FLYSCORE: FLY-INSPIRED SPARSE COHORT RETRIEVAL FOR NEXT-BASKET RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Next-basket recommendation ranks the products that a household is likely to purchase on its next shopping occasion. The task combines recurring patterns in the household's own history with purchase evidence shared across households. We introduce Fly-inspired Sparse Cohort Retrieval (FLYSCORE), a retrieval-based ranking model inspired by the sparse associative organization of the Drosophila mushroom body. For each history–candidate pair, a history-conditioned user signal summarizes purchasing behavior across time scales, while a sparse address selects memory locations containing weighted purchase and opportunity evidence from earlier observations. A support-aware readout combines the retrieved evidence with the user signal to rank the full product catalog. We instantiate this design in three ways. A central nervous system model (CNS) couples learned task interfaces to fixed measured connectivity; a probabilistic model (PROB) expresses memory selection through choice probabilities; and a neural model (NET) uses learned activation strengths. Across four repeat-purchase datasets, the highest-scoring implementation exceeds the strongest of eleven baselines in NDCG@20 by 4.81%–33.60%. We further construct an alternative memory interface from retained input activity and composed connections. Rebuilding historical memory at this interface preserves 97.0%–99.3% of complete CNS NDCG@20. These results support sparse cohort retrieval across measured, probabilistic, and neural implementations.

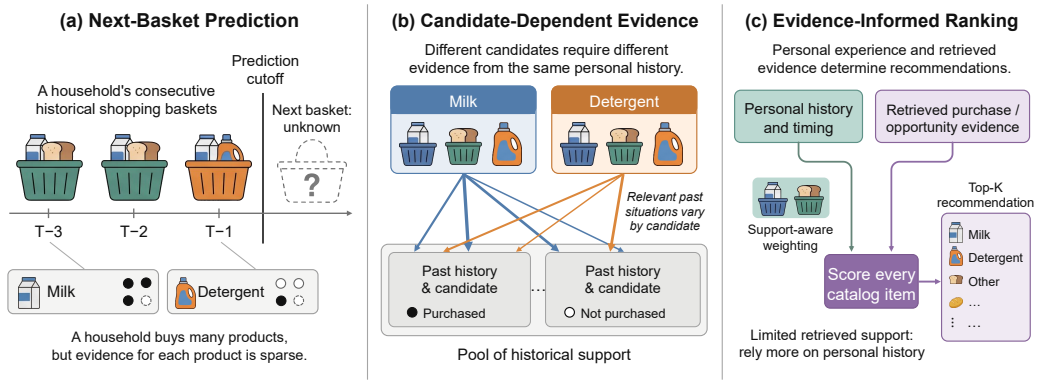
1 INTRODUCTION

Fruit flies learn to associate sensory cues, such as food-related odors, with rewarding or aversive outcomes, allowing stored experience to guide later approach and avoidance (Aso et al., 2014b). This ability motivates a question for recommendation: can a circuit that links current cues to past outcomes help predict a user's preferences among products? We study this question through next-basket recommendation (NBR), focusing on personalized recommendations for recurring consumer needs. The transferable insight is associative retrieval: the current situation selects relevant experience, which then informs a choice.

In NBR, the current situation comprises a household's shopping history and a candidate product. Individual purchase records may be sparse even for an established customer, and useful evidence from other households can depend on the candidate (Li et al., 2023; Ariannezhad et al., 2022). Milk and detergent, for example, can reflect different replenishment needs. Existing methods use personal frequency, temporally weighted neighbors, learned representations, and basket dependencies (Hu et al., 2020; Wan et al., 2018; Yu et al., 2016; Hu & He, 2019; Yu et al., 2020; 2023; Deng et al., 2026). Our question is how a fly-inspired memory mechanism can select evidence for each history–candidate pair and use that evidence to rank the complete product catalog.

The mushroom body supplies a concrete starting point. Its organization distinguishes input encoding, sparse intermediate activation, and an output stage that accesses learned associations (Aso et al., 2014a; Dasgupta et al., 2017). These roles suggest basic computational units for representing a history–candidate pair, selecting relevant experience, and producing a preference score. We introduce Fly-inspired Sparse Cohort Retrieval (FlySCoRe). Its central nervous system (CNS) connectivity branch uses learned input projections to adapt recommendation features to the circuit; measured connections organize activation and access to task-specific outcome memory. Throughout, fly-inspired denotes the transfer of this functional organization to a computational retrieval model. Normalized

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Which past situations should inform each candidate?

Figure 1: Motivation for FlySCoRe. (a) Sparse product-level evidence in next-basket prediction. (b) Candidate-dependent retrieval of past situations. (c) Ranking combines retrieved evidence with personal history and timing.

synaptic contact counts supply a fixed anatomical prior for the network weights (Berg et al., 2026); recommendation experience enters through the adapted inputs and outcome memory. With an adapted readout, CNS exceeds the strongest normalized discounted cumulative gain (NDCG) baseline at rank 20 on three of the four evaluated populations.

This transfer raises a deeper question: how much depends on the particular measured weights, and how much on the memory and retrieval functions that the circuit organizes? An odor activates only a small subset of the mushroom body's intermediate cells; output connections associate that activity with experience (Aso et al., 2014a; Dasgupta et al., 2017). We interpret this selectivity as a possible choice among memory locations. Different inputs can therefore retrieve evidence specific to the current candidate. Our hypothesis is that the useful transfer involves a selective memory mechanism that can also be expressed probabilistically. Structure and information provide complementary descriptions of this process. Connectivity and connection weights organize activation, while selective memory access determines which stored evidence informs the score.

We investigate this hypothesis through two abstractions of the CNS design. A probabilistic retrieval model (PROB) makes selection explicit through probability weights over sparse memory locations. A neural network retrieval model (NET) uses a trainable neural expansion and positive activation strengths to implement a simplified circuit organization. Each combines retrieved purchase and opportunity evidence with a KC-processed user signal. Figure 1 illustrates this common retrieval function. Both abstractions learn their representations and retain explicit outcome memory. Their comparison with CNS examines whether selective access to experience remains useful across different implementations.

We evaluate the three implementations on four repeat-purchase datasets. Under a common full-catalog protocol, CNS leads on Complete Journey, PROB on Ta-Feng, and NET on Dunnhumby-Carbo and Instacart in NDCG@20. On each dataset, the highest-scoring branch exceeds the strongest of eleven baselines by 4.81%–33.60%. These results show that the retrieval function can be effective with both measured connectivity and learned abstractions within the history-eligible query populations.

We further test whether this function can be reconstructed when its original implementation is replaced. Bypassing sparse addressing and its user signal sharply reduces performance. An engineered replacement uses the simulated model's retained input-layer activity and composed connection weights to address outcome memory directly. We rewrite memory from historical support observations at this interface. The reconstructed model reaches 97.0%–99.3% of complete CNS NDCG@20. This computational compensation uses structural information to rebuild memory access at a different layer, preserving task performance through an engineered information path.

The contributions are threefold:

- We transfer the fly's selective associative organization to NBR through adapted input and readout interfaces, with measured connectivity providing fixed weights for candidate-specific memory access.
- We derive probabilistic and neural abstractions of that retrieval function, making the relationship between connectivity, selective access, and stored purchase evidence explicit and comparing their ranking behavior against eleven baselines on four datasets.
- We construct and evaluate an alternative memory path from retained model activity and quantify how much ranking performance this computational compensation preserves under a change of memory interface.

2 Problem Definition

Each NBR query consists of a household's shopping history at a prediction cutoff. We focus on established households with evidence of recurring purchasing in the history observed by that cutoff; this history alone determines query eligibility. For each eligible query, the model ranks every product in the training catalog, and the household's next basket supplies the prediction target. Because a query is defined at a cutoff, one household can contribute several queries over time. Section 4.1 describes the resulting recurring-demand setting.

Past history–candidate pairs with known next-basket outcomes provide the experience available for retrieval. We call these pairs support observations: each records the information preceding a shopping occasion and whether the candidate was purchased on that occasion. Inputs combine shopping history, product attributes, calendar context, and training-only product statistics; unavailable fields are masked. The modeling question is which of these observations should inform the score of a new candidate.

3 Methodology

FlySCoRe uses the current history and candidate to retrieve experience that adjusts a KC-processed user signal. Input Encoding (A) represents the history–candidate pair; Sparse Addressing (K) selects memory locations and supplies the user signal; Memory and Readout (M) stores purchase and opportunity counts and combines retrieved evidence with that signal. We first instantiate these functions in CNS, then derive PROB and NET.

3.1 From Biological Organization to Retrieval Functions

Antennal-lobe projection neurons (ALPNs) relay sensory activity to the mushroom body, where Kenyon cells (KCs) form a sparse representation and mushroom-body output neurons (MBONs) access learned associations through KC connections (Aso et al., 2014a). Fly-inspired hashing illustrates how expansion and sparse activation support similarity-based access (Dasgupta et al., 2017). CNS adapts history and product features to this circuit and assigns purchase outcomes to its memory. Task-specific input projections are learned, while normalized measured connections remain fixed (Berg et al., 2026). Figure 2 maps the biological organization to the three retrieval functions and their abstractions.

3.2 CNS: Connectome-Constrained Retrieval

For a household considering milk, CNS uses the history–milk pair as a cue to access earlier shopping experience through measured ALPN–KC–MBON connections. The same process scores every candidate to produce a ranking.

Input Encoding (A): Representing the Shopping Situation. Learned projections place history and product features in a common space. CNS averages these representations and applies $\tanh$ to obtain input activity. Because both representations enter the cue, a candidate is evaluated in the context of the household's current shopping history. Learning these projections adapts recommendation features to the fixed circuit.

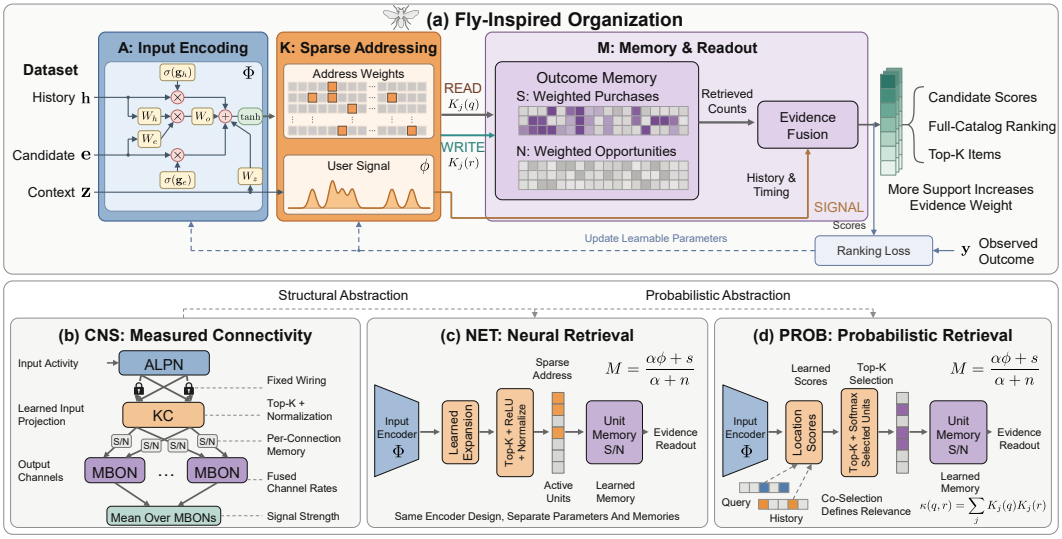


Figure 2: FlySCoRe overview. (a) Input encoding, sparse addressing, outcome-memory readout, and training feedback. (b–d) CNS, NET, and PROB instantiate retrieval through measured connectivity, learned expansion, and probabilistic co-selection, respectively.

Sparse Addressing (K): Selecting Relevant Experience. Measured ALPN–KC connections expand this cue; the strongest positive responses, normalized by their sum, form its sparse address. Address overlap determines which historical pairs contribute and with what weight. This retrieval cohort can include other households and products, and changes with the candidate even for a fixed history.

KC-Processed User Signal (K): Retaining Household Identity. Alongside the address, K supplies a candidate-specific user signal. A history-conditioned mixture of constant and decaying weights combines long-term purchasing identity with recent behavior. Training product frequency stabilizes sparse records, and household-level rescaling makes candidate scores comparable. This signal anchors prediction when retrieved evidence is limited.

Outcome Memory (M): Recording Purchases and Opportunities. Historical pairs follow the same encoding and addressing path. Their selected KC activity is multiplied by outgoing KC–MBON weights and normalized over connections. Each connection stores an opportunity total and a purchase total: the former records every evaluated candidate pair, and the latter records its contribution when the candidate appears in the following basket. Contributions also include a fixed historical-prefix weight. Thus nonpurchases provide evidence, and opportunity counts retain support strength as well as purchase rates. One purchase among two opportunities and ten among twenty have the same rate, but the latter supplies more evidence to the support-aware readout.

