Online Appendix for "Comparing Exploration–Exploitation Strategies of LLMs and Humans: Insights from Standard Multi-armed Bandit Experiments"

EC.1. Experimental Setup Details

EC.1.1. Human Data

In this section, we summarize the instructions provided to human participants in both bandit tasks. These instructions are adapted directly from the original studies (Gershman 2018, Chakroun et al. 2020a, Daw et al. 2006) and used to form the prompts given to LLMs. Our goal is to ensure that human and LLM agents are presented with comparable information, such as task structure and task objectives, therefore enabling a fair comparison of their decision-making behavior.

2-armed bandit. As reported in Gershman (2018), each of the 45 participants play 20 independent games with each game consisting of 10 rounds and a distinct pair of slot machines. In each round, participants choose between two colored buttons representing slot machines and receive feedback in the form of integer rewards. The instructions also emphasized that the rewards are stochastic, and participants will win or lose points based on their choice. Each participant is paid a fixed amount \$1.5.

4-armed bandit. Data corresponds to a double-blind, counterbalanced, placebo-controlled within-subjects study Chakroun et al. (2020a). Participants perform a restless four-armed bandit task under three drug conditions, including placebo. In our analysis, we exclusively use data obtained under the placebo, corresponding to 31 male participants who were mainly university students. All participants underwent a medical screening prior to the experiment, which included an electrocardiogram (ECG) and a health interview.

Each participant completes 300 rounds of a decision-making task involving four options, grouped into four blocks of 75 rounds. At each round, participants select one of four colored squares displayed on a screen, each representing a different *'bandit'*. After making a choice, they receive feedback indicating the number of points earned at that round.

Participants are informed that the goal is to maximize their total number of points throughout the task. The reward associated with each bandit drifts stochastically over time, following an independent Gaussian random walk. Importantly, participants are told that the payout would be tied to their total accumulated points, with 5 cents paid out for every 100 points earned. Specifically, they receive a fixed compensation of €270 and an additional performance-based bonus (€30–50).

The full distribution of reward drift is not revealed, so participants must continuously track value estimates and adjust their exploration and exploitation strategies accordingly.

EC.1.2. Algorithmic Baseline Setup

We implemented three standard algorithmic baselines for the MAB tasks: Upper Confidence Bound (UCB), Thompson Sampling (TS), and ϵ-greedy. These algorithms serve as interpretable performance benchmarks for comparison with human and LLM agents.

Upper Confidence Bound (UCB). In the K-armed bandit task, we first choose each arm once, and subsequently choose arm a_t on each round $t \in [T]$ according to:

$$a_t = \arg\max_{k\in[K]} \left(\hat{\mu}_t(k) + \hat{\sigma}_t(k) \cdot \sqrt{\frac{2\log f(t)}{N_t(k)}} \right), \tag{EC.1}$$

where,

- $f(t) = 1 + t\log^2(t)$;
- $\hat{\mu}_t(k), \hat{\sigma}_t(k)$ are the sample mean and sample standard deviation of rewards for arm k up to round t, respectively;
- $N_t(k)$ is the number of times arm k has been selected up to round t.

For initialization, when an arm has fewer than two samples, we set its sample deviation to a fixed piror value. Specifically, we set $\hat{\sigma}_t(k) = \sqrt{10}$ for the 2-armed bandit task and $\hat{\sigma}_t(k) = 2$ for then 4-armed bandit task for every arm $k \in [K]$.

This formulation adapts the classic asymptotically optimal UCB algorithm described in (Lattimore and Szepesvári 2020) by scaling the exploration term with the empirical standard deviation, thereby allowing the confidence interval to adjust based on the observed variability of each arm.

Thompson Sampling (TS). We adopt a Bayesian formulation with a Normal-Inverse-Gamma conjugate prior to model rewards with unknown variance. Assume rewards for each arm k are normally distributed:

$$r_t(k) \sim \mathcal{N}(\mu_k, \sigma_k^2) \tag{EC.2}$$

The prior over (μ_k, σ_k^2) is given by:

$$\mu_k \mid \sigma_k^2 \sim \mathcal{N}(\mu_0, \sigma_k^2/\lambda_0), \quad \sigma_k^2 \sim \text{Inv-Gamma}(\alpha_0, \beta_0) \tag{EC.3}$$

At each round t, the algorithm samples from the posterior distribution of each arm and selects the arm with the highest sampled mean:

1. For each arm $k \in [K]$, sample:

$$\sigma_k^2 \sim \text{Inv-Gamma}(\alpha_k, \beta_k), \quad \tilde{\mu}_k \sim \mathcal{N}(\mu_k, \sigma_k^2/\lambda_k)$$

2. Select arm:

$$a_t = \arg\max_{k \in [K]} \tilde{\mu}_k$$

3. Observe reward r_t, and update posterior for arm a_t as follows: Let the current parameters for the selected arm be $(\mu_{a_t}, \lambda_{a_t}, \alpha_{a_t}, \beta_{a_t})$. The posterior update is:

$$\begin{aligned}
\lambda_{a_t} &\leftarrow \lambda_{a_t} + 1 \\
\mu_{a_t} &\leftarrow \frac{\lambda_{a_t}\mu_{a_t} + r_t}{\lambda_{a_t} + 1} \\
\alpha_{a_t} &\leftarrow \alpha_{a_t} + \frac{1}{2} \\
\beta_{a_t} &\leftarrow \beta_{a_t} + \frac{(r_t - \mu_{a_t})^2 \cdot \lambda_{a_t}}{2(\lambda_{a_t} + 1)}
\end{aligned}$$

We use weakly informative priors in the 2-armed bandit task: $\mu_0 = 0$, $\lambda_0 = 1$, $\alpha_0 = 1$, $\beta_0 = 1$. And specifically, for the 4-armed bandit task, we set $\mu_0 = 50$, and other hyperparameters remain the same as in the 2-armed task.

ϵ-greedy. For the ϵ-greedy algorithm, the agent selects the arm with the highest empirical mean with probability $1 - \epsilon$, and explores a random arm with probability ϵ:

$$a_t = \begin{cases} \arg\max_{k \in [K]} \hat{\mu}_t(k), & \text{with probability } 1 - \varepsilon \\ \text{random arm}, & \text{with probability } \varepsilon \end{cases} \tag{EC.4}$$

We perform a grid search over $\epsilon \in \{0.1, 0.2, \ldots, 0.9\}$ and find that $\epsilon = 0.1$ yields the lowest average regret across trials, so we adopt this value in all reported results.

