Recommender Systems Project - RecSys Challenge 2025

Sifei Meng, Karim Aitkhadzhaev

November 2025

1 Problem Statement

The ACM RecSys Challenge 2025 is about universal behavioral modeling. Instead of training a separate model for each business goal, participants learn one universal behavioral profile per user that can be reused across multiple prediction tasks. The idea comes from real settings where churn forecasting and propensity modeling rely on the same underlying behavioral signal, even though the targets and evaluation setups are different. That makes robustness and transferability of the user representation the main objective.

The dataset comes from a real e-commerce platform and covers several months of anonymized user activity. It includes product related actions such as purchases, add to cart, and remove from cart, along with page visits represented by URL identifiers and search interactions. Item metadata is also available, for example category identifiers and price expressed as percentile based buckets. Text fields like product names and search queries are provided as quantized embeddings to preserve privacy while keeping semantic similarity. During the pre training period, participants build user profiles for a fixed set of relevant users. The submission is a matrix of float16 user embeddings with up to 2048 dimensions, one vector per user.

Evaluation is done through a standardized downstream model trained by the organizers on a separate training window and tested on a later time window. The public tasks cover churn prediction as a binary problem over whether active purchasers stop buying in the next 14 days, plus product propensity and category propensity as multi label prediction over the most frequent products and categories. There are also hidden tasks designed to test generalization beyond what is disclosed, such as conversion related objectives or propensities tied to emerging items or price ranges. Because the dataset is heavily imbalanced, the main metric is AUROC. For propensity tasks, AUROC is combined with novelty and diversity components to avoid solutions that only learn popularity. The final ranking aggregates performance across tasks to reward user embeddings that transfer well and stay consistent across objectives.

2 Dataset Overview

Data Collection The SYNERISE dataset is a large scale e-commerce dataset collected from a real online retailer and released for research on sequential recommendation and universal behavior modeling [2]. It spans six months of activity and is highly sparse, with far more browsing than purchasing, which matches typical e-commerce behavior . For the competition protocol, the full timeline is split into three disjoint parts: a pre training time frame covering the first five months, a two week training window, and a two week evaluation window . Participants build Universal Behavioral Profiles using only the pre training interactions, while the organizers train and evaluate downstream models on the later windows .

Dataset statistics. Across the complete interaction log described in the dataset paper, the activity tables contain events from 22 million clients and refer to 1.53 million items . Explicit interactions, meaning purchases and cart actions, account for 12.5 million records, while implicit page navigation accounts for 212.6 million records . The detailed counts by table are shown in Table 2, and the overall statistics are shown in Table 1 . Purchase events are comparatively rare, with fewer than 3 million purchases in the full log, which implies that most users never purchase during the observation period .

Table 1: Overall Dataset Statistics

#Clients	#Items	#Explicit Interactions	#Implicit Interactions
22M	1.53M	12.5M	212.6M

Table 2: Activity Log Statistics (Complete Interaction Log)

Name of File	Number of Lines
product_buy	2,318,502
add_to_cart	7,541,117
remove_to_cart	2,688,894
page_visit	199,451,980
search_query	13,223,769

Files and schema. The dataset consists of five interaction logs and one item metadata file . The three product interaction logs, product_buy, add_to_cart, and remove_to_cart, store tuples with client_id, timestamp, and sku . The page_visit log stores client_id, timestamp, and url, where url is an anonymized integer identifier with no semantic relation to the original address . The search_query log stores client_id, timestamp, and query, where query is a quantized embedding of the original text . Item metadata is provided in product_properties as tuples with sku, category, price, and embedding . The price field is anonymized

into 100 percentile style buckets, and text fields such as item names and search queries are converted into similarity preserving embeddings and then quantized into byte vectors of length 16 .

Competition objective and downstream tasks. The core competition objective is to generate a Universal Behavioral Profile for each relevant client as a real valued vector, and submit the matrix

$$P \in \mathbb{R}^{1M \times d}, \quad 0 \leq d \leq 2048,$$

encoded in float16 precision . These profiles are evaluated through downstream prediction tasks that are grouped into churn type tasks and propensity type tasks . The visible tasks include churn prediction, product propensity over the 100 most popular products, and category propensity over the 100 most popular categories . The hidden tasks include conversion prediction for all relevant clients, propensity over 20 popular products that are not present in the pre training period, and propensity over 100 price buckets .

Evaluation model. All submissions are evaluated using the same small neural network, called the Universal Behavioral Model, with minimal task dependent adaptations such as input size, output size, and target construction . The architecture consists of an input linear layer with layer normalization, three bottleneck blocks with a GeLU nonlinearity, and an output linear layer . In the competition setting, the bottleneck dimensions are fixed to hidden_size_thin equal to 2048 and hidden_size_wide equal to 4048 .

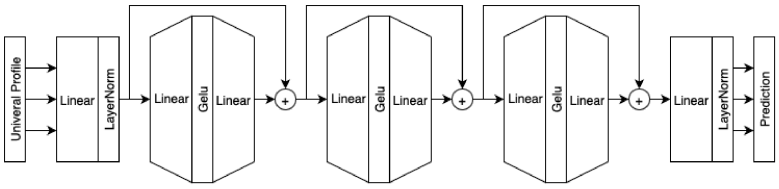

Figure 1: Evaluation model.

Evaluation metrics. The primary metric across tasks is AUROC, which is preferred because the dataset is highly imbalanced and many users have no purchases in the evaluation window . Churn type tasks are scored only by AUROC . Propensity type tasks combine accuracy with beyond accuracy metrics: multi label AUROC contributes 80 percent of the task score, and diversity and novelty contribute 10 percent each . Given a recommendation vector $p \in [0,1]^d$, it is first normalized into a probability distribution

$$\hat{p}_i = \frac{\sigma(p_i)}{\sum_k \sigma(p_k)}, \qquad \sigma(x) = \frac{e^x}{1+e^x},$$

and diversity is computed as the entropy

$$D(p) = -\sum_i \hat{p}_i \log \hat{p}_i.$$

Novelty is defined through the popularity of recommended labels, where q is the ℓ_1 normalized purchase frequency of target labels in the training window, and the novelty score is

$$\text{novelty} = (1 - \text{popularity})^\alpha, \qquad \alpha = 100,$$

with the regularization exponent chosen to make the metric sensitive under sparsity .

3 Exploratory Data Analysis

Data snapshot and the real data-quality risk: event multiplicity. The input consists of four behavioral logs (`search`, `add_to_cart`, `remove_from_cart`, `buy`) and an item table (`products`). We observe 2,832,878 unique users and 1,260,365 unique SKUs, i.e., a massive and extremely sparse user–item universe. Item metadata has no missing values in the provided fields (category, price, anonymized name representation), so side features are consistently available for cold-start and reranking.

The dominant integrity concern is duplication (Table 3), especially for purchases. Importantly, duplicates are ambiguous: they can be (i) genuine repeated actions (quantity-like intensity, repeated purchases, repeated cart edits), or (ii) logging artifacts. This ambiguity matters because naive deduplication can destroy a meaningful intensity signal, while keeping duplicates can inflate frequency features and cause trivial self-transitions in sequential modeling. A robust compromise is: (a) build *deduplicated* event streams for sequence learning and transition analysis, while (b) preserving the multiplicity as a separate *count/weight* feature for aggregation-based models.

Table 3: EDA dataset snapshot and duplicate rates in the provided input logs.

Table name	Rows	Duplicates	Duplicate rate (%)
search	10,218,831	576,286	5.64
add_to_cart	5,674,064	107,117	1.89
remove_from_cart	1,937,170	104,293	5.38
buy	1,775,394	283,545	15.97
products	1,260,365	0	0.00

Behavioral coverage: the funnel exists, but journeys have multiple entry points. Many teams think about behavior as a simple path from search to cart to buy. The user funnel does show a steep drop from exploration to

transactions in Figure 3. This means any purchase-based objective is trained on a very imbalanced signal and cannot rely on buys alone.

The more important point is that users do not follow one standard route. A meaningful fraction reaches cart and sometimes purchase with little or no search. This is consistent with recommendation feeds, direct product links, external campaigns, and repeat purchasing. We can see this in user aggregates where cart or purchase counts are non-zero even when search is zero, and where `n_cart` can exceed `n_search`. That is why search cannot be treated as a universal entry action. The representation has to fuse multiple channels without forcing a fixed order.

