International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 30, No. 12, December, 2025, pp. 3093-3102
Journal homepage: http://iieta.org/journals/isi

Enhancing Sequential Recommendation with Multi-Intent Detection and Preference Scoring from Implicit Transactions

Andy Maulana Yusuf, Adiwijaya*, Agung Toto Wibowo, Z. K. A. Baizal

School of Computing, Telkom University, Bandung 40257, Indonesia

Corresponding Author Email: adiwijaya@telkomuniversity.ac.id

https://doi.org/10.18280/isi.301202

Received: 12 August 2025
Revised: 14 November 2025
Accepted: 29 November 2025
Available online: 31 December 2025

Keywords:
sequential recommendation, multi-intent detection, In-Context Learning, preference scoring, Position-Based Grouping, semantic similarity, Instacart dataset, Large Language Model

ABSTRACT

In general, recommendation systems assume that each transaction made by a user reflects a single purchase intent (mono-intent). However, in reality, a transaction may contain several underlying intentions, such as routine shopping, consumption for specific purposes, or household needs, which cannot be effectively captured by a mono-intent model. Previous studies have proposed multi-intent approaches such as topic modeling, transformer-based, and clustering techniques. However, these methods assume that each intent has the same weight, especially when products are consumed simultaneously and do not have explicit labels that correlate between products, thereby reducing context understanding and personalization capabilities. Based on this gap, we propose a framework for implicit multi-intent detection in shopping transactions using the extensive knowledge of Large Language Models (LLMs) through the In-Context Learning (ICL) prompting technique. As for the preference assessment mechanism, we use the Position-Based Grouping (PBG) method to estimate user preferences based on the order of items added to the cart. The results of our experiments on the Instacart dataset show that our proposal is capable of producing a significant performance improvement compared to existing sequential recommendation systems, where our best model is able to increase Recall by up to 122% and MRR by up to 217%, indicating that it is more effective in capturing user preference trends for specific intentions in the purchase sequence. This work is available at https://huggingface.co/recommender-system/mindfull.

1. INTRODUCTION

E-commerce purchases reflect multiple needs in a single transaction. For example, a user may purchase milk and eggs for daily needs, stationery for work, and snacks for family entertainment, all in one shopping cart. This demonstrates the phenomenon of multi-intent transactions [1, 2].

Looking at current studies on recommendation systems, most still operate based on the assumption that one transaction corresponds to one intent (mono-intent) [3, 4]. This assumption simplifies the complexity of user behavior, resulting in recommendations that are irrelevant to user needs [4, 5], as shown in Figure 1. Seeing this weakness, recent research has begun to develop a multi-intent approach [6-9].

Several studies have proposed multi-intent models [9-11], including those that identify product subgroups within a transaction or apply topic models [10], transformers [11], and clustering techniques to distinguish user intent [9]. However, these methods still consider the weight of each user intent to be equal, especially when products are consumed simultaneously and do not have explicit labels [11].

The latest models in recommendation systems have adopted transformer-based architectures [11] that have strong natural language understanding capabilities. Although previous studies have shown limitations, transformer-based methods with the emergence of Large Language Models (LLMs) are beginning to offer new capabilities, including classification through hint-based inference, which promises to overcome multi-intent challenges, despite high computational demands. Some of the inference techniques with LLMs include (1) Zero-shot Learning [12], which allows the model to perform classification without seeing specific examples from previous tasks, but rather relies on general knowledge acquired during pre-training. (2) Few-shot Learning [13] improves task-specific understanding by including several examples in a single command. (3) In-Context Learning (ICL) [14] presents a series of examples in a single command without requiring fine-tuning, allowing the model to make predictions on target inputs directly.

Seeing the opportunity of LLM-based methods to address multi-intent problems, we propose a new approach that utilizes ICL-based inference prompting and introduces a mechanism to assess the intensity of user preferences for each detected intent. In general, the main contributions of our research are as follows:

a. An ICL-based multi-intent detection method that eliminates the need for explicit labels.
b. Implicit transaction data from purchase sequences as indicators of user interest.
c. A Position-Based Grouping (PBG)-based preference

scoring mechanism to identify dominant intentions.

d. Improved recommendation relevance that takes into account context diversity [15] and the strength of user preferences (intention interpretation) within a single transaction session.

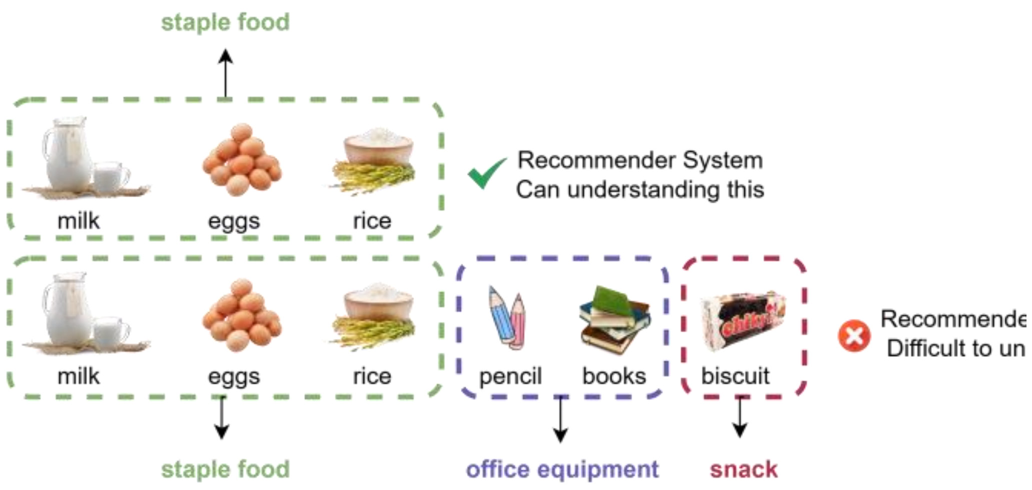


Figure 1. Example of a multi-intent problem
Note: The top represents the case expected by conventional recommendation models, while the bottom depicts a real-world case where user needs are diverse.