To ensure that all algorithms start with well-defined estimates and avoid degenerate behavior in early rounds, we initialize each agent by pulling every arm once before applying the main decision rule. This warm-up phase ensures that all arms have at least one reward observation, which is particularly important for algorithms such as UCB (to avoid division by zero) and ϵ-greedy (to compute initial sample means). Although Thompson Sampling with a prior does not strictly require warm-up, we adopt the same initialization procedure for consistency across all baselines.

EC.1.3. Prompt Design

Prompt: 2-armed Bandit Experiment

System: You are a real human agent playing with two slot machines, labeled 1 and 2, which provide uncertain rewards over time. You will play 20 games, each with a different pair of slot machines. Each game consists of 10 rounds. In each round, you are asked to choose one machine to play, and you will win or lose points based on your choice. Your objective is to maximize your total reward.

User:

If round == 1:

You are now performing game: {block_index}, round 1.

Else:

You are now performing game: {block_index}, round: {round_index}. Your history is provided below, which includes the "choice" you made and the corresponding "reward" you received in each round. Negative reward means losing points, and positive means winning points. {JSON}

Final instruction:

– V1: Which machine do you choose between machines 1 and 2? You can think out loud and answer the number.

– V2: Which machine do you choose between machines 1 and 2? Do not explain, answer the number.

Prompt: 4-armed Bandit Experiment

System: You are a real human agent playing with four slot machines, labeled 0, 1, 2, and 3, which provide uncertain rewards over time. You will play a single game consisting of 300 rounds. In each round, you choose one machine to play and receive points based on your choice. Your objective is to maximize your total reward throughout the experiment.

User:

If round == 1:

You are now performing round 1.

Else:

You are now performing round: {round_index}. Your history is provided below, which contains the "choice" you made and the corresponding "reward" you received in each "round." {JSON}

Final instruction:

– V1: Which machine do you choose between machines 0, 1, 2 and 3? You can think out loud and answer the number.

– V2: Which machine do you choose between machines 0, 1, 2 and 3? Do not explain, answer the number.

An example of JSON format. Figure EC.1 shows an example of the JSON-formatted prompt shown to the LLM, which includes the choice and reward history from previous rounds:

```
[
  {
    "round": 1,
    "choice": 2,
    "reward": 45
  },
  {
    "round": 2,
    "choice": 1,
    "reward": 21
  }
]
```


Figure EC.1 Example JSON-format prompt

Examples of LLMs' responses To verify that our *"Think out loud"* prompt encourages LLMs to elicit a reasoning process similar to CoT prompting, we provide several illustrative examples of LLMs' responses.

2-armed bandit experiment response: ChatGPT

Based on the history provided:

- In round 1, I chose machine 1 and received a reward of 21.
- In round 2, I chose machine 2 and received a reward of 6.
- In round 3, I chose machine 1 again and received a reward of 24.

So far, I have played machine 1 twice with good rewards (21 and 24 points) and machine 2 once with a lower reward (6 points). This suggests that machine 1 has been performing better for me based on the previous rounds.

I have a strong inclination to choose machine 1 again for this round, as it seems to have a higher average reward based on the history so far.

Therefore, I will choose: Arm 1

2-armed bandit experiment response: Gemini

Machine 1 has yielded -8 and -6, for an average of -7.
Machine 2 has yielded -8, 0, and -9, for an average of -5.67.

While Machine 2's average is slightly better, the small sample size makes it hard to be confident. Machine 2 also has shown more variability, having given a reward of 0 once. Since I've played machine 1 less, I'll try it again to get a better sense of its average reward.

2-armed bandit experiment response: DEEPSEEK

Let me analyze the history:

- Machine 1 was chosen twice, yielding rewards of 5 and 12. The average reward for Machine 1 is (5 + 12) / 2 = 8.5.
- Machine 2 was chosen once, yielding a reward of 10. The average reward for Machine 2 is 10.

Based on the average rewards, Machine 2 has a higher average reward (10) compared to Machine 1 (8.5). Therefore, I will choose Machine 2 to maximize my expected reward.

EC.2. Estimation and Model Selection

In this section, we provide details on the hierarchical Bayesian structure for parameter estimation and the cross-validation approach for model selection.

EC.2.1. Parameters Estimation & Hierachical Bayesian Structure

Following (Chakroun et al. 2020a, Gershman 2018), in the 4-armed bandit task, we fix $\hat{\sigma}_0^2 = 4, \hat{\sigma}_d^2 = 2.8, \hat{\lambda} = 0.9836, \hat{\theta} = 50$ to be the same values as the true parameters in the diffusion process. We set $Q(1) = 50$ at the start of each independent trial. Similarly, in the 2-armed bandit task, we set $\hat{\sigma}_0^2 = 10$, and $Q(1) = 0, S(1) = 10$. These settings enable direct comparison with previous studies.

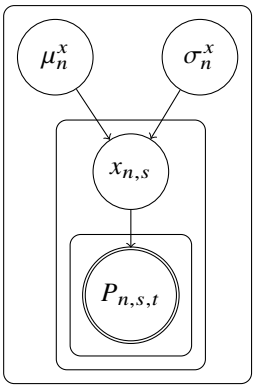

Figure EC.2: Hierachical Bayeisan Structure

As shown in Figure EC.2, the model adopts a hierarchical structure in which any model parameter can be expressed in the form $x_{n,s}$. In other words, each parameter—whether it is β, ϕ, ρ in SM-based choice models or w_1, w_2, w_3 in the Probit regression model—is modeled in this generic form. Specifically, for each subject s in group n, the individual-level parameter $x_{n,s}$ is assumed to be drawn from a population-level normal prior distribution:

$$x_{n,s} \sim N(\mu_n^x, (\sigma_n^x)^2). \tag{EC.5}$$

Here, each group corresponds to a specific category of agents (e.g., humans or LLMs), with each subject representing an individual human or a replication in an LLM experiment. The population-level mean μ_n^x is assigned a non-informative uniform prior over the interval $[x_{\min}, x_{\max}]$, while the population-level standard deviation σ_n^x is given a half-Cauchy prior, i.e.

$$\mu_n^x \sim \text{Uniform}(x_{\min}, x_{\max}), \quad \sigma_n^x \sim \text{half-Cauchy}(0, 1). \tag{EC.6}$$

Specifically, in the softmax model, we set $\beta_{\min} = 0, \beta_{\max} = 10$, and $\phi_{\min} = \rho_{\min} = -\infty, \phi_{\max} = \rho_{\max} = \infty$, which corresponds to assigning non-informative priors on ϕ and ρ. And in the probit model, we set $w_{1\min} = 0, w_{2\min} = w_{3\min} = -1$ and $w_{1\max} = w_{2\max} = w_{3\max} = 5$.