Figure 2 is operationally important because it separates users who only search from users who ever go deeper into cart, remove, or buy. If search-only users are a large share, a model that looks strong on transactional users can still be weak overall. For UBP, this means we need to represent weak intent exploration and deep intent conversion in the same embedding space.

Long-tail dynamics: it changes what is learnable and what is easy to game. Per-user activity is extremely uneven. Figure 4 shows that most users have very few item actions, while a small minority produces many. This is not just a data shape. It changes training and evaluation.

Heavy users can dominate gradients and push the model toward their preferences. This can be reduced by sampling users more uniformly, capping events per user per epoch, or using sequence compression so repetition does not count as new evidence. Low-activity users also create noisy ratios and unstable features. Log transforms and robust scaling help make user-level features usable without letting a few outliers set the scale.

The same pattern appears on the item side. Purchases per SKU in Figure 5 form a strong popularity head and a very long tail. This creates a shortcut. A model can get decent recall by leaning on head popularity, especially if evaluation is popularity-friendly. A competitive UBP needs signals that separate users with similar head exposure and still capture personal preference. Tail-aware negatives and debiasing matter here, otherwise the embedding learns popularity more than intent.

Search queries are identifiers, not natural language. The query strings are anonymized. Query length is almost constant in Figure 6. A single length of 65 characters accounts for 10,131,974 of 10,218,831 search events, about 99.15 percent. This strongly suggests the query field behaves like an identifier, not free-form text.

This changes what features are meaningful. Query length is not a semantic proxy here. The value of a query comes from behavioral co-occurrence, such as which items users click, add, remove, or buy after that query. Modeling should treat query as a categorical entity with a learned embedding, and focus on the query to item and query to user relationships. Figure 7 should be read as a sanity check and a user-type effect, not as evidence that a certain query length

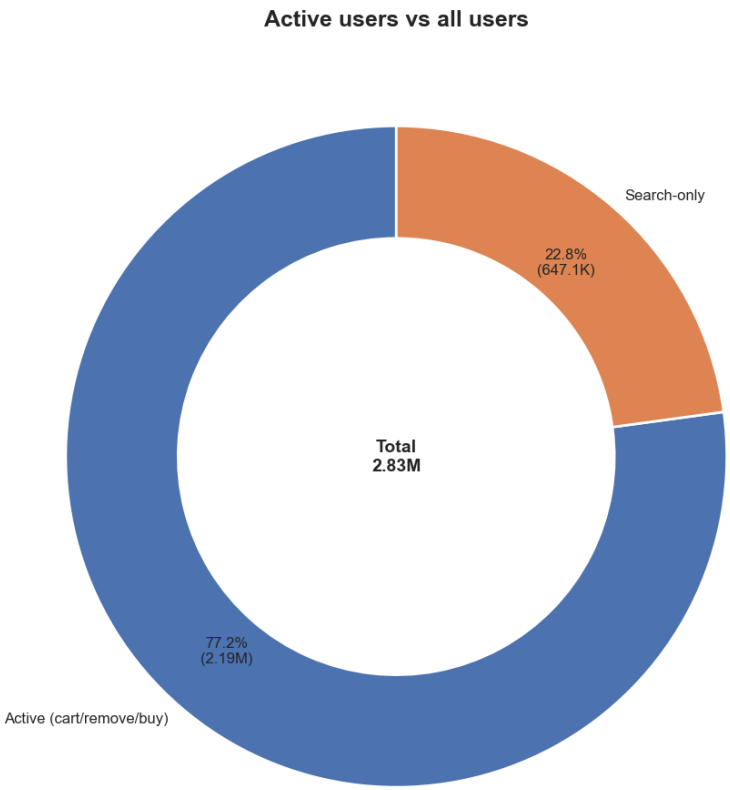

Figure 2: Active users vs all users (donut chart). A substantial search-only mass indicates that many users never enter transactional stages within the log window, so UBP must model both weak-intent exploration and deep-intent conversion behaviors.

causes better conversion.

Temporal dynamics: volatility implies drift and pushes toward recency-aware UBPs. Daily volumes in Figure 8 show dense search and sparse, volatile purchase activity. This suggests non-stationarity from campaigns, seasonality, and assortment changes. It also means random splits can be misleading if they leak future patterns into training.

For modeling, this motivates time-based validation and recency-aware aggregation. A UBP that gives too much weight to old behavior can lag behind current intent. Simple fixes like time buckets, exponential decay, or recency-weighted pooling can improve robustness under drift.

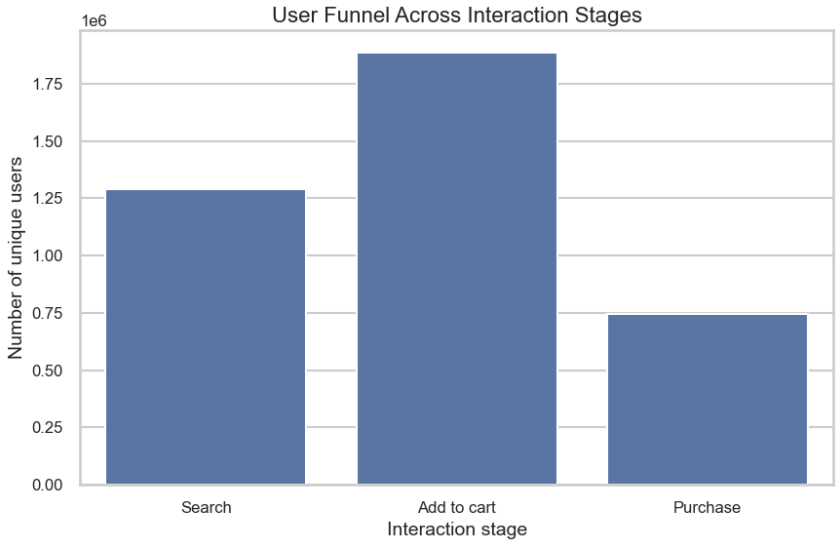

Figure 3: Unique-user funnel across stages. The steep drop quantifies label imbalance for purchase-oriented objectives and motivates leveraging dense signals from search/cart, while recognizing that many journeys enter the funnel midstream (bypass search).

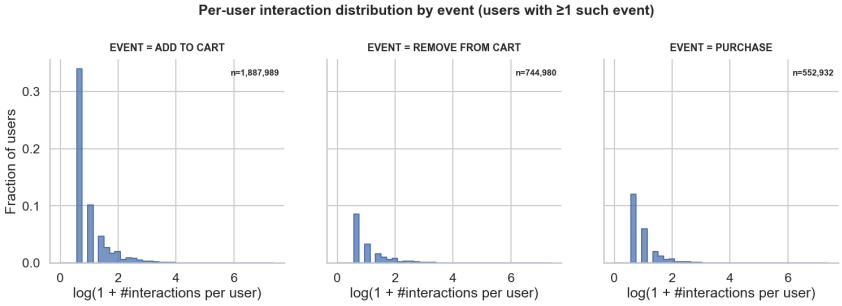

Figure 4: Per-user interaction distributions by event (users with ≥ 1 such event), plotted in log space. Long tails imply strong heterogeneity and motivate robust aggregation (log/robust scaling) and sampling strategies that avoid heavy-user domination.

Conversion heterogeneity: cart to buy ratios expose regimes and potential anomalies. User-level conversion varies widely. Using the ratio `n_buy/n_add` when `n_add` is non-zero, the distribution in Figure 9 is heavily concentrated at zero. The median is 0 and the 75th percentile is also 0, while the mean is 0.233 and the maximum reaches 114.5. This is more than generic heterogeneity. It points to multiple regimes.

