PRICED GUIDANCE: CAN LANGUAGE MODELS GENERATE FUTURE RESEARCH IDEAS?

Anonymous authors
Paper under double-blind review

ABSTRACT

We evaluate language models' capability to generate novel research ideas through the lens of compression. Our goal is to lower-bound the potentially tiny probability that a language model generates the essence of a future research idea without any hints. Rather than estimate this probability through expensive repeated sampling, our Priced Guidance framework measures the compression cost: how many additional bits of information are needed to guide the model to recover the target idea. We prove that if the model can recover the target idea with at most K bits of guidance in expectation, then it can generate the idea without any guidance with probability at least 2^{-K}. In our framework, the language model, called the generator, can pose a sequence of multiple-choice questions, e.g., "what is the topic of the research idea?", and specify a probability distribution over possible answers. A guide, which is a language model with access to the target idea, selects answers. If a selected answer has prior probability p, the generator pays $-\log_2 p$ bits. The generator's goal is to minimize the cumulative cost it needs to pay to produce an idea that matches the essence of the target idea. This cumulative cost equals, up to an additive constant, the number of bits of information sent by the guide. Using this methodology, we evaluate five generator models (Opus 5, Fable 5.1, GPT-6 Astra, GPT-5.6 Sol, and GLM 5.3) on the core ideas in 87 recent high-quality deep learning papers. We use a frontier model as a judge to determine whether the generated idea matches the target in terms of the central research object and defining mechanism. Fable 5.1 achieves the lowest median compression cost at 67.8 bits, substantially lower than gzip's median of 5,712 bits for losslessly compressing the summary of the target idea. A uniform ensemble of Fable 5.1, Opus 5, and GPT-6 Astra further reduces the median compression cost to 53.8 bits and improves the generation probability lower-bound by $16{,}900\times$.

1 INTRODUCTION

The ability to generate novel research ideas is a critical part of advanced intelligence. Yet evaluating LLM-generated ideas is difficult because novelty perceived by humans or language models does not necessarily translate into better research outcomes (Si et al., 2024; 2025; Baig et al., 2026). A potential approach is to test whether a model can generate an unseen future research idea. Existing benchmarks provide partial information such as keywords (Ruan et al., 2024) or parent papers (He-Yueya et al., 2026) to guide the reconstruction, but do not quantify how much information this guidance provides.

In this paper, we measure a language model's idea-generation capability through the lens of **compression** (Shannon, 1948; Schmidhuber, 2009; 2010): how many additional bits are needed to guide the model toward the essence of an idea from a target future paper? The intuition comes from the classical duality between prediction and compression: what a model can already predict well requires little additional information to specify, while what it finds surprising requires more. Thus, the closer a model is to generating an idea on its own, the fewer bits should be needed to guide it there.

We propose *Priced Guidance* (Figure 1), a framework to quantify compression cost of ideas. We take inspiration from the game of Twenty Questions by endowing the model being evaluated, which we call the generator, with a particular structure: before the generator produces its final idea, its reasoning trace contains a sequence of **bids**. Each bid consists of a question and a probability

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.