Evidence Readout (M): Turning Experience into a Score. The current query reads these totals through the same connections. Each MBON combines its retrieved purchase rate with the user signal: limited support leaves the score close to the signal, while larger opportunity totals give retrieved experience more influence. CNS averages the resulting channel scores. Measured connectivity therefore governs both memory storage and access.

Formally, let q be the current history–candidate pair, $\mathbf{h}$ and $\mathbf{e}$ its projected representations, and $\phi(\mathbf{z})$ the user signal computed from purchase and timing evidence in $\mathbf{z}$. The input and KC activities are $\mathbf{A}$ and $\mathbf{v}$. Let C_{AK} be the measured input-to-KC matrix and C_{mj} the nonnegative normalized weight from KC j to MBON m, with B MBONs. The set $\mathrm{Top}_k(\mathbf{v})$ contains the k highest-scoring KCs; j and d index KCs, and $[t]^+ = \max(t, 0)$. Address weight $K_j(q)$ and connection access weight $K_j(q)C_{mj}/Z(q)$ use $Z(q) = \sum_j K_j(q) \sum_{m=1}^{B} |C_{mj}|$. With stored purchase and opportunity counts

S_{mj}, N_{mj} and signal strength $\alpha > 0$, CNS computes

$$\begin{aligned} \mathbf{A} &= \tanh\left[\tfrac{1}{2}(\mathbf{h}+\mathbf{e})\right], \qquad \mathbf{v} = C_{AK}^{\mathsf{T}}\mathbf{A}, \\ K_j(q) &= \frac{\mathbf{1}\{j \in \mathrm{Top}_k(\mathbf{v})\}[v_j]^+}{\sum_{d\in\mathrm{Top}_k(\mathbf{v})}[v_d]^+}, \\ M &= \frac{1}{B}\sum_{m=1}^{B}\frac{(\alpha/B)\phi(\mathbf{z}) + \sum_j (K_j(q)C_{mj}/Z(q))S_{mj}}{\alpha/B + \sum_j (K_j(q)C_{mj}/Z(q))N_{mj}}. \end{aligned} \tag{1}$$

Historical observations use the same connection access rule to write memory. Each channel receives α/B pseudo-observations at rate $\phi(\mathbf{z})$. Zero denominators give zero address or access weights; unsupported channels return the user signal. Activity is continuous-valued.

3.3 Probabilistic and Neural Abstractions

CNS connects a structural description, selective activity routed through connections, to an information-level description, weighted access to stored outcomes. PROB makes that access an explicit choice model; NET learns a neural expansion whose activation strengths control access. Both preserve candidate-dependent addressing and signal-anchored readout, while replacing connection-specific memory and multiple output channels with one pair of counts per memory unit and one readout.

Shared Memory Interface. For historical prefix–candidate pair r, let y_r indicate purchase in the following basket and $w_r \geq 0$ be its fixed prefix weight. Within each branch, $K_j(q)$ and $K_j(r)$ use the same addressing rule for the query and historical pair. The index j now denotes a learned memory unit. Writing and reading through these addresses gives

$$S_j = \sum_r w_r K_j(r) y_r, \qquad N_j = \sum_r w_r K_j(r). \tag{2}$$

$$M = \frac{\alpha\phi(\mathbf{z}) + \sum_j K_j(q) S_j}{\alpha + \sum_j K_j(q) N_j}. \tag{3}$$

Eligible support excludes a training query's own prefix. The query–support overlap is $\kappa(q,r) = \sum_j K_j(q)K_j(r)$. Both branches use a joint encoder $\Phi(\mathbf{h},\mathbf{e},\mathbf{z})$ combining history and product representations with frequency, timing, and history–product similarity evidence. They share its architecture but have separate fitted parameters and outcome memories.

3.3.1 PROB: Probabilistic Retrieval

PROB models access as a choice among memory locations. A learned linear function of the joint encoded features produces scores $\mathbf{v}$; a multinomial-logit rule assigns probabilities within the selected set. For independent choices from query q and historical pair r, $K_j(q)K_j(r)$ is the probability of both choosing unit j. Summing across mutually exclusive shared locations gives $\kappa(q,r)$, their *co-selection probability*. This is the pair's relevance weight in the retrieval cohort; disjoint addresses give zero weight. A history–milk pair receives more weight when its cue favors the query's memory locations.

Expanding the stored counts in Eq. (3) exposes these observation-level contributions:

$$\begin{aligned} K_j(q) &= \Pr(j \mid \mathbf{h},\mathbf{e},\mathbf{z}) = \frac{\mathbf{1}\{j \in \mathrm{Top}_k(\mathbf{v})\}\exp(v_j)}{\sum_{d\in\mathrm{Top}_k(\mathbf{v})}\exp(v_d)}, \\ M &= \frac{\alpha\phi(\mathbf{z}) + \sum_r w_r y_r \kappa(q,r)}{\alpha + \sum_r w_r \kappa(q,r)}. \end{aligned} \tag{4}$$

Thus M is a signal-anchored weighted mean of binary purchase outcomes. The numerator combines weighted purchases with $\alpha\phi(\mathbf{z})$ pseudo-purchase mass, and the denominator combines weighted opportunities with α pseudo-opportunity mass. Because co-selection probabilities apply to separate observations, their weighted sum measures support and need not equal one. Equations (2)–(3) compute the same score by pre-aggregating contributions at memory units. With a shared address, all candidates retrieve identical counts and retain the user signal's ordering. Candidate-dependent addresses allow historical evidence to change that ordering.

3.3.2 NET: NEURAL RETRIEVAL

NET retains CNS's expansion and selective positive activation while learning expansion connections from recommendation data. With expansion matrix W and bias $\mathbf{b}$, it computes

$$\mathbf{A} = \Phi(\mathbf{h}, \mathbf{e}, \mathbf{z}), \qquad \mathbf{v} = W\mathbf{A} + \mathbf{b},$$
$$K_j(q) = \frac{\mathbf{1}\{j \in \mathrm{Top}_k(\mathbf{v})\}[v_j]^+}{\sum_{d \in \mathrm{Top}_k(\mathbf{v})}[v_d]^+}. \tag{5}$$

The address is zero when no selected activation is positive; readout follows Eq. (3). PROB allocates access by exponential choice weights, whereas NET preserves relative positive activation strengths. Together with CNS, they test the same retrieval organization across measured connectivity, probabilistic selection, and learned neural activity. This link also motivates reconstruction: retained activity and connections can define a new interface for writing and reading historical outcomes.

3.4 MODEL TRAINING

The reported stage adapts a learned initialization to ranking using a LambdaRank-style pairwise logistic loss. Purchased candidates are favored over unpurchased candidates, with each pair weighted by the absolute NDCG change from swapping its predicted positions, averaged over evaluation cutoffs (Burges et al., 2006). PROB and NET update their joint feature mixer and address map; CNS updates history and product projections into ALPN activity. Earlier training supplies the feature encoders, user and product representations, and user signal. These feature representations and the user signal, as well as measured connections, remain fixed.

Memory pools candidate pairs from training histories. Queries read only support with an earlier recorded outcome time, excluding their own outcome and later outcomes from the same household. Memory is rebuilt each epoch and held fixed while gradients update query-side addressing. Validation selects among the initialization and subsequent checkpoints; the selected parameters rebuild memory for final ranking. Appendix B specifies compact CNS computation, encoders, the user signal, and the training protocol.

4 EXPERIMENTS

We first evaluate CNS for recommendation, then compare it with PROB and NET to examine retrieval across implementations. Module interventions test how performance changes when parts of CNS are bypassed. Finally, retained activity and connections support reconstruction of memory access at another interface.

4.1 DATASETS

We use Complete Journey (CJ), Ta-Feng (TF), Dunnhumby-Carbo (DC), and Instacart (IC), totaling 871,672 transaction records in 142,236 baskets. Table 1 reports the observed histories and target baskets; a user may contribute multiple training targets. Missing input fields are masked.

We evaluate repeat-purchase recommendation for established households. Query eligibility is determined solely from the household purchase history observed at the prediction cutoff, using the same history-based rule for every method. All methods rank the complete catalog. Household experience can still provide sparse product-level evidence: at training cutoffs, observed history–product pairs have a median purchase count of one on every dataset, and 56.2%–84.8% appear only once. Candidate-specific retrieval pools this evidence across households. Appendix C documents preparation and input features; Appendix D.2 examines variation across the eligible households' history profiles.

4.2 BASELINES

We compare with eleven baselines in three families. Frequency and neighborhood methods directly use purchase records. Global popularity (GPOP) ranks products by training purchase frequency. GP-TopFreq (Li et al., 2023) ranks by household purchase frequency and fills remaining positions

Table 1: Dataset statistics for the history-eligible query populations. History totals count each transaction row and basket once across the pre-query histories. Target baskets are the next baskets to predict; a user may supply multiple training targets. Medians count preceding baskets per target.

Dataset	# Products	Observed history		# Target baskets			# Median baskets		
		Transaction rows	Baskets	Train	Valid.	Test	Train	Valid.	Test
CJ	8,443	138,871	21,246	159	25	32	70	103	126
TF	4,857	7,968	1,907	84	23	68	8	8	9
DC	129	1,461	970	170	156	153	2	2	2
IC	4,487	723,372	118,113	5,686	1,314	1,151	11	11	13

with globally popular products. TIFU-KNN (Hu et al., 2020) combines temporally weighted personal frequency with neighboring shopping histories.

Latent representation methods learn user–product relations. BPR-MF (Rendle et al., 2009) learns user and product factors through pairwise ranking. Triple2Vec (Wan et al., 2018) learns transaction-triple embeddings, with a validation-selected adaLoyal adjustment. LightGCN (He et al., 2020) propagates representations over the interaction graph.

Sequential and temporal-set models capture basket dependencies. DREAM (Yu et al., 2016) models basket sequences recurrently; Sets2Sets (Hu & He, 2019) models temporal sets with repeat information. DNNTSP (Yu et al., 2020) combines element relationships, temporal attention, and shared-pattern fusion. SFCNTSP (Yu et al., 2023) learns set, temporal, and embedding-channel relationships. TIDE (Deng et al., 2026) combines habitual and exploratory signals with time intervals. Appendix C.3 describes their mechanisms; Appendix C.4 provides their configurations. All methods use identical queries, catalogs, and metrics. Validation NDCG@20 selects configurations and checkpoints, with Recall@20 breaking ties.

4.3 Evaluation Metrics

Recall is the fraction of each next basket recovered in the top-K list. NDCG discounts each purchased product by its rank and normalizes by the ideal ordering with purchased products first, distinguishing ranking quality from basket coverage. We report both at $K \in \{10, 20\}$, average equally over queries, and give values to four decimal places. Each query contributes a full-catalog ranking. Appendix E.1 gives the formulas.

4.4 Recommendation Performance

Connectome-Based Retrieval. Table 2 reports Recall and NDCG at ranks 10 and 20. CNS exceeds the strongest NDCG@20 baseline on CJ, DC, and IC; on TF it reaches 0.4326 against the baseline's 0.4602. Thus, learned task interfaces and historical outcome memory allow a circuit with fixed measured connections to support competitive product ranking. The differences across datasets motivate examining which functions remain useful under other implementations.

Probabilistic and Neural Retrieval. PROB leads TF, NET leads DC and IC, and CNS leads CJ in NDCG@20. The highest-scoring branch on each dataset exceeds its strongest baseline by 4.81%–33.60%; PROB and NET each exceed that baseline on three datasets. These comparisons support selective outcome retrieval beyond the measured CNS connections. Each branch fits its own representation and retains explicit outcome memory, so the comparison concerns complete implementations of this functional design. On DC, several methods tie in Recall, while NET's higher NDCG reflects improved ordering of the recovered products.

Recommendation Length. Figure 3 compares all methods at cutoffs from 5 to 40. PROB's advantage on TF persists across these lengths; on DC, basket coverage saturates early while NDCG distinguishes product ordering.

Module Ablations. The final three rows of Table 2 bypass A, K, or M, with remaining parameters fixed. The K bypass removes sparse selection and the user signal together, retaining dense positive

Table 2: Recall and NDCG comparison on four datasets. The last three rows show CNS module bypasses (without, or w/o) with frozen parent parameters. Four decimals are shown. Bold and underlining mark the best and second-best complete methods before rounding; ties share marks.