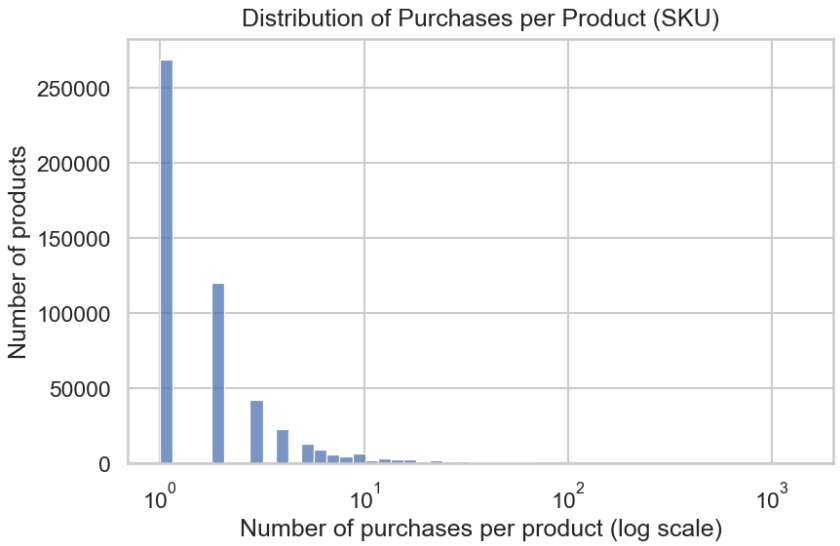

Figure 5: Purchases per SKU (log scale). Strong popularity imbalance implies that naive popularity signals can shortcut recall metrics; tail-aware training and debiased negatives are needed for genuinely personalized representations.

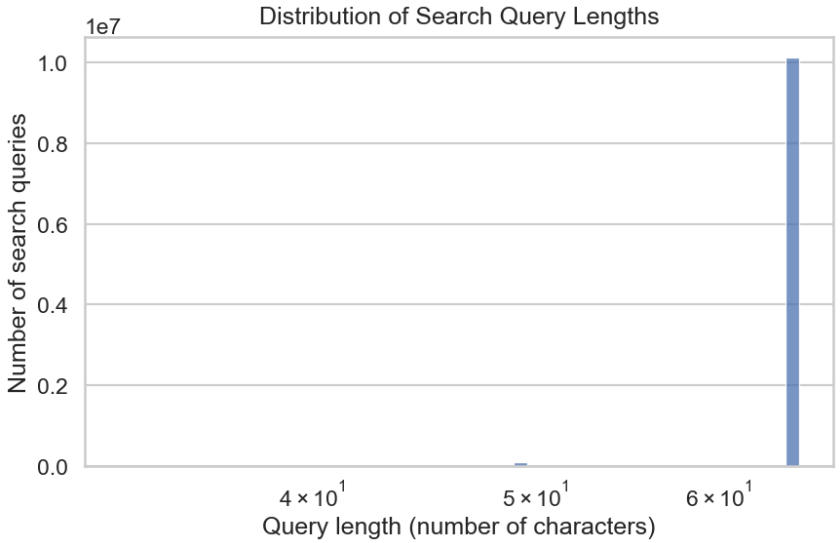

Figure 6: Search query length distribution. The near-degenerate mass at a single length indicates anonymized identifiers; query should be treated as a categorical node rather than natural language.

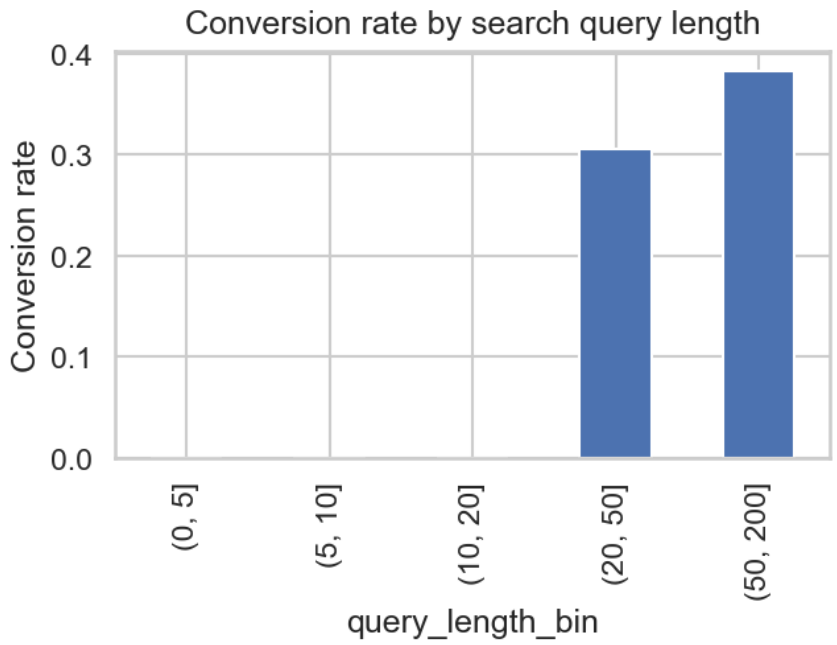

Figure 7: Conversion proxy by query-length bin. This is primarily a *user-type* effect (users who ever buy differ from users who never buy), and should not be interpreted as query "effectiveness" without session/time-window attribution.

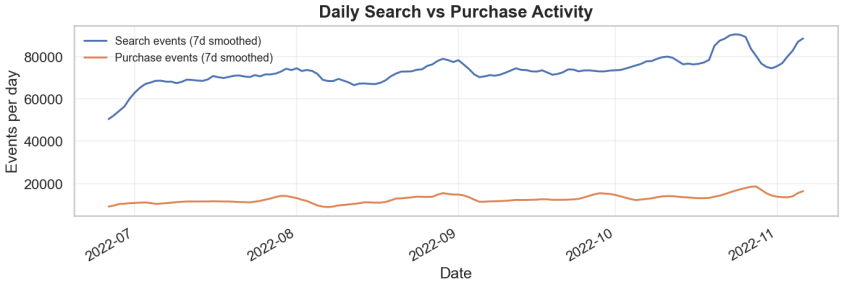

Figure 8: Daily search vs purchase activity (7-day smoothed). Volatility and sparsity in purchases imply drift and severe class imbalance for purchase-centric supervision, motivating recency-aware modeling and time-based splits.

One group adds to cart but rarely buys within the window, which may reflect comparison shopping, friction, or weak intent. Another group converts at moderate rates. The extreme tail likely includes repeat or fast buyers, bypass paths where buying is not preceded by add, and inflated counts from duplicates. This is a strong reason to treat remove-from-cart as an explicit negative feedback

signal rather than discarding it. It can provide hard negatives that purchases alone cannot.

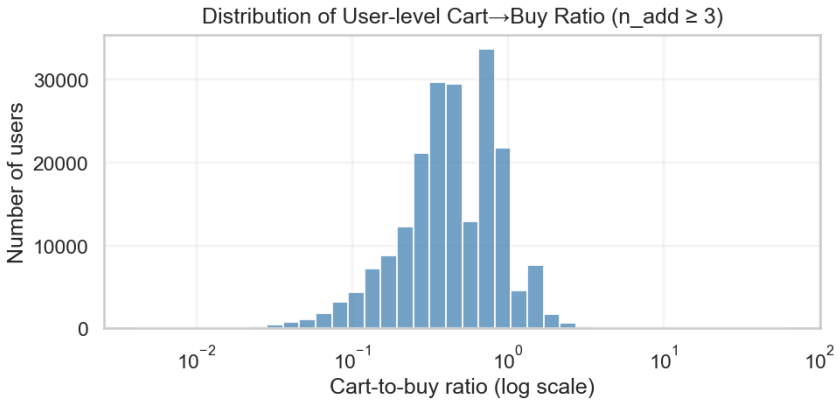

Figure 9: Cart-to-buy ratio distribution (users with `n_add` ≥ 3 shown in the notebook). The mass at 0 indicates many cart interactions do not lead to purchase within the window, while the heavy tail suggests repeat buyers, bypass paths, and/or duplicated purchase intensity.

Lightweight segmentation: clusters are a diagnostic lens, not ground truth. We use a simple KMeans segmentation on standardized user-level aggregates as a lightweight diagnostic baseline. The elbow curve in Figure 11 suggests that a small number of regimes already explains most variance, and with $k = 2$ the split often aligns with low-activity versus higher-activity users.