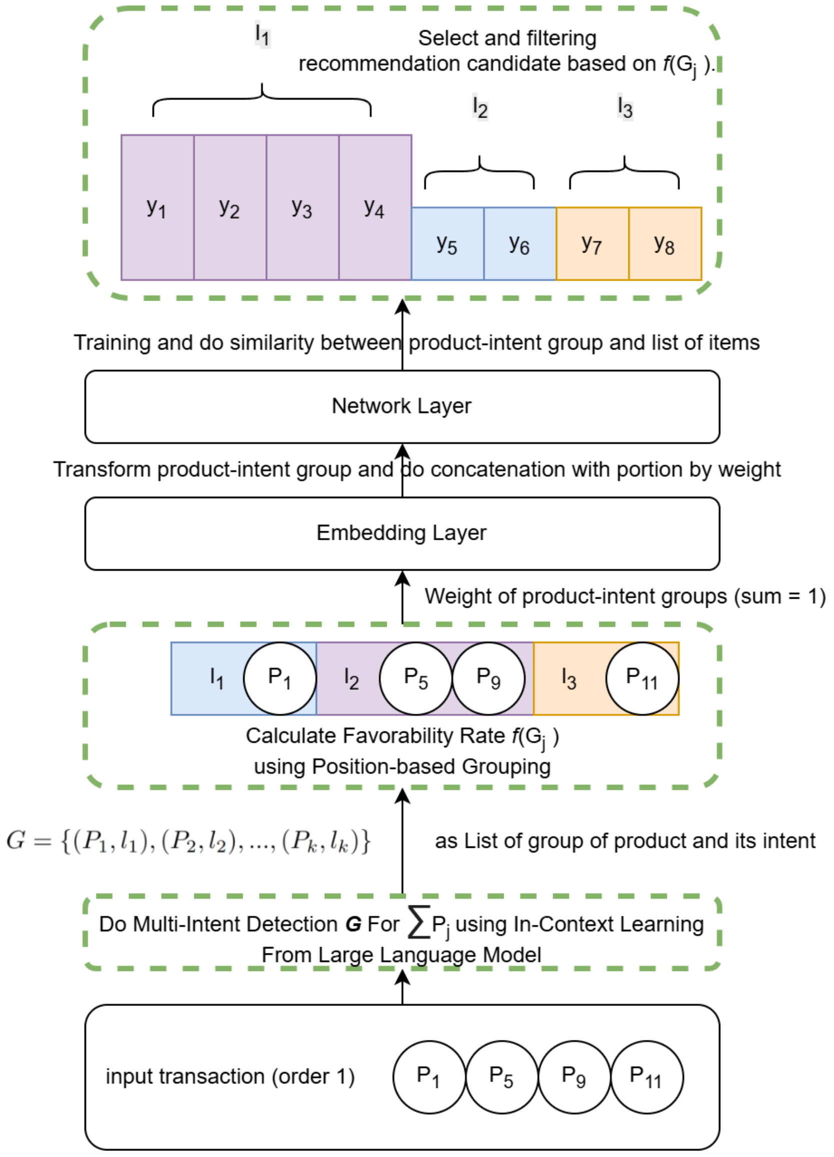

I_1
Select and filtering
recommendation candidate based on $f(G_j)$.
I_2
I_3
y_1
y_2
y_3
y_4
y_5
y_6
y_7
y_8
Training and do similarity between product-intent group and list of items
Network Layer
Transform product-intent group and do concatenation with portion by weight
Embedding Layer
Weight of product-intent groups (sum = 1)
I_1
P_1
I_2
P_5
P_9
I_3
P_{11}
Calculate Favorability Rate $f(G_j)$
using Position-based Grouping
$G = \{(P_1, l_1), (P_2, l_2), ..., (P_k, l_k)\}$
as List of group of product and its intent
Do Multi-Intent Detection *G* For $\sum P_j$ using In-Context Learning
From Large Language Model
input transaction (order 1)
P_1
P_5
P_9
P_{11}

Figure 2. Architecture of the proposed framework
Note: The components highlighted in green represent our novel contributions.

2. PROPOSED METHOD

To implement the proposed method, our approach is applied within a user transaction session framework, where each transaction T is represented as a sequence of products $[p_1, p_2, \ldots, p_n]$ ordered by the time each item p_i was added to the basket. This sequence is assumed to reflect the user's implicit preference level toward each product. The overall architecture of the proposed model is illustrated in Figure 2.

Table 1. List of notation

Notation	Description
T	List of products in a user transaction $T = [p_1, p_2, \ldots, p_n]$
p_i	The *i*-th product in the transaction
G	Set of predicted intent groups $G = \{(P_1, l_1), \ldots, (P_k, l_k)\}$
P_j	Subset of products belonging to the *j*-th intent group
l_j	Intent label for the product group P_j
$s(p_i)$	Base score of product p_i based on its position in the basket
α	Positional score decay coefficient (e.g., 0.1)
$\beta(G_j)$	Bonus score for intent group G_j based on consecutive product pairs
γ	Bonus coefficient for sequential product pairs (e.g., 0.1)
$S(G_j)$	Total score for intent group G_j
$f(G_j)$	Favorability rate, or the user's preference level for intent group G_j
S	Set of sequential product pairs in intent group G_j
$\phi(x)$	Embedding function for converting text into semantic vectors
$\cos(\phi(a), \phi(b))$	Cosine similarity between two text embeddings

The primary task is to detect groups of latent purchase intents G underlying the transaction, represented as $(P_1, l_1), \ldots, (P_k, l_k)$ where $P_j \subseteq T$ and l_j is the intent label assigned to the product subset P_j. In addition to intent detection, the system is designed to compute the user's preference level $f(G_j)$ for each intent group G_j, using a score-based ranking mechanism derived from item ordering. A summary of notations used is provided in Table 1.