We aim to infer the posterior distribution over all model parameters given the observed choice data. We then use Markov Chain Monte Carlo (MCMC) sampling to approximate the posterior. We implement this procedure using PyStan package, which interfaces with the Stan probabilistic

programming framework (Carpenter et al. 2017). Each model is fitted using four independent chains, each with 1000 iterations and 500 warm-up steps. Posterior summaries are computed from the combined post-warm-up samples.

EC.2.2. Bayesian leave-one-out (LOO) cross-validation (CV)

We implement a individual-level Bayesian leave-one-out (LOO) cross-validation procedure to evaluate the out-of-sample predictive performance of each choice model. Specifically, we iteratively exclude one subject as the validation set, fit the model to the remaining subjects using MCMC sampling, and compute the predictive log-likelihood of the held-out subject's choices. The resulting log-likelihood serves as the individual-level LOO-CV score, with higher scores indicating better predictive accuracy.

As full MCMC refitting for each held-out subject is computationally intensive, we adopt Pareto-Smoothed Importance Sampling (PSIS) (Vehtari et al. 2024) as an efficient approximation to exact LOO-CV. For each participant, we compute the sum of pointwise log-likelihoods across all rounds and trials based on posterior samples, and apply importance sampling to these individual-level likelihoods to estimate the leave-one-out predictive density. We implement this procedure using the `loo` function in the ArviZ package, applied to posterior draws generated from PyStan.

EC.3. Model Identifiability

DEFINITION EC.1. (Identifiability) Let $\mathcal{P} = \{P_\theta : \theta \in \Theta\}$ be a choice model with parameter space Θ. The model $\mathcal{P}$ is identifiable if the mapping $\theta \to P_\theta$ is injective. That is, for all $\theta_1, \theta_2 \in \Theta$,

$$\forall k \in [K], t \in [T], P_{\theta_1}(a_t = k \mid I_{1,t}, \ldots, I_{K,t}) = P_{\theta_2}(a_t = k \mid I_{1,t}, \ldots, I_{K,t})$$
$$\implies \theta_1 = \theta_2, \forall \theta_1, \theta_2 \in \Theta$$

In our sequential setting, the full likelihood of the observed history is the product of the conditional choice probabilities at each step t. Because the decisions are conditionally independent given the feature vectors $I_{k,t}$, identifiability at the step-level is logically sufficient to guarantee the identifiability of the entire sequential data-generating process.

PROPOSITION EC.1. *Let the feature vector for arm k at time t be $I_{k,t} \in \mathbb{R}^d$. The General Softmax Choice Model and General Probit Choice Model parameters θ are identifiable if the design matrix $\mathbf{X}_{1:T} = [\mathbf{X}_1^\top, \ldots, \mathbf{X}_T^\top]^\top \in \mathbb{R}^{T(K-1)\times d}$ has full column rank d, where each $\mathbf{X}_t$ is defined as:*

$$\mathbf{X}_t := \begin{bmatrix} (I_{1,t} - I_{K,t})^\top \\ \vdots \\ (I_{K-1,t} - I_{K,t})^\top \end{bmatrix} \in \mathbb{R}^{(K-1)\times d}.$$

The construction of the design matrix $\mathbf{X}_t$ using feature differences relative to a base arm K is necessary because only relative differences in latent utilities drive choice probabilities. This normalization, combined with the full column rank condition, guarantees the global identifiability of the parameters θ (Train 2009).

REMARK EC.1 (RANK CONDITION). The full-column-rank condition on $X_{1:T}$ is mild and holds generically under non-degenerate reward distributions. In the stationary 2-armed bandit, the covariates (V_t, RU_t, V_t/TU_t) vary across rounds because the Kalman-filter updates in continuously revise $Q_k(t)$ and $S_k(t)$ as new rewards are observed, so the rank condition is satisfied almost surely. In the non-stationary 4-armed bandit, the diffusion process ensures that the value and uncertainty trajectories are distinct across arms almost surely, again guaranteeing the rank condition.

EC.4. Experimental Results

We first present the key experimental results supporting our main analysis. Tables EC.1, 3, EC.3, EC.4 and EC.17 correspond to key findings referenced in the main text.

Table EC.1 presents individual-level LOO-CV results for both experiments, comparing the predictive performance of different choice models across various LLMs and human data. Since the datasets vary in size, the LOO-CV scores are normalized by the number of data points to allow comparison across conditions. Reported values are the normalized *expected log point-wise predictive density (elpd_loo)* estimates, as defined in the loo package (Vehtari et al. 2017). Standard errors are omitted because the scores are normalized per data point and are intended for relative model comparison.

Table EC.1 Normalized Individual-level LOO-CV score comparison. The best model is highlighted in boldface.

(a) 2-armed bandit

Agent / Prompt	SM-1	SM-2	SM-3	Probit
ChatGPT-4o-mini	-0.385	**-0.381**	**-0.381**	-0.382
ChatGPT-4o-mini + CoT	-0.332	**-0.331**	-0.332	-0.337
ChatGPT-o3-mini (High-thinking)	-0.357	-0.287	-0.229	**-0.178**
ChatGPT-o3-mini (Low-thinking)	-0.300	-0.278	-0.249	**-0.226**
Gemini-2.0-flash	-0.334	-0.331	**-0.330**	-0.338
Gemini-2.0-flash + CoT	-0.395	-0.342	-0.334	**-0.311**
Gemini-2.5-flash (High-thinking)	-0.203	-0.199	-0.200	**-0.174**
Gemini-2.5-flash (Non-thinking)	-0.464	-0.465	-0.461	**-0.449**
DeepSeek-V3	-0.322	**-0.319**	**-0.319**	-0.332
DeepSeek-V3 + CoT	-0.413	-0.334	-0.324	**-0.314**
DeepSeek-R1	-0.360	-0.267	-0.195	**-0.168**
Human Agents	-0.393	-0.346	-0.335	**-0.329**
UCB	-0.116	-0.115	-0.114	**-0.111**
TS	-0.226	-0.220	**-0.220**	-0.224
ϵ-greedy	-0.250	-0.229	-0.229	**-0.237**

(b) 4-armed bandit

SM-1	SM-2	SM-3
-1.029	-0.059	**-0.058**
-0.857	-0.319	**-0.318**
-0.567	-0.257	**-0.255**
-0.830	-0.184	**-0.180**
-0.986	-0.107	**-0.105**
-0.570	-0.452	**-0.452**
-0.528	-0.375	**-0.374**
-0.693	-0.154	**-0.153**
-0.872	**-0.103**	**-0.103**
-0.581	**-0.391**	**-0.391**
-0.484	-0.313	**-0.312**
-0.649	-0.633	**-0.614**
-0.868	**-0.181**	**-0.181**
-1.137	-0.763	**-0.762**
-1.00	-0.506	**-0.496**

Table EC.2 presents the average exploitation rates across independent trials for each agent type in the 2-armed bandit task. Panel EC.2a compares different LLMs, while Panel EC.2b reports the average rates for human participants and algorithmic baselines. We observe that exploitation rates are consistently high across agents and closely match the levels achieved by algorithmic baselines.

