# Can AI Scientist Agents Learn from Lab-in-the-Loop Feedback? Evidence from Iterative Perturbation Discovery

**Gilles Wainrib** [1] **Barbara Bodinier** [1] **Haithem Dakhli** [1] **Josep Monserrat** [1] **Almudena Espin Perez** [1] **Roberta Codato** [1] **John Klein** [1]

## Abstract

Recent work has questioned whether large language models (LLMs) can perform genuine in-context learning (ICL) for scientific experimental design, with prior studies suggesting that LLM-based agents exhibit no sensitivity to experimental feedback. We shed new light on this question by carrying out 800 independently replicated experiments on iterative perturbation discovery in Cell Painting high-content screening. We simulate a lab-in-the-loop setting offline, replaying feedback from pre-computed perturbation profiles in the JUMP Cell Painting dataset. We compare an LLM agent that iteratively updates its hypotheses using experimental feedback to a zero-shot baseline that relies solely on pretraining knowledge retrieval. Access to feedback yields a +53.4% increase in discoveries per feature on average ($p = 0.003$). To test whether this improvement arises from genuine feedback-driven learning rather than prompt-induced recall of pretraining knowledge, we introduce a random feedback control in which hit/miss labels are permuted. The performance gain disappears under random feedback, indicating that the observed improvement depends on the quality of the feedback signal ($+13.0$ hits, $p = 0.003$). We further examine how model capability affects feedback utilization. Upgrading from Claude Sonnet 4.5 to 4.6 reduces gene hallucination rates from $\sim$33%–45% to $\sim$3–9%, converting a non-significant ICL effect ($+0.8$, $p = 0.32$) into a large and highly significant improvement ($+11.0$, $p = 0.003$) for the best ICL strategy. These results suggest that effective in-context learning from experimental feedback emerges only once models reach a sufficient capability threshold.

[1]Owkin Inc.. Correspondence to: Barbara Bodinier <barbara.bodinier@owkin.com>.

*Accepted at the 2026 Workshop on Generative and Agentic AI for Biology (ICML 2026)*

## 1. Introduction

Large language models encode substantial biological knowledge from their training corpora, raising the prospect of LLM-powered agents that can guide scientific experimentation. BioDiscoveryAgent (Roohani et al., 2024) demonstrated that LLM agents can design genetic perturbation experiments, achieving improvements over Bayesian optimization baselines. However, a fundamental question remains contested: do these agents actually *learn* from experimental feedback, or do they merely leverage static prior knowledge?

Gupta et al. (2025) present striking evidence supporting the latter view. They compare BioDiscoveryAgent with a variant receiving randomly permuted labels, finding comparable performance. They conclude that "LLMs trained on next-token prediction and RLHF fail to perform in-context experimental design" and propose LLMNN, restricting LLMs to prior-based selection while delegating iterative updates to classical nearest-neighbor methods.

We revisit this question with a large-scale empirical study. Using the JUMP Cell Painting dataset, the largest public morphological profiling resource (Chandrasekaran et al., 2023), we conduct a large-scale benchmark comparing 6 agent architectures across 10 target features with 10 replicates each (800 total experiments). Our best feedback-enabled LLM agent achieves $+185\%$ improvement over random selection, compared to $+85\%$ for prior knowledge alone ($p = 0.003$) across features. The ICL effect is also significant within features on all 10 target features ($p < 0.01$), with gains ranging from $+3.7$ (F90) to $+27.4$ hits (F80) additional discoveries.

While the magnitude varies across features, the direction is consistently positive. We further show that model capability matters: upgrading from Claude Sonnet 4.5 to 4.6 boosts dramatically the ICL effect and reduces gene hallucination from $\sim$33%–45% to $\sim$3–9%, enabling the agent to translate its reasoning into effective experimental selections.

## 2. Related Work

**ICL for closed-loop optimization.** The application of LLMs within sequential optimization loops has accelerated rapidly. BioDiscoveryAgent (Roohani et al., 2024) demonstrated 21% improvement over Bayesian Optimization (BO) baselines for genetic perturbation design. Concurrent work applies similar approaches to chemical synthesis (Boiko et al., 2023; Zhang et al., 2025), materials discovery (Abhyankar et al., 2025), and drug design (Wang et al., 2025). LLM-guided Bayesian optimization (Ramos et al., 2023; Liu et al., 2024) uses ICL as an acquisition function, replacing hand-crafted kernels with learned priors. Chen et al. (2023) demonstrate closed-loop evolutionary optimization where LLM-generated candidates are evaluated and high-performing solutions serve as in-context exemplars for subsequent generations.

**Skepticism about ICL for experimental design.** Gupta et al. (2025) challenge whether LLMs genuinely learn from feedback. Their key experiment, comparing BioDiscoveryAgent with a variant receiving randomly permuted outcomes, finds comparable performance, suggesting improvements stem from prior knowledge rather than in-context adaptation. Falck et al. (2024) show that ICL violates exchangeability assumptions when context becomes sufficiently long. Wang et al. (2024) report limited out-of-distribution ICL generalization. These results collectively question whether ICL-based experimental design is a genuine capability or an artifact of encoded priors.

**Cell Painting and morphological profiling.** Cell Painting is a high-content imaging assay capturing cellular phenotypes through six fluorescent stains targeting distinct organelles (Bray et al., 2016). The JUMP consortium has generated profiles for over 116,000 chemical and 22,000 genetic perturbations (Chandrasekaran et al., 2023). Lu et al. (2025) introduce CellCLIP to align language and Cell Painting representations, but do not address sequential perturbation selection. To our knowledge, no prior work uses ICL-based perturbation optimization on Cell Painting data.

## 3. Methods

### 3.1. Data and Task

We use the CRISPR perturbation subset of the JUMP Cell Painting dataset, comprising $\sim$8,000 gene knockouts with morphological profiles across 4,672 CellProfiler features. These features are quantitative measurements of cell shape, texture, and intensity derived from imaging. For each target feature $y$, the hit-or-miss reward for a perturbation $x$ is $r_y(x) = \mathbb{1}[p_y(x) < 0.05]$, where $p_y(x)$ is the reported p-value of knockout $x$ on feature $y$. The task is to select genes to knockout to affect the target morphological feature. The objective is to maximise the number of hits among selected gene knockouts. For a given target feature, we iteratively ask the agent to select a batch of genes to knockout. Each experimental campaign consists of $T$=10 iterations with batch size $K$=100, totaling 1,000 perturbations from $\sim$8,000 available genes. Performance is measured by cumulative unique discoveries. This constitutes an offline simulation of a lab-in-the-loop, where hit/miss feedback is read from pre-computed JUMP profiles to mimic iterative experimental campaigns.

We evaluate on 10 target features spanning the feature difficulty spectrum (baseline hit rates $\sim$1–2%). Each approach is evaluated with 10 independent replicates, for 100 total campaigns per agent type (800 experiments across 8 approaches), providing sufficient statistical power for formal hypothesis testing. Step-by-step descriptions of each agent's prompts and task are provided in Appendices A, B, and C.

### 3.2. Agent Architectures

We compare three LLM agent variants, isolating the effect of experimental feedback. Our results use Claude Sonnet 4.6 (`claude-sonnet-4-6`) (Anthropic, 2025) and Sonnet 4.5 (`claude-sonnet-4-5-20250929`) as backbone LLM to study the effect of model capability:

**Zero-shot (Prior Knowledge Only).** The agent receives only the target feature description and list of available genes. It selects perturbations based solely on encoded biological knowledge. This agent receives *no information* about previous experimental outcomes, providing a clean measure of prior knowledge contribution.

**In-Context Learning from Experimental Feedback (ICL-EF).** The agent observes all previous experimental results: which genes were tested and whether they produced significant effects by exposing p-values. The prompt includes the full outcome history, enabling the agent to identify patterns. For example, "ATP6V1A was a hit, suggesting other V-ATPase subunits may also succeed." Additionally, to guide exploitation, the most frequent alphabetic prefixes among successful genes are appended to the prompt, see Appendix B for full details.

**In-Context Belief Revision from Experimental Feedback (ICBR-EF).** This variant extends the ICL-EF approach by incorporating a phenotypic signature (the top 10 most significantly perturbed features alongside the target feature) into the prompting context and a structured way to accumulate evidence. Phenotypic signatures allow the agent to reason not just about binary success, but about the specific morphological phenotypes induced by each perturbation, mimicking a Bayesian update of underlying biological mechanisms. To do this, the agent maintains an explicit *hypothesis register*,

i.e. beliefs about which biological mechanisms underlie the target phenotype. Each hypothesis is tracked using the following structure: a confidence level (High/Medium/Low) and a status (Active/Weakened/Abandoned). At each iteration, the current register is serialized into the LLM prompt. The LLM returns a JSON object encapsulating the *hypothesis register*, which replaces the current state. The register is entirely LLM-managed: the model itself decides when to weaken, create or abandon a hypothesis based on observed results. See Appendix C for full details.

### 3.3. Baselines

**Random.** Genes selected uniformly at random without replacement (10 replicates).

**GP-UCB.** This baseline is a Gaussian Process regression with Upper Confidence Bound acquisition, using a gene–gene similarity kernel derived from STRING protein–protein interaction scores (Szklarczyk et al., 2023) (10 replicates). We chose STRING PPI scores because they provide a principled, off-the-shelf measure of gene–gene functional similarity. This is a deliberately conservative baseline: richer kernels incorporating gene expression similarity, Gene Ontology semantic similarity, or learned embeddings could yield stronger GP-UCB performance and would narrow the gap with LLM agents.

**Random Feedback.** This baseline uses the same approach as ICL-EF, but the hit/miss labels in the experimental history are randomly permuted within each batch, breaking the correlation between gene identity and outcome while preserving the marginal hit rate per iteration (10 replicates, Sonnet 4.6 only).

### 3.4. Statistical significance of pairwise methods comparison across features

We treat target features as the unit of inference. For each method and feature, performance (cumulative unique discoveries) is first averaged across the 10 independent campaign replicates, which provide repeated measurements of the same feature under identical experimental conditions. Replicates are therefore used to obtain a stable estimate of feature-level performance but are not treated as independent samples in the final significance test. Pairwise method comparisons are then performed on the resulting paired feature-level scores using an exact two-sided sign-flip permutation test across features, which tests whether the mean feature-wise performance difference could arise under the null hypothesis of exchangeable method labels.

Because multiple pairwise method comparisons are performed, we control the false discovery rate (FDR) using the Benjamini–Hochberg procedure applied to the set of permutation p-values across all method pairs. Unless otherwise stated, reported p-values correspond to these Benjamini–Hochberg corrected permutation p-values. In addition, for further descriptive analysis, per-feature comparisons between methods using paired two-sided sign-flip permutation tests on the replicate-level performance differences within each feature are reported in Appendix D. This appendix also shows hierarchical bootstrap confidence intervals that can account for the combined sources of variability arising from both feature-to-feature differences and stochasticity across campaign replicates.

## 4. Results

### 4.1. In-Context Learning Improves Performance

Table 1 shows a consistent improvement when Sonnet 4.6 receives explicit success/failure feedback (ICL-EF), discovering an average of 29.3 hits compared to 20.4 for the zero-shot agent ($p = 0.003$). ICBR-EF, which receives extended phenotypic feedback, achieves the highest overall performance (31.4 mean discoveries) and the difference from ICL-EF is also statistically significant ($p = 0.006$).