2.1 In-Context Learning for multi-intent detection

The multi-intent detection component in this study leverages ICL with a Large Language Model (LLM) to infer latent intent structures within a transaction. In this approach, the LLM is provided with a set of example pairs consisting of products p_i and their corresponding categories c_i in the form of a prompt which serve as contextual demonstrations for the model. After receiving these examples, the model is instructed to cluster the products P_j in the target transaction into multiple intent groups l_j. The output is formally defined in Eq. (1).

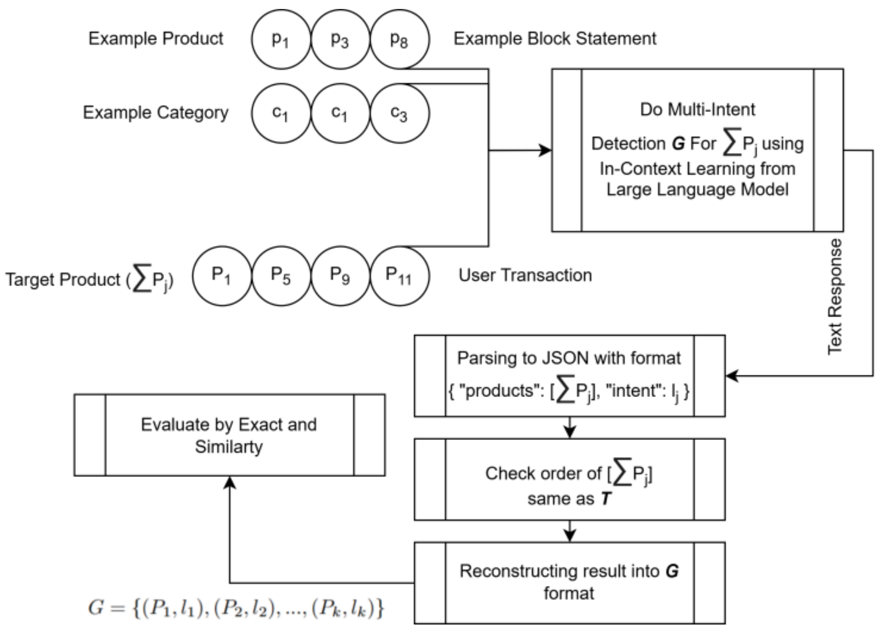
$$G = (P_1, l_1), (P_2, l_2), \ldots, (P_k, l_k) \tag{1}$$

The proposed steps for utilizing ICL in multi-intent detection are illustrated in Figure 3.

The ICL prompt is designed with two main components, namely:

(a) Context Examples, these consist of k product–category pairs (p_i, c_i) that guide the LLM in understanding semantic and functional relationships between products. In our implementation, we use 5-8 examples per prompt, selected from diverse aisles to maximize category coverage. Examples are sampled using a semantic similarity filter, ensuring that products included in the prompt remain representative of user shopping behaviors. Increasing the number of examples tends to improve intent coherence but also increases inference latency. Conversely, fewer examples may yield unstable or inconsistent grouping.

(b) Target Transaction, this component contains the list of products P_j from the user's transaction to be assigned into intent groups l_j. The model is explicitly instructed to generate up to *n* intents (where *n ≤ 5*) to maintain interpretability and avoid excessive fragmentation. An illustration of the prompt structure is shown in Algorithm 1.



Example Product
p1 p3 p8
Example Block Statement
Example Category
c1 c1 c3
Do Multi-Intent Detection G For ∑Pj using In-Context Learning from Large Language Model
Target Product (∑Pj)
P1 P5 P9 P11
User Transaction
Text Response
Parsing to JSON with format { "products": [∑Pj], "intent": lj }
Check order of [∑Pj] same as T
Reconstructing result into G format
Evaluate by Exact and Similarty
G = {(P1, l1), (P2, l2), ..., (Pk, lk)}

Figure 3. Flow design for multi-intent detection using In-Context Learning (ICL)

Algorithm 1. Example of an ICL prompt used for multi-intent detection

```
Products - Categories
- Banana - Fresh Fruits
- Whole Milk - Dairy
- Cheddar Cheese - Cheese

Transaction:
["Banana", "Whole Milk", "Cheddar Cheese", "Notebook"]

Task: Group the products in the transaction into up to 5 purchase
intents.
Return in the format: {products: [...], intent: "..."}
```

Since LLMs generate responses in free-text form, additional post-processing steps are required to ensure structural consistency and compatibility with subsequent modules. These steps include:

a. Extracting product groups and intent labels, each LLM-generated cluster is identified and mapped to a standardized format, ensuring uniform labeling across transactions.
b. Preserving product order, the original order of items, as added to the user's cart, is maintained. This is critical for computing the preference score in the PBG module.
c. Reconstructing the output format, the cleaned and structured results are reformatted according to Eq. (1), ensuring consistent input for downstream components.

2.2 Preference scoring with Position-Based Grouping

Once the intent groups G have been identified, the next step is to measure the degree of dominance of each intent within a transaction, denoted as $S(G_j)$ and $f(G_j)$. This process leverages the order in which products are added to the basket as a signal of implicit user preference.

PBG is the proposed approach to quantify user preference for a given intent within a transaction, based on the relative position of each product when it was added to the cart [16]. The method assumes that the earlier a product is added, the more likely it reflects a dominant or favored intent.

The preference scoring process consists of the following steps:

2.2.1 Base product score based on position

Each product in a transaction is assigned a base score $s(p_i)$, calculated linearly based on its position (Eq. (2)).

$$s(p_i) = 1.0 - \alpha(i-1) \tag{2}$$

where, $\alpha = 0.1$ Products added earlier in the basket receive higher scores, indicating stronger interest.

Example: If the basket sequence is (1) Banana, (2) Whole Milk, (3) Cheddar Cheese, (4) Notebook, then the base scores are $s(p_i) = [1.0, 0.9, 0.8, 0.7]$.

2.2.2 Bonus for Consecutive Products within an Intent Group

If two or more products within the same intent group G_j appear consecutively in the transaction, a bonus score is added (Eq. (3)).

$$\beta(G_j) = \gamma \cdot |S| \tag{3}$$

where, S is the number of consecutive product pairs within G_j, and $\gamma = 0.1$.

