BEYOND FIXED ACQUISITION FUNCTIONS: TEMPORAL DIFFERENCE ACTIVE LEARNING FOR PERTURBATION DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Genetic and drug perturbation screens are important tools for elucidating disease mechanisms and validating therapeutic targets, but experimental costs limit the number of candidates that can be tested. Existing perturbation screening methods typically update the surrogate model with feedback while keeping the acquisition function fixed, limiting their ability to adapt selection to different response landscapes. Experimental feedback reveals current hits and informs subsequent selection by updating candidate predictions, providing a training signal for learning how to select experiments. We formulate budgeted perturbation discovery as a sequential decision problem and propose Temporal Difference Active Learning (TDAL), which learns an acquisition function online through temporal difference learning from a single screening trajectory, accounting for current hits and future discoveries. TDAL uses the Hellinger distance between hit probabilities for unmeasured candidates before and after an experiment to weight successor value, and reuses measured outcomes through leave-batch-out (LBO) to construct virtual transitions and alleviate training data scarcity. Across eight public screens spanning single, double, and triple perturbations under varied experimental configurations, TDAL achieves an average rank of 1.63 among eleven methods and ranks first on four tasks, demonstrating its adaptability across response landscapes.

1 INTRODUCTION

Large-scale perturbational screens deepen the understanding of disease mechanisms and help validate therapeutic targets (Schmidt et al., 2022; Zhuang et al., 2019; Sanchez et al., 2021). The human genome comprises nearly 20,000 single-gene candidates, roughly 200 million pairs, and one trillion triples in terms of perturbation combinations. The costs of testing each perturbation and measuring its phenotype make exhaustive screening infeasible (Mehrjou et al., 2022; Pacchiano et al., 2023). Closed-loop systems have shown that model-guided experiment selection can improve biological discovery under limited resources (Xu et al., 2026; DeMeo et al., 2025). It is thus important to develop strategies that maximize cumulative discoveries within a fixed experimental budget.

Active learning addresses this challenge by iteratively fitting a surrogate model to measured outcomes and using an acquisition function to select subsequent experiments (Figure 1a). GeneDisco evaluates active learning methods for perturbation screening and reports substantial variation in their performance across experimental settings (Mehrjou et al., 2022). Existing methods employ different selection criteria to guide perturbation discovery. OAE (Pacchiano et al., 2023) seeks near-optimal perturbations, DiscoBAX (Lyle et al., 2023) considers response strength and mechanistic diversity, and Probability-of-Hit (PoH) (Rubbi et al., 2026) prioritizes perturbations that exceed a predefined phenotypic threshold. Recent approaches also incorporate language model agents or biological priors into experiment selection (Roohani et al., 2025; Hao et al., 2025; Li et al., 2026a).

Indeed, different response landscapes favor different selection strategies (Figure 1b). In conventional active learning, experimental feedback updates the surrogate model and changes subsequent selections, while the acquisition rule itself typically remains predefined. The same feedback can also inform how experiments should be selected: each experiment yields immediate discoveries and updates predictions for the remaining candidates, influencing future discovery opportunities. This

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.

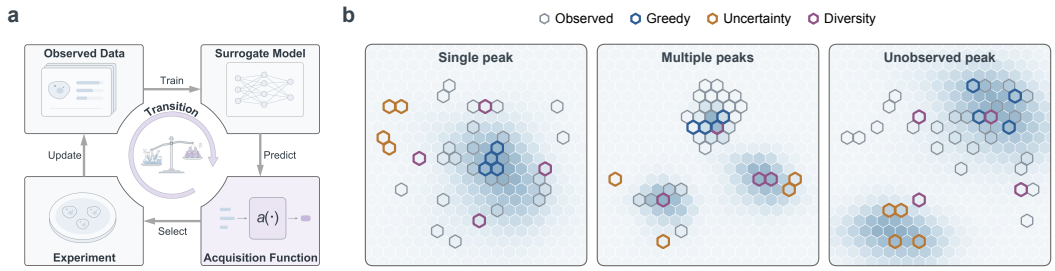


Figure 1: **a,** Active learning fits a surrogate model, selects experiments with an acquisition function, and updates the model with new observations (gray arrows). TDAL learns a dynamic acquisition function from accumulated transitions (purple path). **b,** A surrogate model is fitted to observations. Fixed acquisition functions (Greedy, Uncertainty, and Diversity) use its predictions to select the next batch, yielding different selections across illustrative landscapes. Hexagon fills show responses. Gray and colored outlines mark observed candidates and next batch selections, respectively.

motivates learning an acquisition function that accounts for both current and future discoveries from experimental feedback. We therefore formulate screening as a sequential decision problem and use experimental transitions to learn a dynamic acquisition function online (Figure 1a). However, unlike reinforcement learning with repeated episodes, a screening campaign follows a single trajectory that cannot be reset, with each candidate queried at most once and only one real transition generated per round. Learning an acquisition function from such limited feedback presents a particular challenge.

Here we introduce Temporal Difference Active Learning (TDAL), which learns a dynamic acquisition function online from experimental feedback to maximize cumulative discoveries. Building on surrogate hit probabilities, TDAL uses temporal difference learning to incorporate observed discoveries and feedback-weighted successor value into experiment selection. It weights successor value according to changes in predicted hit probabilities for the remaining candidates, reflecting how experimental feedback updates their predicted outcomes. To alleviate the scarcity of real transitions, TDAL constructs additional training transitions from previously measured outcomes through leave-batch-out (LBO). Our main contributions are threefold.

- We formulate perturbation discovery as a sequential decision problem and learn a dynamic acquisition function from experimental feedback to adapt to different response landscapes.
- We introduce TDAL, which learns online from a single screening trajectory through temporal difference learning. It uses the Hellinger distance to weight successor value and LBO to construct additional training transitions, alleviating training data scarcity.
- We evaluate TDAL across eight screens spanning single, double, and triple perturbations. TDAL ranks first on four tasks and achieves an average rank of 1.63 among eleven methods.

2 Related Work

2.1 Active Learning and Acquisition Function Learning

Active learning alternates between sample selection and model updates to improve predictions with fewer labels. Methods include BALD, which uses information gain (Houlsby et al., 2011), CoreSet, which emphasizes geometric coverage (Sener & Savarese, 2018), and BADGE, which combines uncertainty and diversity (Ash et al., 2020). For discovery, active search (Garnett et al., 2012), ENS (Jiang et al., 2017), and its batch extension (Jiang et al., 2018) evaluate future hits within the budget. IDS (Xu et al., 2022) balances instantaneous regret and information gain to increase cumulative response. These methods adapt sampling to observations but retain predefined acquisition functions.

Other approaches learn acquisition functions from data. Learning Active Learning predicts expected error reduction (Konyushkova et al., 2017). MetaBO trains acquisition functions through reinforcement learning across tasks (Volpp et al., 2020). Amortized active search trains a neural network to imitate ENS selections (Nguyen et al., 2024). In molecular discovery, GLARE updates its selection

policy online using measured activity labels (Chen et al., 2025), while Zhang et al. (2026) learn a policy to identify the optimal molecule. SOFisher uses Double DQN to select spatial-omics fields of view (Li et al., 2026b). Our work focuses on perturbation screens, where TDAL learns a dynamic acquisition function online from experimental feedback.

2.2 Active Learning for Perturbation Discovery

Active learning guides the discovery of perturbations with target phenotypes under a limited experimental budget. GeneDisco provides a unified benchmark for gene perturbations (Mehrjou et al., 2022). OAE seeks near-optimal perturbations with high responses (Pacchiano et al., 2023), DiscoBAX balances response strength and mechanistic diversity (Lyle et al., 2023), and PoH selects candidates by their posterior probability of exceeding a predefined phenotypic threshold (Rubbi et al., 2026). RECOVER and NAIAD target drug and genetic combination discovery, respectively (Bertin et al., 2023; Qin et al., 2025), while BATCHIE uses information-driven batch design to improve predictions of drug combination responses (Tosh et al., 2025). These methods update surrogate models from observations while typically keeping acquisition functions fixed.

Biological priors and agents guide selection. IterPert improves predictions with multimodal priors (Huang et al., 2024), while BioBO combines gene representations and pathway information (Li et al., 2026a). Soper et al. (2025) and DeepAL (Zhu et al., 2026) combine knowledge graphs with Bayesian matrix factorization and representation learning, respectively, for double gene perturbation discovery. BioDiscoveryAgent and PerTurboAgent design perturbations using language models, experimental history, and tools (Roohani et al., 2025; Hao et al., 2025). AIDO.Builder updates models from feedback and reuses experience through memory (Guo et al., 2026). Gupta et al. (2025) found the tested agents insensitive to permuted feedback, whereas Wainrib et al. (2026) reported discovery gains from feedback beyond prior knowledge. We construct transitions from feedback to learn a dynamic acquisition function that adapts to different response landscapes (Appendix A.1).

3 Problem Formulation

We consider a fixed candidate pool and aim to maximize expected cumulative discoveries of perturbations that induce desired phenotypic responses within a fixed experimental budget. We formulate this task as a sequential decision problem and define an objective combining immediate discoveries with successor value weighted according to experimental feedback (notation in Appendix A.2).

3.1 Sequential Perturbation Discovery

Let $\mathcal{X}$ be a finite pool of gene or gene combination perturbations. Testing a candidate $x \in \mathcal{X}$ yields a scalar phenotype $e(x)$. We align effect directions across assays so that larger $e(x)$ indicates closer agreement with the target phenotype. For a threshold τ fixed before screening, the hit indicator is $z(x) = \mathbf{1}\{e(x) \geq \tau\}$ (Rubbi et al., 2026). Let D_0 contain the n_0 candidate–phenotype pairs $(x, e(x))$ observed before sequential selection begins. The phenotypes of initially unmeasured candidates $x \in U_0$ are revealed only when queried. The subsequent screening process consists of T rounds with batch size B, subject to the experimental budget.

In the experimental loop shown in Figure 2, let A_t denote the selected batch at round $t \in \{0, \ldots, T-1\}$ and $O_t = \{(x, e(x)) : x \in A_t\}$ its observed outcomes. Before selecting A_t, the available history is $H_t = (D_0, (A_j, O_j)_{0 \leq j < t})$. The measured dataset and unmeasured candidate pool are

$$D_t = D_0 \cup \bigcup\nolimits_{0 \leq j < t} O_j, \qquad U_t = \mathcal{X} \setminus \{x : (x, e(x)) \in D_t\}. \tag{1}$$

A surrogate model trained on D_t estimates hit probabilities $\hat{p}_t(x)$ for $x \in U_t$. The acquisition function selects $A_t \subseteq U_t$ using the available history and surrogate predictions. Experimentally testing the selected batch yields O_t and a reward equal to the number of discoveries in round t: $R_t = \sum_{x \in A_t} z(x)$. Appending (A_t, O_t) to the history gives H_{t+1}, with $D_{t+1} = D_t \cup O_t$ and $U_{t+1} = U_t \setminus A_t$. Each round thus produces a transition (H_t, A_t, R_t, H_{t+1}). After T rounds, the final output consists of all verified hit candidates recorded in D_T, whose cumulative discovery count is $C_T = \sum_{(x, e(x)) \in D_T} z(x)$.

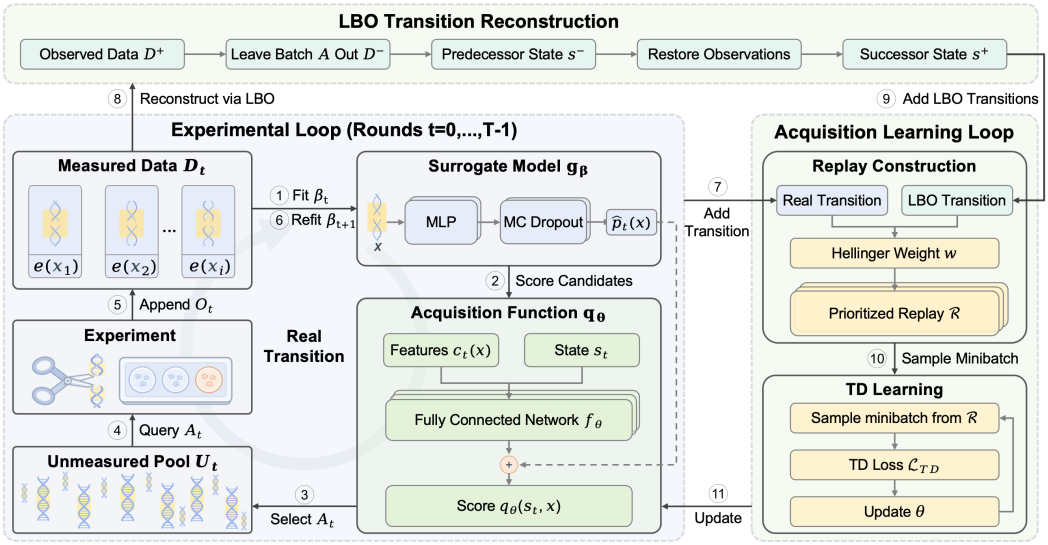


Figure 2: **TDAL framework.** The surrogate is fitted to measurements, and the acquisition function sums hit probabilities and network outputs to select the next batch. TDAL incorporates new observations, refits the surrogate, and constructs real transitions. These enter replay alongside virtual LBO transitions reconstructed from measurements, training the acquisition function through TD learning with Hellinger distance weighting successor value according to prediction changes.

3.2 Sequential Decision Objective

The screening goal is to maximize expected final discoveries $\mathbb{E}_\pi[C_T \mid H_0]$. Each experiment yields the current discovery count and updates the surrogate model's predictions for unmeasured candidates, informing estimates of future discovery value and selection in the next round. We use the current discovery count as the immediate reward and adjust the successor value weight according to prediction changes before and after the experiment, defining a surrogate learning objective weighted by experimental feedback. Let $w_t = w(H_t, A_t, H_{t+1}) \in [0, 1]$ denote the successor value weight determined by the observed transition. For a selection policy π, define

$$J_w(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1}\left(\prod_{j=0}^{t-1} w_j\right) R_t \,\middle|\, H_0\right]. \tag{2}$$

We seek a selection policy π that maximizes $J_w(\pi)$. When $w_t = 1$ for all t, this is equivalent to maximizing expected final discoveries $\mathbb{E}_\pi[C_T \mid H_0]$.

To maximize J_w, let $Q_t^{*,w}(H_t, A)$ denote the expected weighted cumulative discoveries from round t to $T-1$, when batch A is selected given history H_t and subsequent selections follow the optimal policy. This value combines current discoveries with weighted successor value through the Bellman equation for a finite horizon:

$$Q_t^{*,w}(H_t, A) = \mathbb{E}\left[R_t + w_t \max_{A' \subseteq U_{t+1}} Q_{t+1}^{*,w}(H_{t+1}, A') \,\middle|\, H_t, A_t = A\right]. \tag{3}$$

At each round, the optimal selection policy π_w^* selects the batch maximizing $Q_t^{*,w}$ given the current history, thereby maximizing the weighted cumulative discovery objective J_w:

$$\pi_w^*(H_t) \in \arg\max_{A \subseteq U_t} Q_t^{*,w}(H_t, A), J_w(\pi_w^*) = \max_\pi J_w(\pi) = \max_{A \subseteq U_0} Q_0^{*,w}(H_0, A). \tag{4}$$

4 Temporal Difference Active Learning

Within this sequential decision framework, TDAL uses experimental feedback to update a dynamic acquisition function online through TD learning, accounting for current and future discoveries while adapting selection to different response landscapes (Figure 2). See Algorithm 1 (Appendix A.3).

4.1 Dynamic Acquisition Function

The optimal value $Q_t^{*,w}(H_t, A)$ involves expectations over future experimental feedback and is difficult to compute directly. TDAL approximates this value using a dynamic acquisition function learned online from experimental feedback and built on surrogate predictions.

Surrogate model. In the active learning loop, the surrogate model approximates the posterior predictive distribution of unmeasured outcomes, providing a reference for estimating immediate discoveries. We use an MLP g_β with MC dropout and fit parameters β_t to data D_t. For each unmeasured candidate, we estimate $\hat{p}_t(x)$ as the proportion of sampled predictions at or above threshold τ.

Value decomposition. Given the measured data D_t, $\widehat{R}_t(A) = \sum_{x \in A} \hat{p}_t(x)$ estimates the number of discoveries in the unmeasured batch A. Thus, the optimal batch value can be written as

$$Q_t^{*,w}(H_t, A) = \widehat{R}_t(A) + \underbrace{\mathbb{E}\Big[R_t - \widehat{R}_t(A) + w_t \max_{A'} Q_{t+1}^{*,w}(H_{t+1}, A') \,\Big|\, H_t, A_t = A\Big]}_{F_t^*(H_t, A)}. \quad (5)$$

The term $F_t^*(H_t, A)$ accounts for the expected error in this estimate and the weighted value of subsequent discoveries. This decomposition allows value learning to build on the surrogate's estimate of immediate discoveries, which provides a reference grounded in measured data.

