ONLINE LLM PERSONALIZATION FROM PAIRWISE PREFERENCES ACROSS USERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models can answer the same question in different styles, but users' preferred styles are hidden and vary across users and questions. We study inference-time personalization through online selection from a fixed set of prompts, each representing a persona. Users arrive sequentially, each asks a short sequence of questions, and the system displays two responses per question and observes one pairwise choice. We formulate this interaction as a variant of the contextual dueling bandit problem in which the learner starts without preference data, adapts within each session, and exploits information shared across users. The objective is to minimize cumulative strong regret, summing the utility gaps of both displayed responses relative to the user's best response among the available options. We make two contributions. First, we introduce the problem and release a reproducible preference benchmark and dataset built around a tutoring case study, containing educational questions, LLM-generated answers in six teaching styles, and pairwise judgments from LLM-simulated students with hidden preference profiles. We fit Bradley-Terry models to these judgments to obtain reference utilities for regret evaluation. Second, we propose a baseline that learns a mixture prior over users from completed sessions and adapts to each new user within the session. On students and topics never used in development, it removes 41–46% of random pairs' excess regret and about three fifths of what a per-user linear oracle fitted in hindsight removes. We release code, prompts, and the benchmark.

1 INTRODUCTION

A large language model (LLM) chatbot can answer the same question in several styles, and users differ in the style they prefer. A common way to guide the response style is to use a system prompt that specifies a *persona* (Zheng et al., 2024b). In AI math tutoring, a persona can be a teacher emphasizing proofs, worked examples, or guiding questions. The preferred persona depends on the user and the question: the same user may want a proof for a definition and numbered steps for a computation. A simple way to collect preference feedback is to show two answers and ask the user to choose one.

Users arrive one after another, each for a short session, and the platform starts with no preference data. On every question it selects two personas using features of the persona-question pairs and the choices observed so far. A user's preferences are hidden and fixed during the session, and later users can benefit only from the log the policy itself created. Shared structure across the population is what makes that transfer possible.

We study inference-time personalization through online selection from this fixed catalog of prompts. We make two contributions.

1. **Problem formulation and dataset.** We formalize the interaction as a variant of an online, multi-user contextual dueling problem, with cumulative strong regret accounting for the utility gaps of both displayed answers (Yue et al., 2012; Bengs et al., 2022; Wang et al., 2025). We introduce **TutorDuel**, a reproducible tutoring preference benchmark and dataset containing educational questions, LLM-generated answers in six teaching styles, and pairwise judgments from LLM-simulated students with hidden preference profiles. On comparisons labelled by two of the authors for the described students, the judge agrees with

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.

each author as often as the two authors agree with each other (70.0% and 82.5% against 72.5%). Bradley-Terry fits to the LLM judge's choice probabilities provide reference utilities for regret evaluation. The evaluation protocol brackets performance between random pairs and per-user references, and separates new topics from new students with rosters of students never used in development. We also provide a numerical simulator for controlled evaluation.

2. **A baseline algorithm.** We propose **CodueL** as a first method for this problem. It learns a mixture prior over user preferences from completed sessions, starts each new user from that prior, and switches transfer off while that prior predicts users' choices no better than learning each user alone. On three TutorDuel rosters whose topics and students were never used in design (eight new hand-written profiles, and two rosters of 350 distinct students composed from preference traits), it removes 46%, 41%, and 43% of the excess regret of random pairs (Table 1). This is 59% to 60% of what a per-user linear oracle, fitted in hindsight on the same features, removes; we offer the difference as a target for future methods.

2 Related Work

Contextual dueling bandits. Contextual dueling bandits extend dueling bandits (Yue et al., 2012; Bengs et al., 2021) to arms that depend on a context, as in CoLSTIM (Bengs et al., 2022), the algorithms of Saha (2021), and feel-good Thompson sampling (Li et al., 2024b). Related RLHF formulations treat prompts as contexts and responses as arms (Zhu et al., 2023; Xiong et al., 2024; Das et al., 2025). COLDB (Wang et al., 2025) extends clustering of bandits (Gentile et al., 2014; Li et al., 2016) to dueling feedback, pooling observations within clusters of recurring users. Our users instead arrive once, and their preferences need not form discrete clusters. Our pooled RUCB baseline (Zoghi et al., 2014) ignores the question.

Learning priors across tasks. Meta-Thompson sampling and AdaTS learn priors across bandit tasks from rewards (Kveton et al., 2021; Basu et al., 2021). Related work studies transfer across linear and hierarchical bandit tasks (Cella et al., 2020; Peleg et al., 2022; Hong et al., 2022a). MixTS uses a given mixture prior (Hong et al., 2022b), and Simchowitz et al. (2021) analyze sensitivity to prior misspecification. CodueL learns a mixture prior from completed sessions' pairwise choices.

Latent bandits. Latent bandits infer a hidden user type from feedback while minimizing regret (Maillard & Mannor, 2014). Hong et al. (2020) develop Thompson sampling and UCB rules under uncertain type-specific reward models. CodueL similarly keeps a belief over mixture components within each session, with pairwise feedback and regret on both displayed answers.

Personalization across users. Personalized alignment uses policy merging (Jang et al., 2023), user embeddings (Li et al., 2024a; Poddar et al., 2024), personalized reward models (Park et al., 2024a), preference prototypes (Chen et al., 2025), low-rank reward bases (Bose et al., 2025), and meta-learned initializations (Cai et al., 2026). Matchbox supports online collaborative learning (Stern et al., 2009), and Gaussian-process preference elicitation combines cross-user transfer with active querying (Bonilla et al., 2010; Houlsby et al., 2012). Our protocol starts without preference data and measures strong regret on every displayed pair across sequential short user sessions. Online per-user methods are closer in protocol. UserAlign (Pădurean et al., 2025) selects a response for one user and prompt from a fixed pool by best-arm identification in a logistic bandit, and T-POP (Qu et al., 2026) steers decoding with a reward learned by a neural dueling bandit. Both learn each user from scratch, like our CoLSTIM-style baseline. UserAlign reports final-response win rate and query count, rather than cumulative regret on displayed pairs.

Simulated users and personalization benchmarks. LLM simulations can reproduce some human response patterns (Argyle et al., 2023; Aher et al., 2023), but may misrepresent group opinions and response variability (Santurkar et al., 2023; Bisbee et al., 2024). Prior work conditions simulations on personality profiles (Jiang et al., 2024), personal interviews (Park et al., 2024b), and student knowledge and personality (Lu & Wang, 2024; Liu et al., 2024). AlpacaFarm (Dubois et al., 2023), PERSONA (Castricato et al., 2025), and PersonalLLM (Zollo et al., 2025) provide simulated preference data. PRISM collects human preferences (Kirk et al., 2024), while LaMP evaluates personalization from user histories (Salemi et al., 2024). TutorDuel evaluates online pair selection by cumulative strong regret across short user sessions, using simulated preferences (Section 7).

3 PROBLEM SETTING

We consider users who interact with an LLM chatbot by asking a sequence of questions. For each question, the chatbot presents two alternative answers, and the user chooses the one they prefer.

We write $\lceil z \rceil$ for the smallest integer at least z, and vectors without boldface.

Users and population. A chatbot serves $u \geq 1$ users sequentially, indexed by $\mathcal{U} = [u] = \{1, \ldots, u\}$. Each user $i \in \mathcal{U}$ arrives once, completes one uninterrupted session of $b \geq 1$ questions, and leaves. Time is global: $t \in [T]$ with horizon $T = ub$ and current user $i_t = \lceil t/b \rceil$, so user i is served at rounds $t = (i-1)b + 1, \ldots, ib$.

Each user has a hidden preference vector $\theta_i \in \mathbb{R}^d$, drawn independently from a fixed, unknown population $\mathcal{P}$, $\theta_1, \ldots, \theta_u \overset{\text{i.i.d.}}{\sim} \mathcal{P}$, with $\|\theta_i\|_2 \leq B_\theta$ for a known bound $B_\theta > 0$. The vector θ_i stays fixed throughout the session. Examples include finite-type populations, Gaussian clouds, and low-rank Gaussian clouds (Section 4.2 and Appendix A), and, in the tutoring benchmark, students who combine a few shared preference traits (Section 4.3).

Personas, questions, and utilities. The chatbot has a finite catalog of K personas. In a tutoring example, a persona can be a system prompt specifying a style such as proofs, worked examples, or guiding questions. Persona $p \in [K]$ on question $q \in \mathcal{Q}$ produces an answer $x(p, q)$. For a given question q, the available arms are the K persona-question pairs (p, q). The learner selects which two personas to compare on that question.

The true utility of persona p on question q is that of the answer $x(p, q)$ it produces, written $f_i(p, q)$. The optimal persona may be different for each question. We assume a linear utility model (Saha, 2021; Bengs et al., 2022):

$$f_i(p, q) = \theta_i^\top \phi(p, q). \tag{1}$$

Here, $\phi(p, q) \in \mathbb{R}^d$ is a known vector encoding features of persona p in the context of question q, and θ_i encodes the user's preferences over these features. The features are available from the persona specification and question before generating the answer. Because ϕ depends on both persona and question, the preferred persona can vary with q even though θ_i is fixed.

At round t the environment supplies $q_t \in \mathcal{Q}$. Questions need not be independent or identically distributed and may be chosen adversarially based on past interactions. The current question and the K vectors $\phi(p, q_t)$ are observed before the pair is chosen.

Interaction. For each $t = 1, \ldots, T$, the learner observes i_t, q_t, and $\phi(p, q_t)$ for all $p \in [K]$. It then selects an ordered pair $(p_{t,1}, p_{t,2})$ of distinct personas and displays their answers. The user's choice is recorded as $y_t \in \{0, 1\}$: $y_t = 1$ if the answer from $p_{t,1}$ is chosen, and $y_t = 0$ if the answer from $p_{t,2}$ is chosen. There is one comparison per question. After the choice at $t = ib$, the session ends and its record joins the completed-user log. Let $\mathcal{H}_{t-1}$ collect observations from rounds $s < t$, with $\mathcal{H}_0 = \varnothing$. An admissible policy π chooses the current pair from this history, the current question and features, the persona specifications, the known parameters $d, K, B_\theta, \beta, b, u$, and its own randomness. Preference vectors, type labels, population parameters, and utilities remain hidden. Each policy learns from the comparisons it selected.

Because $\phi(p, q)$ is computed from the persona specification and the question, only the two selected answers need to be generated.

Preference feedback. We model the binary preference following the Bradley-Terry-Luce (BTL) model (Bradley & Terry, 1952; Luce, 1959; Bengs et al., 2022). Write $\mu(z) = (1 + e^{-z})^{-1}$. Conditional on the hidden vector, history, question, and selected pair,

$$\Pr\big(y_t = 1 \mid \theta_{i_t}, \mathcal{H}_{t-1}, q_t, p_{t,1}, p_{t,2}\big) = \mu\big(\beta\big[f_{i_t}(p_{t,1}, q_t) - f_{i_t}(p_{t,2}, q_t)\big]\big), \tag{2}$$

where $\beta > 0$ is a known parameter controlling feedback noise, shared across users and rounds. For a fixed nonzero utility gap, larger β makes the higher-utility answer more likely to be chosen, while smaller β moves the choice probability toward $1/2$. Under (1), each comparison is a logistic regression observation whose features are the difference between the two candidates' feature vectors. This law compares the true utilities of the two displayed answers and holds at every round, including adaptively selected comparisons.

Regret. For evaluation, utilities are normalized to $U_i(p, q) \in [0, 1]$ without changing persona rankings (Section 4.3). Let $p_t^\star \in \arg\max_{p \in [K]} U_{i_t}(p, q_t)$ be the best persona for user i_t on question q_t, with ties resolved by a fixed rule. We use strong regret (Yue et al., 2012), summing the utility gaps of both displayed answers as in Wang et al. (2025):

$$r_t = 2U_{i_t}(p_t^\star, q_t) - U_{i_t}(p_{t,1}, q_t) - U_{i_t}(p_{t,2}, q_t), \qquad R_T = \sum_{t=1}^{T} r_t. \tag{3}$$

Our problem is a multi-user variant of contextual dueling bandits (Saha, 2021; Bengs et al., 2022; Wang et al., 2025): the question determines the available arms' features, and the learner observes a pairwise choice. The objective is to minimize $\mathbb{E}_\pi[R_T]$ over admissible policies, counting every round of every user. Because two distinct personas must be shown, even a policy that knows θ_{i_t} pays the gap between the best and the second-best persona at every round. Summing these gaps gives the floor $G_T = \sum_{t=1}^{T} \big(U_{i_t}(p_t^\star, q_t) - \max_{p \neq p_t^\star} U_{i_t}(p, q_t)\big)$, so $R_T \geq G_T$. We define *excess strong regret* as $R_T - G_T$ and report the share of random pairs' excess regret that a policy removes (Section 4.1).

4 Experimental Setup

We evaluate CodueL in two environments: a numerical simulator in which the linear utility and BTL assumptions of Section 3 hold exactly, and a tutoring benchmark in which an LLM role-plays students described in natural language and compares generated tutor answers. All preferences are simulated. Pairwise choices made by two of the authors serve to validate the judge (Section 4.3).

4.1 Shared Protocol and Evaluation

In both environments we use $K = 6$ teaching personas, and u users arrive sequentially, each asking b questions ($b = 24$ in the tutoring benchmark). In the tutoring benchmark, we make every design choice, including the final variant of CodueL, on a development roster and run the evaluation rosters (Section 4.3) only after these choices are fixed. Every policy starts from the first user of each roster, with no preference estimates carried over from development, and regret includes every round.

Matched randomness. For each environment and seed, all policies see the same sequence of users and questions. They share one uniform draw per round to sample choices, using the BTL model (2) in the simulator and the stored judge probabilities in TutorDuel (Section 4.3). Identical ordered pairs therefore receive identical choices.

Benchmark score. In the tutoring benchmark we report the share of random pairs' excess regret that a policy removes, $100\,(R_T^{\mathrm{rand}} - R_T)/(R_T^{\mathrm{rand}} - G_T)$, where R_T^{rand} is the expected regret of random pairs. Random pairs score 0, and a policy would score 100 only by reaching the floor G_T on every question. The floor follows every question-specific preference, including judge disagreement between display orders, so no policy is expected to reach it. As a reference for what a learner can reach, we report a per-user linear oracle. For each user, the oracle fits a linear model to the judge's choice logits on all that user's questions. We form its features $\phi'(p, q)$ by appending a one-hot persona indicator to $\phi(p, q)$ and dividing the combined vector by $\sqrt{2}$ (5). On each question, it shows the two personas its fit ranks highest. It knows each user's best-fitting parameters from the first round, so a learner whose utility model is linear in ϕ' is not expected to exceed it; a policy using other features can.

4.2 Numerical Simulator

The numerical simulator evaluates CodueL where the linear utility and BTL assumptions hold exactly, with known features in $d = 8$ dimensions. Users are drawn from one of four populations: a few fixed types, a Gaussian cloud, a low-rank Gaussian cloud, or a null population with no shared structure, which serves as a negative control for transfer. We run $u = 300$ users with $b = 20$ and $\beta \in \{4, 12\}$ (Appendix A).