Method	Complete Journey				Ta-Feng				Dunnhumby-Carbo				Instacart			
	Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
GPOP	0.0260	0.0521	0.0169	0.0254	0.1397	0.7426	0.0780	0.2252	0.0327	0.1928	0.0173	0.0634	0.0293	0.0368	0.0257	0.0283
GP-TopFreq	0.5109	0.7083	0.2999	0.3624	0.5294	0.7059	0.3217	0.3682	**1.0000**	**1.0000**	0.8152	0.8152	0.6704	0.8153	0.4918	0.5398
TIFU-KNN	0.4375	0.5703	0.2110	0.2529	0.6838	0.8309	0.4067	0.4465	0.9935	**1.0000**	0.8117	0.8135	0.7294	0.8605	0.5429	0.5863
BPR-MF	0.0583	0.1219	0.0281	0.0488	0.4338	0.6250	0.2303	0.2802	**1.0000**	**1.0000**	0.7366	0.7366	0.1899	0.2839	0.1233	0.1525
Triple2Vec	0.5016	0.7109	0.2776	0.3433	0.7426	0.8603	0.4274	0.4602	**1.0000**	**1.0000**	0.8283	0.8283	0.5596	0.7553	0.3278	0.3931
LightGCN	0.0365	0.0365	0.0198	0.0198	0.0956	0.7426	0.0509	0.2295	0.8333	0.9673	0.5054	0.5423	0.0300	0.0386	0.0259	0.0290
DREAM	0.1823	0.3958	0.0702	0.1333	0.7353	0.7353	0.3233	0.3233	0.0294	0.1830	0.0153	0.0582	0.0290	0.0377	0.0256	0.0285
Sets2Sets	0.1786	0.1891	0.1186	0.1227	0.4485	0.7426	0.2595	0.3377	**1.0000**	**1.0000**	0.7769	0.7769	0.6490	0.7851	0.4886	0.5333
DNNTSP	0.2995	0.4245	0.1718	0.2104	0.4412	0.7206	0.2373	0.3107	0.9869	0.9902	0.7541	0.7551	0.6941	0.8314	0.4995	0.5453
SFCNTSP	0.0365	0.1026	0.0260	0.0470	0.0809	0.7353	0.0483	0.2239	0.9673	0.9935	0.6431	0.6504	0.5306	0.7415	0.3531	0.4226
TIDE	0.1911	0.2250	0.1406	0.1531	0.3603	0.4853	0.1634	0.1940	0.7026	0.8235	0.5674	0.6009	0.1291	0.1575	0.1232	0.1326
FlySCoRe-PROB	**0.7005**	0.8568	**0.4321**	0.4800	**0.8309**	**0.9706**	**0.5122**	**0.5495**	**1.0000**	**1.0000**	0.8242	0.8242	**0.7445**	0.8682	0.5706	0.6123
FlySCoRe-NET	0.6953	0.7995	0.4139	0.4454	0.6029	0.8309	0.3641	0.4257	**1.0000**	**1.0000**	**0.8681**	**0.8681**	0.7439	**0.8789**	**0.5947**	**0.6401**
FlySCoRe-CNS	0.6693	**0.8620**	0.4257	**0.4841**	0.6324	0.8603	0.3724	0.4326	**1.0000**	**1.0000**	0.8435	0.8435	0.7436	0.8762	0.5909	0.6357
w/o Encoding	0.6484	0.8542	0.4028	0.4645	0.5588	0.8235	0.3295	0.4004	1.0000	1.0000	0.8381	0.8381	0.7467	0.8741	0.5877	0.6310
w/o Addressing	0.0000	0.0000	0.0000	0.0000	0.0368	0.0368	0.0155	0.0155	0.1144	0.3497	0.0496	0.1096	0.0044	0.0052	0.0033	0.0037
w/o Readout	0.6797	0.8620	0.4149	0.4689	0.5956	0.8309	0.3648	0.4273	1.0000	1.0000	0.8238	0.8238	0.7469	0.8754	0.5872	0.6306

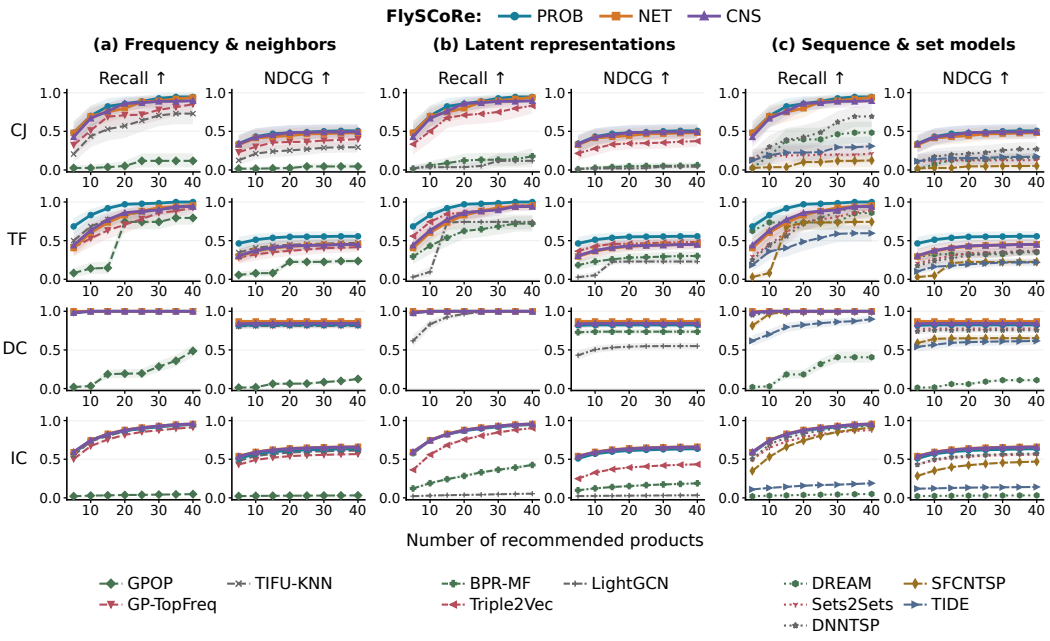


Figure 3: Recommendation accuracy as list length increases. Rows show datasets; column pairs compare FlySCoRe with three baseline families, covering all eleven baselines. Solid curves show PROB, NET, and CNS; dashed and dotted curves show baselines. Higher values are better. Shading gives pointwise 95% intervals from 2,000 paired household bootstrap samples with fixed models.

activation and outcome-memory retrieval. CNS NDCG@20 falls from 0.8435 to 0.1096 on DC and from 0.6357 to 0.0037 on IC; this intervention therefore measures their joint contribution. Appendix E.4 specifies each bypass. The model's retained input activity and connections allow a separate test of whether memory access can be reconstructed at another interface.

4.5 Retrieval Reconstruction from Structural Information

We compose the measured ALPN–KC and KC–MBON connections to relate input activity directly to output channels. Retained input activity is rescaled to nonnegative values and normalized by the composed outgoing connections. These input coordinates replace KC coordinates as memory locations in both writing and retrieval. We then rebuild memory from historical support, keeping each purchase outcome but changing the locations to which it contributes. Input projections and measured connections remain fixed, while the construction supplies a new address rule for historical evidence.

Table 3 compares this construction with random input projections, retrained dense addressing, the unchanged readout bypass, a validation-selected constant fusion weight, and a trained residual readout. Each column is normalized by the strongest complete FlySCoRe branch; the CNS-relative percentages below instead measure recovery of the original circuit. Appendix E.5 specifies the controls and reports raw scores.

Table 3: Relative performance under module replacement (%). Each column uses the highest score among the three complete FlySCoRe branches as 100. The second row uses the strongest benchmark in that column. Remaining rows are CNS module variants, grouped by the function being replaced.

Implementation	Complete Journey				Ta-Feng				Dunnhumby-Carbo				Instacart			
	Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
Best FlySCoRe	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Best benchmark	72.9	82.5	69.4	74.9	89.4	88.6	83.4	83.8	100.0	100.0	95.4	95.4	98.0	97.9	91.3	91.6
Input encoding replacements (A)																
Random encoding (A)	97.8	99.1	91.6	92.9	69.0	83.3	66.2	73.0	100.0	100.0	96.8	96.8	100.6	99.9	98.2	98.0
Memory addressing replacements (K)																
ALPN-addressed memory (K)	97.0	100.0	96.2	97.0	71.7	85.6	71.7	78.2	100.0	100.0	94.7	94.7	100.3	99.8	98.7	98.5
Retrained dense addressing (K)	101.1	100.0	98.5	97.7	72.6	84.1	69.9	75.6	100.0	100.0	96.9	96.9	100.4	99.7	98.9	98.6
Evidence readout variants (M)																
Readout bypass (M)	97.0	100.0	96.0	96.9	71.7	85.6	71.2	77.8	100.0	100.0	94.9	94.9	100.3	99.6	98.7	98.5
Calibrated constant fusion (M)	97.8	97.3	96.8	96.5	71.7	85.6	71.2	77.8	100.0	100.0	97.6	97.6	100.7	100.4	99.4	99.3
Trained residual readout (M)	97.0	100.0	96.0	96.9	71.7	85.6	71.2	77.8	100.0	100.0	95.4	95.4	100.8	100.3	99.7	99.5

ALPN-addressed memory reaches NDCG@20 of 0.4696, 0.4296, 0.8222, and 0.6307 on CJ, TF, DC, and IC, retaining 97.0%, 99.3%, 97.5%, and 99.2% of CNS. The reconstructed model therefore preserves most of the original CNS ranking performance while moving memory access to the input interface. The composed connections define a constructible alternative path for writing and retrieving historical outcomes.

5 Conclusion

FlySCoRe transfers the fly's selective access to learned associations to next-basket recommendation. CNS adapts shopping inputs and outcome memory to measured fly connectivity, while PROB and NET retain these retrieval functions in probabilistic and neural abstractions. Their different strengths across datasets show that selective access to experience remains useful across implementations. On the evaluated repeat-purchase populations, the highest-scoring branch on each dataset exceeds the strongest of eleven NDCG@20 baselines by 4.81%–33.60%.

Retained input activity and composed connections also support a new memory interface. Rebuilding historical memory there preserves 97.0%–99.3% of complete CNS NDCG@20. This construction shows how structural information can organize a new route to historical outcomes at a different memory interface.

This separation makes the source of each score explicit: the user signal summarizes the household's own history, whereas the retrieval path contributes evidence from other households. The present experiments establish this design for grocery next-basket recommendation; transfer to other recommendation domains and online systems remains for future work.

Artificial Intelligence Use Statement

Generative AI tools assisted with polishing writing and code. The authors are responsible for the final text, code, and claims.

Ethics Statement

The study uses de-identified transaction data and publicly available connectome-derived connectivity for computational recommendation experiments. The biological data supply network structure, and the transaction records supply the purchase outcomes used for ranking.

Reproducibility Statement

All datasets and the fly connectome data used in this study are publicly available for download. Training hyperparameters, supplementary metrics, and additional experimental results are provided in the appendix. We will release the trained model weights and complete source code.

References

Mozhdeh Ariannezhad, Sami Jullien, Ming Li, Min Fang, Sebastian Schelter, and Maarten de Rijke. ReCANet: A repeat consumption-aware neural network for next basket recommendation in grocery shopping. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1240–1250, 2022. doi: 10.1145/3477495.3531708. URL https://doi.org/10.1145/3477495.3531708.

Yoshinori Aso, Daisuke Hattori, Yang Yu, Rebecca M Johnston, Nirmala A Iyer, Teri-TB Ngo, Heather Dionne, LF Abbott, Richard Axel, Hiromu Tanimoto, and Gerald M Rubin. The neuronal architecture of the mushroom body provides a logic for associative learning. *eLife*, 3:e04577, 2014a. doi: 10.7554/elife.04577. URL https://doi.org/10.7554/elife.04577.

Yoshinori Aso, Divya Sitaraman, Toshiharu Ichinose, Karla R Kaun, Katrin Vogt, Ghislain Belliart-Guérin, Pierre-Yves Plaçais, Alice A Robie, Nobuhiro Yamagata, Christopher Schnaitmann, William J Rowell, Rebecca M Johnston, Teri-T B Ngo, Nan Chen, Wyatt Korff, Michael N Nitabach, Ulrike Heberlein, Thomas Preat, Kristin M Branson, Hiromu Tanimoto, and Gerald M Rubin. Mushroom body output neurons encode valence and guide memory-based action selection in Drosophila. *eLife*, 3:e04580, 2014b. doi: 10.7554/elife.04580. URL https://doi.org/10.7554/elife.04580.

Ting Bai, Jian-Yun Nie, Wayne Xin Zhao, Yutao Zhu, Pan Du, and Ji-Rong Wen. An attribute-aware neural attentive model for next basket recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1201–1204, 2018. doi: 10.1145/3209978.3210129. URL https://doi.org/10.1145/3209978.3210129.