Acquisition function parameterization. We approximate $F_t^*(H_t, A)$ using the summed outputs $F_\theta(s_t, A)$ of a fully connected acquisition network f_θ. The state s_t summarizes screening progress, measured outcomes, and surrogate prediction statistics for unmeasured candidates. The candidate features $c_t(x)$ combine a fixed biological representation with the surrogate model's predictions. Adding each candidate's network output to its estimated hit probability yields a dynamic acquisition function that scores candidates based on s_t and $c_t(x)$ (Appendices A.4–A.6)

$$q_\theta(s_t, x) = \hat{p}_t(x) + f_\theta(s_t, c_t(x)). \quad (6)$$

We use an additive approximation Q_θ to make batch maximization tractable. Selecting the B unmeasured candidates with the highest scores maximizes

$$Q_\theta(s_t, A) = \sum\nolimits_{x \in A} q_\theta(s_t, x) \approx Q_t^{*,w}(H_t, A). \quad (7)$$

4.2 Temporal Difference Learning

We train the acquisition network f_θ using real and LBO transitions (Appendix A.7). Each transition provides an observed discovery count and a successor state, supplying a learning signal for the acquisition function to adapt its selection strategy to different response landscapes.

Real transitions. After querying batch A_t, we obtain observations O_t and discovery count R_t. We refit the surrogate on D_{t+1} and construct the successor selection state s_{t+1}. With successor value weight w_t, the round is stored as the training record $\xi_t = (s_t, A_t, R_t, w_t, s_{t+1})$.

Virtual transitions through LBO. Screening follows a single trajectory, which cannot be reset to test alternative selection sequences within the same budget. Each round therefore provides only one real batch transition. LBO supplements these transitions by temporarily hiding measured outcomes and then restoring them to reconstruct experimental feedback. Let D^+ and U^+ denote the current measured dataset and unmeasured pool. For a batch A of measured candidates, hiding its outcomes gives $D^- = D^+ \setminus \{(x, e(x)) : x \in A\}$. We refit the surrogate model on D^- to construct the predecessor state s^-, and construct the successor state s^+ from D^+ with the outcomes restored. The discovery count R is computed from the batch's observed outcomes. We compute w by comparing hit probability estimates before and after restoration over U^+. The resulting virtual transition is $\widetilde{\xi} = (s^-, A, R, w, s^+)$. Real and virtual transitions enter the same replay buffer $\mathcal{R}$, providing more training data without additional experiments.

Feedback weights. For each candidate, hit and nonhit probabilities define a Bernoulli distribution. For both real and virtual transitions, we use Hellinger distance (Hellinger, 1909) to compare these distributions before and after feedback over the remaining pool. Unchanged predictions across U_{t+1}

give $w_t = 0$, while larger Hellinger distances assign more weight to successor value:

$$w_t^2 = \frac{\sum_{x \in U_{t+1}} H^2_{\mathrm{Ber}}(\hat{p}_t(x), \hat{p}_{t+1}(x))}{|U_{t+1}|}, \qquad H^2_{\mathrm{Ber}}(p,q) = 1 - \sqrt{pq} - \sqrt{(1-p)(1-q)}. \tag{8}$$

Updating the acquisition function through TD learning. For a replay transition $\xi = (s, A, R, w, s')$, we follow Double DQN. The current acquisition function selects the successor batch A'_θ, and the target network evaluates its batch value $Q_{\bar{\theta}}(s', A'_\theta)$ (van Hasselt et al., 2016). Here, $\bar{\theta}$ denotes the target network parameters. The current discovery count and successor value weighted by w form the TD target Y. Both real and LBO transitions use this target. During each update, the stored surrogate predictions remain fixed, and we optimize only θ.

$$Y = R + wQ_{\bar{\theta}}(s', A'_\theta), \qquad \mathcal{L}_{\mathrm{TD}}(\theta) = \mathbb{E}_{\xi \sim \mathcal{R}} \left[\mathrm{Huber}(Q_\theta(s, A) - Y) \right]. \tag{9}$$

4.3 Theoretical Analysis

We analyze weighted Bellman optimality, predictive improvement, and the final discovery gap.

Proposition 1 (Weighted Bellman optimality). For a finite horizon, let $Q_t^{*,w}$ satisfy the Bellman recursion in equation 3, and let π_w^* select a feasible batch maximizing $Q_t^{*,w}(H_t, A)$ at every reachable history. Then $J_w(\pi_w^*) = \max_\pi J_w(\pi)$. Thus, optimal batch selection maximizes the weighted discovery objective (Appendix B.1).

Proposition 2 (Feedback and predictive improvement). Fix H_t, A_t and $U = U_{t+1}$, with $|U| > 0$. Let $p_x = \Pr(z(x) = 1 \mid H_t, A_t)$ and $q_x = \Pr(z(x) = 1 \mid H_t, A_t, O_t)$ denote hit probabilities before and after feedback. Let Δ_{pred} denote the expected decrease in squared loss from $(p_x - z(x))^2$ to $(q_x - z(x))^2$, averaged over $x \in U$ (Gneiting & Raftery, 2007). With w_t computed from these probabilities, the expected predictive improvement conditional on H_t, A_t satisfies

$$\mathbb{E}[w_t^4] \le \Delta_{\mathrm{pred}} = \frac{1}{|U|} \sum\nolimits_{x \in U} \mathbb{E}[(q_x - p_x)^2] \le 2\mathbb{E}[w_t^2]. \tag{10}$$

This bounds predictive improvement using the Hellinger weight. For a query a revealing only $z(a)$, let $r = \Pr(z(a) = 1) \in (0, 1)$, $p_x^+ = \Pr(z(x) = 1 \mid z(a) = 1)$, and $p_x^- = \Pr(z(x) = 1 \mid z(a) = 0)$, suppressing the same conditioning. The predictive improvement decomposes as

$$\Delta_{\mathrm{pred}}(a) = \frac{r(1-r)}{|U|} \sum\nolimits_{x \in U} (p_x^+ - p_x^-)^2. \tag{11}$$

Outcome uncertainty $r(1-r)$ multiplies the mean squared difference between remaining predictions under the two outcomes. Queries without hits can therefore inform selection (Appendix B.2).

Proposition 3 (Final discovery gap). Under the same initial information and budget, let π_C^* maximize expected final discoveries and let π_θ select a feasible batch maximizing $Q_\theta(s_t, A)$ at each round. For feedback weights $w_t \in [0, 1]$, write $W_t = \prod_{j=0}^{t-1} w_j$, with $W_0 = 1$. Define the value error over feasible batches as $\varepsilon_t^w(H_t) = \max_A |Q_\theta(s_t, A) - Q_t^{*,w}(H_t, A)|$ and the objective weighting bias as $\Delta_C(w) = \mathbb{E}_{\pi_C^*}[\sum_{t=0}^{T-1}(1 - W_t)R_t \mid H_0]$. The final discovery gap satisfies

$$\mathbb{E}_{\pi_C^*}[C_T \mid H_0] - \mathbb{E}_{\pi_\theta}[C_T \mid H_0] \le \Delta_C(w) + 2\mathbb{E}_{\pi_\theta}\left[\sum\nolimits_{t=0}^{T-1} W_t \varepsilon_t^w(H_t) \mid H_0 \right]. \tag{12}$$

The first term measures discovery return attenuated by weighting. The second bounds selection losses from value estimation error. Small terms imply near-optimal expected final discoveries. For constant weights under a common value error bound, smaller weights lower the error bound but raise the bias bound. Larger weights lower the bias bound but raise the error bound (Appendix B.3).

5 Experiments and Results

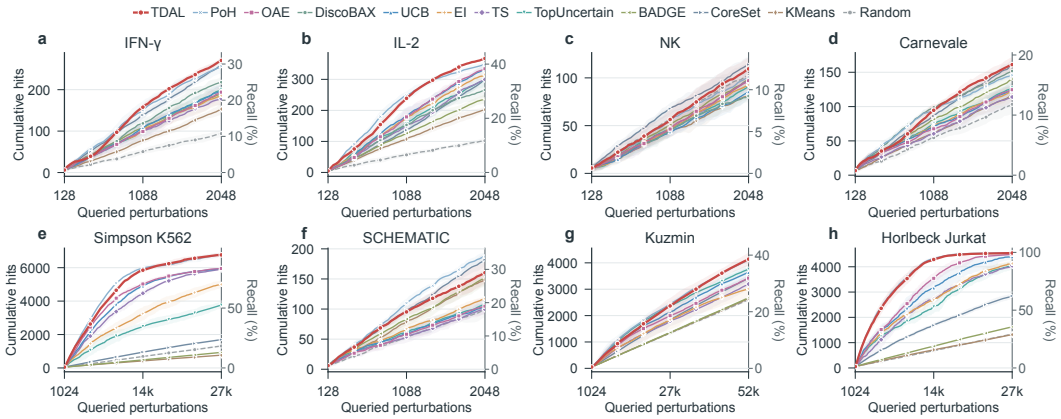
We evaluate hit discovery performance across eight public genetic perturbation datasets under fixed experimental budgets. Then we examine how local landscape structure affects active discovery, assess individual components through ablations, and analyze candidate rankings over selection rounds.

5.1 Experimental Setup

Data and budgets. We evaluate TDAL on eight public datasets recording phenotypic responses to genetic perturbations, aiming to identify perturbations producing desired phenotypic changes (Appendix C.1). These comprise four single perturbation tasks (Schmidt et al., 2022; Zhuang et al., 2019; Carnevale et al., 2022), three double perturbation tasks (Fong et al., 2025; Simpson et al., 2023; Horlbeck et al., 2018), and one triple perturbation task (Kuzmin et al., 2020), with candidate pools ranging from 11,067 to 214,642. Initial sample sizes, batch sizes, and budgets vary across tasks (Table C2).

Experimental protocol. We conduct retrospective sequential evaluation using measured phenotypes from these genetic perturbation datasets. Candidate identities and fixed representations are available at the start. Each method initially receives the phenotypes of n_0 candidates, and the remaining outcomes are revealed as candidates are selected in subsequent rounds. We fix the hit threshold τ before screening to target the top 5% of candidates in the predefined effect direction (Mehrjou et al., 2022). Selection aims to maximize final cumulative hits C_T within the budget, and recall $\mathrm{Recall}_T = C_T / \sum_{x \in \mathcal{X}} z(x)$ measures discovery coverage. Within each of five seeds (0–4), methods share the candidate pool, initial data, budget, and hit threshold (Appendix C.2).

Baselines. We compare diverse baselines with publicly available implementations (Appendix C.3). Perturbation discovery baselines include PoH (Rubbi et al., 2026), OAE (Pacchiano et al., 2023), and DiscoBAX (Lyle et al., 2023). General acquisition functions include UCB (Srinivas et al., 2010), Expected Improvement (Jones et al., 1998), and Thompson Sampling (Thompson, 1933). Active learning baselines include BADGE (Ash et al., 2020), CoreSet (Sener & Savarese, 2018), and TopUncertain and KMeans-Embedding from GeneDisco (Mehrjou et al., 2022). Random serves as a control. For computational scalability, we evaluate DiscoBAX on four single perturbation tasks.



i

	TDAL	PoH	CoreSet	OAE	UCB	TopUnc	DiscoBAX	EI	BADGE	TS	KMeans	Random
IFN-γ	1	3	2	7	6	5	4	9	8	10	11	12
IL-2	1	2	8	3	4	7	9	5	10	6	11	12
NK	2	4	1	7	10	9	12	8	5	6	11	3
Carnevale	1	4	2	7	8	6	3	9	5	11	10	12
Simpson K562	2	1	8	4	3	7	—	6	10	5	11	9
SCHEMATIC	3	1	2	9	8	7	—	6	4	10	5	11
Kuzmin	1	2	6	5	4	3	—	8	9	7	10	11
Horlbeck Jurkat	2	1	8	3	4	6	—	5	9	7	11	10
Mean rank	**1.63**	2.25	4.63	5.63	5.88	6.25	7.00	7.00	7.50	7.75	10.00	10.00

Rank

Figure 3: **Discovery performance across tasks. a–h,** Mean cumulative hits and recall with pointwise 95% confidence intervals across five seeds. **i,** Ranks by mean final hits (darker is higher).

5.2 Budgeted Perturbation Discovery

Single perturbations. Across five seeds, TDAL achieves the highest mean final discovery counts on IFN-γ, IL-2, and Carnevale (Figure 3a–d). Final cumulative hit counts are 270.6 ± 6.1, 367.4 ± 3.0, and 161.2 ± 8.6, corresponding to recalls of 31.00%, 42.08%, and 18.40%, respectively. On NK, CoreSet, TDAL, and Random reach recalls of 13.01%, 12.49%, and 11.71%, respectively, with

small gaps between active methods and Random. TDAL shows strong performance and adaptability across single gene perturbation landscapes.

Combinatorial perturbations. TDAL ranks first on Kuzmin and third on SCHEMATIC, with final recalls of 38.59% and 29.01%, respectively (Figure 3e–h). Although ranked second on Simpson K562 and Horlbeck Jurkat, TDAL approaches complete hit recovery with recalls of 94.40% and 99.45%, respectively, within 0.2 percentage points of the best method. These results show consistently strong performance across double and triple gene perturbation tasks.

Stability across tasks. Fixed acquisition functions favor different response landscapes (Mehrjou et al., 2022). Among eleven methods evaluated on all eight main tasks, TDAL achieves an average rank of 1.63, placing first on four tasks, second on three, and third on one (Figure 3i). TDAL's top three performance on every task supports its adaptability across response landscapes (Appendix D).

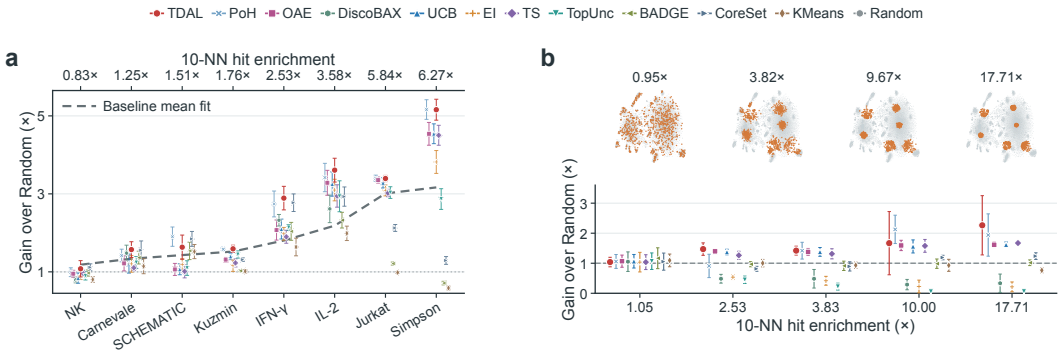


Figure 4: **Local hit structure and discovery gains. a,** Real tasks. **b,** Noisy multicluster landscapes, with example hit locations. Axes show mean enrichment and discovery gains over Random. Points show mean gains with 95% confidence intervals across five repetitions.

5.3 Landscape Structure and Active Discovery

Real landscapes. Active learning gains over Random vary across datasets, with several methods underperforming Random on NK (Figure 3). We examine local hit structure in the fixed candidate representations. Within an analysis sample, enrichment is the mean hit rate among each hit candidate's 10 nearest neighbors divided by the sample's overall hit rate. Across eight tasks, enrichment ranges from $0.83\times$ to $6.27\times$. The gray dashed fit to mean gains across nine shared baselines (Figure 4a) indicates that stronger local hit clustering is associated with greater active learning gains.

Controlled landscapes. To further examine this relationship, we construct simulated landscapes with controlled spatial structure. We keep the positions of 17,660 candidates and the set of phenotype values fixed, changing their assignment so that hits become progressively more clustered. With nearly random hit locations, methods perform close to Random. Across five enrichment levels, stronger clustering yields larger gains for TDAL and several baselines, although the benefit varies across acquisition functions (Figure 4b). These results indicate that local hit structure can provide useful information for active selection. Construction details and results are in Appendix E.

5.4 Component Ablations

Dynamic acquisition function. Surrogate only removes f_θ and selects solely by $\hat{p}_t(x)$, updating the surrogate model while keeping the acquisition function fixed (Appendix C.3). Its mean final discovery counts are lower than full TDAL on all four tasks, supporting learning the acquisition function alongside surrogate updates. w/o successor retains f_θ but sets $w_t = 0$, fitting the residual $R_t - \widehat{R}_t(A_t)$ between observed and surrogate estimated discoveries. Its mean final discovery count exceeds Surrogate only on all four tasks, showing that current feedback alone can train a dynamic acquisition function. Full TDAL adds further gains on every task, indicating that incorporating estimated future discovery value improves selection beyond learning from current feedback alone.

Training and scoring design. Fixing $w_t = 1$ lowers mean final discovery counts below both full TDAL and w/o successor on all four tasks. Undiscounted successor value can therefore impair

selection, supporting adjustment of its weight to experimental feedback. Removing LBO also lowers mean final discoveries on every task, supporting the use of additional reconstructed transitions to train Q_θ when real experimental feedback is sparse. w/o anchor removes the explicit $\hat{p}_t(x)$ from the score while retaining surrogate predictions in the inputs. Its final mean decreases on the two 60-round single perturbation tasks but increases slightly on the two 200-round tasks. With more rounds of online training, the acquisition function maintains performance without the explicit probability baseline. Detailed ablation settings are provided in Appendix F.