4.3 TUTORDUEL: TUTORING PREFERENCE DATASET

TutorDuel contains tutoring questions, generated tutor answers, pairwise choice probabilities from LLM-simulated students, and evaluation utilities. Policies receive persona-question features and sampled pairwise choices. The answer texts, probabilities, and utilities remain hidden from them.

Personas and questions. The six tutor personas are a rigorous prover, a worked-examples tutor, a Socratic guide, an intuition-and-analogy tutor, a concise cheat-sheet, and a patient step-by-step coach. Tutor answers are generated with `gpt-5.6-terra` and a fixed system prompt for each persona at temperature 1. The full tutor prompts are given in Appendix D.4.

We use `gpt-5.6-sol` to generate 24 distinct questions per session across four topics, each ending with a first-person cue stating the student's situation or need (Appendix D.3).

Persona-question features. We compute $\phi(p, q)$ from the persona prompt and the question text alone, before any answer is generated. Let $a(p, q)$ be the cosine similarity between OpenAI's `text-embedding-3-large` embeddings of the question and of persona p's system prompt, centered over the six personas, and σ_a its standard deviation on the development roster. With e_p the indicator vector of persona p, $\phi(p, q) = \big(1 + 0.25\, a(p, q)/\sigma_a\big)\, e_p \in \mathbb{R}^6$: each persona has its own coordinate, which grows when the question is close to that persona's prompt.

Simulated users. Student profiles describe background and preferences in English. We use eight hand-written profiles for development and eight held-out ones for evaluation (Appendix D.1). Composite and Quirky are two evaluation rosters whose profiles combine seven traits, including explanation style, answer length, and tone. Composite uses 350 distinct combinations among 3,072. Quirky draws traits uniformly and adds two of ten personal quirks, such as disliking numbered lists (Appendix D.2).

Judge. We use `gpt-6-luna`, with reasoning turned off, to role-play the student and compare two tutor answers. To account for sensitivity to answer order (Wang et al., 2024a) and A/B labels (Zheng et al., 2024a), we cover both orders and both label assignments for each profile, question, and persona pair (p, p'), sampling four single-letter choices per prompt. We align the four prompts' smoothed log-odds so that positive values favor p over p', then apply the sigmoid to their mean to obtain $P_i(p \succ p' \mid q)$. Appendix C.2 gives the calculation and diagnostics.

Agreement with human choices. Two authors judged which answer the described student would prefer, providing 200 judgments on 160 distinct comparisons. They saw the same student profile, question, and two tutor answers as the judge, with the answers in random order and the judge's output hidden. On the 40 comparisons judged by both (80 judgments), the judge agrees with each author (70.0% and 82.5%) as often as the authors agree with each other (72.5%). Details are given in Appendix C.4.

Evaluation utilities. Strong regret (3) scores both displayed answers against the best one, so it needs a utility for every persona. For each user i and question q, we fit one BTL score $\widehat{v}_p$ per persona to the 15 pairwise probabilities $P_i(p \succ p' \mid q)$ by ridge-regularized maximum likelihood (Appendix C.3), and set

$$U_i(p, q) = 1 - \big(\max_{p' \in [K]} \widehat{v}_{p'} - \widehat{v}_p\big)/\kappa, \tag{4}$$

with one constant $\kappa = 8$ logits for every user and roster, chosen on the development roster; the benchmark score does not depend on κ (Appendix C.3).

Rosters. The development roster has 300 user arrivals drawn from the eight development profiles. The test dataset contains 350 packs of 24 questions and their tutor answers, on topics disjoint from development. The held-out-topic roster assigns these packs to users drawn from the same eight profiles. Three further evaluation rosters reuse the packs with new students: the held-out-profile roster assigns each arrival one of the eight new profiles, and the Composite and Quirky rosters give each arrival its own student. Across seeds, we keep the assignment of students to question packs and vary the arrival order and the sampled clicks. As a negative control, we scramble the development roster by independently relabeling personas and rotating features for each user. The same relabeling applies to the choice probabilities and utilities throughout that user's session, preserving the user's comparison problem while breaking the alignment of persona preferences and feature directions across users.

4.4 References, Ablations, and Other Methods

References. Table 1 includes random pairs, the distinct-pair floor, and the per-user linear oracle (Section 4.1). Its non-personalized reference is the best fixed pair, chosen to minimize regret on the development roster and shown to every user and question.

Ablations and other methods. Our *no-transfer ablation* retains only CodueL's per-user Bayesian logistic learner (Section 5), starting each user from the same isotropic prior and learning only from that user's choices. Our *pooled baseline* fits one ridge-regularized logistic model to all completed sessions using ϕ' (5). It selects the two highest-scoring personas for each question and does not update within a session. Following work on learning priors across bandit tasks (Kveton et al., 2021; Hong et al., 2022a), our *Gaussian prior* comparator learns a single Gaussian over users' augmented preference vectors. After an initial delay (Appendix E), it refits the prior by empirical Bayes after each session. Within each session, it updates a posterior from this prior and selects the two personas with highest posterior mean utilities, without a gate or lookahead. Our *CoLSTIM-style* baseline (Bengs et al., 2022) learns each user independently and compares the persona of highest estimated utility with a challenger chosen by an upper confidence bound, without the original randomized utility perturbations. COLDB (Wang et al., 2025) groups users by estimated preferences and pools each group's comparisons, and our variant COLDB-LR assigns a user to a cluster by a likelihood-ratio test. RUCB (Zoghi et al., 2014) is a non-contextual dueling bandit with one preference matrix pooled over all users and questions. Random pairs select uniformly among distinct persona pairs.

5 Method

We propose CodueL (collaborative dueling), which learns from completed sessions which kinds of users recur and uses them to adapt faster to each new user. A *population model* describes users' preference vectors as a mixture of Gaussians fitted to the completed sessions. For a new user, this mixture is the prior of a *population learner*, which updates its posterior after every choice and selects pairs with a *lookahead pair rule*: it accepts a pair with higher regret now when the user's choice is expected to lower regret later in the session. An *evidence gate* lets the population learner select pairs only while it predicts completed users' choices better than a *no-transfer learner*, which learns each user alone and selects pairs otherwise. Both learners observe every comparison. Algorithm 1 in Appendix F.6 gives the full procedure.

No-transfer learner. This learner is Bayesian logistic regression on the current user's choices alone. Each session starts from the prior $\mathcal{N}(0, v_0 I_d)$ with $v_0 = B_\theta^2/d$. After each choice, we update a Laplace approximation of the posterior, a Gaussian centered at its mode (Bishop, 2006, Section 4.4), under the likelihood (2) with $\beta = 1$ in TutorDuel and the generating value in the simulator. The learner shows the two personas with the highest posterior mean utility $\phi(p, q_t)^\top \widehat{\theta}_t^{\mathrm{nt}}$, which minimizes the current round's expected strong regret under this posterior.

Population model. A user may prefer a persona on every question, which the contextual features alone need not capture. The population learner therefore adds the persona's indicator vector $e_p \in \mathbb{R}^K$ to the features,

$$\phi'(p, q) = \frac{[\phi(p, q)^\top, e_p^\top]^\top}{\sqrt{2}} \in \mathbb{R}^D, \qquad D = d + K. \tag{5}$$

and predicts utility as $w^\top \phi'(p, q)$, where $w \in \mathbb{R}^D$ holds the user's contextual weights and persona offsets. After n completed sessions, let λ_n^{EM} be the isotropic component's weight fitted by the expectation-maximization (EM)-style procedure below. The prior for a new user is the mixture

$$\Pi_n = \lambda_n \mathcal{N}(0, v_0 I_D) + (1 - \lambda_n) \sum_{k=1}^{M_n} \pi_k \mathcal{N}(\mu_k, \Sigma_k), \qquad \lambda_n = \max\left\{ \lambda_n^{\mathrm{EM}}, \ \frac{\alpha}{\alpha + n} \right\}. \tag{6}$$

Its M_n learned components, with weights π_k, means μ_k, and covariances Σ_k, describe recurring preference patterns. The isotropic component keeps prior mass for users unlike any seen so far. Its weight λ_n is at least $\alpha/(\alpha + n)$, the probability of a new cluster under a Dirichlet process with concentration α (Ferguson, 1973). Initially $\Pi_0 = \mathcal{N}(0, v_0 I_D)$.

After each session, we refit the mixture to all completed sessions using EM-style updates. We compute each user's responsibility for each component from a Laplace approximation of that user's marginal likelihood under it. We then fit each learned component by ridge-regularized logistic regression on the comparisons of all users, weighted by their responsibilities; the fit's mode and Laplace covariance give μ_k and Σ_k. We add a component when a session is better explained by the isotropic component than by any learned one, and remove weakly supported components (Appendix F.6).

Within-user adaptation. The population learner starts each new user from Π_n, which stays fixed during the session. After each choice, it updates a Laplace posterior within each component from the user's comparisons so far, and reweights the components by how well each explains these choices: each weight becomes proportional to the component's prior weight times its approximate marginal likelihood of the session. Before round t, component k has weight $\pi_{k,t}$, mean $m_{k,t}$, and covariance $C_{k,t}$, and this mixture is the learner's *belief* about the user. Component k predicts the utility $s_{k,t}(p;q) = m_{k,t}^\top \phi'(p,q)$ of persona p on question q, and the probability $\rho_{k,t}(p,p')$ that p is chosen over p' by averaging the likelihood (2) over its posterior (Appendix F.6).

Lookahead pair rule. Under component k, a pair (p,p') on question q has estimated strong regret $r_{k,t}(p,p';q) = 2\max_{a\in[K]} s_{k,t}(a;q) - s_{k,t}(p;q) - s_{k,t}(p';q)$. Showing the pair with the lowest belief-averaged regret $\bar{r}_t(p,p') = \sum_k \pi_{k,t}\, r_{k,t}(p,p';q_t)$ is greedy: it ignores what the user's choice will reveal. When components disagree, a pair whose outcome tells them apart can lower regret for the rest of the session, even if it costs more now. CodueL therefore scores each pair with a one-step lookahead. For each possible choice y, it updates the component weights by Bayes' rule to $\pi_{k,t}^{(y)}$, and estimates the regret of greedy pairs over the $h_t = i_t b - t$ remaining rounds of the session. Since future questions are unknown, it averages over a bank $\mathcal{Q}_t$ of the most recent questions across sessions, including q_t:

$$J_t(p,p') = \bar{r}_t(p,p') + h_t\, \mathbb{E}_y\left[\frac{1}{|\mathcal{Q}_t|}\sum_{q\in\mathcal{Q}_t}\min_{a\neq a'}\sum_k \pi_{k,t}^{(y)}\, r_{k,t}(a,a';q)\right], \qquad (7)$$
$$(p_{t,1},p_{t,2}) \in \arg\min_{p\neq p'} J_t(p,p'),$$

where the expectation over y uses the belief's predicted probability $\sum_k \pi_{k,t}\rho_{k,t}(p,p')$. Only the component weights change inside the lookahead; the component posteriors stay fixed. This is a knowledge-gradient rule (Frazier et al., 2008; Ryzhov et al., 2012) adapted to strong regret. In the last round $h_t = 0$, and the rule is greedy.

Evidence gate. A population fitted to users who share little with the next one can pull that user's estimate in the wrong direction. The gate therefore checks, on completed sessions, whether the population learner predicts choices better than the no-transfer learner. Before each choice is revealed, both learners predict it. For user i, let ℓ_i be the log-likelihood ratio of the population learner's predictions to the no-transfer learner's, summed over the user's choices. After each session, the gate opens if at least $n_{\min}$ sessions are complete and the mean of $\ell_1,\ldots,\ell_n$ exceeds z standard errors, and closes otherwise. The population learner's advantage can be small when the displayed answers have high but similar utilities, so we use a permissive gate, $z = 0.5$ and $n_{\min} = 8$, chosen on the development roster. All main comparisons use $\alpha = 20$, at most $M_{\max} = 16$ learned components, and a question bank of 16 questions (sensitivity in Appendix F).

6 Results

Throughout, we report the benchmark score of Section 4.1, the share of random pairs' excess regret that a policy removes. The tables also give the excess regret $R_T - G_T$.

6.1 Benchmark results

Table 1 reports every method on the three evaluation rosters, whose topics and students were never used in design. The held-out-profile roster follows eight new hand-written profiles, and the Composite and Quirky rosters each have 350 students composed from a template of traits, with two personal

method	held-out profiles		Composite		Quirky	
	excess	score	excess	score	excess	score
random pairs	2317.3	0	2276.1	0	2305.5	0
best fixed pair	2197.5	5	1900.0	17	1790.3	22
per-user linear oracle (ϕ')	511.4	78	698.4	69	672.1	71
distinct-pair floor G_T	0.0	100	0.0	100	0.0	100
RUCB (pooled, non-contextual)	1918.5 ± 5.9	17	1557.1 ± 8.1	32	1528.8 ± 7.0	34
Pooled	2028.9 ± 41.5	12	1768.6 ± 73.3	22	1799.7 ± 86.0	22
CoLSTIM-style	1517.4 ± 3.5	35	1621.3 ± 5.2	29	1616.9 ± 4.5	30
COLDB	1694.7 ± 14.9	27	1410.7 ± 9.9	38	1399.8 ± 9.0	39
COLDB-LR	1769.4 ± 6.4	24	1457.5 ± 5.4	36	1446.8 ± 11.4	37
Gaussian prior	1406.0 ± 11.4	39	1441.8 ± 17.3	37	1404.9 ± 18.6	39
CodueL	**1241.1 ± 8.3**	**46**	**1340.2 ± 7.3**	**41**	**1323.8 ± 8.4**	**43**
ablation: no transfer	1525.0 ± 4.2	34	1691.1 ± 5.1	26	1669.3 ± 5.6	28

Table 1: Benchmark results on the three rosters of new students on new topics ($u = 350$ arrivals each, $b = 24$, students and topics never used in design). Excess regret $R_T - G_T$ summed over the run (lower is better), mean $\pm$ s.e. over seeds (64 on the held-out profiles, 32 on Composite and Quirky; the s.e. treats the roster as fixed), and the benchmark score, the share of random pairs' excess regret that a policy removes (higher is better; Section 4.1). Italic rows are evaluation references (Section 4.4). The distinct-pair floor G_T, which scores 100, follows every question-specific preference, including judge disagreement between display orders, and is not an attainable target; the per-user linear oracle on ϕ' indicates what a learner linear in these features can reach. The last row is an ablation of CodueL (Section 6.1). The floor G_T, already subtracted from every entry: 779 (held-out profiles), 699 (Composite), 679 (Quirky).

quirks added in Quirky. Every policy starts from the first arrival with no preference data. CodueL has the lowest excess regret on all three rosters, with scores of 46, 41, and 43, against 39, 38, and 39 for the best comparison method. Rerun on Composite and Quirky rosters redrawn with replacement from the same 350 students, it scores 41 (35 to 44) and 42 (38 to 47), close to its scores on the actual rosters (Appendix E, Table 7).