Both feedback-enabled LLMs outperform the GP-UCB BO baseline (19.5, $p = 0.003$). This suggests that LLM-based semantic reasoning over biological literature (ingested during pretraining) can be more effective for this task than statistical regression over structural networks, though we note the GP-UCB kernel was deliberately conservative (see Subsection 3.3). All methods surpass the random baseline (11.0 hits). Further method performance gap significance analyses can be found in Appendix D.

### 4.2. ICL-EF vs ICBR-EF: qualitative comparison

While the performance gain with ICBR-EF over the simpler ICL-EF agent is modest (+2.1 discoveries), the ICBR-EF agent deploys sophisticated reasoning by leveraging both the phenotypic state inference and persistent hypothesis registries (see Section 4.5). Disentangling the individual contributions of phenotypic fingerprints and the hypothesis register is left to future work.

Qualitative analysis reveals two primary explanations for the limited increase in discovery rates with ICBR-EF compared to ICL-EF. First, the ICL-EF agent is highly efficient at exploitation. Once a few hits in major protein complexes (e.g., transcription machinery, nuclear envelope, or the ubiquitin-proteasome system) are found, the basic feedback mechanism allows ICL-EF to mine out virtually the entire complex before the 1,000-test budget is exhausted. Second, while the ICBR-EF agent's capacity to infer novel morphological connections (phenotypic clustering) successfully identifies genes across disjoint pathways, the absolute number of these pure, multi-pathway targets that are actually contained

*Table 1.* Cumulative discoveries across 10 target features. Features are labeled by their index in the CellProfiler feature matrix (F0–F90). Per-feature columns report mean ± std over 10 replicates. The "All" column reports the mean and std of the 10 per-feature means, capturing between-feature heterogeneity. Bold indicates best per feature. The Sonnet 4.5 panel shows that ICL-EF fails to produce significant gains compared to zero-shot ($+0.8$, $p = 0.32$), while Sonnet 4.6 yields large ICL-EF effects ($+8.9$, $p = 0.003$). Moreover, ICBR-EF exhibits marginal but consistent improvement over ICL-EF ($+2.1$, $p = 0.006$). The last row reports the total number of ground truth hits per feature in the full dataset, providing an upper bound on discoverable genes.

| Method | F0 | F10 | F20 | F30 | F40 | F50 | F60 | F70 | F80 | F90 | All |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Random | 18.3±3.7 | 15.1±4.2 | 12.8±3.0 | 11.9±1.9 | 10.7±4.1 | 10.9±2.8 | 11.8±3.3 | 5.8±3.0 | 8.7±2.0 | 4.0±1.5 | 11.0±4.0 |
| GP-UCB | 27.1±3.7 | 26.8±4.3 | 23.9±4.4 | 20.6±4.6 | 18.9±2.9 | 16.3±2.9 | 17.6±3.5 | 16.1±3.8 | 17.3±4.8 | 10.7±2.2 | 19.5±4.9 |
| Zero-shot (4.5) | 27.0±2.4 | 20.0±4.0 | 25.2±3.2 | 21.1±2.5 | 20.1±1.8 | 17.4±3.5 | 18.0±2.0 | 18.8±4.8 | 21.1±3.7 | 12.1±1.8 | 20.1±3.9 |
| ICL-EF (4.5) | 28.3±5.2 | 27.6±5.5 | 24.9±4.2 | 18.4±3.1 | 19.8±6.5 | 18.4±1.9 | 20.0±3.3 | 16.3±5.6 | 31.4±6.5 | 12.9±0.9 | 21.8±5.6 |
| Zero-shot (4.6) | 24.7±2.5 | 26.4±3.3 | 27.0±3.6 | 21.8±2.5 | 19.2±2.0 | 18.0±2.1 | 17.3±2.3 | 19.7±1.8 | 17.0±4.4 | 12.8±1.3 | 20.4±4.3 |
| Random FB (4.6) | 28.6±5.8 | 24.2±4.2 | 19.5±6.0 | 22.4±3.8 | 20.0±3.8 | 17.6±1.7 | 17.7±2.5 | 13.5±3.5 | 12.5±3.9 | 7.6±3.1 | 18.4±5.8 |
| ICL-EF (4.6) | 32.7±2.9 | 37.5±9.0 | **35.7**±2.1 | 26.8±2.7 | 27.6±3.3 | 21.7±2.2 | 21.2±1.7 | 32.7±2.2 | 43.1±3.6 | 14.3±1.0 | 29.3±8.2 |
| ICBR-EF (4.6) | **33.4**±3.8 | **48.3**±2.6 | 35.3±3.1 | **28.7**±3.2 | **28.8**±3.4 | **22.4**±1.6 | **21.7**±2.2 | **34.8**±2.0 | **44.4**±6.7 | **16.5**±0.8 | **31.4**±9.5 |
| Total hits | 152 | 124 | 93 | 88 | 85 | 90 | 87 | 51 | 65 | 33 | – |

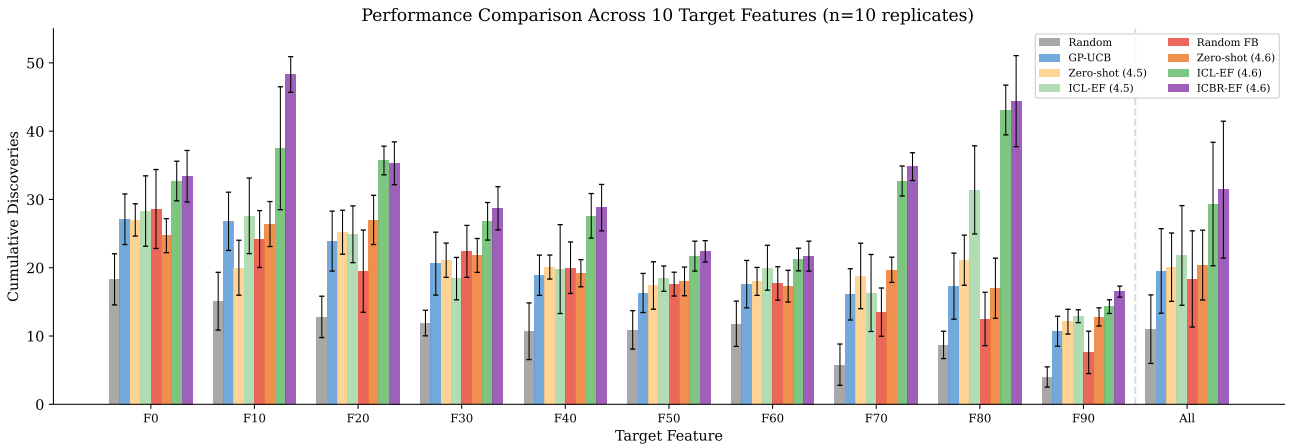



*Figure 1.* Performance comparison of six methods across 10 target features (10 replicates per condition). Error bars show standard deviation. Random FB sits below the zero-shot agent, while ICL-EF and ICBR-EF dominate all baselines.

within the physical screening library is small. Thus, the additional effort of complex phenotypic inference yields only a few additional discoveries per feature, creating a potential point of diminishing returns for purely in-context learning loops.

Figure 2 shows learning curves. The gap between ICL-EF and the zero-shot baseline widens progressively as observations accumulate, consistent with genuine in-context learning. By iteration 10, the ICL-EF agent has discovered 10 more genes on average than the zero-shot baseline. The cumulative discoveries achieved by ICL-EF and ICBR-EF remain on par until iteration 6, after which a contrast is visible but remains marginal. Notably, the curves have not plateaued by iteration 10, we leave the exploration of longer horizons for future work.

### 4.3. Feature-Level Heterogeneity

Table 1 and Figure 3 reveal that while ICL is consistently beneficial, its magnitude varies substantially. Compared to the random baseline, the ICBR-EF agent's largest relative improvements appear on F70 ($6.0\times$ more discoveries), F80 ($5.1\times$), and F10 ($3.2\times$), where gene family structure allows the agent to exploit discovered hits aggressively. Even on the most difficult features like F90, learning provides a significant benefit ($4.1\times$).

This heterogeneity has practical implications. The overall effect ($+20.4$ discoveries) is largest on features where gene family structure creates learnable correlations, and small (but still significant) on features with less exploitable structure.

The higher variance of both ICL-EF (std=8.2) and ICBR-EF (std=9.5) vs. zero-shot (std=4.3) reflects this: feedback creates both productive (gene family exploitation) and unproductive (misplaced obstinacy) behavioral modes, leading

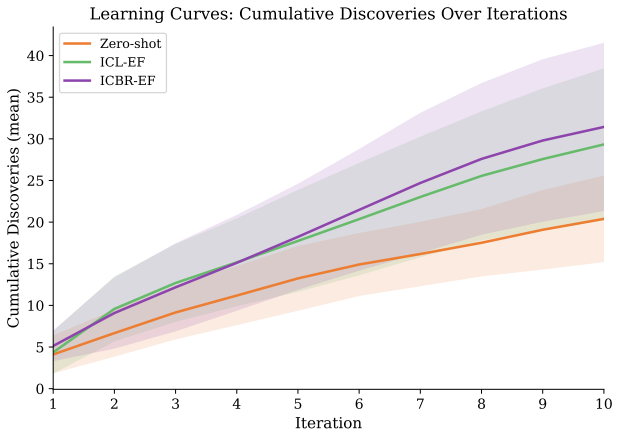



*Figure 2.* Cumulative discoveries over 10 iterations. ICL-EF progressively outpaces the zero-shot baseline as feedback accumulates.

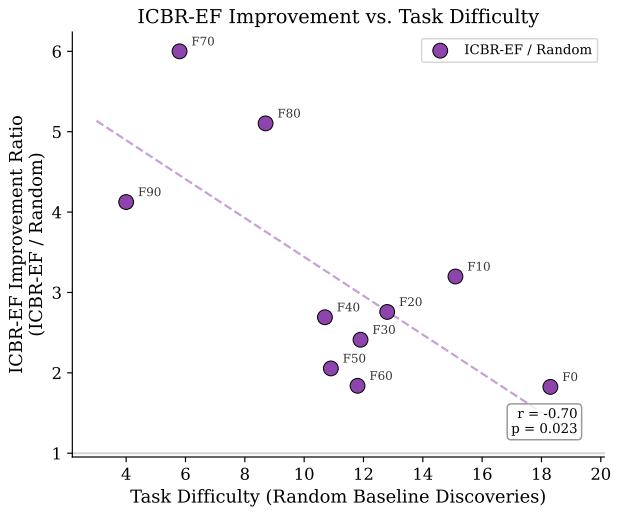



*Figure 3.* ICBR-EF improvement vs. Random baseline. Values above 1.0 indicate the ICBR-EF agent consistently outperformed random selection, achieving up to a $6.0\times$ improvement on F70.

to higher upside but also more variable outcomes.