Dataset	Surrogate only	w/o adaptive w_t	w/o LBO	w/o successor	w/o anchor	Full TDAL
IFN-γ	186.4 ± 27.7	191.8 ± 41.1	222.2 ± 79.8	260.0 ± 11.7	$\underline{262.2 \pm 15.3}$	$\mathbf{270.6 \pm 6.1}$
IL-2	253.8 ± 15.2	319.6 ± 16.8	352.2 ± 11.5	$\underline{361.4 \pm 3.4}$	342.8 ± 43.2	$\mathbf{367.4 \pm 3.0}$
Kuzmin	3277.8 ± 35.5	3724.6 ± 142.4	4096.4 ± 208.1	4086.8 ± 89.3	$\mathbf{4174.4 \pm 121.4}$	$\underline{4141.0 \pm 194.0}$
Jurkat	4518.2 ± 13.3	4301.2 ± 248.3	4510.4 ± 39.3	4536.0 ± 15.3	$\mathbf{4550.2 \pm 3.8}$	$\underline{4542.8 \pm 4.8}$

Table 1: **Core ablations on four tasks.** Final cumulative hits across five seeds (mean $\pm$ SD). Bold and underlined values mark the highest and second-highest means.

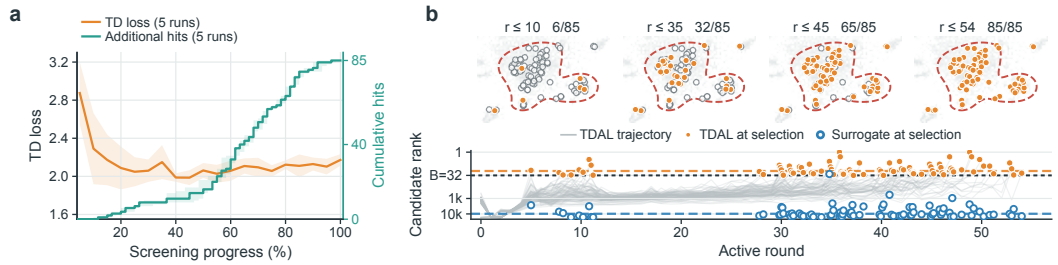


Figure 5: **Online learning and candidate reranking. a,** Training loss and cumulative discoveries over screening rounds. **b,** Hit discovery in a t-SNE projection of representations (top) and ranking trajectories (bottom), with a comparison between TDAL and surrogate prediction ranks.

5.5 Online Feedback and Selection Paths

To examine how online learning changes candidate selection, we analyze 85 IL-2 hits discovered in all five TDAL runs but in none of the Surrogate only runs. Discoveries concentrate in the second half of screening, with the median cumulative count rising from 16 after 30 rounds to 85 at completion (Figure 5a). Candidate ranks under TDAL progressively improve before selection (Figure 5b). Across the 85 candidates, their mean ranks at selection are 16.6 under TDAL and 9,780 under concurrent surrogate predictions, showing how accumulated feedback helps prioritize hits ranked poorly by the surrogate alone (Appendix Figure F1).

6 Conclusion

We introduce TDAL to learn a dynamic acquisition function online from a single perturbation screening trajectory. Building on surrogate-estimated hit probabilities, TDAL uses temporal difference learning to incorporate observed discoveries and feedback-weighted successor value, while LBO reconstructs additional training transitions from measured outcomes. By learning how to select experiments from feedback accumulated within the ongoing screen, TDAL adapts candidate selection to both immediate discoveries and subsequent discovery opportunities under limited experimental feedback. Across eight public screens spanning single, double, and triple perturbations, TDAL achieves an average rank of 1.63 among eleven methods, demonstrating consistent discovery performance across diverse response landscapes.

Limitations. Our evaluation is retrospective, and the additive acquisition function does not explicitly model interactions among candidates within a batch. Feedback weighting also relies on surrogate-estimated hit probabilities, motivating further investigation of its robustness to prediction errors. Future work will explore batch-aware selection and alternative feedback weighting strategies, alongside prospective validation in closed-loop biological experiments.

Reproducibility statement

The TDAL implementation, task configurations, and data preparation scripts are available at https://anonymous.4open.science/r/TDAL/, with data sources and running instructions. Appendices A–C provide implementation details, theoretical assumptions and proofs, and experimental settings, respectively.

AI use statement

The research ideas and methodological design originated from the authors. Generative AI tools were used to assist with code development, checking mathematical derivations and proofs, and improving manuscript clarity and grammar. The authors reviewed and verified all AI-assisted content and take full responsibility for the final work.

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

Paul Bertin, Jarrid Rector-Brooks, Deepak Sharma, Thomas Gaudelet, Andrew Anighoro, Torsten Gross, Francisco Martínez-Peña, Eileen L Tang, M. S. Suraj, Cristian Regep, et al. RECOVER identifies synergistic drug combinations in vitro through sequential model optimization. *Cell Reports Methods*, 3(10):100599, 2023. doi: 10.1016/j.crmeth.2023.100599. URL https://doi.org/10.1016/j.crmeth.2023.100599.

Julia Carnevale, Eric Shifrut, Nupura Kale, William A Nyberg, Franziska Blaeschke, Yan Yi Chen, Zhongmei Li, Sagar P Bapat, Morgan E Diolaiti, Patrick O'Leary, et al. RASA2 ablation in T cells boosts antigen sensitivity and long-term function. *Nature*, 609(7925):174–182, 2022. doi: 10.1038/s41586-022-05126-w. URL https://doi.org/10.1038/s41586-022-05126-w.

Yicong Chen, Jiahua Rao, Jiancong Xie, Dahao Xu, Zhen Wang, and Yuedong Yang. Reinforced active learning for large-scale virtual screening with learnable policy model. In *Advances in Neural Information Processing Systems*, volume 38, pp. 60831–60856, 2025. doi: 10.52202/085713-2034. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/57ac7ccded0eaaec4ad6f05612289b79-Paper-Conference.pdf.

Hyunghoon Cho, Bonnie Berger, and Jian Peng. Compact integration of multi-network topology for functional analysis of genes. *Cell Systems*, 3(6):540–548.e5, 2016. doi: 10.1016/j.cels.2016.10.017. URL https://doi.org/10.1016/j.cels.2016.10.017.

Benjamin DeMeo, Charlotte Nesbitt, Samuel A Miller, Daniel B Burkhardt, Inna Lipchina, Doris Fu, Peter Holderrieth, David Kim, Sergey Kolchenko, Artur Szalata, et al. Active learning framework leveraging transcriptomics identifies modulators of disease phenotypes. *Science*, 390(6776):eadi8577, 2025. doi: 10.1126/science.adi8577. URL https://doi.org/10.1126/science.adi8577.

Samson H Fong, Brent M Kuenzi, Nicole M Mattson, John Lee, Kyle Sanchez, Ana Bojorquez-Gomez, Kyle Ford, Brenton P Munson, Katherine Licon, Sarah Bergendahl, et al. A multilineage screen identifies actionable synthetic lethal interactions in human cancers. *Nature Genetics*, 57(1):154–164, 2025. doi: 10.1038/s41588-024-01971-9. URL https://doi.org/10.1038/s41588-024-01971-9.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pp. 1050–1059. PMLR, 2016. URL https://proceedings.mlr.press/v48/gal16.html.

Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff G. Schneider, and Richard P. Mann. Bayesian optimal active search and surveying. In *ICML*, 2012. URL http://icml.cc/2012/papers/618.pdf.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437.

Han Guo, Youwei Liang, Xingyi Cheng, Caleb Ellington, Pengtao Xie, Le Song, and Eric Xing. Using AI to build AI: AIDO.Builder enables autonomous machine learning model building for biomedicine. *bioRxiv*, 2026. doi: 10.64898/2026.04.20.719735. URL https://www.biorxiv.org/content/10.64898/2026.04.20.719735v2.

Rushil Gupta, Jason Hartford, and Bang Liu. LLMs for Bayesian optimization in scientific domains: Are we there yet? In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 15482–15510, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-emnlp.838. URL https://aclanthology.org/2025.findings-emnlp.838/.

Minsheng Hao, Yongju Lee, Hanchen Wang, Gabriele Scalia, and Aviv Regev. PerTurboAgent: An LLM-based agent for designing iterative Perturb-Seq experiments. In *MLCB*, pp. 44–64, 2025. URL https://proceedings.mlr.press/v311/hao25b.html.

E. Hellinger. Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271, 1909. doi: 10.1515/crll.1909.136.210. URL https://doi.org/10.1515/crll.1909.136.210.

Max A Horlbeck, Albert Xu, Min Wang, Neal K Bennett, Chong Y Park, Derek Bogdanoff, Britt Adamson, Eric D Chow, Martin Kampmann, Tim R Peterson, et al. Mapping the genetic landscape of human cells. *Cell*, 174(4):953–967.e22, 2018. doi: 10.1016/j.cell.2018.06.010. URL https://doi.org/10.1016/j.cell.2018.06.010.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011. URL https://arxiv.org/abs/1112.5745.

Kexin Huang, Romain Lopez, Jan-Christian Hütter, Takamasa Kudo, Antonio Rios, and Aviv Regev. Sequential optimal experimental design of perturbation screens guided by multi-modal priors. In *International Conference on Research in Computational Molecular Biology*, pp. 17–37. Springer, 2024. doi: 10.1007/978-1-0716-3989-4_2. URL https://doi.org/10.1007/978-1-0716-3989-4_2.

Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *ICML*, pp. 1714–1723, 2017. URL http://proceedings.mlr.press/v70/jiang17d.html.

Shali Jiang, Gustavo Malkomes, Matthew Abbott, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic batch active search. In *NeurIPS*, pp. 1107–1117, 2018. URL http://papers.nips.cc/paper/7387-efficient-nonmyopic-batch-active-search.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998. doi: 10.1023/A:1008306431147. URL https://doi.org/10.1023/A:1008306431147.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/8ca8da41fe1ebc8d3ca31dc14f5fc56c-Abstract.html.

Elena Kuzmin, Benjamin VanderSluis, Alex N Nguyen Ba, Wen Wang, Elizabeth N Koch, Matej Usaj, Anton Khmelinskii, Mojca Mattiazzi Usaj, Jolanda Van Leeuwen, Oren Kraus, et al. Exploring whole-genome duplicate gene retention with complex genetic interaction analysis. *Science*, 368(6498):eaaz5667, 2020. doi: 10.1126/science.aaz5667. URL https://doi.org/10.1126/science.aaz5667.

Yanke Li, Tianyu Cui, Tommaso Mansi, Mangal Prakash, and Rui Liao. BioBO: Biology-informed Bayesian optimization for perturbation design. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=CF3kJrAwmV.

Zhuo Li, Weiran Wu, Chuangyi Han, Yan Cui, Tian Lu, Rongqin Ke, Jian Sun, and Zhiyuan Yuan. SOFisher: reinforcement learning-guided experiment designs for spatial omics. *Nature Communications*, 17:6830, 2026b. doi: 10.1038/s41467-026-73404-6. URL https://doi.org/10.1038/s41467-026-73404-6.

Clare Lyle, Arash Mehrjou, Pascal Notin, Andrew Jesson, Stefan Bauer, Yarin Gal, and Patrick Schwab. DiscoBAX: Discovery of optimal intervention sets in genomic experiment design. In *ICML*, pp. 23170–23189, 2023. URL https://proceedings.mlr.press/v202/lyle23a.html.

Arash Mehrjou, Ashkan Soleymani, Andrew Jesson, Pascal Notin, Yarin Gal, Stefan Bauer, and Patrick Schwab. GeneDisco: A benchmark for experimental design in drug discovery. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=-w2oomO6qgc.

Robin M Meyers, Jordan G Bryan, James M McFarland, Barbara A Weir, Ann E Sizemore, Han Xu, Neekesh V Dharia, Phillip G Montgomery, Glenn S Cowley, Sasha Pantel, et al. Computational correction of copy number effect improves specificity of CRISPR–Cas9 essentiality screens in cancer cells. *Nature Genetics*, 49(12):1779–1784, 2017. doi: 10.1038/ng.3984. URL https://doi.org/10.1038/ng.3984.

Quan Nguyen, Anindya Sarkar, and Roman Garnett. Amortized nonmyopic active search via deep imitation learning. *arXiv preprint arXiv:2405.15031*, 2024. URL https://arxiv.org/abs/2405.15031.

Aldo Pacchiano, Drausin Wulsin, Robert A Barton, and Luis Voloch. Neural design for genetic perturbation experiments. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=TUBpc5rqGA.

Jason Qin, Hans-Hermann Wessels, Carlos Fernandez-Granda, and Yuhan Hao. Active learning for efficient discovery of optimal combinatorial perturbations. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=fGbGF0kl5L.

Yusuf H Roohani, Andrew H. Lee, Qian Huang, Jian Vora, Zachary Steinhart, Kexin Huang, Alexander Marson, Percy Liang, and Jure Leskovec. BioDiscoveryAgent: An AI agent for designing genetic perturbation experiments. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HAwZGLcye3.

Andrea Rubbi, Arpit Merchant, Samuel Ogden, Amir Akbarnejad, Pietro Lio, Sattar Vakili, and Mohammad Lotfollahi. Many needles in a haystack: Active hit discovery for perturbation experiments. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=iZN85pjpsS.

Carlos G Sanchez, Christopher M Acker, Audrey Gray, Malini Varadarajan, Cheng Song, Nadire R Cochran, Steven Paula, Alicia Lindeman, Shaojian An, Gregory McAllister, et al. Genome-wide CRISPR screen identifies protein pathways modulating tau protein levels in neurons. *Communications Biology*, 4(1):736, 2021. doi: 10.1038/s42003-021-02272-1. URL https://doi.org/10.1038/s42003-021-02272-1.

Igal Sason and Sergio Verdú. f-divergence inequalities. *IEEE Transactions on Information Theory*, 62(11):5973–6006, 2016. doi: 10.1109/TIT.2016.2603151.

Ralf Schmidt, Zachary Steinhart, Madeline Layeghi, Jacob W Freimer, Raymund Bueno, Vinh Q Nguyen, Franziska Blaeschke, Chun Jimmie Ye, and Alexander Marson. CRISPR activation and interference screens decode stimulation responses in primary human T cells. *Science*, 375 (6580):eabj4008, 2022. doi: 10.1126/science.abj4008. URL https://doi.org/10.1126/science.abj4008.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

Danny Simpson, Jia Ling, Yangwode Jing, and Britt Adamson. Mapping the genetic interaction network of PARP inhibitor response. *bioRxiv*, 2023. doi: 10.1101/2023.08.19.553986. URL https://doi.org/10.1101/2023.08.19.553986.

Braden Soper, Michal Lisicki, Mary Silva, Jose Cadena, Haonan Zhu, Shivshankar Sundaram, Priyadip Ray, and Jeff Drocco. Knowledge graph-aided Bayesian active learning for top-K genetic interaction discovery. *Scientific Reports*, 15(1):31196, 2025. doi: 10.1038/s41598-025-13972-7. URL https://doi.org/10.1038/s41598-025-13972-7.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In Johannes Fürnkranz and Thorsten Joachims (eds.), *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 1015–1022. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/422.pdf.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. doi: 10.1093/biomet/25.3-4.285. URL https://doi.org/10.1093/biomet/25.3-4.285.

Christopher Tosh, Mauricio Tec, Jessica B White, Jeffrey F Quinn, Glorymar Ibanez Sanchez, Paul Calder, Andrew L Kung, Filemon S Dela Cruz, and Wesley Tansey. A Bayesian active learning platform for scalable combination drug screens. *Nature Communications*, 16 (1):156, 2025. doi: 10.1038/s41467-024-55287-7. URL https://doi.org/10.1038/s41467-024-55287-7.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. doi: 10.1609/aaai.v30i1.10295. URL https://doi.org/10.1609/aaai.v30i1.10295.

Michael Volpp, Lukas P. Fröhlich, Kirsten Fischer, Andreas Doerr, Stefan Falkner, Frank Hutter, and Christian Daniel. Meta-learning acquisition functions for transfer learning in Bayesian optimization. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryeYpJSKwr.

Gilles Wainrib, Barbara Bodinier, Haithem Dakhli, Josep Monserrat, Almudena Espin-Perez, Roberta Codato, and John Klein. Can AI scientist agents learn from lab-in-the-loop feedback? evidence from iterative perturbation discovery. In *The 2026 Workshop on Generative and Agentic AI for Biology*, 2026. URL https://openreview.net/forum?id=sZijNyjuCF.

Yue Xu, Haotian Cui, Kuan Pang, Gen Li, Fanglin Gong, Songtao Dong, Bo Wang, and Bowen Li. LUMI-lab: a foundation model-driven autonomous platform enabling discovery of ionizable lipid designs for mRNA delivery. *Cell*, 189(6):1620–1635.e25, 2026. doi: 10.1016/j.cell.2026.01.012. URL https://doi.org/10.1016/j.cell.2026.01.012.

Ziping Xu, Eunjae Shim, Ambuj Tewari, and Paul Zimmerman. Adaptive sampling for discovery. In *Advances in Neural Information Processing Systems*, volume 35, pp. 1114–1126, 2022. doi: 10.52202/068431-0082. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/07bc8125400bf4b140c332010756bd9b-Abstract-Conference.html.