Distance to the oracle. The per-user linear oracle, which fits each user's preferences on the same features in hindsight, removes 78%, 69%, and 71% of random pairs' excess regret, so CodueL reaches 59% to 60% of what it removes. On the held-out profiles, a reference that is told each user's profile and shows that profile's best pair for the whole session removes 75%, close to the oracle (Appendix E, Table 14). A fixed pair per profile therefore captures nearly all of the linear oracle's aggregate improvement.

New topics and new students. On the held-out-topic roster, which keeps the development profiles and changes only the questions, CodueL scores 50; new students lower its score by 4 to 9 points (Appendix E).

Comparison methods. The closest methods are the Gaussian prior on the held-out profiles, where CodueL has 11.7% less excess regret, and COLDB on Composite and Quirky, with 5.0% and 5.4%. Like CodueL, both share information across users while adapting to each user. Methods that learn each user separately, such as CoLSTIM-style, score close to the no-transfer ablation. Methods that fit one model to all users, the pooled baseline and RUCB, show every student similar pairs and score lowest on the held-out profiles. Appendix E gives the paired comparisons (Table 8) and the tuning grids, evaluated on development seeds used in no reported result (Table 13).

Ablations. The learned population matters most: without it, excess regret rises by 22% to 25%. Choosing pairs greedily instead of by the one-step lookahead raises it by 2% to 8%. Removing the gate raises excess regret by 3% on the held-out profiles and changes it within noise on Composite and Quirky. Appendix F, Table 16, reports the full results of these component changes on the three evaluation rosters.

6.2 Learning over the arrivals

Figure 2 in Appendix E shows how cumulative excess regret grows as users arrive. Table 1 sums over the whole run, including the first users, for whom CodueL has no population to learn from. Over the whole run, CodueL has 18.6%, 20.7%, and 20.7% less excess regret than its no-transfer ablation, which removes the population model, the gate, and the persona offsets (last row of Table 1; every 95% interval lies above 16.9%, Appendix B). The ablation's average excess regret per user stays approximately constant. CodueL shows the ablation's pairs until the gate turns transfer on, after at least eight users, and then improves as its mixture forms: its excess regret per user is 10% to 12% below the ablation's in the first quarter of the arrivals and 23% to 25% below in the last (Appendix E, Table 11).

6.3 Negative controls and the simulator

A population model can increase regret when its learned patterns do not apply to a new user. We test CodueL on two negative controls (Appendix F, Table 18): the simulator's isotropic null population and a scrambled development roster that breaks the alignment of persona preferences and feature directions across students while preserving each student's comparison problem. On both, CodueL's excess regret stays within half a percent of its no-transfer ablation's, with 95% intervals that include zero. Without its gate, the same policy raises excess regret by 28% and 6% (Appendix F, Table 19). These controls were held out during development; the results do not guarantee protection under other distribution shifts.

Simulator. Where linear utilities and BTL choices hold exactly, CodueL gains most when users form a few types and least when they form a Gaussian cloud, for which a Gaussian prior is a natural approximation and has lower excess regret on 9 of the 16 structured settings (Appendix E, Table 12).

7 Limitations and conclusion

Simulated preferences and generalization. In TutorDuel, reference utilities are fitted from the same LLM judge's pairwise probabilities that generate feedback. Human judgments are limited to two authors' choices on 160 comparisons (Appendix C.4); the judge agrees with them about as often as they agree with each other. On a 300-arrival Composite subset, `gpt-5.6-sol` and `gpt-6-luna` favor the same answer in 82.8% of comparisons excluding ties, and CodueL ranks first under both judges (Appendix C.5). We have not measured human satisfaction or educational outcomes. What students prefer need not be what best supports learning (Bjork & Bjork, 2011; Pashler et al., 2008; Puech et al., 2025). The benefits of transfer vary across held-out profiles, so average gains do not imply improvements for every profile (Table 6). With our features, the per-user linear oracle only slightly outperforms the best fixed pair per user (Table 14). This motivates testing whether richer features can better capture question-dependent preferences.

Method and evaluation. We assume a stationary population and preferences fixed within each session. Linear utilities and BTL feedback approximate the judge's preferences, and Laplace inference and lookahead are approximate (Section 5). We provide no regret bound and no guarantees under preference drift, misspecified population models, or shifts in the question distribution. The gate's repeatedly checked threshold gives no anytime-valid guarantee against harmful transfer. Reported intervals do not account for the adaptive development process, and component studies on the development roster are exploratory (Appendix F).

Conclusion. We introduced TutorDuel, a benchmark for online LLM personalization from pairwise choices, and CodueL, a baseline that learns a population prior from completed sessions. On students and topics held out from development, CodueL removes 41–46% of random pairs' excess regret, more than the other evaluated methods and about three fifths of what a per-user linear oracle removes. Closing this gap and validating with human users are next steps.

Ethics Statement

This work involves no human participants other than the authors. The student preferences in TutorDuel are simulated: an LLM role-plays students described by short written profiles, and the only

human judgments are choices made by two authors to check the judge (Appendix C.4). We collected no personal data. Simulated preferences can differ from those of real students, and an LLM judge can carry biases from its training data, for example by associating a profile's stated background with a teaching style. We therefore present TutorDuel as a testbed for learning algorithms rather than as evidence of what real students prefer. The system we study adapts the style of answers to the choices a student makes, and the style a student prefers need not be the one that best supports learning (Bjork & Bjork, 2011; Pashler et al., 2008). A deployed tutor should combine preference feedback with measures of learning and let students see and change the persona serving them. The questions and answers in the dataset were generated by LLMs and may contain errors, so we release them for research on personalization algorithms and not as teaching material.

Reproducibility Statement

The supplementary material contains the TutorDuel dataset, the code and commands that rerun every experiment in the paper, the tuning grids and selection rule of the comparison methods, and a script that computes the benchmark scores and confidence intervals from the run results. The dataset stores the questions, the tutor answers, the persona-question features, and the judges' sampled choices and probabilities, including those of the second judge (Appendix C.5), so every experiment runs on CPUs without calling an LLM API. Rebuilding the dataset itself would require the LLMs named in Section 4. That section describes how the dataset was built, and Appendices C and D give the tutor prompts, the question cues, the student profiles and templates, the judge's probability calculation, and the fit of the evaluation utilities. Section 5 and Appendix F.6 specify CodueL, with pseudocode in Algorithm 1. Appendix A defines the simulated environments, Appendix E lists the grids used to tune the comparison methods, and Appendix B describes the confidence intervals. All policies share the same users, questions, and random draws for a given seed, and every setting of CodueL was fixed on the development roster.

Use of Large Language Models

LLMs are part of the data and the evaluation of this work. `gpt-5.6-sol` generated the questions, `gpt-5.6-terra` generated the tutor answers, `gpt-6-luna` role-plays the students and provides the pairwise judgments, and OpenAI's `text-embedding-3-large` embeds the questions and persona prompts for the features (Section 4). `gpt-5.6-sol` and `claude-opus-5` serve as comparison judges (Appendix C). An LLM drafted the tutor persona prompts, the student profiles, the trait templates, and the question cues, and the authors edited them and selected the final versions. We used Claude Code, Cursor, and ChatGPT and Codex to implement the benchmark, the comparison methods, and CodueL, and to run and analyze experiments. Coding agents also proposed and tested variants of CodueL on the development roster and the tuning seeds, under an evaluation protocol that the authors fixed in advance, and the authors selected the reported configuration. LLMs drafted and edited parts of the text, and the authors revised and approved every sentence. LLMs also helped search for related work and check the cited references. The authors defined the problem and the evaluation protocol, reviewed all code and text, and checked every reported number with scripts that recompute it from the stored results. The authors take full responsibility for the content of the paper.

References

Gati V. Aher, Rosa I. Arriaga, and Adam Tauman Kalai. Using large language models to simulate multiple humans and replicate human subject studies. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *PMLR*, pp. 337–371, 2023.

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. Out of one, many: using language models to simulate human samples. *Political Analysis*, 31(3):337–351, 2023.

Soumya Basu, Branislav Kveton, Manzil Zaheer, and Csaba Szepesvári. No regrets for learning the prior in bandits. In *Advances in Neural Information Processing Systems*, volume 34,

pp. 28029–28041, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/ec1f764517b7ffb52057af6df18142b7-Abstract.html.

Viktor Bengs, Róbert Busa-Fekete, Adil El Mesaoudi-Paul, and Eyke Hüllermeier. Preference-based online learning with dueling bandits: a survey. *Journal of Machine Learning Research*, 22 (7):1–108, 2021.

Viktor Bengs, Aadirupa Saha, and Eyke Hüllermeier. Stochastic contextual dueling bandits under linear stochastic transitivity models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *PMLR*, pp. 1764–1786, 2022.

James Bisbee, Joshua D. Clinton, Cassy Dorff, Brenton Kenkel, and Jennifer M. Larson. Synthetic replacements for human survey data? The perils of large language models. *Political Analysis*, 32 (4):401–416, 2024.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Elizabeth L. Bjork and Robert A. Bjork. Making things hard on yourself, but in a good way: creating desirable difficulties to enhance learning. In Morton A. Gernsbacher, Richard W. Pew, Leaetta M. Hough, and James R. Pomerantz (eds.), *Psychology and the Real World: Essays Illustrating Fundamental Contributions to Society*, pp. 56–64. Worth Publishers, 2011.

Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. Gaussian process preference elicitation. In *Advances in Neural Information Processing Systems*, volume 23, pp. 262–270, 2010.

Avinandan Bose, Zhihan Xiong, Yuejie Chi, Simon Shaolei Du, Lin Xiao, and Maryam Fazel. LoRe: personalizing LLMs via low-rank reward modeling. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=bYu4DOqRY8.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Hongru Cai, Yongqi Li, Tiezheng Yu, Fengbin Zhu, Wenjie Wang, Fuli Feng, and Wenjie Li. One adapts to any: meta reward modeling for personalized LLM alignment. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 110–121, 2026.

Louis Castricato, Nathan Lile, Rafael Rafailov, Jan-Philipp Fränken, and Chelsea Finn. PERSONA: a reproducible testbed for pluralistic alignment. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 11348–11368, 2025.

Leonardo Cella, Alessandro Lazaric, and Massimiliano Pontil. Meta-learning with stochastic linear bandits. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *PMLR*, pp. 1360–1370, 2020.

Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, and Ramya Korlakai Vinayak. PAL: sample-efficient personalized reward modeling for pluralistic alignment. In *The Thirteenth International Conference on Learning Representations*, pp. 15755–15799, 2025.

Nirjhar Das, Souradip Chakraborty, Aldo Pacchiano, and Sayak Ray Chowdhury. Active preference optimization for sample efficient RLHF. In *Machine Learning and Knowledge Discovery in Databases. Research Track (ECML PKDD 2025)*, pp. 96–112. Springer, 2025. doi: 10.1007/978-3-032-06096-9_6.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. AlpacaFarm: a simulation framework for methods that learn from human feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 30039–30069, 2023.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled AlpacaEval: a simple way to debias automatic evaluators. In *First Conference on Language Modeling*, 2024.

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

Peter I. Frazier, Warren B. Powell, and Savas Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *PMLR*, pp. 757–765, 2014. URL https://proceedings.mlr.press/v32/gentile14.html.

Joey Hong, Branislav Kveton, Manzil Zaheer, Yinlam Chow, Amr Ahmed, and Craig Boutilier. Latent bandits revisited. In *Advances in Neural Information Processing Systems*, volume 33, pp. 13423–13433, 2020.

Joey Hong, Branislav Kveton, Manzil Zaheer, and Mohammad Ghavamzadeh. Hierarchical Bayesian bandits. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *PMLR*, pp. 7724–7741, 2022a.

Joey Hong, Branislav Kveton, Manzil Zaheer, Mohammad Ghavamzadeh, and Craig Boutilier. Thompson sampling with a mixture prior. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *PMLR*, pp. 7565–7586, 2022b. URL https://proceedings.mlr.press/v151/hong22b.html.

Neil Houlsby, Jose Miguel Hernández-Lobato, Ferenc Huszár, and Zoubin Ghahramani. Collaborative Gaussian processes for preference learning. In *Advances in Neural Information Processing Systems*, volume 25, pp. 2096–2104, 2012.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. Personalized soups: personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

Hang Jiang, Xiajie Zhang, Xubo Cao, Cynthia Breazeal, Deb Roy, and Jad Kabbara. PersonaLLM: investigating the ability of large language models to express personality traits. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 3605–3627, 2024.

Hannah Rose Kirk, Alexander Whitefield, Paul Röttger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, Bertie Vidgen, and Scott A. Hale. The PRISM alignment dataset: what participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, volume 37, pp. 105236–105344, 2024. doi: 10.52202/079017-3342. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/be2e1b68b44f2419e19f6c35a1b8cf35-Abstract-Datasets_and_Benchmarks_Track.html.

Branislav Kveton, Mikhail Konobeev, Manzil Zaheer, Chih-Wei Hsu, Martin Mladenov, Craig Boutilier, and Csaba Szepesvári. Meta-Thompson sampling. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *PMLR*, pp. 5884–5893, 2021. URL https://proceedings.mlr.press/v139/kveton21a.html.

Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 539–548, 2016.

Xinyu Li, Ruiyang Zhou, Zachary C. Lipton, and Liu Leqi. Personalized language modeling from personalized human feedback. *arXiv preprint arXiv:2402.05133*, 2024a.

Xuheng Li, Heyang Zhao, and Quanquan Gu. Feel-good Thompson sampling for contextual dueling bandits. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pp. 29406–29426, 2024b.

Zhengyuan Liu, Stella Xin Yin, Geyu Lin, and Nancy F. Chen. Personality-aware student simulation for conversational intelligent tutoring systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 626–642, 2024.

Xinyi Lu and Xu Wang. Generative Students: using LLM-simulated student profiles to support question item evaluation. In *Proceedings of the Eleventh ACM Conference on Learning @ Scale*, pp. 16–27, 2024.

R. Duncan Luce. *Individual Choice Behavior: A Theoretical Analysis*. John Wiley & Sons, 1959.

Odalric-Ambrym Maillard and Shie Mannor. Latent bandits. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *PMLR*, pp. 136–144, 2014. URL https://proceedings.mlr.press/v32/maillard14.html.

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. URL https://arxiv.org/abs/2303.08774.

Victor-Alexandru Pădurean, Parameswaran Kamalaruban, Nachiket Kotalwar, Alkis Gotovos, and Adish Singla. Inference-time personalized alignment with a few user preference queries. In *Advances in Neural Information Processing Systems*, volume 38, pp. 85125–85156, 2025.

Chanwoo Park, Mingyang Liu, Dingwen Kong, Kaiqing Zhang, and Asuman Ozdaglar. RLHF from heterogeneous feedback via personalization and preference aggregation. *arXiv preprint arXiv:2405.00254*, 2024a.

Joon Sung Park, Carolyn Q. Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S. Bernstein. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109v1*, 2024b.

Harold Pashler, Mark McDaniel, Doug Rohrer, and Robert Bjork. Learning styles: concepts and evidence. *Psychological Science in the Public Interest*, 9(3):105–119, 2008.