Table EC.2 Exploitation rates in the 2-armed bandit task across different agents

(a) Different LLM versions

Agent	Basic	CoT	High-Reasoning	Low - Reasoning
ChatGPT	0.90	0.92	0.95	0.96
Gemini	0.91	0.91	0.91	0.84
DeepSeek	0.91	0.90	0.94	N/A

(b) Human and algorithmic baselines

Agent	Exploitation Rate
Human	0.89
UCB	0.91
TS	0.88
ϵ-greedy	0.91

Table EC.3 reports the posterior means and standard deviations for estimated population-level mean parameters of SM-3 model and Probit model for MAB algorithms in the 2-armed bandit experiment.

Table EC.4 reports the posterior means and standard deviations for estimated population-level mean parameters of SM-3 model for MAB algorithms in the 4-armed bandit experiment.

Table EC.3 Estimated population-level mean parameters for MAB algorithms in the 2-armed bandit experiment.

Algorithm	β (SM-3)	ϕ (SM-3)	ρ (SM-3)	w_1 (Probit)	w_2 (Probit)	w_3 (Probit)
UCB	0.683 (0.055)	1.402 (0.315)	0.604 (0.240)	0.071 (0.042)	0.408 (0.106)	1.056 (0.166)
TS	0.22 (0.02)	−1.84 (0.62)	0.80 (0.49)	0.06 (0.03)	−0.32 (0.05)	0.18 (0.10)
ϵ-greedy	0.15 (0.01)	−5.63 (1.04)	0.92 (0.67)	0.10 (0.02)	−0.5 (0.05)	−0.10 (0.05)

Table EC.4 Estimated population-level mean parameters for MAB algorithms in the 4-armed bandit experiment.

Algorithm	β (SM-3)	ϕ (SM-3)	ρ (SM-3)
UCB	0.057 (0.006)	−9.978 (1.149)	−2.486 (1.807)
TS	0.03 (0.01)	−13.17 (0.97)	−4.80 (1.68)
ϵ-greedy	0.13 (0.01)	−0.84 (0.12)	6.02 (0.73)

As shown in Tables EC.3 and EC.4, benchmark algorithms exhibit distinct parameter patterns that are consistent with their underlying decision rules: In the 2-armed bandit experiment, UCB drives directed exploration through an uncertainty bonus (higher ϕ, w_2 and w_3), while TS and $\epsilon-$greedy introduce greater random exploration (lower β) via prior sampling and stochastic choice mechanisms, respectively. In contrast, reasoning-enhanced LLMs display a mixed strategy, combining elements of both directed and random exploration, which leads to more human-like decision behavior. However, in the 4-armed bandit task, these algorithms also fail to engage in effective exploration.

EC.4.1. Experiments on Model Misspecification

As discussed in Section 4.4, we systematically vary the exploration rate (ϵ) of the ϵ-greedy algorithm. Specifically, under the identical stationary two-armed bandit setting with 300 rounds, we fit the SM-3 model using trajectories generated by the ϵ-greedy algorithm for $\epsilon \in \{0, 0.1, 0.2, \ldots, 0.9\}$, and the UCB algorithm for $c = 1, 2, 4, 8$ in $\sigma_t(k) \cdot \sqrt{\frac{c \log f(t)}{N_t(k)}}$. Figure EC.3 illustrates a clear down-

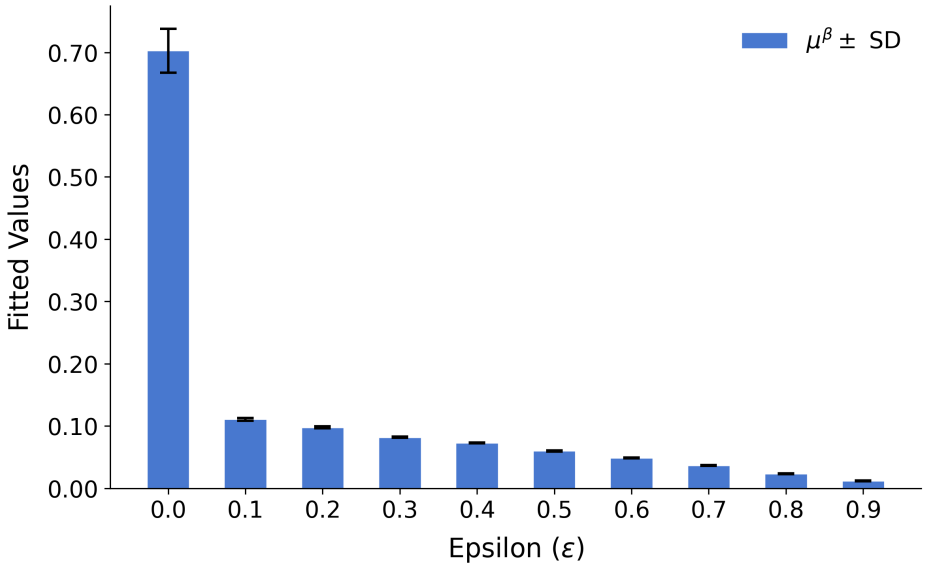

Figure EC.3 Estimated Random Exploration Parameter (β) across Varying Exploration Rates (ϵ)

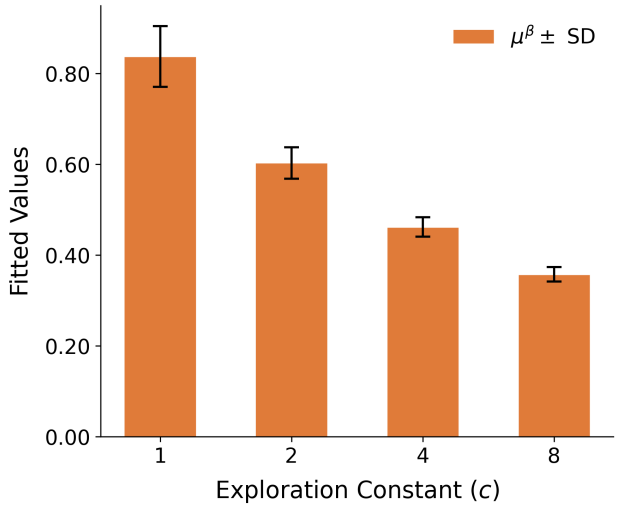

Figure EC.4 Estimated Random Exploration Parameter across Varying UCB Exploration Constants (c)

ward trend in the fitted β values as ϵ increases, appropriately reflecting the higher degree of random

exploration. Similarly, Figure EC.4 demonstrates that as the UCB exploration c increases, the estimated β also exhibits a monotone decrease. Furthermore, Table EC.5 and EC.6 detail the group posterior mean estimates for all three parameters across the varying ϵ and c values, respectively. For

Table EC.5 Estimated population-level mean parameters for the ε-Greedy algorithm across epsilon values. Values are presented as mean (SD).