Hanwen Zhang, Wencheng Liu, Wenhao Zheng, Ting Kang, Shilong Hu, Zhuang Miao, Jin Liu, Bowen Ke, Xianggen Liu, and Jiancheng Lv. Policy-based active learning for efficient molecular identification. *Journal of Chemical Information and Modeling*, 66(10):5851–5868, 2026. doi: 10.1021/acs.jcim.6c00474. URL https://doi.org/10.1021/acs.jcim.6c00474.

Haonan Zhu, Mary Silva, Jose Cadena, Braden Soper, Michał Lisicki, Braian Peetoom, Sergio E Baranzini, Shivshankar Sundaram, Priyadip Ray, and Jeff Drocco. Deep active learning-based experimental design to uncover synergistic genetic interactions for host-targeted therapeutics. *Bioinformatics Advances*, 6(1):vbaf228, 2026. doi: 10.1093/

bioadv/vbaf228. URL https://academic.oup.com/bioinformaticsadvances/article/6/1/vbaf228/8262951.

Xiaoxuan Zhuang, Daniel P Veltri, and Eric O Long. Genome-wide CRISPR screen reveals cancer cell resistance to NK cells induced by NK-derived IFN-γ. *Frontiers in Immunology*, 10: 2879, 2019. doi: 10.3389/fimmu.2019.02879. URL https://doi.org/10.3389/fimmu.2019.02879.

APPENDIX

A ALGORITHM AND IMPLEMENTATION DETAILS

A.1 COMPARISON WITH RELATED METHODS

Table A1 compares representative methods for perturbation screening, molecular virtual screening, and spatial omics sampling across five mechanisms including surrogate updates, biological knowledge, online acquisition updates, value bootstrapping, and experience augmentation.

Table A1: Comparison of active screening and experimental design methods. ✓/× indicate whether a method uses each mechanism.

Method	Research task	Surrogate updates	Biological knowledge	Online acquisition updates	Value bootstrapping	Experience augmentation
GeneDisco surrogate baselines	Perturbation screening	✓	×	×	×	×
OAE	Perturbation screening	✓	×	×	×	×
DiscoBAX	Perturbation screening	✓	×	×	×	×
PoH	Perturbation screening	✓	×	×	×	×
BioBO	Perturbation screening	✓	✓	×	×	×
BioDiscoveryAgent	Perturbation screening	×	✓	×	×	×
PerTurboAgent	Perturbation screening	✓	✓	×	×	×
GLARE	Molecular virtual screening	✓	×	✓	×	×
SOFisher	Spatial omics sampling	×	×	×	✓	×
TDAL	Perturbation screening	✓	×	✓	✓	✓

Surrogate updates use newly measured outcomes to fit a model that predicts unmeasured phenotypes or activities. Biological knowledge guides selection through pathway information, literature, or knowledge encoded in language models. BioBO and PerTurboAgent combine such knowledge with response modeling, whereas BioDiscoveryAgent incorporates experimental observations into its language model context without fitting a response surrogate. SOFisher uses observed spatial features as inputs to its sampling policy rather than fitting a phenotype prediction model.

Online acquisition updates use feedback from the current task to update the acquisition function's parameters. Value bootstrapping constructs training targets from current rewards and estimated successor values. GLARE updates its selection policy during screening but does not explicitly bootstrap a successor value. SOFisher trains its sampling policy through value bootstrapping before application and uses fixed policy parameters on new tissue sections, while incorporating newly sampled observations into the state. TDAL performs both acquisition updates and successor value bootstrapping within the current screening task.

Experience augmentation reconstructs additional transitions from measured outcomes to train the acquisition function. In budgeted experimental design without replacement, the measured dataset

grows while the unmeasured pool and remaining budget shrink. A screen therefore provides a single executed trajectory and cannot return to an earlier state to test alternative experimental sequences within the same budget. LBO reconstructs predecessor states in which a measured batch has not yet been observed, then restores that batch's real outcomes to form additional transitions to a known successor state. The same experimental outcomes can thus support batch value learning under different information states, expanding online training experience without additional experiments.

A.2 NOTATION

Table A2 summarizes the main notation used in the problem formulation and TDAL.

Table A2: Key notation for the screening objective and TDAL.

Symbol	Meaning
$\mathcal{X}$	Fixed pool of candidate perturbations.
$e(x)$	Phenotype of candidate x after aligning the effect direction.
τ	Threshold defining the desired phenotypic response.
$z(x)$	Indicator that candidate x meets the hit threshold.
D_t	Measured data available before selection in round t.
U_t	Unmeasured candidates available in round t.
H_t	History of observations and selections before round t.
A_t	Batch selected for experimentation in round t.
B	Number of candidates queried per round.
T	Number of active screening rounds.
R_t	Observed number of discoveries in batch A_t.
C_T	Final verified discovery count, including initial discoveries.
π	Policy that selects each batch from the available history.
$J_w(\pi)$	Expected cumulative discoveries weighted by experimental feedback.
$Q_t^{*,w}(H_t, A)$	Optimal weighted value of selecting batch A given history H_t.
w_t	Successor value weight determined by prediction changes.
g_β	Surrogate model predicting perturbation phenotypes.
$\hat{p}_t(x)$	Estimated hit probability of an unmeasured candidate.
$\widehat{R}_t(A)$	Estimated immediate discovery count from surrogate hit probabilities.
$\phi(x)$	Fixed biological representation of candidate x.
s_t	Global summary of screening progress, observations, and predictions.
$c_t(x)$	Candidate features combining biological representation and predictions.
f_θ	Acquisition network mapping the state and candidate features to an output.
$q_\theta(s_t, x)$	Dynamic acquisition function scoring candidate x.
$Q_\theta(s_t, A)$	Batch value estimate obtained by summing candidate scores.
ξ_t	Transition generated by an experimental round.
$\widetilde{\xi}$	Virtual transition reconstructed by LBO from measured outcomes.
Y_i	TD target for transition i.

A.3 COMPLETE TDAL ALGORITHM

Algorithm 1 summarizes TDAL from initialization through the end of screening. Starting from the initial observations D_0, TDAL fits a surrogate model, estimates hit probabilities for candidates in the initial unmeasured pool U_0, and forms the initial selection state. It then constructs initial LBO transitions from D_0 to initialize the acquisition network through TD training before the first query.

At each round, the acquisition function scores candidates in U_t, and TDAL queries the B candidates with the highest scores. The observed outcomes are added to D_t, and the queried candidates are removed from U_t. TDAL refits the surrogate model on the updated data. Before the next round, it updates predictions over U_{t+1} to construct the successor state. Comparing hit probabilities before and after the experiment over this pool gives the Hellinger weight. The selected batch, observed

discovery count, feedback weight, and current and successor states form a real transition. Real and online LBO transitions enter the replay buffer, from which TD updates train the acquisition network for subsequent selection. After T rounds, TDAL returns all verified hit candidates.

Algorithm 1 TDAL

Input Candidate pool $\mathcal{X}$ with representations $\phi(x)$, initial data D_0, rounds T, batch size B, threshold τ, and LBO transition limits.

Initialize $U_0 \leftarrow \mathcal{X} \setminus \{x : (x, e(x)) \in D_0\}$.
Initialize acquisition network f_θ with a zero output layer.
Initialize target network $f_{\bar{\theta}}$ with $\bar{\theta} \leftarrow \theta$.
Initialize replay buffer $\mathcal{R} \leftarrow \varnothing$.
Fit surrogate g_{β_0} to D_0 with validation early stopping.
Estimate $\widehat{\mu}_0(x)$, $\widehat{\sigma}_0(x)$, and $\hat{p}_0(x)$ using 50 MC dropout samples for $x \in U_0$.
Construct s_0 and candidate features $\{c_0(x) : x \in U_0\}$.
if $|D_0| > B$ **then**
    Add initial LBO transitions to $\mathcal{R}$ using Algorithm 2.
    for 20 initial TD updates **do**
        Update f_θ using the TD loss on a replay minibatch.
        Soft update the target parameters $\bar{\theta}$.
        Refresh the sampled transitions' replay priorities.
    end for
end if
for $t = 0, \ldots, T-1$ **do**
    Compute $q_\theta(s_t, x) = \hat{p}_t(x) + f_\theta(s_t, c_t(x))$ for $x \in U_t$.
    Select the B highest scoring candidates as A_t.
    Experimentally query A_t to obtain $O_t = \{(x, e(x)) : x \in A_t\}$.
    Compute the discovery count $R_t = \sum_{x \in A_t} z(x)$.
    Update measured data $D_{t+1} \leftarrow D_t \cup O_t$.
    Update the unmeasured pool $U_{t+1} \leftarrow U_t \setminus A_t$.
    Refit $g_{\beta_{t+1}}$ to D_{t+1} with validation early stopping.
    if $t < T-1$ **then**
        Estimate $\widehat{\mu}_{t+1}(x)$, $\widehat{\sigma}_{t+1}(x)$, and $\hat{p}_{t+1}(x)$ by MC dropout for $x \in U_{t+1}$.
        Construct s_{t+1} and candidate features $\{c_{t+1}(x) : x \in U_{t+1}\}$.
        Compute Hellinger weight w_t from $\hat{p}_t(x)$ and $\hat{p}_{t+1}(x)$ over U_{t+1}.
    else
        Set the terminal successor state s_T.
        Set $w_t \leftarrow 0$.
    end if
    Add $\xi_t = (s_t, A_t, R_t, w_t, s_{t+1})$ and its prediction inputs to $\mathcal{R}$.
    Add online LBO transitions using Algorithm 2.
    for 20 online TD updates **do**
        Update f_θ using the TD loss on a replay minibatch.
        Soft update the target parameters $\bar{\theta}$.
        Refresh the sampled transitions' replay priorities.
    end for
end for
return verified hit set $\{x : (x, e(x)) \in D_T,\ z(x) = 1\}$.

A.4 Candidate Representations

The mapping $\phi(x)$ encodes candidate perturbation x as a vector of fixed dimension, providing gene functional information to the surrogate and acquisition network.

Human tasks use genome-wide CRISPR–Cas9 knockout screening data from the Broad Institute's Project Achilles. The CERES gene dependency profiles in DepMap 20Q4 provide 808-dimensional gene representations. Each dimension corresponds to a cancer cell line and describes the effect of knocking out the gene on cell growth. CERES corrects for copy number effects (Meyers et al., 2017). The Kuzmin yeast task uses the 500-dimensional STRING network embeddings released

with Mashup (Cho et al., 2016). These base gene representations remain fixed during screening. For a single gene perturbation, $\phi(x)$ is the corresponding gene embedding.

For a candidate x containing $m \in \{2, 3\}$ genes with embeddings $\mathbf{v}_1, \ldots, \mathbf{v}_m$, we define

$$\phi(x) = \left[\frac{1}{m} \sum_{i=1}^{m} \mathbf{v}_i,\ \frac{2}{m(m-1)} \sum_{i<j} |\mathbf{v}_i - \mathbf{v}_j|,\ \frac{2}{m(m-1)} \sum_{i<j} \mathbf{v}_i \odot \mathbf{v}_j \right]. \tag{13}$$

The brackets concatenate the mean embedding, mean elementwise absolute difference, and mean elementwise product, with $\odot$ denoting elementwise multiplication. Averaging over all members and unordered pairs makes this construction invariant to gene order.

A.5 Surrogate Model and Hit Probability Estimation

Model and training. The surrogate takes a fixed vector representation (embedding) $\phi(x)$ of candidate perturbation x as input and predicts the phenotype $e(x)$ after aligning the effect direction. Its two hidden layers have widths 128 and 64. Each applies a linear transformation followed by ReLU and dropout. A final linear layer produces a scalar output. The model is

$$\begin{aligned} \mathbf{h}_{\text{sur}}^{(1)}(x) &= \text{Dropout}_{0.2}(\text{ReLU}(W_1^{\text{sur}} \phi(x) + b_1^{\text{sur}})) \in \mathbb{R}^{128}, \\ \mathbf{h}_{\text{sur}}^{(2)}(x) &= \text{Dropout}_{0.2}\Big(\text{ReLU}(W_2^{\text{sur}} \mathbf{h}_{\text{sur}}^{(1)}(x) + b_2^{\text{sur}})\Big) \in \mathbb{R}^{64}, \\ g_\beta(\phi(x)) &= W_{\text{out}}^{\text{sur}} \mathbf{h}_{\text{sur}}^{(2)}(x) + b_{\text{out}}^{\text{sur}} \in \mathbb{R}. \end{aligned} \tag{14}$$

Here, $\text{Dropout}_{0.2}$ denotes dropout with probability 0.2, and β contains all weights and biases of the hidden and output layers. Predictive uncertainty is estimated through MC dropout (Gal & Ghahramani, 2016).

At each round, the measured data D_t are split into a training set D_t^{tr} and a validation set D_t^{val} for early stopping, and the model parameters are reinitialized. The surrogate fits measured phenotypes by minimizing the mean squared error

$$\mathcal{L}_{\text{sur}}(\beta) = \frac{1}{|D_t^{\text{tr}}|} \sum_{(x, e(x)) \in D_t^{\text{tr}}} \big(g_\beta(\phi(x)) - e(x)\big)^2. \tag{15}$$

Dropout is active during training and disabled when evaluating validation mean squared error after each epoch. Let $\beta_t^{(j)}$ denote the parameters after epoch j, and let J_t be the number of epochs completed in the current round. We select the final model by minimizing validation error,

$$\begin{aligned} j_t^* &= \arg\min_{1 \le j \le J_t} \frac{1}{|D_t^{\text{val}}|} \sum_{(x, e(x)) \in D_t^{\text{val}}} \big(g_{\beta_t^{(j)}}(\phi(x)) - e(x)\big)^2, \\ \beta_t &= \beta_t^{(j_t^*)}. \end{aligned} \tag{16}$$

Predictive statistics. Dropout remains active at inference. For each unmeasured candidate $x \in U_t$, we independently sample K forward passes. Denote the kth stochastic forward pass by $g_{\beta_t}^{(k)}$, giving the phenotype prediction $\widehat{e}_t^{(k)}(x) = g_{\beta_t}^{(k)}(\phi(x))$. The predictive mean, uncertainty, and hit probability are estimated by the sample mean, standard deviation, and fraction reaching the hit threshold, respectively,

$$\begin{aligned} \widehat{\mu}_t(x) &= \frac{1}{K} \sum_{k=1}^{K} \widehat{e}_t^{(k)}(x), \\ \widehat{\sigma}_t(x) &= \left[\frac{1}{K} \sum_{k=1}^{K} \big(\widehat{e}_t^{(k)}(x) - \widehat{\mu}_t(x)\big)^2 \right]^{1/2}, \\ \hat{p}_t(x) &= \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}\{\widehat{e}_t^{(k)}(x) \ge \tau\}. \end{aligned} \tag{17}$$

These statistics define candidate features and the selection state. The hit probability also provides the acquisition function's base score and is used to compute the Hellinger weight.

A.6 State Representation and Acquisition Function Parameterization

Dynamic candidate features. For each unmeasured candidate $x \in U_t$, the acquisition network receives the shared selection state s_t and candidate features $c_t(x)$. These features combine the fixed biological representation $\phi(x)$ with the surrogate model's predicted mean, uncertainty, and hit probability. We center the mean at the hit threshold and scale both the mean and uncertainty, giving

$$c_t(x) = \left[\phi(x),\ \frac{\widehat{\mu}_t(x) - \tau}{a_t},\ \log\left(1 + \frac{\widehat{\sigma}_t(x)}{a_t}\right),\ \hat{p}_t(x)\right]. \tag{18}$$

Here, $\widehat{\mu}_t(x)$ and $\widehat{\sigma}_t(x)$ are the predictive mean and standard deviation defined in Appendix A.5. The hit probability $\hat{p}_t(x)$ is included without rescaling.

The normalization denominator a_t is computed from the phenotypes already measured in D_t. It is the larger of their interquartile range (75th minus 25th percentile) and population standard deviation, with a minimum of 10^{-6} to prevent division by zero. The same denominator is used for all candidates scored at that round. Their predicted means and uncertainties remain specific to each candidate. We recompute a_t when the measured data change.

Selection state. The selection state s_t is a vector summary of the current screening history H_t that provides global information for the dynamic acquisition function. Shared by all candidates, it combines screening progress, measured outcomes, and surrogate prediction statistics over the unmeasured pool U_t. Its 15 components follow the network input order

$$\begin{aligned} s_t = \Big[&\frac{|D_t|}{M},\ \frac{M - |D_t|}{M},\ \frac{\lceil \min\{M - |D_t|, |U_t|\}/B \rceil}{\lceil M/B \rceil},\ \frac{|U_t|}{|\mathcal{X}|},\ \frac{\min\{B, M - |D_t|, |U_t|\}}{B}, \\ &\tau,\ \max_{(x,e(x))\in D_t} e(x),\ a_t,\ \mathrm{Mean}_{U_t}(\hat{p}_t),\ \mathrm{SD}_{U_t}(\hat{p}_t),\ \mathrm{P}_{90,U_t}(\hat{p}_t), \\ &\mathrm{Mean}_{U_t}\left(\frac{\widehat{\mu}_t - \tau}{a_t}\right),\ \mathrm{SD}_{U_t}\left(\frac{\widehat{\mu}_t - \tau}{a_t}\right), \\ &\mathrm{Mean}_{U_t}\left(\frac{\widehat{\sigma}_t}{a_t}\right),\ \mathrm{P}_{90,U_t}\left(\frac{\widehat{\sigma}_t}{a_t}\right)\Big] \in \mathbb{R}^{15}. \end{aligned} \tag{19}$$

Here, Mean_{U_t}, SD_{U_t}, and P_{90,U_t} denote the mean, population standard deviation, and 90th percentile over candidates $x \in U_t$. The first five components describe the measured budget fraction, remaining budget fraction, normalized remaining rounds, remaining pool fraction, and query batch size relative to B. These ratios are clipped to $[0, 1]$. The batch size ratio is one for the full batches used here.