### 4.4. Model Capability as a Critical Factor

Table 1 and Figure 4 present a direct comparison between Claude Sonnet 4.5 and 4.6 using the same agent architecture and prompts. Based on per-feature p-values and effect confidence intervals (CIs) presented in Appendix D, the contrast is stark: with Sonnet 4.5, the ICL effect is only $+1.7$ discoveries (99% effect CI $= [-2.03, +6.40]$, not significant), positive on 5/10 features. With Sonnet 4.6, the ICL effect is $+8.9$ (99% effect CI $= [+4.50, +15.33]$, significant) for ICL-EF, positive on 10/10 features, and significant at $p < 0.01$ on 7 of them. As expected, the effect is stronger with ICBR-EF (99% effect CI $= [+5.81, +17.93]$, significant), positive and significant on 10/10 features.

One measurable mechanism is **gene hallucination**. By parsing the LLM's raw completions against the valid gene library ($\sim$8,000 JUMP genes), we measure the fraction of proposed genes that do not exist in the screening library:

- **Sonnet 4.5**: 45.3% hallucination rate, 39–50% of test slots filled by random fallback (interquartile range across replicates);
- **Sonnet 4.6**: 9.1% hallucination rate, 3–11% random fallback (interquartile range across replicates).

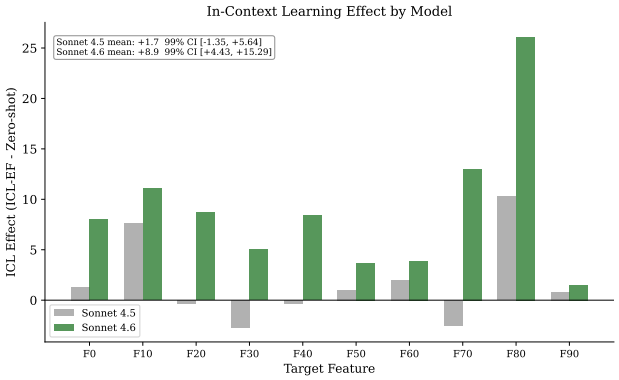
Importantly, the "hallucinated" genes are not random strings, i.e. they are real biological genes (e.g., EXOSC1, KDM3A, MED18) from the correct gene families, but absent from the $\sim$8,000-gene JUMP screening library. The failure is one of *instruction-following*: the model ignores the provided list of available (untested) genes and instead proposes genes from its own biological knowledge. This worsens over iterations (from $0\%$ at iteration 1 to $\sim 60\%$ at iteration 6 plateauing at this level in the remaining iterations) as the untested list shrinks and the model increasingly defaults to its priors. Sonnet 4.6's improvement reflects better instruction-following. It reliably constrains its output to the provided gene list. The model upgrade produces a $\sim 5\times$ reduction in out-of-library proposals and a corresponding $34\%$ increase in absolute discoveries (from 21.8 to 29.3 for ICL-EF).

However, a model upgrade changes multiple capabilities simultaneously (world knowledge, reasoning quality, instruction adherence, hallucination rate, etc.) so the improvement cannot be attributed to hallucination reduction alone. The Sonnet 4.6 model may also have better biological priors or better ability to process long contexts. What we can conclude is that sufficient model capability is necessary for ICL to manifest: studies using earlier or weaker models may have failed to observe ICL partly because hallucination prevents the model from executing its reasoning.

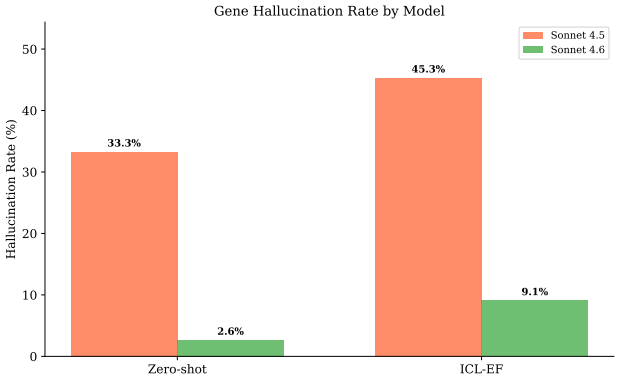
### 4.5. Behavioral Analysis: The Evolution of Search Strategies

Analysis of the agents' reasoning traces reveals exactly how experimental feedback and phenotypic context drive the stepwise jumps in performance. An in-depth analysis is proposed in Appendices A to C. Overall, we identify a three-tiered evolution in search strategy:

**Zero-shot vs. ICL-EF: From Static Priors to Pathway Exploitation.** The zero-shot agent relies entirely on its biological priors, making static best-guesses (e.g., broadly nominating "actin regulators" or "kinases"). It cannot adapt.




*(a)* Per-feature ICL effect (ICL-EF − Zero-shot) for Sonnet 4.5 vs. 4.6. With 4.5, ICL is inconsistent (5/10 positive); with 4.6, it is universally positive.




*(b)* Gene hallucination rate drops from ∼33%–45% to ∼3–9%.

*Figure 4.* Effect of model capability on ICL and hallucination.

In contrast, the ICL-EF agent uses binary hit/miss feedback to confirm an initial guess, and then aggressively pivots. If it discovers a single hit in the SWI/SNF complex, it immediately targets the remaining untested members of that specific complex. Semantic analysis of completion logs shows ICL-EF uses terminology related to targeted "complex/family/subunit" exploitation an average of 7.9 times per iteration, versus 2.3 for the zero-shot agent (a 3.4× higher rate consistent with active exploitation of experimental feedback). This allows it to systematically mine horizontal literature connections that zero-shot approaches cannot see.

**ICL-EF vs. ICBR-EF: From Pathway Exploitation to Phenotypic State Inference.** While powerful, ICL-EF is fundamentally limited to proposing genes that share a known literature or structural relationship to a confirmed hit. The ICBR-EF agent transcends this limitation. Because its prompt includes a rich, multidimensional phenotypic fingerprint (the top 10 most significant feature perturbations for each tested gene), it performs *phenotypic state inference*. Rather than just grouping genes by name, it clusters hits by their multi-dimensional cellular effects.

For example, on F80, the ICBR-EF agent recognizes that hits spanning disjoint functional pathways (e.g., DNA damage repair, cyclin-CDK complexes, and mitotic kinases) all produce identical perturbations in "angular second moment" and "nuclear texture." The agent correctly deduces these diverse targets converge on the same underlying physiological state: mitotic arrest. Semantic analysis confirms this strategic shift: the ICBR-EF agent explicitly references phenotypic markers and cellular states 22.6 times per iteration (compared to just 0.7 times for binary ICL-EF). By clustering genes by their *effects* rather than their *names*, the ICBR-EF agent can jump across disconnected biological pathways to discover novel, non-obvious hits.

**Stateful Tracking vs. Reactive Prompting.** Finally, tracing the evolution of the ICBR-EF agent's *hypotheses register* reveals a critical architectural advantage. While both agents receive the cumulative history of all tested genes and their outcomes, only the ICBR-EF agent maintains an explicit theory layer on top of this history. The ICL-EF agent re-derives its strategy from scratch at each iteration: it sees the full observation log but has no mechanism to preserve the *reasoning* behind previous selections. Over 10 iterations, the interpretation of early hits can drift as newer results dominate the prompt, and there is no way to mark a line of inquiry as exhausted. The ICBR-EF agent, by contrast, carries forward a structured hypotheses register across iterations, explicitly tracking the lifecycle of its theories (tagging them as *Active*, *Confirmed*, or *Weakened*) alongside explicit justifications (e.g., "*PSMB8,9,10 (immunoproteasome) are MISSes, confirming specificity to constitutive proteasome*"). This persistent reasoning scaffold prevents interpretive drift and allows the agent to systematically close out search spaces rather than revisiting them. Ultimately, this combination of phenotypic state inference and persistent hypothesis tracking drives the ICBR-EF agent's superior performance.

### 4.6. Reconciling with Prior Work

Why do we observe significant ICL where Gupta et al. (2025) report none? To directly test their hypothesis, i.e. that agents spuriously benefit from the prompt's length rather than its factual content, we run a **Random Feedback** (Random FB) control across all 10 target features (10 replicates each, 100 experiments total). In this condition, the agent receives an identically structured prompt containing the full experimental history, but the hit/miss labels are randomly permuted among the tested genes within each batch. This scheme directly tests whether the agent uses the *content* of feedback (which gene was a hit) rather than its *structure* (how many hits occurred).

The results do not support the (pretraining) knowledge retrieval hypothesis. Averaged across all 10 features, the zero-shot agent discovers 20.4 hits. The Random FB agent,

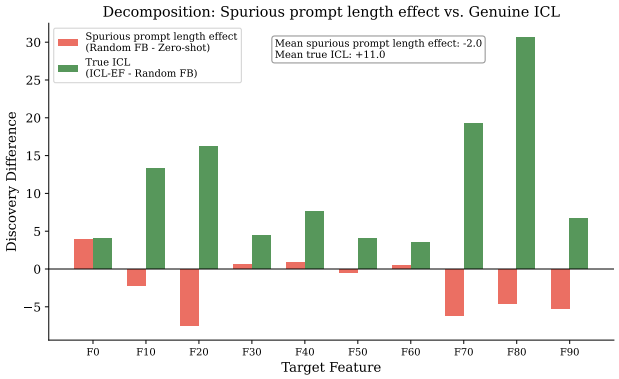



*Figure 5.* Decomposition of the learning effect into memory jogging (Random FB − Zero-shot, red) and genuine ICL (ICL-EF − Random FB, green) per feature with Sonnet 4.6. The spurious prompt length effect is negative on 6/10 features; genuine ICL is positive on all 10.

receiving permuted outcomes, discovers only 18.4 hits, on par or worse than no feedback at all ($-2.0$, $p = 0.15$ not significant). A true feedback agent such as ICL-EF discovers 29.3 hits, a $+10.9$ improvement over Random FB ($p = 0.003$). ICBR-EF discovers 31.4 hits, a $+13.0$ improvement over Random FB ($p = 0.003$).

Figure 5 reveals that the randomization effect is negative on 6/10 features and significantly harmful on 4 (F20, F70, F80, F90). This reveals a notable asymmetry: random feedback actively harms the agent's reasoning. When the agent receives false positive signals, it pursues unproductive gene families (a form of misplaced obstinacy effect), wasting its budget on misleading leads. On the hardest features (F70, F80, F90), where exploitable structure is sparse, random feedback reduces discoveries by $30-40\%$ relative to no feedback. The agent is not merely reading the prompt but is acting on the specific factual content, and false content produces systematically worse outcomes than no content at all.

We identify several factors explaining the divergence from Gupta et al. (2025):

1. **Model capability.** As shown in Section 4.4, ICL is not significant with Sonnet 4.5 ($p = 0.55$) but highly significant with Sonnet 4.6 ($p = 0.003$). The $5\times$ reduction in hallucination enables the agent to execute its reasoning. Gupta et al. primarily evaluated open-source and earlier Claude models, which may have lacked the requisite fidelity.

2. **Dataset structure.** JUMP Cell Painting data contains rich gene family correlations that create learnable structure for ICL exploitation.

3. **Statistical power.** With 10 replicates per condition (800 total experiments), we have sufficient power to detect the effect.