Stuart Berg, Isabella R. Beckett, Marta Costa, Philipp Schlegel, Michał Januszewski, Elizabeth C. Marin, Aljoscha Nern, Stephan Preibisch, Wei Qiu, Shin-ya Takemura, Alexandra M.C. Fragniere, Andrew S. Champion, Diane-Yayra Adjavon, Michael Cook, Marina Gkantia, Kenneth J. Hayworth, Gary B. Huang, William T. Katz, Florian Kämpf, Zhiyuan Lu, Christopher Ordish, Tyler Paterson, Tomke Stürner, Eric T. Trautman, Catherine R. Whittle, Laura E. Burnett, Judith Hoeller, Feng Li, Frank Loesche, Billy J. Morris, Tobias Pietzsch, Markus W. Pleijzier, Valeria Silva, Yijie Yin, Iris Ali, Griffin Badalamente, Alexander Shakeel Bates, Rory J. Beresford, John Bogovic, Paul Brooks, Sebastian Cachero, Brandon S. Canino, Bhumpanya Chaisrisawatsuk, Jody Clements, Arthur Crowe, Inês de Haan Vicente, Georgia Dempsey, Erika Donà, Márcia Dos Santos, Marisa Dreher, Christopher R. Dunne, Katharina Eichler, Samantha Finley-May, Miriam A. Flynn, Imran Hameed, Gary Patrick Hopkins, Philip M. Hubbard, Ladann Kiassat, Julie Kovalyak, Shirley A. Lauchie, Meghan Leonard, Alanna Lohff, Kit D. Longden, Charli A. Maldonado, Ilina Moitra, Sung Soo Moon, Caroline Mooney, Eva J. Munnelly, Nneoma Okeoma, Donald J. Olbris, Anika Pai, Birava Patel, Emily M. Phillips, Stephen M. Plaza, Alana Richards, Jennifer Rivas Salinas, Ruairí J.V. Roberts, Edward M. Rogers, Ashley L. Scott, Louis A. Scuderi, Pavithraa Seenivasan, Laia Serratosa Capdevila, Claire Smith, Rob Svirskas, Satoko Takemura, Ibrahim Tastekin, Alexander

Thomson, Lowell Umayam, John J. Walsh, Holly Whittome, C. Shan Xu, Emily A. Yakal, Tansy Yang, Arthur Zhao, Reed George, Viren Jain, Vivek Jayaraman, Wyatt Korff, Geoffrey W. Meissner, Sandro Romani, Jan Funke, Christopher Knecht, Stephan Saalfeld, Louis K. Scheffer, Scott Waddell, Gwyneth M. Card, Carlos Ribeiro, Michael B. Reiser, Harald F. Hess, Gerald M. Rubin, and Gregory S.X.E. Jefferis. Sexual dimorphism in the complete Drosophila male central nervous system connectome. *Cell*, 189(18):5504–5526.e15, 2026. doi: 10.1016/j.cell.2026.08.015. URL https://doi.org/10.1016/j.cell.2026.08.015.

Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with nonsmooth cost functions. In B. Schölkopf, J. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/af44c4c56f385c43f2529f9b1b018f6a-Paper.pdf.

Yanan Cao, Ashish Ranjan, Sinduja Subramaniam, Evren Korpeoglu, Kaushiki Nag, and Kannan Achan. CASE: Cadence-aware set encoding for large-scale next basket repurchase recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 4480–4484, 2026. doi: 10.1145/3805712.3808504. URL https://doi.org/10.1145/3805712.3808504.

Sophie J. C. Caron, Vanessa Ruta, L. F. Abbott, and Richard Axel. Random convergence of olfactory inputs in the Drosophila mushroom body. *Nature*, 497(7447):113–117, 2013. doi: 10.1038/nature12063. URL https://doi.org/10.1038/nature12063.

Sanjoy Dasgupta, Charles F. Stevens, and Saket Navlakha. A neural algorithm for a fundamental computing problem. *Science*, 358(6364):793–796, 2017. doi: 10.1126/science.aam9868. URL https://doi.org/10.1126/science.aam9868.

Zhiying Deng, Yuan Fu, Usman Farooq, Ziwei Tian, Wei Liu, and Jianjun Li. Time-interval-aware disentangled expert modeling for next-basket recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 268–279, 2026. doi: 10.1145/3805712.3809615. URL https://doi.org/10.1145/3805712.3809615.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 639–648, 2020. doi: 10.1145/3397271.3401063. URL https://doi.org/10.1145/3397271.3401063.

Toshihide Hige, Yoshinori Aso, Gerald M. Rubin, and Glenn C. Turner. Plasticity-driven individualization of olfactory coding in mushroom body output neurons. *Nature*, 526(7572):258–262, 2015. doi: 10.1038/nature15396. URL https://doi.org/10.1038/nature15396.

Haoji Hu and Xiangnan He. Sets2Sets: Learning from sequential sets with neural networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1491–1499, 2019. doi: 10.1145/3292500.3330979. URL https://doi.org/10.1145/3292500.3330979.

Haoji Hu, Xiangnan He, Jinyang Gao, and Zhi-Li Zhang. Modeling personalized item frequency information for next-basket recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1071–1080, 2020. doi: 10.1145/3397271.3401066. URL https://doi.org/10.1145/3397271.3401066.

Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Hadamard product for low-rank bilinear pooling. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/pdf?id=r1rhWnZkg.

Janne K. Lappalainen, Fabian D. Tschopp, Sridhama Prakhya, Mason McGill, Aljoscha Nern, Kazunori Shinomiya, Shin-ya Takemura, Eyal Gruntman, Jakob H. Macke, and Srinivas C. Turaga. Connectome-constrained networks predict neural activity across the fly visual system. *Nature*, 634 (8036):1132–1140, 2024. doi: 10.1038/s41586-024-07939-3. URL https://doi.org/10.1038/s41586-024-07939-3.

Duc-Trong Le, Hady W. Lauw, and Yuan Fang. Correlation-sensitive next-basket recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 2808–2814, 2019. doi: 10.24963/ijcai.2019/389. URL https://doi.org/10.24963/ijcai.2019/389.

Feng Li, Jack W Lindsey, Elizabeth C Marin, Nils Otto, Marisa Dreher, Georgia Dempsey, Ildiko Stark, Alexander S Bates, Markus William Pleijzier, Philipp Schlegel, Aljoscha Nern, Shin-ya Takemura, Nils Eckstein, Tansy Yang, Audrey Francis, Amalia Braun, Ruchi Parekh, Marta Costa, Louis K Scheffer, Yoshinori Aso, Gregory SXE Jefferis, Larry F Abbott, Ashok Litwin-Kumar, Scott Waddell, and Gerald M Rubin. The connectome of the adult Drosophila mushroom body provides insights into function. *eLife*, 9:e62576, 2020. doi: 10.7554/elife.62576. URL https://doi.org/10.7554/elife.62576.

Ming Li, Sami Jullien, Mozhdeh Ariannezhad, and Maarten de Rijke. A next basket recommendation reality check. *ACM Transactions on Information Systems*, 41(4):1–29, 2023. doi: 10.1145/3587153. URL https://doi.org/10.1145/3587153.

Ming Li, Yuanna Liu, Sami Jullien, Mozhdeh Ariannezhad, Andrew Yates, Mohammad Aliannejadi, and Maarten de Rijke. Are we really achieving better beyond-accuracy performance in next basket recommendation? In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 924–934, 2024. doi: 10.1145/3626772.3657835. URL https://doi.org/10.1145/3626772.3657835.

Andrew C Lin, Alexei M Bygrave, Alix de Calignon, Tzumin Lee, and Gero Miesenböck. Sparse, decorrelated odor coding in the mushroom body enhances learned odor discrimination. *Nature Neuroscience*, 17(4):559–568, 2014. doi: 10.1038/nn.3660. URL https://doi.org/10.1038/nn.3660.

Ashok Litwin-Kumar, Kameron Decker Harris, Richard Axel, Haim Sompolinsky, and L.F. Abbott. Optimal degrees of synaptic connectivity. *Neuron*, 93(5):1153–1164.e7, 2017. doi: 10.1016/j.neuron.2017.01.030. URL https://doi.org/10.1016/j.neuron.2017.01.030.

Bo Peng, Zhiyun Ren, Srinivasan Parthasarathy, and Xia Ning. m^2: Mixed models with preferences, popularities and transitions for next-basket recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):4033–4046, 2023. doi: 10.1109/tkde.2022.3142773. URL https://doi.org/10.1109/tkde.2022.3142773.

Yuqi Qin, Pengfei Wang, and Chenliang Li. The world is binary: Contrastive learning for denoising next basket recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 859–868, 2021. doi: 10.1145/3404835.3462836. URL https://doi.org/10.1145/3404835.3462836.

Ashish Ranjan, Ayush Agarwal, Shalin Barot, and Sushant Kumar. Scalable permutation-aware modeling for temporal set prediction. *arXiv preprint arXiv:2504.17140*, 2025. doi: 10.48550/arXiv.2504.17140. URL https://arxiv.org/abs/2504.17140.

Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. RepeatNet: A repeat aware neural recommendation machine for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4806–4813, 2019. doi: 10.1609/aaai.v33i01.33014806. URL https://doi.org/10.1609/aaai.v33i01.33014806.

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*, pp. 452–461, 2009. URL https://hilpub.uni-hildesheim.de/entities/publication/9f2935bc-ea87-42f7-bd08-ea662702da67.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pp. 811–820, 2010. doi: 10.1145/1772690.1772773. URL https://doi.org/10.1145/1772690.1772773.

Jaiyam Sharma and Saket Navlakha. Improving similarity search with high-dimensional locality-sensitive hashing. *arXiv preprint arXiv:1812.01844*, 2018. URL https://arxiv.org/abs/1812.01844.

Philip K. Shiu, Gabriella R. Sterne, Nico Spiller, Romain Franconville, Andrea Sandoval, Joie Zhou, Neha Simha, Chan Hyuk Kang, Seongbong Yu, Jinseop S. Kim, Sven Dorkenwald, Arie Matsliah, Philipp Schlegel, Szi-chieh Yu, Claire E. McKellar, Amy Sterling, Marta Costa, Katharina Eichler, Alexander Shakeel Bates, Nils Eckstein, Jan Funke, Gregory S. X. E. Jefferis, Mala Murthy, Salil S. Bidaye, Stefanie Hampel, Andrew M. Seeds, and Kristin Scott. A Drosophila computational brain model reveals sensorimotor processing. *Nature*, 634(8032):210–219, 2024. doi: 10.1038/s41586-024-07763-9. URL https://doi.org/10.1038/s41586-024-07763-9.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/8fb21ee7a2207526da55a679f0332de2-Abstract.html.

Mengting Wan, Di Wang, Jie Liu, Paul N. Bennett, and Julian McAuley. Representing and recommending shopping baskets with complementarity, compatibility and loyalty. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1133–1142, 2018. doi: 10.1145/3269206.3271786. URL https://doi.org/10.1145/3269206.3271786.

Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for next basket recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 403–412, 2015. doi: 10.1145/2766462.2767694. URL https://doi.org/10.1145/2766462.2767694.

Shoujin Wang, Liang Hu, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Longbing Cao. Intention2Basket: A neural intention-driven approach for dynamic next-basket planning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 2333–2339, 2020. doi: 10.24963/ijcai.2020/323. URL https://doi.org/10.24963/ijcai.2020/323.

Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 729–732, 2016. doi: 10.1145/2911451.2914683. URL https://doi.org/10.1145/2911451.2914683.

Le Yu, Leilei Sun, Bowen Du, Chuanren Liu, Hui Xiong, and Weifeng Lv. Predicting temporal sets with deep neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1083–1091, 2020. doi: 10.1145/3394486.3403152. URL https://doi.org/10.1145/3394486.3403152.

Le Yu, Zihang Liu, Tongyu Zhu, Leilei Sun, Bowen Du, and Weifeng Lv. Predicting temporal sets with simplified fully connected networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4835–4844, 2023. doi: 10.1609/aaai.v37i4.25609. URL https://doi.org/10.1609/aaai.v37i4.25609.

A Related Work

Learning from Basket Sequences. Next-basket models must represent both persistent preferences and changes across shopping occasions. Factorized personalized Markov chains combine user preferences with item transitions (Rendle et al., 2010), while hierarchical representations and recurrent models learn how users and successive baskets interact (Wang et al., 2015; Yu et al., 2016). Within-basket structure supplies additional information: product complementarity motivates transaction embeddings (Wan et al., 2018), and item correlations or attributes can guide basket encoding (Le et al., 2019; Bai et al., 2018). Temporal-set models extend this representation of basket contents and their evolution through attention, graph-based relationships, or permutation-preserving transformations (Hu & He, 2019; Yu et al., 2020; 2023; Ranjan et al., 2025). These methods develop representations of sets and their evolution. FlySCoRe connects such history and product information to an explicit outcome memory: the representation determines which historical observations contribute to each candidate's score.