The hit threshold τ, largest measured phenotype $\max_{(x,e(x))\in D_t} e(x)$, and normalization denominator a_t retain the phenotype scale after aligning the effect direction. Statistics of the hit probability $\hat{p}_t(x)$, normalized predicted effect $(\widehat{\mu}_t(x) - \tau)/a_t$, and scaled uncertainty $\widehat{\sigma}_t(x)/a_t$ summarize predictions over $x \in U_t$. The uncertainty statistics use the scaled standard deviation.

Acquisition network. The acquisition network f_θ is the trainable component of $q_\theta = \hat{p}_t + f_\theta$. For each unmeasured candidate $x \in U_t$, its encoding and network layers are

$$\begin{aligned} u_\theta(x) &= \mathrm{ReLU}\left(W_{\mathrm{enc}} \frac{\phi(x)}{\max\{\|\phi(x)\|_2, 10^{-8}\}} + b_{\mathrm{enc}}\right) \in \mathbb{R}^{64}, \\ \mathbf{h}_t^{(0)}(x) &= \left[s_t,\ u_\theta(x),\ \frac{\widehat{\mu}_t(x) - \tau}{a_t},\ \log\left(1 + \frac{\widehat{\sigma}_t(x)}{a_t}\right),\ \hat{p}_t(x)\right] \in \mathbb{R}^{82}, \\ \mathbf{h}_t^{(k)}(x) &= \mathrm{ReLU}\left(W_k^{\mathrm{acq}} \mathbf{h}_t^{(k-1)}(x) + b_k^{\mathrm{acq}}\right) \in \mathbb{R}^{128}, \quad k = 1, 2, \\ f_\theta(s_t, c_t(x)) &= W_{\mathrm{out}}^{\mathrm{acq}} \mathbf{h}_t^{(2)}(x) + b_{\mathrm{out}}^{\mathrm{acq}} \in \mathbb{R}. \end{aligned} \tag{20}$$

Here, $u_\theta(x)$ encodes the normalized fixed representation. The input $\mathbf{h}_t^{(0)}(x)$ concatenates the 15 state features, 64 encoding features, and three surrogate prediction features from $c_t(x)$.

The parameters θ include the weights and biases of the projection, two hidden layers, and output layer. The static embeddings remain fixed, while the projection is learned with the acquisition network. The output layer's weights and bias are initialized to zero, so that $f_\theta = 0$ and $q_\theta = \hat{p}_t$ before initial TD training.

A.7 TD Learning and Experience Replay

LBO transition construction. Initial LBO hides batches from D_0 to train the acquisition network before the first query. Online LBO draws hidden batches from candidates measured after initialization, so a batch can combine observations from different rounds. Algorithm 2 gives the construction for both stages.

The current measured dataset D^+ and unmeasured pool U^+ define the successor state s^+. Each hidden batch produces a different predecessor state, but restoring its outcomes returns to this same successor state. For a nonterminal successor, let $\hat{p}^+(x)$ denote the MC dropout hit probability estimated by the surrogate trained on D^+, for $x \in U^+$. For initial LBO, the successor dataset is $D^+ = D_0$, giving $\hat{p}^+(x) = \hat{p}_0(x)$. For online LBO after round t, it is $D^+ = D_{t+1}$, giving $\hat{p}^+(x) = \hat{p}_{t+1}(x)$.

For each hidden batch A, removing its outcomes from D^+ gives the predecessor dataset D^-, and returning its candidates to the unmeasured pool gives U^-. We fit the predecessor surrogate g_{β^-} to D^- with validation early stopping. Its MC dropout predictions give predecessor hit probabilities $\hat{p}^-(x)$ for $x \in U^-$. These predictions also define s^- and the candidate features, with the phenotype scale and screening progress recomputed from D^-. The Hellinger weight compares $\hat{p}^-(x)$ and $\hat{p}^+(x)$ over the shared remaining pool $U^+ \subseteq U^-$. The observed discoveries in A give the transition reward R. The predecessor state, hidden batch, reward, weight, and successor state then form the LBO transition.

The set $\mathcal{S}$ prevents repeated hidden batches for a given successor and excludes the current real batch during online LBO. The list $\mathcal{C}$ contains the accepted hidden batches. Initial and online transition limits L and the generation schedule are specified in Table C3.

Successor selection and TD target. For a replay transition $\xi_i = (s_i, A_i, R_i, w_i, s_i')$, the TD target combines its observed reward with an estimate of weighted future discoveries. To estimate this successor value, the current acquisition function scores every candidate in the recorded successor pool U_i' using

$$q_\theta(s_i', x) = \hat{p}_i'(x) + f_\theta(s_i', c_i'(x)), \qquad x \in U_i'. \tag{21}$$

The state s_i', candidate features $c_i'(x)$, and hit probabilities $\hat{p}_i'(x)$ are saved when the transition is constructed. Replay keeps these inputs fixed and applies the current network to them. As θ changes, the network can select a different successor batch from the same recorded pool.

Following Double DQN, we select the B candidates with the highest current scores to form $A'_{\theta,i}$ and evaluate them with the target network (van Hasselt et al., 2016). This gives

$$\begin{aligned} Q_{\bar{\theta}}(s_i', A'_{\theta,i}) &= \sum\nolimits_{x \in A'_{\theta,i}} \left[\hat{p}_i'(x) + f_{\bar{\theta}}(s_i', c_i'(x))\right], \\ Y_i &= R_i + w_i Q_{\bar{\theta}}(s_i', A'_{\theta,i}). \end{aligned} \tag{22}$$

This successor selection is used to calculate the training target and does not trigger additional experiments. For terminal transitions, $Y_i = R_i$. Real and LBO transitions use the same target, which is held constant when computing gradients.

Experience replay and parameter updates. Real and LBO transitions enter a shared prioritized replay buffer. Let n be the number of stored transitions and ρ_i the priority of transition i. Each update samples a training batch $\mathcal{B}$ without replacement using normalized sampling weights

$$P(i) = \frac{(\rho_i + \epsilon_{\mathrm{PER}})^{\alpha_{\mathrm{PER}}}}{\sum_{j=1}^{n} (\rho_j + \epsilon_{\mathrm{PER}})^{\alpha_{\mathrm{PER}}}}. \tag{23}$$

The exponent α_{PER} controls the preference for high priority transitions. Frequently sampled transitions receive smaller importance weights in the loss, reducing the imbalance introduced by prioritized sampling. We use

$$\widetilde{\kappa}_i = (nP(i))^{-\beta_{\mathrm{IS}}}, \qquad \kappa_i^{\mathrm{IS}} = \frac{\widetilde{\kappa}_i}{\max_{j \in \mathcal{B}} \widetilde{\kappa}_j}. \tag{24}$$

Algorithm 2 LBO transition construction

Input Measured data D^+, unmeasured pool U^+, initial data D_0, successor state s^+, successor estimates $\{\hat{p}^+(x) : x \in U^+\}$ when nonterminal, stage, batch size B, threshold τ, transition limit L, replay buffer $\mathcal{R}$, and current real batch A_t for online LBO.

Output Replay buffer $\mathcal{R}$ augmented with up to L LBO transitions.

if initial LBO **then**
  Set eligible candidates $E \leftarrow \{x : (x, e(x)) \in D_0\}$.
  Initialize used batches $\mathcal{S} \leftarrow \varnothing$.
else
  Set $E \leftarrow \{x : (x, e(x)) \in D^+ \setminus D_0\}$.
  Initialize $\mathcal{S} \leftarrow \{A_t\}$.
end if
Initialize the hidden batch list $\mathcal{C} \leftarrow [\,]$.
Generate seeded random permutations of E.
for each generated permutation **do**
  Partition it into complete batches of size B.
  for each batch A in this partition **do**
    if $A \notin \mathcal{S}$ **then**
      Append A to $\mathcal{C}$.
      Add A to $\mathcal{S}$.
    end if
  end for
end for
Retain at most the first L batches in $\mathcal{C}$.
for each hidden batch $A \in \mathcal{C}$ **do**
  Hide its outcomes $D^- \leftarrow D^+ \setminus \{(x, e(x)) : x \in A\}$.
  Restore its candidates to the pool $U^- \leftarrow U^+ \cup A$.
  Fit predecessor surrogate g_{β^-} to D^- with validation early stopping.
  Draw $K = 50$ MC dropout predictions from g_{β^-} for $x \in U^-$.
  Estimate $\hat{p}^-(x) \leftarrow K^{-1} \sum_{k=1}^{K} \mathbf{1}\{g_{\beta^-}^{(k)}(\phi(x)) \geq \tau\}$ for $x \in U^-$.
  Construct s^- and $\{c^-(x) : x \in U^-\}$ from D^- and its predictive statistics.
  Compute $R \leftarrow \sum_{x \in A} z(x)$ from the hidden outcomes.
  if s^+ is terminal **then**
    Set $w \leftarrow 0$.
  else
    Compute w from Hellinger distances between $\hat{p}^-(x)$ and $\hat{p}^+(x)$ for $x \in U^+$.
  end if
  Add $\widetilde{\xi} = (s^-, A, R, w, s^+)$ and its prediction inputs to $\mathcal{R}$.
end for
return $\mathcal{R}$.

Here, β_{IS} controls the strength of the weighting, and normalization sets the largest weight within $\mathcal{B}$ to one. With TD error $\delta_i = Q_\theta(s_i, A_i) - Y_i$, the training loss is

$$\widehat{\mathcal{L}}_{\mathrm{TD}}(\theta) = \frac{1}{|\mathcal{B}|} \sum\nolimits_{i \in \mathcal{B}} \kappa_i^{\mathrm{IS}} \, \mathrm{Huber}_1(\delta_i). \tag{25}$$

The Huber loss is quadratic for absolute errors up to 1 and linear above 1. Gradients update only θ, with surrogate predictions held fixed. After each update, the sampled transitions receive priorities $\rho_i \leftarrow |\delta_i| + \epsilon_{\mathrm{PER}}$, and the target parameters follow

$$\bar{\theta} \leftarrow (1 - \tau_{\mathrm{target}})\bar{\theta} + \tau_{\mathrm{target}}\theta. \tag{26}$$

The optimizer and training settings are specified in Table C3.

B Theoretical Analysis

We prove Propositions 1–3, establishing weighted Bellman optimality, relating the Hellinger weight to predictive improvement, and bounding the gap in final discoveries.

B.1 Weighted Bellman Optimality

We fix the surrogate training and prediction procedures and condition on their randomness. Given the history H_t and query batch A_t, the observed outcomes determine the discovery count R_t and feedback weight $w_t = w(H_t, A_t, H_{t+1}) \in [0, 1]$. Both quantities are therefore determined by the successor history H_{t+1}. At history H, the feasible batch set is

$$\mathcal{A}_t(H) = \{A \subseteq U_t(H) : |A| = B\}. \tag{27}$$

For any selection policy π, define the weighted discovery count from round t to the end of screening as

$$G_t^w = \sum\nolimits_{k=t}^{T-1} \left(\prod\nolimits_{j=t}^{k-1} w_j \right) R_k, \qquad G_T^w = 0. \tag{28}$$

The current count R_t has weight one, while each subsequent count is weighted by the product of the intervening feedback weights. The corresponding policy value is

$$V_t^{w,\pi}(H) = \mathbb{E}_\pi[G_t^w \mid H_t = H], \tag{29}$$

the expected weighted number of remaining discoveries when selection follows π from the current screening history H. The terminal value is zero because no experiments remain.

Separating the current round from subsequent rounds gives

$$G_t^w = R_t + w_t G_{t+1}^w. \tag{30}$$

Given H_{t+1}, both R_t and w_t are known, and the conditional expectation of the remaining return is $V_{t+1}^{w,\pi}(H_{t+1})$. Averaging over the current batch and its observations, we obtain the following by the law of iterated expectations:

$$V_t^{w,\pi}(H) = \sum\nolimits_{A \in \mathcal{A}_t(H)} \pi_t(A \mid H) \mathbb{E}\left[R_t + w_t V_{t+1}^{w,\pi}(H_{t+1}) \mid H_t = H, A_t = A \right]. \tag{31}$$

Here, $\pi_t(A \mid H)$ is the probability of selecting batch A given history H. This expression averages the expected current discoveries and weighted successor value over the policy's batch choices.

Starting from the terminal condition $V_T^{*,w} = 0$, define recursively

$$\begin{aligned} Q_t^{*,w}(H, A) &= \mathbb{E}\left[R_t + w_t V_{t+1}^{*,w}(H_{t+1}) \mid H_t = H, A_t = A \right], \\ V_t^{*,w}(H) &= \max\nolimits_{A \in \mathcal{A}_t(H)} Q_t^{*,w}(H, A). \end{aligned} \tag{32}$$

This recursion starts at the last round, where only current discoveries contribute. Earlier rounds also include the weighted successor value already determined at the next round. Here, $Q_t^{*,w}(H, A)$ is the conditional expected value of selecting batch A, and $V_t^{*,w}(H)$ takes the maximum over feasible batches. The finite horizon and finite batch sets uniquely determine these values. Since $0 \le R_t \le B$ and $0 \le w_t \le 1$,

$$0 \le Q_t^{*,w}(H, A) \le V_t^{*,w}(H) \le (T - t)B. \tag{33}$$

To establish optimality, we first show that every policy has value at most $V_t^{*,w}$. At the terminal round $t = T$, every policy satisfies

$$V_T^{w,\pi}(H) = V_T^{*,w}(H) = 0. \tag{34}$$

Suppose that, at round $t + 1$, every policy and reachable history satisfy

$$V_{t+1}^{w,\pi}(H_{t+1}) \le V_{t+1}^{*,w}(H_{t+1}). \tag{35}$$

At round t, retain the policy's current batch selection probabilities and replace its successor value by this upper bound. Since $w_t \ge 0$, this replacement cannot decrease the conditional expected value. Equations 31 and 32 therefore give

$$\begin{aligned} V_t^{w,\pi}(H) &\le \sum\nolimits_{A \in \mathcal{A}_t(H)} \pi_t(A \mid H) \mathbb{E}\left[R_t + w_t V_{t+1}^{*,w}(H_{t+1}) \mid H_t = H, A_t = A \right] \\ &= \sum\nolimits_{A \in \mathcal{A}_t(H)} \pi_t(A \mid H) Q_t^{*,w}(H, A) \\ &\le V_t^{*,w}(H). \end{aligned} \tag{36}$$

The last inequality holds because any probability-weighted average of batch values cannot exceed their maximum. Thus, the value bound at the next round implies the bound at the current round. Backward induction from the terminal round establishes $V_t^{w,\pi}(H) \leq V_t^{*,w}(H)$ for every round, policy, and reachable history.

We next show that this upper bound is attained. Let π_w^* select a feasible batch maximizing $Q_t^{*,w}$ at every round and reachable history. At the terminal round,

$$V_T^{w,\pi_w^*}(H) = V_T^{*,w}(H) = 0. \tag{37}$$

Suppose this policy attains $V_{t+1}^{*,w}$ at every reachable history at round $t+1$. After selecting $\pi_w^*(H)$ at round t, its continuation therefore attains the optimal successor value in the definition of $Q_t^{*,w}$.

$$\begin{aligned} V_t^{w,\pi_w^*}(H) &= Q_t^{*,w}\big(H, \pi_w^*(H)\big) \\ &= \max_{A \in \mathcal{A}_t(H)} Q_t^{*,w}(H, A) \\ &= V_t^{*,w}(H). \end{aligned} \tag{38}$$

The first equality follows from the induction hypothesis, the second from the policy's maximizing batch choice, and the third from the definition of $V_t^{*,w}$. Backward induction therefore establishes that this policy attains the bound at every round and reachable history.

Finally, since $J_w(\pi) = V_0^{w,\pi}(H_0)$, every policy's objective is bounded above by $V_0^{*,w}(H_0)$, and π_w^* attains this value. Therefore,

$$\begin{aligned} J_w(\pi_w^*) &= V_0^{w,\pi_w^*}(H_0) = V_0^{*,w}(H_0) \\ &= \max_{A \in \mathcal{A}_0(H_0)} Q_0^{*,w}(H_0, A) = \max_\pi J_w(\pi). \end{aligned} \tag{39}$$

Thus, maximizing the optimal batch value at each round maximizes the weighted cumulative discovery objective. This proves Proposition 1.