We visualize the same clustering with two projections that emphasize different structure. The PCA scatter in Figure 10 is a linear projection that preserves directions of maximum global variance. In our data the first principal component is dominated by overall activity volume, so most users spread along PC1 while a compact low-activity mass collapses near the left side. This makes the clusters look like a small wedge separated from a long heavy tail, but the apparent boundary is partly a projection effect.

The t-SNE map in Figure 12 is a nonlinear visualization that preserves local neighborhoods rather than global geometry. It typically forms compact blobs and can exaggerate gaps between groups, and the axes are not directly interpretable. The spiral-like shapes and cluster separation are therefore expected artifacts of the embedding objective and hyperparameters, not evidence of truly discrete behavioral types.

In practice we do not treat either plot as a proof of separability. We use them to stress-test whether a single UBP is dominated by heavy users or collapses to the low-activity majority. Cluster-wise feature summaries and downstream metrics are more reliable than visual boundaries.

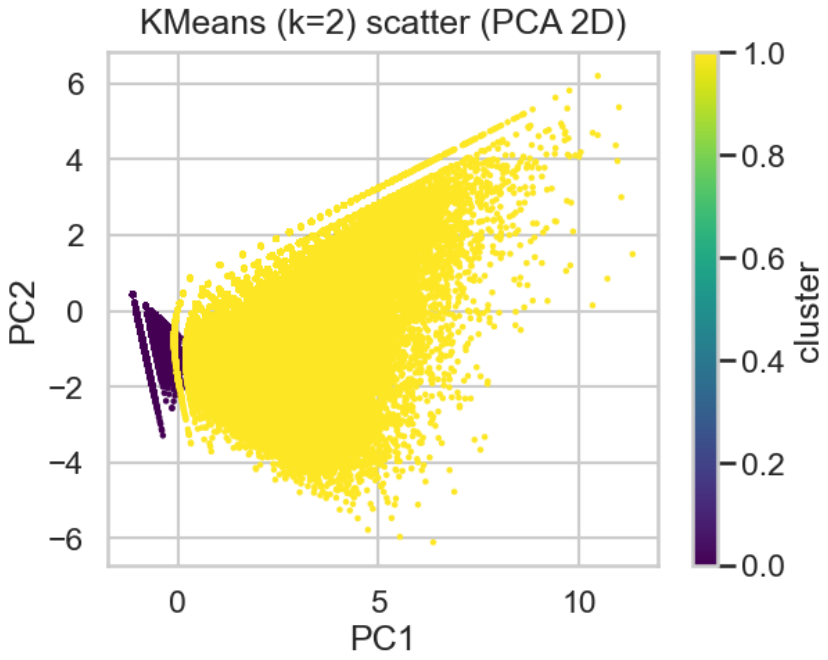

Figure 10: KMeans clusters visualized in a 2D PCA projection. PCA preserves global variance and is dominated by overall activity volume, so low-activity users collapse into a compact region while higher-activity users form a long heavy-tail along PC1.

Sequential structure: without care, the model learns repetition and loop behavior. The transition matrix in Figure 13 shows strong self-transitions, such as search followed by search, and loops between cart and remove. This is typical browsing and cart editing. If we train a next-event objective naively, the model can learn a cheap rule that predicts repetition, especially when duplicates are present.

Two practical fixes are sessionization and event compression. Sessionization splits sequences by time gaps so the model learns within-session intent progression rather than mixing unrelated visits. Compression merges consecutive identical events while keeping multiplicity as an intensity feature, so repetition is not treated as new information but also is not discarded. The frequent 3-gram motifs in Table 4 confirm that repeated-action loops dominate, while patterns like search, search, cart still provide useful short-range signals for self-supervised learning beyond sparse purchase labels.

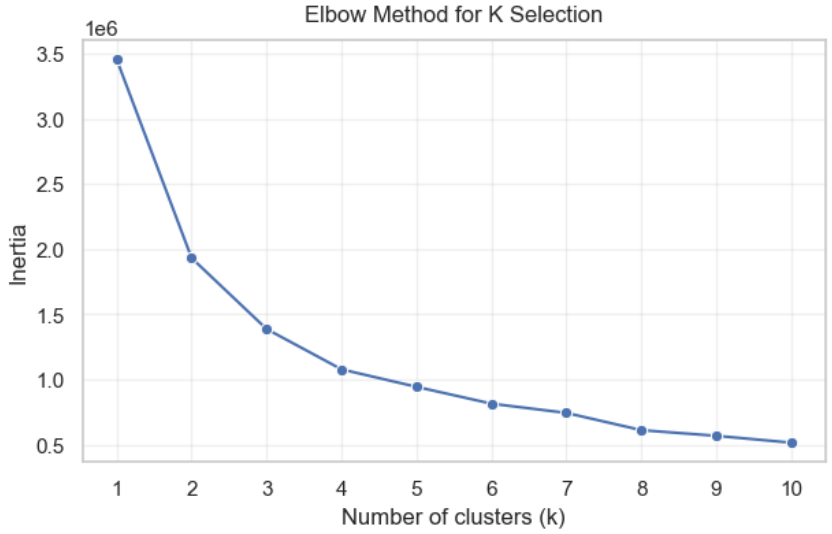

Figure 11: Elbow method for KMeans selection. A small k already captures major behavioral variance, consistent with a few dominant engagement regimes (e.g., low- vs high-activity).

Table 4: Most frequent length-3 event patterns extracted from user sequences. Counts are sliding-window occurrences, hence dominated by repeated-action loops.

Pattern (3-gram)	Count
(search, search, search)	6,286,894
(cart, cart, cart)	1,201,390
(search, search, cart)	807,015
(cart, search, search)	645,256
(remove, remove, remove)	497,435
(search, cart, search)	494,864
(cart, cart, remove)	485,482
(buy, buy, buy)	379,078
(search, cart, cart)	356,200
(cart, remove, remove)	296,420

4 Related Work

Prior work on the RecSys 2025 competition converges on a shared view of Universal Behavioral Profiles as reusable user representations learned from heterogeneous, time-stamped interaction logs . Although the downstream evaluation is fixed by the organizers, leading solutions consistently follow a modular pipeline: they first merge multiple event streams into a single chronological sequence, encode each event with its type and available attributes, and then train one or

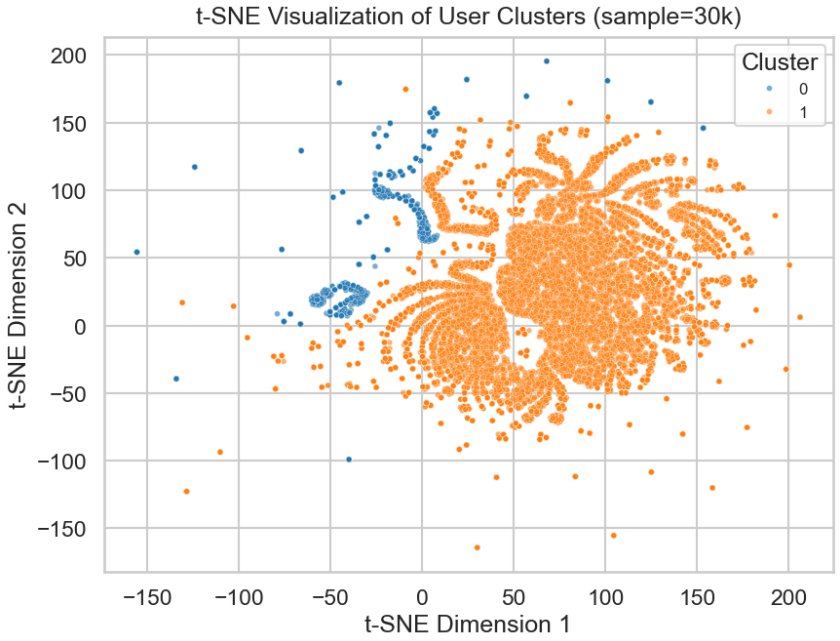

Figure 12: t-SNE visualization of user clusters (sample of 30k). Partial overlap suggests a continuous behavioral manifold; embeddings should generalize across regimes rather than specialize to one interaction style.

more representation learners on self-supervised or weakly supervised objectives. The resulting user vectors are typically augmented with collaborative signals and aggregated statistics, and finally fused into a single submission embedding .