Repeat Purchases and Relevant History. Recurring purchases provide a strong signal in grocery recommendation. Personalized frequency and temporal neighbors exploit this signal directly (Hu et al., 2020), while ReCANet models repeat consumption explicitly (Ariannezhad et al., 2022). RepeatNet separates repeat and explore choices in the related task of session-based recommendation (Ren et al., 2019). Other NBR models combine preferences, popularity, and transitions (Peng et al., 2023), infer intentions behind basket sequences (Wang et al., 2020), or separate habitual demand from exploratory interest while accounting for elapsed time (Deng et al., 2026). Calendar-time representations also support item-specific repurchase cadences (Cao et al., 2026).

The relevance of historical information is itself a modeling question. Contrastive basket denoising learns which historical items matter for a target item (Qin et al., 2021). FlySCoRe makes a related selection over labeled history–candidate observations, storing their purchase and opportunity counts so that shared experience can adjust a personal-history score. Prior analyses also show that repeat and explore items contribute differently to NBR accuracy (Li et al., 2023) and distinguish accuracy from other recommendation objectives (Li et al., 2024). This motivates the history-based recurring-demand setting and full-catalog protocol used here.

Associative Computation in the Mushroom Body. The mushroom body offers a biological example of how sensory cues can access experience. Anatomical and functional studies link its compartmental organization and output neurons to associative learning and memory-guided behavior (Aso et al., 2014a;b). Projection-neuron inputs converge in diverse combinations onto Kenyon cells (Caron et al., 2013); feedback inhibition helps maintain sparse, decorrelated activity and odor discrimination (Lin et al., 2014). Output representations reflect experience-dependent plasticity (Hige et al., 2015). Dense connectomic analysis further reveals structured sensory inputs, output pathways, and feedback within this system (Li et al., 2020). These findings motivate selective associative organization as a design principle. In FlySCoRe, shopping observations supply the memory content, and the recommendation task determines the meaning of the readout.

Sparse Retrieval and Connectome-Constrained Models. Computational analyses relate sparse connectivity to the dimensionality of representations used for associative learning (Litwin-Kumar et al., 2017). Fly-inspired hashing connects expansion and sparse activity to similarity search (Dasgupta et al., 2017; Sharma & Navlakha, 2018). Explicit memory is also used in machine learning: end-to-end memory networks learn attention-based access to stored representations (Sukhbaatar et al., 2015). FlySCoRe uses sparse address overlap to retrieve outcome statistics, allowing its score to depend on both the observed purchases and the amount of supporting evidence.

Measured connectivity provides another way to constrain computation. Connectome-constrained networks have been optimized for visual motion processing (Lappalainen et al., 2024), while whole-brain LIF models have been used to study sensorimotor responses (Shiu et al., 2024). MaleCNS supplies the measured connections used in our recommendation model (Berg et al., 2026). CNS first transfers this structural organization to outcome retrieval; PROB and NET then express the retrieval function through alternative access rules. The reconstruction experiment examines a further use of structure: composing retained connections and input activity to rebuild memory access at a different interface.

B Model Implementation

B.1 Connectivity and Dimensions

All branches use PyTorch with continuous-valued activations, the same pairwise ranking objective, and validation-based checkpoint selection. CNS uses MaleCNS v1.0 mushroom-body connectivity (Berg et al., 2026): 686 ALPNs, 4,064 KCs, and 97 MBONs, with 22,586 ALPN–KC and 61,210 KC–MBON connections. Measured synaptic contact counts are normalized by total incoming contacts and remain fixed. PROB and NET each use 128-dimensional input encodings and 512-dimensional addresses.

B.2 CNS Memory Implementation

Each CNS output channel gathers evidence through its measured incoming connections. Normalizing connection activity puts the channels on a common scale before they contribute to the final score. The formulas below specify this access rule and an equivalent compact computation. The input factor $1/2$ in Eq. (1) keeps the combined history and product scale comparable to either projected input before applying $\tanh$.

Fix a history–candidate pair $q = (u, i)$ and write its address as $\mathbf{K}(q)$, as in the branch descriptions. Let C_{mj} be the nonnegative normalized weight from KC j to MBON m. Its total connection activity $Z(q)$ and connection access weights are

$$\begin{aligned} Z(q) &= \sum_j K_j(q) \sum_{m=1}^{B} |C_{mj}|, \\ f_{mj}(\mathbf{K}(q)) &= \begin{cases} K_j(q) C_{mj} / Z(q), & Z(q) > 0, \\ 0, & Z(q) = 0. \end{cases} \end{aligned} \tag{6}$$

For each output channel m, historical access weights $f_{mj}(\mathbf{K}(r))$ write S_{mj}, N_{mj}. Reading these statistics with the current query gives purchase total $s_m = \sum_j f_{mj}(\mathbf{K}(q)) S_{mj}$ and opportunity total $n_m = \sum_j f_{mj}(\mathbf{K}(q)) N_{mj}$. The M row of Eq. (1) combines these totals with the user signal.

Writing and reading through the same connection uses its weight twice. This permits a compact computation that sums historical contributions directly, while preserving the connection-specific evidence totals. For observed history–product pair r, the address weights $K_j(r)$ have normalization $Z(r) = \sum_j K_j(r) \sum_{m=1}^{B} |C_{mj}|$. For nonzero query and support activity, substituting the memory write into the read yields

$$s_m = \sum_r w_r y_r \sum_j \frac{K_j(q) K_j(r) C_{mj}^2}{Z(q) Z(r)}. \tag{7}$$

The opportunity total n_m replaces y_r by one. Zero-activity inputs contribute zero by Eq. (6).

B.3 Encoders, User Signal, and Training

Encoder Architecture. The encoder combines history, product, and candidate evidence so that memory selection can depend on their interactions. Learned projections put the input sources on compatible scales, and a bilinear term represents history–product interactions (Kim et al., 2017). The following expansion specifies this input interface; retrieval uses its joint output.

For query u and candidate i, write the history and product features as $\mathbf{x}_u, \mathbf{x}_i$. Learned projections give $\mathbf{h}_u = \tanh(P_h \mathbf{x}_u + \mathbf{b}_h)$ and $\mathbf{e}_i = \tanh(P_e \mathbf{x}_i + \mathbf{b}_e)$, where P_h, P_e are matrices and $\mathbf{b}_h, \mathbf{b}_e$ are biases. Candidate evidence $\mathbf{z}_{ui}$ contains the encoder features above and the user-signal statistics defined next. For PROB and NET, learned gains $\mathbf{g}_h, \mathbf{g}_e$ control the history and product contributions; W_z projects the encoder block of $\mathbf{z}$, and W_h, W_e, W_o form the bilinear interaction. With coordinatewise multiplication $\odot$ and sigmoid $\sigma(t) = 1/(1 + e^{-t})$, the encoder is

$$\Phi(\mathbf{h}, \mathbf{e}, \mathbf{z}) = \tanh\big[\sigma(\mathbf{g}_h) \odot \mathbf{h} + \sigma(\mathbf{g}_e) \odot \mathbf{e} + W_z \mathbf{z} + W_o((W_h \mathbf{h}) \odot (W_e \mathbf{e}))\big]. \tag{8}$$

Each branch has separately fitted parameters. During the reported adaptation, PROB and NET update the joint mixer and the subsequent address map; CNS updates the history and product projections into ALPN activity.

Learned Starting Points. The reported adaptation begins with saved learned components. Complete-input models trained on original training histories supply feature encoders and the KC-processed user signal. Training-basket Triple2Vec embeddings initialize the user and product relation representations. A subsequent ranking stage on the complete repeat-purchase training pool supplies the address or ALPN projection initialization. That stage permits six epochs and materializes up to 6,000 training queries, processing at most 1,000 per epoch. The final adaptation retains the feature and user-signal components and updates only the branch-specific parameters described above. The supplied initialization checkpoints are therefore part of the experimental specification.

Historical Support and Memory Estimation. Each support row pairs a training-history prefix with its successor basket. A household-balanced sample of up to 3,000 prefixes is augmented with all scheduled training queries. Each retained prefix has unit weight; all of its successor purchases contribute to memory, including products new to that household. Memory pools these candidate pairs across households and products through their addresses. The purchase totals include every positive outcome. For CJ, TF, and IC, opportunity totals combine those exact positives with uniformly sampled nonpurchases, weighted by the inverse sampling probability, using 256 catalog candidates per support batch. DC enumerates its complete 129-product catalog. The opportunity total thus estimates full-catalog candidate-pair support.

Support is drawn from recorded household history preceding the corresponding prediction cutoff. IC records household-relative elapsed days, whereas the other datasets provide calendar dates. Within each household, the strict ordering excludes the current outcome, same-day outcomes, and later outcomes. Memory construction, feature learning, and product-frequency estimation use the available fitting histories.

KC-Processed User Signal. The user signal balances recent purchases, persistent household identity, and training product frequency. A frozen history-conditioned router supplies mixture weights for a constant component and eight exponential-decay components, preserving both long-term and recent evidence. The signal rescales these statistics within each household. Let $\mathcal{I}$ be the product catalog. For query u and product i, let c_{ui} be the sum of the router-weighted temporal contributions over earlier purchases, and let f_i be the fraction of training baskets containing i. With smoothing weight $\beta > 0$, the signal is $\phi(\mathbf{z}_{ui}) = (c_{ui} + \beta f_i)/(\max\{1, \max_{g\in\mathcal{I}} c_{ug}\} + \beta)$, where g ranges over the catalog. This fixed computation uses the user-signal statistics in $\mathbf{z}_{ui}$. The optimization protocol below specifies which signal quantities remain fixed.

Training Objective. The training loss favors purchased candidates over unpurchased candidates, with greater weight on pairs that affect the evaluation ranking most. At prediction time, products are ranked by decreasing score M_{ui} for query u; the first K form the recommendation. During training, i^+ denotes a purchased candidate and i^- an unpurchased candidate in the sampled comparison set. Scores are expressed as log-odds $q_{ui} = \log(M_{ui}/(1 - M_{ui}))$, which preserve their order. Following LambdaRank (Burges et al., 2006), the nonnegative pair weight $\lambda_{ui^+i^-}$ is the absolute NDCG change from swapping the pair's ranks, averaged over training cutoffs and adjusted for candidate sampling. NDCG is defined in Section 4.3. All branches minimize

$$\text{Loss} = \frac{\sum_{u,i^+,i^-} \lambda_{ui^+i^-} \log(1 + \exp(q_{ui^-} - q_{ui^+}))}{\sum_{u,i^+,i^-} \lambda_{ui^+i^-}}. \tag{9}$$

Pair weights are fixed during each gradient calculation. PROB and NET learn their address parameters; CNS learns its input projections. User embeddings, the user signal, and measured connections remain fixed. Memory writes are computed without gradients and refreshed once per epoch; gradients pass through the query address. The forward pass uses the seven selected coordinates, with a dense straight-through relaxation for its backward pass. Candidate sampling supplies the pairwise comparisons, with up to 32 high-scoring sampled negatives per positive and capped importance weights.

Optimization and Model Selection. Training adapts the input and addressing parameters to product ranking, while validation selects the checkpoint. Memory is rebuilt from historical observations under the selected parameters. We set the number of selected memory coordinates to $k = 7$ and use adaptive moment estimation with decoupled weight decay (AdamW) with a learning rate and weight decay of 10^{-5}. For the three complete branches, training runs for at most six epochs and stops after two validation checks without improvement. For each branch, we select the highest validation

NDCG@20 among its saved initialization and subsequent training epochs. All three branches are evaluated over the complete product catalog.

For the KC-processed user signal defined in Section 3.2, the router parameters and decay time scales are inherited from the complete-input backbone and held fixed throughout the reported adaptation; $\beta = 2$. Preprocessing computes the normalization scale across the entire catalog before selecting candidates. This bounded preference score smooths the retrieved evidence in M. We set signal strength to a fixed multiple of the median retrieved opportunity support on training data and then hold it constant during optimization. On CJ, the multiplier is 0.1 for PROB and NET and 0.5 for CNS. All three use 1 on TF and DC. On IC, PROB, NET, and CNS use 1, 5, and 2, respectively.

Triple2Vec selects embeddings by training NCE loss and its adaLoyal adjustment by validation NDCG@20.

C Data and Experimental Setup

C.1 Dataset Preparation

We study the recurring-demand setting for established households. The focal query population is determined solely from recurring-purchase evidence in the household history available at the prediction cutoff. Every branch uses the same history-eligible queries and ranks the complete product catalog. Appendix D.2 provides the complete stratified analysis across history length, product variety, and repeated purchasing; these supplementary comparisons characterize other history profiles, whereas the paper's focal setting remains recurring demand.