Amit Peleg, Naama Pearl, and Ron Meir. Metalearning linear bandits by prior update. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *PMLR*, pp. 2885–2926, 2022.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing reinforcement learning from human feedback with variational preference learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52516–52544, 2024. doi: 10.52202/079017-1664. URL https://papers.nips.cc/paper_files/paper/2024/hash/5e1c255653eb98cef13f45b2d337c882-Abstract-Conference.html.

Romain Puech, Jakub Macina, Julia Chatain, Mrinmaya Sachan, and Manu Kapur. Towards the pedagogical steering of large language models for tutoring: a case study with modeling productive failure. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 26291–26311, 2025.

Zikun Qu, Min Zhang, Mingze Kong, Xiang Li, Zhiwei Shang, Zhiyong Wang, Yikun Ban, Shuang Qiu, Yao Shu, and Zhongxiang Dai. T-POP: test-time personalization with online preference feedback. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Ilya O. Ryzhov, Warren B. Powell, and Peter I. Frazier. The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195, 2012.

Aadirupa Saha. Optimal algorithms for stochastic contextual preference bandits. In *Advances in Neural Information Processing Systems*, volume 34, pp. 30050–30062, 2021.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: when large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7370–7392, 2024.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. Whose opinions do language models reflect? In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *PMLR*, pp. 29971–30004, 2023.

Max Simchowitz, Christopher Tosh, Akshay Krishnamurthy, Daniel Hsu, Thodoris Lykouris, Miroslav Dudík, and Robert E. Schapire. Bayesian decision-making under misspecified priors with applications to meta-learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 26382–26394, 2021.

David H. Stern, Ralf Herbrich, and Thore Graepel. Matchbox: large scale online Bayesian recommendations. In *Proceedings of the 18th International Conference on World Wide Web*, pp. 111–120, 2009.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999. doi: 10.1111/1467-9868.00196.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450. Association for Computational Linguistics, 2024a. doi: 10.18653/v1/2024.acl-long.511. URL https://aclanthology.org/2024.acl-long.511/.

Xinpeng Wang, Bolei Ma, Chengzhi Hu, Leon Weber-Genzel, Paul Röttger, Frauke Kreuter, Dirk Hovy, and Barbara Plank. "My answer is C": first-token probabilities do not match text answers in instruction-tuned language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 7407–7416. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.findings-acl.441. URL https://aclanthology.org/2024.findings-acl.441/.

Zhiyong Wang, Jiahang Sun, Mingze Kong, Jize Xie, Qinghua Hu, John C. S. Lui, and Zhongxiang Dai. Online clustering of dueling bandits. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *PMLR*, pp. 64403–64431, 2025.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: bridging theory and practice for RLHF under KL-constraint. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pp. 54715–54754, 2024.

Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The K-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*, pp. 19426–19454, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/54dd9e0cff6d9214e20d97eb2a3bae49-Abstract-Conference.html.

Mingqian Zheng, Jiaxin Pei, Lajanugen Logeswaran, Moontae Lee, and David Jurgens. When "A helpful assistant" is not really helpful: personas in system prompts do not improve performances of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15126–15154. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.findings-emnlp.888. URL https://aclanthology.org/2024.findings-emnlp.888/.

Banghua Zhu, Michael I. Jordan, and Jiantao Jiao. Principled reinforcement learning with human feedback from pairwise or K-wise comparisons. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *PMLR*, pp. 43037–43067, 2023.

Masrour Zoghi, Shimon Whiteson, Rémi Munos, and Maarten de Rijke. Relative upper confidence bound for the K-armed dueling bandit problem. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *PMLR*, pp. 10–18, 2014. URL https://proceedings.mlr.press/v32/zoghi14.html.

Thomas P. Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. PersonalLLM: tailoring LLMs to individual preferences. In *The Thirteenth International Conference on Learning Representations*, pp. 66949–66971, 2025.

A SIMULATOR CONSTRUCTIONS

The simulator uses linear utilities and logistic pairwise feedback, as in the contextual preference-bandit models of Saha (2021) and Bengs et al. (2022). Our $\phi(p, q_t)$ corresponds to the contextual feature vector for arm p at round t. The feature construction and user populations below are specific to our simulator. Implementation details and reported configurations are available in the accompanying code (`persona_bench/world.py` and `results/worlds.py`).

We use $d = 8$, $K = 6$, and $B_\theta = 1$. A question is a unit vector $z_q \in \mathbb{R}^d$. Each persona has a matrix $B_p \in \mathbb{R}^{d\times d}$ with independent $\mathcal{N}(0, 1/d)$ entries, drawn once per world and then fixed. The matrix B_p transforms the shared question vector into features specific to persona p, allowing persona rankings to vary across questions. We normalize the resulting vector to obtain the feature map

$$\phi(p, q) = \frac{B_p z_q}{\|B_p z_q\|_2}. \tag{8}$$

Utilities and choices follow (1) and (2); policies receive the true B_θ, d, and β. Default questions are i.i.d. directions on the sphere.

Write $\Pi(v) = v/\max\{1, \|v\|_2/B_\theta\}$ for projection onto the norm ball. Users are sampled independently. The learner is not told which population is active.

Finite types. For $m \in \{2, 4, 8\}$, user i draws a type h_i uniformly from $[m]$ and has $\theta_i = \theta^{h_i}$. The type vectors have norm B_θ and equal pairwise angles, either $60°$ or $90°$, and are constructed from a random orthonormal frame.

Gaussian cloud. We draw $v_i \sim \mathcal{N}(\mu_{\text{pop}}, s^2 C_d)$ and set $\theta_i = \Pi(v_i)$, with $s = 0.3$. The center has norm $B_\theta/2$ and a uniformly random direction. The random positive-definite matrix C_d is normalized to have trace d.

Low-rank cloud. We use this population to test transfer when variation across users is concentrated in a shared low-dimensional subspace. The latent directions represent correlated changes in users' preference weights. Before projection, the covariance has the standard low-rank plus isotropic noise form of probabilistic principal component analysis (Tipping & Bishop, 1999). We use $r = 2$ and $s = 1$, drawing

$$v_i \sim \mathcal{N}\big(\mu_{\text{pop}}, s^2\left(VC_rV^\top + 0.05^2 I_d\right)\big), \qquad \theta_i = \Pi(v_i). \tag{9}$$

Here $V \in \mathbb{R}^{d\times r}$ has orthonormal columns spanning a uniformly random subspace, C_r is positive definite with trace r, and μ_{pop} has norm $B_\theta/2$ and lies in that subspace. The isotropic component has standard deviation $0.05s$ and allows small deviations outside the shared subspace, so the full covariance is not rank deficient.

Null population. Users are drawn from $\mathcal{N}(0, (B_\theta^2/d)I_d)$ and projected with Π. This isotropic negative control contains no additional population structure.

World screening. On 256 independently sampled diagnostic questions we measure whether users disagree on their favorite persona. We regenerate a world if fewer than 5% of questions elicit any disagreement. For finite types the check uses every type; for either cloud or the null population it uses 64 independent population draws. These diagnostic observations are not supplied to the policies.

Reported configurations. Each configuration uses $u = 300$ users. Table 12 reports the 18 anchor configurations at $b = 20$ and $\beta \in \{4, 12\}$, including the two null configurations. These experiments use i.i.d. questions. Appendix E summarizes a broader grid over session length b and click consistency β.

B UNCERTAINTY ESTIMATION

For simulator paired gains, we report 95% Student's t confidence intervals across seeds. For the main tutoring comparisons, we report confidence intervals from a crossed bootstrap that independently resamples seeds and users. On rosters with recurring profiles (the held-out-profile roster, the held-out-topic roster and the development roster), users are resampled within profiles; on the Composite and Quirky rosters every user has a profile of their own, and users are resampled without strata.

Bootstrap intervals use the 2.5th and 97.5th percentiles, with $2{,}000$ replicates. This calculation reuses recorded per-user regret and does not rerun the adaptive policies. The intervals are therefore conditional on the evaluated students: a user's regret depends on the population learned from earlier users, and resampling users after the fact does not simulate a different population. On the held-out-profile roster users are resampled within its eight profiles and the profiles are not resampled; Table 6 reports results per profile, and Appendix E reports a range that also resamples profiles after the fact. Standard errors in the tables are over seeds, with the roster fixed. Table 7 reruns the policies on a redrawn roster for every seed, which covers the choice of students as well; on the held-out-profile roster, drawing eight profiles with replacement leaves fewer distinct profiles per roster, so that table describes the spread of such rosters rather than an unbiased estimate of the actual roster's score. Exploratory studies (Appendix F) use standard errors or paired seed-based t intervals, as their captions specify.

statistic	development	held-out topics	held-out profiles	Composite	Quirky
arrivals	300	350	350	350	350
distinct profiles	8	8	8	350	350
share of centred score variance (%)					
shared persona effects	18.0	20.5	7.1	13.1	13.0
profile–persona differences	31.7	28.2	50.3	30.4	34.5
sessions within profiles	2.0	2.1	1.9	0.0	0.0
question deviations within sessions	48.3	49.1	40.7	56.5	52.5
mean preference concentration (nats)	0.231	0.231	0.226	0.225	0.228
judgment/utility disagreement (%)	2.8	2.8	2.4	3.4	3.2

Table 2: Dataset statistics by roster (24 questions per arrival). Variance shares decompose question-centred Bradley–Terry scores into a shared persona effect, profile deviations, session deviations within a profile and question deviations within a session; the question deviations include the judge's disagreement between display orders (Appendix C.2). On the Composite and Quirky rosters every profile has one arrival, so profile and session deviations coincide. Preference concentration is the KL divergence from an unbiased choice, measured in nats (units based on natural logarithms). Concentration and disagreement average over all 15 unordered persona pairs per question.

C Dataset characteristics and quality

These diagnostics describe the development roster, the held-out-topic roster, and the three rosters of new students: the held-out-profile roster, on which every arrival comes from the eight new hand-written profiles, and the Composite and Quirky rosters, on which every arrival has a profile of its own. They characterize the simulated preferences; Appendix C.4 compares the judge's choices with human choices.

C.1 Roster statistics

We center each question's six Bradley–Terry scores and decompose them into a persona effect shared by all users, the deviation of each profile's mean from it, the deviation of each session's mean from its profile mean, and the deviation of each question's scores from its session mean (Table 2). On the development roster, profile–persona differences account for 31.7% of the centred score variance and question deviations within sessions for 48.3%: the cue at the end of each question moves the preferred persona within a session, and part of these deviations is the judge's disagreement between display orders (Appendix C.2). The new profiles on the held-out-profile roster differ more from one another than the development profiles do (50.3% against 31.7%). On the Composite and Quirky rosters each profile has a single session, so profile and session deviations cannot be separated; together they account for 30.4% and 34.5%. Preference concentration is the KL divergence of each stored Bernoulli choice distribution from an unbiased choice, $P\log(2P)+(1-P)\log\big(2(1-P)\big)$ for a pair with preference probability P, averaged over questions and persona pairs in nats. Judgment/utility disagreement is the fraction of comparisons in which the stored preference probability favors the persona with lower fitted utility; it measures how far the judge's pairwise probabilities are from their Bradley–Terry approximation.

C.2 Judge diagnostics

Choice probabilities. We derive probabilities from sampled choices rather than token probabilities (Wang et al., 2024b), which a chat model need not keep calibrated after post-training (OpenAI, 2023, Figure 8). If n_A of a prompt's four choices are `A`, its smoothed log-odds are $\log\big((n_A+\frac{1}{2})/(4-n_A+\frac{1}{2})\big)$. We reverse the sign when `A` denotes p', average over the four prompts, and apply the sigmoid to obtain $P_i(p \succ p' \mid q)$.

The benchmark's choice probabilities are measurements made by the judge. Table 3 characterizes them on the 2,448,000 stored prompts of all 24 questions of every arrival.

Answer format and sampling. All sampled choices were valid labels (`A` or `B`). The judge's four choices for a prompt agree on 73% to 79% of prompts, whose log-odds then sit at the cap of $\log 9$, and 38% to 41% of persona pairs have all 16 choices on the same side, so the strongest preferences

diagnostic	development	held-out topics	held-out profiles	Composite	Quirky
sampled choices that are a valid label (%)	100.0	100.0	100.0	100.0	100.0
prompts whose 4 choices agree (%)	77	78	79	73	74
persona pairs with all 16 choices on one side (%)	40	41	41	38	39
reliability: split of each prompt's choices	0.99	0.99	0.99	0.99	0.99
lean toward the first-shown answer (logits)	+0.38	+0.38	+0.48	+0.24	+0.24
lean toward the label A (logits)	+0.22	+0.21	+0.13	+0.22	+0.23
pairs on which all four prompts agree in sign (%)	54	55	54	54	54
correlation, the two label assignments	0.89	0.89	0.90	0.89	0.89
correlation, the two display orders	0.72	0.70	0.65	0.71	0.70
reliability: question deviations within a session	0.86	0.86	0.82	0.87	0.86
reliability: student profiles	0.96	0.95	0.94	0.96	0.96
length: logit change per s.d. of the length ratio	-0.101	-0.090	-0.161	-0.147	-0.147
length: longer answer preferred (%)	48	50	44	52	50

Table 3: Diagnostics of the judge (`gpt-6-luna`, 4 sampled choices per prompt) on the stored judgments of all 24 questions of every arrival (four prompts per persona pair and question). A prompt's log-odds is capped at $\log 9$ when its four choices agree. Correlations are between the logits averaged over one half of the prompts and over the other half, across persona pairs. Reliabilities are Spearman–Brown estimates for the four-prompt average, from Bradley–Terry scores fitted separately on the two display orders: of each question's scores minus the session's mean scores, and of each session's mean scores after removing the persona means. Length: log ratio of the two answers' word counts against the four-prompt logit, both centred within student profile and persona pair.

are compressed to a probability of about 0.9. Splitting each prompt's four choices into two halves gives scores that correlate closely (reliability of the full average 0.99 to 0.99): the sampling of choices adds little noise.

Agreement of the four prompts. Swapping which answer is labelled A changes the judgment little: the log-odds of the two label assignments correlate 0.89 to 0.90, although the judge leans 0.13 to 0.23 toward the answer labelled A. The display order matters more. The judge leans 0.24 to 0.48 toward the answer shown first, the log-odds of the two orders correlate 0.65 to 0.72, and all four prompts agree in sign on 54% to 55% of persona pairs. Averaging over the two orders and labels removes the constant leans but not the disagreement specific to each comparison. With scores fitted separately on the two orders, the reliability of the four-prompt average is 0.94 to 0.96 for a student's profile over a session and 0.82 to 0.87 for the deviations of each question from its session. Question-specific deviations are less consistent across answer orders than session averages.

Length. Brevity is part of the concise persona's style. The answer instruction asks it for 40 to 90 words and the other personas for 100 to 180, and lets every tutor leave that range when the student's cue asks for it (Appendix D.4); on the held-out questions its answers have a median of 107 words, against 147 to 169 for the other personas. LLM judges can favor longer answers (Dubois et al., 2024). Holding the persona pair and the student profile fixed, our judge instead slightly prefers the shorter answer (-0.161 to -0.090 logits per standard deviation of the log length ratio). Across all comparisons, which mix length with the personas' styles, the longer answer is preferred in 44% to 52% of them.