A first family of approaches emphasizes temporal generalization through contrastive alignment between past and future behavior. The first place solution trains a two-tower Transformer and optimizes a contrastive objective so that embeddings computed from a user history before a split time are close to embeddings computed from the same user after the split time, while embeddings from different users are pushed apart [9, 1]. In addition to temporal features, this method incorporates collaborative structure through SVD-derived vectors and improves robustness by reweighting training examples so that non-relevant users provide diversity without overwhelming the objective [9]. The same work further combines a supervised multi-task encoder and an aggregated-feature embedding and uses stacking to integrate these heterogeneous views into a representation that remains strong on hidden tasks [9].

A second line of work treats user representation learning as task-agnostic compression via sequence reconstruction. The second place solution trains a GRU autoencoder to reconstruct multi-field events, including event type and key categorical attributes, using teacher forcing and per-field classification losses [5].

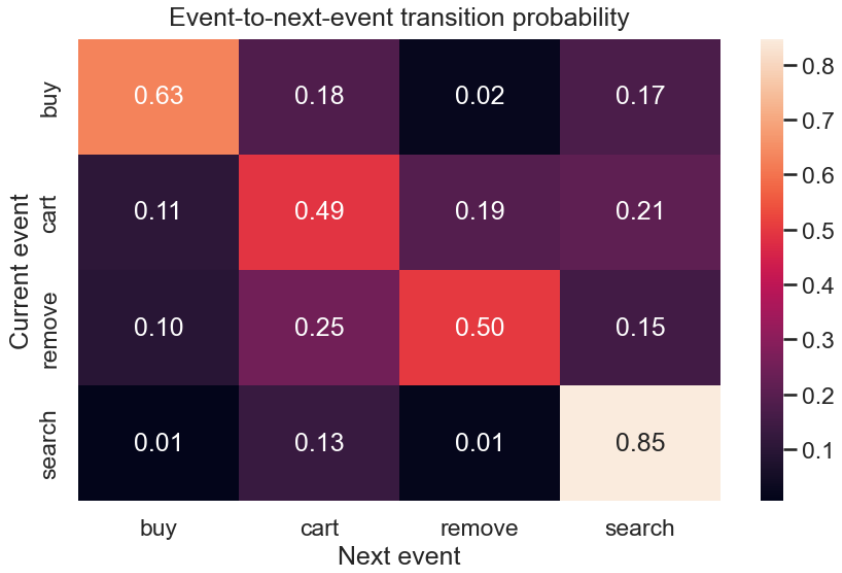

Figure 13: Event-to-next-event transition probabilities. Dominant self-transitions and cart–remove loops indicate strong within-stage iteration; sessionization and event compression help sequence models learn intent progression rather than repetition.

Multiple variants trade expressiveness for stability, which is especially valuable when hidden targets are sensitive to overfitting [5]. This approach is also notable for systematically enriching the reconstructed representation with complementary modules such as implicit-feedback matrix factorization, factorization-machine-style modeling, next-event Transformers, language-model-derived embeddings, and handcrafted statistics, followed by careful normalization and dimensionality reduction before concatenation [5, 4, 6].

Other competitive solutions adopt explicitly hybrid objective sets that combine reconstruction with direct prediction. The EmbedNBreakfast approach integrates sequence reconstruction, supervised multi-task learning with attention-based pooling for the open tasks, and future interaction prediction where a long historical window is used to predict the subsequent window [10]. It additionally incorporates Bayesian Personalized Ranking matrix factorization models at the product and category levels to strengthen propensity-related signals [10, 8]. The main advantage of such hybrids is that each objective encourages a different invariance, with reconstruction preserving broad history, supervised heads aligning to disclosed targets, and future prediction capturing short-term trends that may correlate with churn and conversion.

Finally, the fourth place solution highlights the benefit of combining strong sequential encoders with explicit relational structure and engineered aggregates. It trains a large causal Transformer on multiple self-supervised next-event ob-

jectives to learn a sequence embedding, complements it with a heterogeneous graph component based on TwHIN trained via link prediction on purchase and cart relations, and transfers the sequential item encoder to initialize the graph model [7, 3]. This work also emphasizes extensive feature engineering across multiple recency windows and fuses learned and engineered signals through a deep cross network that models higher-order feature interactions, regularized with contrastive augmentation to improve generalization [7, 11, 1]. Across these approaches, the most consistent pattern is that robust UBPs rarely come from a single objective or model class; rather, top systems combine complementary sequence learning, collaborative structure, and stable aggregate features to obtain representations that transfer across open and hidden tasks .

5 Data Preprocessing

We start from raw interaction logs then we produce user embeddings for Universal Behavioral Profiles. The core idea is to enrich events first, then normalize all signals into a stable numeric space, then learn a user representation from sequences. We keep a lightweight user level aggregation only as an internal diagnostic and as an ablation.

The input consists of product actions such as add to cart, buy, remove from cart, page visits, and search queries, plus a product table with category and price and anonymized embeddings for names. We always filter events to the provided relevant clients to ensure the embedding output matches the required client id set. After feature construction we write intermediate parquet tables for product events, search events, and visit sessions. These tables share a common philosophy. Every row is an event with an attached feature vector that mixes global context, user local context, and short range temporal context. The final user embedding is learned from the event sequences.

5.1 Scaling

Scaling is a model design decision in this dataset because the raw variables differ by orders of magnitude and the distributions are heavy tailed. Without careful scaling the representation collapses into a near one dimensional axis of activity volume. This hurts transfer across tasks because churn, propensity, and retrieval do not all depend on the same notion of volume.

We normalize features by type instead of applying a single scaler to the full table. Count like features are first compressed to reduce tail dominance and then mapped into a bounded range. In practice we apply a power style transform such as Yeo Johnson to counts and popularity statistics, then apply min max scaling into the interval from minus one to one. This improves two things at once. It makes the feature space comparable across event types, and it stabilizes sequence training because gradients do not explode on heavy users.

Ratio features such as conversion rates are treated separately. They are already bounded in principle, but the raw computation produces infinities and

missing values when denominators are zero. We replace infinities with a finite maximum and fill missing values with zero. After this cleaning step we min max scale ratios into minus one to one. This ensures that ratio signals remain visible next to counts. It also prevents the model from learning numerical artifacts.

Entropy features are also scaled separately. Entropy is conceptually a concentration signal, but in sparse regimes its empirical distribution can behave like a heavy tail. We therefore apply a log transform and then min max scale into minus one to one. This makes entropy act as a smooth preference diversity indicator rather than an outlier detector.

Embedding like vectors such as query centroids and name cluster centroids require explicit normalization. If we attach a centroid vector to every event without scaling, a few dimensions can dominate the event representation simply due to numeric range. We therefore compute the global minimum and maximum across all centroid values and map every dimension into minus one to one. This keeps centroid information in the same numeric scale as engineered scalar features. It also makes float16 export safe.

Finally we reserve minus one as a padding value when we create fixed length sequences for the GRU model. This only works if valid features rarely take the exact padding value. The bounded scaling makes this practical. Missing joins are filled with zero rather than minus one. Zero is treated as neutral evidence. Padding remains a separate concept that indicates there is no event at that position.

5.2 Outlier elimination

Outlier elimination is not limited to extreme numeric values. It also includes duplicated events, division by zero artifacts, and repeated action loops that can bias sequence objectives.

We handle numerical outliers before scaling. For heavy tailed count and popularity columns, a single extreme value can compress the entire distribution under min max scaling. We therefore clip columns at a high percentile before the final min max step. This preserves resolution among typical users while still keeping heavy users bounded. This choice matters because the goal is not to erase heavy users, but to prevent them from becoming the only learnable signal.

We handle ratio outliers by construction. Conversion rates can be infinite when a user never added but did buy, or when a sku has buys but no adds in the observed window. These cases are common because real journeys do not always follow a clean funnel. We replace infinities with a finite cap and fill NaNs with zero. The important interpretation is that extreme conversion values do not necessarily mean perfect conversion. They often mean missing pathways within the logged window. Treating them as bounded signals allows the model to use them as hints without letting them destabilize training.