C.2 Input Features

The input schema accommodates the different shopping and product fields supplied by each dataset. CJ supplies product hierarchy, manufacturer, brand, package text, price and discount variables, quantity, store, and time. IC supplies product text and hierarchy, cart position, reorder status, inter-order time, and calendar context. TF supplies hierarchy, quantity, asset and price, demographic group, region code, and calendar context. DC supplies commodity, brand, description, size, basket size, and time.

C.3 Baseline Mechanisms

Frequency and Neighborhood Methods. Global popularity (GPOP) ranks products by training purchase frequency. Personal top frequency with global-popularity completion (GP-TopFreq) ranks by household purchase frequency and fills remaining positions with globally popular products (Li et al., 2023). Temporal-item-frequency-based user k-nearest neighbors (TIFU-KNN) combines temporally weighted personal frequency with neighboring shopping histories (Hu et al., 2020).

Latent Representation Methods. Bayesian personalized ranking with matrix factorization (BPR-MF) learns user and product factors through pairwise ranking (Rendle et al., 2009). Triple2Vec learns transaction-triple embeddings by noise-contrastive estimation (NCE), with the authors' validation-selected adaLoyal adjustment (Wan et al., 2018). The light graph convolution network (LightGCN) propagates representations over the interaction graph (He et al., 2020).

Sequential and Temporal-Set Models. The dynamic recurrent basket model (DREAM) models basket sequences recurrently (Yu et al., 2016); Sets2Sets models temporal sets with repeat information (Hu & He, 2019). The deep neural network for temporal sets prediction (DNNTSP) combines element relationships, temporal attention, and shared-pattern fusion (Yu et al., 2020). Simplified fully connected networks for temporal sets prediction (SFCNTSP) learn set, temporal, and embedding-channel relationships (Yu et al., 2023). Time-interval disentangled experts (TIDE) combine habitual and exploratory signals with time intervals (Deng et al., 2026).

C.4 Baseline Configurations

Baseline configurations adapt each method to the available histories while using validation data to choose its settings. Appendix C.3 introduces their mechanisms; the settings below make those comparisons reproducible.

Validation NDCG@20 selects configurations and checkpoints, with Recall@20 breaking ties. DNNTSP, SFCNTSP, and TIDE use at most 20 baskets and 64 historical products. DNNTSP and SFCNTSP use batch size 64 for up to 40 epochs. SFCNTSP uses adaptive moment estimation (Adam) with learning rate 10^{-3}, dropout 0.1, one final training prefix per household, and 256 importance-weighted negative targets. Validation selects the paper's full element operator with all released dropout sites on DC; CJ, TF, and IC use the released diagonal operator with input/channel dropout. Sets2Sets uses up to 1,000 baskets, batch size 64, and 20 epochs; CJ uses batch size 16 and 40 epochs, with early stopping after 1,500 updates. TIFU-KNN uses clustered neighbor search on IC and exact search elsewhere. TIDE retains at most 50,000 pairs from 100,000 training baskets, considering the first 20 products in product-identifier order per basket.

BPR-MF uses batches of 131,072 (DC: 100 batches of 2,048 per epoch). LightGCN uses full-edge gradients on TF/DC and ten batches of 2,048 per epoch on CJ/IC. DREAM samples one positive–negative pair per transition. Triple2Vec uses 250 batches of 2,048 per epoch.

D Purchase History and Recommendation Analysis

D.1 Purchase History Statistics

Long histories and sparse product-level evidence can coexist because shopping occasions are spread over many products. The distributions below make this distinction visible and give empirical context to the candidate-specific retrieval example in the introduction. Figure 4 describes every training sample in Table 1. Each point represents one training query and summarizes the baskets available before that prediction. A household with several training targets contributes one point at each cutoff. The cumulative distributions give equal weight to each observed sample–product pair within a dataset, using its preceding purchase count.



Figure 4: Purchase histories for all training samples in Table 1. (a) Each point is one training query, positioned by its numbers of preceding baskets and distinct products. Color indicates the fraction of products purchased only once; both axes are logarithmic. The panels contain 159, 84, 170, and 5,686 samples for CJ, TF, DC, and IC, respectively. (b) Cumulative purchase-count distributions over the 275,695 observed sample–product pairs across these training histories.

D.2 Performance Across Purchase Histories

Aggregate accuracy combines households with different amounts and kinds of shopping experience. We examine how performance varies with history length, product variety, and repeated purchasing, alongside the prevalence of each type of history. Figure 5 compares the three branches with GP-TopFreq, TIFU-KNN, Triple2Vec, and TIDE, representing frequency, neighborhood, latent-representation, and temporal-expert approaches. These groups use the history-eligible households; the preceding figure describes training queries.

Each characteristic is measured from baskets available to the model before prediction. History length counts these baskets; product variety counts their distinct products; the repeat fraction is the share of basket–product occurrences beyond each product's first occurrence. We divide each observed feature range into six equal-width bins and retain all evaluated households. Bars show the proportion of households in a bin on the right axis. Lines show mean Recall@20 or NDCG@20 within the same bin on the left axis, with points placed at bin centers. Lines connect the nonempty groups in order along the horizontal axis. Empty bins have no performance estimate.

For each nonempty bin, all methods use the same households. We obtain pointwise 95% percentile intervals from 2,000 paired household bootstrap samples; intervals are shown when a bin contains at least two households. The resulting bands describe evaluation-sample variation for fixed models. Together, the bars and curves distinguish performance in common history groups from estimates based on sparsely populated groups.

On IC, accuracy generally decreases as histories become longer or cover more products. NET and CNS retain higher NDCG than PROB in the longest history groups. On TF, PROB has the highest NDCG among the three branches in every populated history-length and product-variety group. On DC, the branches have similar Recall, while NET provides stronger ordering across the history-length groups. CJ shows more variation in which branch leads. These patterns locate differences between the implementations within particular types of purchase history.

D.3 Recommendation Distribution

Accuracy describes how well a ranked list recovers the next basket. We also examine how its recommendations are distributed across products and households. Figure 6 pairs Recall@20 and NDCG@20 with three complementary descriptions of the same top-20 lists, for every complete method. Catalog coverage is the percentage of catalog products recommended to at least one household. Exposure concentration is the Gini coefficient of recommendation counts over the complete catalog, including products that receive no recommendations. Personalization is one minus the average overlap between two households' lists, with overlap measured as the number of shared products divided by 20. Thus coverage describes breadth, the Gini coefficient describes concentration, and personalization describes variation between households.

Each point represents one method evaluated on the same households and catalog within a dataset. The plots place ranking accuracy alongside the breadth, concentration, and household specificity of the recommendations. Together with the purchase-history analysis, they describe both who receives accurate recommendations and how the resulting lists are distributed.

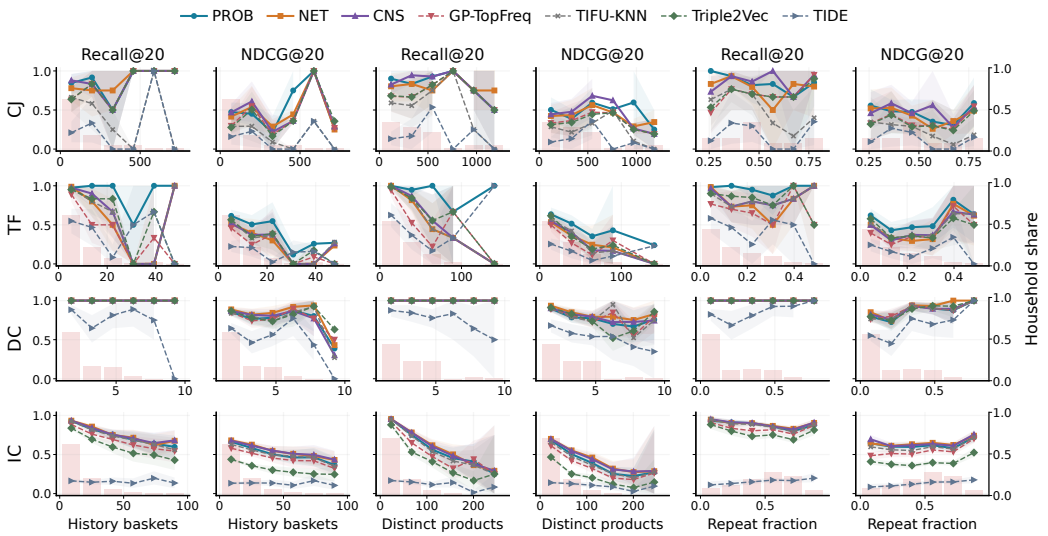


Figure 5: Purchase-history characteristics and recommendation performance. Rows show datasets; column pairs show Recall@20 and NDCG@20 for each characteristic. Bars show household proportions on the right axis. Lines and 95% household-bootstrap intervals use the left axis for accuracy. Solid lines show FlySCoRe branches and dashed lines show representative baselines. Lines connect actual nonempty-group means.

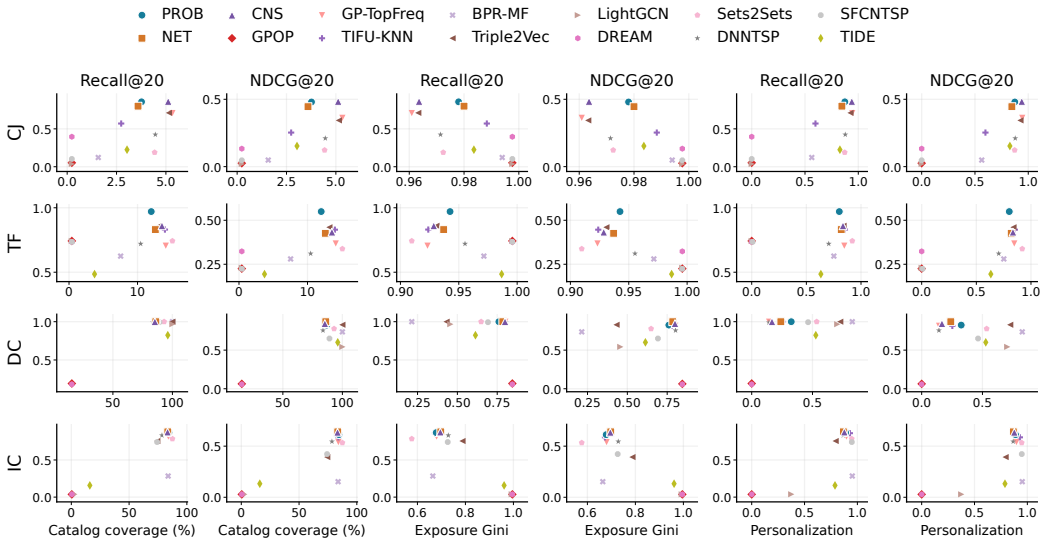


Figure 6: Accuracy and recommendation distribution for all fourteen complete methods. Rows show datasets. Pairs of columns relate Recall@20 and NDCG@20 to catalog coverage, exposure Gini, and personalization. Each point uses the same saved top-20 rankings as the corresponding accuracy result. Axis ranges adapt to each dataset; the measures are descriptive properties of the recommendations.

E SUPPLEMENTARY EXPERIMENTS

E.1 EVALUATION METRIC DEFINITIONS

For query u, write R_u^K for its top-K recommendations and Y_u for its actual next basket. The evaluated queries form $\mathcal{Q}$.

Recall. Recall measures the fraction of each true basket recovered:

$$\text{Recall}\,@K = \frac{1}{|\mathcal{Q}|}\sum_{u\in\mathcal{Q}}\frac{|R_u^K \cap Y_u|}{|Y_u|}. \tag{10}$$

NDCG. NDCG discounts each correct recommendation by its rank and normalizes the total by an ideal ordering with purchased products first. Let $i_{u,t}$ be the product at rank t; the indicator is one for a purchase and zero otherwise. Then

$$\text{NDCG}\,@K = \frac{1}{|\mathcal{Q}|}\sum_{u\in\mathcal{Q}}\frac{\displaystyle\sum_{t=1}^{K}\frac{\mathbf{1}\{i_{u,t}\in Y_u\}}{\log_2(t+1)}}{\displaystyle\sum_{t=1}^{\min(K,|Y_u|)}\frac{1}{\log_2(t+1)}}. \tag{11}$$

The normalized score lies in $[0, 1]$. Together, Recall and NDCG distinguish basket coverage from ranking quality.

E.2 ADDITIONAL RANKING METRICS

The supplementary metrics assess recommendation relevance and whether a query receives at least one correct suggestion. Table 4 reports Precision and PHR at ranks 10 and 20 on four datasets. It uses the same methods, history-eligible queries, and table layout as Table 2. Recall and NDCG are defined in Appendix E.1. Using the same notation, we define the two supplementary metrics below. Values are query averages, reported to four decimal places.

