BEHAVIORAL CLONING FOR THE ACTION POLICY OF A CUSTOMER SIMULATOR: STRUCTURED SCORES, METRIC-AWARE DECISIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Simulating a customer journey requires predicting not only which product a shopper engages with, but whether they search, refine, navigate, convert, or stop interacting. Recent shopping simulators build this on language models, whose post-training targets helpful interaction rather than a cohort's action propensities, and task adaptation does not by itself guarantee recovering them—prior work reports homogenised, over-agreeable behavior when they stand in for people (Zhou et al., 2026; Sharma et al., 2024), a distortion that would land on the conversion and termination rates a simulator exists to produce. We factor a simulator step by the chain rule into behavior selection and semantic realisation, and ground the first in logged behavior by cloning. We isolate a journey-spanning action policy covering query formulation, item engagement, conversion, navigation and termination; 40.4% of test steps carry an action no standard item-ID next-item objective represents. We introduce STRUCTBC, a compact causally masked Transformer score model encoding product-slate structure, short-range temporal dependence, page-local state and multi-scale behavioral context—a composition that lifts a recurrent and an attentive backbone alike. Our corpus is $452{,}296$ real shopping episodes from $6{,}824$ shoppers, split customer-disjointly into train, validation and test, over a state audited channel by channel for time direction (the *past-only state*: every shopper-specific channel is prefix-valid, bar two disclosed non-shopper transforms). On it the 3.4M-parameter policy reaches 0.5065 macro-F1 and 59.0% top-1 accuracy over ten actions, against 0.2784 for the strongest count-based baseline on the same split. Over the nine directly observed actions the score model alone reaches 0.5375 under the same $\arg\max$ decoding, against 0.5278 for an adapted NARM encoder, and it also outperforms, under a fixed single-GPU compute envelope, a LoRA fine-tune of a 1.5B instruction model on our corpus and label. Under the same recipe extended to all ten actions, the LLM predicts termination on 0.15% of matched test steps against 4.61% observed—a prevalence error our own scorer reproduces under a comparably data-starved, one-epoch control. A nine-parameter metric-aware decision layer, fit post-hoc with no retraining of the score model, takes the complete policy to 0.5488 over the nine observed actions ($+0.0113$ of that from the layer), and contributes $+0.0152$ over all ten labels. It also cuts the total-variation distance between predicted and observed action distributions from 0.1055 to 0.0498. We evaluate the action-selection factor open-loop: the discrete policy of a customer simulator, composable with the semantic realisation we do not evaluate.

1 INTRODUCTION

A simulator of shopper behavior is no better than its action propensities: whatever it renders on screen, the conversion and session-termination rates it reports are determined by which actions it chooses, so those choices have to be grounded in what shoppers were observed to do. Producing a plausible session means answering two questions of different natures at each step: *which* action the shopper takes from a small closed set, and what that action contains. Language models are well suited to the second, and recent shopping simulators are built on that basis (Lu et al., 2026; Zhang

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.

et al., 2026; Wang et al., 2026; 2025). The first is a different matter: a post-trained model is a strong semantic model, but its training objective does not target a cohort's empirical action propensities, and task adaptation does not by itself guarantee recovering them; prior work reports agreeableness and behavioral homogenisation where such models stand in for populations (§2). A *directional* shift in high-level choices would distort the conversion and session-termination readouts a simulator exists to produce. The application motivating this is pre-launch triage of candidate customer experiences, where online A/B tests take weeks to read out and expose live customers to potentially harmful variants—and which additionally requires correct response to an unseen interface, a requirement grounded propensities are a prerequisite for rather than a substitute for.

We factor behavior selection from semantic realisation, and the logs supervise the first directly. Task-adapted LLM simulators learn action selection and realisation jointly. Writing a_t for the discrete behavioral action and z_t for its semantic realisation—the query text, the item chosen, the content of the step, degenerate where an action realises no content—the chain rule factors the step exactly, with no independence assumed: $p(a_t, z_t \mid s_{\leq t}) = p(a_t \mid s_{\leq t})\, p(z_t \mid a_t, s_{\leq t})$. The factorization is an identity; what it buys is that the two factors may be fit by different means. We model the first by behavioral cloning on observed customer actions and reserve a language model for the second. Only the first is evaluated here, which is the scope of every claim below (Figure 1a). A clickstream log records the customer's next action at every state, over a closed vocabulary of ten: nine are read directly off the recorded event, and one, ending the session, is derived from the session boundary. Every step therefore carries a label, which makes behavioral cloning (Bain & Sammut, 1995) the natural formulation: no preference to elicit, no reward to model, no rollouts. The supervision is also plentiful: $452{,}296$ episodes, nearly three orders of magnitude more than the annotated OPeRA corpus (Wang et al., 2026). The target is the action *type*, over a space spanning major stages of the journey: query formulation, item engagement, conversion, navigation and termination. So 40.4% of test steps carry an action not directed at an item, which standard item-ID next-item objectives do not represent (§3.1).

Approach and headline. The state is structured in ways a flat encoder discards. The products on a search page form a ranked slate whose members are meaningful only relative to one another, page type strongly structures the action distribution, and behavior carries dependence at several timescales at once. The action distribution itself spans 1.5% to 25.6% of steps, which interacts with how scores become discrete predictions. We therefore clone the policy as a structured state, a behavioral score model ℓ_θ over the ten actions, and an explicit mechanism converting scores into a discrete decision. On a state audited channel by channel for time direction (the *past-only state* throughout: every shopper-specific channel is prefix-valid, bar two disclosed non-shopper transforms, §3.2), the resulting 3.4M-parameter policy attains 0.5065 macro-F1 and 59.0% top-1 accuracy over ten actions against the same-dataset baseline hierarchy of §5.1, with no generative LLM in the action-selection loop. That is open-loop next-action prediction: every prediction is conditioned on real logged history and scored against the action the customer took, so this is a necessary component of a customer simulator and not a complete one: semantic realisation, closed-loop fidelity under the policy's own rollouts and counterfactual evaluation of interface changes are outside what we measure (§6).

Contributions. (1) A closed, coarse action space spanning major journey stages, cloned end to end. We formulate high-level customer behavior as behavioral cloning over the ten journey-spanning actions above (§3.1). Among the shopping-simulation and session-recommendation formulations compared in Table 3, shopping simulators bind the action type to a target or content string, or restrict to three types, and session-based recommendation labels the item (§2); we instead make query formulation, item engagement, conversion, navigation and termination first-class outcomes of one closed policy, and adapt their recipes rather than import their numbers (Appendix A.1).

(2) Structured inductive biases give a small gain, replicated at two seeds and across two sequence families. On a causally masked Transformer backbone we compose four established components, each addressing one structural property named above (§3.3): on-page product-slate structure, short-range temporal dependence, page-local state, and multi-scale behavioral context. The composition transfers across sequence families—$+0.0055$ on a matched attentive backbone, $+0.0058$ on a matched recurrent one—and the resulting policy outperforms the learned baselines evaluated on the same past-only state—a matched GRU, a bare Transformer and the adapted NARM encoder—and the compute-bounded fine-tuned LLM (§5.1, Appendix A.1). Most componentwise

effects lie within the measured seed-replication reference and the one that separates on ten actions is not stable over nine, so we claim the composition rather than individual modules (§5.2).

(3) Metric-aware action selection as an explicit layer. We separate behavioral scoring from the discrete decision, and show that a nine-parameter per-class logit offset fit post-hoc against validation macro-F1—no retraining of the score model—improves macro-F1 on held-out test on every one of the eighteen checkpoints we applied it to, by $+0.0105$ to $+0.0180$; on the reported policy it also moves the predicted action mix toward the observed one (§5.3).

2 Related Work

Supervision for behavior models. A recent line of work simulates shopping with LLMs, differing mainly in where the learning signal comes from: in-context prompting and supervised fine-tuning on real sessions (Lu et al., 2026), policy-gradient RL against a self-certainty reward over a three-action space (Zhang et al., 2026), annotated next-action datasets (Wang et al., 2026), and persona-conditioned agents trained by SFT then RL (Wang et al., 2025). We adopt their problem definition, but not their label: theirs binds fine-grained target or content to the action type, so published accuracies are not commensurable with a ten-way choice (Table 3), and we adapt the recipes to our corpus and our own label instead (Appendix A.1). This choice of supervision matters because generic post-training is not designed to preserve cohort-level action propensities, and task-specific adaptation does not by itself guarantee recovering them: Zhou et al. (2026) attribute a behavioral Sim2Real gap to post-training that pulls models toward a homogeneous, overly agreeable register, and the alignment literature traces that tendency to preference data favouring agreeing responses (Perez et al., 2023; Wei et al., 2023; Sharma et al., 2024; Shapira et al., 2026). Realising a chosen action in content is where language models are appropriate, and web-agent benchmarks (Yao et al., 2022; Zhou et al., 2024; Koh et al., 2024; Deng et al., 2023) supply machinery for that half, though they measure *goal-directed task completion* given an instruction rather than the unconditioned behavior we model. Behavioral cloning (Bain & Sammut, 1995) is the natural formulation for densely-labelled trajectories; its known limitation is that a cloned policy compounds its own errors once it drives the state distribution (Ross et al., 2011)—outside the open-loop task this literature evaluates.

Predicting the next action in a session. Session-based and sequential recommendation model a closely related sequential object—the next event in a logged interaction sequence—with recurrent (Hidasi et al., 2016), attention-over-history (Li et al., 2017) and self-attentive (Kang & McAuley, 2018; Sun et al., 2019) encoders over item histories. Two differences set our task apart. First, their label space is the catalogue of items to recommend, whereas ours is a small closed set of *behavioral* action types, so a decision to refine, add to cart or end the session is a first-class target rather than a by-product of which item is scored next. Second, they optimise a ranking objective decoded by top-k, where our imbalanced ten-way classification makes the operating point (§3.4) a design decision in its own right. Their item-ranking heads do not transfer to our schema, which carries no item identifiers—no embedding table, no ranking target—so we port their encoders onto our state: our recurrent and attentive rungs are the GRU4Rec and SASRec backbones (Hidasi et al., 2016; Kang & McAuley, 2018), with NARM (Li et al., 2017) alongside them (Appendix A.1). Learned recommendation simulators (Shi et al., 2019; Zhao et al., 2023) depend on a next-action policy of the kind we study. The constituent structural devices are likewise established—joint encoding of a candidate set (Cao et al., 2007; Ai et al., 2018), multi-scale temporal encoding (Chung et al., 2017), residual paths that bypass a recurrence (He et al., 2016; Srivastava et al., 2015), and feature-wise conditioning (Perez et al., 2018)—and our contribution is their composition into a journey-level behavioral policy and its controlled evaluation. On persona specifically, OPeRA reports that adding it "does not consistently improve" accuracy and "may introduce noise" (Wang et al., 2026), and Customer-R1 lists stronger persona integration as future work (Wang et al., 2025)—which is why we test it as an explicit axis (Appendix C.4).

From scores to decisions. It is classical that an F-measure is optimised at a threshold rather than at the $\arg\max$ that $0/1$ loss calls for: this is known for cost-sensitive classification (Elkan, 2001), for binary F-measure (Lipton et al., 2014), and for general non-decomposable metrics (Koyejo et al., 2014; Narasimhan et al., 2014)—though those consistency results are stated for the population conditional probabilities, so they motivate our rule rather than transferring a guarantee to it, and we

fit the offset empirically and claim no optimality (§3.4). Additive class-dependent logit offsets are close to the logit adjustment of Menon et al. (2021) for long-tail recognition, which derives its shift from the label prior rather than fitting it to a target metric; we claim no novelty in class offsets. In the behavior-simulation formulations compared here, the score-to-decision operating point is not an explicit modeling choice; we separate fitted scores from the point at which they are read (§5.3).

3 METHOD

3.1 PROBLEM FORMULATION

We model a shopping session as a sequence of (state, action) pairs and learn a policy $\pi_\theta(a_t \mid s_{\le t})$ over ten actions by behavioral cloning (Bain & Sammut, 1995), minimising cross-entropy against the logged label at every state—nine observed customer actions and one derived session-boundary label.

The tenth label. Nine of the ten actions are recorded interactions. The tenth, SESSION_END, is derived: it labels an episode's last state—which must reach three steps to qualify (§4)—and means no further recorded interaction from that customer within the 30-minute inactivity window, so it carries no claim about intent or purchases made elsewhere. We therefore repeat the backbone, structural-block and decision-layer comparisons over the nine *observed* actions alone; conclusions are unchanged (§5.2, Appendix D).

The action space spans the journey, not the item interactions. The ten actions cover a session end to end: arrival and query formulation (SEARCH, REFINE_SEARCH), item engagement (CLICK_PRODUCT, CLICK_RECOMMENDATION), conversion (ADD_TO_CART, PURCHASE_COMPLETE), navigation and exploration (GO_BACK, BROWSE, OTHER), and termination (SESSION_END). 40.4% of test steps (35.6% excluding the derived boundary) carry an action that is not directed at an item and so is not directly represented by a standard item-ID next-item objective (the LLM simulators of §2 bind action type to a target string, and one restricts to three action types (Zhang et al., 2026), and Table 3 names what each family models). The space is not exhaustive: our schema records no scroll or dwell, so within-page attention is absent from both state and action set.

The cloned policy has two design layers. We separate a *structured behavioral score model* $\ell_\theta(s_{\le t}) \in \mathbb{R}^{10}$, which sets what information determines the action scores (§3.2, §3.3), from a *metric-aware action decision*, which converts those scores into discrete behavioral decisions (§3.4):

$$\hat{a}_t = \arg\max_c \big(\ell_{\theta,c}(s_{\le t}) + b_c\big), \qquad b \in \mathbb{R}^{10}, \quad b_{\text{SEARCH}} \equiv 0. \tag{1}$$

STRUCTBC is the score model, Equation 1 with a fitted b the decision layer, and the reported policy the pair; we measure both (§5.2, §5.3).

The layer is defined by the metric it is fit against: $b^\star(M) = \arg\max_b M\big(y, \hat{y}(b)\big)$ for an evaluation metric M over a labelled sample. Macro-F1 is the instantiation we fit, defining the primary reported policy; Appendix E refits the same offsets against distribution match as an exploratory criterion-dependence check; they move.