B.2 Feedback and Predictive Improvement

Fix the current history H_t and query batch A_t, with remaining pool $U = U_{t+1}$ and $|U| > 0$. Under a common joint probability model, define the conditional hit probabilities before and after feedback as

$$\begin{aligned} p_x &= \Pr(z(x) = 1 \mid H_t, A_t), \\ q_x &= \Pr(z(x) = 1 \mid H_t, A_t, O_t). \end{aligned} \tag{40}$$

We analyze the Hellinger weight computed from these probabilities. Before the batch outcomes are observed, averaging the updated probability q_x over their possible values recovers the current probability p_x, giving

$$\mathbb{E}[q_x \mid H_t, A_t] = p_x. \tag{41}$$

This analysis uses conditional hit probabilities before and after experimental feedback. TDAL approximates these probabilities using MC dropout predictions.

The expected reduction in mean squared prediction error is

$$\Delta_{\text{pred}} = \frac{1}{|U|} \sum\nolimits_{x \in U} \mathbb{E}\big[(p_x - z(x))^2 - (q_x - z(x))^2\big]. \tag{42}$$

The two terms are the binary Brier losses before and after feedback (Gneiting & Raftery, 2007). Expanding their difference gives

$$(p_x - z(x))^2 - (q_x - z(x))^2 = (q_x - p_x)^2 + 2(q_x - p_x)(z(x) - q_x). \tag{43}$$

By the definition of q_x,

$$\mathbb{E}[z(x) - q_x \mid H_t, A_t, O_t] = \Pr(z(x) = 1 \mid H_t, A_t, O_t) - q_x = 0. \tag{44}$$

Given the feedback, $q_x - p_x$ is fixed, so the cross term has zero conditional expectation. The law of iterated expectations therefore yields

$$\Delta_{\text{pred}} = \frac{1}{|U|} \sum\nolimits_{x \in U} \mathbb{E}[(q_x - p_x)^2]. \tag{45}$$

Thus, feedback does not increase mean prediction error in expectation, and the expected reduction equals the mean squared change in hit probabilities.

To relate this reduction to the Hellinger weight, write

$$\begin{aligned} h_x^2 &= 1 - \sqrt{p_x q_x} - \sqrt{(1-p_x)(1-q_x)}, \\ w_t^2 &= \frac{1}{|U|} \sum\nolimits_{x \in U} h_x^2. \end{aligned} \tag{46}$$

For any $p, q \in [0,1]$, let $h = H_{\mathrm{Ber}}(p,q)$. The Bernoulli Hellinger bounds (Sason & Verdú, 2016) follow directly here. Since $\sqrt{ab} \geq \min(a,b)$,

$$h^2 \leq 1 - \min(p,q) - \min(1-p, 1-q) = |p-q|. \tag{47}$$

Conversely, define

$$u = (\sqrt{p}, \sqrt{1-p}), \qquad v = (\sqrt{q}, \sqrt{1-q}). \tag{48}$$

Cauchy–Schwarz gives

$$\begin{aligned} |p-q| &= \frac{1}{2} \sum\nolimits_{i=1}^{2} |u_i - v_i| \, |u_i + v_i| \\ &\leq \frac{1}{2} \|u-v\|_2 \|u+v\|_2 = h\sqrt{2-h^2} \leq \sqrt{2}\, h. \end{aligned} \tag{49}$$

Hence

$$h_x^4 \leq (q_x - p_x)^2 \leq 2h_x^2. \tag{50}$$

Jensen's inequality gives, for each feedback outcome,

$$\begin{aligned} w_t^4 &= \left(\frac{1}{|U|} \sum\nolimits_{x \in U} h_x^2 \right)^2 \\ &\leq \frac{1}{|U|} \sum\nolimits_{x \in U} h_x^4 \\ &\leq \frac{1}{|U|} \sum\nolimits_{x \in U} (q_x - p_x)^2 \leq 2w_t^2. \end{aligned} \tag{51}$$

Taking conditional expectations gives

$$\mathbb{E}[w_t^4] \leq \Delta_{\mathrm{pred}} \leq 2\mathbb{E}[w_t^2]. \tag{52}$$

An experiment therefore does more than establish whether queried candidates are hits. It can improve predictions for unmeasured candidates. These bounds connect that expected improvement to the Hellinger weight, supporting the use of feedback's effect on remaining predictions to weight successor value.

Now consider querying one candidate a and observing only its hit indicator. Suppress conditioning on the current history and query, and define

$$\begin{aligned} r &= \Pr(z(a) = 1) \in (0,1), \\ p_x^+ &= \Pr(z(x) = 1 \mid z(a) = 1), \\ p_x^- &= \Pr(z(x) = 1 \mid z(a) = 0). \end{aligned} \tag{53}$$

The law of total probability gives

$$p_x = r p_x^+ + (1-r) p_x^-. \tag{54}$$

After feedback, $q_x = p_x^+$ with probability r and $q_x = p_x^-$ with probability $1-r$. Hence

$$\begin{aligned} \mathbb{E}[(q_x - p_x)^2] &= r(p_x^+ - p_x)^2 + (1-r)(p_x^- - p_x)^2 \\ &= r(1-r)^2 (p_x^+ - p_x^-)^2 + (1-r) r^2 (p_x^+ - p_x^-)^2 \\ &= r(1-r)(p_x^+ - p_x^-)^2. \end{aligned} \tag{55}$$

Averaging over the remaining candidates yields

$$\Delta_{\mathrm{pred}}(a) = \frac{r(1-r)}{|U|} \sum\nolimits_{x \in U} (p_x^+ - p_x^-)^2. \tag{56}$$

This proves Proposition 2.

Outcome uncertainty and its influence on remaining candidates jointly determine predictive improvement. If neither outcome changes remaining predictions, the feedback provides no predictive improvement. Otherwise, even a query without a hit changes remaining predictions and produces a nonzero weight. This explains how TDAL retains the current discovery reward while allowing experiments without hits to contribute to learning through successor value.

The decomposition into two outcomes is the special case of one query revealing only a binary label. TDAL observes a batch of continuous phenotypes instead. The general identity for predictive improvement conditions on the complete batch observations O_t and also applies to that feedback under the same assumptions about the joint model. It does not require a binary observation space.

B.3 Final Discovery Gap

Fix the initial history H_0, surrogate update rule, and algorithmic randomness. At each round, $0 \le R_t \le B$, and the feedback weight is a measurable function $w_t = w(H_t, A_t, H_{t+1}) \in [0, 1]$. This permits the weight, current discovery count, and successor value to depend on the same feedback. Write

$$W_t = \prod_{j=0}^{t-1} w_j, \qquad W_0 = 1. \tag{57}$$

At each history H, let $s = s_t(H)$ be the screening state and let θ denote the network parameters after all online updates at that history. The learned policy selects a feasible batch maximizing $Q_\theta(s, A)$. Its value estimation error is

$$\varepsilon_t^w(H) = \max_{A \in \mathcal{A}_t(H)} |Q_\theta(s, A) - Q_t^{*,w}(H, A)|. \tag{58}$$

First consider how this error affects the current batch choice. Fix a history H, let A^* maximize $Q_t^{*,w}(H, A)$, and let $\widehat{A}$ maximize $Q_\theta(s, A)$. By the error definition,

$$\begin{aligned} V_t^{*,w}(H) &= Q_t^{*,w}(H, A^*) \\ &\le Q_\theta(s, A^*) + \varepsilon_t^w(H) \\ &\le Q_\theta(s, \widehat{A}) + \varepsilon_t^w(H) \\ &\le Q_t^{*,w}(H, \widehat{A}) + 2\varepsilon_t^w(H). \end{aligned} \tag{59}$$

The second inequality uses the maximizing property of $\widehat{A}$. Thus,

$$0 \le V_t^{*,w}(H) - Q_t^{*,w}(H, \widehat{A}) \le 2\varepsilon_t^w(H). \tag{60}$$

The gap between the optimal batch value and the value of the selected batch is therefore at most twice the estimation error.

Next, accumulate these losses across the screening trajectory. Let $\widehat{A}_t$ be the batch selected by the learned policy at round t. The weighted Bellman equation gives

$$Q_t^{*,w}(H_t, \widehat{A}_t) = \mathbb{E}\Big[R_t + w_t V_{t+1}^{*,w}(H_{t+1}) \mid H_t, A_t = \widehat{A}_t\Big]. \tag{61}$$

Since W_t is determined by the current history and $W_{t+1} = W_t w_t$, taking expectations along the learned policy gives

$$\begin{aligned} &\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} W_t\big(V_t^{*,w}(H_t) - Q_t^{*,w}(H_t, \widehat{A}_t)\big) \mid H_0\right] \\ &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} \big(W_t V_t^{*,w}(H_t) - W_t R_t - W_{t+1} V_{t+1}^{*,w}(H_{t+1})\big) \mid H_0\right] \\ &= V_0^{*,w}(H_0) - J_w(\pi_\theta). \end{aligned} \tag{62}$$

Value terms from adjacent rounds cancel, leaving the initial value $V_0^{*,w}(H_0)$. The terminal term vanishes because $V_T^{*,w} = 0$. Proposition 1 and the batch value gap then imply

$$\max_\pi J_w(\pi) - J_w(\pi_\theta) \le 2\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} W_t \varepsilon_t^w(H_t) \mid H_0\right]. \tag{63}$$

Finally, relate the weighted objective to final discoveries. Since

$$C_T = C_0 + \sum_{t=0}^{T-1} R_t, \tag{64}$$

every policy satisfies

$$\mathbb{E}_\pi[C_T \mid H_0] = C_0 + J_w(\pi) + \mathbb{E}_\pi\left[\sum_{t=0}^{T-1}(1 - W_t)R_t \mid H_0\right]. \tag{65}$$

The last term counts the expected discoveries omitted when each round receives weight W_t instead of one. It is nonnegative because $R_t \geq 0$ and $W_t \leq 1$. Let π_C^* maximize expected final discoveries under the same initial information and budget, and write

$$\Delta_C(w) = \mathbb{E}_{\pi_C^*}\left[\sum_{t=0}^{T-1}(1 - W_t)R_t \mid H_0\right]. \tag{66}$$

Subtracting the final discovery counts and using the nonnegative weighting bias of the learned policy gives

$$\begin{aligned}
&\mathbb{E}_{\pi_C^*}[C_T \mid H_0] - \mathbb{E}_{\pi_\theta}[C_T \mid H_0] \\
&\quad \leq \Delta_C(w) + J_w(\pi_C^*) - J_w(\pi_\theta) \\
&\quad \leq \Delta_C(w) + \max_\pi J_w(\pi) - J_w(\pi_\theta) \\
&\quad \leq \Delta_C(w) + 2\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} W_t \varepsilon_t^w(H_t) \mid H_0\right].
\end{aligned} \tag{67}$$

This proves Proposition 3.

Feedback weights and the two error terms. Consider two weight rules w, w' in the same experimental environment. If, at every reachable history, batch, and feedback outcome,

$$w'(H, A, H') \geq w(H, A, H'), \tag{68}$$

then $W_t' \geq W_t$ on every trajectory. Taking expectations under the same optimal discovery policy π_C^* yields

$$\Delta_C(w') \leq \Delta_C(w). \tag{69}$$

Thus, increasing feedback weights reduces the objective weighting bias.

To examine the error term, consider constant weights $w_t = w$ and assume all weight settings being compared share a common value error bound

$$\varepsilon_t^w(H) \leq \varepsilon. \tag{70}$$

Then $W_t = w^t$, and $R_t \leq B$ gives

$$\Delta_C(w) \leq B\sum_{t=0}^{T-1}(1 - w^t), \tag{71}$$

while

$$2\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} W_t \varepsilon_t^w(H_t) \mid H_0\right] \leq 2\varepsilon\sum_{t=0}^{T-1} w^t. \tag{72}$$

The bias bound is nonincreasing in w, whereas the estimation error term's bound is nondecreasing under the common error bound ε. Smaller weights reduce the error bound but attenuate more future discovery return. Larger weights retain more future return but increase the error bound. In the learned model, $\varepsilon_t^w(H)$ and the selected trajectory depend on w. Comparing weight settings therefore also involves their different value estimation errors. When $w = 1$, the objective weighting bias vanishes and the final discovery gap is at most $2T\varepsilon$. If also $\varepsilon = 0$, the learned policy achieves the optimal expected final discovery count.

C Datasets and Experimental Setup

C.1 Datasets and Hit Definitions

The eight public screens comprise four single gene tasks, three double gene tasks, and one triple gene task, covering T cell function, cancer cell growth, immune killing, and genetic interactions.

Table C1: Perturbation types, phenotypes, and hit rules for eight tasks.

Dataset	Perturbation	Phenotype	Hit rule
IFN-γ (Schmidt et al., 2022)	Single gene	IFN-γ production by T cells	Top 5% by absolute score
IL-2 (Schmidt et al., 2022)	Single gene	IL-2 production by T cells	Top 5% by absolute score
NK (Zhuang et al., 2019)	Single gene	Cancer cell sensitivity to NK cell killing	Top 5% by absolute score
Carnevale (Carnevale et al., 2022)	Single gene	T cell proliferation under adenosine suppression	Top 5% by absolute score
Simpson K562 (Simpson et al., 2023)	Two genes	K562 cell growth after dual gene repression	Top 5% by raw score
SCHEMATIC (Fong et al., 2025)	Two genes	Genetic interactions across cancer cell backgrounds	Bottom 5% by raw score
Kuzmin yeast (Kuzmin et al., 2020)	Three genes	Genetic interactions in triple gene yeast perturbations	Bottom 5% by raw score
Horlbeck Jurkat (Horlbeck et al., 2018)	Two genes	Jurkat cell growth after dual gene repression	Bottom 5% by raw score

Each candidate is a single gene or combinatorial perturbation. Table C1 summarizes perturbation types, phenotypes, and hit rules.

Raw phenotype scores are transformed into effects $e(x)$ according to the direction in the table. For a pool of $N = |\mathcal{X}|$ candidates, the score at rank $\max\{1, \lfloor 0.05N \rfloor\}$ in descending effect order defines the fixed threshold τ, with hits given by $z(x) = \mathbf{1}\{e(x) \geq \tau\}$. The target hit proportion is 5%, and τ is the corresponding phenotype score threshold. The threshold is fixed before screening. The phenotype and hit label of each candidate are revealed when it is queried.

IFN-γ. This task identifies genes that regulate interferon-γ production in T cells. Enrichment differences between populations with high and low cytokine levels measure perturbation effects. We target strong effects in either direction (Schmidt et al., 2022).

IL-2. This task identifies regulators of interleukin-2 production following T cell stimulation. Enrichment differences between populations with high and low expression define the phenotype, with strong effects in either direction included as discovery targets (Schmidt et al., 2022).

NK. This task studies cancer cell genes that affect sensitivity or resistance to natural killer cells. The phenotype is the sum of standardized enrichment scores across two replicates for perturbed K562 cells exposed to NK cells relative to controls. Strong effects in either direction are targeted (Zhuang et al., 2019).

Carnevale. This task studies genes that affect T cell proliferation under immunosuppressive conditions. We use enrichment scores associated with proliferation from the screen under adenosine suppression and target perturbations that strongly increase or decrease proliferation (Carnevale et al., 2022).

Simpson K562. The original study examines how DNA repair genes jointly affect cell growth and PARP inhibitor response. We use its double gene CRISPRi growth phenotype and target combinations with the highest growth scores (Simpson et al., 2023).

SCHEMATIC. This task seeks gene combinations that cause synthetic lethality across cancer cell backgrounds. We use the multilineage interaction score to identify strong negative interactions whose growth or survival defects exceed expectations from individual perturbations (Fong et al., 2025).

Kuzmin yeast. This task examines how duplicate genes compensate for one another and how this compensation depends on other genes. We use the corrected trigenic interaction score to target combinations with stronger growth defects than expected (Kuzmin et al., 2020).

Horlbeck Jurkat. This task examines the effect of jointly repressing two genes on Jurkat cell growth. We use the double gene CRISPRi growth phenotype and target combinations that strongly inhibit growth (Horlbeck et al., 2018).

C.2 Experimental Protocol and Hyperparameters

Table C2 summarizes candidate counts, pool hit counts, and screening configurations.

Table C2: Candidate pools, hits, and budgets for eight tasks.

Dataset	N	Pool hits	n_0	B	T	M
IFN-γ	17,470	873	128	32	60	2,048
IL-2	17,470	873	128	32	60	2,048
NK	17,533	876	128	32	60	2,048
Carnevale	17,526	876	128	32	60	2,048
Simpson K562	143,345	7,167	1,024	128	200	26,624
SCHEMATIC	11,067	553	128	32	60	2,048
Kuzmin yeast	214,642	10,732	1,024	256	200	52,224
Horlbeck Jurkat	91,378	4,568	1,024	128	200	26,624

N is the number of queryable perturbations with valid measured phenotypes and fixed representations in the retrospective evaluation. Combinatorial tasks count gene combinations as candidates. All candidate identities and representations are known at the start. Each method initially receives the phenotypes of the n_0 candidates in D_0, while the phenotypes of the remaining $N - n_0$ candidates are withheld. After batch A_t is selected, the evaluation environment returns its recorded experimental outcomes, adds these observations to the measured data, and removes the candidates from the unmeasured pool. Screening returns all verified hit candidates, whose count C_T measures final discovery.