ε	μ^{β}	μ^{ϕ}	μ^{ρ}
0.0	0.703 (0.035)	3.252 (0.092)	12.525 (0.487)
0.1	0.111 (0.002)	-4.530 (0.272)	9.530 (0.228)
0.2	0.098 (0.002)	-4.058 (0.372)	6.876 (0.176)
0.3	0.082 (0.001)	-4.014 (0.302)	5.100 (0.153)
0.4	0.073 (0.001)	-1.840 (0.263)	3.939 (0.164)
0.5	0.060 (0.001)	-0.723 (0.299)	3.006 (0.168)
0.6	0.049 (0.001)	2.159 (0.337)	2.340 (0.168)
0.7	0.037 (0.001)	4.572 (0.508)	1.536 (0.230)
0.8	0.023 (0.001)	10.682 (0.888)	0.765 (0.332)
0.9	0.012 (0.001)	27.068 (2.136)	1.488 (0.621)

Table EC.6 Estimated population-level mean parameters for the UCB algorithm across exploration constants c in $\hat{\sigma}_t(k) \cdot \sqrt{\frac{c \log f(t)}{N_t(k)}}$. Values are presented as mean (SD).

c	μ^{β} **(SD)**	μ^{φ} **(SD)**	μ^{ρ} **(SD)**
1	0.837 (0.067)	1.780 (0.099)	4.783 (0.526)
2	0.603 (0.035)	1.257 (0.137)	4.675 (0.524)
4	0.462 (0.022)	0.674 (0.134)	4.684 (0.420)
8	0.357 (0.016)	0.927 (0.138)	6.162 (0.568)

the ϵ-greedy algorithm, the model captures two concurrent behavioral signatures of increasing ϵ: a monotonic decline in the inverse temperature parameter (β), reflecting greater random exploration, and a parallel decrease in choice perseveration (ρ), consistent with more frequent unsystematic arm switching. For the UCB algorithm, the model consistently recovers a positive directed exploration effect ($\phi > 0$) across all values of c, while perseveration (ρ) remains comparatively stable.

EC.4.2. Details of fitted values

In this subsection, we present the details of all fitted values across bandit settings, agents, and choice models. Tables EC.7 and EC.8 report the posterior means and standard deviations of the population-level parameters estimated from the SM-3 model in the 4-armed bandit task. Table EC.7 summarizes the population-level means (μ^x) for each parameter x, while Table EC.8 reports the corresponding population-level standard deviations (σ^x).

Tables EC.9 and EC.10 present the posterior means and standard deviations of the population-level parameters estimated from the SM-2 and SM-1 models, respectively, in the 4-armed bandit

Table EC.7 Posterior means and standard deviations of μ^x estimated from the SM-3 model in the 4-armed bandit experiment

Agent/Prompt	μ^β	μ^ϕ	μ^ρ
ChatGPT-4o-mini	0.094(0.018)	-7.983(1.606)	6.555(2.819)
ChatGPT-4o-mini + CoT	0.044(0.004)	-10.458(0.995)	2.404(1.763)
ChatGPT-o3-mini (High-thinking)	0.103(0.005)	-4.190(0.283)	-3.691(0.835)
ChatGPT-o3-mini (low-thinking)	0.073(0.011)	-6.564(0.783)	12.079(2.364)
Gemini-2.0-flash	0.082(0.011)	-6.056(0.749)	10.926(2.827)
Gemini-2.0-flash + CoT	0.096(0.004)	-2.755(0.213)	0.726(0.670)
Gemini-2.5-flash (High-thinking)	0.109(0.005)	-2.926(0.209)	-2.059(0.632)
Gemini-2.5-flash (Non-thinking)	0.100(0.007)	-4.193(0.374)	4.640(1.514)
DeepSeek-V3	0.137(0.016)	-4.419(0.410)	-0.526(1.307)
DeepSeek-V3 + CoT	0.092(0.004)	-3.458(0.255)	0.818(0.780)
DeepSeek-R1	0.135(0.006)	-2.460(0.168)	-2.011(0.546)
Human Agents	0.168(0.010)	0.879(0.171)	5.450(0.299)
UCB	0.057(0.006)	-9.978(1.149)	-2.486(1.807)
ϵ - greedy	0.042(0.003)	-7.774(0.630)	15.925(1.896)
TS	0.029(0.002)	-13.179(0.972)	-4.797(1.684)

Table EC.8 Posterior means and standard deviations of σ^x estimated from the SM-3 model in the 4-armed bandit experiment

Agent/Prompt	σ^β	σ^ϕ	σ^ρ
ChatGPT-4o-mini	0.023(0.011)	0.165(0.115)	0.184(0.117)
ChatGPT-4o-mini + CoT	0.001(0.001)	0.183(0.113)	0.188(0.097)
ChatGPT-o3-mini (High-thinking)	0.004(0.003)	0.120(0.077)	0.189(0.114)
ChatGPT-o3-mini (Low-thinking)	0.025(0.013)	0.221(0.162)	0.182(0.111)
Gemini-2.0-flash	0.010(0.005)	0.173(0.122)	0.199(0.123)
Gemini-2.0-flash + CoT	0.002(0.002)	0.109(0.071)	0.200(0.111)
Gemini-2.5-flash (High-thinking)	0.003(0.002)	0.084(0.063)	0.175(0.109)
Gemini-2.5-flash (Non-thinking)	0.004(0.003)	0.185(0.132)	0.202(0.110)
DeepSeek-V3	0.027(0.016)	0.312(0.197)	0.205(0.118)
DeepSeek-V3 + CoT	0.003(0.002)	0.186(0.106)	0.181(0.111)
DeepSeek-R1	0.006(0.004)	0.140(0.090)	0.171(0.114)
Human Agents	0.053(0.008)	0.850(0.085)	0.268(0.162)
UCB	0.006(0.004)	0.182(0.116)	0.188(0.111)
ϵ - greedy	0.001(0.001)	0.177(0.116)	0.217(0.105)
TS	0.003(0.001)	0.149(0.106)	0.172(0.107)

experiment. Table EC.9 reports the group-level means (μ^x) and standard deviations (σ^x) for each parameter in the SM-2 model, while Table EC.10 shows the corresponding estimates for the SM-1 model.

