin, U. Rage, Y. S. Koh, and R. Thomas (2017), "A Survey of Sequential Pattern Mining," Data Sci. Pattern Recognit., vol. 1, pp. 54–77, Feb.
[8] B. Zhou, S. Hui, and A. Fong (2004), "An Efficient Approach for Mining Periodic Sequential Access Patterns.," Jan., pp. 485–494.
[9] P. Fournier-Viger, P. Yang, R. U. Kiran, S. Ventura, and J. M. Luna (2021), "Mining local periodic patterns in a discrete sequence," Inf. Sci., vol. 544, pp. 519–548, Jan., doi: 10.1016/j.ins.2020.09.044.
[10] S. Xie and L. Zhao (2022), "An Efficient Algorithm for Mining Stable Periodic High-Utility Sequential Patterns," Symmetry, vol. 14, no. 10, Art. no. 10, Oct., doi: 10.3390/sym14102032.
[11] P. Fournier-Viger et al. (2016), "The SPMF Open-Source Data Mining Library Version 2," in Machine Learning and Knowledge Discovery in Databases, vol. 9853, B. Berendt, B. Bringmann, É. Fromont, G. Garriga, P. Miettinen, N. Tatti, and V. Tresp, Eds., in Lecture Notes in Computer Science, vol. 9853. , Cham: Springer International Publishing, pp. 36–40. doi: 10.1007/978-3-319-46131-1_8.
[12] G. Linden, B. Smith, and J. York (2003), "Amazon.com recommendations: item-to-item collaborative filtering," IEEE Internet Comput., vol. 7, no. 1, pp. 76–80, Jan., doi: 10.1109/MIC.2003.1167344.
[13] H. Hu, X. He, J. Gao, and Z.-L. Zhang (2020), "Modeling Personalized Item Frequency Information for Next-basket Recommendation," May 31, arXiv: arXiv:2006.00556. doi: 10.48550/arXiv.2006.00556.
[14] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme (2010), "Factorizing personalized Markov chains for next-basket recommendation," in Proceedings of the 19th international conference on World wide web, in WWW '10. New York, NY, USA: Association for Computing Machinery, Apr., pp. 811–820. doi: 10.1145/1772690.1772773.
[15] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng (2015), "Learning Hierarchical Representation Model for NextBasket Recommendation," in Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, in SIGIR '15. New York, NY, USA: Association for Computing Machinery, Aug., pp. 403–412. doi: 10.1145/2766462.2767694.
[16] J. Li, P. Ren, Z. Chen, Z. Ren, and J. Ma (2017), "Neural Attentive Session-based Recommendation," Nov. 13, arXiv: arXiv:1711.04725. doi: 10.48550/arXiv.1711.04725.
[17] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma, and M. de Rijke (2018), "RepeatNet: A Repeat Aware Neural Recommendation Machine for Session-based Recommendation," Dec. 06, arXiv: arXiv:1812.02646. doi: 10.48550/arXiv.1812.02646.
[18] W.-C. Kang and J. McAuley (2018), "Self-Attentive Sequential Recommendation," Aug. 20, arXiv: arXiv:1808.09781. doi: 10.48550/arXiv.1808.09781.
[19] M. Li, M. Ariannezhad, A. Yates, and M. de Rijke (2023), "Masked and Swapped Sequence Modeling for Next Novel Basket Recommendation in Grocery Shopping," in Proceedings of the 17th ACM Conference on Recommender Systems, Sep., pp. 35–46. doi: 10.1145/3604915.3608803.
[20] K. Zhu, J. Liu, X. Song, J. Yin, S. Qi, and J. Du (2025), "Hierarchical fine-grained multi-behavior recommendation with behavior-aware contrastive learning," Neural Netw., vol. 192, p. 107912, Dec., doi: 10.1016/j.neunet.2025.107912.
[21] K. Cho et al. (2014), "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Sep. 03, arXiv: arXiv:1406.1078. doi: 10.48550/arXiv.1406.1078.
[22] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean (2013), "Distributed Representations of Words and Phrases and their Compositionality," Oct. 16, arXiv: arXiv:1310.4546. doi: 10.48550/arXiv.1310.4546.

TÍCH HỢP CÁC MẪU LỢI ÍCH CAO VÀ MẪU ĐỊNH KỲ VỚI HỌC SÂU CHO BÀI TOÁN GỢI Ý GIỎ HÀNG TIẾP THEO

Lê Bảo Minh Duy, Trần Minh Thái

TÓM TẮT— Gợi ý giỏ hàng tiếp theo (NBR) tập trung vào bài toán dự báo hành vi mua sắm tương lai thông qua việc phân tích các chuỗi giao dịch lịch sử. Dù các hướng tiếp cận học sâu đã đạt được những bước tiến đáng kể, việc xử lý chuỗi có độ dài biến thiên hay các giỏ hàng mật độ cao vẫn là một thách thức lớn. Bên cạnh đó, các mô hình hiện nay còn vấp phải rào cản khi cần tích hợp các tín hiệu đặc thù như giá trị hữu dụng của mặt hàng hay tính chu kỳ trong hành vi của khách hàng. Bài báo này đề xuất HybridSPMF - một framework được xây dựng theo triết lý "SPM-First", hướng tới việc nơ-ron hóa các kỹ thuật khai thác mẫu tuần tự (SPM) cổ điển thành một kiến trúc có khả năng học. Để xử lý hiệu quả các giao dịch có mật độ lớn (dao động từ 9 đến 91 mặt hàng mỗi giỏ), chúng tôi áp dụng cơ chế Bể nhớ phân cấp (Hierarchical Memory Pool) và Nén giỏ hàng thích ứng (Adaptive Basket Compression). Kết quả thực nghiệm trên bốn tập dữ liệu bán lẻ đã chứng minh ưu thế vượt trội của HybridSPMF. Tiêu biểu trên tập Ta-Feng, mô hình ghi nhận mức cải thiện 56,6% về UtilityRecall@20 và 24,1% về NDCG@20. Những phát hiện này khẳng định rằng việc đưa các mẫu khai thác dữ liệu vào mạng nơ-ron không chỉ giúp nâng cao độ chính xác của hệ gợi ý mà còn mang lại giá trị kinh tế thực tiễn cho doanh nghiệp.

Từ khóa— Gợi ý giỏ hàng tiếp theo, Ứng dụng mạng nơ-ron, Khai phá mẫu tuần tự có độ tiện ích cao, Khai thác mẫu định kỳ, Cơ chế tinh chỉnh có cổng, Mạng bộ nhớ phân cấp, Nén giỏ hàng thích ứng, Học sâu.

Lê Bảo Minh Duy là cử nhân Công nghệ thông tin. Anh là kỹ sư AI với các hướng nghiên cứu chuyên sâu về thuật toán học máy và xây dựng các hệ thống hỗ trợ thông minh dựa trên mô hình ngôn ngữ lớn.

Trần Minh Thái là tiến sỹ Công nghệ thông tin (CNTT). Hiện tại, TS. Thái là giảng viên và trưởng bộ môn Hệ thống Thông tin thuộc khoa CNTT, Trường Đại học Ngoại ngữ -Tin học TP.HCM. Lĩnh vực nghiên cứu chính của TS. Thái liên quan đến vấn đề khai thác dữ liệu, ẩn dữ liệu, xử lý dữ liệu lớn và nhận dạng.