Our objective is not the unmodified likelihood: because the action distribution spans 1.5% to 25.6% of steps, we apply inverse-frequency class weights (exponent 0.5), label smoothing 0.05 and an auxiliary next-page-category cross-entropy at $\lambda = 0.1$ (§4). The weighting keeps frequent classes from dominating the objective. It need not preserve observed action marginals, so we report total-variation distance between predicted and observed action distributions beside macro-F1 (§5.3).

3.2 WHAT GOES INTO THE STATE

LLM-based simulators represent the session as free text; we ask how far a compact structured state goes instead. The state s_t has four groups: **what is on the page** (o_t, the only page-type-routed group and the input to slate self-attention—for a search page, the visible slate of 32 ranked products with their commercial attributes); **what has happened so far this session** (c_t, shared across page types, split by time scale by the multi-scale encoder); **when it is happening** (elapsed time, dwell, cyclic calendar features, within c_t); and **what the session is for** (ρ_t, a 128-d embedding of the episode's first search query: a placeholder for the steps before that query appears, its embedding from then on).

Two properties matter downstream. First, we audited every channel for time direction, which hold-out performance cannot do. Two of its findings are repaired here: a category-match channel assembling its purchase-category set corpus-wide, removed from training and evaluation, and a purpose channel that was an episode constant, rebuilt as the step function above. Two are retained rather than repaired (Table 13); the larger is a per-product popularity scalar aggregated over the whole corpus window, held constant as a ranking proxy on the assumption that a product's standing among products is slow-moving over it. Appendix E defines both and prices the popularity one. Second, c_t *already carries several profile-like behavioral signals*—recent action mix, price level and browse-versus-buy tendency, through its rolling action statistics, counters and price features (Appendix E)—so the state is not fully persona-free even before a profile is attached, and the persona null persists once both category-match channels are removed (Appendix C.3, Appendix C.4).

Backbone. The page observation is encoded by a page-type-routed branch into $\mathrm{state_enc}_t \in \mathbb{R}^{d_s}$ (Appendix E), which is concatenated with the context and purpose encodings; an MLP maps the result to the model width $d = 256$. The sequence model is a 3-layer causally masked Transformer encoder (Vaswani et al., 2017) whose output h_t feeds an action head. We call this the *bare backbone*; everything in Section 3.3 is measured against it.

3.3 Structural Composition: the StructBC Score Model

A flat encoder over o_t does not preserve four kinds of structure present in a shopping state: the on-page product slate is a *set of mutually competing options*, not 416 independent numbers; the predictive content of a behavioral motif is short-range; some actions depend far more on the current page than the history, yet must survive a recurrence to reach the output; and the context vector mixes signals at different time scales. Modules (i)–(iv) below restore them in that order (Figure 1).

(i) Slate self-attention. The flat search-page MLP is replaced by a 2-layer Transformer encoder over the slate $\mathbf{Q} \in \mathbb{R}^{32\times 13}$, with a learned positional embedding encoding each product's *rank* and a mean-pool over the attended set. The inductive bias is *relative* evaluation: an item's appeal depends on what it is shown beside—a comparison a flat encoder must learn per feature and attention expresses directly—and rank enters positionally because position on a search page is itself a behavioral prior.

(ii) Windowed causal temporal attention. A second attention pass runs over the sequence model's own outputs, restricted to a causal window of $K = 8$ steps and fused through a projection, residual and layer norm. The motifs are short and sharp (search → refine → refine, or click → back → click), while a full-context Transformer spreads attention over the entire prefix; it is partially redundant with such a backbone by construction.

(iii) Page-state residual skip, and (iv) multi-scale context encoding. The raw state encoding is concatenated with the sequence output before the head, $[\,h_t \| \mathrm{state_enc}_t\,]$. The action distribution depends strongly on page type: ADD_TO_CART and REFINE_SEARCH are each concentrated on one page context, though neither is confined to it. Forcing that signal through three Transformer layers risks diluting it, so the standard skip (He et al., 2016; Srivastava et al., 2015) gives page-dependent actions a direct path to the logits. Separately, the 61-d context vector is not homogeneous: its leading coordinates describe the immediately preceding step, the middle band a short rolling history, the tail session-level aggregates. It is therefore encoded band-by-band rather than through one linear layer shared across signals differing in magnitude and volatility.

We treat the four components as one structural block. Each module is a known device (Cao et al., 2007; Ai et al., 2018; He et al., 2016; Chung et al., 2017) targeting one of the state properties above; what we introduce is the *composition*, measured on two sequence backbones in §5.2, so the block rather than the module is our unit of claim (selected from a controlled search, Appendix B).

3.4 The Decision Layer

Choosing the operating point for the reported metric. We use macro-F1 as the primary metric because it weights all ten actions equally despite their 1.5%–25.6% prevalence range, and report accuracy alongside it. That choice calls for a different rule: macro-F1 is *non-decomposable*, so the

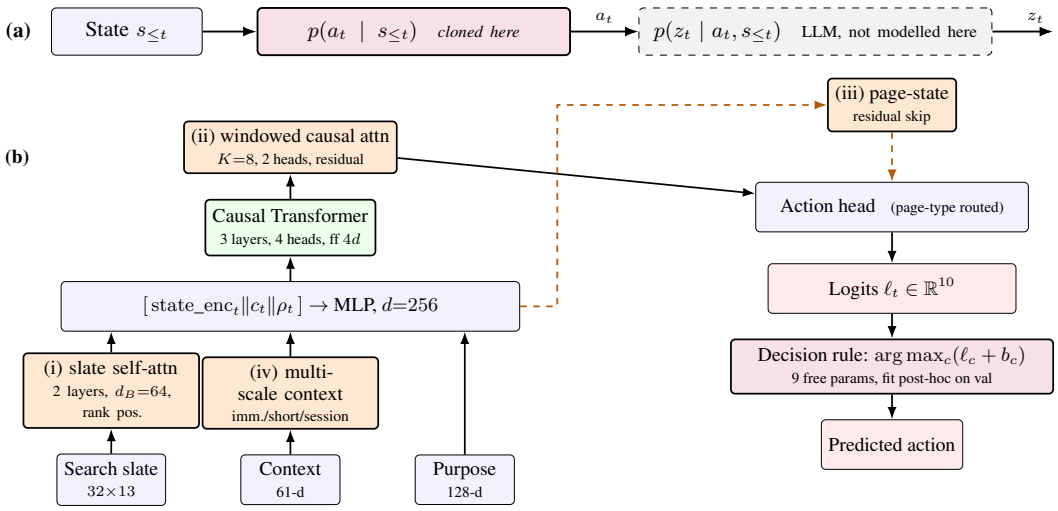


Figure 1: **(a)** Scope: the step factors into an action policy and a semantic realisation, and we fit the first from logs (§§3.1–3.2). **(b)** Inside the action policy: StructBC and the metric-aware decision rule. Solid arrows carry the state/history pathway; orange dashed arrows mark the page-state residual skip (iii). The decision rule (purple) is fit after training on cached validation logits.

optimal decision for one example depends on the predictions made for all others. On our reported checkpoint $\arg\max$ is conservative on low-prevalence actions—SESSION_END has much lower recall than precision (Table 11)—so the separation is in the scores and the rule reads them too cautiously.

Motivated by the literature on threshold adjustment for non-decomposable metrics (§2), the decision layer of Equation 1 takes the simplest form that can express this: a per-class additive offset in logit space, fit directly on validation macro-F1, leaving the scores untouched and moving only the decision boundary between classes. We pin $b_{\text{SEARCH}} = 0$ because only the *differences* between offsets affect the $\arg\max$; leaving all ten free adds a flat direction along which the fit wanders. We make no population-optimality claim: $b^\star$ maximises macro-F1 on the validation *sample*, with test used only to report the resulting policy. The contribution is empirical: this layer, fit this way, is worth a measurable gain on the metric it is fit against, at nine parameters and no retraining (§5.3).

We fit b by coordinate ascent on cached validation logits, maximising macro-F1 directly (grid and stopping rule in Appendix E); gradient optimisation through the $\arg\max$ is uninformative rather than merely hard, since macro-F1 is piecewise constant in b.

4 Experimental Setup

Dataset and protocol. We use de-identified clickstream logs from a large e-commerce platform, covering 6,897 customers before filtering, extracted between April 7 and June 5, 2026. Customers are sampled from strata defined by account tenure and purchase volume, filled to a target count per stratum, with no demographic or age criterion; the sample is therefore stratified by shopping activity rather than proportional to the platform population. Sessionization is two-stage: the raw event stream is cut into sessions at gaps over two hours, and each session is re-split into *active interaction bursts* at 30-minute inactivity gaps and wherever a burst would exceed four hours. A burst becomes an episode if it has at least three steps, opens on a search, browse or detail page, and resumes within five minutes at its first recorded gap: 452,296 of 676,107 bursts qualify, which is 66.9% of bursts but 85.9% of recorded interactions, since the three-step minimum removes almost only one- and two-step bursts. Seventy-three customers are removed outright, leaving 6,824; Appendix E attributes the loss to each criterion. Capped at 200 steps, these split into 363,562 training, 42,358 validation and 46,376 test episodes, yielding 745,260 scored validation and 958,146 scored test steps—a mean of 17.6 and 20.7 per episode. Splits are *customer-disjoint*, so every reported number is generalisation to unseen shoppers *in the same logged environment*—bar one input channel aggregated across shoppers, the corpus-wide popularity lookup (§3.2). The purpose embedding is PCA-reduced Sentence-BERT

Table 1: Same-dataset baseline hierarchy on the *past-only* state; score models fit on train, scored on held-out test (958,146 steps, 10 actions), under $\arg\max$ except the last, whose nine offsets are fit on validation. Count-based rungs condition only on page type and previous action, untouched by the channel removal. Neural rows use seed 42; across-seed means of these cells in Table 7.

Predictor	Conditions on	Macro-F1	Accuracy
Majority class	nothing	0.0408	0.2564
Page-type lookup	current page type	0.2519	0.4564
First-order Markov	previous action	0.2597	0.4647
Page × prev. action	both of the above	0.2784	0.4818
Matched GRU backbone	structured state, recurrent	0.4798	0.5745
Bare Transformer backbone	structured state, attentive	0.4890	0.5787
STRUCTBC (score model)	+ four structural modules	0.4913	0.5848
STRUCTBC + metric-aware decision rule	+ nine-offset decision layer	0.5065	0.5904

(Reimers & Gurevych, 2019), with the 128-d basis fit on train-split episodes alone (§3.2). All GRU and Transformer score models here share one optimisation protocol—AdamW, OneCycle, batch 128, up to 30 epochs, early stopping and checkpoint selection on validation macro-F1, class-weighted cross-entropy, and an auxiliary next-page-type cross-entropy at $\lambda = 0.1$ (Appendix E); every such model is multi-task, and we evaluate the action head (the LLM rungs follow a separate protocol, Appendix A.1). A cell takes 4.6–9.3 h on one NVIDIA A10G. Appendices B, C.4 and E hold the structural and capacity search, persona controls, and protocol details and excluded runs.

Metrics, the floor, and the seed-replication reference. We report macro-F1 (primary) and top-1 next-action accuracy, both *beside a majority-class floor*. The majority action is CLICK_RECOMMENDATION on both splits but at different rates—29.04% of validation steps against 25.64% of test steps—so a constant predictor scores 0.2564 accuracy and 0.0408 macro-F1 on test (analytically $2p/(1+p)/10$). Retraining matched configurations at seeds 42 and 43 gives the run-to-run spread we read deltas against, defined *per regime* because the regimes were measured separately: 0.0022 on the past-only state the headline comparisons use, 0.0015 on the reduced state of the auxiliary-loss control and Table 8's cells, and ±0.0019 on the original state of the earlier selection search (Appendices C.3 and B.2). The three are never mixed, and no claim rests on a difference smaller than its own regime's reference—cells inside it are marked null. All three are descriptive spreads, not confidence intervals; test-sample uncertainty is the customer-clustered bootstrap.

5 RESULTS

5.1 RQ1: HOW FAR LOGGED STATE GOES

The count-based rungs of Table 1 are exact parameter-free lookups over the immediate situation.

Page type alone carries about 60% of the full policy's accuracy gain over the majority predictor, so accuracy quoted against that floor overstates a model's contribution. But it is narrow: the strongest count-based rung scores exactly 0 F1 on REFINE_SEARCH, GO_BACK and SESSION_END, and is never the plurality in any (page, previous-action) cell. The learned gain is concentrated there: +0.228 macro-F1 over a page-and-history lookup rather than +0.466 over a constant, against +0.109 accuracy.

Prior recipes, adapted and run here. Table 2 trains four published recipes on our corpus, against our own label and on our own held-out partition; their reported accuracies are not commensurable with a nine-way choice (§2), but their recipes port. A 3.4M-parameter structured scorer—roughly 440× smaller—holds a predictive advantage over the adapted 1.5B fine-tune under the tested protocol and compute envelope. The ported session-based-recommendation encoders are strong controls: NARM (Li et al., 2017) lands within a ten-thousandth of our own matched GRU rung, so the gain we report is over a well-tuned recurrent scorer from that literature. On the rows both models scored, matching the LLM's data budget narrows the macro-F1 gap to +0.0205 ([+0.0106, +0.0308]) and reverses the accuracy ordering (−0.0396), both excluding zero—resolved in sign, not size (Appendix A.1).

Table 2: Prior recipes adapted to our corpus, **nine observed actions**, past-only state, same held-out test partition—the LLM on a uniform fixed-seed 2.7% draw of it, every other row on all of it; across-seed means where several seeds ran, arg max decoding except the last row. Compute-bounded, not data-matched: one A10G throughout, the LLM taking the larger share, 14.0 GPU-h against 4.6–9.3 h per StructBC cell. *Not* comparable with Table 1, which is the ten-action label space.