## 5. Future Work

ICL has inherent limitations: the agent must fit all observations into a fixed context window, learning is ephemeral (reset each session), and the agent is never explicitly trained on experimental outcomes. Reinforcement learning offers a natural extension. The JUMP dataset provides a simulated environment for RL training (Reinforcement Learning from Verifiable Rewards), and the gene family correlation or misplaced obstinacy patterns we observe map directly onto the exploration–exploitation trade-off that RL is designed to optimize. An RL-trained agent could also learn transferable strategies across feature types. We leave this direction for future investigation.

## 6. Discussion

**Practical implications.** Our results suggest that LLM agents can provide benefits for iterative experimental design when model capability is sufficient. The magnitude of improvement varies across features but the effect is consistently positive in our setting. Practitioners should expect the largest gains where gene family structure creates learnable correlations. Model capability matters: upgrading from Sonnet 4.5 to 4.6 reduced hallucination from $\sim$33%–45% to $\sim$3–9%, corresponding to a 34% increase in discoveries for the ICL-EF agent.

**Cost considerations.** Each 10-iteration campaign requires 10 LLM API calls (one per iteration), consuming on average $\sim$410,000 input tokens and $\sim$19,000 output tokens. At current Claude Sonnet pricing ($3/MTok input, $15/MTok output), this yields a cost of $\sim$$1.50 per campaign ($1.41 for zero-shot, $1.46 for ICL-EF, $1.77 for the ICBR-EF agent). The ICBR-EF agent's higher cost reflects its longer outputs (33K vs. 14K tokens per campaign). For the 600 LLM-based campaigns in this study, total API cost was approximately $900. At $\sim$30 discoveries per campaign (ICL-EF), the cost per discovery is roughly $0.05. GP-UCB requires no API calls but does require precomputing the gene–gene kernel matrix. Batch API pricing (50% discount) would reduce LLM costs to $\sim$$450 total.

**Data contamination.** Frontier LLMs are trained on large web corpora that may include papers discussing the JUMP dataset, Cell Painting protocols, or specific gene–phenotype associations measured in JUMP. This means the LLM's "prior knowledge" may partially derive from having seen analyses of the same dataset during pretraining, rather than from general biological knowledge alone. While our Random Feedback control confirms that the ICL effect depends

on the *content* of feedback (not just static priors), the absolute performance of the zero-shot agent (prior-only) may be inflated by data contamination. This concern applies broadly to any LLM-based scientific agent evaluated on public datasets.

**Relationship to LLMNN.** Our work argues against the interpretation of Gupta et al. (2025) that LLMs cannot learn from feedback but does not directly evaluate their proposed LLMNN method, which delegates iterative updates to a classical nearest-neighbor algorithm while using the LLM only for prior-based initialization. A direct comparison between our ICL approach and LLMNN on the same benchmark would be informative: LLMNN may offer advantages in settings where model hallucination is high or feedback is noisy, while pure ICL may be preferred when model capability is sufficient. We leave this comparison for future work.

**Summarized limitations.** (1) We evaluate only two Claude models (Sonnet 4.5 and 4.6); results may not generalize to open-source or other proprietary models, and the improvement between versions highlights strong model-dependence. The Sonnet 4.5 $\rightarrow$ 4.6 comparison is confounded: the upgrade changes world knowledge, reasoning quality, instruction adherence, and hallucination simultaneously, so the ICL improvement cannot be attributed to any single factor. (2) All experiments use pre-computed p-values from the JUMP dataset. Real experimental campaigns involve batch effects, measurement noise, and variable hit rates across plates that could substantially alter ICL dynamics. The clean, binary, noise-free feedback in our setup may be particularly favorable for ICL; real experiments would introduce stochasticity that could erode the advantage. How robust ICL is to noisy or delayed feedback remains an open question. (3) Cell Painting with CellProfiler features represents one assay type with specific correlation structures. Future work is warranted to evaluate whether these findings transfer to other assay types, such as transcriptomics readouts. (4) Our Random FB control uses within-batch permutation; across-iteration permutation would test a different confound (see Section 4.6). (5) The GP-UCB baseline uses a conservative STRING PPI kernel which may hamper performances and precludes a definitive conclusion on the comparison between BO and LLM agents. (6) We do not evaluate the LLMNN hybrid method proposed by Gupta et al. (2025), limiting our ability to compare pure ICL against their proposed solution.

## 7. Conclusion

Across 800 independently replicated experiments, we find that a frontier LLM agent can learn from experimental feedback for iterative perturbation discovery. The feedback-enabled agent achieves 31.4 mean discoveries ($+185\%$ over random), with the ICL effect significant at $p = 0.003$. A random feedback control shows the learning effect is attributable to ICL ($+13.0$ for our best agent over permuted feedback, $p = 0.003$), with random feedback actually harming performance. Model capability is a prerequisite: the ICL effect is non-significant with Sonnet 4.5 (45.3% hallucination) but highly significant with Sonnet 4.6 (9.1% hallucination). These results are specific to one dataset (JUMP Cell Painting), one assay type, and two proprietary models using retrospective data. Whether these findings generalize to other experimental domains, noisier feedback settings, and open-source models remains to be established.

## Impact Statement

This work studies AI-guided iterative experimental design in high-content screening, a setting where autonomous agents select biological perturbations for testing. While our experiments are conducted retrospectively on a public dataset, the methods we develop are intended to operate in closed-loop laboratory settings. We encourage practitioners deploying LLM agents to guide real-world biological experiments to maintain human oversight of agent-proposed experiments and to validate key findings with independent assays. More broadly, accelerating biological discovery through AI could yield societal benefits in drug development and basic science, but also underscores the need for responsible deployment practices as the scope of autonomous experimentation expands.

## References

Abhyankar, N., Kabra, S., Desai, S., and Reddy, C. K. Accelerating materials design via llm-guided evolutionary search. *arXiv preprint arXiv:2510.22503*, 2025.

Anthropic. Claude sonnet 4 model family. https://www.anthropic.com/claude, 2025. Model identifiers: `claude-sonnet-4-5-20250929` and `claude-sonnet-4-6`. Accessed: 2025-11-25.

Boiko, D. A., MacKnight, R., Kline, B., and Gomes, G. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

Bray, M.-A., Singh, S., Han, H., Davis, C. T., Borgeson, B., Hartland, C., et al. Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nature Protocols*, 11(9):1757–1774, 2016.

Chandrasekaran, S. N., Ceulemans, H., Boyd, J. D., Carpenter, A. E., and Kalinin, A. Jump cell painting dataset: morphological impact of 136,000 chemical and genetic perturbations. *bioRxiv*, 2023.

Chen, A., Dohan, D., and So, D. Evoprompting: Language models for code-level neural architecture search. *Advances in neural information processing systems*, 36: 7787–7817, 2023.

Falck, F., Wang, Z., and Holmes, C. Is in-context learning in large language models bayesian? a martingale perspective. *arXiv preprint arXiv:2406.00793*, 2024.

Gupta, R., Hartford, J., and Liu, B. Llms for bayesian optimization in scientific domains: Are we there yet? In *Findings of EMNLP 2025*, 2025.

Liu, T., Astorga, N., Seedat, N., and van der Schaar, M. Large language models to enhance bayesian optimization. *arXiv preprint arXiv:2402.03921*, 2024.

Lu, M., Weinberger, E., Kim, C., and Lee, S.-I. Cellclip–learning perturbation effects in cell painting via text-guided contrastive learning. *arXiv preprint arXiv:2506.06290*, 2025.

Ramos, M. C., Michtavy, S. S., Porosoff, M. D., and White, A. D. Bayesian optimization of catalysts with in-context learning. *arXiv preprint arXiv:2304.05341*, 2023.

Roohani, Y., Lee, A., Huang, Q., Vora, J., Steinhart, Z., Huang, K., Marson, A., Liang, P., and Leskovec, J. Biodiscoveryagent: An ai agent for designing genetic perturbation experiments. In *ICLR 2025*, 2024.

Szklarczyk, D., Kirsch, R., Koutrouli, M., Nastou, K., Mehryary, F., Hachilif, R., Gable, A. L., Fang, T., Doncheva, N. T., Pyysalo, S., et al. The string database in 2023: protein–protein association networks and functional enrichment analyses for any sequenced genome of interest. *Nucleic acids research*, 51(D1):D483–D489, 2023.

Wang, Q., Wang, Y., Wang, Y., and Ying, X. Can in-context learning really generalize to out-of-distribution tasks? *arXiv preprint arXiv:2410.09695*, 2024.

Wang, Z., Wen, Y., Pattie, W., Luo, X., Wu, W., Hu, J. Y.-C., Pandey, A., Liu, H., and Ding, K. Polo: Preference-guided multi-turn reinforcement learning for lead optimization. *arXiv preprint arXiv:2509.21737*, 2025.

Zhang, Y., Han, Y., Chen, S., Yu, R., Zhao, X., Liu, X., Zeng, K., Yu, M., Tian, J., Zhu, F., et al. Large language models to accelerate organic chemistry synthesis. *Nature Machine Intelligence*, pp. 1–13, 2025.

# A. Zero-shot Agent: Prompting Strategy and Execution Trace

The zero-shot agent is our zero-shot LLM baseline. Unlike ICL-EF and ICBF-EF, it receives *no experimental feedback*: previous test results are recorded internally for evaluation purposes but are never included in the prompt. Every iteration, the LLM selects the next batch of 100 genes using only its biological prior knowledge and the list of untested gene names.

## A.1. System Prompt

The system prompt is fixed across all iterations:

```
You are an expert computational biologist AI agent discovering gene perturbations
that induce significant cellular responses.

TASK: Find gene knockouts where p-value for 'Nuclei_Intensity_MeanIntensity_AGP' is < 0.05.

## Strategy - Use Biological Knowledge

You must select genes based ONLY on your biological knowledge:

1. Gene families: Genes with similar names often share functions
   - Transporter families
   - Complex subunit families
   - Modifier enzyme families

2. Functional categories relevant to nuclear phenotypes:
   - Chromatin regulators
   - Nuclear envelope proteins
   - Transcription machinery
   - DNA repair genes

3. Known phenotype associations:
   - Think about which gene categories typically affect Nuclei features

## Available Data:
- Total perturbations: 7975
- Target: Nuclei_Intensity_MeanIntensity_AGP

IMPORTANT: You have NO information about previous test results.
Select genes based purely on biological reasoning.
```

## A.2. Per-Iteration User Prompt

At each iteration $t$, the user prompt contains three components and nothing else.

**1. Iteration counter.** A single line `Iteration` $t$ provides context on how far into the experiment the agent is.

**2. Already-tested and untested gene lists.** The full list of already-tested gene IDs (to prevent repeats) and the full list of untested IDs are provided verbatim.

**3. Self-check and output format.** Identical to ICL-EF and ICBF-EF:

```
## Iteration 2

Select 100 genes to test from the untested perturbations below.

Use your biological knowledge to prioritize genes most likely to affect
'Nuclei_Intensity_MeanIntensity_AGP'.

ALREADY TESTED PERTURBATIONS (DO NOT SUGGEST THESE):
ARID1A, ATM, ATR, ...   [100 gene names]

UNTESTED PERTURBATIONS (7875 remaining):
```

```
A2M, A3GALT2, A4GALT,  ...   [7875 gene names]

CRITICAL REQUIREMENT:
Before returning your final JSON, you MUST explicitly double-check your proposed
genes ONE BY ONE. For each gene you want to propose, verify:
1. It is EXACTLY present in the UNTESTED list above.
2. It is NOT present in the ALREADY TESTED list.
If you realize a gene violates these rules, propose a replacement.

After your verification, return the final 100 genes strictly inside a JSON code block:
```json
["GENE1", "GENE2", "GENE3", ...]
```
```

### A.3. Execution Trace

We trace the agent on `Nuclei_Intensity_MeanIntensity_AGP` / F0 (152 significant genes out of 7,975), with batch size 100 and 10 iterations (1,000 total tests).