We also reduce structural outliers in sequences. For visits and searches we collapse sessions so that long repeated loops do not dominate the event stream. Repetition is still preserved as intensity through derived counts and nearby event statistics, but we do not force the GRU to reconstruct thousands of nearly

identical events. This makes the sequential objective focus on transitions and local context rather than on trivial self transitions.

We treat duplicate rows as a risk. For aggregated statistics duplicates inflate counts and can create false confidence in conversion. For sequence learning duplicates create easy shortcuts where the next event is just the same event. When duplicates are exact duplicates we remove them before feature computation. When repetition is meaningful, such as repeated views in a session, we encode it as counts rather than as repeated tokens.

5.3 Feature engineering

Feature engineering follows one principle. We build features that separate intent, preference, and stability, while keeping the representation robust under sparsity and long tails. We do this by enriching each event with signals at multiple levels, then letting the sequence model integrate them over time.

On the product side we build a product events table by concatenating add, buy, and remove actions and joining product properties. For each sku and each category we compute global action counts, and we compute user local action counts for the same entities. This yields a clean separation between what is popular in the system and what is frequent for the user. We then derive conversion rates at multiple granularities. We compute sku level global conversion and user local conversion, and category level global conversion and user local conversion. The reason to keep both is that global conversion captures friction that is intrinsic to an item or category, while local conversion captures the user intent regime. Two users can add the same sku, but the meaning is different if one user typically converts adds into buys and the other rarely does. We also add top sku and top category indicators based on global purchase frequency. These features help represent head exposure, but they are treated carefully in evaluation because they can become a popularity shortcut.

A key addition is semantic clustering of anonymized embeddings. Product names and queries are provided as quantized embeddings rather than text. We cluster these embeddings with KMeans and assign a cluster id. This creates a stable intermediate semantic unit that is less sparse than raw ids. We then compute cluster level action counts and user local cluster counts, and we attach the cluster centroid vector to events. This has a practical benefit. Even if a specific sku is rare, its cluster membership provides a denser signal that generalizes across similar items. The centroid vector provides a continuous semantic hint that the GRU can combine with behavioral context.

We also compute entropy features as concentration signals. We compute entropy over sku, category, and semantic clusters, and we compute user level entropy counterparts. Entropy captures something that raw counts miss. It distinguishes focused users from exploratory users. This is important for universal profiles because downstream tasks interpret the same behavior differently. A focused regime can indicate repeat purchase propensity, while a broad regime can indicate browsing intent that is useful for retrieval.

On the search side we treat the query string as an identifier. We map unique

queries to query ids and parse the embedding representation. We cluster query embeddings and compute query and cluster scores at global and user local levels. We also compute entropy features for queries and query clusters. This yields a representation where search behavior is not just frequency, but also semantic direction and diversity.

On the visit side we build sessions by collapsing consecutive visit events and computing url counts and user url counts. We compute entropy over urls and user entropy for url choice. This captures whether the user repeatedly visits a narrow set of pages or spreads attention widely.

The most important cross channel feature is nearby product context. For both search sessions and visit sessions we attach summaries of nearby product events in time. This is implemented by counting product interactions that occur around the query or the visit window, with special emphasis on top skus and top categories. The reason is that the funnel is not single entry. Many users reach cart or purchase with little or no search, and many signals are only visible when we connect channels by time proximity. Nearby features create a lightweight bridge between search, visit, and product intent without building a full graph. They are especially helpful when query and url ids are opaque.

Knowledge mining via category graph. We also added a lightweight knowledge-mining component that captures cross-category structure directly from behavior. Using product metadata, we map SKUs to categories and mine a category–category co-occurrence graph from session-level baskets of add and buy events. We convert co-occurrence counts into lift-like association weights so edges reflect meaningful co-interest rather than raw popularity. For each user we build a weighted category-affinity vector from buy/add/remove signals and apply a graph diffusion step to propagate mass to neighboring categories. This yields a compact category-graph embedding that is stable under sparsity and unseen SKUs, and it is exported as an additional user-embedding block for later fusion.

After engineering we perform feature selection explicitly. For each event type we keep a predefined set of columns grouped by semantic type such as counts, conversion rates, and entropies. This avoids accidental leakage of raw identifiers and avoids unstable columns that appear only for rare cases. We then remove near zero variance columns and columns that are constant after filtering to relevant clients. This reduces noise and makes the embedding more stable across splits.

Method application is straightforward once the event tables are built. We unify schemas across product, search, and visit events, attach an event type indicator, attach simple time context such as hour and weekday, sort by timestamp within each client, and build fixed length sequences with padding. We train a GRU based representation learner that reconstructs or predicts event representations. The user embedding is taken from the hidden state or pooled latent. We export embeddings in float16 for efficiency and compliance with the submission format.

Comparison and interpretation are handled as controlled ablations inside the same pipeline. We measure what happens when we remove semantic clustering, when we remove nearby context, when we remove entropy features, and when we replace the GRU learner with simple pooling over event vectors. This tells us whether gains come from better features or from sequence modeling. In our analysis, semantic clustering improves generalization to tail entities because it provides a shared abstraction. Nearby context improves linkage between search and purchase because it encodes multi entry journeys. Entropy features improve robustness across user regimes because they model preference concentration rather than volume. Scaling and clipping are necessary for all of the above to be learnable, otherwise the model regresses to activity and popularity.

There are also failure modes that we track explicitly. Popularity based features and top indicators can inflate retrieval metrics by over recommending head items. Global statistics can leak information if computed on the full window while evaluating on a later slice. Min max scaling can lose resolution if outliers are not clipped. We address these by time aware computation of global statistics during validation, by clipping before min max, and by reporting metrics stratified by user activity and by head versus tail exposure. The goal is a universal profile that is not just strong on average, but stable across regimes and useful across tasks.

6 Methodology

We approached the Universal Behavioral Profiles task as an iterative model selection process under tight time constraints. The goal was not only to obtain a competitive embedding, but also to understand which signals are stable and transferable across objectives. We therefore started with simple baselines, moved to classical factorization models, tested a lightweight LLM-style embedding approach, and then converged on a sequence-based encoder inspired by the top leaderboard solutions. At each step we focused on two questions. What does this method capture well in this dataset, and what does it systematically miss.

6.1 Baseline evaluation

We first ran the official baseline without adding any custom features and used it purely as a reproduction check. Our goal at this stage was not to improve performance, but to verify that we could match the reference score reported in the task description and that our data loading, preprocessing assumptions, and evaluation pipeline were correct. The resulting performance was in the same range as the baseline, which confirmed that our environment and submission format were consistent with the organizers' setup.

This reproduction result was still informative for modeling. It highlighted that the baseline representation is intentionally simple and mostly captures broad activity and popularity effects. It offers strong speed and reproducibility

and provides an end-to-end submission path with minimal engineering. However, its expressiveness is limited. It does not explicitly encode temporal structure, and it does not fuse multi-channel behavior in a way that reflects real user journeys where visits and search can precede, follow, or bypass product actions. This observation motivated exploring methods that can represent user–item structure and sequential dynamics more directly.

6.2 Classical collaborative filtering baselines: ALS and factorization machines

We then tested matrix factorization style approaches because they are strong in implicit feedback settings and can be trained quickly at scale.

We considered ALS on a user–item interaction matrix, with variants that treat different actions as different weights. ALS is attractive because it produces compact user factors and item factors and typically performs well when the dominant signal is repeated preference for a subset of items. It is also easy to implement and tune. However, ALS has two structural weaknesses in this competition setting. First, it largely ignores event order and short-term intent. It can represent that a user likes a category, but not that a user shifted categories yesterday. Second, it struggles to integrate heterogeneous tokens such as URLs and query identifiers unless we construct separate matrices or add ad-hoc fusion, which quickly becomes brittle.

We also tested factorization machines via LightFM, including settings that combine interaction signals with side features. FMs are more flexible than ALS because they can incorporate additional categorical features such as category, price bucket, or action type. They can also be used for ranking losses that resemble retrieval objectives. The trade off is that their representational power still relies on pairwise interactions, and without careful feature design they revert to learning popularity. In this dataset, where many identifiers are anonymized embeddings and where user behavior is highly non-stationary, FMs provide a solid but limited improvement. They are good at global preference modeling, but weak at capturing the local temporal structure that distinguishes weak-intent browsing from deep-intent conversion.