Method	Source of the recipe	Macro-F1	Accuracy
Majority class	—	0.0472	0.2694
Page type × prev. action	—	0.3192	0.5060
Fine-tuned LLM (Qwen2.5-1.5B, LoRA)	Lu et al. (2026), Wang et al. (2025)	0.4552	0.6018
GRU scorer	Hidasi et al. (2016)	0.5279	0.6032
NARM	Li et al. (2017)	0.5278	0.6038
Bare Transformer scorer	Kang & McAuley (2018)	0.5345	0.6097
StructBC	ours	0.5375	0.6137
StructBC + metric-aware decision rule	ours	0.5488	0.6293

The tested 1.5B fine-tune strongly under-predicts stopping. Repeating the fine-tune over all ten labels, so that stopping is available to it as an action, gives 0.4042 / 0.5772. Scored against StructBC on the *identical* 25,000 test steps (Appendix A.2), its SESSION_END F1 is 0.022 against 0.248 for ours, and it takes that action on 0.15% of steps where shoppers took it on 4.61%—a signed prevalence error of -0.0446 ($[-0.0490, -0.0406]$, clustered over 656 shoppers). Its largest surplus is CLICK_RECOMMENDATION (41.2% against 25.2%), and the deficit is not confined to the derived label: it also under-predicts GO_BACK, read directly off the recorded event, at 0.74% against 4.81%. This model therefore severely under-supplies both stopping and backtracking. Our own scorer under a matched data budget and epoch count under-supplies stopping more severely still—0.00% of the same steps at both seeds—and so does the rung's own un-post-trained sibling under its recipe (0.13%, Appendix A.2)—so the pattern is a fidelity failure we report, not evidence about alignment training. SESSION_END is a derived inactivity boundary (§3.1) and short visits are filtered, so these are rates within the retained-session population.

5.2 RQ2: The Structural Composition Transfers Across Sequence Families

Every cell below is trained and scored on the past-only state, replicated across seeds and read against the measured 0.0022 spread, with the full validation ladder in Table 5 (Appendix B.1). Each delta is a mean over the *two same-seed* comparisons, never a difference of best cells: at these magnitudes a cross-seed reference moves a number by more than the effect measured.

The same block lifts a recurrent and an attentive backbone alike. Applied to a matched GRU the four components are worth $+0.0058$ on validation; applied to the causally masked Transformer they are worth $+0.0055$. The two agree to within the 0.0022 replication reference, so the structural effect transfers across the two tested sequence families: the gain holds when the base sequence encoder is changed from self-attention to recurrence. The transfer holds on held-out test: $+0.0045$ on the attentive backbone and $+0.0059$ on the recurrent one, in across-seed means (test macro-F1, Table 7). Separately, at seed 42 the attentive gain is $+0.0023$ ($0.4890 \rightarrow 0.4913$, accuracy $+0.0061$) with a paired customer-clustered 95% interval of $[+0.0013, +0.0035]$ over 683 test customers; the interval belongs to that seed, not the means above. Swapping the recurrent backbone for the attentive one is worth $+0.0069$ alone ($+0.0079$, $+0.0058$), so the two choices we varied in the score model—sequence family and structural block—are of the same order in these runs, their difference below the 0.0022 reference. Both carry the corpus-wide popularity channel; without it the attentive gain is $+0.0005$ ($[-0.0004, +0.0014]$), which includes zero—the composition is not verified independently of it (Appendix E).

The aggregate gain is not attributable at finer granularity. We also measured the four components individually, two ways and replicated across seeds—*added-alone* onto the bare backbone and *leave-one-out* from StructBC—and only one (slate self-attention) clears the 0.0022 reference in either view, while three reverse sign between seeds in at least one column. We therefore pitch the claim at the composition (all twenty-four cells in Table 5, Appendix B.1).

Nine observed actions, same conclusions. Because SESSION_END is derived, not observed (§3.1), we retrained without it. This preserves the GRU $<$ bare Transformer $<$ STRUCTBC ordering at every seed and both aggregate signs (structural $+0.0027$, backbone $+0.0063$), while the one component-level effect that cleared the ten-action reference no longer separates reliably: -0.0021 against a 0.0018 reference, per-seed values straddling it (-0.0010, -0.0032)—aggregate claims survive the label we construct, the component-level one does not (Appendix D).

5.3 RQ3: METRIC-AWARE DECODING ADDS A GAIN WITHOUT RETRAINING

Nine free parameters, fit post-hoc, no retraining. All are fit on validation, test read only after fitting (all variants in Table 6). On the past-only state they are worth $+0.0152$ macro-F1 ($[+0.0131, +0.0170]$) on held-out test, fit in seconds. Within seed 42—the one cell where both carry paired customer-clustered intervals—that exceeds the structural block's $+0.0023$ ($[+0.0013, +0.0035]$) with non-overlapping intervals. We compare only there: the block's across-seed test mean is the larger $+0.0045$ and carries no interval (§5.2). The gain is consistent across the tested backbones, lifting the matched GRU to 0.4980 and the bare Transformer to 0.5039 (two-seed means, Table 7): crossing 0.50 here does not require the structural modules. Nor do the gains add: both move mass to the rare classes macro-F1 weights equally, so the block's nine-action margin falls from $+0.0030$ to $+0.0008$ once both are offset-decoded (Appendix C)—complementary axes, not summable gains.

On accuracy the two layers are comparable—$+0.0056$ for the offsets against $+0.0061$ for the modules—though the offsets are fit for macro-F1, not accuracy. Nor are the offsets fit to validation noise: fitting on one customer-disjoint half transfers to the other on all eighteen checkpoints ($+0.0105$ to $+0.0168$), moving the test result by at most 0.0009 (Appendix C). The fitted vector (Appendix E) suppresses what $\arg\max$ over-predicts and lifts what it under-predicts: on the reported cell (Table 11), SESSION_END recall rises from 0.150 to 0.282, PURCHASE precision from 0.617 to 0.730, and nine of the ten per-class F1 scores improve—macro-F1 weights every class equally and $0/1$ loss does not. The correction also moves the predicted action mix *toward* the observed one—total variation $0.1055 \rightarrow 0.0498$—though PURCHASE action prevalence remains under-predicted (0.050 against 0.056), as does ADD_TO_CART (0.077 against 0.085); these are marginal action frequencies over logged states (Appendix C). Three cheaper accounts—a page-type action mask fitted from train support, logit adjustment, and inverted training class weights—are scored on the same cached logits and none explains it (Table 7); nor does an original-state control under unweighted cross-entropy, where the offsets are worth *more* (Appendix C.2).

6 CONCLUSION

We learn the high-level layer of a customer simulator from labels the logs densely supply: nine targets are observed interactions, the tenth a session boundary derived from the log (§3.1), over $452{,}296$ episodes. On the reported state the score model's structure is productive as a block—$+0.0055$ on validation for the four components, transferring to a matched recurrent backbone ($+0.0058$) and to held-out test ($+0.0045$)—but not reliably attributable at finer granularity within our seed budget. And when the reported metric is non-decomposable, the operating point is a first-class design decision: nine offsets fit post-hoc on validation are worth $+0.0152$ on held-out test at no retraining cost (§5.3).

Limitations. Evaluation is open-loop on one corpus, and splits hold out customers, not treatments, so closed-loop fidelity, semantic realisation and counterfactual use are unmeasured, as is per-state calibration—the reported policy is deterministic, so our action-mix fidelity aggregates over logged states: this is a component of a simulator, not one. The offsets tune that argmax, not the probabilities behind it: termination stays over-supplied at 125% of the observed rate (Appendix A.2), which a closed-loop rollout could compound. The ≥ 3-step retention criterion under-represents immediate-bounce and short visits, so termination rates are conditional on the retained-session population. Seed replication follows the claims: the reported comparisons and structural sweep are replicated against a measured reference; most of the original selection search is single-seed. The persona null is specific to this state and corpus (Appendix C.4). Our experiments evaluate predictive fidelity, not the mechanism behind the LLM gap (Appendix A.2). We do not attribute the roughly 40% we misclassify to behavioral stochasticity, unobserved state or a limit of this representation: separating them needs repeated observations of one shopper in one state, which our corpus lacks.

Reproducibility Statement

The model, training, and evaluation procedures are specified in the paper and this appendix. Exact numerical reproduction additionally requires the proprietary clickstream corpus and the derived per-product popularity artefact, whose scale our documented form does not recover (Appendix E). The state construction is described in Section 3.2 and its exact dimensions, per-group feature lists and action indices in Appendix E. All architectural modules are defined in Section 3.3, with every hyperparameter—optimiser, schedule, batch size, class weighting, auxiliary loss weight, early-stopping criterion, module widths and depths, and seeds—in the "Hyperparameters" paragraph of Appendix E, along with the wall-clock cost of a single training cell. The decision rule is given in closed form in Section 3.4, its fitted nine-parameter vector in Section 5.3, and its full fitting protocol—grid, round limit, convergence criterion, and the val-A/val-B confirmation split—in Appendix E. Implementation details a reimplementation will otherwise get wrong are documented deliberately: the class of positional feature that is not computable at serving time is identified together with the assertion that detects it; the category-match channel, whose purchase-history input is not strictly past-only, is removed from training and evaluation, with a whole-channel sensitivity analysis and a re-evaluation of the headline comparisons without it; and the purpose channel is built as a step function that carries the empty-query embedding until the query is observed, on a basis fit on training customers alone—with the exposure the superseded episode-level construction would have carried counted in full; and the per-product popularity channel is accumulated over the whole corpus window rather than over each step's past, and is retained as a population-level ranking proxy rather than repaired, as is the promotion counters' min–max rescaling over a calendar span fixed in the generator—all of them in Appendix E. Our evaluation protocol reports a majority-class floor beside macro-F1 and accuracy (Section 4) and a measured seed-replication reference, so any replication attempt can tell agreement from coincidence.

Ethics Statement

The clickstream logs underlying this work are de-identified before any modelling: records are keyed by anonymised identifiers with no name, address, or payment instrument, and we neither attempt nor require re-identification. One customer-authored free-text field does enter the model: the search query the shopper has issued by the current step, which the state encodes as a fixed 128-dimensional PCA-reduced sentence embedding, the empty-query embedding until one is issued (Section 3.2). The raw query string is used only to compute that vector and is neither stored with the training tensors nor reported, and no query appears anywhere in the paper. Supervised trajectories and labels are customer-disjoint; the disclosed popularity lookup is the only behavioral aggregate constructed across splits. We report only aggregate statistics; no individual session or purchase appears in the paper. A potential downstream use of a policy of this kind is pre-launch triage of candidate interfaces, which could reduce the number of real customers exposed to weak variants; that use requires the closed-loop and counterfactual validity we do not measure here (Section 6). We note the corresponding risk: a behavioral simulator can in principle be used to optimise an interface against predicted customer responses in ways that serve the operator rather than the shopper. The original architecture-search state included a category-match feature derived from prior purchase history (Section 3.2). After we identified that its construction is not strictly past-only, our final reported performance is obtained with both category-match channels removed from training and evaluation (Appendix C.3); the original state appears only in explicitly labelled exploratory capacity/hyperparameter and persona analyses, and is labelled as such wherever its numbers appear. The reported policy therefore consumes neither an explicit persona embedding nor a purchase-history-derived channel. Finally, the model is trained on and may inherit correlations present in the logged behavior and the ranking systems that generated it—the observed slates are the output of a recommender—so it should not be read as a model of shopper preference independent of the platform that shaped it.

The Use of Large Language Models

We used generative AI tools for code assistance and debugging, manuscript drafting and editing, literature search and summarisation, and feedback on problem framing, experimental controls, and interpretation. All research decisions, data processing, experiments, and reported numerical results

were executed and independently verified by the authors. AI-generated suggestions were reviewed critically and were not treated as empirical evidence. The authors take full responsibility for the final paper.

REFERENCES

Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. Learning a deep listwise context model for ranking refinement. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2018.

Michael Bain and Claude Sammut. A framework for behavioural cloning. *Machine Intelligence 15*, 1995.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. In *International Conference on Machine Learning (ICML)*, 2007.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Xiang Deng et al. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Charles Elkan. The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 973–978, 2001.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2016.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining (ICDM)*, 2018.

Jing Yu Koh et al. Visualwebarena: Evaluating multimodal agents on realistic visually grounded web tasks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Oluwasanmi Koyejo, Nagarajan Natarajan, Pradeep Ravikumar, and Inderjit S. Dhillon. Consistent binary classification with generalized performance metrics. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2017.

Zachary C. Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. Optimal thresholding of classifiers to maximize f1 measure. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, 2014.

Yuxuan Lu, Jing Huang, Yan Han, Bingsheng Yao, Sisong Bei, Yaochen Xie, Yisi Sang, Qi He, and Dakuo Wang. Can LLM agents simulate multi-turn human behavior? evidence from real online customer behavior data. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43961–43977, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.2034. URL https://aclanthology.org/2026.acl-long.2034/.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *International Conference on Learning Representations (ICLR)*, 2021.

Harikrishna Narasimhan, Rohit Vaish, and Shivani Agarwal. On the statistical consistency of plug-in classifiers for non-decomposable performance measures. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, 2018.

Ethan Perez, Sam Ringer, Kamilė Lukošiūtė, Karina Nguyen, Edwin Chen, et al. Discovering language model behaviors with model-written evaluations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13387–13434, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.847.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Itai Shapira, Gerdus Benade, and Ariel D. Procaccia. How rlhf amplifies sycophancy. *arXiv preprint arXiv:2602.01002*, 2026.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2019.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *ACM International Conference on Information and Knowledge Management (CIKM)*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Jing Huang, and Dakuo Wang. Customer-r1: Personalized simulation of human behaviors via rl-based llm agent in online shopping. *arXiv preprint arXiv:2510.07230*, 2025.

Ziyi Wang, Yuxuan Lu, Wenbo Li, et al. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. arXiv:2506.05606.

Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou, and Quoc V. Le. Simple synthetic data reduces sycophancy in large language models. *arXiv preprint arXiv:2308.03958*, 2023.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Yimeng Zhang et al. Shop-r1: Rewarding llms to simulate human behavior in online shopping via reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2507.17842.

Kesen Zhao et al. Kuaisim: A comprehensive simulator for recommender systems. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Shuyan Zhou et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Xuhui Zhou, Weiwei Sun, Weihua Du, Jiarui Liu, Haojia Sun, Qianou Ma, Tongshuang Wu, Yiming Yang, and Maarten Sap. Odyssim: Building foundation models for human behavior simulation. *arXiv preprint arXiv:2606.14199*, 2026.

Rounding convention. Deltas, ratios, and derived quantities throughout the paper are computed from full-precision metrics; displayed cell values are rounded. Recomputing a difference from two rounded cells may therefore disagree with the printed difference in its last digit.

A Prior Formulations in Detail

Section 2 groups prior work by methodological question and compresses the individual formulations. This appendix states them, because the differences in corpus size and prediction target are what rule out a direct numeric comparison (Table 3).