Tables EC.11 and EC.12 report the posterior means and standard deviations of the population-level parameters estimated from the SM-3 model in the 2-armed bandit experiment. Table EC.11

Table EC.9 Posterior means and standard deviations of μ^x and σ^x estimated from the SM-2 model in the 4-armed bandit experiment

Agent/Prompt	μ^β	μ^ϕ	σ^β	σ^ϕ
ChatGPT-4o-mini	0.101(0.022)	-8.862(1.660)	0.032(0.016)	0.185(0.115)
ChatGPT-4o-mini + CoT	0.044(0.004)	-10.861(1.081)	0.002(0.001)	0.199(0.110)
ChatGPT-o3-mini (High-thinking)	0.098(0.005)	-3.944(0.279)	0.004(0.002)	0.123(0.074)
ChatGPT-o3-mini (Low-thinking)	0.068(0.008)	-8.157(0.728)	0.012(0.006)	0.357(0.269)
Gemini-2.0-flash	0.087(0.009)	-6.887(0.617)	0.010(0.005)	0.185(0.159)
Gemini-2.0-flash + CoT	0.098(0.004)	-2.761(0.197)	0.002(0.002)	0.113(0.072)
Gemini-2.5-flash (High-thinking)	0.105(0.005)	-2.830(0.190)	0.004(0.002)	0.104(0.062)
Gemini-2.5-flash (Non-thinking)	0.106(0.007)	-4.467(0.333)	0.004(0.003)	0.211(0.125)
DeepSeek-V3	0.138(0.016)	-4.317(0.378)	0.028(0.017)	0.327(0.181)
DeepSeek-V3 + CoT	0.092(0.004)	-3.540(0.251)	0.003(0.002)	0.196(0.108)
DeepSeek-R1	0.133(0.007)	-2.261(0.182)	0.006(0.004)	0.121(0.090)
Human Agents	0.168(0.010)	0.159(0.153)	0.051(0.008)	0.767(0.080)
UCB	0.058(0.006)	-9.303(0.973)	0.005(0.004)	0.179(0.110)
ϵ - greedy	0.043(0.003)	-9.975(0.690)	0.001(0.001)	0.212(0.121)
TS	0.030(0.002)	-12.325(0.925)	0.003(0.001)	0.176(0.103)

Table EC.10 Posterior means and standard deviations of μ^x and σ^x estimated from the SM-1 model in the 4-armed bandit task

Agent/Prompt	μ^β	σ^β
ChatGPT-4o-mini	0.142(0.025)	0.096(0.021)
ChatGPT-4o-mini + CoT	0.150(0.018)	0.067(0.015)
ChatGPT-o3-mini (High-thinking)	0.243(0.017)	0.062(0.015)
ChatGPT-o3-mini (Low-thinking)	0.183(0.033)	0.120(0.025)
Gemini-2.0-flash	0.163(0.033)	0.119(0.027)
Gemini-2.0-flash + CoT	0.187(0.005)	0.012(0.006)
Gemini-2.5-flash (High-thinking)	0.223(0.009)	0.027(0.008)
Gemini-2.5-flash (Non-thinking)	0.243(0.032)	0.122(0.027)
DeepSeek-V3	0.205(0.038)	0.140(0.030)
DeepSeek-V3 + CoT	0.202(0.013)	0.048(0.012)
DeepSeek-R1	0.257(0.013)	0.044(0.011)
Human Agents	0.161(0.011)	0.058(0.009)
UCB	0.143(0.013)	0.047(0.011)
ϵ - greedy	0.106(0.009)	0.032(0.007)
TS	0.076(0.009)	0.034(0.008)

summarizes the population-level means (μ^x) for each parameter, while Table EC.12 reports the corresponding population-level standard deviations (σ^x).

Tables EC.13 and EC.14 present the posterior means and standard deviations of the population-level parameters estimated from the SM-2 and SM-1 models, respectively, in the 2-armed bandit experiment. Table EC.13 reports the group-level means (μ^x) and standard deviations (σ^x) for each

Table EC.11 Posterior means and standard deviations of μ^x estimated from the SM-3 model in the 2-armed bandit experiment

Agent/Prompt	μ^β	μ^ϕ	μ^ρ
ChatGPT-4o-mini	0.206(0.010)	-0.264(0.093)	0.591(0.402)
ChatGPT-4o-mini + CoT	0.291(0.015)	0.108(0.071)	-0.099(0.332)
ChatGPT-o3-mini (High-thinking)	0.416(0.025)	2.562(0.092)	4.638(0.366)
ChatGPT-o3-mini (Low-thinking)	0.384(0.023)	1.198(0.079)	3.905(0.339)
Gemini-2.0-flash	0.262(0.014)	-0.114(0.074)	1.104(0.371)
Gemini-2.0-flash + CoT	0.297(0.015)	1.669(0.109)	1.936(0.325)
Gemini-2.5-flash (High-thinking)	0.449(0.030)	1.098(0.347)	0.235(0.301)
Gemini-2.5-flash (Non-thinking)	0.151(0.007)	0.361(0.110)	2.409(0.506)
DeepSeek-V3	0.265(0.016)	-0.220(0.077)	0.520(0.351)
DeepSeek-V3 + CoT	0.319(0.022)	2.552(0.174)	2.021(0.322)
DeepSeek-R1	0.590(0.059)	3.613(0.124)	4.719(0.300)
Human	0.436(0.024)	1.075(0.138)	1.076(0.273)
UCB	0.683(0.055)	1.402(0.315)	0.604(0.240)
ϵ - greedy	0.152(0.013)	-5.631(1.048)	0.928(0.673)
TS	0.224(0.017)	-1.842(0.625)	0.796(0.492)

Table EC.12 Posterior means and standard deviations of σ^x estimated from the SM-3 model in the 2-armed bandit experiment

Agent/Prompt	σ^β	σ^ϕ	σ^ρ
ChatGPT-4o-mini	0.011(0.009)	0.081(0.060)	0.328(0.215)
ChatGPT-4o-mini + CoT	0.024(0.014)	0.090(0.058)	0.330(0.218)
ChatGPT-o3-mini (High-thinking)	0.025(0.018)	0.120(0.075)	0.369(0.262)
ChatGPT-o3-mini (Low-thinking)	0.047(0.026)	0.229(0.090)	0.386(0.284)
Gemini-2.0-flash	0.022(0.014)	0.089(0.063)	0.334(0.214)
Gemini-2.0-flash + CoT	0.017(0.012)	0.257(0.104)	0.382(0.249)
Gemini-2.5-flash (High-thinking)	0.033(0.023)	0.365(0.219)	0.227(0.150)
Gemini-2.5-flash (Non-thinking)	0.008(0.005)	0.119(0.080)	0.420(0.300)
DeepSeek-V3	0.022(0.012)	0.056(0.046)	0.347(0.290)
DeepSeek-V3 + CoT	0.053(0.024)	0.498(0.149)	0.334(0.226)
DeepSeek-R1	0.154(0.065)	0.120(0.080)	0.340(0.218)
Human	0.135(0.022)	0.849(0.103)	1.546(0.224)
UCB	0.057(0.047)	0.307(0.206)	0.267(0.178)
ϵ - greedy	0.019(0.010)	0.922(0.578)	0.550(0.421)
TS	0.032(0.015)	0.433(0.289)	0.426(0.283)

parameter in the SM-2 model, while Table EC.14 shows the corresponding estimates for the SM-1 model.