Precision. Precision measures the fraction of recommendations that are purchased:

$$\text{Precision}\,@K = \frac{1}{|\mathcal{Q}|}\sum_{u\in\mathcal{Q}}\frac{|R_u^K \cap Y_u|}{K}. \tag{12}$$

Together, Recall and Precision distinguish target-basket coverage from the proportion of recommended products that are relevant.

PHR. PHR measures the fraction of queries with at least one correct recommendation:

$$\text{PHR}\,@K = \frac{1}{|\mathcal{Q}|}\sum_{u\in\mathcal{Q}}\mathbf{1}\{|R_u^K \cap Y_u| > 0\}. \tag{13}$$

E.3 RECOMMENDATION LENGTH

Figure 3 in the main text compares all methods over recommendation lengths from 5 to 40. PROB's advantage on TF persists across these lengths. On DC, coverage saturates early, while NDCG distinguishes product ordering. On IC, the three branches have similar coverage, with differences in ranking quality. The pointwise 95% intervals use 2,000 paired household bootstrap samples with fitted models fixed, measuring evaluation-sample variation rather than training variation.

E.4 MODULE ABLATIONS

The bypasses measure how rankings change when encoding, addressing, or readout is removed from a fixed CNS model. The last three rows of Table 2 use the selected CNS model as their reference.

Table 4: Precision and PHR comparison on four datasets. The last three rows show CNS module bypasses (without, or w/o) with frozen parent parameters. Four decimals are shown. Bold and underlining mark the best and second-best complete methods before rounding; ties share marks.

Method	Complete Journey				Ta-Feng				Dunnhumby-Carbo				Instacart			
	Precision@K ↑		PHR@K ↑		Precision@K ↑		PHR@K ↑		Precision@K ↑		PHR@K ↑		Precision@K ↑		PHR@K ↑	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
GPOP	0.0063	0.0063	0.0625	0.0938	0.0206	0.0426	0.1765	0.7941	0.0065	0.0127	0.0654	0.2418	0.0099	0.0063	0.0912	0.1138
GP-TopFreq	0.0906	0.0656	0.6562	0.8750	0.0676	0.0449	0.5735	0.7206	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.1536	0.0976	0.8297	0.9123
TIFU-KNN	0.0813	0.0547	0.5625	0.6875	0.0868	0.0522	0.7206	0.8529	0.1229	**0.0618**	0.9935	**1.0000**	0.1687	0.1036	0.8671	0.9418
BPR-MF	0.0156	0.0156	0.1250	0.2188	0.0515	0.0375	0.4853	0.6912	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.0394	0.0305	0.3128	0.4335
Triple2Vec	0.0875	0.0656	0.6562	0.8750	0.0912	0.0537	0.7794	0.8824	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.1223	0.0884	0.7428	0.8827
LightGCN	0.0094	0.0047	0.0625	0.0625	0.0147	0.0426	0.1176	0.7941	0.1013	0.0595	0.8693	0.9804	0.0096	0.0065	0.0904	0.1147
DREAM	0.0219	0.0266	0.2188	0.5000	0.0838	0.0419	0.7941	0.7941	0.0059	0.0118	0.0588	0.2288	0.0101	0.0063	0.0930	0.1129
Sets2Sets	0.0375	0.0203	0.2812	0.2812	0.0574	0.0471	0.4706	0.7794	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.1411	0.0890	0.8184	0.9053
DNNTSP	0.0594	0.0422	0.3750	0.5312	0.0574	0.0463	0.4853	0.7647	0.1216	0.0611	0.9935	0.9935	0.1629	0.1014	0.8454	0.9253
SFCNTSP	0.0094	0.0141	0.0625	0.2188	0.0132	0.0419	0.1324	0.7941	0.1190	0.0611	0.9804	**1.0000**	0.1196	0.0885	0.7228	0.8732
TIDE	0.0375	0.0234	0.3125	0.4062	0.0426	0.0279	0.3824	0.5147	0.0850	0.0503	0.7712	0.8562	0.0361	0.0224	0.2997	0.3458
FlySCoRe-PROB	**0.1250**	0.0781	0.8125	0.9375	**0.1044**	**0.0603**	**0.8676**	**0.9853**	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.1721	0.1049	**0.8844**	**0.9453**
FlySCoRe-NET	**0.1250**	0.0734	0.8125	0.9062	0.0765	0.0529	0.6324	0.8529	**0.1235**	**0.0618**	**1.0000**	**1.0000**	**0.1758**	**0.1080**	0.8818	**0.9453**
FlySCoRe-CNS	0.1187	**0.0797**	**0.8438**	**0.9688**	0.0824	0.0544	0.6765	0.8676	**0.1235**	**0.0618**	**1.0000**	**1.0000**	0.1748	0.1077	0.8801	**0.9453**
w/o Encoding	0.1188	0.0797	0.8125	1.0000	0.0735	0.0522	0.6176	0.8382	0.1235	0.0618	1.0000	1.0000	0.1752	0.1074	0.8818	0.9444
w/o Addressing	0.0000	0.0000	0.0000	0.0000	0.0059	0.0029	0.0588	0.0588	0.0157	0.0219	0.1503	0.3987	0.0011	0.0008	0.0113	0.0165
w/o Readout	0.1219	0.0797	0.8438	0.9688	0.0779	0.0529	0.6471	0.8382	0.1235	0.0618	1.0000	1.0000	0.1754	0.1075	0.8810	0.9453

w/o Encoding bypasses A, concatenating the 96 history and 48 product features and zero-padding them to the 686-dimensional ALPN interface, without learned projection or nonlinear mixing. *w/o Addressing* removes top-k selection and the user signal from K, retaining fixed expansion, positive activation, and normalization to pass all positive coordinates to M. Outcome Memory still writes and retrieves S/N. Without a user signal, M averages the channel purchase rates s_m/n_m, assigning zero to unsupported channels. This removes the user-signal path and lets the retrieved evidence determine the score directly. *w/o Readout* bypasses M, preserves A and K, and ranks products directly by the user signal supplied by K. It omits S/N writing, retrieval, and evidence fusion, without adding a prediction head. Remaining parameters are fixed, with no retraining or signal-strength calibration.

E.5 Module Replacement and Retrieval Reconstruction

The replacement experiments use other representations and readouts to carry out the model's computational roles. For each control, we specify the information retained, the operation replaced, and the parameters eligible for fitting. Table 5 reports the raw Recall and NDCG values; Table 3 expresses them relative to the highest score among the complete FlySCoRe branches in each column. All variants use the same history-eligible queries and complete catalogs.

Input Encoding Replacements (A). This control tests the contribution of learned input representations. Random encoding replaces the learned ALPN projections with nontrainable Gaussian projections of matching dimensions, retaining the original input processing and nonlinear activation. The remaining parameters are fixed.

Memory Addressing Replacements (K). These replacements test memory access at another layer and with a different selection rule. ALPN-addressed memory composes the measured ALPN–KC and KC–MBON operators into a direct map. Each input coordinate A_j is transformed to $(A_j + 1)/2$ and normalized by the composed outgoing connection strength. These normalized input coordinates replace KC addresses in memory writing and retrieval. We rewrite the memory statistics from historical support observations at this input-layer interface. The original input projections, measured matrices, and remaining model settings remain fixed. This reconstruction uses activity retained by the simulated model. Retrained dense addressing removes sparse selection and refits the encoder and signal strength under the original training protocol and matched initialization.

Evidence Readout Replacements (M). These controls vary how retrieved evidence is converted into a score. Readout bypass is the M-module ablation control from Table 2: it omits outcome-

memory writing, retrieval, and evidence fusion, returning the score supplied by the unchanged K module without refitting. Calibrated constant fusion replaces support-dependent fusion by a single mixture weight selected on validation data. Trained residual readout replaces S/N retrieval with an affine head on MBON activity, added to the existing user-signal log-odds; only this 98-parameter head is fitted, for up to 30 epochs.

Retrained dense addressing reaches NDCG@20 of 0.8412 on DC against full CNS at 0.8435; constant fusion reaches 0.8474. A compact residual head reaches 0.6372 on IC against 0.6357 for CNS. These controls show how alternative addressing and readout operations affect the final ranking.

Table 5: Performance under module replacement. CNS replacements use the same queries and catalogs as the complete methods. The first two rows give columnwise maxima across the three complete FlySCoRe branches and eleven benchmarks, respectively. Four decimal places are shown.

Implementation	Complete Journey				Ta-Feng				Dunnhumby-Carbo				Instacart			
	Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑		Recall@K ↑		NDCG@K ↑	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
Best FlySCoRe	0.7005	0.8620	0.4321	0.4841	0.8309	0.9706	0.5122	0.5495	1.0000	1.0000	0.8681	0.8681	0.7445	0.8789	0.5947	0.6401
Best benchmark	0.5109	0.7109	0.2999	0.3624	0.7426	0.8603	0.4274	0.4602	1.0000	1.0000	0.8283	0.8283	0.7294	0.8605	0.5429	0.5863
Input encoding replacements (A)																
Random encoding (A)	0.6849	0.8542	0.3956	0.4497	0.5735	0.8088	0.3389	0.4013	1.0000	1.0000	0.8399	0.8399	0.7491	0.8785	0.5839	0.6274
Memory addressing replacements (K)																
ALPN-addressed memory (K)	0.6797	0.8620	0.4156	0.4696	0.5956	0.8309	0.3670	0.4296	1.0000	1.0000	0.8222	0.8222	0.7470	0.8771	0.5870	0.6307
Retrained dense addressing (K)	0.7083	0.8620	0.4258	0.4731	0.6029	0.8162	0.3582	0.4152	1.0000	1.0000	0.8412	0.8412	0.7478	0.8763	0.5882	0.6315
Evidence readout variants (M)																
Readout bypass (M)	0.6797	0.8620	0.4149	0.4689	0.5956	0.8309	0.3648	0.4273	1.0000	1.0000	0.8238	0.8238	0.7469	0.8754	0.5872	0.6306
Calibrated constant fusion (M)	0.6849	0.8385	0.4181	0.4672	0.5956	0.8309	0.3648	0.4273	1.0000	1.0000	0.8474	0.8474	0.7497	0.8824	0.5911	0.6359
Trained residual readout (M)	0.6797	0.8620	0.4149	0.4689	0.5956	0.8309	0.3648	0.4273	1.0000	1.0000	0.8280	0.8280	0.7505	0.8815	0.5931	0.6372

E.6 Interpretation of Retrieval and Reconstruction

Candidate-dependent access explains how the memory can change a ranking. The user signal gives an initial ordering, and the address determines which additional outcomes can adjust each score. In PROB, the overlap $\kappa(q, r)$ measures the probability that a query and support observation select the same unit. Equation (4) translates this overlap into the observation's contribution.

The model comparison and CNS interventions examine this mechanism at different levels. PROB and NET fit their parameters and calibrate signal strengths separately, so their results compare complete implementations of a common retrieval function. The CNS interventions change specified computations within one model. ALPN-addressed memory uses the retained activity and composed connections to construct access at a new interface. The composed map connects input coordinates to output channels, and the new address uses those connections to retrieve historical evidence and reconstruct the ranking behavior at that interface.

The four datasets cover different catalog sizes, history lengths, and purchase patterns. IC provides the largest history, with 723,372 transaction records in 118,113 baskets. Appendix D.2 reports the complete stratified analysis across history length, product variety, and repeated purchasing. These results provide complementary context across observed history profiles, while the paper's claims remain focused on the recurring-demand setting. The repeated adaptation runs quantify sampling and optimization variation from a common learned starting point, and the household-bootstrap intervals in the supplementary figures describe evaluation-sample variation with the fitted models fixed.

E.7 Spiking Dynamics Experiments

The retrieval interpretation also permits a controlled change in how activity is represented. We examine leaky integrate-and-fire (LIF) dynamics at two scales: signal propagation through a frozen MaleCNS graph and spike-count addressing in the recommendation model. The first experiment characterizes downstream spiking responses to stimulation. The second uses the same queries, complete catalogs, and memory-readout procedure as CNS to assess the task consequences of replacing continuous KC activity with finite-window spike counts. Both experiments are computational checks under fixed simulation settings: they examine propagation and representation changes within the transferred architecture rather than biological plausibility.

Connectome Simulation. We test whether sensory stimulation produces spikes in downstream memory and output populations, and how this response changes with input strength. The annotated MaleCNS graph contains 166,700 neurons, 25,582,938 directed neuron-pair edges, and 124,177,617 synaptic contacts. All connections and physiological parameters remain fixed. The simulator uses a 0.1 ms time step, resting and reset potentials of -52 mV, a firing threshold of -45 mV, a 20 ms membrane time constant, a 5 ms synaptic time constant, a 1.8 ms transmission delay, and a 2.2 ms refractory period. Synaptic efficacies use contact counts with a 0.275 mV contact gain and a fixed global multiplier of 0.30. Signs follow the annotation-based transmitter mapping: acetylcholine is positive; γ-aminobutyric acid (GABA), glutamate, and histamine are negative; remaining entries use the positive default.