Language-model simulators of shopping behavior. Lu et al. (2026) provide the first large-scale accuracy (rather than believability) evaluation, on 31,865 real sessions, conditioning on interaction history only, and compare prompting against supervised fine-tuning. Shop-R1 (Zhang et al., 2026) adds policy-gradient RL against a self-certainty and difficulty-aware reward over a three-action space (CLICK/TYPE/TERMINATE). OPeRA (Wang et al., 2026) contributes a persona-, observation-, rationale-, action-annotated dataset (51 effective participants, 692 sessions) and formalises next-action prediction as an evaluation task. Customer-R1 (Wang et al., 2025) conditions a 7B-parameter agent on a persona block, first by supervised fine-tuning and then by RL, on the 49-user OPeRA subset. These works establish the task and its evaluation and we adopt their problem definition. They differ from ours in what they predict as well as how they learn it: their action representations include fine-grained target/content information in addition to the high-level action type, whereas we predict a ten-way high-level choice over 452,296 episodes—nearly three orders of magnitude more sessions than the annotated datasets above.

Web agents and learned simulators. WebShop (Yao et al., 2022), WebArena (Zhou et al., 2024), VisualWebArena (Koh et al., 2024) and Mind2Web (Deng et al., 2023) evaluate agents on completing web tasks and consistently report large human–agent gaps. Those benchmarks measure *goal-directed task completion* given an instruction, whereas we model *unconditioned customer behavior*—what a shopper would freely do next, with no task assigned. On the simulator side, Virtual-Taobao (Shi et al., 2019) learns a customer simulator adversarially and demonstrates real-world policy transfer, and KuaiSim (Zhao et al., 2023) builds a multi-behavior cross-session recommender simulator; both depend on a next-action policy of the kind we study, which is what makes its estimation worth isolating.

Table 3: Where methods sit in the decomposition of §1: the high-level policy decides which action occurs, semantic realisation its content. Prior LLM-based work jointly represents action type and fine-grained target/content; we isolate the former as a ten-way policy and evaluate it against logged action labels. *Published* accuracies are not listed because a joint type-and-target accuracy is not commensurable with a ten-way choice; the last column says whether we adapted the recipe to our corpus, and Table 2 gives those numbers.

Method family	Behavioral supervision	Policy / realisation	Adapted here
Prompted LLM[a]	none (in-context)	LLM / LLM	no
Fine-tuned LLM[b]	demonstrations	LLM / LLM	**yes**
RL-trained LLM[c]	task-specific reward signal	LLM / LLM	no
Session-based rec.[d]	next-item click	scorer / —	**yes**
Structured BC (StructBC)	9 logged +1 derived label	StructBC + b / downstream	**yes**

[a] Lu et al. (2026), Wang et al. (2026); [b] Lu et al. (2026), Wang et al. (2025);
[c] Zhang et al. (2026), Wang et al. (2025); [d] Li et al. (2017), Hidasi et al. (2016), Kang & McAuley (2018)

A.1 Prior methods adapted to our corpus

Prior shopping simulators report accuracy over a label that binds action type to a target or content string, so their published numbers cannot be placed beside a nine-way choice. Their *recipes* can: Table 2 trains each on our corpus, against our label, and scores it on the same held-out test partition. Two facts come out of it. The first is one of scale—a fine-tuned 1.5B model reaches 0.4552 macro-F1

against 0.5375 for a 3.4M-parameter score model under the same $\arg\max$ decoding (0.5488 once the decision layer is applied) —so a compact structured scorer over a closed action set has a predictive advantage over this adapted LLM recipe under the tested protocol and compute envelope. That advantage is not read off the difference of those two cells, which are not scored on the same rows—the rung's figure is a 2.7% draw of the partition and ours is all of it. It is measured directly in the budget-matched control below, paired on the $25{,}000$ rows both models scored. And the session-based recommendation backbones are competitive: NARM lands at 0.5278, numerically indistinguishable from our own matched GRU rung, and on the ten-action state NARM's two-seed mean, 0.4798, agrees to four decimals with that rung's seed-42 test score, 0.4798—so the gain we report is over a well-tuned recurrent scorer from that literature, not over a strawman.

Residual asymmetries, in both directions. The remaining asymmetries do not run one way: the LLM has external pretraining and a free-text representation, while its serialisation itemises only the top 8 of up to 32 recorded slate positions—the remainder as a count and price range—and the last 8 actions, so some fields the structured scorer reads are absent from the prompt. No prompt is truncated (longest 779 of 1024 tokens), and no product or query text is available to either arm: the corpus carries no titles, brands, category names or query strings, the prompt says so explicitly, and the purpose channel the structured scorer reads is a 128-d projection of a query embedding rather than the query itself. Its decoding is otherwise unconstrained. Length-normalised scoring was specified before evaluation; for completeness the summed-logprob alternative is *higher* (0.4692 / 0.6138).

The recommendation rungs are the published *backbones*, not the published models. GRU4Rec, NARM, SASRec and BERT4Rec (Hidasi et al., 2016; Li et al., 2017; Kang & McAuley, 2018; Sun et al., 2019) all score over an item-ID vocabulary and decode by top-k ranking; our state schema carries no item identifiers, so the item-embedding table and the ranking objective have nothing to attach to. We therefore port the sequence encoder—GRU (Cho et al., 2014), attention-over-history, self-attention—onto our state and decode a nine-way action choice, which is the strongest form of the comparison the two problem definitions have in common. All three ported encoders are causal; BERT4Rec's is bidirectional over a masked-item objective, so it has no counterpart among the rungs.

The fine-tuned LLM rung, in full. Qwen2.5-1.5B-Instruct with LoRA ($r = 16$, $\alpha = 32$, dropout 0.05, no bias term) on the attention and feed-forward projections (`q`, `k`, `v`, `o`, `gate`, `up`, `down`); AdamW at 10^{-4} with a cosine schedule and 3% warmup; micro-batch 4 at gradient accumulation 8 (effective batch 32), one epoch, gradient checkpointing on, sequence length 1024 tokens with *left* truncation so the preamble naming the action set is the part dropped last, seed 42. Each state is serialised as text—page type, the top 8 slate positions with price, rank and rating (of up to 32 recorded, the remainder as a count and price range), the detail-page block, cart and session context, and the last 8 actions—and the model scores each candidate action string, length-normalised, over that prompt. Within the 14.0-hour envelope that one epoch covers 3.0% of the training steps, where the score-model rows take four epochs over the full split; evaluation is the uniform fixed-seed 2.7% draw of Table 2's caption. Both subsamples are drawn with NumPy's PCG64 generator seeded at 42. The training subset retains each step independently with probability equal to the requested count over the split total, so its realised size varies around the request instead of matching it: $199{,}820$ steps for the nine-action run and $199{,}624$ for the ten-action one. The evaluation draw is $25{,}000$ row indices taken without replacement and is therefore of exact size. The matched control of the next paragraph uses the same generator and seed.

A unique-data-budget-matched control. We match access to unique logged examples, not optimizer updates or epochs. The data asymmetry is the one that runs in our favour, so we measure it instead of arguing about it. We retrained StructBC on the LLM rung's own data budget—2.99% of nine-action training steps against the LLM's 2.97%—and scored it on the same full test partition as every other score-model row. It reaches 0.4740 macro-F1 and 0.5596 accuracy there. Those two figures describe the full partition, and the rung's does not, so we do not subtract them. Instead we score both arms on the rows the rung itself was scored on. The fine-tuned LLM was re-run to dump per-row predictions, which lets the difference be bootstrapped within one measurement condition: over $25{,}000$ shared rows and 657 resampled customers, the matched-budget model reaches 0.4757 against the rung's 0.4552, a paired difference of $+0.0205$ ($[+0.0106, +0.0308]$), while on accuracy it is behind at -0.0396 ($[-0.0477, -0.0322]$). Both intervals exclude zero, so the ordering reversal is a property of one population rather than an artefact of two draws. We read this as resolved in *sign* only:

the macro-F1 interval's width is 98% of the point estimate it brackets, which bounds the direction of the matched-budget gap without bounding its size. The same rows let the reported cell be compared on the same terms: StructBC reaches 0.5380 there, a paired $+0.0828$ ($[+0.0732, +0.0935]$) on macro-F1 and $+0.0126$ ($[+0.0061, +0.0187]$) on accuracy, so at full data it leads the rung on both measures and no reversal remains. We still stop short of reading the matched-budget margin as a measure of what the design is worth. The two cells differ in unique-data coverage *and* in training regime, and the interval that brackets their difference is nearly as wide as the difference itself. What the control establishes is narrower and worth stating plainly: on these shared rows the full-data gap of $+0.0828$ shrinks to $+0.0205$ under matched unique-data coverage, and at that budget the macro-F1 ordering favours StructBC while the accuracy ordering favours the rung.

Moving from the unique-data-matched regime to the full-data training regime, against the seed-42 full-data checkpoint, adds $+0.0625$ macro-F1, $[+0.0587, +0.0671]$, clustered over the same 683 customers. Coverage and schedule both change between those two cells, so this prices the move between regimes and does not isolate the contribution of data quantity. Read operationally it is the point of the design: a 3.4M-parameter model can exploit the full corpus within the stated compute envelope, where the LLM reaches 3.0% of it. The matched cell takes 30 epochs over the reduced split rather than one, because a single pass at our effective batch is 84 optimizer updates against the LLM's $\sim$6,244; the 30-epoch cell scores $+0.2647$ macro-F1 *above* a single pass, which reaches only 0.2093, so the one-pass cell is update-starved and is not the control. A fine-tune differing from the nine-action cell only in its label set—all ten, so termination is a predictable action rather than a derived boundary—evaluated on a uniform fixed-seed 2.6% draw, reaches 0.4042 / 0.5772. The nine- and ten-action cells are drawn from the same test partition but not from the same sample of it, so we read each against StructBC and not against each other; the ten-action cell's per-class comparison against StructBC is made on matched rows in §A.2.

A.2 Matched-Row Action-Propensity Diagnostic

A low SESSION_END F1 is symmetric: it is equally consistent with predicting that action far too often and with almost never predicting it. To recover the direction, we re-score the ten-action fine-tune with its per-step predictions dumped, and score StructBC on exactly those $25{,}000$ steps. Table 4 reports the resulting action mixes. Intervals in the text are percentile bootstraps clustered on the shopper (656 shoppers, $2{,}000$ paired draws), since steps within a session are not independent.

The LLM's distortion is concentrated and directional. It predicts SESSION_END on 0.15% of steps against an observed 4.61% (signed error -0.0446, $[-0.0490, -0.0406]$); its largest surplus is CLICK_RECOMMENDATION, at 41.2% against 25.2%. Its mix sits 0.216 in total variation from the observed one, against 0.106 for StructBC under $\arg\max$ and 0.052 under the decision layer; the paired LLM-minus-decision-layer difference is $+0.164$ $[+0.154, +0.174]$. On these same rows the SESSION_END F1 ordering is 0.022 / 0.199 / 0.248. Under-prediction of stopping is not unique to the LLM: our own $\arg\max$ decoder also under-predicts this rare class, at -2.14 points. The distortion is nonetheless much larger for the LLM—it predicts SESSION_END at 3% of the observed rate against 54% for StructBC under $\arg\max$—and the paired difference on these same rows is -0.0233 ($[-0.0267, -0.0201]$), so the gap survives the artefact rather than consisting of it. What the table cannot separate is *why*: a lightly fine-tuned model on a rare label would look similar, and we therefore report the behaviour and not its cause.

The fitted offsets reverse the sign of this error but reduce its absolute magnitude, from 2.14 to 1.14 points, while improving SESSION_END F1 (0.248 against 0.199). With the offsets StructBC over-predicts termination, at 125% of the observed rate; both intervals exclude zero, so the over-supply is real and not noise. Over-prediction is still error, and we report it as one.

A budget- and epoch-matched control on the same rows. The LLM's under-supply of stopping invites a reading in terms of what post-training does to a model, and that reading does not survive a control for how little data the cell saw. The fine-tuned rung is trained on 3.0% of the unique training steps for one pass; we retrained StructBC at that same budget—2.8132% of unique steps against the rung's 2.8159%, one epoch—and scored it on these identical $25{,}000$ rows. Under $\arg\max$ it predicts SESSION_END on 0.00% of them, at both seeds, for a SESSION_END F1 of 0.0000: a starved copy of our own architecture under-supplies stopping *more* severely than the LLM's 0.15%, not less. A second cell tests whether the rung's unweighted objective is required. Matching that same budget

Table 4: Action prevalence on the 25,000 test steps for which the ten-action LLM cell dumped predictions, with StructBC scored on the identical rows under both decodings. Cells are percentages of steps, so each model's signed prevalence error for an action is its cell minus the observed cell on the same line, up to the rounding of both—the errors quoted in the text are computed before rounding. All four columns are one measurement condition and may be compared with each other, but not with Table 11, which is the full test partition.

			StructBC	
Action	observed	LLM	arg max	+ metric-aware decision rule
SEARCH	23.89	17.02	19.54	20.38
REFINE_SEARCH	3.88	2.54	7.34	4.85
CLICK_PRODUCT	20.03	25.69	16.42	20.68
CLICK_RECOMMENDATION	25.23	41.18	29.36	27.28
GO_BACK	4.81	0.74	4.29	4.58
ADD_TO_CART	8.43	6.18	8.58	7.77
PURCHASE_COMPLETE	5.75	5.11	6.52	4.96
BROWSE	1.40	1.02	2.30	1.70
OTHER	1.97	0.39	3.18	2.04
SESSION_END	4.61	0.15	2.48	5.76
SESSION_END F1	—	0.022	0.199	0.248
total variation	—	0.216	0.106	0.052
macro-F1	—	0.4042	0.4883	0.5054

and epoch count while retaining our inverse-frequency weighting gives 0.00% as well, so the same low-data, one-epoch regime still produces the collapse under our weighted objective: the unweighted objective is not required for it. The converse cell completes the pair and separates: on the full training split with our weighting removed, SESSION_END is predicted on 0.65% of these rows for an F1 of 0.100, against the reported checkpoint's 2.48% above. Removing the weighting therefore cuts the predicted share to roughly a quarter without collapsing it—the two budget-matched cells never emit the class at all, and this one does—so we attribute the collapse to the low-resource training regime rather than to the unweighted objective. These controls do not separate unique-data quantity from the number of optimizer updates: at one epoch the two move together (§A.1). A control within the LLM's own family settles by measurement what the starved copy of our architecture could only suggest by analogy. We fine-tuned the *un*-post-trained sibling of the rung's checkpoint—identical architecture and vocabulary, the same LoRA recipe and training examples, one epoch, scored on these same 25,000 rows, with the presence of post-training the only difference—and it predicts SESSION_END on 0.13% of them for a macro-F1 of 0.4071, against the rung's 0.15% and 0.4042. Removing post-training leaves the prevalence error where it was: at these counts the two checkpoints are indistinguishable on this class, and neither comes near the 4.61% observed. We therefore report the LLM's prevalence error as a fidelity failure and draw no inference from it about alignment training. Both arms are one-epoch cells at that same small budget, so this control widens the range of checkpoints over which the starvation confound holds rather than removing it, and it is one seed of one model family. The collapse is also a property of the decoder and not only of the checkpoint: the same starved weights under offsets fit on their own validation split recover SESSION_END at 6.03% and 0.89% at the two seeds, against 4.61% observed. That is a wide spread on a rare class from a one-epoch fit, and we read it only as showing the layer can repair what the starved decoder loses.