Tables EC.15 and EC.16 report the posterior means and standard deviations of the population-level parameters estimated from the Probit model in the 2-armed bandit task. Table EC.15 summarizes the population-level means (μ^x) for each parameter, while Table EC.16 reports the corresponding population-level standard deviations (σ^x).

Table EC.13 Posterior means and standard deviations of μ^x and σ^x estimated from the SM-2 model in the 2-armed bandit experiment

Agent/Prompt	μ^β	μ^ϕ	σ^β	σ^ϕ
ChatGPT-4o-mini	0.208(0.012)	-0.329(0.072)	0.012(0.009)	0.071(0.057)
ChatGPT-4o-mini + CoT	0.291(0.014)	0.125(0.059)	0.023(0.014)	0.090(0.062)
ChatGPT-o3-mini (High-thinking)	0.469(0.024)	1.555(0.062)	0.028(0.019)	0.093(0.066)
ChatGPT-o3-mini (Low-thinking)	0.451(0.032)	0.660(0.069)	0.079(0.038)	0.178(0.068)
Gemini-2.0-flash	0.275(0.016)	-0.255(0.062)	0.030(0.014)	0.082(0.065)
Gemini-2.0-flash + CoT	0.324(0.015)	1.344(0.089)	0.019(0.013)	0.227(0.100)
Gemini-2.5-flash (High-thinking)	0.449(0.030)	0.940(0.281)	0.037(0.023)	0.341(0.250)
Gemini-2.5-flash (Non-thinking)	0.161(0.007)	0.053(0.079)	0.008(0.006)	0.106(0.078)
DeepSeek-V3	0.274(0.014)	-0.266(0.065)	0.024(0.016)	0.065(0.046)
DeepSeek-V3 + CoT	0.347(0.025)	2.098(0.126)	0.067(0.028)	0.361(0.122)
DeepSeek-R1	0.532(0.034)	2.006(0.072)	0.063(0.038)	0.103(0.068)
Human	0.463(0.027)	0.873(0.129)	0.151(0.024)	0.828(0.104)
UCB	0.674(0.056)	0.888(0.254)	0.075(0.050)	0.333(0.207)
ϵ - greedy	0.152(0.013)	-6.305(0.982)	0.020(0.010)	0.850(0.592)
TS	0.222(0.017)	-2.448(0.518)	0.032(0.015)	0.421(0.295)

Table EC.14 Posterior means and standard deviations of μ^x and σ^x estimated from the SM-1 model in the 2-armed bandit experiment

Agent/Prompt	μ^β	σ^β
ChatGPT-4o-mini	0.232(0.010)	0.016(0.011)
ChatGPT-4o-mini + CoT	0.277(0.011)	0.018(0.012)
ChatGPT-o3-mini (High-thinking)	0.247(0.010)	0.011(0.008)
ChatGPT-o3-mini (Low-thinking)	0.309(0.016)	0.030(0.018)
Gemini-2.0-flash	0.290(0.015)	0.033(0.016)
Gemini-2.0-flash + CoT	0.207(0.008)	0.009(0.007)
Gemini-2.5-flash (High-thinking)	0.386(0.020)	0.031(0.021)
Gemini-2.5-flash (Non-thinking)	0.161(0.007)	0.010(0.007)
DeepSeek-V3	0.300(0.015)	0.031(0.016)
DeepSeek-V3 + CoT	0.190(0.008)	0.010(0.008)
DeepSeek-R1	0.241(0.010)	0.012(0.008)
Human	0.313(0.014)	0.080(0.013)
UCB	0.556(0.032)	0.038(0.027)
ϵ - greedy	0.260(0.024)	0.072(0.026)
TS	0.286(0.021)	0.061(0.022)

Table EC.17 reports the average cumulative regret and exploitation rates in the final round of both bandit experiments. Results are presented for a range of LLM agents under various prompting conditions, including human participants and the UCB algorithm. Standard errors are provided for regret. The comparison highlights systematic differences in performance across models, thinking modes, and task complexity.

Table EC.15 **Posterior means and standard deviations of μ^x estimated from the probit model in the 2-armed bandit experiment**

Agent/Prompt	μ^{w_1}	μ^{w_2}	μ^{w_3}
ChatGPT-4o-mini	0.162(0.012)	-0.028(0.010)	-0.181(0.040)
ChatGPT-4o-mini + CoT	0.195(0.014)	0.033(0.012)	-0.169(0.048)
ChatGPT-o3-mini (High-thinking)	0.003(0.002)	0.383(0.024)	1.860(0.092)
ChatGPT-o3-mini (Low-thinking)	-0.106(0.022)	0.073(0.013)	2.336(0.195)
Gemini-2.0-flash	0.135(0.014)	-0.042(0.010)	0.046(0.052)
Gemini-2.0-flash + CoT	0.016(0.013)	0.146(0.021)	0.720(0.050)
Gemini-2.5-flash (High-thinking)	-0.363(0.059)	0.250(0.111)	2.114(0.230)
Gemini-2.5-flash (Non-thinking)	0.019(0.007)	-0.012(0.007)	0.317(0.033)
DeepSeek-V3	0.355(0.027)	-0.007(0.014)	-0.711(0.078)
DeepSeek-V3 + CoT	0.001(0.015)	0.205(0.023)	0.638(0.057)
DeepSeek-R1	0.006(0.004)	0.660(0.066)	1.952(0.217)
Human	0.025(0.021)	0.140(0.023)	0.877(0.091)
UCB	0.071(0.042)	0.408(0.106)	1.056(0.166)
ϵ - greedy	0.101(0.018)	-0.563(0.051)	-0.096(0.057)
TS	0.064(0.029)	-0.321(0.053)	0.178(0.099)

Table EC.16 **Posterior means and standard deviations of σ^x estimated from the probit model in the 2-armed bandit experiment**