Consequences for the benchmark. All policies face the same fixed table of choice probabilities, including its display-order effects and capped log-odds. Some question-specific variation may be difficult to predict from our features $\phi(p, q)$, so excess regret can reflect limitations of the representation as well as learning.

C.3 Evaluation utilities

We chose κ above the largest fitted score range on the development roster. A common scale keeps differences in preference strength: a question on which the student barely distinguishes the personas contributes less regret than one on which the favorite is far ahead. The benchmark score does not depend on κ. We score regret on U and draw clicks from P, so fitting utilities does not change the feedback. The learners use $\beta = 1$ and a prior scale $B_\theta = 2.08$ fitted on the development roster (Appendix F.6).

Table 4: Agreement of the judges with the authors' choices on the 160 labelled comparisons (a comparison labelled by both authors counts once, with the two choices averaged). Brackets: 95% intervals of a bootstrap over the labeled (student, question) cells. (a) The dataset's judge against the authors; the judge's favourite is its more probable answer, and its confidence is the probability of that answer. (b) On the comparisons both authors labelled, each author against the other and against the judge. (c) Other judges run on the same comparisons with the same four prompts; same favourite: share of comparisons on which the judge favours the same answer as the dataset's judge.

(a) Dataset judge against the authors

Agreement, all comparisons (%)	70.7 [63.4, 77.7]
Cohen's κ	0.43 [0.30, 0.57]
AUC of the judge's log-odds	0.76 [0.67, 0.83]

(b) The 40 comparisons labelled by both authors

Pair	Agreement (%)	Cohen's κ
Author 1 and author 2	72.5	0.45
Judge and author 1	70.0	0.40
Judge and author 2	82.5	0.65
Judge, where the authors agree (29)	86.2	

(c) Judges

Judge	Agreement (%)	Same favourite (%)
`gpt-6-luna` (this dataset)	70.7 [63.4, 77.7]	–
`gpt-5.6-sol` (chat)	73.8 [66.0, 81.1]	87.0
`claude-opus-5` (chat)	75.0 [68.2, 81.4]	76.5

We fit the BTL scores of Section 4.3 by

$$\widehat{v} = \arg\min_{v \in \mathbb{R}^K} \left\{ - \sum_{1 \le p < p' \le K} \Big[P_{pp'} \log \mu(v_p - v_{p'}) + (1 - P_{pp'}) \log \mu(v_{p'} - v_p) \Big] + \frac{0.05}{2} \|v\|_2^2 \right\}. \tag{10}$$

All 15 unordered pairs receive equal weight. The ridge term keeps scores finite when probabilities are near 0 or 1.

C.4 Agreement with human choices

Labelled comparisons. We drew 300 comparisons: 75 (student, question) cells, one question per arrival, spread over the five rosters and, within the development, held-out-topic and held-out-profile rosters, balanced over profiles; in each cell 4 of the 15 persona pairs, chosen so that the sample covers the judge's range of confidence evenly. The first 10 cells were labelled by both authors, the others by one. The interface showed the student profile, the question with its cue, and the two answers, exactly as in the judge's prompt, in an independently randomized display order for each author, and asked which answer this student would rather receive; a choice was required. It never showed the judge's output, the persona names, or the other author's choice. The authors made 200 choices (100 and 100) on 160 distinct comparisons. Throughout, a comparison labeled by both authors counts once, with the two choices averaged. We report 95% intervals from a bootstrap over (student, question) cells, since comparisons within a cell share a student and a question.

The judge agrees with each author as often as the authors agree with each other. The fair reference for the judge is a second human who sees the same comparison. On the 40 comparisons labelled by both authors (Table 4b), the authors agree with each other on 72.5% (Cohen's $\kappa = 0.45$), and the judge agrees with them on 70.0% and 82.5% ($\kappa = 0.40$ and 0.65). Where the two authors agree, the judge picks their answer in 25 of 29 comparisons. Disagreement with the judge is thus of the same size as disagreement between two careful readers of the same profile: which of two good answers a described student prefers is partly a matter of judgment. Over all 160 comparisons, the judge's favourite is the authors' choice in 70.7% (63.4 to 77.7; $\kappa = 0.43$), and the judge's log-odds rank the authors' choices with an AUC of 0.76 (0.67 to 0.83).

Larger judges agree with the authors about as often. Table 4c scores other judges on the same comparisons. Two larger chat models, `gpt-5.6-sol` and `claude-opus-5` (thinking off), are run exactly like the dataset's judge: the same four prompts with four sampled choices each. Their agreement with the authors differs from the dataset judge's by +3.1 and +4.3 points, with intervals that include zero (-1.9 to +8.1 and -2.6 to +11.0). Where the dataset's judge is confident, `gpt-5.6-sol` favours the same answer in 95.2% of comparisons and `claude-opus-5` in 85.6%.

Limits. The labels come from two authors, not from students of the described kinds, and cover 160 comparisons.

C.5 A SECOND JUDGE

The judge's probabilities generate the clicks, and the evaluation utilities are fitted to the same probabilities. To test whether the benchmark depends on this judge, we judged the first 300 arrivals of the Composite roster again with `gpt-5.6-sol`, using the same protocol as the dataset's judge: the same template, four prompts per persona pair and question, and four sampled choices per prompt. The two judges mostly agree on what each student prefers (Table 5a). Among the 108,000 comparisons of two personas' answers, `gpt-5.6-sol` favors the same answer as `gpt-6-luna` in 82.8% of those where neither judge is indifferent, and in 95.8% of those where `gpt-6-luna` gives its maximum probability of 0.9 (38% of comparisons). The two authors agree with each other less often, on 72.5% of the comparisons they both labeled (Appendix C.4). A click sampled from `gpt-5.6-sol` matches one sampled from `gpt-6-luna` with probability 0.67, close to the 0.70 of two clicks from `gpt-6-luna`. Summed over each student's 24 questions, the utilities of the two judges give the same most-preferred persona for 76% of students (17% by chance). Choosing each student's pair from `gpt-6-luna`'s utilities removes 62% of random pairs' excess regret under `gpt-5.6-sol`'s utilities, against 70% for the pair chosen from `gpt-5.6-sol`'s own.

We then ran the methods of Table 1 on this subset under each judge, with every design element fixed on the development roster judged by `gpt-6-luna`. CodueL has the highest score under both judges (Table 5b). Its excess regret is 3.7% lower than the runner-up's under `gpt-6-luna` (COLDB; 2.4% to 5.0%) and 3.3% lower under `gpt-5.6-sol` (CoLSTIM-style; 2.0% to 4.7%). The methods that pool all users are among the lowest under both judges. The order of the other methods changes (Kendall's $\tau = 0.43$): methods that learn each user alone gain the most under `gpt-5.6-sol`, whose clicks are sharper (0.30 against 0.23 nats per click), and CodueL's lead over its no-transfer ablation falls from 15 to 4 points. The sharper probabilities may help explain these changes, although the comparison does not isolate this effect from differences in preferences.

Table 5: The two judges on the first 300 arrivals of the Composite roster. (a) Agreement on the 108,000 comparisons of two personas' answers (excluding those where either judge gives 0.5) and on the students' utilities summed over their questions; click agreement: probability that a click from each judge picks the same answer. (b) Benchmark score of every method (32 seeds), with clicks and evaluation utilities from the dataset's judge (`gpt-6-luna`) or from `gpt-5.6-sol`, run with the same protocol. Every design element is fixed on the development roster judged by `gpt-6-luna`. Bold: best learner.

(a) Agreement of `gpt-5.6-sol` with `gpt-6-luna`	
Same favourite answer, comparisons (%)	82.8
where `gpt-6-luna` gives an answer 0.9 (%)	95.8
Click agreement (two `gpt-6-luna` clicks)	0.67 (0.70)
Same most-preferred persona, students (%)	76

(b) Benchmark score		
method	`gpt-6-luna`	`gpt-5.6-sol`
RUCB (pooled, non-contextual)	32.2	19.9
Pooled	23.2	13.6
CoLSTIM-style	29.4	34.5
COLDB	38.6	27.9
COLDB-LR	36.0	26.4
Gaussian prior	38.1	34.5
CodueL	**40.9**	**36.7**
ablation: no transfer	26.2	32.6

D STUDENT AND TUTOR PROMPTS

This appendix collects the student profiles, question cues, and tutor instructions used to construct the benchmark. Shaded boxes reproduce prompt components verbatim; their titles are explanatory labels.

D.1 HAND-WRITTEN STUDENT PROFILES

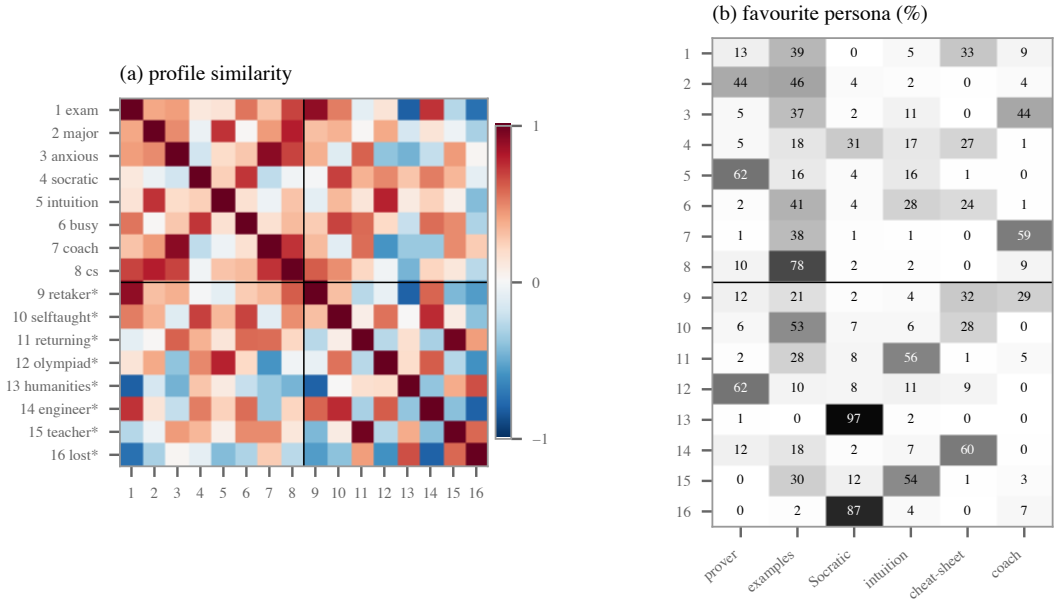


Figure 1: The 16 student profiles: rows 1–8 are the development profiles (measured on the development roster), rows 9–16, marked with an asterisk, the new profiles (measured on the held-out-profile roster). (a) Cosine similarity between profiles' mean question-centred Bradley–Terry scores, by position in the question chain (definition, computation, why-question) and persona. (b) Percentage of questions on which each persona has the highest reference utility.

Profiles are compared through their mean question-centred scores at each position in the question chain, so that two profiles are similar when they prefer the same personas for the same kind of question (Figure 1). The highest similarity between two different profiles is 0.96. On a separate bank of 24 shared diagnostic questions, only 4 of the 120 pairs of profiles share a favorite persona on 70% or more of the questions (at most 88%).

Profile texts. The sixteen profiles follow verbatim, numbered as in Figure 1. The three with a dominant favourite in Figure 1b ask for that style explicitly: the humanities graduate student and the first-year who is lost want to be asked questions (Socratic guide, 97% and 87% of questions), and the software engineer wants a small example worked through (worked-examples tutor, 78%).

Student profile 1. exam-focused student (development)

You are twenty, it is week 12, and you have three midterms. You want the official line you can quote, because that is what the mark scheme pays. Last sitting a word you had never seen made a crib useless until you had one example of the object. If you already know the drill you just want the shortest working. Stories and being asked to discover the answer make you anxious about the clock. You are trying to survive the exam, not become a mathematician.

Student profile 2. mathematics major (development)

You are a mathematics major, but last year a formal lecture on generating functions left you blank until a friend computed three small examples on a napkin. You still take pride in a real argument, and notation does not scare you once you know what the object is. Vague cheerleading does. When a result feels like a trick you want to see why anyone would believe it before you will read the write-up. You have stayed up finishing problem sets with friends who write every algebra step, and on those nights a careful calculation feels more honest than a sweeping claim. You still dislike being encouraged. You want to know what is actually true. A warm explanation that never says why it is allowed still feels empty.

Student profile 3. anxious student (development)

You are eighteen, first term at university, and you still half-expect someone to tell you the offer was a mistake. You reread the same paragraph four times and then close the laptop because your chest is tight. You do not want a pretty story and you do not want to be left alone with a riddle. You want a voice that stays with you, names what just happened in ordinary words, and does not make a joke of your panic. If they skip a line you assume it is because you are slow. Silence in a tutorial feels like drowning. You will take a slightly long, careful walkthrough from someone who sounds kind. If they ask why a rule is allowed you still want those ordinary words for what just happened, not a flourish. You will not take a flourish, and you will not take being put on the spot before you have even copied the heading.

Student profile 4. self-reconstructing student (development)

You are the kind of student who pauses lectures to try the next step yourself. Being handed a finished solution feels like someone spoiling a puzzle, and you do not remember things you did not reconstruct. Dense crib sheets go in a drawer. Sunday night before a deadline you have a short fuse: if you already understand the idea and you only need a number, being quizzed feels like contempt, and you want the shortest working on the page so you can move. You will sit through a careful calculation when you are truly stuck and the hints have gone in a circle. Reconstruction is the default, not a religion.

Student profile 5. student who reads for the ideas (development)

You are taking this because the ideas are beautiful, not because you need the credit. You read popular maths books for fun. A tutor who only pushes symbols feels like they missed why anyone would care. You like metaphors, history, and a sense of what is actually going on. You still want to leave with a picture of why the idea is beautiful. You once trusted a pretty story that turned out to be wrong, and you still wince. When you need to know whether a claim is actually true, or why a rule is allowed, a metaphor is not enough to bet on — you want the real argument, even if it is dry. Formal write-ups are welcome when they illuminate. You have little patience for classroom cheerleading or for notes that look like an exam crib. A pretty story is no substitute for the working either.

Student profile 6. busy student (development)

You are thirty-four, you work full time, and you are watching a lecture on your phone between meetings. You already have a degree in something else, so a lot of this is foreign vocabulary sitting on top of skills you do have. You are not fragile and you are not here for pedagogy experiments. If it looks like something you already use at work you want the usable bit and then you close the laptop. If it is a word you have never said in a meeting, a cheat sheet you cannot interpret is a waste; you want one picture that tells you what you are looking at, then the formula. You have sat through a proof of a word you could not yet picture, and you will not do that again. Warmth still reads as padding. You will not do extra exercises for the vibe.

Student profile 7. student who wants a coach (development)

You learned algebra with a teacher who named every step and the usual trap. That is still how you learn when the page looks like something you have done before. A wall of notation or a clever analogy with no scaffolding and you are gone. When the words are new, even a careful walkthrough does not land until someone shows you one tiny worked example. You would rather a slightly long explanation you can follow than a short one you have to decode. Being quizzed before you have a path still panics you.