Overall, ALS and LightFM served as strong classical baselines. Their strength was stability, training speed, and good performance on head items. Their weakness was that they cannot naturally represent multi-entry journeys and short-term dynamics, which are central to universal behavioral profiles.

6.3 Small LLM-style embedding approach

We also tested an LM-based embedding approach following the reference method. The idea is to use a small pre-trained language model as a frozen feature extractor after converting user interactions into text. Each user history is serialized with one event per line. For every event we record the event type and timestamp. For add-to-cart, remove-from-cart, and buy events we additionally include SKU, category, price, and the quantized item-name representation. For page visits we

include the encoded URL. For search we include the quantized query representation. Because context length is limited, we truncate to the most recent 64 interactions and embed the textualized sequence with SmolLM2-135M. We refer to this variant as SmolLM2 all interactions. We also implement a search-only variant, motivated by the greater homogeneity of search events. Here we keep only search events, use a compact tabular text format, and truncate to the most recent 90 events per user. We refer to this variant as SmolLM2 search queries.

This approach is appealing because it offers a single encoder that can fuse heterogeneous channels without hand-crafted cross features. It is also easy to extend when new event fields are added. The main drawback is that most fields are anonymized identifiers rather than natural language, so the model cannot rely on lexical meaning and often reduces to learning short-window co-occurrence. Truncation is another hard limit. Keeping only the most recent events over-emphasizes recency and can discard stable long-term preferences, especially for high-activity users. In practice, the LM embeddings were useful as a compact summary of recent behavior and as a baseline for fusion, but they were not consistently stronger than classical methods on their own under our tuning and compute constraints.

6.4 Knowledge-mining embedding block

In parallel, we evaluated a simple knowledge-mining baseline based on category structure. Instead of learning a sequence encoder, we mine a category co-occurrence graph from pre-training interactions and derive a user embedding by aggregating category affinities and diffusing them through the graph. This approach is cheap, interpretable, and naturally tail-friendly because it relies on category neighborhoods rather than SKU identity. We used it both as a standalone baseline and as a modular embedding block that can be concatenated with ALS/LightFM/LLM representations in the merge stage.

6.5 Incorporating ideas from top solutions

After establishing the behavior of these baselines, we studied the top-ranked public solutions and identified two recurring design principles.

First, strong solutions treat the task as sequence modeling rather than static profiling. They build event-level representations, align multiple event types into a common space, and then learn a user encoder that respects time order and recency.

Second, strong solutions build cross-channel bridges. They do not assume a single funnel. They explicitly connect search, visits, and product actions using semantic grouping and local temporal context, so the model can learn pathways that bypass search or bypass cart.

These principles directly address the weaknesses we observed. Classical factorization methods are good at stable preference but poor at short-term intent. The top solutions improve intent modeling by encoding sequences and by enriching events with contextual signals.

6.6 Final model: GRU-based sequence encoder over enriched events

We implemented a GRU-based sequence encoder inspired by the top two solutions and adapted it to our setting. The pipeline constructs enriched event vectors for products, URLs, and queries. For product events we include counts and conversion-like signals at multiple levels such as item, category, and semantic clusters. For query and visit events we add local context features such as nearby product actions and diversity signals such as entropy. We then normalize features into a stable numeric range, build per-user time-ordered sequences, and train a GRU encoder to produce a user embedding.

We chose a GRU rather than heavier transformer architectures for three reasons. It is efficient and stable on long sequences, it works well with continuous event vectors, and it is easier to tune under limited time. It also matches the empirical structure of the data, which contains many short loops and repeated actions. In such settings, a GRU can capture recency and short dependencies without overfitting to spurious long-range patterns.

The key advantage of the final approach is that it models both long-term preference and short-term intent. Long-term preference comes from repeated exposure to the same semantic clusters and categories. Short-term intent comes from recency, from local bursts, and from cross-channel nearby context. This produces embeddings that are more transferable across downstream tasks than purely collaborative factors or static aggregates.

The main disadvantage is complexity. It requires careful preprocessing, stable scaling, and thoughtful feature selection. It also introduces more hyperparameters and more failure modes, such as learning popularity shortcuts if head-item signals are too dominant. We addressed these risks by systematic tuning and ablations.

6.7 Hyperparameter selection and ablation analysis

We selected the final configuration through a lightweight hyperparameter search combined with ablation analysis. Hyperparameters included sequence length, GRU hidden size, pooling strategy, dropout, learning rate, and the weighting of different event types. We also tested variants of the feature set. We removed semantic clustering features, removed nearby context features, removed entropy features, and simplified conversion rates. This allowed us to identify which components produce consistent gains rather than occasional improvements.

The final model and feature set were chosen because they offered the best trade-off between performance, stability, and interpretability. The ablations showed that sequence encoding is necessary but not sufficient. The quality gains come from the combination of event enrichment and sequence modeling. Semantic grouping improves generalization to tail entities. Nearby context improves cross-channel alignment. Diversity signals improve robustness across user regimes. Together these components produce a universal profile that is less dominated by activity volume and more reflective of intent and preference.

7 Experimental Setup

All feature engineering and preprocessing were performed locally on an Apple MacBook Air with an M3 chip. This choice was sufficient for the data cleaning, parquet processing, clustering, and construction of event-level feature tables.

Model training and embedding generation were executed on an NVIDIA H100 GPU. We used the same train/validation protocol as provided by the challenge pipeline and reported the official metrics on the validation split. To keep results comparable across methods, we fixed the relevant client set, used consistent sequence truncation rules where applicable, and exported the final user embeddings in float16.

8 Results

Table 5: Comparison of validation metrics across all models and tasks. Each task corresponds to a different prediction objective: churn forecasting, category-level propensity modeling, and SKU-level propensity modeling. Metrics are taken from the final epoch (Epoch 2/2) logs.

Model	Task	Val AUROC	Val Diversity	Val Novelty
`Baseline`	Churn	0.694	–	–
`Baseline`	Category Propensity	0.657	0.907	0.851
`Baseline`	SKU Propensity	0.691	0.901	0.988
`LightFM`	Churn	0.623	–	–
`LightFM`	Category Propensity	0.714	0.976	0.669
`LightFM`	SKU Propensity	0.685	0.977	0.704
`ALS`	Churn	0.724	–	–
`ALS`	Category Propensity	0.747	0.543	0.868
`ALS`	SKU Propensity	0.724	0.622	0.990
`LLM`	Churn	0.688	–	–
`LLM`	Category Propensity	0.738	0.901	0.785
`LLM`	SKU Propensity	0.698	0.863	0.914
`Knowledge mining`	Churn	0.696	–	–
`Knowledge mining`	Category Propensity	0.741	0.886	0.937
`Knowledge mining`	SKU Propensity	0.711	0.881	0.996
`Our`	Churn	0.802	–	–
`Our`	Category Propensity	0.792	0.633	0.881
`Our`	SKU Propensity	0.761	0.555	0.988

8.1 Discussion and interpretation

The results in Table 5 show a consistent pattern across tasks. Methods that model user–item structure or sequence structure improve AUROC, but they trade off recommendation diversity and novelty in different ways.

For churn, the ranking is clear. Our model achieves the strongest AUROC at 0.802, followed by ALS at 0.724, then the baseline at 0.694 and the LLM at 0.688, while LightFM performs worst at 0.623. This ordering matches what we expect from the supervision signal. Churn is a user-level outcome where recency, stability, and evolving engagement matter. The baseline has some of this information implicitly through aggregate activity, which explains why it is already reasonably strong. ALS improves further because latent factors capture consistent preference intensity and engagement patterns that correlate with churn risk. Our method adds explicit temporal modeling and multi-channel sequence fusion, which likely captures the early signs of churn such as changes in browsing cadence, reduced cross-channel transitions, and shorter or less diverse sessions. The gap between ALS and our model suggests that churn benefits from temporal structure beyond static preference.

For category propensity, all models cluster in a relatively high AUROC regime, from 0.657 to 0.792. Our method again ranks first at 0.792, with ALS (0.747), LLM (0.738), LightFM (0.714), and the baseline (0.657) behind it. The reason category-level propensity is "easier" than SKU-level is that categories are coarse, dense targets. Even simple representations can learn that a user repeatedly interacts with a category and therefore is likely to convert within it. The improvements from ALS and our method indicate that there is still useful structure beyond raw counts. ALS benefits from capturing shared latent preference across categories. Our method likely gains from modeling short-term intent and session context, for example a recent burst in category exploration or cart-edit loops tied to a category, which can predict near-future conversion more reliably than long-term averages.