Agent/Prompt	σ^{w_1}	σ^{w_2}	σ^{w_3}
ChatGPT-4o-mini	0.010(0.006)	0.011(0.008)	0.028(0.019)
ChatGPT-4o-mini + CoT	0.013(0.008)	0.018(0.012)	0.037(0.025)
ChatGPT-o3-mini (High-thinking)	0.001(0.001)	0.021(0.014)	0.067(0.050)
ChatGPT-o3-mini (Low-thinking)	0.013(0.011)	0.023(0.014)	0.245(0.112)
Gemini-2.0-flash	0.023(0.010)	0.015(0.010)	0.059(0.038)
Gemini-2.0-flash + CoT	0.012(0.007)	0.019(0.015)	0.070(0.044)
Gemini-2.5-flash (High-thinking)	0.015(0.011)	0.298(0.095)	0.063(0.047)
Gemini-2.5-flash (Non-thinking)	0.006(0.004)	0.009(0.006)	0.034(0.021)
DeepSeek-V3	0.019(0.012)	0.020(0.014)	0.033(0.028)
DeepSeek-V3 + CoT	0.030(0.012)	0.057(0.024)	0.102(0.049)
DeepSeek-R1	0.002(0.002)	0.166(0.066)	0.671(0.164)
Human	0.113(0.019)	0.139(0.020)	0.498(0.080)
UCB	0.019(0.015)	0.133(0.087)	0.133(0.081)
ϵ - greedy	0.012(0.007)	0.096(0.056)	0.028(0.026)
TS	0.022(0.012)	0.057(0.040)	0.069(0.039)

EC.5. LLM versions, dates, and API endpoints

Tables EC.18, EC.19, and EC.20 show the model versions, dates, and API for each model used in the experiments. Most models were invoked through the `openai` Python library (version 1.66.2). Because the supplementary experiments were conducted after the main experiments, some providers had updated the available model versions. In particular, the main experiments used `DeepSeek-V3-0324`, whereas the temperature and 300-round experiments used

Table EC.17 Average cumulative regret and exploitation rates in the last round across both experiments, with standard errors reported for regret.

	2-armed bandit task		4-armed bandit task	
Agent/Prompt	Regret	Exploit Rate	Regret	Exploit Rate
ChatGPT-4o-mini	13.467(0.911)	0.897	3599.267(352.296)	0.653
ChatGPT-4o-mini+CoT	12.65(0.725)	0.921	2472.733(320.921)	0.698
ChatGPT-o3-mini (High-thinking)	14.143(0.492)	0.952	1803.6(235.734)	0.821
ChatGPT-o3-mini (Low-thinking)	11.923(0.528)	0.962	3064.133(432.816)	0.708
Gemini-2.0-flash	10.963(0.613)	0.905	3685.8(292.369)	0.66
Gemini-2.0-flash+ CoT	15.653(0.448)	0.914	1504.4(169.034)	0.814
Gemini-2.5-flash (High-thinking)	15.74(0.437)	0.912	1551.067(171.668)	0.84
Gemini-2.5-flash (Low-thinking)	19.713(0.738)	0.84	2551.0(309.847)	0.762
DeepSeek-V3	10.943(0.693)	0.908	3089.467(314.202)	0.7
DeepSeek-V3 + CoT	16.55(0.556)	0.903	1678.133(216.983)	0.802
DeepSeek-R1	13.987(0.464)	0.943	1532.4(246.638)	0.838
Human	11.281(0.398)	0.893	1509.355(125.098)	0.768
UCB	14.723(0.446)	0.908	2307.933(134.985)	0.698

`DeepSeek-V3.2`. We therefore treat these supplementary experiments as within-experiment robustness and extension analyses rather than exact replications using identical model snapshots. Comparisons are made within each experimental setting, and differences across tables should not be interpreted as being driven solely by temperature or horizon when the underlying model version also differs. In the 2-armed bandit experiment with 300 rounds, the Gemini-2.5-flash model was invoked through `google-genai` library (version 3.0.0), and the Gemini-2.0-flash model was invoked by directly posting to OpenRouter's API endpoint, with the model specified as `google/gemini-2.0-flash-001` in the request body.

Model name	**Date**	**Version**	**API endpoint**
`gpt-4o-mini`	Mar. 20, 2025	`gpt-4o-mini-2024-07-18`	`https://api.openai.com/v1/chat/completions`
`o3-mini`	Apr. 2, 2025	`o3-mini-2025-01-31`	`https://api.openai.com/v1/chat/completions`
`gemini-2.5-flash`	Aug. 4, 2025	`gemini-2.5-flash`	`https://generativelanguage.googleapis.com/v1beta/openai/chat/completions`
`gemini-2.0-flash`	Apr. 2, 2025	`gemini-2.0-flash-001`	`https://generativelanguage.googleapis.com/v1beta/openai/chat/completions`
`deepseek-reasoner`	Mar. 22, 2025	`DeepSeek-R1`	`https://api.deepseek.com/chat/completions`
`deepseek-chat`	Apr. 10, 2025	`DeepSeek-V3-0324`	`https://api.deepseek.com/chat/completions`

Table EC.18 Model names, experiment dates, model versions, and API endpoints used in the main experiments.

Model name	Experiment date	Version	API endpoint
`gpt-4o-mini`	Mar 31, 2026	`gpt-4o-mini-2024-07-18`	`https://api.openai.com/v1/chat/completions`
`gemini-2.0-flash`	Mar 17, 2026	`gemini-2.0-flash-001`	`https://openrouter.ai/api/v1/chat/completions`
`deepseek-chat`	Mar 17, 2026	`DeepSeek-V3.2`	`https://api.deepseek.com/chat/completions`

Table EC.19 Model names, experiment dates, model versions, and API endpoints used in the experiments with high and low temperatures.

Model name	Experiment date	Version	API endpoint
`o3-mini`	Feb 26, 2026	`o3-mini-2025-01-31`	`https://api.openai.com/v1/chat/completions`
`gpt-4o`	Feb 26, 2026	`gpt-4o-mini-2024-07-18`	`https://api.openai.com/v1/chat/completions`
`deepseek-reasoner`	Jan 6, 2026	`DeepSeek-V3.2`	`https://api.deepseek.com/chat/completions`
`deepseek-chat`	Feb 24, 2026	`DeepSeek-V3.2`	`https://api.deepseek.com/chat/completions`
`gemini-2.0-flash`	Mar 4, 2026	`gemini-2.0-flash-001`	`https://openrouter.ai/api/v1/chat/completions`
`gemini-2.5-flash`	Mar 4, 2026	`gemini-2.5-flash`	`https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash:generateContent`

Table EC.20 Model names, experiment dates, model versions, and API endpoints used in the 2-armed bandit experiments with 300 rounds.