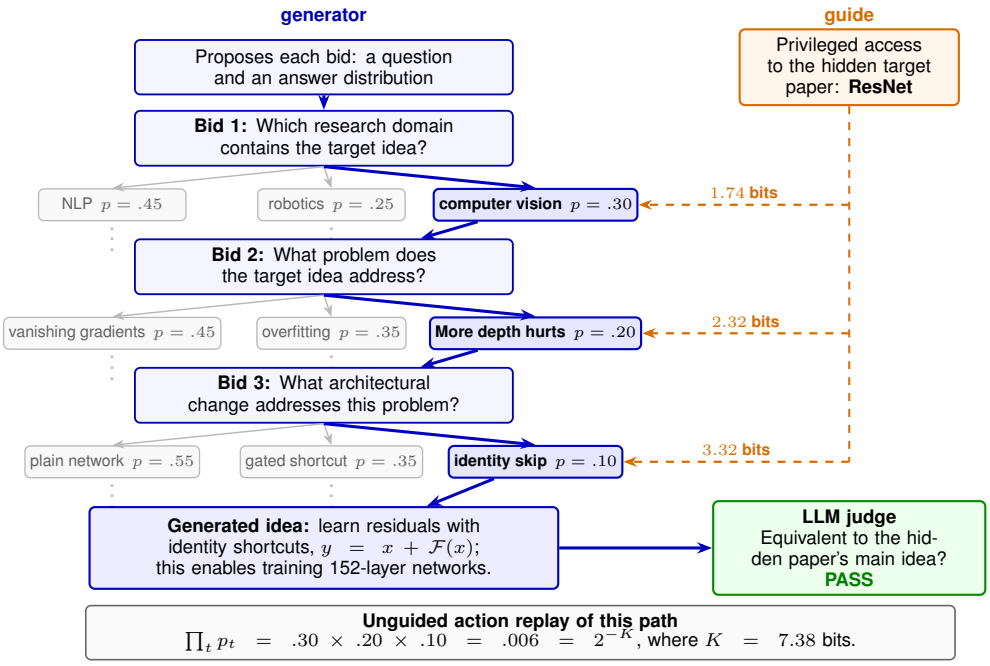

Figure 1: A top-down view of the Priced Guidance protocol as tree search, illustrated with the ResNet idea. At each bid, the generator asks a question and assigns probabilities to possible answers, defining branches of the tree. The guide uses privileged access to the target paper to select an answer. Orange arrows show the information sent from the guide to the generator, labeled by its cost in bits; blue highlights the resulting trajectory. For a fixed generator realization and a judge that accepts this idea with probability one, 2^{-K} lower-bounds the probability of generating an equivalent idea without guidance. Probabilities and questions are stylized here.

distribution over a finite set of candidate answers. We then introduce a guide language model with privileged access to the target paper, which chooses among these answers and thereby guides the generator toward the target idea. When the guide chooses an answer to which the generator assigned probability p, the generator pays a cost of $-\log_2 p$ bits. The interaction ends when the generator chooses to submit an idea, and an LLM judge evaluates whether the generated idea is equivalent in essence to the target idea. If the final idea passes this judge, the sequence of guide answers can then be viewed as a compressed representation of the additional guidance needed for the generator to recover the target idea. With the generator and its randomness fixed, arithmetic coding lets us reconstruct the answer sequence and hence the generated idea from at most the total cost plus $O(1)$ bits. We therefore call the total answer cost the **compression cost** (see Section 2 for a formal definition). Here the guide may be arbitrarily knowledgeable and computationally powerful because the protocol measures the cost of the information it communicates rather than restricting how it obtains that information.

Priced Guidance generalizes the use of language models to compress a target text verbatim (Delétang et al., 2024), with each bid asking for the next token of the target summary. In our framework, the generator may perform multi-step reasoning between bids and is allowed to generate an idea equivalent in essence to the target rather than a particular textual realization. This also avoids requiring token log-probabilities, which are unavailable for many closed-weight frontier models. Just as a lower compression cost on text indicates a stronger language model, a lower compression cost under Priced Guidance indicates a stronger idea generator.

The generator structure allows us to connect compression cost directly to the probability of reconstructing the target idea without any additional information. In addition to the *guided* mode, where the guide selects answers using its knowledge of the target idea, we can run the same generator in an *unguided* mode, where it instead samples an answer from the distribution specified in each bid. We prove that, for a fixed target, if K is the random compression cost of a guided run, then the corresponding unguided generator outputs an idea that passes the judge with probability at least $\mathbb{E}[2^{-K}]$ (Theorem 2.2), where the expectation is over the randomness of the generator and guide. Thus, a low compression cost implies a high probability that the unguided generator produces an idea that passes the judge. Because the relationship is exponential, reducing the compression cost of