Example: After applying intent prediction with ICL, the resulting groups are, (1) Fresh Fruits – Banana, (2) Dairy Products – Whole Milk + Cheddar Cheese, (3) Stationery – Notebook.

The intent group Dairy Products qualifies for the bonus since its products appear consecutively. Thus, the bonus is reflected in the final scores as $\beta(G_j) = [(1.0 + 0.0), (0.9 + 0.8 + 0.1), (0.7 + 0.0)]$.

2.2.3 Total intent score

The total score for each intent group is computed by summing the base scores of all products within the group and adding the bonus score if applicable (Eq. (4)).

$$S(G_j) = \sum_{p \in P_j} s(p) + \beta(G_j) \tag{4}$$

Example: After including the consecutive-item bonus, the total intent score becomes $S(G_j) = [1.0+1.8+0.7] = 3.5$.

2.2.4 Normalization into favorability rate

To assess the dominance of one intent over others within the same transaction, each total intent score is normalized (Eq. (5)).

$$f(G_j) = \frac{S(G_j)}{\sum_{m=1}^{k} S(G_m)} \tag{5}$$

The value of $f(G_j) \in [0,1]$ represents the user's preference level for intent G_j. The sum of all $f(G_j)$ values in a transaction equals 1.0.

Example: After obtaining the total intent scores, normalization is applied to compute the favorability rates $f(G_j)$:

$$f(\text{Fresh}) = \frac{1.0}{3.5} = 0.286$$
$$f(\text{Dairy}) = \frac{1.8}{3.5} = 0.514$$
$$f(\text{Stationery}) = \frac{0.7}{3.5} = 0.200$$

By leveraging the Favorability Rate, the system can generate a composition of recommendations derived from diverse user intentions and rank the Top-K recommendations based on the weight of the most dominant intent. This results in increased contextual relevance, increased diversity in recommendations, and improved interpretability.

2.3 Similarity-based recommendation

After obtaining the favorability rates, the next stage involves generating recommendations. This begins with transforming the item embeddings $\phi(x)$ for each product in the intent group. If an intent contains more than one product, their embeddings are concatenated using max pooling to form a single representation.

Next, the number of recommendation candidates allocated to each intent is determined proportionally based on its favorability score relative to the total Top-K size. For example, in a Top-10 recommendation setting, if the Dairy Products intent has a favorability score of 0.5, it will be allocated 5 recommendation slots.

To generate the final recommendations, we utilize a vector database (ChromaDB (https://www.trychroma.com/)) in combination with cosine similarity to retrieve candidate items that semantically align with the detected intents. Cosine similarity is employed for its effectiveness in measuring the similarity between two objects $\cos(\phi(a), \phi(b))$, the effectiveness of measuring the similarity between two objects, such as "Bread" and "Rice" which are the "staple food" group will be separated by "Coffee" which is a "drink", by considering the direction of the vector representation, not just the magnitude [17].

3. EXPERIMENTS

In this study, a systematic preparation phase was conducted prior to the experimental stage to ensure the validity and reliability of the results. The preparatory steps covered several key aspects as outlined below.

3.1 Objectives

The experiments conducted using our framework in this study aim to achieve the following goals.

a. To evaluate how effectively LLMs can infer users' favorability toward items based on their needs within a transaction, and how this affects sequential recommendation outcomes.
b. To understand the impact of LLMs and the ICL scheme in detecting diverse user needs (multi-intent) within a single transaction session.

3.2 Dataset

The dataset used in this study is the Instacart Online Grocery Shopping Dataset 2017 (https://www.kaggle.com/competitions/basket-analysis/overview), which contains historical online shopping data from 206,209 users. In total, there are 3,421,083 shopping transactions (orders), covering 49,688 unique products. Each product is classified into one of 134 subcategories (aisles), which are further grouped into 21 main categories (departments) [18]. Every user has a chronological sequence of transactions, including timestamps and inter-order intervals. A summary of the dataset is presented in Table 2.

Table 2. Summary of the Instacart 2017 dataset

Component	Count	Description
Users	206,209	Each user has a sequence of transactions with timestamps.
Orders	3,421,083	Includes prior, train, and test orders.
Products	49,688	Unique products with names and IDs.
Aisles (Subcategories)	134	Product subcategories such as fresh vegetables, candy chocolate, etc.
Departments (Categories)	21	Main product categories such as produce, dairy eggs, beverages, etc.

To ensure the dataset's suitability for our experiments, several pre-processing steps were applied:

a. Session construction for each user based on the chronological order of transactions.
b. Filtering of users and products based on a minimum number of interactions (e.g., ≥ 5).
c. Tokenization of product names to enable sequential input representation for the LLM.
d. Dataset partitioning into training, validation, and test sets using a chronological split method, where early data is used for training and the most recent data is reserved for next-product prediction evaluation.

3.3 Models used

This experiment compares the proposed framework utilizing LLM with classical baseline models for the sequential recommendation task. The evaluated models are grouped into two categories as follows:

3.3.1 Large language model as model for extract multi-intent from transaction

The primary models used for multi-intent detection in this experiment consist of three recent LLMs:

a. LLaMA 3, a language model from Meta AI, designed based on an auto-regressive transformer architecture.
b. Qwen 3, an Alibaba open-source LLM, optimized for both efficiency and performance in natural language understanding and generation tasks.
c. Gemma 3, a lightweight model from Google, designed for high efficiency and easy deployment, particularly suited for recommendation and sequential processing tasks.

These models were selected due to their strong capabilities in extracting information and understanding context for predictions via ICL, as well as their open-source availability and compatibility with our hardware requirements [19].

3.3.2 Baseline model for sequential recommendation

To ensure a fair and comprehensive evaluation, we also implemented the following baseline models:

a. Popularity-Based: Recommends products based on overall purchase frequency.
b. GRU4Rec: A recurrent neural network (GRU-based) model effective for session-based recommendation tasks.
c. SASRec: A self-attention-based model that explicitly models item sequences.
d. BERT4Rec: A transformer-based model that leverages bidirectional context for item sequence modeling.