For SKU propensity, the table highlights why fine-grained prediction is structurally harder. The baseline has 0.694 AUROC, the three collaborative/sequence methods achieve meaningful discrimination: ALS at 0.724, LLM at 0.698, LightFM at 0.685, and our method at 0.761. This gap is a strong signal that item-level propensity requires either a user–item factorization view or a sequence encoder that can represent item identities and short-range dependencies. Aggregates that work for churn or category-level tasks do not preserve enough item-specific information. Our method's advantage over ALS suggests that sequence context matters even at the item level. ALS captures stable preferences, but a sequence encoder can also capture "what the user is currently trying to buy," which is often a recency-driven signal.

Interpreting the knowledge mining results. The knowledge-mining embedding built from the category graph is a strong example of a non-neural, structure-driven signal. It achieves solid AUROC on category and SKU propensity while maintaining high novelty and strong diversity, indicating that it is not

simply ranking the popularity head. Its limited gain on churn reinforces that churn depends more on temporal engagement patterns than on static preference neighborhoods. Practically, this makes the category-graph block a good fusion candidate: it contributes tail-aware structure for propensity tasks and improves beyond-accuracy behavior, while sequence models remain responsible for capturing recency and churn-related dynamics.

Overall , the best AUROC comes from models that concentrate on intent-relevant items. Our method achieves the best AUROC on all three tasks, indicating that multi-channel sequence encoding yields the most transferable universal profile.

8.2 Disclaimer

Due to the organizers closing the official submission channel during our project window, we could not upload a final submission to the public leaderboard. We contacted the organizers to request access or an alternative validation route, but we did not receive a response by the deadline. Therefore, all reported results are computed on the official local validation protocol and should be interpreted as controlled offline comparisons rather than confirmed leaderboard standings.

9 Conclusion

In this project we selected the RecSys Challenge 2025 dataset on Universal Behavioral Profiles, a large-scale multi-channel e-commerce log with heterogeneous user actions (search, page visits, add-to-cart, remove-from-cart, purchases) and anonymized semantic representations for queries and item names. The dataset is realistic in two key ways: it is extremely sparse at the purchase level and heavily long-tailed at both user and item level, and it contains multiple valid user journeys rather than a single funnel. These properties make it a good testbed for universal user embeddings that must generalize across objectives.

Our exploratory data analysis confirmed steep funnel drop-off and strong class imbalance for purchase-centric supervision, heavy-tailed per-user activity, long-tail SKU popularity, near-degenerate query length distributions consistent with anonymized identifiers, and volatile temporal dynamics that suggest drift and the need for recency-aware modeling. We also observed loop-heavy sequential patterns such as search-to-search and cart-editing cycles, which motivated sessionization and sequence compression instead of naively treating logs as independent events.

Based on these findings, we implemented a single end-to-end preprocessing and representation pipeline. Feature engineering was performed at the event level and focused on signals that are stable under sparsity. We added multi-granular statistics for product events (global and user-local counts at SKU and category level), conversion-like ratios capturing add-to-buy behavior, entropy

features measuring preference concentration and diversity, and lightweight semantic abstraction by clustering anonymized query and name embeddings with KMeans to obtain cluster identities and centroids. We further linked channels using time-local context by adding "nearby" product-event summaries to search and visit sessions, which helps represent multi-entry journeys where users bypass search or cart. All engineered features were scaled into a consistent numeric range to stabilize sequence learning and float16 export, with explicit handling of division-by-zero artifacts and long-tail outliers.

We evaluated multiple modeling families and used them to diagnose what is learnable from the data. The official baseline was run unchanged to reproduce the reference score and validate our training and evaluation setup. Classical collaborative filtering models (ALS) and factorization machines (LightFM) provided strong structure-aware baselines for propensity tasks, while an LM-based feature extractor (SmolLM2) offered a compact summary of recent multi-channel activity but was limited by anonymized identifiers and truncation. Our final model, a GRU-based sequence encoder over the enriched event streams, consistently achieved the best discrimination on the local validation protocol. In particular, our model reached Val AUROC of 0.802 on churn, 0.792 on category-level propensity, and 0.761 on SKU-level propensity, outperforming ALS, LightFM, the baseline, and the LLM embeddings in AUROC.

The main strengths of our approach are that it encodes heterogeneous channels in a unified sequential representation, captures both long-term preference and short-term intent, and makes anonymized embeddings usable through semantic clustering and centroid features. The main weaknesses are increased pipeline complexity and sensitivity to preprocessing decisions. Popularity and head-item signals can still become shortcuts, scaling can lose resolution if extreme outliers are not clipped carefully, and time-agnostic global statistics can create optimistic evaluation if not recomputed strictly within the training window. In addition, due to the organizers closing the submission channel during our project window, we could not validate our embeddings on the official leaderboard. We contacted the organizers but did not receive a response, so all reported results should be interpreted as offline validation comparisons rather than confirmed leaderboard standings.

There are several clear directions for future improvements. A stronger temporal protocol is needed, including time-based splits and time-conditioned features to reduce drift sensitivity. Sequence modeling could benefit from better sessionization and event compression that preserve intensity while reducing trivial self-transitions. The "nearby event" linkage can be upgraded into a lightweight temporal graph and trained with contrastive objectives to better align queries, URLs, and SKUs. Finally, diversity-aware regularization and debiased negative sampling can reduce popularity shortcuts and improve tail personalization, which is crucial for universal profiles that must transfer across downstream objectives.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 2020.

[2] Jacek Dabrowski, Maria Janicka, Łukasz Sienkiewicz, Gergely Stomfai, Dietmar Jannach, Francesco Barile, Marco Polignano, Claudio Pomo, and Abhishek Srivastava. The SYNERISE dataset: An E-Commerce dataset for sequential recommendation, universal behavior modeling and deep relational learning. In *RecSys Challenge 2025 (RecSysChallenge '25)*, Prague, Czech Republic, September 2025. ACM.

[3] Ahmed El-Kishky, Thomas Markovich, Serim Park, Chetan Verma, Baekjin Kim, Ramy Eskander, Yury Malkov, Frank Portman, Sofía Samaniego, Ying Xiao, et al. Twhin: Embedding the twitter heterogeneous information network for personalized recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*, pages 2842–2850. ACM, 2022.

[4] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining (ICDM)*, pages 263–272. IEEE, 2008.

[5] Anton Klenitskiy, Artem Fatkulin, Daria Denisova, Anton Pembek, and Alexey Vasilev. Encode me if you can: Learning universal user representations via event sequence autoencoding. In *RecSys Challenge 2025 (RecSysChallenge '25)*, Prague, Czech Republic, September 2025. ACM. arXiv:2508.07748.

[6] Maciej Kula. Metadata embeddings for user and item cold-start recommendations, 2015.

[7] Sergei Makeev, Alexandr Andreev, Vladimir Baikalov, Vladislav Tytskiy, Aleksei Krasilnikov, and Kirill Khrylchenko. Blending sequential embeddings, graphs, and engineered features: 4th place solution in recsys challenge 2025. In *RecSys Challenge 2025 (RecSysChallenge '25)*, Prague, Czech Republic, September 2025. ACM. arXiv:2508.06970.

[8] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09)*, pages 452–461, 2009.

[9] Yuki Sawada, Rintaro Hasegawa, Yuhi Nagatsuma, Shugo Takei, Kazuhito Yonekawa, and Hiromu Auchi. Toward universal user representations: Contrastive learning with transformers and embedding ensembles. In *RecSys*

Challenge 2025 (RecSysChallenge '25), Prague, Czech Republic, September 2025. ACM.

[10] EmbedNBreakfast Team. Acm recsys challenge 2025: Final model description (team solution summary), 2025. Unpublished team report / solution description (as provided in the competition write-up).

[11] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. DCN V2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of The Web Conference 2021 (WWW '21)*, pages 1785–1797. ACM, 2021.