**Iteration 1 — Biology-prior initialization.** With no feedback available, the LLM first interprets the target feature: the AGP channel (actin, Golgi, plasma membrane) measures cytoskeletal staining that can overlap with nuclei, so nuclear intensity in this channel is expected to be sensitive to nuclear envelope/lamina integrity, chromatin compaction, cell cycle state, histone modifications, and transcription factor localisation. It then selects 100 genes organised by these mechanisms: chromatin regulations and remodeling (`HDAC1-3`, `EZH2`, `KDM1A`, `KDM5B`, `SETD2`, `CHD4`), nuclear transport and nuclear envelope integrity maintenance (`LMNA`, `LBR`, `NUP98`, `NUP214`), cell cycle (`CDK1/2/4`, `CCNA2`, `PLK1`, `AURKA/B`), DNA-damage checkpoints (`ATM`, `ATR`, `BRCA1/2`, `PCNA`), and transcriptional regulation (`MYC`, `EP300`, `TP53`, `MDM2`). It also explicitly verifies each selected gene against the untested pool before outputting the final list. Hits: `CHD4, PCNA, MYC, EP300, MDM2` (5 of 100).

**Iteration 2 — Continued prior-based selection.** Still operating without experimental feedback, the agent broadens its prior-based sweep across various mechanistic categories: remaining KDM/HAT/HDAC family members (`KDM1B`, `KDM5A/C/D`, `KDM7A`, `KAT6A/B`, `KAT7/8`, `HDAC4-9`), SET-domain writers (`EZH1`, `EHMT1`, `SETD7`, `SETDB2`), nuclear transport (`KPNA1/2`, `KPNB1`, `XPO1`), extended CDK family (`CDK5-9`, `CDK12`), transcription factors (`FOXM1`, `FOXO1/3`, `RUNX1/2`, `STAT3`), chromatin readers and remodellers (`BRD2-4`, `PHF` proteins, `UHRF1/2`, `TRIM28/33`), CHD-family remodellers (`CHD1/2/7/8`), and signalling (`MTOR`). Hits: `XPO1, TRRAP, UHRF1, KMT2B` (4 of 100).

**Iterations 3–10 — Exhaustive prior coverage without adaptation.** Across the remaining eight iterations the agent's strategy does not evolve: lacking any feedback, it restarts the same biological reasoning from scratch each round, re-identifying the same four to five categories (transcription factors, chromatin remodellers, nuclear transport, cytoskeletal and kinase regulators) and simply advancing through the untested pool within those categories. Hit rates remain low but non-zero (iter. 3: 4; 4: 1; 5: 3; 6: 1; 7: 6; 8: 2; 9: 1; 10: 1), reflecting a gradually depleted prior rather than learned focus. The spike in iteration 7 arises incidentally when the agent reaches RNA-polymerase and Mediator subunits (`POLR2A`, `MED21`, `MED27`) by exhausting adjacent chromatin-remodelling families, not by inference from earlier hits. Because results are never fed back, the agent cannot learn that its hits are enriched in the transcription-coactivation cluster and therefore fails to efficiently target related machinery in later iterations.

**Summary.** After 1,000 tests the agent discovers 28 of 152 significant genes (18.4% recall): `CHD4, PCNA, MYC, EP300, MDM2, XPO1, TRRAP, UHRF1, TRIM28, KMT2B, SUPT6H, TAF1, TAF5, AURKC, CBFA2T3, PSMC2, IRF3, RAC1, POLR2A, MED21, MED27, RBM14, PRPF19, CAND1, ZBTB45, ZSCAN9, ATP6V0D1, KIF11`. Hit rate drops after iteration 3 as the agent exhausts its highest-confidence prior categories, then partially recovers in iteration 7 when it independently reaches new gene families. Because results are never fed back, the agent cannot exploit for example the CHD4/EP300 hits from iteration 1 to suggest TAF1 in iteration 2 which is also an epigenetic regulator gene.

## B. ICL-EF Agent: Prompting Strategy and Execution Trace

The ICL-EF (In-Context Learning with Experimental Feedback) agent is the core feedback-enabled agent in our study. At each iteration the LLM receives the full outcome history of all tested genes (hit/miss and target p-value) and uses its

biological prior knowledge to select the next batch. To guide the exploitation of successful hits, we also provide gene name patterns. We extract the leading alphabetic characters (up to four) of each successful gene name, count how often each prefix recurs across all hits, and expose the five most frequent prefix–count pairs to the LLM so that it can immediately identify which gene families are over-represented among successes and prioritize untested members of those families in the next batch.

### B.1. System Prompt

The system prompt primes the model for gene-family and pathway reasoning and is fixed across all iterations:

```
You are an expert computational biologist AI agent discovering gene perturbations
that induce significant cellular responses.

TASK: Find gene knockouts where p-value for
      'Nuclei_Intensity_MeanIntensity_AGP' is < 0.05.

KEY STRATEGY - Use your biological knowledge:
1. Gene families: Genes with similar names often have related functions
2. Pathways: If a gene in a pathway shows effect, test other genes in same pathway
3. Protein complexes: Subunits of same complex often show similar phenotypes
4. Functional categories: Transcription factors, kinases, transporters may cluster

PATTERN RECOGNITION:
- When successful genes share naming patterns or prefixes, prioritize similar names
- Look for gene family numbers - if one member works, test other family members
- Consider biological process: if mitochondrial genes work, test more mitochondrial
  genes

AVAILABLE DATA:
- Total perturbations: 7975
- Features: 100 CellProfiler measurements (p-values)
- Target: Nuclei_Intensity_MeanIntensity_AGP

Remember: You're looking for genes affecting nuclear morphology/intensity.
Think about:
- Chromatin regulators (histones, KMTs, KDMs, CHDs)
- Nuclear envelope proteins
- Transcription machinery (TAFs, POLRs)
- Signaling to nucleus
```

### B.2. Per-Iteration User Prompt

The user prompt is assembled from four components at each iteration $t$.

**1. Observation summary.** All tested genes are listed with a hit/miss label and their target p-value:

```
TESTED PERTURBATIONS: 100
Successes: 8, Failures: 92

RESULTS:
  CHD1 :  MISS (p=0.6244)
  CHD2 :  MISS (p=0.6439)
  ...
  CHD4 :  HIT  (p=0.0004)
  KMT2  : HIT  (p=0.0024)
  TRRAP:  HIT  (p=0.0308)
  MYC. :  HIT  (p=0.0070)
  ...
```

**2. Gene prefix summary.** The framework extracts the four leading alphabetic characters of each hit gene name, counts prefix frequencies, and appends the top-5 as a compact hint:

```
SUCCESSFUL GENE PATTERNS: CHD(1), KMTB(1), EP(1), TAF(1), POLR(1)
```

**3. Already-tested and untested gene lists.** Full ID lists so the agent can verify its proposals do not re-test any gene.

**4. Strategy and self-check.** Instructs the agent to exploit observed hits, explore novel categories, and verify each proposed gene before returning the final list:

```
STRATEGY FOR SELECTION:
1. EXPLOIT: Find genes with similar names/prefixes to successful ones
2. EXPLOIT: Test genes in same pathway/complex as successes
3. EXPLORE: Include some genes from untested categories for diversity

CRITICAL REQUIREMENT:
Before returning your final JSON, you MUST explicitly double-check your proposed
genes ONE BY ONE. For each gene you want to propose, verify:
1. It is EXACTLY present in the UNTESTED list above.
2. It is NOT present in the ALREADY TESTED list.
If you realize a gene violates these rules, propose a replacement.

After your verification, return the final 100 genes strictly inside a JSON
code block:
```json
["GENE1", "GENE2", "GENE3", ...]
```
```

### B.3. Execution Trace

We trace the agent on `Nuclei_Intensity_MeanIntensity_AGP` / F0 (152 significant genes out of 7,975), batch size 100, 10 iterations (1,000 total tests).

**Iteration 1 — Prior-guided initialisation.** With no observations yet, the agent draws on its biological prior and explicitly lists nine mechanistic categories: chromatin remodelers (`CHD1/2/4/7/8`, full SWI/SNF complex); histone methyltransferases and demethylases (`KMT2A/B/C/D`, `KMT5A/B`, `SUV39H1/2`, `SETD2/DB1`, `KDM1A/2A/B/4A/5A/B/C`); histone acetyltransferases and deacetylases (`KAT2A/B/5/6A/7/8`, `EP300`, `CREBBP`, `HDAC1-6/8`); nuclear envelope (`LMNA`); transcription machinery (`TAF1/2/4/6/7/10`, `POLR2A/B/C`, `TRRAP`, `MYC`); DNA-damage response (`BRCA1/2`, `ATM`, `ATRX`, `PARP1`, `TOP2A/B`, `PCNA`); Polycomb/trithorax (`SUZ12`, `EED`, `RING1`, `BMI1`, `JARID2`, `PHF1/8`, `CBX2/7/8`); NuRD/co-repressor complexes (`RBBP4/7`, `MTA1/2`, `SIN3A/B`, `NCOR1/2`); and key nuclear signalling (`BRD2/4`, `MEN1`, `RUVBL1/2`). Hits: `CHD4, KMT2B, EP300, TAF1, POLR2A, PCNA, TRRAP, MYC` (8 of 100).

**Iteration 2 — Gene-family exploitation begins.** The agent explicitly analyses all eight hits and derives twelve expansion directions: from `TAF1` and `POLR2A` it targets remaining TAF subunits (`TAF3/5/8/9B/11-15`), all remaining POLR2 subunits (`POLR2D-L`), Pol II CTD kinases (`CDK7/8/9/12/13`, cyclins `CCNT1/2`, `CCNH`), Mediator subunits (`MED1/4/12/14/17/23`), general transcription factors (`TBP`, `GTF2B/E1/F1/H1`, `YY1`, `SP1/3`), and elongation factors (`SUPT6H`, `SUPT16H`, `SNW1`, `TCERG1`); from `TRRAP` and `EP300` it adds TRRAP-associated HAT complexes (`JADE1/2/3`, `BRD3/8`); from `MYC` it tests `MYCN`, `MAX`, and `MXD1`; from `CHD4` it expands to `CHD5/6/9/1L`; from `KMT2B` it adds `KMT2E`, `KMT5C`, `DOT1L`, `SETD7`, `SMYD2/3`, `EHMT1/2`; and from `PCNA` it tests replication-fork partners (`RFC1-5`, `POLE`, `POLD1`). Hits: `TAF5, SUPT6H, POLR3B` (3 of 100).