These baseline models were chosen due to their robust performance and popularity in the sequential recommendation domain. However, a significant limitation of most of these models is their reliance on item ID representations, i.e., numeric IDs, for recommendations without leveraging the rich semantics of item attributes. To address this, we modify these baseline models to allow the use of item attributes by introducing a mapping mechanism from item IDs to their corresponding attribute vectors [20].

3.4 Evaluation protocol and metrics

The experiments were conducted using a leave-one-out evaluation approach, which involves the following steps:

a. For each user, the most recent purchase session is treated as the ground truth.
b. All items not previously purchased by the user are considered as negative candidates.
c. The model is tasked with predicting the next item(s) based on the user's preceding session history.

To assess prediction performance, two standard evaluation metrics commonly used in recommender systems are employed: Recall@K and MRR@K. The values of K tested in this study are 5, 10, 15, and 20, in accordance with established practices in session-based recommendation experiments.

3.5 Implementation detail

The experiments were implemented in Python using the Hugging Face Transformers library for Large Language Model (LLM) integration. Tokenization was performed on either product IDs or product names within each purchase session.

Several key hyperparameters were used consistently across all models. The maximum user interaction history length was set to *max_len = 50*. For semantic product representation, a combined text-based encoder, *all-MiniLM-L6-v2*, was used as the *embedding_model*. Item embeddings were aggregated using *max_pooling*. The *batch size* during training was set to *128*, while evaluation was performed with a *batch size* of *1*.

The loss function used was *BCEWithLogitsLoss()*, suitable for multi-label prediction scenarios. Model performance was evaluated using two primary metrics: *Recall@K* and *MRR@K*, with *K* values varied across *[5, 10, 15, 20]*. All models were trained for *50* epochs, as specified by *training_epochs = 50*. Parameter optimization was performed using the *Adam optimizer* with a *learning rate* of *0.001*, denoted as *Adam (lr = 0.001)*. All experiments were executed on a *12 GB NVIDIA GPU (RTX 4070 Super)*.

3.6 Experimental results

Overall performance. As presented in Table 3, the proposed framework consistently outperforms the classical baseline models in almost all evaluation metrics. Recall@K measures how many relevant items are successfully retrieved in the Top-K list of recommended items, where a higher score indicates better coverage of user preferences. Meanwhile, MRR@K evaluates the ranking position of the first relevant item in the Top-K list; a higher score indicates that the relevant item appears earlier in the recommendation list. The improvements are quite significant, with the framework achieving up to +122.56% improvement in Recall and +217.30% improvement in MRR compared to the best baseline model. This highlights the framework's ability to capture diverse user needs (multi-intent) and semantic representations more effectively than traditional sequence-based models such as GRU4Rec or SASRec.

Table 3. Comparison of recommender system performance

Model	Recall@5	Recall@10	Recall@15	Recall@20	MRR@5	MRR@10	MRR@15	MRR@20
Popularity	**0.1358**	**0.1618**	**0.1693**	**0.1693**	<u>0.2150</u>	0.2283	0.2298	0.2293
GRU4Rec	0.0554	0.0802	0.0938	<u>0.1233</u>	**0.2295**	<u>0.2402</u>	<u>0.2432</u>	<u>0.2472</u>
SASRec	<u>0.0854</u>	<u>0.0890</u>	<u>0.1089</u>	0.1203	0.2121	**0.2733**	**0.2752**	**0.2761**
BERT4Rec	0.0485	0.0723	0.0872	0.1046	0.1983	0.2127	0.2152	0.2193
Ours-gemma3-4b	**0.2585**	**0.2786**	**0.3103**	<u>0.3080</u>	**0.7282**	**0.7934**	**0.6931**	**0.6538**
Ours-llama3-8b	<u>0.2165</u>	<u>0.2415</u>	<u>0.2913</u>	**0.3768**	<u>0.4029</u>	<u>0.3652</u>	<u>0.4351</u>	<u>0.4680</u>
Ours-qwen3-4b	0.1238	0.2071	0.2171	0.2704	0.2331	0.2818	0.2756	0.2769
Improvement (%)	**+90.35%**	**+72.19%**	**+83.28%**	**+122.56%**	**+217.30%**	**+190.30%**	**+151.85%**	**+136.80%**

Note: **Bold** values indicate the best overall performance, while *underlined* values denote the best among baseline models.

Table 4. Statistical significance (p-values) across metrics

Model	Metric	Popularity	Bert4Rec	GRU4Rec	SASRec
gemma3	MRR@5	0.0000***	0.0000***	0.0000***	0.0004***
gemma3	Recall@5	0.0000***	0.0000***	0.0000***	0.0000***
gemma3	MRR@10	0.0000***	0.0000***	0.0001***	0.0138*
gemma3	Recall@10	0.0000***	0.0000***	0.0000***	0.0006***
gemma3	MRR@15	0.0001***	0.0000***	0.0010**	0.1098
gemma3	Recall@15	0.0000***	0.0000***	0.0001***	0.0133*
gemma3	MRR@20	0.0001***	0.0000***	0.0024**	0.1858
gemma3	Recall@20	0.0001***	0.0005***	0.1569	0.0874
llama3	MRR@5	0.0766	0.0093**	0.3322	0.408
llama3	Recall@5	0.0000***	0.0000***	0.0017**	0.0626
llama3	MRR@10	0.0156*	0.0013**	0.062	0.9219
llama3	Recall@10	0.0000***	0.0007***	0.0152*	0.2002
llama3	MRR@15	0.071	0.0098**	0.3142	0.3774
llama3	Recall@15	0.0000***	0.0246*	0.1476	0.8336
llama3	MRR@20	0.1201	0.0299*	0.5761	0.2845
llama3	Recall@20	0.0000***	0.356	0.2455	0.3758
qwen3	MRR@5	0.0170*	0.1151	0.0020**	0.0000***
qwen3	Recall@5	0.0000***	0.0056**	0.0602	0.3749
qwen3	MRR@10	0.0009***	0.0144*	0.0001***	0.0000***
qwen3	Recall@10	0.0000***	0.3039	0.8345	0.4785
qwen3	MRR@15	0.0007***	0.0056**	0.0001***	0.0000***
qwen3	Recall@15	0.0000***	0.1867	0.0451*	0.0011**
qwen3	MRR@20	0.0005***	0.0029**	0.0000***	0.0000***
qwen3	Recall@20	0.0000***	0.0405*	0.0003***	0.0006***

Note: Values represent *p-values*. Asterisks denote the statistical significance level of the performance difference between the proposed and baseline models: * indicates $p < 0.05$, ** indicates $p < 0.01$, and *** indicates $p < 0.001$. Absence of asterisks indicates no statistical significance ($p \geq 0.05$).