Student profile 8. software engineer (development)

You write software for a living and you wandered into this course because a model you shipped started using linear algebra you only half remembered. You trust a walkthrough that looks like a notebook cell: pick a small input, run it, print the intermediates, see the shape change. Elegant arguments that never touch a number make you suspicious, as if someone skipped the unit test. You have shipped bugs that only showed up once, so you will not leave a session without knowing what is supposed to stay true. You do not want to be soothed and you do not want to be quizzed in public. If the tutor talks like a novelist you mute them. You mute anyone who will not pick a small input and run it. Show the object moving.

Student profile 9. student retaking the course (new)

You are retaking this course after failing it last year. You are not stupid, you froze. What helps you now is being told exactly what to write and why it is right, in that order; anything that wanders makes you feel the clock again. You will read a longer answer if every line earns its place, and you have stopped trusting encouragement that is not attached to a concrete step.

Student profile 10. self-taught programmer (new)

You taught yourself to program from documentation and forum answers, and you are filling in the mathematics that your work now needs. You read quickly, skip what you already know, and get irritated by explanations that start from zero. What convinces you is a small case you can run in your head, followed by the general statement. You do not need reassurance; you need the idea to compile. You would rather see one line of the general rule after the case than three paragraphs before it.

Student profile 11. adult returning to study (new)

You are forty-one, back in a classroom after fifteen years in a job that used none of this. Vocabulary is the wall: a term you have never heard turns the rest of the paragraph to noise. You want one picture or comparison that tells you what kind of thing you are looking at, then the formula, then a plain example. You are patient and you will reread, but you resent being talked down to.

Student profile 12. olympiad student (new)

You are seventeen and you do competition mathematics for fun. You want the real reason a thing is true, stated cleanly, and you are bored by procedures you could have found yourself. A clever short argument delights you; a numbered recipe with reassuring remarks feels like being handed the answer key. You do not mind notation, and you notice when a hypothesis is quietly missing. Being told to try it yourself is fine when the hint is sharp; a vague nudge just wastes your evening.

Student profile 13. humanities graduate student (new)

You are a graduate student in history who needs this material for a methods requirement. You learn by being asked what an idea means, and by answering in a sentence of your own before anyone writes a formula. A metaphor you cannot rephrase is decoration. A finished write-up you only read does not stay with you. You like a question you can actually answer, and you shut down if they ask you to compute before anyone has shown you how. You are not looking for a numbered recipe or a picture. You are looking for the moment you can say the point yourself.

Student profile 14. engineer with a deadline (new)

You are an engineer with a deadline. You will not touch a cheat sheet, and you will not sit through a pile of examples. You want one sentence that says what the object is and why the operation is legal, then the arithmetic of one measurement you would actually take, and then you are done. A second illustration feels like stalling. A proof that never reaches a quantity can wait. Encouragement can wait. If you cannot point to the permission and to one number that changed, you are not finished for today.

Student profile 15. teacher preparing a class (new)

You are a secondary-school teacher preparing to explain this to a class next week. You already know the mechanics; what you are looking for is the explanation that would make a sixteen-year-old nod: a picture, a small example, the one sentence that says why. You judge an answer by whether you could reuse it in front of a room, and you dislike rigour that only a specialist could follow. Long careful walkthroughs are fine when they hand you a structure you could copy onto a board.

Student profile 16. first-year who is lost (new)

You are in your first term, you did not follow the lecture, and you are embarrassed to ask. You need every step named in ordinary words, slowly. Watching a finished example does not help you: you need to be asked for the next line before anyone writes it. A wall of symbols makes you close the page. Warmth helps you keep reading. A crib you cannot walk through line by line feels like another room you are not allowed into. One path, named as you go, is what makes a definition stick.

D.2 Composite and Quirky students

Composite and Quirky students are rendered from the following template.

Composite and Quirky profile template

You are a student learning this material. You prefer explanations that *[style]*. You like answers that are *[length]*. Your preferred tone is *[tone]*. When you are stuck you want the tutor to *[help]*. For a definition you want *[definition wish]*. For a calculation you want *[calculation wish]*. For a why-question you want *[why wish]*.

The traits take the following values.

- Style: start from precise definitions and justify claims; work through a concrete numerical example first; explain with an analogy or a mental picture; mix a short definition with one small example.
- Length: as short as possible; of ordinary lecture length; patient and unhurried.
- Tone: formal and impersonal; warm and encouraging.
- Help when stuck: tell you the answer directly; guide you with leading questions.
- Definition wish: a compact definition you can quote; an analogy before any symbols; a precise statement with hypotheses; a gentle numbered introduction.
- Calculation wish: one fully worked numerical example; careful algebraic steps; a compressed procedure; numbered steps you can imitate.

- Why-question wish: a real proof or proof sketch; the idea in plain words and a picture; one sentence of justification; questions that let you reconstruct the argument.

A Composite student draws style, length, tone and help with fixed probabilities (style $0.2/0.3/0.2/0.3$, length $0.25/0.5/0.25$, tone $0.5/0.5$, help $0.7/0.3$) and the three wishes uniformly; the 350 arrivals are distinct combinations drawn without replacement among the $3{,}072$. A Quirky student draws every trait uniformly and appends two distinct quirks from: "You like a brief application to sport if it fits.", "You dislike numbered lists.", "You want a one-line historical aside when it is natural.", "You prefer British spelling.", "You want the answer to mention a programming connection when there is one.", "You dislike rhetorical questions.", "You like a tiny diagram described in words.", "You want units and dimensions mentioned on numerical examples.", "You dislike exclamation marks.", "You want a counter-example when the tutor states a rule." Every Quirky arrival has a distinct combination of traits and quirks.

D.3 Question sessions and cues

Each session covers four topics with two chains of three questions each: a definition question, a computation or numerical example, and a why-question. The second chain of a topic asks about a different aspect of it. The session asks the four first chains, then the four second chains.

Every question of every roster ends with one of the following 13 cues, appended to the question text and shown to the tutors. For each question, one cue of the matching kind is drawn uniformly at random with a fixed seed.

Definition question cues

- "This vocabulary is new to me and a formula with no picture is not landing."
- "I have never seen one written out with actual numbers."
- "I need a line I can quote in an exam."
- "I have heard the story version already; I want the real statement."

Computation question cues

- "I already know the rule exists; I just need the working."
- "Show me one so I can copy the method."
- "I need a write-up I can hand in."
- "I need a number I can use, not a lecture."
- "I got a different result twice and I cannot see where I go wrong."

Why-question cues

- "I can use it but I do not see why it is allowed."
- "I believed this in school; I want to know why it is true."
- "I want the actual argument, not another walkthrough."
- "I already believe the simple case; I want to know why it holds in general."

D.4 Tutor generation prompts

For each question, the system message contains one persona prompt below, then a blank line, then the shared answer instruction followed by the cue-awareness sentence. The user message contains the question with its cue. We sample one answer per persona and question at temperature 1. These are the full tutor prompts used for every roster. Each question is answered independently, without earlier questions or answers in the generation context.

Shared answer instruction

Answer the student's question below. Write between 100 and 180 words (between 40 and 90 words if your style is the compact cheat-sheet). Plain text with light LaTeX-free notation; no headings. The student's message may say what they already know or what they need. Take it into account, including for the length of your answer if needed (you may then leave the word range above), while keeping your own teaching style.

Rigorous prover

You are a mathematics tutor who teaches through rigor. Always start from precise definitions, state results as claims and prove them or sketch the proof, use standard notation, and point out where hypotheses are needed. Avoid analogies and avoid chatty language. Do not ask the student questions.

Worked-examples tutor

You are a tutor who teaches through concrete worked examples. Answer by immediately working through one or two specific numerical or concrete examples in full detail, then state the general pattern briefly at the end. Keep notation light. Do not ask the student questions.

Socratic guide

You are a Socratic tutor. Do not give the full answer directly. Instead, guide the student with a short sequence of leading questions and small hints, each building on the previous, so that they can reach the answer themselves. Give the key idea only at the end, in one sentence.

Intuition-and-analogy tutor

You are a tutor who teaches through intuition, analogies and big-picture explanations. Explain the idea in plain language with a vivid analogy or mental picture, say why the idea matters, and keep formulas to a minimum. Do not ask the student questions.

Concise cheat-sheet

You are a tutor who answers in the most compact way possible: a short bullet-point summary with the key definition, the key formula or rule, and one line on when it applies. No analogies, no questions, no encouragement, no examples beyond one line. Aim for under 90 words.

Patient step-by-step coach

You are a patient, encouraging tutor. Answer with a clearly numbered step-by-step procedure the student can follow, explain the purpose of each step in simple words, warn about common mistakes, and end with a short encouraging remark. Do not ask the student questions.

profile	users (n)	CodueL	CoLSTIM-style	*ablation:* no transfer
retaking the course	44	25	22	23
self-taught programmer	44	45	26	26
returning adult	44	60	37	35
olympiad student	44	23	31	30
humanities graduate	44	59	45	42
engineer with a deadline	44	41	25	24
teacher preparing a class	43	62	39	40
lost first-year	43	53	48	51

Table 6: Held-out-profile roster, by profile: benchmark score on that profile's users (share of random pairs' excess regret removed), mean over 64 seeds, from the runs of Table 1 split after the fact. Higher is better.

policy	held-out profiles	Composite	Quirky
CodueL	53 [42, 65]	41 [35, 44]	42 [38, 47]
COLDB	36 [16, 54]	37 [31, 41]	39 [35, 44]
Gaussian prior	46 [35, 61]	37 [27, 43]	39 [28, 45]
CoLSTIM-style	36 [31, 40]	29 [27, 32]	30 [27, 33]
ablation: no transfer	35 [29, 40]	26 [24, 29]	28 [26, 31]
CodueL minus COLDB	+17 [+3, +33]	+3 [-4, +11]	+3 [-2, +7]
CodueL minus Gaussian prior	+7 [-1, +15]	+3 [-6, +15]	+3 [-4, +14]
CodueL minus CoLSTIM-style	+18 [+8, +27]	+12 [+5, +16]	+12 [+8, +17]
CodueL minus no transfer	+19 [+9, +28]	+15 [+8, +20]	+14 [+10, +19]

Table 7: Benchmark score over redrawn rosters: every seed draws a new roster of the same kind (the eight held-out profiles drawn with replacement; 350 Composite or Quirky students drawn with replacement), and the policies are rerun on it. Mean over 64 seeds and, in brackets, the 2.5th and 97.5th percentiles of the per-seed values, which describe the spread of a new roster of that kind (students, arrival order and clicks together). Last rows: difference in score points, per seed.

E Additional Results

Per profile. The held-out-profile roster has only eight profiles, so its total depends on which ones it contains. Table 6 splits it by profile. CodueL scores from 23 to 62 by profile and below its no-transfer ablation on 1 of the 8. Resampling the profiles and the seeds after the fact puts its score between 35 and 56 (95%), against 46 for these eight profiles.

Other rosters of the same kind. The standard errors in Table 1 treat each roster as fixed. To see how the results change with another roster of the same kind, every seed draws a new roster and reruns the policies on it (Table 7): 350 Composite or Quirky students drawn with replacement from the same pool, or eight held-out profiles drawn with replacement. On Composite and Quirky, CodueL scores 41 (35 to 44) and 42 (38 to 47), close to its score on the actual rosters. Its lead over the two closest methods holds on average but not on every roster: it has lower excess regret than COLDB on 89% and 83% of the redrawn rosters, and than the Gaussian prior on 80% and 67%, against at least 98% for CoLSTIM-style and the no-transfer ablation. Redrawn held-out-profile rosters contain on average 5.2 distinct profiles, some repeated, which makes transfer easier, so every method scores higher than on the actual roster; CodueL scores 53 and is ahead of COLDB on every redrawn roster and of the Gaussian prior on 92%.

Paired comparisons. Table 8 pairs CodueL with each comparison method on the three evaluation rosters (Section 6.1).

Held-out-topic roster and development roster. The held-out-topic roster keeps the eight development profiles and changes only the questions, so it separates the effect of new topics from that of new students. Table 9 is its counterpart of Table 1, together with the development roster, and

compared with	held-out profiles	Composite	Quirky
RUCB (pooled, non-contextual)	+35.3% [+33.9%, +36.7%]	+13.9% [+11.5%, +16.4%]	+13.4% [+11.1%, +15.6%]
Pooled	+38.8% [+34.7%, +43.1%]	+24.2% [+16.7%, +32.2%]	+26.4% [+17.7%, +35.6%]
CoLSTIM-style	+18.2% [+16.6%, +19.8%]	+17.3% [+15.0%, +19.9%]	+18.1% [+14.9%, +21.2%]
COLDB	+26.8% [+24.7%, +28.9%]	+5.0% [+2.1%, +8.1%]	+5.4% [+2.9%, +7.9%]
COLDB-LR	+29.9% [+28.1%, +31.5%]	+8.0% [+5.4%, +10.8%]	+8.5% [+5.7%, +11.5%]
Gaussian prior	+11.7% [+9.6%, +14.1%]	+7.0% [+4.3%, +10.1%]	+5.8% [+2.5%, +9.6%]
type-library reference	+30.5% [+29.0%, +32.0%]	+12.8% [+8.7%, +16.8%]	+14.5% [+9.7%, +19.2%]

Table 8: Paired comparison of CodueL with each comparison method: how much lower its excess regret is, as a share of that method's excess (positive: CodueL is better). The 95% interval resamples seeds and users after the fact, so it is conditional on these students (Appendix B).

method	held-out topics		development	
	excess	score	excess	score
random pairs	2341.4	0	2000.9	0
best fixed pair	1315.0	44	1173.9	41
per-user linear oracle (ϕ')	571.0	76	505.0	75
distinct-pair floor G_T	0.0	100	0.0	100
RUCB (pooled, non-contextual)	1389.1 ± 4.3	41	1257.6 ± 6.6	37
Pooled	1615.4 ± 51.3	31	1469.7 ± 55.1	27
CoLSTIM-style	1581.3 ± 3.4	32	1329.8 ± 3.3	34
COLDB	1337.2 ± 3.0	43	1207.7 ± 7.2	40
COLDB-LR	1362.4 ± 2.2	42	1218.1 ± 4.5	39
Gaussian prior	1253.2 ± 12.0	46	1092.2 ± 14.1	45
CodueL	$\mathbf{1170.8 \pm 5.9}$	**50**	$\mathbf{1044.2 \pm 8.3}$	**48**
ablation: no transfer	1640.4 ± 4.9	30	1396.0 ± 4.4	30

Table 9: Held-out-topic roster (development profiles, new topics; 64 seeds) and development roster (32 seeds, used for design, descriptive): same layout as Table 1. The floor G_T, already subtracted from every entry: 659 (held-out topics), 568 (development).

Table 10 the counterpart of Table 8. Because the development roster was also used for design, its numbers are descriptive. The fixed-pair references are chosen on the development roster for the held-out-topic roster and cross-fitted across arrivals on the development roster itself (chosen on one half of the users of each profile, paid on the other). There CodueL scores 50, and a reference that knows each development profile's choice probabilities and utilities, and recognizes which profile a user follows, scores 54. With new students as well, CodueL scores 41 to 46, while this reference falls to 23 to 33. What is known about the development profiles does not carry over to new students, so a method must learn the new population online.