**Iterations 3–10 — Sustained exploitation with adaptive pivots.** At the start of every iteration the agent re-reads all accumulated hits and derives the next batch by explicit pattern analysis, so its strategy genuinely evolves with feedback. Iterations 3–5 continue systematic transcription-machinery sweeps seeded in iteration 2: Pol III subunit expansion yields `POLR3K` (iter. 3) alongside Mediator hits `MED21/27`; nuclear-export and TFIIIC probes return `XPO1` and `GTF3C5` (iter. 4); DNA-replication/repair exploration adds `DNA2` and `UHRF1` (iter. 5). Iteration 6 marks an inflection: the `TRIM28` and `MDM2` hits signal ubiquitin-pathway involvement, and the agent pivots accordingly in iteration 7 to the CUL3/RBX1/UBE2M E3-ligase complex and the PSMC2 proteasome subunit, yielding 6 hits in a single batch. Iterations 8–9 exploit newly identified family anchors — ZSCAN zinc-fingers (`ZSCAN9`), KLK serine proteases (`KLK7/10/11`), and PPP phosphatases (`PPP2CA`, `PPP1CB`, `PPP2R1A`) — while iteration 10 returns no hits as the productive threads are exhausted. More precisely, KLK4 and ZSCAN5A

*Table 2.* Family exploitation and exhaustion across iterations for the ICL-EF agent (F0). Each cell shows hits/tested for that gene family at that iteration. Dashes indicate the family was not tested. Major early families (CHD, TAF, POLR, MED) are exhausted by iteration 7, forcing the agent to pivot to newly identified anchors (ZSCAN, KLK, PPP) at iterations 8–10. The table reports 26 hits that can be assigned to a gene family, 9 hits correspond to singleton genes (no family).

| Family | It. 1 | It. 2 | It. 3 | It. 4 | It. 5 | It. 6 | It. 7 | It. 8 | It. 9 | It. 10 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CHD | 1/5 | 0/4 | — | — | — | — | — | — | — | — | 1/9 |
| KMT2 | 1/4 | 0/1 | — | — | — | — | — | — | — | — | 1/5 |
| TAF | 1/6 | 1/14 | — | 0/3 | — | — | — | — | — | — | 2/23 |
| POLR | 1/3 | 1/11 | 1/10 | — | — | — | — | — | — | — | 3/24 |
| MED | — | 0/6 | 2/10 | 0/1 | — | — | — | — | — | — | 2/17 |
| HDAC | 0/7 | — | — | — | 0/4 | — | — | — | — | — | 0/11 |
| SMARC/ARID | 0/10 | 0/2 | — | 0/4 | — | — | — | — | — | — | 0/16 |
| KDM | 0/7 | — | — | 0/3 | — | — | — | — | — | — | 0/10 |
| CUL | — | — | — | — | — | — | 1/5 | 0/3 | — | — | 1/8 |
| TRIM | — | — | — | — | — | 1/3 | — | 0/8 | 0/3 | 0/9 | 1/23 |
| IRF | — | — | — | — | — | — | 1/7 | 0/2 | 0/1 | — | 1/10 |
| CDK | — | 0/5 | 0/4 | — | — | — | — | — | — | — | 0/9 |
| PSM | — | — | — | — | — | — | 1/4 | 0/2 | 0/13 | 0/5 | 1/24 |
| UBE2 | — | — | — | — | — | — | 1/5 | — | 0/4 | 0/19 | 1/28 |
| USP | — | — | — | — | — | — | 0/1 | 0/12 | — | — | 0/13 |
| ZSCAN | — | — | — | — | — | 1/1 | — | 1/12 | — | — | 2/13 |
| KLK | — | — | — | — | — | 1/1 | — | 3/10 | 0/3 | 0/1 | 4/15 |
| PPP | — | — | — | — | — | — | — | 1/1 | 2/4 | 0/18 | 3/23 |
| KIF | — | — | — | — | — | — | — | 1/4 | 0/7 | — | 1/11 |
| GTF | — | 0/4 | — | 1/14 | 0/1 | — | — | — | — | — | 1/19 |
| SUPT | — | 1/2 | 0/4 | — | — | — | — | — | — | — | 1/6 |
| MCM | — | 0/2 | 0/5 | — | — | — | — | — | — | — | 0/7 |
| RNF | — | — | — | — | — | — | 0/3 | — | — | 0/6 | 0/9 |
| SIRT | — | — | — | — | 0/3 | — | — | 0/3 | — | — | 0/6 |
| NUP | — | — | — | 0/3 | — | — | — | — | — | — | 0/3 |
| FBX | — | — | — | — | — | 0/2 | 0/1 | 0/6 | — | 0/7 | 0/16 |
| PRMT | — | — | — | — | — | — | 0/4 | 0/2 | — | — | 0/6 |

were caught at iteration 6 and tagged at "diverse" in the LLM completion. The exploitation of those families begins at iteration 8 perhaps because other previously exploited families are exhausted as show in Table 2.

**Summary.** After 1,000 tests the agent discovers 35 of 152 significant genes (23.0% recall): `CHD4`, `KMT2B`, `EP300`, `TAF1`, `POLR2A`, `PCNA`, `TRRAP`, `MYC`, `TAF5`, `SUPT6H`, `POLR3B`, `POLR3K`, `MED21`, `MED27`, `XPO1`, `GTF3C5`, `DNA2`, `UHRF1`, `TRIM28`, `KLK4`, `ZSCAN5A`, `IRF3`, `MDM2`, `CUL3`, `RBX1`, `PSMC2`, `UBE2M`, `ZSCAN9`, `KLK7`, `KLK10`, `KLK11`, `KIF11`, `PPP2CA`, `PPP1CB`, `PPP2R1A`. The trajectory reveals a clear exploitation dynamic: transcription-machinery hits in iterations 1–5 trigger systematic family sweeps (TAF, POLR, Mediator, GTF subunits); iteration 6 seeds three non-obvious families via `TRIM28`, `KLK4`, and `ZSCAN5A`; iteration 7 then pivots to the ubiquitin-proteasome system, yielding 6 hits in a single batch (`IRF3`, `MDM2`, `CUL3`, `RBX1`, `PSMC2`, `UBE2M`); and iteration 8 expands the KLK and ZSCAN families while discovering PPP phosphatases (`PPP2CA`), which yield two further hits in iteration 9. This name-prefix-driven exploitation accounts for the performance gap over the zero-shot agent (see Appendix A).

## C. ICBR-EF Agent: Prompting Strategy and Execution Trace

The ICBR-EF (In-Context Belief Revision from Experimental Feedback) agent extends ICL-EF with two mechanisms. First, we provide a *phenotypic fingerprint* that gives the LLM the top-10 co-affected CellProfiler features for a selection of the tested gene (8 most recent hits and 4 most recent misses), enabling mechanism-based clustering. Besides, we also compute co-occurring significant side-effects across all hits: for each successful gene, we count how many other CellProfiler features (besides the target) also reach $p < 0.05$. The top-5 most frequently co-significant features are then reported as a shared phenotypic signature, e.g. - AGP: 7/10 HITs. This tells the LLM which secondary cellular changes reliably accompany the target effect yielding effectively a data-driven phenotypic cluster that can guide selection of untested genes likely to share the same mechanism.

Second, an explicit *hypothesis register* (structured as a JSON) is updated by the LLM across iterations, tracking mechanistic beliefs with confidence levels (`High`/`Medium`/`Low`/`Neutral`) and statuses (`Active`/`Weakened`/`Abandoned`/`New`). Below we reproduce the exact system prompt, the hypothesis register schema, the user prompt structure, and an annotated execution trace.

### C.1. System Prompt

The system prompt is fixed across all iterations. It is constructed at agent initialization by filling in the target feature name, pool size, and feature count:

```
You are an expert computational biologist AI agent discovering gene perturbations
that induce significant cellular responses.

TASK: Find gene knockouts where p-value for 'Nuclei_Intensity_MeanIntensity_AGP'
      is < 0.05.

## STRATEGY: Phenotypic Clustering & Mechanism Inference

Your key advantage is seeing the FULL PHENOTYPIC FINGERPRINT of each tested gene:
- Target feature p-value (what we're optimizing)
- Top 10 side effect p-values (other affected features)

### Use phenotypes to identify mechanisms:
1. Phenotype Clustering: Genes with similar side-effect profiles likely share
   mechanisms
2. Pathway Inference: If a gene affects features X,Y,Z and is a HIT, test other
   genes that might affect the same features
3. Functional Categories: Group genes by phenotypic signature, not just name

### Hypothesis-Driven Selection:
- Form hypotheses about which mechanisms cause hits
- Test hypotheses by selecting genes predicted to share that mechanism
- Update confidence based on results

## AVAILABLE DATA:
- Total perturbations: 7975
- Features: 100 CellProfiler measurements
- Target: Nuclei_Intensity_MeanIntensity_AGP

## OUTPUT FORMAT:
Return a JSON object with:
1. "hypotheses_register": Updated list of your hypotheses with confidence levels
2. "selection": List of gene names to test next

Example:
{
  "hypotheses_register": [
    {
      "hypothesis": "Transporter family genes cause intensity changes",
      "confidence": "High",
      "status": "Active",
      "reasoning": "Multiple transporter genes showed strong effects"
    },
    {
      "hypothesis": "Kinases in general affect this phenotype",
      "confidence": "Low",
      "status": "Weakened",
      "reasoning": "Tested 3 kinases, all MISS"
    }
  ],
  "selection": ["GENE1", "GENE2", "GENE3", ...]
}

Remember: Use phenotypic similarity to guide selection, not just gene name patterns!
```

### C.2. Hypothesis Register Schema

The hypothesis register is a JSON array, entirely authored and updated by the LLM. Each entry has four fields:

```
[
  {
    "hypothesis": <string>,   // Free-text mechanistic claim
    "confidence": <string>,   // "High" | "Medium" | "Low" | "Neutral"
    "status":     <string>,   // "Active" | "Weakened" | "Abandoned" | "New"
    "reasoning":  <string>    // Evidence cited: gene names, p-values, phenotypes
  },
  ...
]
```

The framework seeds the register with a single entry before iteration 1:

```
[{"hypothesis": "Global Exploration", "confidence": "Neutral", "status": "Active",
  "reasoning": "Starting broad search to identify active phenotypes."}]
```

At every subsequent iteration the LLM replaces this array entirely, adding, updating, or removing entries as evidence accumulates.

### C.3. Per-Iteration User Prompt

At each iteration $t$, the user prompt is assembled dynamically from three components.