These performance gains are statistically validated in Table 4. The significance tests confirm that the improvements are robust and not due to random chance. Specifically, the Ours-gemma3-4b variant demonstrates highly significant superiority (*p* < *0.001*) against almost all baselines, particularly in MRR@5 and MRR@10. This indicates that the model is exceptionally reliable in ranking relevant items at the very top of the list. While Ours-llama3-8b and Ours-qwen3-4b show significant improvements against weaker baselines (e.g., Popularity, BERT4Rec), their statistical advantage narrows against stronger baselines like SASRec at higher *K* values (e.g., MRR@20), suggesting that while they improve recall, their ranking precision is competitive but less dominant than the Gemma variant.

Among the proposed variants, Ours-gemma3-4b stands out as the premier model, achieving peak performance with a Recall@5 of 0.2585 and MRR@10 of 0.7934. The dominant statistical significance *(p < 0.001)* observed in Table 4 correlates with its precise execution of ICL. Gemma3-4b effectively identifies and groups products from single transactions into meaningful intent clusters, directly translating to higher MRR scores.

Conversely, while Ours-llama3-8b and Ours-qwen3-4b perform well, they exhibit instability in ICL-based multi-intent detection. Despite having larger parameters (in Llama's case), Table 4 shows less consistent significance levels. Qualitatively, this is observed when transactions contain products with weak explicit semantic alignment. For instance, given items like whole milk, organic eggs, and butter cookies, these models occasionally mapped them to broad categories like dairy or miscellaneous instead of specific intended taxonomies. This "label smoothing" effect likely dilutes the precision needed for higher MRR scores, explaining why their statistical significance is less pronounced compared to the Gemma variant.

Among the baseline models, the Popularity-based model achieved the highest Recall scores. However, this approach suffers from significant limitations. As illustrated in Figure 4, the model is static and provides the same recommendation to all users based solely on item popularity, ignoring user preferences and context. Over time, this lack of variation and personalization may lead to reduced user engagement due to recommendation fatigue.

In contrast, SASRec and GRU4Rec outperform BERT4Rec, consistent with their design focus on modeling user interaction sequences (sequential recommendations). SASRec leverages a self-attention mechanism, effectively capturing both short-range and long-range dependencies. GRU4Rec, based on an RNN architecture, excels at learning explicit user interaction sequences. In contrast, transformer-based BERT4Rec suffers from a setback, as its masked language modeling approach, similar to BERT, is less effective in sequential recommendation settings due to the loss of explicit sequences, especially in data-constrained scenarios.

Impact of session length: To assess the effect of the number of items in a transaction on recommendation quality, we refer to Figure 5, which shows that the majority of transactions consist of between 2 and 15 products. This distribution is used as a basis for analyzing how session length influences the relevance of recommendations generated by each model.

As illustrated in Figure 6, the proposed framework consistently outperforms all baseline models across various session lengths (2, 5, 8, 10, and 15 items), for both Recall@20 and MRR@20 metrics. This performance advantage highlights the strength of LLMs and the ICL approach in understanding user intent, even in very short sessions.

Specifically, variant llama3-8b excels in short sessions due to its ability to generalize and capture purchase patterns from minimal input. Meanwhile, gemma3-4b and qwen3-4b demonstrate more stable performance in longer sessions, indicating their adaptability to more complex contextual information.

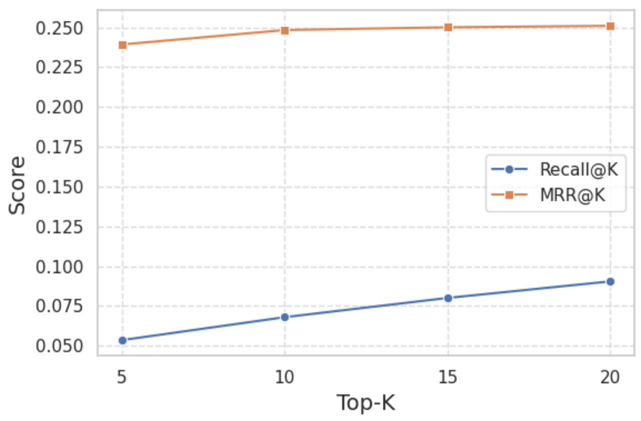


Figure 4. Average results of the popularity-based approach over 30 runs

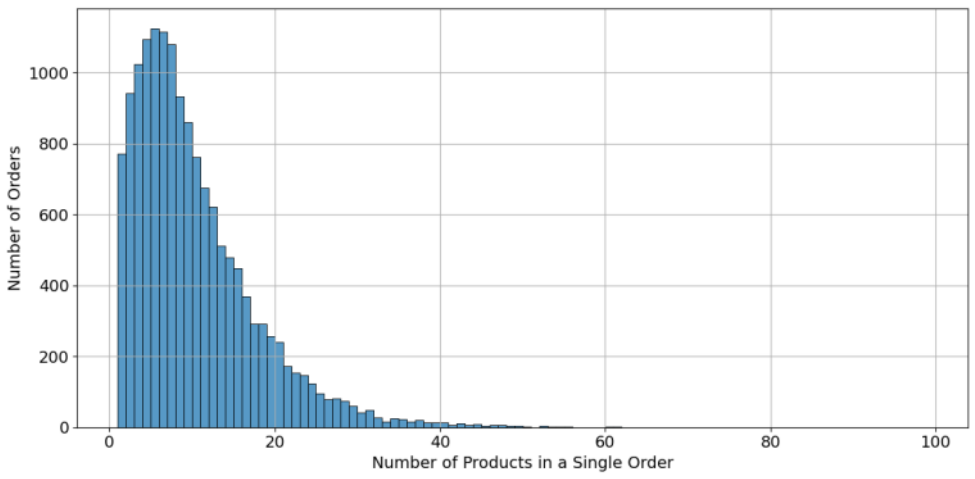


Figure 5. Distribution of transaction sizes (range 2-15)