We stimulate all 686 ALPNs with independent Poisson inputs at 0, 50, or 200 Hz for 50 ms, starting from rest. Input neurons emit the imposed spikes, which propagate through the frozen graph. KC and MBON neurons receive no direct stimulation or output clamping. We repeat each condition with three seeds and report population-mean firing rates, averaged across the 686 ALPNs, 4,064 KCs, and 97 MBONs. The graph fingerprint remains unchanged in every run, and repeating the final seed reproduces its spike counts exactly.

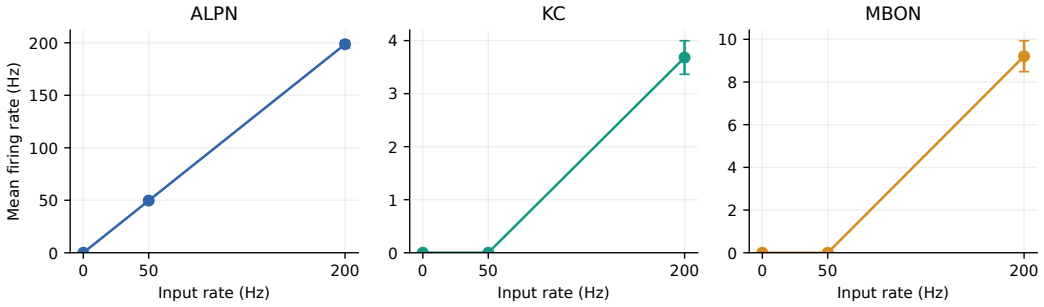


Figure 7: Population responses in the full-graph LIF simulation. Each panel shows one population's mean firing rate as the imposed ALPN input increases. Error bars are sample standard deviations over three independent Poisson input seeds. All conditions use the same frozen graph and a 50 ms window.

With zero input, all three populations remain silent. At 50 Hz, ALPNs follow the imposed input while KC and MBON spike counts remain zero within the 50 ms window. At 200 Hz, mean firing rates are 198.60 ± 3.36 Hz in ALPNs, 3.68 ± 0.32 Hz in KCs, and 9.21 ± 0.72 Hz in MBONs. Activity therefore reaches both downstream populations. These responses demonstrate threshold-dependent propagation under the specified stimulation and simulation settings.

Recommendation with LIF Activations. We next replace CNS's KC activation strengths with finite-window LIF spike counts. The trained CNS parameters and measured connections remain fixed. Each history–candidate pair uses its existing measured projection $\mathbf{v} = C_{AK}^{\mathsf{T}}\mathbf{A}$. Positive projected activity is encoded as a constant current with a 20 mV peak per pair. Let U_j be membrane potential relative to rest, I_j its constant input, and $\tau = 20$ ms:

$$I_j = 20\,\frac{[v_j]^+}{\max_d [v_d]^+}, \qquad \tau\frac{\mathrm{d}U_j}{\mathrm{d}t} = -U_j + I_j. \tag{14}$$

All currents are zero when the maximum is zero. Each neuron starts at $U_j = 0$, emits a spike at 7 mV, resets to zero, and remains refractory for 2.2 ms. We count spikes over 50 ms using exact threshold-crossing times under constant input. This calculation agrees with an independent event-by-event simulation for 1,001 currents spanning 0–20 mV.

The original seven selected KC coordinates are retained, and their spike counts are normalized to unit total weight to form the address; an all-zero count vector gives a zero address. Historical support observations rebuild the outcome memory with these addresses. The original connection-weighted readout then produces the ranking. All settings are fixed before evaluation, with no fitting of model or simulation parameters. This control measures the effect of spike-count addressing within the same retrieval architecture; the full-graph simulation above examines propagation separately.

Table 6: Matched KC-LIF recommendation control with frozen CNS parameters and rebuilt outcome memory.

Dataset	Activation	Recall@10 ↑	Recall@20 ↑	NDCG@10 ↑	NDCG@20 ↑
Complete Journey	Continuous	0.6693	0.8620	0.4257	0.4841
Complete Journey	LIF counts	0.6693	0.8620	0.4257	0.4842
Ta-Feng	Continuous	0.6324	0.8603	0.3724	0.4326
Ta-Feng	LIF counts	0.6324	0.8603	0.3726	0.4330
Dunnhumby-Carbo	Continuous	1.0000	1.0000	0.8435	0.8435
Dunnhumby-Carbo	LIF counts	1.0000	1.0000	0.8411	0.8411
Instacart	Continuous	0.7436	0.8762	0.5909	0.6357
Instacart	LIF counts	0.7436	0.8767	0.5909	0.6358

The spike-count control retains 99.71–100.08% of continuous CNS NDCG@20 across the four datasets. Complete Journey, Ta-Feng, and Instacart show small increases, whereas Dunnhumby-Carbo shows a small decrease. With parameters fixed and memory rebuilt, finite-window spike counts provide effective addresses for the same retrieval computation. These results support the use of measured connectivity for memory access under both continuous and spike-count activation.

E.8 Adaptation Data Sensitivity

This experiment examines how additional supervised adaptation changes retrieval from a learned starting point. We use 25%, 50%, or 100% of the training queries in Table 1. Smaller samples are nested within larger ones, and all branches share the sampled queries for each dataset and seed. Each branch starts from its original learned initialization and adapts the same parameters as in the main experiment. Historical features and evaluation queries, and candidate catalogs remain fixed. Memory is rebuilt from the standard bounded historical support and the selected queries.

Three seeds govern query selection, candidate sampling for memory estimation, and optimization. Each run calibrates signal strength from its support and then holds it fixed. Training allows up to six epochs; validation NDCG@20 selects the checkpoint, including the initialization. Validation occurs every epoch on CJ, TF, and DC and every two epochs on IC; two successive checks without improvement stop training. Figure 8 shows mean evaluation scores and one sample standard deviation across the three runs. Table 7 provides the values and query counts.

Dataset-Specific Adaptation. PROB's mean NDCG@20 on IC increases from 0.6019 to 0.6057 and 0.6116 as the adaptation fraction grows. Its score also rises on DC, from 0.8203 to 0.8252. NET leads DC and IC throughout, and PROB leads TF; their relative strengths therefore persist across the three query budgets. CJ shows a different pattern: PROB and CNS have similar means, with PROB leading at 25% and 100% and CNS at 50%. Increasing the adaptation fraction thus changes the branches differently across datasets.

DC reaches Recall@20 of one in every run, while NDCG still distinguishes the ordering of the recovered products. CNS's DC scores are identical across conditions; its CJ mean NDCG changes by less than 0.0001 across fractions. Validation selects the initial checkpoint in 51 of the 108 runs. This selection permits a learned starting point to remain in use when further optimization does not improve validation performance. The resulting curves characterize additional adaptation with fixed historical features and previously learned components.

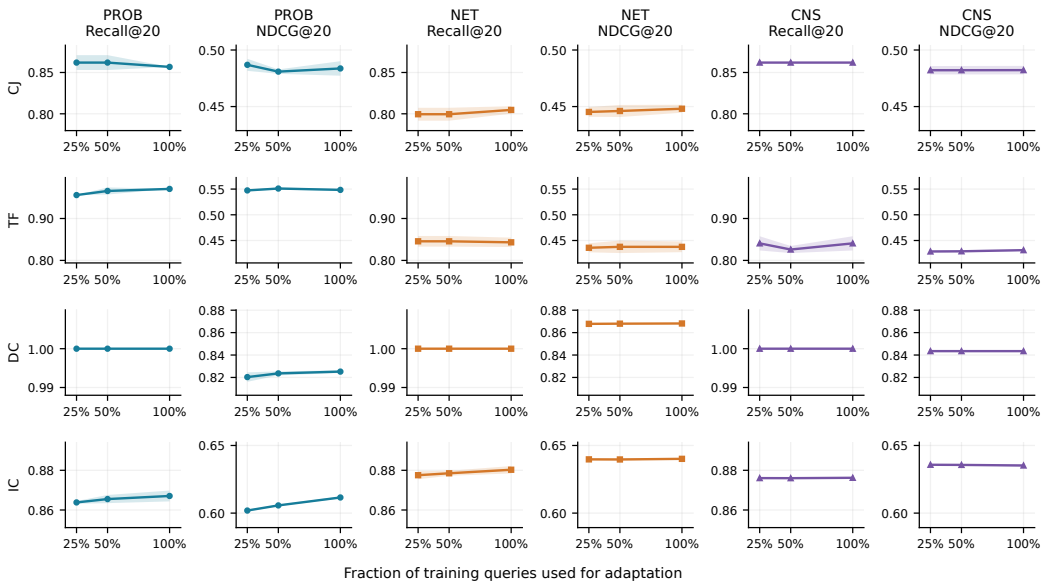


Figure 8: Recommendation performance across adaptation-query fractions. Rows show datasets; column pairs show PROB, NET, and CNS. Points summarize three sampling and optimization seeds from a common learned initialization; bands show one sample standard deviation. Recall@20 and NDCG@20 use common scales across branches within each dataset.

Table 7: Repeated adaptation results: mean ± sample standard deviation of three runs. Training-query counts at 25%/50%/100% are CJ 40/80/159; TF 21/42/84; DC 43/85/170; IC 1,422/2,843/5,686.

Dataset	Branch	Metric	25%	50%	100%
CJ	PROB	Recall@20 ↑	0.8620 ± 0.0090	0.8620 ± 0.0090	0.8568 ± 0.0000
		NDCG@20 ↑	0.4869 ± 0.0053	0.4809 ± 0.0020	0.4837 ± 0.0064
	NET	Recall@20 ↑	0.7995 ± 0.0078	0.7995 ± 0.0078	0.8047 ± 0.0045
		NDCG@20 ↑	0.4452 ± 0.0043	0.4461 ± 0.0053	0.4481 ± 0.0035
	CNS	Recall@20 ↑	0.8620 ± 0.0000	0.8620 ± 0.0000	0.8620 ± 0.0000
		NDCG@20 ↑	0.4821 ± 0.0037	0.4821 ± 0.0037	0.4822 ± 0.0038
TF	PROB	Recall@20 ↑	0.9559 ± 0.0000	0.9657 ± 0.0085	0.9706 ± 0.0000
		NDCG@20 ↑	0.5474 ± 0.0009	0.5511 ± 0.0022	0.5484 ± 0.0018
	NET	Recall@20 ↑	0.8456 ± 0.0127	0.8456 ± 0.0127	0.8431 ± 0.0112
		NDCG@20 ↑	0.4359 ± 0.0088	0.4377 ± 0.0122	0.4376 ± 0.0104
	CNS	Recall@20 ↑	0.8407 ± 0.0170	0.8260 ± 0.0085	0.8407 ± 0.0170
		NDCG@20 ↑	0.4287 ± 0.0021	0.4290 ± 0.0021	0.4312 ± 0.0014
DC	PROB	Recall@20 ↑	1.0000 ± 0.0000	1.0000 ± 0.0000	1.0000 ± 0.0000
		NDCG@20 ↑	0.8203 ± 0.0042	0.8236 ± 0.0018	0.8252 ± 0.0012
	NET	Recall@20 ↑	1.0000 ± 0.0000	1.0000 ± 0.0000	1.0000 ± 0.0000
		NDCG@20 ↑	0.8678 ± 0.0003	0.8680 ± 0.0003	0.8681 ± 0.0000
	CNS	Recall@20 ↑	1.0000 ± 0.0000	1.0000 ± 0.0000	1.0000 ± 0.0000
		NDCG@20 ↑	0.8435 ± 0.0000	0.8435 ± 0.0000	0.8435 ± 0.0000
IC	PROB	Recall@20 ↑	0.8638 ± 0.0007	0.8655 ± 0.0021	0.8671 ± 0.0027
		NDCG@20 ↑	0.6019 ± 0.0006	0.6057 ± 0.0005	0.6116 ± 0.0008
	NET	Recall@20 ↑	0.8775 ± 0.0020	0.8785 ± 0.0013	0.8803 ± 0.0019
		NDCG@20 ↑	0.6397 ± 0.0006	0.6397 ± 0.0007	0.6401 ± 0.0002
	CNS	Recall@20 ↑	0.8761 ± 0.0001	0.8760 ± 0.0000	0.8762 ± 0.0000
		NDCG@20 ↑	0.6357 ± 0.0003	0.6356 ± 0.0002	0.6352 ± 0.0009