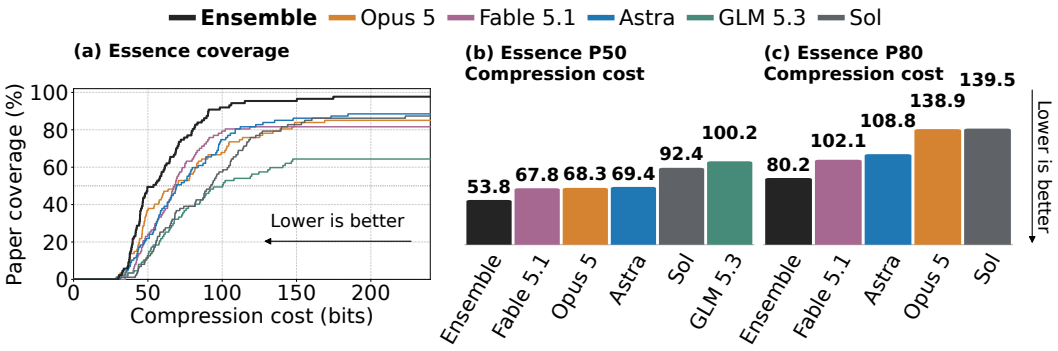

Figure 2: Test-set results under the Essence judge on the 87-paper test set. (a) Coverage versus compression cost. Curves show the percentage of papers successfully compressed at each compression cost for each generator model. The black ensemble uniformly mixes Fable 5.1, Opus 5, and Astra. (b) Median (P50) and (c) 80th-percentile (P80) compression cost. Here we define the qth-percentile compression cost as the minimal compression cost required to output an idea that passes the judge on $q\%$ of the test papers. GLM 5.3 is omitted from the P80 panel as it didn't pass the essence judge on 80% of the papers within the guide's API budget. The guide model is fixed to be Fable 5 here.

each guided run by 10 bits improves the resulting upper bound on the expected number of unguided generations needed to produce an idea that passes the judge by roughly $1000\times$.

We instantiate the generator as a specialized agent scaffold that asks questions ranging from the broad topic of the paper to the key technical details of its target idea, and use Fable 5 as a fixed guide. We then evaluate five generators on 87 recent Top-1 Hugging Face Daily Papers (Hugging Face, n.d.). Our essence LLM judge requires the generator to propose a concrete idea that recovers the target paper's research direction and central research object (such as the core method).

Our framework yields much more efficient compression of research ideas than verbatim text compression using a language model or gzip. For GLM 5.3, the median compression cost under the Essence judge is 100.2 bits, compared with 1,147.1 bits for compressing the gold idea summary verbatim with the same model and 5,712 bits with gzip. We found that frontier models such as Fable 5.1 and Astra perform significantly better than the open-source model GLM 5.3, especially on papers that are harder to compress. Opus 5 also performs competitively with the best models on papers that are easier to compress. A uniform ensemble of Fable 5.1, Opus 5, and Astra (Section 2.3) further reduces the median compression cost, demonstrating headroom for further reductions. Our ablation between GLM 5.2 and GLM 5.3 shows that generator post-training can substantially reduce compression cost (Section 5).

2 Priced Guidance

Our goal is to quantify the cost of compressing a target idea, with equivalence evaluated by a fixed, possibly stochastic judge. Our protocol pairs a stochastic *generator*, which poses multiple-choice questions with preassigned answer probabilities, with a *guide*, which knows the target and guides generation solely by selecting offered answers. In *unguided generation*, the same generator instead samples answers using those probabilities. The Priced Guidance Theorem (Theorem 2.2) lower-bounds unguided acceptance probability using the compression cost, including a cost for the final idea's judge pass rate. Figure 1 illustrates this setting as a path through a tree of possible generations.

2.1 Priced Guidance Protocol

Let T be a target idea. A guided run under the priced generation protocol proceeds as follows.

1. **Loop.** For $t = 1, 2, \ldots$, until the generator submits a final idea instead of a bid:
   (a) **Bid.** Without access to T, the generator reasons and makes a bid. Let τ_t denote the trajectory up to step t: the generator's reasoning so far, its previous bids with their selected answers,

and its current bid. The generator's *bid* consists of a question, a finite set $\mathcal{A}_t(\tau_t)$ of answer choices called actions, and a probability $p_t(a \mid \tau_t)$ for each action a. The probabilities satisfy $p_t(a \mid \tau_t) > 0$ and $\sum_{a \in \mathcal{A}_t(\tau_t)} p_t(a \mid \tau_t) = 1$.

(b) **Select and pay.** The guide, which knows T, selects an action $a_t \in \mathcal{A}_t(\tau_t)$, and the generator pays $c_t = -\log_2 p_t(a_t \mid \tau_t)$.

(c) **Update.** The selected action a_t is appended to the trajectory. We write $g_t(\cdot \mid \tau_t, a_t)$ for the distribution of the next trajectory τ_{t+1}, which adds the generator's subsequent reasoning and its next bid or final idea.

2. **Judge.** The judge returns a possibly stochastic verdict $J_T(x) \in \{0, 1\}$ on the final idea x, with one denoting acceptance as equivalent to T. The judge's pass rate for x is $\Pr[J_T(x) = 1]$, where the probability is over the judge's randomness with x and T fixed. The generator pays $c_J(x \mid T) = -\log_2 \Pr[J_T(x) = 1]$.