For seed i, let $C_{t,i}$ be the number of hits in the measured set after t active rounds, with $C_{0,i}$ the number in the initial set. The final cumulative hit count is

$$C_{T,i} = C_{0,i} + \sum_{t=0}^{T-1} R_{t,i}. \tag{73}$$

This metric counts distinct discovered hits within the total budget, including initialization. Recall is $C_{T,i}$ divided by the fixed pool hit count in Table C2. Batch size B gives the number of new queries per round, T is the number of active rounds, and the total budget $M = n_0 + BT$ includes initialization and all subsequent queries.

Each task is repeated with seeds 0–4. Methods share the candidate pool, initial set, hit threshold, experimental budget, and underlying phenotype dataset within each seed. Method differences are paired within each task and seed before aggregation across seeds.

Hyperparameter settings. Table C3 summarizes TDAL's model and training settings.

C.3 Baseline Methods

For batch perturbation screening, we compare representative methods for hit discovery, Bayesian optimization, and batch active learning, with random selection as a reference. UCB, EI, TS, Top-Uncertain, and OAE use the GeneDisco Bayesian MLP and retrain their surrogate after each round of new observations (Mehrjou et al., 2022).

PoH. Many Needles selects the B unmeasured candidates with the highest estimated probability of exceeding the hit threshold (Rubbi et al., 2026). At each round, we refit a Bayesian MLP with two hidden layers of widths 128 and 64 on the measured data. For each candidate $x \in U_t$, we obtain 50 MC dropout predictions and use their mean and standard deviation to form a Gaussian predictive approximation. The probability of exceeding τ then serves as the acquisition score.

Table C3: Key TDAL hyperparameters.

Component	Parameter	Value
Surrogate	Hidden widths and dropout	Widths 128, 64, dropout probability 0.2
Surrogate	Training/validation split	80%/20%
Surrogate	Optimizer	Adam, learning rate 10^{-3}
Surrogate	Training batch and epochs	Batch size 64, at most 100 epochs, early stopping patience 10
Surrogate	MC samples	$K = 50$
Acquisition network	Projection and hidden widths	Projection dimension 64, hidden widths 128, 128
TD learning	Optimizer	AdamW, learning rate 5×10^{-4}
TD learning	Training batch and updates	At most 16 transitions per update, 20 updates initially and after each active round
Initial LBO	Transition limit	$\min\{80,\ 2\lfloor \lvert D_0 \rvert / B \rfloor\}$
Online LBO	Rounds 1–8/9–32/33–64	At most 8/4/2 transitions per round
Online LBO	From round 65	At most 1 transition every 4 rounds

OAE. OAE uses optimistic estimates for batch discovery of candidates with high responses. We use its released Bayesian variant, which combines predictive means and uncertainty to select each batch after aligning predictions with the effect direction (Pacchiano et al., 2023).

DiscoBAX. DiscoBAX targets a diverse subset of candidates with high responses and uses fantasized observations to estimate information gain about that subset. We use the authors' surrogate training and acquisition implementation, with two hidden layers of 128 and 64 units and dropout 0.2 to match TDAL and PoH in network capacity. We evaluate this implementation on the four single gene perturbation tasks (Lyle et al., 2023).

UCB. Upper Confidence Bound combines the predictive mean with a weighted uncertainty term. It favors candidates with high predicted effects or high uncertainty to balance exploitation and exploration (Srinivas et al., 2010).

Expected Improvement (EI). EI estimates the expected improvement over the best effect observed so far and prioritizes candidates with larger expected gains (Jones et al., 1998).

Thompson Sampling (TS). TS draws an effect function from the surrogate posterior and selects a batch by ranking unmeasured candidates under that draw, allowing model uncertainty to influence selection (Thompson, 1933).

TopUncertain. TopUncertain ranks candidates by predictive standard deviation and allocates experiments to regions where the surrogate is most uncertain (Mehrjou et al., 2022).

BADGE. BADGE selects diverse batches using k-means++ in a gradient representation space. We use the GeneDisco regression implementation and its proxy gradient representations (Ash et al., 2020; Mehrjou et al., 2022).

CoreSet. CoreSet greedily expands geometric coverage in the candidate representation space extracted by the surrogate. It successively selects the candidate farthest from the existing selection centers (Sener & Savarese, 2018).

KMeans-Embedding. This method uses k-means to cluster candidate representations extracted by the surrogate and selects candidates near cluster centers to represent different regions of the candidate space (Mehrjou et al., 2022).

Random. Random samples uniformly without replacement from the current unmeasured pool, providing a reference for random screening.

D Complete Benchmark Results

D.1 Single Gene Perturbation Results

Table D1 reports the complete endpoint results for the four single gene perturbation tasks in Figure 3a–d, supplementing the discovery trajectories in the main text with cumulative hit counts for each method, variation across seeds, and recall.

Table D1: Final cumulative hits and recall on the four single gene perturbation tasks. Entries are mean ± sample SD of cumulative hits across five random seeds, with mean recall in parentheses. The second highest mean in each column is underlined.

Method	IFN-γ	IL-2	NK	Carnevale
TDAL	270.6 ± 6.1 (31.00%)	367.4 ± 3.0 (42.08%)	$\underline{109.4 \pm 9.3}$ ($\underline{12.49\%}$)	161.2 ± 8.6 (18.40%)
PoH	256.6 ± 14.6 (29.39%)	$\underline{347.8 \pm 6.0}$ ($\underline{39.84\%}$)	101.0 ± 4.1 (11.53%)	145.6 ± 2.7 (16.62%)
OAE	194.0 ± 13.0 (22.22%)	334.2 ± 6.4 (38.28%)	97.2 ± 7.2 (11.10%)	124.6 ± 5.8 (14.22%)
DiscoBAX	218.0 ± 10.4 (24.97%)	266.0 ± 17.7 (30.47%)	81.0 ± 5.2 (9.25%)	151.8 ± 5.9 (17.33%)
UCB	196.4 ± 12.4 (22.50%)	331.0 ± 10.2 (37.92%)	85.4 ± 6.2 (9.75%)	123.8 ± 8.3 (14.13%)
Expected Improvement	185.8 ± 16.9 (21.28%)	315.4 ± 7.1 (36.13%)	92.6 ± 4.1 (10.57%)	123.2 ± 8.3 (14.06%)
Thompson Sampling	178.2 ± 12.1 (20.41%)	299.8 ± 6.5 (34.34%)	98.4 ± 8.4 (11.23%)	114.0 ± 10.8 (13.01%)
TopUncertain	200.8 ± 10.1 (23.00%)	298.6 ± 13.6 (34.20%)	91.4 ± 4.4 (10.43%)	127.2 ± 5.5 (14.52%)
BADGE	190.8 ± 8.6 (21.86%)	237.0 ± 12.0 (27.15%)	99.0 ± 5.2 (11.30%)	140.2 ± 7.8 (16.00%)
CoreSet	$\underline{259.8 \pm 2.2}$ ($\underline{29.76\%}$)	298.4 ± 4.8 (34.18%)	114.0 ± 6.1 (13.01%)	$\underline{159.4 \pm 7.3}$ ($\underline{18.20\%}$)
KMeans-Embedding	153.2 ± 11.2 (17.55%)	202.4 ± 6.5 (23.18%)	82.0 ± 3.4 (9.36%)	117.2 ± 8.9 (13.38%)
Random	94.0 ± 6.0 (10.77%)	102.2 ± 7.0 (11.71%)	102.6 ± 10.5 (11.71%)	103.4 ± 10.8 (11.80%)

D.2 Combinatorial Perturbation Results

Table D2 reports the complete endpoint results for the four combinatorial perturbation tasks in Figure 3e–h, providing cumulative hit counts for each method, variation across seeds, and recall for the discovery performance comparison in the main text.

Table D2: Final cumulative hits and recall on the four main combinatorial perturbation tasks. Entries are mean ± sample SD of cumulative hits across five random seeds, with mean recall in parentheses. The second highest mean in each column is underlined.

Method	Simpson K562	SCHEMATIC	Kuzmin yeast	Horlbeck Jurkat
TDAL	$\underline{6765.4 \pm 102.5}$ $\underline{(94.40\%)}$	160.4 ± 14.2 (29.01%)	4141.0 ± 194.0 (38.59%)	$\underline{4542.8 \pm 4.8}$ $\underline{(99.45\%)}$
PoH	6771.0 ± 75.7 (94.47%)	187.2 ± 5.1 (33.85%)	$\underline{4111.4 \pm 42.2}$ $\underline{(38.31\%)}$	4551.4 ± 1.5 (99.64%)
OAE	5951.2 ± 36.4 (83.04%)	104.8 ± 10.0 (18.95%)	3420.4 ± 83.1 (31.87%)	4489.2 ± 23.3 (98.27%)
UCB	5955.4 ± 46.0 (83.09%)	105.0 ± 6.6 (18.99%)	3636.6 ± 157.1 (33.89%)	4394.0 ± 103.0 (96.19%)
Expected Improvement	5005.2 ± 176.9 (69.84%)	117.0 ± 2.9 (21.16%)	3013.4 ± 278.6 (28.08%)	4134.2 ± 92.5 (90.50%)
Thompson Sampling	5903.8 ± 17.1 (82.37%)	99.6 ± 6.9 (18.01%)	3196.2 ± 164.5 (29.78%)	4044.8 ± 48.9 (88.55%)
TopUncertain	3758.0 ± 181.5 (52.43%)	107.8 ± 7.6 (19.49%)	3759.6 ± 145.6 (35.03%)	4059.8 ± 136.6 (88.87%)
BADGE	932.8 ± 13.3 (13.02%)	151.4 ± 11.2 (27.38%)	2671.8 ± 25.2 (24.90%)	1622.6 ± 30.0 (35.52%)
CoreSet	1693.6 ± 44.4 (23.63%)	$\underline{182.0 \pm 8.0}$ $\underline{(32.91\%)}$	3417.0 ± 101.9 (31.84%)	2845.6 ± 78.3 (62.29%)
KMeans-Embedding	767.4 ± 18.8 (10.71%)	149.8 ± 7.8 (27.09%)	2646.6 ± 49.2 (24.66%)	1316.0 ± 17.8 (28.81%)
Random	1313.4 ± 59.8 (18.33%)	99.2 ± 10.0 (17.94%)	2603.2 ± 45.3 (24.26%)	1339.0 ± 25.3 (29.31%)

D.3 Rank Consistency Across Tasks

We rank the eleven common methods by their mean final discovery counts over five seeds on each task. DiscoBAX is evaluated on four tasks and is excluded from this comparison. Table D3 summarizes mean ranks, variation across tasks, and top three coverage. TDAL has the best mean rank of 1.63 and the smallest rank standard deviation of 0.74, and is the only method ranked among the top three on all eight tasks. Omitting each task in turn leaves TDAL with the best average rank over the remaining seven tasks. Thus, its leading average rank does not depend on any single task.

To assess variation across seeds, we first rank methods separately for each task and seed. We then calculate the rank standard deviation across the five seeds within each task and average it over tasks. TDAL has the smallest rank variation across tasks, while CoreSet has the smallest variation across seeds.

E Landscape Structure and Active Discovery

E.1 Local Hit Enrichment and Discovery Gains on Real Tasks

Local hit enrichment. We compare the hit fraction around hit candidates with the overall hit fraction in the analysis sample to measure how hits cluster in candidate representation space. Neighbors are determined by cosine distance after standardizing the fixed candidate representations $\phi(x)$ and applying L2 normalization.

Table D3: Rank consistency across the eight main tasks. Task ranks are based on mean final discovery counts over five seeds among the eleven common methods. Ties receive average ranks. Rank SD is the sample standard deviation across task ranks. Seed rank SD is the mean of the sample standard deviations across five seeds within each task.

Method	Mean rank	Rank SD	Range	Top three	Seed rank SD
TDAL	1.63	0.74	1–3	8/8	0.713
PoH	2.13	1.13	1–4	7/8	0.686
CoreSet	4.63	3.16	1–8	4/8	0.498
OAE	5.38	2.07	3–9	2/8	1.233
UCB	5.63	2.45	3–10	1/8	1.131
TopUncertain	6.00	1.93	3–9	1/8	1.004
EI	6.75	1.39	5–8	0/8	0.878
BADGE	7.13	2.47	4–10	0/8	0.757
TS	7.50	1.93	5–10	0/8	1.003
KMeans-Embedding	9.63	2.00	5–11	0/8	0.594
Random	9.63	2.77	3–11	1/8	0.753

For each task, we randomly sample hit candidates and other candidates in their proportions in the full candidate pool, forming an analysis sample $\mathcal{P}$ of at most 10,000 candidates. Let $\mathcal{P}_+ = \{x \in \mathcal{P} : z(x) = 1\}$ be its hit set. For each hit candidate x, let $\mathcal{N}_{10}(x)$ contain the ten nearest other candidates within $\mathcal{P}$. We compute the hit fraction among these ten neighbors, average it over all hit candidates, and divide by the overall hit fraction in the sample to obtain

$$\mathrm{Enrich}_{10} = \frac{\dfrac{1}{|\mathcal{P}_+|}\displaystyle\sum_{x\in\mathcal{P}_+}\frac{1}{10}\sum_{y\in\mathcal{N}_{10}(x)} z(y)}{|\mathcal{P}_+|/|\mathcal{P}|}. \tag{74}$$

The numerator is the mean neighbor hit fraction around hit candidates, and the denominator is the overall hit fraction in the analysis sample. For example, an overall hit fraction of 5% and a mean neighbor hit fraction of 15% give 3× enrichment. Values above one indicate that other hits are more common near hit candidates. Values near one indicate a similar hit fraction to the sample overall, and values below one indicate a lower local hit fraction. We repeat sampling and calculation three times for each task. Table E1 reports the mean enrichment over these three samples.

Discovery gains. For each task, we divide each method's final cumulative hit count by Random's count under the same seed, then average over five seeds. The shared methods are PoH, OAE, UCB, EI, TS, TopUncertain, BADGE, CoreSet, and KMeans-Embedding. Their average gain is the mean of these nine methods' results. Across the eight tasks, this average gain increases with local hit enrichment (Spearman $\rho = 0.95$, $n = 8$). Both enrichment and discovery gains are near random levels on NK and higher on Simpson K562, showing a close association between local hit structure and active discovery gains across tasks.

Table E1: Hit clustering and active discovery gains on real tasks.

Dataset	Hit enrichment (10 nearest neighbors)	Shared methods / Random	TDAL / Random
NK	0.83×	0.94×	1.08×
Carnevale	1.25×	1.27×	1.57×
SCHEMATIC	1.51×	1.36×	1.63×
Kuzmin	1.76×	1.28×	1.59×
IFN-γ	2.53×	2.15×	2.89×
IL-2	3.58×	2.91×	3.61×
Horlbeck Jurkat	5.84×	2.61×	3.39×
Simpson K562	6.27×	3.11×	5.16×

E.2 Noisy Multicluster Landscapes

Landscape construction. We construct landscapes in which hits are dispersed or concentrated in six clusters. The 17,660 candidates have fixed two-dimensional coordinates that are defined without phenotype information. KMeans divides these coordinates into six regions, each with a response peak at its center.

Within each region, candidates closer to the center receive higher response scores. We sort candidates by distance, negate their ranks, and divide by the region's hit quota. The quotas are 148 for one region and 147 for each of the other five, giving 883 hits in total. This scaling produces clusters of similar hit counts when no noise is added. We standardize the scores across candidates and mix them with independent standardized Gaussian noise. Increasing the noise contribution disperses hits away from the cluster centers.

To keep phenotype values comparable, we rank the mixed scores and assign them the same fixed set of standard normal quantiles. The top 5% are designated as hits. All settings therefore share the phenotype distribution and 883 hits, while hit locations range from nearly random at $1.05\times$ mean enrichment to six concentrated clusters at $17.71\times$. Surrogates receive only candidate coordinates and measured phenotypes.

Screening settings. We use $n_0 = 128$, $B = 32$, and $T = 60$, giving a total query budget of 2,048. Each landscape setting has five paired repetitions with independent noise and selection seeds 5–9. Methods share the landscape, initial set, and budget within each repetition. The $17.71\times$ landscape contains no noise, so only selection seeds vary in this setting. Model and training settings follow the real tasks. TDAL and PoH retain their empirical hit frequency and Gaussian tail estimators, respectively.

The comparison includes TDAL and the eleven baselines in Figure 4, giving 300 runs across five settings. Their mean hit enrichments are 1.05, 2.53, 3.83, 10.00, and $17.71\times$. Enrichment is computed using ten nearest neighbors in the full pool with Euclidean distances in the fixed coordinates, then averaged over the five realizations. The four snapshots in Figure 4b show the first realization with its own enrichment. Table E2 reports final recall for these same methods and settings, and Figure E1 shows their discovery trajectories.

Discovery performance. With nearly random hit locations, mean discovery gains are close to one. Between the least and most clustered settings, TDAL's mean gain over Random rises from $1.04\times$ to $2.26\times$, and PoH's from $1.06\times$ to $1.94\times$. OAE, UCB, and TS reach gains of $1.62\times$, $1.62\times$, and $1.67\times$, respectively, at the highest enrichment. Thus, clustered hits provide useful information for several acquisition functions, although their gains differ across settings. BADGE and KMeans-Embedding do not improve consistently with clustering, while EI, TopUncertain, and DiscoBAX remain below Random in the four structured settings.

Table E2: Final recall (%) on the five multicluster landscapes in Figure 4b. Column headings give mean hit enrichment. Values are means ± sample standard deviations over five paired repetitions.