**1. Observation summary.** For each tested gene the agent receives a *phenotypic fingerprint*: target p-value and the five most significant co-affected features (abbreviated CellProfiler names). For example, after the first batch of 100 genes:

```
TESTED: 100 | HITs: 7 | MISSes: 93

RECENT HITs (with phenotypic fingerprints):
  EP300:  [HIT] Target_p=0.0073 | Side-effects: Cells_Texture_DifferenceEntropy_AGP=0.000,
     Cells_Texture_InverseDifferenceMoment_AGP=0.000, ...
  ...
  KMT2B:  [HIT] Target_p=0.0024 | Side-effects: Cells_Texture_DifferenceEntropy_AGP=0.000,
     Cells_Texture_InverseDifferenceMoment_AGP=0.000, ...

RECENT MISSes (sample):
  PRMT1:  [MISS] Target_p=0.7797 | Side-effects: Nuclei_Texture_AngularSecondMoment_AGP=0.543,
     Nuclei_Texture_AngularSecondMoment_AGP=0.551, ...
  ...
  MBD2:   [MISS] Target_p=0.4998 | Side-effects: Cells_Texture_InverseDifferenceMoment_AGP=0.597,
     Nuclei_Texture_InverseDifferenceMoment_AGP=0.597, ...
```

Once at least 3 hits have been observed, a co-occurrence analysis is appended to the summary. For each successful gene, all CellProfiler features reaching $p < 0.05$ (other than the target) are counted; the 5 most frequent are reported as a shared phenotypic signature:

```
### PHENOTYPE PATTERNS IN HITs:

 Common side-effects in HITs: - Cytoplasm_Texture_Entropy_AGP: 6/7 HITs - Nuclei_Texture_AngularSecondMoment_AGP: 5/7 HITs
     - Cells_Texture_InverseDifferenceMoment_AGP: 5/7 HITs -  Nuclei_Texture_InverseDifferenceMoment_AGP: 5/7 HITs -
     Cytoplasm_Texture_InverseDifferenceMoment_AGP: 5/7 HITs
```

This block tells the LLM which secondary phenotypic changes reliably co-occur with target-feature hits, providing a data-driven anchor for hypothesis formation and phenotypic clustering.

**2. Current hypothesis register.** The full JSON array from the previous iteration is serialized and appended.

```
### CURRENT HYPOTHESES:
[
```

```
  { "hypothesis": "Chromatin remodeling and epigenetic regulators affect nuclear AGP intensity",
    "confidence": "Medium",
    "status": "Active",
    "reasoning": "Histone modifiers (HDACs, EZH2, KDM1A, BRD4) alter chromatin compaction and nuclear architecture,
     potentially affecting AGP staining intensity in nuclei"
  },
...
]
```

**3. Task specification.** We list already-tested and untested gene IDs, then asks the LLM to (a) update the hypothesis register and (b) propose the next 100 genes. An explicit self-check instruction is added:

```
YOUR TASK:
  1 Update your hypotheses based on the phenotypic patterns observed
  2 Select 100 perturbations that will:
     \textbullet{} Test your high-confidence hypotheses (exploitation)
     \textbullet{} Explore new potential mechanisms (exploration)
     \textbullet{} Use phenotypic similarity to find related genes


CRITICAL REQUIREMENT: Before returning your final JSON, you MUST explicitly double-check your proposed genes ONE BY ONE.
     For each gene you want to propose, verify:

  1 It is EXACTLY present in the UNTESTED list above.
  2 It is NOT present in the ALREADY TESTED list. If you realize a gene violates these rules, propose a replacement and
     double-check the replacement.

After your verification, return the JSON strictly inside a markdown code block exactly like this:
  {
    "hypotheses_register": [...],
    "selection": ["GENE1", "GENE2", ...]
  }
```

### C.4. Execution Trace

We trace the agent on `Nuclei_Intensity_MeanIntensity_AGP` / F0 (152 significant genes out of 7,975), with batch size 100 and 10 iterations (1,000 total tests, 12.5% of the pool).

**Iteration 1 — Broad initialization.** With no observations yet, the agent interprets the target feature as reflecting nuclear AGP-channel protein accumulation and constructs seven mechanism-grounded hypotheses: chromatin remodelling/epigenetics (HDACs, EZH2, BRD4), cell-cycle control (CDKs, Aurora kinases), nuclear transport (XPO1, importins, nucleoporins), DNA-damage response (ATM, ATR, BRCA1/2, PARP1), master transcription factors (MYC, SP1, CTCF), kinase signalling (AKT, MTOR, MAPK), and ubiquitin/proteasome (MDM2, CUL3). It selects 100 genes spanning all seven categories. Hits: EP300, XPO1, MYC, MDM2, CUL3, CHD4, KMT2B (7/100).

**Iteration 2 — Focused exploitation with hypothesis refinement.** Observing that all 7 hits are major chromatin/ubiquitin/transcription regulators sharing AGP-texture side-effect signatures, the agent abandons the broad cell-cycle, DNA-damage-response, and kinase-signalling hypotheses (all returned only misses) and replaces them with eight targeted hypotheses: KMT-family histone methyltransferases (from KMT2B), CHD/SWI–SNF remodellers (from CHD4), NuRD complex subunits (CHD4 is a NuRD core component), cullin-RING ubiquitin ligases (from CUL3, the strongest hit), importin/exportin nuclear transport (from XPO1), MYC-network partners (MAX, MXI1, MXD1), Mediator subunits (from EP300/CREBBP), and HAT/coactivator complexes (from EP300). The 100-gene batch covers all eight hypotheses. Hits: UHRF1 (1/100).

**Iteration 3 — Pruning failed branches, opening new ones.** The analysis of MISSes from iterations 1–2 leads the agent to drop four previously active hypotheses: HDAC-family members (HDAC1–8 all missed), CHD paralogs beyond CHD4 (CHD1/2/5/6/7/8/9 all missed), KMT paralogs beyond KMT2B (KMT2A/C/D/E all missed), and importin-side nuclear transport (most importins missed). Four new hypotheses are raised: CUL3 substrate adaptors with BTB domains

(KEAP1 and related, motivated by CUL3 being the strongest hit), PHD-finger chromatin readers (PHF family connecting to the NuRD/CHD4 hit), PRDM/SET-domain methyltransferases (extending from KMT2B), and transcription elongation factors (SUPT and TAF families, motivated by the EP300/MYC/TRRAP context). The batch covers these four new strands alongside the continuing Mediator, NuRD, and cullin-RING hypotheses. Hits: TRIM28, MED27, SUPT6H, TAF1, TAF5 (5/100).

**Iterations 4–10 — Cascading hypothesis expansion driven by new hits.** Iterations 4–5 exploit the TAF/Mediator/SUPT leads from iteration 3, yielding `MED21`, `POLR2A`, `RBM14`, and `POLR3B`. `POLR2A` triggers a major pivot: the agent raises a new high-confidence hypothesis on RNA Pol II/III machinery and shifts selections towards RNA polymerase subunits, general transcription factors (`GTF` family), and transcription-associated kinases (`CDK7/8/9`). Iteration 6 confirms this strand with five hits (`POLR3K`, `GTF3C5`, `PRPF19`, `EXOSC9`, `XAB2`), establishing RNA Pol III transcription and pre-mRNA splicing/RNA surveillance as new dominant hypotheses. Iteration 7 investigates DDX helicases and spliceosome components (which are co-transcriptional RNA processing extending the transcriptional strand), hitting `DDX39A`, `ZFP36L2`, `CAND1`, and `EIF3G`. The `ZFP36L2` hit (AU-rich element mRNA decay) opens yet another hypothesis strand on mRNA stability, while `EIF3G` suggests translation factors as a new target class. Iterations 8–9 further expand into proteasome (`PSMC2`), DNA replication (`PCNA`, `DNA2`), RNA modification (`METTL1`), nuclear export (`RAE1`), ribosomal proteins (`RPL4`, `RPL7`), translation elongation (`EEF2`), and DNA repair (`PNKP`), accumulating 11 additional hits. Iteration 10 returns zero hits as the agent exhausts high-confidence candidates and sweeps lower-priority targets (DNA polymerases, cohesin, checkpoint genes). Table 3 report how gene family are exploited by the agent across iterations.

**Summary.** After 1,000 tests the agent discovers 37 of 152 significant genes (24.3% recall): `EP300, XPO1, MYC, MDM2, CUL3, CHD4, KMT2B, UHRF1, TRIM28, TRRAP, MED27, SUPT6H, TAF1, TAF5, MED21, POLR2A, RBM14, POLR3B, POLR3K, PRPF19, GTF3C5, EXOSC9, XAB2, DDX39A, ZFP36L2, CAND1, EIF3G, PCNA, DNA2, POP4, METTL1, PSMC2, RAE1, EEF2, PNKP, RPL4, RPL7`. The trajectory reveals a clear mechanistic drift: the first three iterations concentrate on chromatin regulators, ubiquitin ligases, and nuclear transport; iterations 4–6 pivot to the transcription machinery (TFIID, Mediator, RNA Pol II/III); iterations 7–9 cascade outward into RNA processing, mRNA stability, translation, ribosome biogenesis, proteasome, and DNA repair, each pivot triggered by a surprise hit whose phenotypic fingerprint points to an unexpected but related pathway. The hypothesis register expands from 7 broad priors at iteration 1 to over 12 mechanistically distinct entries by iteration 9, spanning chromatin remodelling, ubiquitin/CUL3 ligases, Mediator/TFIID, RNA Pol II/III, spliceosome, RNA surveillance/exosome, mRNA stability, translation, ribosome biogenesis, proteasome, DNA replication, and DNA repair that were all inferred autonomously from phenotypic co-occurrence patterns in the observed hits.

## D. Additional statistical significance assessment of performance gaps

In this appendix, we report pairwise per-feature significance of performance gaps in Table 4. Those p-values are obtained by applying paired two-sided sign-flip permutation tests to the replicate-level performance differences within each feature.

There is strong evidence that all baselines and the investigated approaches outperform choosing random genes at random. This is also true for Random FB which is indicative that the LLM is probably able to filter out the noisy feedback and leverage its sole prior knowledge. Moreover, when using Claude Sonnet 4.6, ICL approaches very consistently outperforms the zero-shot agent. The significance of the comparison of ICL methods vs Random FB is less sharp but remains the observed trend. Finally, the performance gain of ICBR-EF vs ICL-EF cannot be confirmed on a per-feature basis.

To clarify this latter point, we also provide 99% hierarchical bootstrap confidence intervals obtained by resampling features and, within each sampled feature, resampling replicates in Table 5. These intervals provide an uncertainty estimate for the average feature-wise performance gap that complements the permutation-based significance tests reported in the main text. Significant performances correspond to confidence intervals in which zero is excluded. The ICBR-EF vs ICL-EF remains borderline with the reported interval upper-bound being equal to zero but this analysis supports the conclusions drawn from Table .1 In this more holistic way of assessing statistical significance, the added of value of ICL compared to Random FB is clear.