B Structural, Capacity, and Hyperparameter Sweeps

This appendix has two halves, and they are *not* on the same footing. §B.1 is confirmatory: every cell is trained *and* evaluated on the past-only state at every seed, and it is the evidence behind §5.2. §B.2 is exploratory: it is the capacity and hyperparameter part of the original selection search, run on the *original* full state at seed 42 only, and it predates the removal of the future-leaking channel (Appendix C.3). **Cells from the two halves must not be differenced.** A number from §B.2 minus a number from §B.1 measures the difference between two state constructions plus one unreplicated seed draw, not the effect being named.

B.1 CONFIRMATORY STRUCTURAL MATRIX: PAST-ONLY STATE, REPLICATED SEEDS

Table 5 is the complete two-view structural sweep behind §5.2: sixteen component-ablation cells—four components × two views × two seeds—plus eight aggregate and reference cells, all on one state, with every delta taken against the same-seed reference.

Table 5: The complete structural sweep, validation macro-F1, every cell trained *and* evaluated on the reported past-only state at every seed. "Added alone" adds one component to the bare Transformer; "leave-one-out" removes one from STRUCTBC. Each Δ is the mean over seeds of the per-seed difference from the *same-seed* reference. Deltas inside the 0.0022 two-seed reference (the spread of STRUCTBC itself on this state) are greyed, and † marks the four whose sign is not consistent across seeds. Exactly one component clears the reference, and it does so symmetrically in both columns—so the sweep licenses the aggregate row, not the per-component ones.

	Added alone			Leave-one-out		
Component	s42	s43	Δ added	s42	s43	Δ removed
(i) slate self-attention	0.4720	0.4720	+0.0042	0.4702	0.4713	−0.0025
(ii) windowed temporal attn	0.4695	0.4689	+0.0015	0.4723	0.4728	−0.0007†
(iii) page-state skip	0.4676	0.4687	+0.0004†	0.4739	0.4732	+0.0003†
(iv) multi-scale context	0.4701	0.4694	+0.0020	0.4730	0.4716	−0.0009†
References and the aggregate ladder (validation macro-F1, same past-only state, the same seeds)						
matched GRU, bare	0.4609	0.4609	—			
matched GRU + all four	0.4676	0.4659	+0.0058			
bare Transformer	0.4688	0.4668	+0.0069 *vs. GRU*			
STRUCTBC (all four)	0.4722	0.4744	+0.0055 *vs. bare*			

Two things follow from the table directly. The leave-one-out column is not a capacity ablation in disguise—removing slate attention *raises* the parameter count, for the reason given in the caption of Table 8—and the four components sum to +0.0080 added alone against the +0.0055 they deliver jointly. That gap is not resolvable, and the reason is worth stating: every added-alone delta is taken against the same bare cell, so the sum inherits four copies of that cell's own seed spread. Per seed the four sum to +0.0042 and +0.0119 against aggregates of +0.0034 and +0.0076—so the gap is −0.0008 at one seed and −0.0043 at the other, and we can resolve neither super- nor sub-additivity at this scale.

The structural block transfers to a recurrent backbone. Fitting the same four components onto the matched GRU raises its validation macro-F1 from 0.4609 to 0.4667 (+0.0058, mean over seeds on the past-only state), matching the +0.0055 they buy on the Transformer to within the 0.0022 reference. The block therefore transfers across both families we tested rather than depending on the attentive one. Neither GRU leg was tuned, and the 0.0003 difference between the two gains is far inside that reference, so we read neither an ordering nor a redundancy story off it.

B.2 EXPLORATORY CAPACITY AND HYPERPARAMETER SEARCH: ORIGINAL STATE, SEED 42

This is the capacity and hyperparameter part of the search that *selected* the reported configuration. It ran on the original full state at seed 42 before the future-leaking channel was identified, and it was not repeated afterwards, so its cells are **not** comparable to §B.1's and its deltas are read against the legacy ±0.0019 single-state spread of that search, not against 0.0022. We swept backbone depth (2–6 layers) and width (128–512), spanning 1.08M to 11.7M parameters, and the module knobs: slate-attention width (32/64/128) and depth (1/2/3), the feed-forward multiplier (2/4/8), head count (4/8), and dropout (0.1/0.2). Two things came out of it, both as model-selection findings on that state and seed. Nothing cleared that search's own reference on the upside across a 1.96× parameter range around the anchor (0.4764 validation macro-F1 at 3.25M parameters), while the downside was not bounded—width 128 cost −0.0101—which is why the anchor's scale was carried forward rather than grown. Neither observation is evidence about the reported past-only state, on which no capacity sweep was run; §B.1's leave-one-out arms are shown not to be capacity ablations by their own parameter counts instead. The anchor also carried a hierarchical action head, dropped because

its removal sat inside the seed-variation reference and the routed head replacing it is simpler. Final hyperparameters are in §E.

C Full Result Tables

C.1 The Decision Rule Across Architectures and Actions

Table 6 reports the variants of the rule itself on the reported checkpoint, all fit on validation with test read only after fitting. Neither richer parameterisation buys anything, and for the temperature that is an identity rather than a finding: under $\arg\max$, $\arg\max_c(\ell_c/T + b_c) = \arg\max_c(\ell_c + Tb_c)$ for any $T > 0$, and the pin survives the reparameterisation because $T \cdot 0 = 0$. A scalar temperature composed with per-class offsets therefore selects exactly the actions the offsets Tb select on their own, and adds *no* degree of freedom to the decision: the ten-parameter row of Table 6 is the nine-parameter family written twice. What it does change is the search geometry, and that is the whole of its $+0.0002$—the offsets are fit on the rescaled logits over the same 0.125 grid (§E), so $T{=}0.625$ places the equivalent original-scale offsets on multiples of 0.078125, a finer grid than the reported fit's, reached in two coordinate-ascent rounds rather than four. The 36-parameter step-conditional form does add degrees of freedom, and it is 0.0004 *worse*. Nine parameters are therefore the complete family short of conditioning on the step, which is why we report the simplest one. The nine-parameter form is the original parameterisation and the richer ones were added afterwards, initialised from its fitted solution; it is the incumbent rather than the winner of a comparison, and the spread among the three is well inside the 0.0022 replication reference. On the reported past-only cell the rule's val gain is $+0.0139$ against a test gain of $+0.0152$, a generalisation gap of -0.0013: the gain does not shrink out of sample. Re-fitting the offsets on half of validation and confirming them on the other half reproduces the test gain to within 0.0009 on all eighteen checkpoints of Table 7 (median 0.0004), so the nine parameters are not fit to validation noise. Across all seven class-weighted architectures of the original-state family the offsets improve both macro-F1 ($+0.0111$ to $+0.0166$) and accuracy, so on that family the macro-F1 gain costs no accuracy; the trade-off appears only under unweighted cross-entropy (§C.2).

Table 6: Decision-rule variants on StructBC, fit on validation and scored on the held-out test split (958,146 steps), **on the original full state**—the only state on which the temperature and step-conditional variants were fit, which is why the ladder is shown here rather than on the reported cells; Table 7 is the corresponding past-only state ladder over the decodings that were. "Gen. gap" is the validation gain minus the test gain. The nine-parameter constant offset is the form the reported rule takes; its own past-only state gap is -0.0013, given above, and the two should not be differenced.

Decision rule	Free params	val F1	test F1	Δ test	Gen. gap
$\arg\max$	0	0.4759	0.4958	—	—
constant per-class offset	9	0.4900	0.5105	+0.0147	−0.0006
temperature + offsets (T=0.625)	10	0.4906	0.5107	+0.0149	−0.0002
step-conditional offsets (4 buckets)	36	0.4904	0.5101	+0.0144	+0.0002

The spread in post-rule test macro-F1 across the seven architectures of Table 10 is 0.0159, comparable to what the rule buys on any one of them ($+0.0111$ to $+0.0166$): choosing the operating point matters about as much as choosing among these architectures. On the original full-state search the same layer comparison reads $+0.0147$ for the rule against $+0.0052$ for the selected configuration.

Every decoding we can defend, on one cached forward pass. Table 7 is the control behind §5.3: eighteen checkpoints—five configurations on the ten-action state and four over the nine observed actions, two seeds each—with every decoding scored on the *same* logit matrix per checkpoint, so no variant can win by being measured on a different label set. Two of the four alternatives are exactly inert. The train-support action mask vetoes no argmax prediction on any cell, so page feasibility is not what the offsets enforce. That was originally asserted from the argmax decode alone, which does not settle it: the offsets are chosen to lift the rare classes, so they are the decode that could put one in a page context train never showed. Measured on the reported checkpoint, the offset-decoded predictions violate train support at rate 0.0 on validation and test alike, and applying the mask on top of the offsets leaves test macro-F1 at 0.5065—the mask is inert against the decode the paper reports,

not only against $\arg\max$. The hand-written page-semantics mask is excluded from the table entirely because it rejects 2.6–2.7% of true labels, which makes it a wrong mask and its scores a measurement of our error. Logit adjustment's τ grid selects 0.00 at all eighteen cells, so the val-optimal prior correction is no correction at all. De-weighting is the one alternative that moves, and it moves the wrong way for the reported metric: -0.0177 macro-F1 for $+0.0337$ accuracy. Nine-class macro-F1 is not comparable to ten-class; only deltas within one block are.

Table 7: Decodings of one cached logit matrix, test macro-F1, mean of seeds 42 and 43; offsets fit on validation and scored on test after fitting. The upper block is the reported ten-action state, the lower the nine observed actions with the offsets *refit* in that label space. Absolute numbers are not comparable across the two blocks.

Score model	$\arg\max$	+ mask	$-\log w_c$	$-\tau \log \pi_c$	+ offsets
ten actions					
GRU	0.4806	0.4806	0.4585	0.4806	0.4980
GRU + block	0.4865	0.4865	0.4608	0.4865	0.5015
Bare Transformer	0.4884	0.4884	0.4728	0.4884	0.5039
StructBC − slate	0.4910	0.4910	0.4724	0.4910	0.5063
StructBC	0.4929	0.4929	0.4727	0.4929	0.5079
nine observed actions, derived label held out					
GRU	0.5279	0.5279	0.5109	0.5279	0.5419
Bare Transformer	0.5345	0.5345	0.5218	0.5345	0.5480
StructBC − slate	0.5349	0.5349	0.5254	0.5349	0.5486
StructBC	0.5375	0.5375	0.5196	0.5375	0.5488

Read within the nine-action block, the decision rule is worth $+0.0113$ against $+0.0030$ for the structural block ($3.8\times$ in per-seed point estimates). The rule's per-seed gain is $+0.0121$ and $+0.0105$ and the structural gain $+0.0018$ and $+0.0042$—so the structural effect's two-seed spread (0.0024) is four fifths of the effect itself, while the rule's (0.0016) is a seventh of it. A paired customer-clustered bootstrap over the same 683 test customers confirms the separation at interval bounds, not only in point estimates: at seed 42 the rule's gain is $+0.0121$ $[+0.0095, +0.0142]$ and the structural gain $+0.0018$ $[+0.0007, +0.0030]$, both excluding zero and disjoint from each other, with the rule's lower bound exceeding the block's upper bound by $3.2\times$. Both sides are scored over exactly nine classes—the derived label's logit forced to $-\infty$ and its column removed, with the macro denominator pinned at nine on every resample—and no resample lost a class, so the denominator is nine throughout rather than varying from draw to draw.

The two interventions are sub-additive, and the rule is the larger one. Comparing rungs *within* the offsets column rather than within a row shows the structural block and the decision rule are not separable contributions to be added. On the nine observed actions the block's margin over the bare backbone falls from $+0.0030$ under $\arg\max$ to $+0.0008$ once both models are offset-decoded, below the 0.0018 replication spread; the slate leave-one-out falls from $+0.0026$ to $+0.0002$ on the same pair of columns. On ten actions the block's margin survives ($+0.0045$ to $+0.0040$, both above 0.0022) but its slate leave-one-out does not ($+0.0019$ to $+0.0016$). The rule, by contrast, is worth between $+0.0135$ and $+0.0174$ at every rung in the table and holds on both backbones we decoded: it lifts the GRU from 0.4806 to 0.4980 and the bare Transformer from 0.4884 to 0.5039, so crossing 0.50 on this corpus does not require StructBC. The mechanism is the obvious one—both interventions move mass toward the rare classes the macro average is dominated by, so they repair overlapping errors and their gains cannot be summed. This is the same conclusion as §5's, measured one level down: the operating point is the larger, cheaper and architecture-independent share of the final level, and the structural block is a smaller effect that is only separable from replication noise before the rule is applied.

The decoding also moves the predicted action mix on logged states toward the observed one. A simulator is read for rates—conversion, cart-add, session termination—so a decoder that wins macro-F1 while distorting the predicted action distribution would be unsuitable for estimating behavioral rates, and a rule fit against macro-F1 is precisely what one would suspect of doing that. Measured on the same test predictions, it does the opposite: total variation between the predicted and observed

action mix falls from 0.1055 under $\arg\max$ to 0.0498 with the offsets, and Jensen-Shannon from 0.0090 to 0.0016, replicated at the second seed ($0.0981 \rightarrow 0.0522$) and in the nine-action label space ($0.0907 \rightarrow 0.0536$). Analytic de-weighting moves it the wrong way (0.1383), which is the distributional reading of the same trade-off it makes in F1. Two biases survive the correction and we name them: the PURCHASE *action* is predicted on 0.050 of test steps against an observed 0.056, and ADD_TO_CART on 0.077 against 0.085. Every number in this paragraph is read *off* the macro-F1 fit rather than fitted to the mix; refitting the same nine offsets against total variation directly takes it to 0.0208 for 0.0021 of test macro-F1, and we keep the macro-F1 operating point throughout because it is the criterion the entire comparison ladder is denominated in. Appendix E gives both vectors and the reasons that 0.0208 carries less than it appears to. These are marginal action frequencies over logged states under open-loop scoring, not episode-level business rates: the latter are a property of trajectories the policy generates for itself, which we do not evaluate. The reported policy is not fit against a distributional target; these numbers are §3.4's general form evaluated on a rule fit for macro-F1, and $b^\star(M)$ for a distributional M is the exploratory refit above.