Method	$1.05\times$	$2.53\times$	$3.83\times$	$10.00\times$	$17.71\times$
TDAL	11.48 ± 0.39	17.33 ± 1.87	16.78 ± 1.38	18.01 ± 9.40	24.28 ± 9.23
PoH	11.66 ± 1.05	10.83 ± 4.00	16.76 ± 2.62	22.74 ± 4.19	20.82 ± 7.05
OAE	11.78 ± 0.60	16.49 ± 1.11	16.24 ± 0.74	16.99 ± 1.07	17.17 ± 0.17
DiscoBAX	11.51 ± 1.56	5.73 ± 1.46	5.73 ± 2.92	3.15 ± 1.57	3.53 ± 2.52
UCB	11.73 ± 0.75	16.19 ± 1.30	16.26 ± 1.09	16.60 ± 1.01	17.19 ± 0.17
EI	11.33 ± 1.62	6.34 ± 0.76	4.92 ± 1.37	2.45 ± 1.85	2.27 ± 1.35
TS	11.39 ± 1.08	14.93 ± 1.82	15.52 ± 1.41	16.81 ± 1.00	17.80 ± 0.95
TopUncertain	11.71 ± 0.98	5.30 ± 1.21	2.63 ± 1.08	0.61 ± 0.10	0.66 ± 0.17
BADGE	12.75 ± 1.33	11.42 ± 0.83	10.80 ± 1.47	10.51 ± 1.19	10.83 ± 0.87
CoreSet	11.60 ± 1.95	9.60 ± 0.85	10.55 ± 1.25	12.71 ± 0.43	13.16 ± 0.78
KMeans-Embedding	12.07 ± 0.94	11.87 ± 1.60	11.01 ± 0.67	9.90 ± 1.31	8.13 ± 0.49
Random	11.17 ± 1.30	11.80 ± 0.79	11.82 ± 0.55	10.69 ± 0.57	10.65 ± 0.45

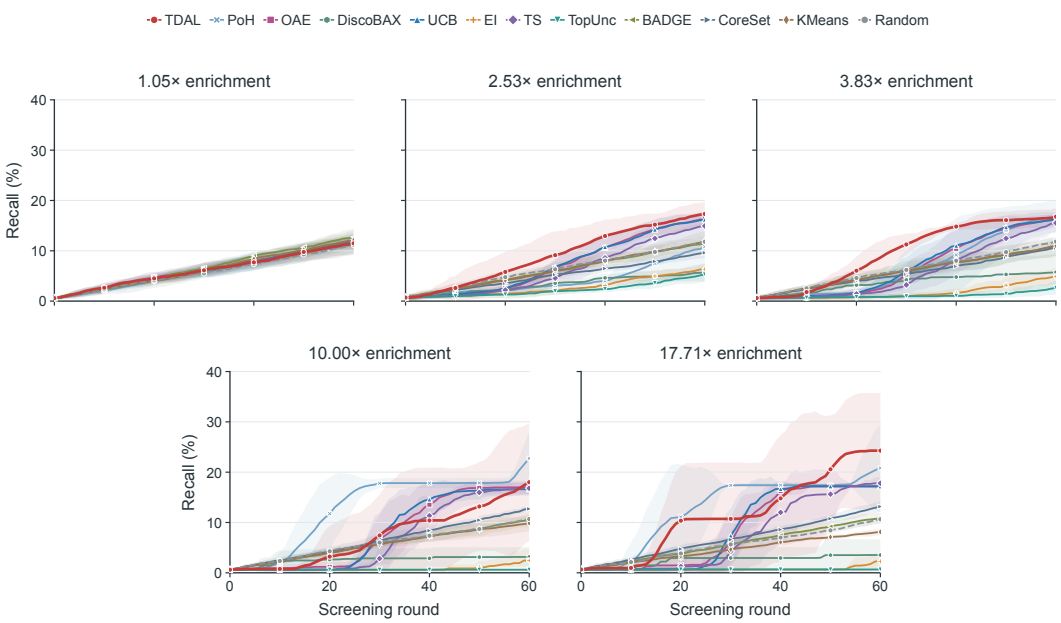


Figure E1: Discovery trajectories for all twelve methods in Figure 4b. Panel titles give mean hit enrichment. Lines show mean recall over five paired repetitions, with shaded 95% confidence intervals. Round 0 is the initial set of 128 candidates. All panels share axis ranges.

F Ablation Studies and Selection Dynamics

F.1 Component Ablations

We compare five ablations with full TDAL on IFN-γ, IL-2, Kuzmin, and Horlbeck Jurkat. All methods share the surrogate configuration, fixed candidate representations, initial sets, experimental budgets, and five random seeds. Final cumulative hit counts are reported in Table 1.

Surrogate only. This ablation refits the same surrogate as TDAL after each round of observations but selects candidates with the highest $\hat{p}_t(x)$ without training an acquisition network, breaking exact probability ties at random. Its mean final recall is 0.54–13.01 percentage points lower than full TDAL across the four tasks. Updating response predictions alone therefore does not recover the full model's discovery gains. Learning a selection function from batch feedback further improves screening.

w/o adaptive w_t. This ablation retains successor value bootstrapping but replaces the adaptive Hellinger weight with 1, applying the same successor weight in every round. Mean final recall decreases by 3.88–9.03 percentage points across the four tasks. Giving successor value full weight at every round therefore performs less well than adjusting its contribution to the observed prediction changes.

w/o LBO. This ablation disables LBO transitions constructed from the initial set, the associated initial TD training, and online LBO. The acquisition network learns only after receiving real batch feedback. Mean final recall is 0.42–5.54 percentage points lower than full TDAL across the four tasks. LBO reuses measured outcomes to provide training experience at initialization and supplement the single real transition obtained per online round.

w/o successor. This ablation sets the training target to $Y = R$, retaining the hit probability term, acquisition network, and both real and LBO transitions. With surrogate predictions held fixed, the network fits the difference between the observed batch discovery count and its surrogate estimate. Mean final recall exceeds Surrogate only on all four tasks but is 0.15–1.21 percentage points lower than full TDAL. Current feedback can therefore train an effective selection function, while successor value bootstrapping further increases final discoveries.

w/o anchor. This ablation uses $q_\theta(s_t, x) = f_\theta(s_t, c_t(x))$. Both current and successor batch values sum network outputs without explicitly adding $\hat{p}_t(x)$. Surrogate prediction features remain in the inputs. Relative to full TDAL, mean final recall decreases by 0.96 and 2.82 percentage points on IFN-γ and IL-2, respectively, and increases by 0.31 and 0.16 points on Kuzmin and Horlbeck Jurkat. The two single perturbation tasks each use 60 screening rounds, whereas the two combinatorial tasks each use 200. More rounds provide the acquisition network with more feedback and training opportunities. The explicit hit probability term is more beneficial on the two tasks with 60 rounds. This pattern is consistent with a useful probability reference early in learning and motivates gradually reducing its contribution as feedback accumulates. Testing such a schedule within the same task would separate the effect of screening horizon from differences between datasets.

F.2 Training Dynamics and Candidate Reranking

Candidates analyzed. On IL-2, 85 hit candidates are discovered during active screening in all five TDAL runs and none of the five Surrogate only runs. We compare their discovery times and selection ranks across the five random seeds.

Training and discovery across seeds. Figure F1a–b shows TD loss per round and cumulative discoveries among these 85 candidates. Early loss changes vary in direction and magnitude across seeds. For all five seeds, fluctuations are smaller in the last 30 than in the first 10 rounds. Discoveries concentrate in the second half, with 74.1%–87.1% of the set discovered in the last 30 rounds (Table F1). This common timing pattern shows that these candidates mostly enter selected batches after feedback accumulates.

Candidate reranking at selection. We compare each candidate's TDAL score rank and concurrent surrogate hit probability rank at selection, using the same unmeasured pool and surrogate predictions. Across seeds, mean TDAL ranks are 16.1–17.2, whereas mean surrogate ranks are 9,442–9,780. Of 425 selection records, 413 have surrogate ranks outside the top $B = 32$ (97.2%, Figure F1c–g). Across seeds, the acquisition network consistently promotes candidates ranked low by concurrent surrogate predictions into the selected batch.

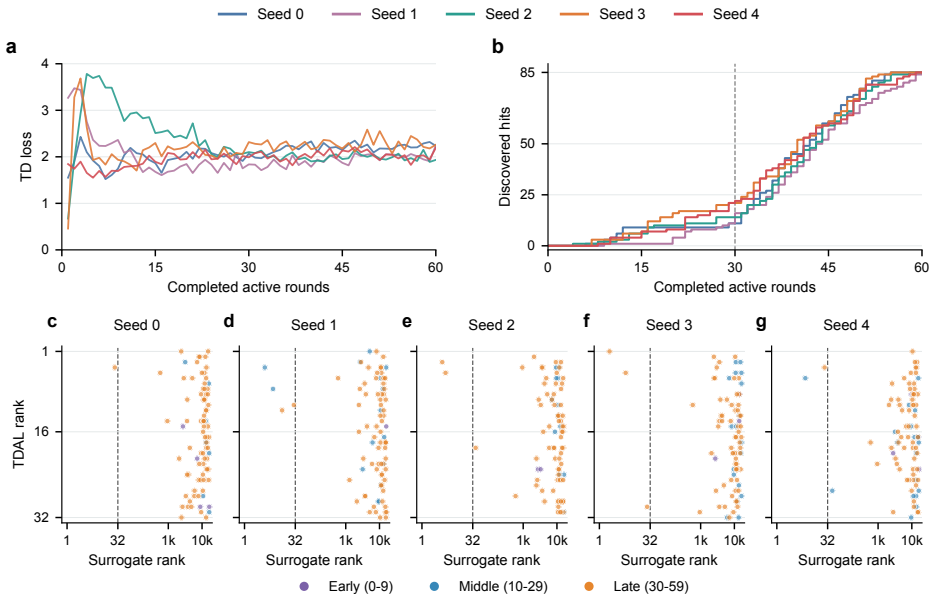


Figure F1: Training dynamics and candidate reranking on IL-2 across five random seeds. **a,** TD loss per round. **b,** Cumulative discoveries among the 85 hit candidates analyzed. The dashed line marks 30 completed rounds. **c–g,** Surrogate and TDAL ranks at selection for seeds 0–4, with a logarithmic horizontal axis and a dashed line at surrogate rank 32. Colors indicate selection in rounds 0–9, 10–29, and 30–59.

Table F1: Discovery stages and selection ranks of the hit candidates analyzed across five seeds. Stage columns count newly discovered candidates. Ranks are computed in the same unmeasured pool at selection and averaged over the 85 candidates.

Seed	Rounds 0–9	Rounds 10–29	Rounds 30–59	Mean TDAL rank	Mean surrogate rank
0	4	7	74	16.6	9,780
1	1	15	69	16.1	9,442
2	2	12	71	16.8	9,574
3	3	18	64	17.2	9,522
4	4	18	63	16.4	9,479

G Batch Size Sensitivity and Computational Cost

G.1 Batch Size Sensitivity

Experimental settings. We vary batch size at fixed initial sample size and query budget on IL-2 and Simpson K562 to compare discovery at different feedback frequencies. IL-2 uses $n_0 = 128$, $M = 2{,}048$, and $B \in \{16, 32, 64, 128\}$. Simpson K562 uses $n_0 = 1{,}024$, $M = 26{,}624$, and $B \in \{64, 128, 256, 512\}$. Tables G1–G2 give the corresponding numbers of rounds. TDAL, PoH, UCB, BADGE, and CoreSet retain their respective model and training hyperparameters and share the candidate pool, initial set, and budget within each task and seed. Each setting uses five seeds.

Single perturbations. On IL-2, TDAL has the highest mean at $B = 16, 32, 64$. At $B = 128$, TDAL's mean decreases to 252.2 with greater variation across seeds, while PoH reaches 351.4. This setting provides only 15 rounds of experimental feedback and no initial LBO, illustrating performance under limited training experience.

Table G1: Batch size comparison at a fixed budget on IL-2. Values are final cumulative hits, reported as mean ± sample standard deviation over five seeds. The highest mean in each column is bold and the second highest is underlined.

Method	$B = 16$ $T = 120$	$B = 32$ $T = 60$	$B = 64$ $T = 30$	$B = 128$ $T = 15$
TDAL	$\mathbf{355.4 \pm 7.8}$	$\mathbf{367.4 \pm 3.0}$	$\mathbf{353.8 \pm 10.1}$	252.2 ± 59.5
PoH	$\underline{348.8 \pm 7.5}$	$\underline{347.8 \pm 6.0}$	$\underline{347.0 \pm 11.0}$	$\mathbf{351.4 \pm 8.0}$
UCB	336.6 ± 6.2	331.0 ± 10.2	340.0 ± 18.4	$\underline{333.2 \pm 15.3}$
BADGE	263.8 ± 3.6	237.0 ± 12.0	216.8 ± 9.3	205.8 ± 8.5
CoreSet	301.2 ± 8.8	298.4 ± 4.8	297.0 ± 5.8	297.2 ± 9.4

Double perturbations. On Simpson K562, TDAL stays in the top two at all four batch sizes, close to PoH. TDAL has the highest means at $B = 64, 256$, while PoH is slightly higher at $B = 128, 512$. TDAL's mean declines with larger batches, remaining above UCB, BADGE, and CoreSet. At a fixed query budget, larger batches leave fewer rounds for incorporating experimental feedback and training the acquisition function. Smaller batches provide more opportunities to update selection, while PoH remains competitive at larger batches.

Table G2: Batch size comparison at a fixed budget on Simpson K562. Values are final cumulative hits, reported as mean ± sample standard deviation over five seeds. The highest mean in each column is bold and the second highest is underlined.

Method	$B = 64$ $T = 400$	$B = 128$ $T = 200$	$B = 256$ $T = 100$	$B = 512$ $T = 50$
TDAL	**6868.2 ± 65.8**	6765.4 ± 102.5	**6669.2 ± 153.1**	6638.0 ± 81.9
PoH	6775.8 ± 41.5	**6771.0 ± 75.7**	6655.6 ± 78.8	**6670.8 ± 80.0**
UCB	5955.6 ± 52.5	5955.4 ± 46.0	5935.4 ± 50.3	5874.0 ± 25.6
BADGE	921.2 ± 27.9	932.8 ± 13.3	1006.8 ± 59.0	1015.8 ± 40.4
CoreSet	1827.0 ± 38.0	1693.6 ± 44.4	1522.2 ± 32.9	1377.6 ± 27.3

G.2 Computational Cost

On IL-2, we use $n_0 = 128$, $B = 32$, and the full budget $M = 2{,}048$, measuring active rounds 2–4 for each method with seeds 0–2. Each run has exclusive access to one NVIDIA A800 80 GB GPU and eight CPU cores, with runs for different seeds executed sequentially. Mean time per round excludes initialization and the first round, and includes selection and all model updates within the measured rounds. Peak host memory and reserved GPU memory are recorded over the same window, including persistent model and replay state.

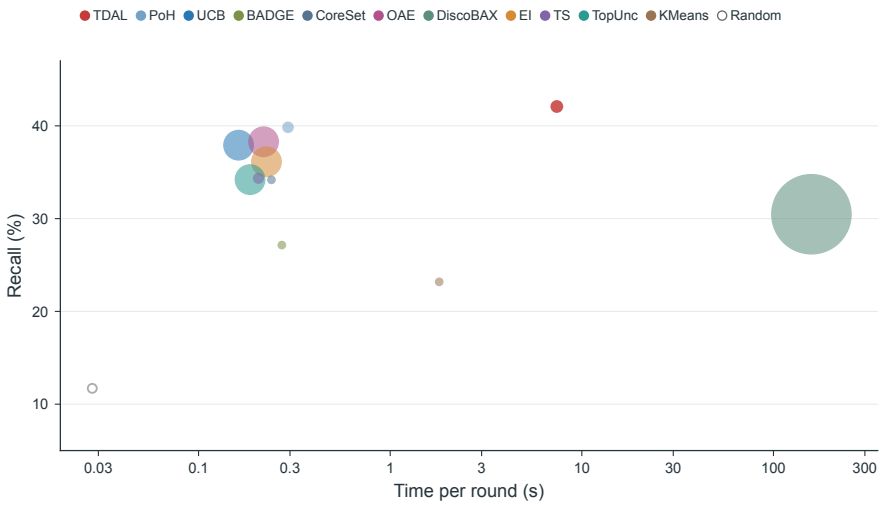


Figure G1: Discovery performance and computational cost on IL-2. The horizontal axis shows mean time per round on a logarithmic scale, and the vertical axis shows final hit recall for twelve methods. Circle area is proportional to peak reserved GPU memory. Random uses zero reserved GPU memory and is shown as a hollow circle. Time and memory are means over three seeds over rounds 2–4. Recall is the mean over five seeds in the full screening experiment of 60 rounds.

TDAL achieves 42.08% final recall on IL-2, with a mean time of 7.40 seconds per round, 1.12 GiB peak host memory, and 0.051 GiB peak reserved GPU memory (Figure G1). DiscoBAX takes 157.78 seconds per round, 21.3 times the TDAL runtime, and reserves 2.674 GiB of GPU memory.