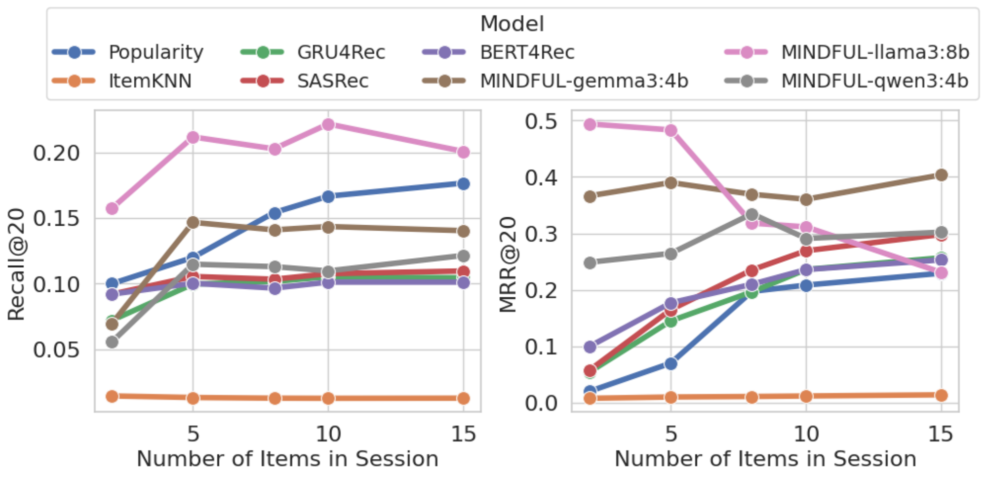


Figure 6. Impact of session length on recommendation relevance across models. (Left): Recall@20, (Right): MRR@20

Among the baseline models, Popularity shows relatively high Recall but low MRR, indicating that while popular products are frequently chosen, they are not always ranked in positions that reflect personal relevance. GRU4Rec and SASRec, both sequential models, show steady and improved performance as session length increases, reflecting their strength in leveraging user interaction history.

Overall, while longer sessions provide additional context that can benefit most models, the proposed framework remains superior, even in short-session scenarios, demonstrating an ability to generate context-aware and relevant recommendations under limited information.

Ablation study: To evaluate whether each component of the propose framework contributes significantly to the performance of the recommender system, we conducted a series of ablation studies. This experimental approach involves modifying or removing specific parts of the model architecture to observe their individual impact on performance. The primary objective is to verify that each component integrated into the proposed framework provides meaningful contributions to the overall results. Table 5 summarizes the results.

Table 5. Ablation study results on proposed framework components

Model	Recall@20	MRR@20
Intent Detection Removed		
without intent detection	0.1380	0.1136
Favorability Rate Removed		
Variant gemma3-4b (no favorability)	0.1732	0.2447
Variant llama3-8b (no favorability)	0.1502	0.2219
Variant qwen3-4b (no favorability)	**0.1902**	**0.3572**
Mean Pooling Embedding		
Variant gemma3-4b + mean pooling	**0.1715**	**0.4940**
Variant llama3-8b + mean pooling	0.1001	0.3476
Variant qwen3-4b + mean pooling	0.1056	0.2040

Note: **Bold** values indicate the best performance in each experimental block.

Removing the intent detection stage causes the most substantial decline in performance, with Recall@20 and MRR@20 dropping by more than 50% relative to the full model. This degradation occurs not only because the intent module is removed, but also because its absence alters the semantic structure of the input representation. Without intent grouping, all items in a transaction are merged into a single undifferentiated sequence, causing: (a) Loss of multi-intent structure, items associated with different underlying user goals are treated as belonging to the same intent. (b) Weaker embedding quality, the aggregated representation becomes noisier, as unrelated item signals collapse into a single vector. (c) Reduced alignment with LLM reasoning, since the downstream encoder expects structured intent groups, removing them breaks the intended information flow. Thus, the observed performance drop is a combined effect of removing intent detection and the resulting disruption in input organization.

Removing the preference scoring module results in a 40–50% decrease in accuracy. This decrease stems from the loss of relative preference signals arising from the order in which items are added to the shopping cart. Without this module, the framework can be falling about (a) All items are treated uniformly, (b) The influence of dominant intentions is not emphasized, and (c) the model cannot infer which sub-intentions are most likely to drive transactions. Therefore, the performance decrease is directly due to the removal of the scoring mechanism, which plays a crucial role in weighting intentions based on user behavior.

Replacing max pooling with mean pooling also leads to a notable drop in Recall (up to 50%). This occurs because mean pooling dilutes dominant behavioral signals by averaging them with irrelevant or low-importance items. Max pooling, in contrast, preserves the strongest activation dimensions that often correspond to high-salience items within an intent cluster.

Interestingly, the decline in MRR is smaller, suggesting that while fewer relevant items are retrieved overall, the model is still able to rank the most representative items reasonably well. This indicates that mean pooling primarily affects breadth of retrieval, whereas ranking fidelity remains partially intact.

4. CONCLUSIONS

This study introduces a novel recommendation system framework based on LLMs to address the challenges of multi-intent behavior within user transaction sessions. By integrating Deep Learning in Context (ICL) for intent mapping, a preference estimation mechanism through liking assessment, and an optimized embedding aggregation strategy, the proposed architecture effectively captures semantic and functional relationships between items. Experimental evaluations show that the framework consistently outperforms baseline models across various session length configurations and remains robust even in short-session scenarios. Further ablation studies verify that each component, particularly the intent detection, liking ratings, and max-pooling strategies, contributes significantly to performance.