C.2 Is the Offset Gain Just Undoing Class Weighting?

The offsets are fit on a class-weighted model, which invites the reading that they merely undo the distortion that weighting introduced. We retrained the reported configuration under otherwise identical *unweighted* cross-entropy, on the original full state at seed 42, and scored both on the held-out test split. Under $\arg\max$ unweighted CE reaches 0.4787 macro-F1 at 0.6185 accuracy against the class-weighted 0.4958 at 0.5829; with offsets the two land at 0.5083 and 0.5105 macro-F1. The rule is therefore worth more under unweighted CE, not less ($+0.0295$ against $+0.0147$), so the effect is not an artefact of weighting; and since the two objectives finish 0.0022 macro-F1 apart, weighting and post-hoc offsets are partly substitutable ways of reaching the same operating point. The proper scores, measured on the $\arg\max$-stage probabilities before any offset, make the trade explicit: unweighted CE is better calibrated on all three (NLL 1.068 against 1.171, Brier 0.520 against 0.562, ECE 0.0175 against 0.0537—a $3.1\times$ difference), which is the probabilistic quality the weighted objective spends to buy discriminative balance. The effect is asymmetric: on the unweighted model the offsets raise macro-F1 but cost 2.3 points of accuracy ($+0.0295$ macro-F1 at -0.0232 accuracy), whereas on the seven class-weighted architectures we scored they improve both. The no-accuracy-cost claim of §5.3 is therefore a property of the class-weighted family we report, not of the offset rule in general.

C.3 Confirmation After Category-Match Removal

The category-match defect above affects the *state*, so it is a potential confound for any conclusion drawn from that state, not only for the feature's own contribution. We therefore retrained the headline comparisons with both category-match channels zeroed at train *and* evaluation time, so every delta below is internal to the reduced state; no number here is ever compared against a with-channel cell. Table 8 reports the result.

The structural gain, the matched-capacity persona null, and the decision-rule gain all survive removal of the channel. The structural gain replicates at every seed ($+0.0038$, $+0.0056$) and on held-out test ($0.4905 \rightarrow 0.4934$, $+0.0029$); the persona null holds with matched capacity at both seeds, zeroed content scoring $+0.0048$ and $+0.0084$ above real content; and the decision rule is worth $+0.0149$ macro-F1 on test, with the temperature and step-conditional variants still no better (-0.0001, -0.0003). One quantity changes materially: the structural gain on test falls from $+0.0052$ to $+0.0029$, so the gap between the two layers widens—the nine-parameter rule is worth $+0.0149$ macro-F1 against $+0.0029$ for the structural stack, where the full-state numbers were $+0.0147$ against $+0.0052$. One conclusion does not survive: the per-module asymmetry we previously reported was an artefact of reading one ablation column off the contaminated state and the other off the clean one, and it disappears once both columns are retrained on a single state at multiple seeds—first here, and again on the reported state (Table 5)—which is why the architecture is now reported as one aggregate choice (§5.2, Appendix B).

Held-out test on the reduced state. Both sides of the structural comparison were re-scored on the $958{,}146$-step test split with the channel zeroed, and the offsets refit on the corresponding reduced-state validation logits rather than reused from the with-channel fit. These were the reported numbers

Table 8: Headline comparisons re-derived on a reduced state: both category-match channels zeroed at train and evaluation time in every cell, validation macro-F1. Parameter counts are unchanged from the corresponding with-channel cells, confirming the ablation alters no shape. Removing slate attention *raises* the parameter count, to 3,494,628 against StructBC's 3,445,796: the flat MLP that replaces it consumes the whole 416-dimensional concatenated slate at once and so is 48,832 parameters larger than the per-product attention path it substitutes for. Its leave-one-out cost is therefore not a capacity reduction.

Comparison	Cell	seed 42	seed 43
StructBC vs. bare backbone	bare Transformer backbone	0.4702	0.4699
	StructBC (four modules)	0.4740	0.4755
	Δ structural	+0.0038	+0.0056
slate-attention leave-one-out	StructBC minus slate attention	0.4713	0.4729
	Δ removed	−0.0028	−0.0026
persona, matched capacity	real persona content	0.4686	0.4651
	zeroed persona content	0.4734	0.4735
	Δ from zeroing	+0.0048	+0.0084

at this stage of the repair; the reported performance in §5 is the *past-only* state that §3.2's second repair produced, and the two should not be differenced. Under $\arg\max$ the bare backbone reached 0.4905 macro-F1 and StructBC 0.4934, against the same 0.2564 floor; applying the refit offsets to StructBC gave 0.5083. Both orderings the paper draws on held here—the rule worth +0.0149 against +0.0029 for the four modules, where the with-channel state gave +0.0147 against +0.0052—and both persist into the past-only state (§5.3). The structural gain is the one headline quantity either repair moved materially. Both confirmatory suites retrain preselected configurations: we did not repeat architecture selection after either repair, so +0.0029 is a StructBC-versus-bare gain and not the outcome of a search, and we make no claim that StructBC remains the optimum under the repaired states.

Bootstrap intervals on the reported test deltas. The seed-variation reference of §5.2 is descriptive and says nothing about sampling error on the test split, so we add a paired bootstrap, recomputed from scratch on the reported past-only state cells so that the intervals correspond to the point estimates of §5—an interval is a statement about one specific pair of scored predictions and does not carry across a change of state. The unit of independence is the *customer*, not the episode and not the step: the splits are customer-disjoint by construction and each of the 683 test shoppers contributes about 68 sessions, whose behavior is correlated by precisely what the model conditions on, so an episode-level bootstrap is anti-conservative and a step-level one badly so. We perform a paired customer-cluster bootstrap with 2,000 resamples: each draw samples test customers with replacement and includes *all* episodes belonging to each selected customer, both compared systems are scored on the same draw so that between-customer variance cancels in the difference, and macro-F1 is recomputed from the pooled confusion counts of the resample. We report percentile 95% intervals. The macro average is taken over a denominator pinned to the full label set on every draw, and we record the smallest number of distinct classes any resample contained: ten of ten on the ten-action comparisons and nine of nine on the nine-action ones, so no interval here averages over a denominator that moves between draws. Bootstrap intervals quantify test-sample uncertainty conditional on the fitted checkpoints; training-seed variability is a separate source, evaluated through replicated runs (§5.2).

Table 9 reports the two deltas the headline claims rest on, and both exclude zero. Clustering widens every interval relative to an episode-level resample—measured on the reduced-state cells, by 2.2× for the decision rule and 1.3× for the structural delta—and neither crosses zero as a result. The ordering the paper draws on survives at the interval bounds rather than only at the point estimates: the rule's lower bound (+0.0131) exceeds the structural gain's upper bound (+0.0035) by about 3.7×. The structural interval is narrow but its magnitude remains small in absolute terms: the effect is separated from sampling error, not large.

As a diagnostic on the same bootstrap, removing slate attention from StructBC costs +0.0014 ([+0.0003, +0.0024], SE 0.0005, Δ accuracy +0.0036) on test. We do not present this as attribution to that component: the interval speaks to test-sample uncertainty for one pair of checkpoints, while

the question of whether the effect is a component effect at all is answered by replication, and Table 5 plus Appendix D answer it negatively.

Table 9: Paired *customer-clustered* bootstrap (2,000 resamples, 683 test customers carrying 46,376 episodes and 958,146 steps), both cells on the reported past-only state. Each row resamples customers once and scores both models on the same draw; intervals are percentile 95%. The two rows are the two headline effect-size claims in §5.2–§5.3—the decision layer against arg max, and the structural block against the bare backbone. The decision-rule row uses one checkpoint on both sides, differing only in whether the fitted offsets are applied—the cleanest available pairing.

Comparison	Δ macro-F1	95% interval	SE	Δ acc.
metric-aware decision rule vs. arg max (same ckpt.)	+0.0152	[+0.0131, +0.0170]	0.0010	+0.0056
StructBC vs. bare backbone (structural)	+0.0023	[+0.0013, +0.0035]	0.0006	+0.0061

Table 10 supports Section 5.3: the decision rule improves all seven scored architectures on *both* accuracy and macro-F1. Within the class-weighted architecture family, that rules out the interpretation that the macro-F1 gain is purchased by lower accuracy. It also shows the two metrics ranking the models near-identically, with our model best on both. Table 11 gives the per-action breakdown of the reported StructBC cell on test under the offset rule. Note that even the weakest of the seven reaches 58.2% accuracy under the rule, +32.5 points over the 25.6% floor, so the headline result is not a lucky checkpoint but a property shared across the family.

Table 10: The decision rule applied to seven architectures spanning both design axes, **on the original full state**, all scored on the held-out test split with the bias fit on validation. It improves every model on *both* metrics. The claim this table carries is that the rule generalises across architectures, which is why it is not re-run on the reported state; the reported cell's own gain is +0.0152 (Table 1) and is not comparable row-wise with these. Majority-class floor: 0.2564 accuracy, 0.0408 macro-F1.

	Accuracy		Macro-F1		
Model	arg max	+ offsets	arg max	+ offsets	Δ
StructBC (ours, no persona)	0.5829	0.5962	0.4958	0.5105	+0.0147
Transformer + persona (best)	0.5755	0.5914	0.4856	0.4997	+0.0141
same model, persona zeroed	0.5802	0.5912	0.4908	0.5055	+0.0147
GRU + persona (best combination)	0.5768	0.5890	0.4907	0.5018	+0.0111
GRU + persona (single mechanism)	0.5758	0.5884	0.4842	0.5007	+0.0166
GRU + persona (all), zeroed	0.5774	0.5884	0.4875	0.5015	+0.0140
GRU + persona (all mechanisms)	0.5723	0.5819	0.4812	0.4946	+0.0134
Majority-class predictor	0.2564	—	0.0408	—	—

The decision layer improves nine of the ten per-class F1 scores and costs 0.0075 on CLICK_RECOMMENDATION, the majority class, which is the trade the macro average is meant to make: the largest single gain is SESSION_END (0.2021 → 0.2575), the class arg max is most reluctant to select.

C.4 The Persona Null: Probe, Mechanism, and Scope

This appendix expands the persona control discussed in §3.2. The null is a statement about our state construction, not about persona information, and the two are separable by measurement.

What was tested, and the matched-capacity ablation. We tested six injection mechanisms—numeric gating, concatenation, per-field pooling, per-field cross-attention over the slate, FiLM (Perez et al., 2018), and a late-fusion logit gate—all their combinations, a contrastively fine-tuned text encoder, and a low-capacity bottleneck. None helps. The cleanest statement is a matched-capacity ablation that holds the parameter count fixed: keeping every persona module in the graph but feeding all-zero content *improves* validation macro-F1 by +0.0058 on the Transformer family (mean of seeds 42 and 43, which agree to 0.00007), and removing the pathway outright gives 0.4711 against 0.4658 with real personas. A second family replicates the direction but not the magnitude (+0.0040, its two

Table 11: Per-action performance of the reported StructBC cell plus the nine-parameter decision layer, on the past-only state and the held-out test split (958,146 steps), sorted by support. Overall accuracy 0.5904, macro-F1 0.5065—the headline pair of Table 1, recovered here through a separate per-class scoring path. The right-hand three columns are the same checkpoint under $\arg\max$ (macro-F1 0.4913), so each row shows what the decision layer does to that class on all three measures—the two cells §5.3 quotes are SESSION_END recall and PURCHASE_COMPLETE precision, read across. Macro-F1 weights all ten rows equally, so the headline is not inflated by any single easy class.

	Decision layer (reported)			$\arg\max$			
Action	Prec.	Rec.	F1	Prec.	Rec.	F1	Support (%)
CLICK_RECOMMENDATION	0.6945	0.7484	0.7204	0.6778	0.7861	0.7280	25.6
SEARCH	0.6768	0.5889	0.6298	0.6856	0.5708	0.6230	23.6
CLICK_PRODUCT	0.5680	0.5897	0.5786	0.6030	0.4991	0.5462	19.8
ADD_TO_CART	0.5515	0.4975	0.5231	0.5199	0.5189	0.5194	8.5
PURCHASE_COMPLETE	0.7298	0.6472	0.6860	0.6174	0.7246	0.6667	5.6
GO_BACK	0.3912	0.3555	0.3725	0.3957	0.3393	0.3653	4.9
SESSION_END	0.2372	0.2816	0.2575	0.3096	0.1500	0.2021	4.8
REFINE_SEARCH	0.3711	0.4760	0.4171	0.3126	0.6079	0.4129	3.7
OTHER	0.3985	0.4251	0.4114	0.3165	0.5110	0.3909	2.0
BROWSE	0.4454	0.4954	0.4690	0.3791	0.5802	0.4585	1.5
Macro average	0.5064	0.5105	0.5065	0.4817	0.5288	0.4913	—

seeds 0.0035 apart), so the effect is firmly established on one family and directional on the second. Under the same operating-point protocol the conclusion is unchanged: the best persona model reaches 0.4997 test macro-F1 under the offset rule, below the no-persona 0.5105—both rows of Table 10, so both on the original state. Wang et al. (2026) report the same direction at nearly three orders of magnitude smaller scale.

The persona embeddings do carry signal. A ridge probe from the same embeddings to a customer's *action-frequency histogram* beats predicting the training population's mean histogram by 13.0% MAE (0.0553 → 0.0481): fit on the eligible training customers, evaluated on the 1,322 validation and test customers with ≥ 20 steps, on the same customer-disjoint partition as every other cell, so the probe is held out from the same shoppers the model is. It does so unevenly across actions, explaining $R^2 = 0.38$ of the variance in a customer's PURCHASE_COMPLETE rate and 0.22 of their ADD_TO_CART rate against 0.05 for BROWSE. A five-fold variant within the training split agrees to within a point (13.6%), so the result is not an artefact of which customers were held out. Persona carries real signal, concentrated in exactly the transactional propensities a simulator is commercially used for, and it nevertheless provides no measurable incremental next-action benefit in this setting.