*Table 3.* Family exploitation across iterations for the ICBR-EF agent (F0). Each cell shows hits/tested. The agent performs shallow within-family probing before rapidly pivoting across gene families that share high-level phenotypic signatures (e.g., RNA-binding, essentiality, central dogma involvement), despite limited functional or mechanistic continuity. The table reports 20 hits that can be assigned to a gene family, 17 hits correspond to singleton genes (no family).

| Family | It. 1 | It. 2 | It. 3 | It. 4 | It. 5 | It. 6 | It. 7 | It. 8 | It. 9 | It. 10 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CHD | 1/2 | 0/6 | — | — | — | — | — | — | — | — | 1/8 |
| KMT2 | 1/3 | 0/2 | — | — | — | — | — | — | — | — | 1/5 |
| TAF | — | — | 1/3 | 1/16 | 0/4 | — | — | — | — | — | 2/23 |
| POLR | — | — | — | — | 1/12 | 2/12 | — | — | — | — | 3/24 |
| MED | — | 0/4 | 1/9 | 1/4 | — | — | — | — | — | — | 2/17 |
| HDAC | 0/3 | 0/5 | — | 0/3 | — | — | — | — | — | — | 0/11 |
| SMARC/ARID | 0/5 | 0/6 | — | — | — | — | — | — | — | 0/4 | 0/15 |
| KDM | 0/6 | — | 0/3 | — | — | — | — | — | — | — | 0/9 |
| CUL | 1/3 | 0/4 | — | — | — | — | — | — | — | 0/1 | 1/8 |
| TRIM | — | — | 1/3 | 0/32 | 0/8 | — | — | — | — | — | 1/43 |
| CDK | 0/4 | — | — | — | 0/6 | — | — | — | — | — | 0/10 |
| PSM | 0/2 | — | — | — | — | — | — | — | 1/20 | 0/2 | 1/24 |
| UBE2 | — | — | — | — | — | — | — | — | 0/4 | — | 0/4 |
| GTF | — | — | — | — | 0/15 | 1/4 | — | — | — | — | 1/19 |
| SUPT | — | — | 1/5 | 0/1 | — | — | — | — | — | — | 1/6 |
| EXOSC | — | — | — | — | — | 1/5 | — | — | — | — | 1/5 |
| PRPF | — | — | — | — | — | 1/4 | — | — | — | — | 1/4 |
| DDX | 0/1 | — | — | — | — | 0/8 | 1/12 | 0/11 | 0/4 | — | 1/36 |
| EIF | — | — | — | — | — | 0/5 | 0/3 | 1/8 | 0/7 | 0/5 | 1/28 |
| METTL | — | — | — | — | — | — | — | 1/6 | — | — | 1/6 |
| EEF | — | — | — | — | — | — | — | — | 1/7 | — | 1/7 |
| MCM | — | — | — | — | — | — | — | 0/7 | — | — | 0/7 |
| RNF | — | 0/2 | 0/3 | 0/1 | — | — | — | — | — | — | 0/6 |
| SIRT | 0/3 | — | — | — | — | — | — | — | — | — | 0/3 |
| NUP | 0/3 | — | — | — | — | — | 0/1 | — | — | — | 0/4 |
| PRMT | 0/2 | 0/3 | 0/1 | — | — | — | — | — | — | — | 0/6 |

| Pair | F0 | F10 | F20 | F30 | F40 | F50 | F60 | F70 | F80 | F90 | $p < 0.01$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Random vs GP-UCB | **0.004** | **0.004** | **0.002** | **0.002** | **0.008** | 0.012 | **0.008** | **0.002** | **0.008** | **0.002** | 9/10 |
| Random vs Zero-shot (4.5) | **0.004** | 0.076 | **0.002** | **0.002** | **0.002** | **0.004** | 0.012 | **0.002** | **0.002** | **0.002** | 8/10 |
| Random vs ICL-EF (4.5) | **0.004** | **0.002** | **0.002** | **0.004** | **0.006** | **0.002** | **0.002** | **0.006** | **0.002** | **0.002** | 10/10 |
| Random vs Zero-shot (4.6) | **0.008** | **0.004** | **0.002** | **0.002** | **0.002** | **0.002** | **0.004** | **0.002** | **0.004** | **0.002** | 10/10 |
| Random vs Random FB | **0.008** | **0.006** | **0.006** | **0.002** | **0.004** | **0.002** | **0.010** | **0.006** | 0.074 | **0.010** | 9/10 |
| Random vs ICL-EF (4.6) | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | 10/10 |
| Random vs ICBR-EF (4.6) | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | 10/10 |
| GP-UCB vs Zero-shot (4.5) | 1.000 | 0.016 | 0.523 | 0.828 | 0.344 | 0.633 | 0.840 | 0.266 | 0.176 | 0.258 | 0/10 |
| GP-UCB vs ICL-EF (4.5) | 0.648 | 0.770 | 0.662 | 0.311 | 0.773 | 0.152 | 0.119 | 0.982 | **0.002** | 0.031 | 1/10 |
| GP-UCB vs Zero-shot (4.6) | 0.266 | 0.859 | 0.172 | 0.600 | 0.875 | 0.102 | 0.898 | 0.047 | 0.891 | 0.031 | 0/10 |
| GP-UCB vs Random FB | 0.553 | 0.223 | 0.051 | 0.275 | 0.555 | 0.293 | 1.000 | 0.117 | 0.057 | 0.035 | 0/10 |
| GP-UCB vs ICL-EF (4.6) | **0.002** | 0.021 | **0.002** | 0.021 | **0.004** | **0.008** | **0.008** | **0.002** | **0.002** | **0.004** | 8/10 |
| GP-UCB vs ICBR-EF (4.6) | 0.012 | **0.002** | **0.002** | **0.004** | **0.002** | **0.002** | 0.012 | **0.002** | **0.002** | **0.002** | 8/10 |
| Zero-shot (4.5) vs ICL-EF (4.5) | 0.500 | 0.020 | 0.928 | 0.031 | 0.941 | 0.344 | 0.244 | 0.410 | **0.008** | 0.312 | 1/10 |
| Zero-shot (4.5) vs Zero-shot (4.6) | 0.117 | 0.012 | 0.373 | 0.648 | 0.438 | 0.758 | 0.566 | 0.672 | 0.078 | 0.520 | 0/10 |
| Zero-shot (4.5) vs Random FB | 0.586 | 0.055 | **0.010** | 0.328 | 1.000 | 0.949 | 0.836 | 0.029 | **0.006** | **0.004** | 3/10 |
| Zero-shot (4.5) vs ICL-EF (4.6) | **0.008** | **0.002** | **0.002** | **0.002** | **0.002** | **0.002** | **0.006** | **0.002** | **0.002** | **0.004** | 10/10 |
| Zero-shot (4.5) vs ICBR-EF (4.6) | **0.002** | **0.002** | **0.004** | **0.004** | **0.002** | **0.010** | 0.021 | **0.002** | **0.002** | **0.002** | 9/10 |
| ICL-EF (4.5) vs Zero-shot (4.6) | 0.086 | 0.625 | 0.055 | 0.062 | 0.848 | 0.703 | 0.086 | 0.141 | **0.002** | 1.000 | 1/10 |
| ICL-EF (4.5) vs Random FB | 0.941 | 0.250 | 0.084 | 0.016 | 0.973 | 0.469 | 0.172 | 0.234 | **0.002** | **0.004** | 2/10 |
| ICL-EF (4.5) vs ICL-EF (4.6) | 0.062 | **0.008** | **0.002** | **0.002** | **0.010** | **0.004** | 0.441 | **0.002** | **0.002** | **0.008** | 8/10 |
| ICL-EF (4.5) vs ICBR-EF (4.6) | 0.053 | **0.002** | **0.004** | **0.002** | 0.016 | **0.006** | 0.312 | **0.002** | **0.006** | **0.002** | 7/10 |
| Zero-shot (4.6) vs Random FB | 0.082 | 0.344 | 0.033 | 0.734 | 0.477 | 0.703 | 0.832 | **0.008** | 0.102 | **0.008** | 2/10 |
| Zero-shot (4.6) vs ICL-EF (4.6) | **0.004** | 0.012 | **0.002** | **0.004** | **0.002** | 0.020 | **0.008** | **0.002** | **0.002** | 0.055 | 7/10 |
| Zero-shot (4.6) vs ICBR-EF (4.6) | **0.002** | **0.002** | **0.004** | **0.004** | **0.002** | **0.004** | **0.004** | **0.002** | **0.002** | **0.002** | 10/10 |
| Random FB vs ICL-EF (4.6) | 0.074 | 0.014 | **0.002** | 0.027 | **0.006** | 0.014 | 0.020 | **0.002** | **0.002** | **0.002** | 5/10 |
| Random FB vs ICBR-EF (4.6) | 0.117 | **0.002** | **0.002** | **0.004** | **0.008** | **0.004** | 0.025 | **0.002** | **0.002** | **0.004** | 8/10 |
| ICL-EF (4.6) vs ICBR-EF (4.6) | 0.740 | 0.016 | 0.836 | 0.184 | 0.572 | 0.512 | 0.672 | 0.121 | 0.646 | 0.012 | 0/10 |

*Table 4.* Per-feature paired sign-flip permutation p-values. Bold: $p < 0.01$.

| Pair | Obs. diff | 99% CI |
|---|---|---|
| **Random vs GP-UCB** | **-8.530** | **[-10.610, -6.580]** |
| **Random vs Zero-shot (4.5)** | **-9.080** | **[-11.540, -6.640]** |
| **Random vs ICL-EF (4.5)** | **-10.800** | **[-15.270, -7.800]** |
| **Random vs Zero-shot (4.6)** | **-9.390** | **[-12.000, -7.050]** |
| **Random vs Random FB** | **-7.360** | **[-9.650, -5.070]** |
| **Random vs ICL-EF (4.6)** | **-18.330** | **[-25.090, -12.610]** |
| **Random vs ICBR-EF (4.6)** | **-20.430** | **[-28.000, -13.900]** |
| GP-UCB vs Zero-shot (4.5) | -0.550 | [-2.760, +2.280] |
| GP-UCB vs ICL-EF (4.5) | -2.270 | [-6.630, +0.640] |
| GP-UCB vs Zero-shot (4.6) | -0.860 | [-2.700, +0.970] |
| GP-UCB vs Random FB | +1.170 | [-1.200, +3.680] |
| **GP-UCB vs ICL-EF (4.6)** | **-9.800** | **[-16.010, -5.230]** |
| **GP-UCB vs ICBR-EF (4.6)** | **-11.900** | **[-18.560, -6.530]** |
| Zero-shot (4.5) vs ICL-EF (4.5) | -1.720 | [-5.600, +1.390] |
| Zero-shot (4.5) vs Zero-shot (4.6) | -0.310 | [-2.940, +1.990] |
| Zero-shot (4.5) vs Random FB | +1.720 | [-1.490, +5.140] |
| **Zero-shot (4.5) vs ICL-EF (4.6)** | **-9.250** | **[-14.900, -4.690]** |
| **Zero-shot (4.5) vs ICBR-EF (4.6)** | **-11.350** | **[-18.580, -5.820]** |
| ICL-EF (4.5) vs Zero-shot (4.6) | +1.410 | [-2.030, +6.400] |
| ICL-EF (4.5) vs Random FB | +3.440 | [-0.630, +9.090] |
| **ICL-EF (4.5) vs ICL-EF (4.6)** | **-7.530** | **[-11.720, -3.660]** |
| **ICL-EF (4.5) vs ICBR-EF (4.6)** | **-9.630** | **[-15.030, -4.950]** |
| Zero-shot (4.6) vs Random FB | +2.030 | [-1.080, +5.170] |
| **Zero-shot (4.6) vs ICL-EF (4.6)** | **-8.940** | **[-15.330, -4.500]** |
| **Zero-shot (4.6) vs ICBR-EF (4.6)** | **-11.040** | **[-17.930, -5.810]** |
| **Random FB vs ICL-EF (4.6)** | **-10.970** | **[-18.550, -5.080]** |
| **Random FB vs ICBR-EF (4.6)** | **-13.070** | **[-21.080, -6.380]** |
| ICL-EF (4.6) vs ICBR-EF (4.6) | -2.100 | [-5.490, +0.000] |

*Table 5.* Hierarchical bootstrap comparison (sorted by $p$-value). Bold rows: 99% CI excludes 0.