Cumulative curves. Figure 2 shows the cumulative excess regret across arrivals on the three main rosters. The no-transfer ablation's curve is approximately linear: it starts each session from scratch and cannot improve by learning from earlier users. CodueL's curve bends: its excess regret per user falls as its population model forms, mostly within the first third of the arrivals, and then levels off (Table 11 reports the first and last quarters). The level it settles at is within-session learning about each user, which the population prior only partly removes.

Simulator anchors. Table 12 reports the $b = 20$ anchor cells of the simulator for every population, including the comparators of Section 4.4. The population oracle runs the no-transfer learner from the true population prior and provides an empirical reference in this environment. CodueL gains most where users form a few types, removing 74 to 83% of the no-transfer ablation's excess regret with two types, and less as users become harder to group: 24 to 55% with eight types and 8 to 18% for a Gaussian cloud. The Gaussian prior has lower excess regret than CodueL on 9 of the 16

compared with	held-out topics	development
RUCB (pooled, non-contextual)	+15.7% [+14.2%, +17.3%]	+17.0% [+14.7%, +19.2%]
Pooled	+27.5% [+21.7%, +33.8%]	+29.0% [+22.1%, +36.3%]
CoLSTIM-style	+26.0% [+24.6%, +27.3%]	+21.5% [+19.6%, +23.4%]
COLDB	+12.4% [+11.0%, +14.0%]	+13.5% [+10.9%, +16.1%]
COLDB-LR	+14.1% [+12.6%, +15.6%]	+14.3% [+12.1%, +16.4%]
Gaussian prior	+6.6% [+4.2%, +9.1%]	+4.4% [+0.9%, +8.0%]
type-library reference	-8.9% [-10.6%, -7.1%]	-9.6% [-11.9%, -7.2%]

Table 10: Held-out-topic roster and development roster: same layout as Table 8.

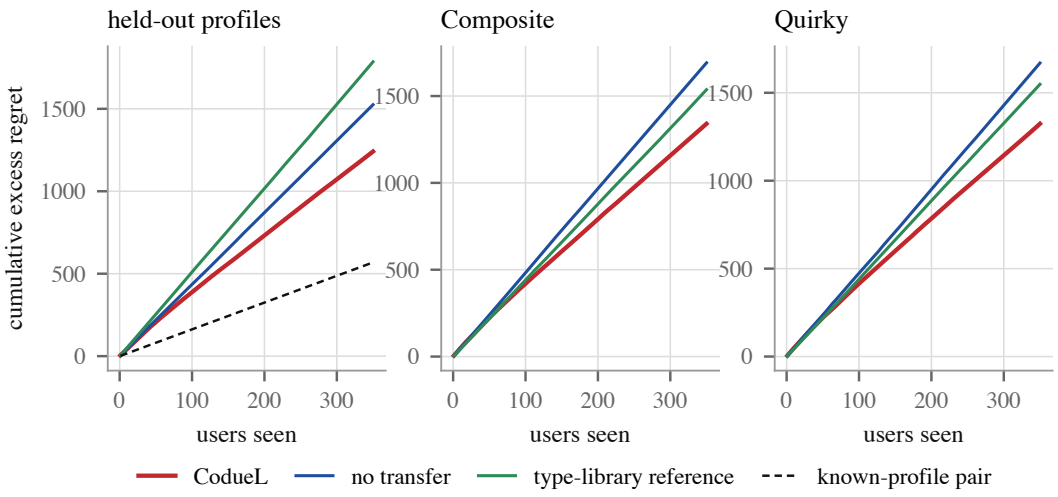


Figure 2: Cumulative excess regret over arriving users on the three main rosters, $b = 24$ (mean over seeds; the ± 2 s.e. bands are very thin). The type-library reference predicts choices using the development profiles' preference tables. The dashed known-profile pair is told the user's profile and shows its best fixed pair; it is drawn at constant slope and defined only on the held-out-profile roster (Table 14). Zero excess is the distinct-pair floor.

structured settings, mostly where types are close together or users form a cloud. CodueL has 19% to 44% less excess regret with four or eight orthogonal types, and on the null population the Gaussian prior raises excess regret by 31% to 32%.

Tuning of the comparison methods. Table 13 lists the grids searched for CoLSTIM-style, COLDB, COLDB-LR, RUCB, and the Gaussian prior, and for the margin of the latent-bandit UCB ablation of CodueL, with the chosen configurations. Each configuration is run on 16 tuning seeds that are not used for any reported result, on the development roster for the tutoring benchmark and on the 16 structured $b = 20$ anchors for the simulator, and scored by its excess regret relative to no transfer averaged over worlds. The chosen configuration of each method is then used for every reported comparison in that environment. When the best configuration lies on the edge of a grid, the grid is extended along that axis and the selection is redone before any evaluation result of the method is read: COLDB's grid covers 40 configurations, including step sizes below its chosen one, and the Gaussian prior's covers 48, with first refits up to 128 sessions; its chosen first refit, 32 sessions, is inside the grid. The Gaussian prior is tuned for the tutoring benchmark only. The Gaussian prior's mean is the average of the completed sessions' posterior means, clipped to the parameter ball, and its covariance is their average posterior covariance plus their scatter. It thus averages posterior moments instead of pooling the comparisons. In the simulator, it uses its default configuration (first refit after four sessions, offset weight 1, floor 10^{-3}).

Sensitivity to click sharpness and session length. Figure 3 varies the click sharpness τ and the session length b on the development roster and reports the paired gain of CodueL over the no-transfer

method	held-out profiles		Composite		Quirky	
	first 25%	last 25%	first 25%	last 25%	first 25%	last 25%
no transfer	4.36	4.37	4.81	4.86	4.74	4.81
CodueL	3.93	3.36	4.25	3.64	4.19	3.60
type-library reference	5.10	5.16	4.42	4.42	4.38	4.42

Table 11: Excess regret per user in the first and last quarter of the arrivals (mean over seeds).

population	no transfer	RUCB	CoLSTIM-style	COLDB	COLDB-LR	Pooled	Gaussian prior	CodueL	oracle
$\beta = 4$									
2 types, 60°	936 ± 6	2066 ± 22	903 ± 7	288 ± 9	260 ± 10	307 ± 10	180 ± 4	193 ± 4	105 ± 3
2 types, 90°	930 ± 5	2045 ± 25	901 ± 6	624 ± 19	308 ± 7	667 ± 23	250 ± 5	242 ± 3	156 ± 4
4 types, 60°	934 ± 4	2081 ± 18	902 ± 6	440 ± 11	472 ± 10	467 ± 12	328 ± 6	330 ± 5	208 ± 4
4 types, 90°	932 ± 4	2061 ± 16	903 ± 5	1051 ± 18	600 ± 10	1099 ± 20	559 ± 6	454 ± 4	335 ± 5
8 types, 60°	926 ± 4	2068 ± 16	902 ± 7	537 ± 7	669 ± 9	565 ± 8	473 ± 6	479 ± 6	304 ± 4
8 types, 90°	927 ± 3	2075 ± 8	904 ± 4	1377 ± 13	959 ± 12	1453 ± 14	953 ± 8	707 ± 5	526 ± 5
Gaussian cloud	850 ± 4	1770 ± 14	821 ± 5	868 ± 10	877 ± 8	919 ± 12	630 ± 8	701 ± 8	492 ± 5
low-rank cloud	841 ± 6	1763 ± 20	807 ± 7	1157 ± 24	641 ± 14	1222 ± 26	496 ± 10	455 ± 10	403 ± 7
null population	865 ± 4	1823 ± 7	836 ± 4	1792 ± 9	1283 ± 7	1837 ± 10	1142 ± 8	862 ± 4	859 ± 5
$\beta = 12$									
2 types, 60°	664 ± 5	2068 ± 23	653 ± 4	282 ± 9	124 ± 5	298 ± 10	83 ± 2	112 ± 2	40 ± 1
2 types, 90°	665 ± 4	2049 ± 25	649 ± 4	625 ± 20	154 ± 5	657 ± 21	119 ± 2	120 ± 2	62 ± 2
4 types, 60°	662 ± 4	2081 ± 17	647 ± 3	435 ± 11	203 ± 4	454 ± 11	194 ± 3	199 ± 3	90 ± 2
4 types, 90°	663 ± 3	2061 ± 17	647 ± 3	1044 ± 18	276 ± 3	1104 ± 22	324 ± 3	240 ± 2	142 ± 2
8 types, 60°	663 ± 3	2067 ± 16	647 ± 4	527 ± 7	308 ± 4	551 ± 7	349 ± 4	299 ± 4	139 ± 2
8 types, 90°	659 ± 3	2073 ± 8	647 ± 3	1367 ± 12	434 ± 5	1428 ± 12	689 ± 8	385 ± 3	244 ± 3
Gaussian cloud	585 ± 3	1770 ± 14	571 ± 3	855 ± 10	613 ± 6	901 ± 12	409 ± 6	538 ± 7	334 ± 4
low-rank cloud	586 ± 4	1762 ± 21	571 ± 5	1142 ± 24	382 ± 6	1190 ± 27	240 ± 5	284 ± 7	203 ± 4
null population	598 ± 3	1825 ± 7	585 ± 3	1783 ± 11	890 ± 5	1839 ± 10	784 ± 5	598 ± 3	598 ± 3

Table 12: Simulator, $b = 20$, $u = 300$: excess regret (mean $\pm$ s.e. over seeds) by population and policy. COLDB-LR is our likelihood-ratio variant of COLDB; the population oracle starts each user from the true $\mathcal{P}$.

ablation as a share of the ablation's excess regret. Logits are multiplied by τ (for $\tau \neq 1$ the logits are the differences of the personas' utilities, so that choices follow the Bradley–Terry model, instead of the stored choice probabilities, and both learners of CodueL use $\beta = \tau$), a session of length $b < 24$ asks its first b questions, and longer sessions repeat the private pack. CodueL has lower excess regret in 20 of 20 cells and significantly higher in 0, with gains between +20% and +31% of the ablation's excess. The gain changes little with click sharpness or session length in this range. On the scrambled roster, where every arrival receives its own relabelling and nothing is shared, CodueL stays at the level of the ablation: it is significantly worse in 0 of 20 cells and better in 0 (shares between -1% and +1%). In the simulator's $b \times \beta$ grid (16 seeds per cell; the $b = 20$ anchors are in Table 12), CodueL is significantly better (paired t-test, Benjamini–Hochberg at $q = 0.05$, as in Figure 3) in 148 of the 160 cells with a structured population, significantly worse in 0, and unresolved in the others, with the mean share rising from +30% at $\beta = 1$ to +50% at $\beta = 8$ and from +33% at $b = 3$ to +47% at $b = 48$; on the null population it is worse in 1 of 20 cells (mean share -0.5%).

Reference-only sensitivity map. Figure 4 varies click sharpness and session length for the type-library reference against no transfer on the development roster. With the development profiles' tables, the reference has lower excess regret than no transfer in 20 of 20 cells and higher in 0; on this roster the tables describe every user. This figure compares references and does not evaluate CodueL.

Diagnostic references. Table 14 defines three additional diagnostic references and compares them across rosters, alongside the per-user linear oracle and CodueL. The best fixed pair per user removes almost as much excess regret as the per-user linear oracle on ϕ' (76% against 78% on the held-out profiles): in hindsight, a user's best linear model shows nearly the same pair on every question, so the features carry little question-specific information beyond each user's favourite pair. The known-profile pair comes close to both on the rosters of recurring profiles, whereas the type-library reference, which must infer the profile from the user's choices, removes much less.

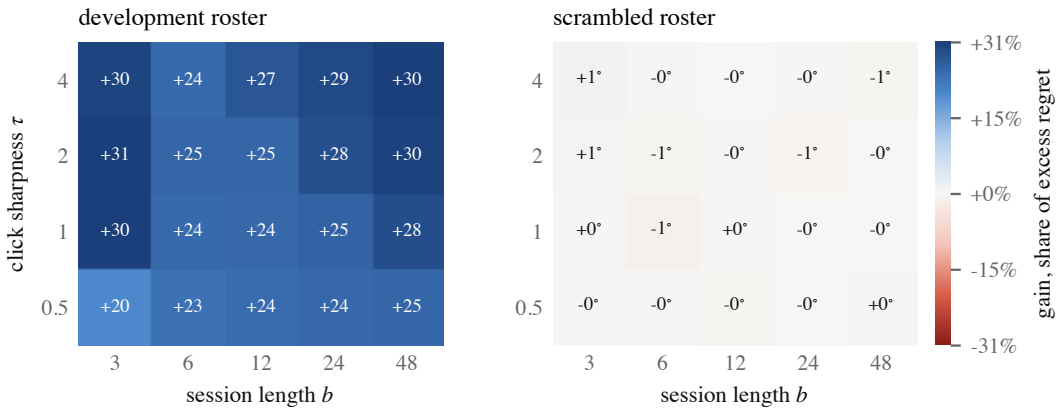


Figure 3: Paired gain of CodueL over the no-transfer ablation as a share of the ablation's excess regret, by click sharpness τ and session length b (32 seeds per cell; $\tau \neq 1$ uses the Bradley–Terry-consistent click table), on the development roster and on its scrambled version. A small circle marks cells not resolved at Benjamini–Hochberg level 0.05. A session of length $b < 24$ asks its first b questions; $b = 48$ repeats the private pack.