Why both results hold at once. The two results are consistent with *incremental* redundancy, the difference being what is asked: persona predicts *who a shopper is across a session*, while the next-action task asks what they do at one page. Two measurements bound this. Page type alone—a single categorical—already yields 45.6% next-action accuracy against a 25.6% floor (Table 1), so much of the decision is page-determined. And c_t contains behavioral signals that overlap with the persona—recent action mix, price level and browse-versus-buy tendency—all present in a shopper's own rolling action statistics, counters and price features. The experiments do not isolate which state components absorb the persona signal, so we treat redundancy as the supported interpretation rather than a demonstrated causal mechanism. Redundancy on its own predicts a null, and what we measure is a small, consistent *cost*. That asymmetry needs its own account, and the matched-capacity design rules out the simplest one: parameters are held fixed and only the content is zeroed, so the degradation is not capacity dilution. What it does not rule out is the content itself acting as an identifier. A 3,090-dimensional vector that is constant within a customer and nearly unique between them lets the model fit customer-specific mappings that cannot transfer across a customer-disjoint split, which would show up as exactly this sign. Our runs do not separate that from redundancy, and we report the degradation as unexplained rather than adopt either account.

Scope conditions and caveats. The redundancy reading predicts where persona *should* pay off: under a state with no accumulated in-session history—cold-start at step 0, or a simulator conditioned only on the current page—a static prior is the only source of customer-level variation. It predicts the converse too, for session-level aggregates such as conversion rate or basket composition, which is what the probe above actually predicts. Neither regime is isolated by our runs, so both are experiments this null motivates rather than ones we have run. Two caveats bound the null itself. Our persona representation is homogeneous—mean pairwise cosine similarity between persona texts is 0.70—so a stronger encoder is the natural rescue experiment. And the persona analysis—sweep and operating-point confirmation alike—ran on the original state, which did contain category-match to purchase history, itself a persona-derived channel. That makes it a test of *additional* persona content rather than of persona content as such. We therefore do not claim persona information is useless, only that explicit persona conditioning does not improve next-action prediction on this task, from these personas, under any of the six mechanisms tested, on a state that already carries in-session behavioral history. Wang et al. (2025) list persona integration as open.

D Robustness to the Derived Label: Nine Observed Actions

SESSION_END is the one label we construct (§3): it marks the boundary of an interaction burst under a 30-minute inactivity window rather than a recorded click. Predicting it is therefore partly predicting our own labelling rule. To bound what that is worth we retrained the ladder over the nine *observed* actions alone, holding out the one step per episode whose label is the boundary, on the reported past-only state at every seed. Table 12 gives the cells. Nine-class macro-F1 averages over nine terms instead of ten and the two are *not* comparable as absolute numbers; what transfers between them is the ordering and the deltas.

Table 12: The ladder retrained over the nine observed actions, validation macro-F1, on the reported past-only state, every seed. Δ is the mean over seeds of the same-seed difference against that row's comparison base: the matched GRU for the bare Transformer (the backbone effect), and the bare Transformer for StructBC (the structural effect, which therefore skips the row between them). The two-seed spread of the nine-action StructBC cell, 0.0018, is the reference. Nine-class macro-F1 averages over nine terms and ten-class over ten, so these levels are not comparable with Table 5's; only the ordering and the Δs transfer.

Configuration	s42	s43	Δ
matched GRU, bare	0.5082	0.5085	—
bare Transformer	0.5150	0.5143	+0.0063
StructBC − slate attn.	0.5155	0.5151	—
StructBC (all four)	0.5165	0.5183	+0.0027
slate-attention leave-one-out (StructBC − slate attn. minus StructBC)			−0.0021

Three readings follow, and we take all three. First, the ordering GRU $<$ bare $<$ StructBC is preserved at every seed, so the reported result is not an artefact of predicting our own construction. Second, both aggregate effects keep their sign—structural $+0.0027$ against $+0.0055$ on ten actions, backbone $+0.0063$ against $+0.0069$—though the structural effect roughly halves in this label space, and unlike the ten-action cells the two effects here are separated by more than the 0.0018 reference. We do not read that as the ordering: it holds in one of the two label spaces we ran. Third, the one component-level effect that cleared the ten-action reference does not clear the nine-action one *robustly*: slate attention's removal costs -0.0021 against a 0.0018 reference—sign-consistent across seeds, but with per-seed magnitudes straddling that reference (-0.0010 and -0.0032), so the across-seed mean clears it and replication does not confirm that it does. One number looks like it overturns this and does not: the clustered bootstrap of that same leave-one-out gives $+0.0012$ $[+0.0002, +0.0024]$, excluding zero—by 0.0002. It resamples test customers with both checkpoints held fixed, so it bounds sampling error *given one training run*, while the effect fails on the axis 0.0018 measures, seed variability, which no bootstrap at fixed checkpoints can observe. We report it because it was run and treat it as a diagnostic, which is the second and independent reason §5.2 reports the block as one aggregate rather than attributing it.

All four rungs above are validation, and all four are test-scored in Table 7 with the offsets refit in the nine-class label space.

The auxiliary next-page loss is not what produces the result. Every score model in our comparisons is trained multi-task, with an auxiliary next-page-category cross-entropy at $\lambda = 0.1$ (§4), so the structural comparison could in principle be a statement about that auxiliary task. Retraining StructBC at $\lambda = 0$ on the same reduced state at every seed gives 0.4753 and 0.4751 validation macro-F1 against 0.4740 and 0.4755 with the auxiliary loss—a same-seed mean of $+0.0004$, inside that state's 0.0015 two-seed reference, and opposite in sign between the seeds. We therefore find no measurable effect of the auxiliary head on next-action macro-F1 at this resolution; we keep it because every cell of the search was trained with it, and changing it would break comparability with the search, not because it is measurably worth anything.

E Reproducibility

Episode filter, attributed by criterion. Of the $676{,}107$ bursts, $452{,}338$ survive the three retention criteria of §4. The three-step minimum drives most of the loss in *burst count*—22.96% of bursts—but only 2.40% of recorded interactions, since what it removes is almost entirely bursts of one or two steps. The entry-page restriction is the reverse: 6.31% of bursts and 7.80% of interactions. In total the filter retains 66.90% of bursts and 85.85% of interactions, and removes 73 customers outright, leaving $6{,}824$. These counters are recomputed from the pre-filter corpus; they had not been logged at the time the filter was written. Of those $452{,}338$ filtered bursts, $452{,}296$ reach the built dataset; the 42-episode shortfall (0.009%) is the dataset builder re-applying the three-step minimum to the rows it reads. Every episode count reported in this paper is the built figure, $452{,}296$.

Split assignment. The split is over customers, not episodes or steps. We take the unique `pseudo_customerid` values in order of first appearance in the corpus, permute them once with NumPy's Mersenne Twister seeded at 42, and assign the first 80% to train, the next 10% to validation and the remainder to test, each size truncated to an integer: $5{,}459$ / 682 / 683 shoppers. All episodes of a shopper follow that shopper, which is what makes the splits customer-disjoint, and the assignment is fixed across every cell in this paper—no run reshuffles it, so the seed reported per cell governs initialisation and batch order only. There is no temporal holdout: the date range (20260407–20260605) is shared by all three splits.

Action space. The ten actions, in index order, are: SEARCH (0), REFINE_SEARCH (1), CLICK_PRODUCT (2), CLICK_RECOMMENDATION (3), GO_BACK (4), ADD_TO_CART (5), PURCHASE_COMPLETE (6), BROWSE (7), OTHER (8), SESSION_END (9). SESSION_END is terminal: it is the final action of every episode and occurs nowhere else, which is why any feature encoding a step's *relative* position within its episode is not computable at serving time; an assertion in our data builder rejects that class of feature for this reason. The decision-rule bias vectors given later in this appendix are in this order.

The session-boundary label needs no synthetic state. The action target for step t is what the shopper does *from* page t, read off the next logged row; for the last real page that target is SESSION_END. The appended terminal row therefore supplies a label and is itself excluded from the states array, so no state is synthesised for it and no feature exists that occurs only at the boundary: an episode of n logged rows contributes $n - 1$ scored steps, each with a genuine observed page and history. The shift is also why the builder re-applies a minimum length to the rows it reads, which is the source of the 42-episode shortfall of §4.

Long episodes are truncated, not chunked. Episodes are capped at 200 logged rows. The terminal SESSION_END row is appended to the *complete* session and the cap is applied afterwards, so an episode longer than the cap loses its genuine terminal row along with its tail rather than receiving an artificial terminal label at the cut: no truncated episode carries an SESSION_END label anywhere. Verified by counting labels in the built splits. Truncation affects $2{,}567$ of $363{,}562$ train episodes (0.71%; 205 of $42{,}358$ validation, 379 of $46{,}376$ test), mean episode length 19.5 steps. Every untruncated episode has exactly one SESSION_END label and it is always the final step; the count

of SESSION_END labels equals the count of untruncated episodes exactly, on all three splits, and no SESSION_END label occurs at a non-final step. The cost of this choice is the converse of contamination: a small minority of episodes contribute states without ever contributing their session-boundary label.

The two repaired features, and what their repairs cost. The category-match channel (§3.2) compares the product in view against the set of categories the customer has purchased in. That set is assembled from the full corpus rather than from each state's past-only prefix, so for customers who purchase on more than one date—96.2% of ours—it can be decided by a purchase occurring after the state being scored. On $400{,}000$ sampled states the channel fires at 0.771 under the corpus-wide set against 0.611 past-only, differing on 16.0% of states. Rebuilding the corpus past-only would require re-running every cell reported here, so we instead remove the channel entirely and re-derive the affected conclusions on the reduced state. One retained finding, the per-product popularity scalar, is transductive corpus-level information rather than shopper-specific future leakage, and is documented with the feature inventory above, where the assumption behind retaining it is stated and the cost of removing it instead is measured. Removing the entire category-match channel changes validation macro-F1 by -0.0019 ($0.4759 \to 0.4740$), which is a sensitivity analysis for this feature but does not isolate the effect of its future-only component: zeroing the channel removes its past-only information as well, and in a nonlinear model whole-channel removal does not bound the future-only part. Nor is the comparison against a backbone that still contains the channel informative. We therefore re-ran the headline comparisons with the channel removed from *both* sides (§C.3), which is what the conclusions rest on.

The purpose channel. At each state ρ_t is the empty-query embedding until the first search query of the episode's scored window is observed, and that query's 128-dimensional embedding from that step onward:

$$\rho_t = \begin{cases} \rho_{\varnothing} & t < k \text{ (no query yet; and every } t \text{ if} \\ & \quad \text{the episode has none)} \\ \mathrm{PCA}_{128}(\mathrm{SBERT}(q)) & t \geq k, \end{cases}$$

where k is the state index at which the query first appears and $\rho_{\varnothing}$ is the embedding of the empty-query token, the same vector the no-query episodes receive. The PCA basis is fit on training customers' episodes alone. No state therefore carries query information from a later event in its own episode, and the transform saw no held-out customer. The step index is derived under the dataset builder's own row ordering and gated against the split parquets: the builder's n_{steps} must equal $n_{\text{rows}} - 1$ for every one of the $452{,}296$ episodes, or the index would point at the wrong state. A query first appearing at or beyond the scored window counts as no query.

The superseded channel, and why it was replaced. An earlier exploratory state used the episode's first search query held constant across the *whole* episode. That is past-only only for episodes that open on a search, which 40.1% do; the filter of §4 also admits detail-entry (53.9%) and browse-entry (6.0%) episodes, in which the query can arrive later. Measured over all $452{,}296$ episodes, 36.7% carried their query at the opening step and 19.2% at a later step, so 8.0% of all scored steps (7.8% of test steps, a median of 3 leading steps per affected episode) were conditioned on a query not yet issued; its PCA basis was also fit on a random 80% of *all* episodes rather than on training customers alone. Both defects were found by a per-feature audit rather than by any held-out number, and neither is present in the construction above. All headline performance, structural, nine-action and decision-layer comparisons use the reported past-only state; explicitly labelled exploratory, persona and auxiliary controls may use earlier states. §3.2 states the same restriction in the main text.

Table 13: Per-feature audit of the reported past-only state. "Past-only" means the channel is computable from information observed strictly before the scored action; the name asserts this of every shopper-specific channel, and the two channels outside its scope—a corpus-window product-popularity constant, and a fixed-span rescaling of the promotion counters—are marked in the table, the first quantified below. The audit is over feature *groups* as the dataset builder writes them, and it returns four findings about feature construction (§3.2), two of which surface in more than one feature group. Two are repaired in the reported state. Two are retained and disclosed: item popularity, and the promotion counters' rescaling; both are transductive corpus- or calendar-level information rather than shopper-specific future leakage. Popularity is kept as a ranking proxy on the stationarity assumption stated below—at a measured 0.0037 of test macro-F1, the cost of zeroing it and retraining (paragraph after this table). For the two repairs, no held-out number distinguishes the corrected state from the uncorrected one; §C.3 re-derives the headline comparisons inside the corrected state instead.

Feature group	Dim.	Source	Past-only	Correction applied
Search slate	416	page as rendered at t	yes	—
of which cat.-match-to-purchase-history	32	purchase history, corpus-wide	**no**	slot retained, **zeroed** in training and evaluation
of which item popularity	32	view/impression counts, corpus-wide	**no**	retained as a ranking proxy
Detail page	26	page as rendered at t	yes	—
of which cat.-match-to-purchase-history	1	purchase history, corpus-wide	**no**	**zeroed** with the slot above
General/cart	10	within-session counters	yes	—
Session context	61	prior steps of this episode	yes	—
of which item popularity of the viewed product	1	view/impression counts, corpus-wide	**no**	retained as a ranking proxy
of which promotion counters	5	platform-level counts, contemporaneous; rescaled over a fixed calendar span	**no**[†]	retained, rescaling disclosed
Purpose	128	first query observed *up to* t	yes	rebuilt from an episode constant to a step function; PCA basis refit on train customers
Persona	$384{+}7{\times}384{+}18$	profile built outside the episode	n/a[‡]	**excluded** from every reported cell (§C.4)

[†]The counts are contemporaneous and available online, but each series is min–max rescaled with constants taken over a fixed 108-day promotion calendar (2026-03-15 to 06-30) that extends beyond the extraction window, so the scaled value is not computable strictly before t. The transform is affine, shared by every state, and derived from no shopper's behaviour; together with the popularity lookup above, these are the only non-shopper-specific inputs in the reported state. [‡]Exploratory only: the profile is built from a history window we did not constrain relative to the episode, so we do not claim it is past-only. It enters no reported cell.