Despite these promising results, several limitations require further investigation. First, the baseline models used in previous studies (e.g., SASRec and GRU4Rec) are primarily ID-based rather than text-based, requiring modifications to their embedding layers to accommodate textual representations. This modification may contribute to the performance degradation observed in some baseline models. Second, Transformer-based methods and LLM inference require substantial computational resources, including high memory capacity and careful management of disk-based processing to avoid memory exhaustion issues.

Furthermore, our current framework has only been evaluated on the Instacart dataset. Therefore, assessing its generalizability across multiple sequential recommendation datasets remains an important direction for future research. Another opportunity for improvement lies in exploring the diversity of recommendations to ensure that users not only receive suggestions that align with their preferences but also benefit from unintentional exposure to new items.

ACKNOWLEDGMENT

This research was funded by the Directorate of Research, Technology, and Community Service (DRTPM) under the Dissertation Doctor Research Scheme, with contract number 125/C3/DT.05.00/PL/2025. We would like to express our gratitude for their support, which made this research possible.

REFERENCES

[1] Najmani, K., Sael, N., Zellou, A. (2022). A systematic literature review on recommender systems for MOOCs. Ingenierie des Systemes d'Information, 27(6): 895-902. https://doi.org/10.18280/isi.270605

[2] Chen, C.W., Lamere, P., Schedl, M., Zamani, H. (2018). Recsys challenge 2018: Automatic music playlist continuation. In Proceedings of the 12th ACM Conference on Recommender Systems, pp. 527-528. https://doi.org/10.1145/3240323.3240342

[3] Hidasi, B., Karatzoglou, A. (2018). Recurrent neural networks with top-k gains for session-based recommendations. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 843-852. https://doi.org/10.1145/3269206.3271761

[4] Pan, Z., Cai, F., Chen, W., Chen, H., De Rijke, M. (2020). Star graph neural networks for session-based recommendation. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 1195-1204. https://doi.org/10.1145/3340531.3412014

[5] Yuan, J., Song, Z., Sun, M., Wang, X., Zhao, W.X. (2021). Dual sparse attention network for session-based recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 35(5): 4635-4643. https://doi.org/10.1609/aaai.v35i5.16593

[6] Liu, Z., Li, X., Fan, Z., Guo, S., Achan, K., Yu, P.S. (2020). Basket recommendation with multi-intent translation graph neural network. In 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, pp. 728-737. https://doi.org/10.1109/BigData50022.2020.9377917

[7] Bhattacharya, B., Burhanuddin, I., Sancheti, A., Satya, K. (2017). Intent-aware contextual recommendation system. In 2017 IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, LA, USA, pp. 1-8. https://doi.org/10.1109/ICDMW.2017.8

[8] Zhu, N., Cao, J., Lu, X., Xiong, H. (2021). Learning a hierarchical intent model for next-item recommendation. ACM Transactions on Information Systems (TOIS), 40(2): 1-28. https://doi.org/10.1145/3473972

[9] Liu, Y., Zhu, S., Xia, J., Ma, Y., Ma, J., Liu, X., Yu, S., Zhang, K., Zhong, W. (2024). End-to-end learnable clustering for intent learning in recommendation. Advances in Neural Information Processing Systems, 37: 5913-5949. https://doi.org/10.52202/079017-0192

[10] Choi, M., Kim, H.Y., Cho, H., Lee, J. (2024). Multi-intent-aware session-based recommendation. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2532-2536. https://doi.org/10.1145/3626772.3657928

[11] Zou, D., Wei, W., Zhu, F., Xu, C., Zhang, T., Huo, C. (2024). Knowledge enhanced multi-intent transformer network for recommendation. In Companion Proceedings of the ACM Web Conference 2024, pp. 1-9. https://doi.org/10.1145/3589335.3648296

[12] Ding, H., Ma, Y., Deoras, A., Wang, Y., Wang, H. (2021). Zero-shot recommender systems. arXiv preprint arXiv:2105.08318. https://doi.org/10.48550/arXiv.2105.08318

[13] Wang, Z. (2024). Empowering few-shot recommender systems with large language models-enhanced representations. IEEE Access, 12: 29144-29153. https://doi.org/10.1109/ACCESS.2024.3368027

[14] Bao, K., Yan, M., Zhang, Y., Zhang, J., Wang, W., Feng, F., He, X. (2024). Real-Time personalization for LLM-based recommendation with customized In-Context Learning. arXiv preprint arXiv:2410.23136. https://doi.org/10.48550/arXiv.2410.23136

[15] Omar, H.K., Frikha, M., Jumaa, A.K. (2024). PyTorch and TensorFlow performance evaluation in big data recommendation system. Ingenierie des Systemes d'Information, 29(4): 1357-1364. https://doi.org/10.18280/isi.290411

[16] Zehlike, M., Yang, K., Stoyanovich, J. (2022). Fairness in ranking, part I: Score-based ranking. ACM Computing Surveys, 55(6): 1-36. https://doi.org/10.1145/3533379

[17] Shrivastava, R., Sisodia, D.S. (2019). Product recommendations using textual similarity based learning models. In 2019 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, pp. 1-7. https://doi.org/10.1109/ICCCI.2019.8821893

[18] Rao, S., Zhang, L. (2021). The algorithms that make Instacart roll: How machine learning and other tech tools guide your groceries from store to doorstep. IEEE Spectrum, 58(3): 36-42. https://doi.org/10.1109/MSPEC.2021.9370062

[19] Chang, Y.P., Wang, X., Wang, J.D., Wu, Y., et al.

(2024). A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, 15(3): 1-45. https://doi.org/10.1145/3641289

[20] Betello, F., Purificato, A., Siciliano, F., Trappolini, G., Bacciu, A., Tonellotto, N., Silvestri, F. (2024). A reproducible analysis of sequential recommender systems. IEEE Access. 13: 5762-5772. https://doi.org/10.1109/ACCESS.2024.3522049