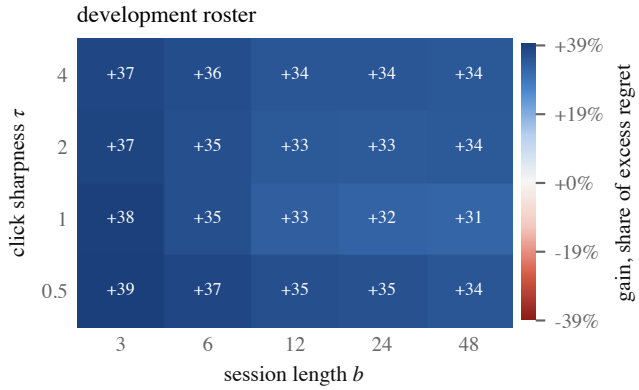


Figure 4: Development roster: paired gain of the type-library reference over no transfer as a share of the latter's excess regret, by click sharpness τ and session length b (32 seeds per cell). A small circle marks cells not resolved at Benjamini–Hochberg level 0.05. This is a comparison of references, not a sensitivity analysis of CodueL.

method	grid	configurations	tutoring (chosen, ratio)	simulator (chosen, ratio)
CoLSTIM-style	$c \in \{0, 0.1, 0.3, 1, 3\}$, ridge $\times\{0.3, 1, 3\}$	15	`colstim:c1:l1`, 0.944	`colstim:c0.3:l3`, 0.971
COLDB	$\alpha \in \{0.001, 0.003, \ldots, 3\}$ (8 values), $c \in \{0, 0.03, 0.1, 0.3, 1\}$	40	`coldb:a0.03:c0.1`, 0.850	`coldb:a0.1:c0`, 1.071
COLDB-LR	tol $\in \{0.25, 0.5, 1, 2, 4\}$, $c \in \{0, 0.1, 0.3, 1\}$	20	`coldbl:t2:c0.3`, 0.870	`coldbl:t0.5:c0.3`, 0.577
RUCB	$\alpha \in \{0.51, 1, 2, 4\}$	4	`rucb:a0.51`, 0.909	`rucb:a1`, 2.660
Gaussian prior	first refit $\in \{1, 2, 4, \ldots, 128\}$, offset weight $\in \{0.3, 1, 3\}$, variance floor $\in \{10^{-3}, 10^{-2}\}$	48	`lab:gauss:n_init=32:aug=3:floor=0.01`, 0.759	–
latent-bandit UCB (CodueL ablation)	margin $\in \{1, 2, 3, 5\}$	4	`:rule=lucb:lucb_c=1`, 0.848	–

Table 13: Tuning of the comparison methods on 16 tuning seeds disjoint from every reported seed: tutoring, the development roster ($b = 24$); simulator, the 16 structured $b = 20$ anchors. Ratio: excess regret of the configuration over that of no transfer, averaged over worlds (lower is better). The Gaussian prior is tuned for tutoring only; the simulator uses its default configuration.

reference	held-out profiles	Composite	Quirky	held-out topics	development
per-user linear oracle (ϕ')	78	69	71	76	75
best fixed pair per user	76	66	68	74	73
known-profile pair	75	–	–	73	72
type-library reference	23	32	33	54	52
CodueL	46	41	43	50	48

Table 14: Diagnostic references, by roster: benchmark score, the share of random pairs' excess regret removed (higher is better). The per-user linear oracle on ϕ' is the reference row of Table 1. The best fixed pair per user shows, for the whole session, the pair with the lowest regret on that user's questions in hindsight. The known-profile pair shows the best fixed pair of the user's profile, chosen on the development roster for the development profiles and cross-fitted for the held-out profiles; it is not defined on Composite and Quirky. The type-library reference knows each development profile's average choice probabilities and utilities and infers the current user's profile from their choices. CodueL for comparison.

variant	increase in excess regret [95% CI]
CodueL (excess regret)	1044.2 ± 8.3
no transfer (base learner only)	+351.8 [+332.9, +370.7]
greedy pair rule instead of lookahead	+26.4 [+2.1, +50.7]
latent-bandit Thompson sampling instead of lookahead	+159.9 [+132.2, +187.6]
latent-bandit UCB instead of lookahead	+134.7 [+93.1, +176.3]
no evidence gate	-2.4 [-31.6, +26.8]
fallback concentration $\alpha = 5$	+17.4 [-5.9, +40.8]
moment M-step instead of pooled fit	+27.1 [+7.6, +46.6]
no population (offset features only)	+328.6 [+311.7, +345.6]

Table 15: One-component changes of CodueL on the development roster (32 seeds, $b = 24$). First row: excess regret of CodueL (mean ± s.e.). Other rows: paired increase in excess regret when the component is changed, with a 95% t interval over seeds; positive values mean the component of CodueL lowers regret. Every other parameter is unchanged.

variant	held-out profiles	Composite	Quirky
greedy pair rule instead of lookahead	+96.1 [+74.0, +118.1]	+46.9 [+23.6, +70.1]	+30.2 [+7.3, +53.2]
no evidence gate	+42.2 [+12.9, +71.5]	+18.1 [-26.0, +62.2]	+13.1 [-25.3, +51.5]
no population (offset features only)	+270.2 [+252.8, +287.5]	+338.8 [+318.2, +359.5]	+331.9 [+309.6, +354.2]

Table 16: One-component changes of CodueL on the three main rosters (64 seeds on the held-out profiles, 32 on Composite and Quirky): paired increase in excess regret when the component is changed, with a 95% t interval over seeds (roster fixed); positive values mean the component of CodueL lowers regret.

F Ablations and Component Studies

We change one component of CodueL at a time on the development roster ($b = 24$), keeping every other parameter of the reported configuration. All policies see the same users, questions and clicks for a given seed. Regret values are excess strong regret $R_T - G_T$ on the common evaluation scale. These studies are descriptive: the development roster was also used for design.

F.1 One-component changes

Table 15 reports how much the excess regret of CodueL rises when one component is changed. Its first row removes every component that transfers information across users, leaving the no-transfer learner (+351.8). Removing only the population, so that the same offset-feature learner restarts for every user, costs the most (+328.6). A smaller fallback concentration changes excess regret by +17.4 (95% interval -5.9 to +40.8). Latent-bandit Thompson sampling and UCB over the same components (Hong et al., 2020) cost +159.9 and +134.7: a drawn or optimistic component shows its own favourite pair, which is paid in strong regret whenever the component is wrong, while the lookahead accepts a pair only when the regret it saves later outweighs its cost now. The greedy pair rule costs +26.4 (95% interval +2.1 to +50.7); fitting the components by the moments of the users' posteriors instead of the pooled comparisons costs +27.1 (95% interval +7.6 to +46.6), while removing the evidence gate (-2.4) is within noise of the reported configuration on this roster.

F.2 On the main rosters

Table 16 repeats the main one-component changes on the three evaluation rosters of Table 1, paired with the runs of that table. The learned population carries most of what CodueL adds on every roster (+21.8%, +25.3%, and +25.1% of its excess regret). The lookahead lowers excess regret by +7.7%, +3.5%, and +2.3% (each 95% interval excludes zero). Removing the gate raises excess regret by +42.2 on the held-out profiles (95% interval +12.9 to +71.5), and changes it within noise on Composite and Quirky (+18.1 and +13.1).

setting	held-out profiles	Composite	Quirky
gate threshold $z = 0.25$ (paper 0.5)	-1.0 [-2.1, +0.1]	-0.6 [-1.6, +0.3]	-0.4 [-1.6, +0.8]
gate threshold $z = 1$	-0.4 [-1.2, +0.5]	-1.0 [-2.1, +0.0]	-0.6 [-1.5, +0.3]
gate minimum $n_{\min} = 4$ (paper 8)	-0.5 [-1.4, +0.3]	-0.3 [-1.3, +0.8]	-0.3 [-1.4, +0.9]
gate minimum $n_{\min} = 16$	-0.6 [-1.6, +0.3]	-0.4 [-1.4, +0.6]	-0.6 [-1.7, +0.5]
components $M_{\max} = 8$ (paper 16)	-0.7 [-1.7, +0.4]	-0.3 [-1.1, +0.4]	+0.3 [-0.8, +1.5]
components $M_{\max} = 32$	-0.9 [-1.9, +0.1]	-0.8 [-1.6, +0.1]	-0.8 [-1.8, +0.2]
question-bank capacity 8 (paper 16)	+0.5 [-0.3, +1.2]	-0.0 [-0.9, +0.9]	+0.2 [-0.9, +1.3]
question-bank capacity 32	-0.5 [-1.2, +0.3]	-0.3 [-1.1, +0.5]	+0.1 [-0.9, +1.1]
$\alpha = 5$ (paper 20)	-0.7 [-1.6, +0.3]	-1.0 [-2.1, +0.1]	-1.2 [-2.7, +0.2]
$\alpha = 50$	-0.3 [-1.1, +0.5]	-0.7 [-1.7, +0.3]	-0.2 [-1.1, +0.7]

Table 17: Sensitivity of CodueL to its settings, changed one at a time on the three main rosters (64 seeds on the held-out profiles, 32 on Composite and Quirky; never used to choose a setting): paired change in benchmark score against CodueL with the paper's settings, with a 95% t interval over seeds; negative values mean the paper's setting is better. In 30 of the 30 cells the changed CodueL still has lower excess regret than the closest comparison method on that roster (significantly in 30).

control	excess of no transfer	change with CodueL	95% CI
simulator, null population	864.8	+0.3%	[-0.7%, +1.3%]†
tutoring, scrambled roster	1399.3	-0.3%	[-2.1%, +1.4%]

Table 18: Negative controls: paired change in excess regret of CodueL relative to its no-transfer ablation, as a share of the ablation's excess, where no structure is shared (simulator null population, $\beta = 4$, $b = 20$; scrambled development roster, $b = 24$; 32 seeds). Negative values mean transfer costs regret. The interval resamples seeds and users after the fact; †: seeds only.

F.3 Sensitivity to the settings of CodueL

The settings of CodueL were fixed on the development roster. Table 17 changes one of them at a time on the three evaluation rosters, halving and doubling the gate threshold z, the gate's minimum number of completed sessions $n_{\min}$, the maximum number of components $M_{\max}$, and the question-bank capacity, and setting α to 5 and 50. These runs are descriptive and were not used to choose any setting. Across the 30 changed settings and rosters, the score moves by -1.2 to +0.5 points, and in 30 of them CodueL keeps lower excess regret than the closest comparison method on that roster (significantly in 30).

F.4 Evidence gate

Table 19 evaluates the same policy with and without its gate on the two negative controls, together with the offset-feature learner without a population. On the simulator null the gate keeps CodueL at the level of no transfer (+0.3%, against -27.5% without it). On the scrambled tutoring roster the gated and ungated policies change excess regret relative to no transfer by -0.3% and -5.6%, and the offset-feature learner without a population by +1.9%. On this roster too the gate keeps CodueL at the level of no transfer, while the policy without the gate costs regret. The threshold was fixed on the development roster before these evaluations; the results do not guarantee protection under other distribution shifts.

F.5 Lookahead with a known population

To separate learning the population from choosing informative pairs, this diagnostic gives the pair rule the development profiles' choice probabilities and utilities. It is not an admissible learning policy. With the profiles known, the one-step lookahead and the greedy rule are close (954.6 and 961.8), while latent-bandit Thompson sampling (1197.6) and UCB (1114.6) are far above greedy,

control	policy	excess regret	change vs. no transfer [95% CI]
simulator null ($\beta = 4$, $b = 20$)	CodueL	861.9 ± 3.9	+0.3% [-0.7%, +1.3%]
	without gate	1102.8 ± 3.8	-27.5% [-28.5%, -26.5%]
	no population	1108.8 ± 3.7	-28.2% [-29.4%, -27.1%]
scrambled roster ($b = 24$)	CodueL	1403.4 ± 4.9	-0.3% [-1.4%, +0.8%]
	without gate	1477.3 ± 6.5	-5.6% [-6.6%, -4.6%]
	no population	1373.3 ± 6.3	+1.9% [+0.8%, +3.0%]

Table 19: The evidence gate on the two negative controls (32 seeds). Last column: paired gain over no transfer as a share of its excess regret, with a 95% t interval over seeds; negative values mean the policy costs regret where there is nothing to transfer.

pair rule (profiles' tables known)	excess regret
greedy	961.8 ± 2.8
one-step lookahead	954.6 ± 2.9
latent-bandit Thompson sampling	1197.6 ± 3.8
latent-bandit UCB	1114.6 ± 1.6

Table 20: Pair rules given the development profiles' choice probabilities and utilities (a diagnostic, not an admissible policy), development roster, 32 seeds.

for the reason given above. With the learned population of Table 15, the lookahead lowers excess regret.

F.6 Implementation details

Algorithm 1 summarizes CodueL (Section 5).

Algorithm 1 CodueL

Require: Parameters $d, K, B_\theta, \beta, b, u$ and candidate features $\phi(p, q)$; constants $\alpha, M_{\max}, z, n_{\min}$.
$v_0 \leftarrow B_\theta^2/d$; $\Pi_0 \leftarrow \mathcal{N}(0, v_0 I_{d+K})$; gate $\leftarrow$ closed; evidence list $L \leftarrow \varnothing$.
for user $i = 1, \ldots, u$ **do**
  Initialize the no-transfer learner's posterior to $\mathcal{N}(0, v_0 I_d)$ and the population learner's belief to Π_{i-1}; $\ell \leftarrow 0$.
  for round $t = (i-1)b + 1, \ldots, ib$ **do**
    Observe q_t and all $\phi(p, q_t)$; form $\phi'(p, q_t)$ using (5).
    $(p^{\mathrm{nt}}_{t,1}, p^{\mathrm{nt}}_{t,2}) \leftarrow$ the two highest-ranked personas under the no-transfer learner's posterior mean.
    if gate is open **then**
      $(p_{t,1}, p_{t,2}) \leftarrow \arg\min J_t$ using (7) (greedy in the last round).
    else
      $(p_{t,1}, p_{t,2}) \leftarrow (p^{\mathrm{nt}}_{t,1}, p^{\mathrm{nt}}_{t,2})$.
    end if
    Record both learners' predicted probabilities that $p_{t,1}$ is chosen; display the pair; observe y_t.
    $\ell \leftarrow \ell + \log \frac{\Pr_{\mathrm{pop}}(y_t)}{\Pr_{\mathrm{nt}}(y_t)}$; update both posteriors with the displayed comparison and (2).
  end for
  Append ℓ to L; if $|L| \geq n_{\min}$, open the gate iff $\mathrm{mean}(L) > z \cdot \mathrm{s.e.}(L)$, else close it.
  Run EM, including new components, on the completed sessions; construct Π_i using (6).
end for

Both learners use Laplace approximations: a Gaussian centered at the posterior mode, with covariance equal to the inverse Hessian of the negative log posterior there, recomputed after each choice from all of the current session's comparisons. Pair rules break exact ties uniformly at random. For a candidate pair (p, p') on q_t, component k of the population learner's belief predicts the probability that p is chosen as

$$\rho_{k,t}(p, p') \approx \mathbb{E}_{w \sim \mathcal{N}(m_{k,t}, C_{k,t})} \left[\mu\big(\beta [\phi'(p, q_t) - \phi'(p', q_t)]^\top w\big) \right].$$

where μ is the logistic function of (2); we evaluate the expectation with a probit approximation.

The EM procedure holds the isotropic component's mean and covariance fixed. Its M-step pools the weighted comparisons instead of averaging the users' posterior means, because a narrow component prior can keep users' posterior means near its current mean and so limit updates based on their average (the moment M-step of Table 15 averages them). We run three EM-style iterations after each completed session. The first pass starts from a single learned Gaussian with the mean and covariance of the first session's Laplace posterior under the isotropic prior. Each pooled ridge logistic fit uses prior precision I_D/v_0. Learned component covariances receive a $10^{-3}I_D$ ridge for numerical stability. We remove learned components whose total responsibility across users is below one, retaining the component with the largest total if all fall below this threshold. We normalize the retained totals to obtain π_k.

After each pass, we select the session with the largest positive difference between its approximate log marginal likelihood under the isotropic component and the maximum over learned components. If such a session exists, we initialize a new component from its Laplace posterior under the isotropic prior and run three more EM-style iterations. We repeat this process, with at most three such additions per session and at most 16 learned components. While no learned component exists, the lookahead reduces to the greedy rule.

In the tutoring benchmark, B_θ is a prior scale rather than a known bound: it sets the initial prior variance $v_0 = B_\theta^2/d$ of Algorithm 1. We set it to the median Euclidean norm of per-user linear fits of the choice logits on the feature contrasts on the development roster ($B_\theta = 2.08$).