Feature dimensions. Search observation $416 = 32$ products $\times$ 13 features: price and a price-present gate, rank, fulfillment, brand presence, rating, review count and a reviews-present gate, the **category-match-to-purchase-history slot (zeroed in the reported past-only state, Appendix C.3)**, variant count, seller count, an exclusive-offer flag, and an item popularity scalar. The last is a fixed per-product lookup, $\log(1+v)$ of a view and impression count v for the product, rescaled to $[0,1]$; v is accumulated over the *entire* corpus window, so this channel is aggregated across dates later than the scored step and across customers in all three splits. It is outside the scope of the "past-only" name—as is the promotion rescaling marked in Table 13—because that name asserts prefix-validity of every shopper-specific channel, which is what the two repairs restored, and this channel is not shopper-specific at all. The audit marks it accordingly. Three properties separate it from the two repaired findings: the table is a single scalar per product shared by every state, so it contains no focal-shopper-specific future event and no per-state target field; nothing in it is specific to the shopper being scored; and what it supplies to the model is an ordering of products by aggregate demand rather than a time series. We retain it as a constant proxy for that ordering, on the assumption that a

product's standing among products is slow-moving relative to our 59-day extraction window (§4), so that a prefix-restricted count would rank the catalogue much as the window-wide count does. We state that assumption rather than verify it: we do not measure the rank stability of the counts. What the retrain does measure is how much the reported cell owes the channel at all. Rebuilding the state with all thirty-three popularity dimensions set to zero, retraining the reported cell at the same seed under an otherwise identical configuration, and scoring it with the same dimensions zeroed—so that training and evaluation see the same state, rather than measuring a train/serve mismatch—costs 0.0048 of validation macro-F1, against the 0.0022 two-seed spread measured on that same state and split; on test it costs 0.0037 at the reported operating point and 0.0047 under argmax. The channel therefore carries roughly twice the run-to-run spread, and the reported cell leans on it by that much. For scale, 0.0037 is under a quarter of the $+0.0152$ the decision layer is worth on this same cell and split. What it establishes is a level: the reported cell scores 0.0037 lower without the channel. We have now re-estimated the attentive pair of §5.2 inside the ablated state, at seed 42, and the structural block's advantage over the bare Transformer does not survive there: the paired customer-clustered gain is $+0.0005$ with a 95% interval of $[-0.0004, +0.0014]$, which includes zero. In the reported state, at the same seed and over the same 683 test customers, the same comparison is $+0.0023$ $[+0.0013, +0.0035]$, which excludes it. We read each interval against zero within its own state and do not difference them, since the two states are two measurement conditions. The consequence for the paper is that the structural composition gain is *not* verified independently of the popularity channel: a channel worth 0.0037 overall is enough to absorb an effect of that size, and our evidence does not separate the two. $+0.0069$ and the seed-43 arm remain un-estimated in the ablated state, so how much of the backbone effect depends on the channel is still not measured. What 0.0037 prices is the whole channel, its past-dated content included, and a past-only rebuild would retain most of that content: it is the channel's net contribution, not the contribution of its future-dated part, which a whole-channel removal does not bound in either direction in a nonlinear model. It is also silent on the rank stability assumed above. The channel is present in every cell we report bar the ablation just priced, and in the fine-tuned baseline's prompt as well—the shared serialiser prints the same popularity scalar for the top eight products of a search slate and for the product in view, nine of the thirty-three values, so neither side of that comparison is alone in seeing aggregate demand. Reproducing the exact scalar additionally requires the artefact rather than the formula: the values we train on are not recovered by the expression above under the normalisation we document, so a reimplementation should expect a different scaling of this channel, and one whose induced ordering we have not shown to agree. Detail page 26: the same commercial attributes for the product in view, how the shopper reached it, a revisit flag, a within-session category-match feature—distinct from the slot above, since it compares against categories viewed *in this session* and is strictly past-only—and a second category-match-to-purchase-history slot, zeroed with the first. General/cart 10; session context 61, split by the multi-scale encoder into the 1-based inclusive ranges $[1{:}10]$ immediate, $[11{:}37]$ short-term, $[38{:}61]$ session-level; purpose 128 (PCA of the 768-d `paraphrase-multilingual-mpnet-base-v2` Sentence-BERT encoder). The general/cart group is the cart-add count, a cart-non-empty flag, session elapsed time, a five-way page-category indicator, and two within-session purchase counters. The context group comprises the previous action, a rolling histogram of the last eight actions, counters (searches, back-presses, cart adds, distinct products and queries seen), steps-since-last-event for four event types, an already-purchased flag, and price statistics of the current slate relative to what the shopper has clicked. Time features are session elapsed time, dwell since the previous step, cyclic hour-of-day and day-of-week, and five promotion counters, all keyed on the calendar date of the step and shared by every shopper active that day: the count of live promotions normalised by the platform maximum over the window, and the counts of price adjustments, fixed discounts, variable discounts and non-discount promotions, each normalised the same way. Together with the popularity lookup, these are the only features in the state not derived from the shopper's own trace. Persona embeddings, where used: 384-d pooled MiniLM (Wang et al., 2020) (`all-MiniLM-L6-v2`) $+ 7 \times 384$-d per-field $+$ 10-d z-scored numeric $+ 8$ categorical indices.

Page-type routing, in full. Two modules branch on the page category (SEARCH, DETAIL, CART, CHECKOUT, GENERAL, OTHER), and both branch identically: SEARCH takes the first branch, DETAIL the second, and the remaining four categories share the third. First, the page observation. o_t is not one vector but whichever of three disjoint feature groups the current page populates, and each group has its own encoder into $\mathbb{R}^{d_s}$: the 416-d slate through $416 \to 256 \to 256$ (replaced in STRUCTBC by slate self-attention, whose mean-pooled 64-d output is projected to d_s), the 26-d detail block through

$26 \to 128 \to 256$, and the 10-d general/cart block through $10 \to 64 \to 256$, each with ReLU and dropout 0.1 between layers. The step's page-observation encoding $\mathrm{state_enc}_t \in \mathbb{R}^{d_s}$ is the output of whichever branch its category selects; the other two contribute nothing to that step and take no gradient from it. That is the symbol used by the residual skip of §3.3, and it denotes the page observation alone: the concatenation $[\,\mathrm{state_enc}_t \| \mathrm{ctx_enc}_t \| \mathrm{purpose_enc}_t\,] \in \mathbb{R}^{3d_s}$ is what the fusion MLP maps to the model width d. Second, the action head. It is three parallel MLPs $512 \to 128 \to 10$ with ReLU, one per branch, selected by the same page category, for $3 \times 66{,}954 = 200{,}862$ parameters. The routing is hard and fixed rather than learned—no gate, no mixture, no shared trunk—so each branch's parameters are fit on the steps of that branch only. The input width 512 is $[\,h_t \| \mathrm{state_enc}_t\,]$ under the residual skip, and $d = 256$ without it.

The training objective, combined. With $m_t \in \{0, 1\}$ the valid-step mask and pc_{t+1} the next page category, the loss summed over a batch's steps is

$$\mathcal{L} \;=\; \frac{\sum_t m_t \,\mathrm{CE}_{w,\,\varepsilon}(a_t)}{\sum_t m_t} \;+\; \lambda \,\frac{\sum_t m_t \,\mathrm{CE}\big(\mathrm{pc}_{t+1}\big)}{\sum_t m_t}, \qquad \lambda = 0.1,$$

where the first term is taken over the action head's logits and the second over the auxiliary next-page head's, and $\mathrm{CE}_{w,\,\varepsilon}$ is cross-entropy with class weights w and label smoothing $\varepsilon = 0.05$. The weights are $w_a \propto f_a^{-\alpha}$ at $\alpha = 0.5$ over the training action frequencies f_a, rescaled to sum to the number of classes so that the mean weight is 1; the auxiliary term is neither weighted nor smoothed. The unweighted-CE control of §5.2 sets $\alpha = 0$, which makes w all-ones and leaves the rest of the expression unchanged, and the nine-action cells zero $w_{\text{SESSION_END}}$ and rescale over the remaining nine so that those runs optimise a loss on the same scale as the ten-action ones.

Hyperparameters. $d_{\text{state}} = 256$, model width $d = 256$; Transformer 3 layers, 4 heads, feed-forward $4d$, sinusoidal positional encoding, causal + padding masks; slate self-attention $d_B = 64$, $L_B = 2$, 4 heads, feed-forward $4d_B$, learned rank positional embedding; windowed temporal attention $K = 8$, 2 heads; multi-scale band encoders of widths 64 (immediate), 128 (short-term), 64 (session-level); AdamW lr 10^{-4}, weight decay 10^{-2}, OneCycle with warmup $\leq 10\%$; batch 128; ≤ 30 epochs, early stop patience 5 on validation macro-F1; class-weighted cross-entropy with inverse-frequency weights at exponent $\alpha = 0.5$ and label smoothing 0.05; auxiliary next-page-type loss $\lambda = 0.1$; mixed precision; gradient clipping 1.0; seed 42 (seed 43 for replication cells). The reported StructBC model has 3,445,796 parameters against the sweep anchor's 3,251,603: dropping the anchor's hierarchical action head removes its 6,669 parameters but restores a 200,862-parameter page-type-routed head, which is why the smaller module set is the larger model. A single cell takes 4.6–9.3 hours on one A10G GPU (mean 6.7); the decision rule fits in seconds on cached logits.

Protocol for the baseline hierarchy. Every rung of Table 1 is fit on train and scored on the same 958,146 test steps. The four count-based rungs are exact most-frequent-action lookups over their conditioning context, with unseen contexts falling back to the global majority action; they have no hyperparameters. The gradient-boosted rung is a histogram-based gradient-boosting classifier (300 maximum iterations, learning rate 0.1, 63 leaves, early stopping on a 10% split of train, converging at 75 iterations) fit on the same inverse-frequency class weights at exponent 0.5 as the networks, so it is not handicapped on macro-F1 by an imbalance the networks correct. It receives the current step's page type, detail state, general state and context vector, plus a flat summary of the 32×13 product slate (per-feature mean, min and max over the slate and the top-ranked product's raw features, 150 features in total). It is *non-sequential*: it consumes the context vector's aggregates but no ordered history and no raw slate. It is fit on a 400,000-step random subsample of train rather than all 7.4M steps. It reaches 0.4512 macro-F1 and 0.5618 accuracy—but on the original state, so it differs from every row of Table 1 in three ways at once. Those departures—the state, the summarised slate and the training subsample—mean its gap to StructBC is not a clean measurement of what sequence modelling buys, in either direction, and no claim in the paper reads it; the matched GRU of Table 1 is the controlled backbone comparison.

Protocol for the decision rule. Coordinate ascent over $b \in \mathbb{R}^{10}$ with $b_{\text{SEARCH}} \equiv 0$, each coordinate swept over $[-2, +2]$ in steps of 0.125, for at most four rounds or until a round yields no improvement (convergence occurs in two to three). Coordinates are visited in ascending action index with the pinned class skipped, each fitted against the coordinates already updated in the same round rather

than against the values they held when the round opened, and each sweep is accepted only on a strict improvement over the incumbent value—so on the plateaus a piecewise-constant objective necessarily produces, ties retain the incumbent, and among strictly better grid points the ascending sweep returns the most negative one. Macro-F1 is evaluated directly on cached validation logits at every grid point; there is no surrogate objective and no gradient. The stricter protocol splits validation in half by customer, fits on val-A, and confirms on val-B before test is scored. On the reported past-only state STRUCTBC cell the val-B gain is $+0.0137$ ($0.4729 \rightarrow 0.4866$) and test reaches 0.5072 against 0.5065 for the full-validation fit—a 0.0007 difference, so the operating point does not depend on having seen all of validation. The same control on the reduced-state cell gave a val-B gain of $+0.0134$ ($0.4751 \rightarrow 0.4885$) with test at 0.5077 against 0.5083, and on the original-state cell of the selection search $+0.0138$ ($0.4768 \rightarrow 0.4906$) with test at 0.5104. The vector recovered this way differs in two coordinates by a single grid step, which bounds the fit's own variance. The step-conditional variant of Table 6 buckets by *absolute* step index, which is available at prediction time. We reject any variant bucketed by relative position within the episode, and enforce this by requiring $P(\text{SESSION_END} \mid \text{bucket})$ to be non-degenerate in every bucket holding at least $1{,}000$ steps: under relative-position bucketing that quantity measures $[0,\ 0,\ 0,\ 0.210]$, so three of four buckets contain no positive example and the fourth carries the entire class, which is the terminal-action structure of SESSION_END restated as a feature.

The fitted bias vector, and a second criterion. The reported nine-parameter rule of Section 5.3 is

$$b_{\mathrm{F1}} = [\,0,\ -0.25,\ +0.125,\ -0.25,\ -0.125,\ -0.25,\ -0.75,\ -0.75,\ -0.625,\ +0.25\,],$$

indexed in the action order given above, with $b_{\text{SEARCH}} \equiv 0$ pinned by construction (the leading zero of the vector; the action list is 0-based, so no index is written here). Section 3.4 defines the layer against a criterion M that the construction leaves free, so we refit the same nine parameters on the same cached validation logits, with the same grid and the same pinned coordinate, against the total variation between the decoded and the observed action mix instead:

$$b_{\mathrm{TV}} = [\,0,\ -0.625,\ 0,\ -0.375,\ -0.125,\ -0.125,\ -0.5,\ -1.0,\ -0.75,\ +0.125\,].$$

The two differ in eight of the nine free coordinates and by up to 0.375 — three grid steps, against a search whose smallest expressible difference is one grid step, which is the threshold we fixed in advance for calling the operating point criterion-dependent. They trade against each other on test: b_{TV} takes total variation from 0.0498 to 0.0208 and macro-F1 from 0.5065 to 0.5044, a macro-F1 cost of 0.0021 against the 0.0022 seed-replication reference.

Three things this does not show. It is not an ordering of the two criteria, which are not on a common scale; it is evidence that the operating point moves with M, and the vector we report everywhere else is b_{F1}. The saturation argument and the out-of-sample one were also run together and are separated here. On validation, nine free parameters against a ten-cell marginal is a nearly saturated fit, so a near-zero total variation there is arithmetic rather than a finding. 0.0208 is measured on test and is a real held-out observation, but not a claim that the simulator reproduces the observed action mix: the criterion was chosen after seeing the data and the offsets were fit on validation, so the only total-variation number here that carries a claim is 0.0498, from the vector that was never fitted to it.