An *unguided* run is the same loop, except that step (b) samples $a_t \sim p_t(\cdot \mid \tau_t)$ from the generator's own bid. The complete *trajectory* τ consists of all reasoning, bids, and selected answers, the final idea $x(\tau)$, and the judge's verdict $B(\tau) \in \{0, 1\}$. We write $|\tau|$ for its number of answered bids. From the guide, the generator receives only the selected actions. It never sees T or the judge's verdict, because the judge acts after the loop ends. Guided and unguided runs differ only in how step (b) selects the action. The Priced Guidance Theorem in Section 2.2 relies on this property.

The total compression cost of a trajectory τ is $K(\tau) = \sum_{t=1}^{|\tau|} c_t + c_J(x(\tau) \mid T)$. Equivalently, $2^{-K(\tau)} = \Pr[J_T(x(\tau)) = 1] \prod_t p_t(a_t \mid \tau_t)$. An answer costs more when the generator assigned it lower probability. A final idea costs more when the judge accepts it less often. We set $K = +\infty$, with $2^{-\infty} = 0$, when no final idea is returned or its pass rate is zero. A run succeeds when the judge accepts its final idea. The set of successful trajectories is $\mathcal{S}_T = \{\tau \mid B(\tau) = 1\}$. Trajectories with different final wordings or different sequences of questions can all belong to $\mathcal{S}_T$. Let P_{guide} and P_{unguide} denote the distributions of complete trajectories from guided and unguided runs, respectively. Section 2.2 lower-bounds the unguided success probability $P_{\text{unguide}}(\mathcal{S}_T)$ using the compression cost of guided runs.

2.2 Priced Guidance Theorem

The compression cost K has two interpretations, which we state as two theorems. Theorem 2.1 shows that K bounds the number of bits needed to describe a guided trajectory to the unguided generator. Theorem 2.2 shows that K lower-bounds the probability that the unguided generator produces an idea that the judge accepts.

Theorem 2.1 (Compression)**.** *For fixed generator randomness, a single prefix code encodes the actions the guide picks in every guided run τ, for any target and guide, using at most $K(\tau) + O(1)$ bits.*

Theorem 2.2 (Priced Guidance Theorem)**.** *For the stochastic generator and possibly stochastic guide and judge above, the acceptance probability of a fresh unguided run satisfies*

$$P_{\text{unguide}}(\mathcal{S}_T) \geq \mathbb{E}_{\tau \sim P_{\text{guide}}} \left[2^{-K(\tau)} \right] \geq 2^{-\mathbb{E}_{\tau \sim P_{\text{guide}}}[K(\tau)]}. \tag{1}$$

The expectation is over all guided runs, including generator and guide randomness. The terminal judge's randomness is averaged into its pass rate.

Both theorems rest on one observation: the compressed string is the sequence of actions that the guide picked, and with the generator's randomness fixed, $2^{-K(\tau)}$ is the probability that the unguided generator samples this same sequence of actions and passes the judge (Appendix A.3 gives the proofs). Appendix A.4 discusses the evidence lower bound (ELBO) interpretation of Theorem 2.2. Appendix A.5 shows that standard perplexity on the original text is a special case in which the generator predicts the next token.

2.3 Extensions: multiple candidates, recovery, and ensembling

We made some extensions to our framework. A generator that is unsure about the target should be able to brainstorm several candidate ideas at once. The guide we use is imperfect and sometimes

(a) Gold setting and main object

Category. Infrastructure / Resource (open skill repository, ontology, evaluation protocol, and toolkit).

Problem & motivation.

- Agents lack a unified mechanism to consolidate episodic experience into durable, reusable skills, so they repeatedly "reinvent the wheel."
- Existing skill repositories/marketplaces rely on manual curation and only indirect (downstream-task) quality signals, leaving skills fragmented, redundant, and unvalidated.

Main object & crux. SkillNet: an open full-lifecycle skill infrastructure that:

- Auto-creates skills from heterogeneous sources (execution trajectories, GitHub repos, documents, prompts) via LLM pipelines.
- Evaluates each skill intrinsically along Safety, Completeness, Executability, Maintainability, Cost-awareness.
- Connects skills into a typed relation graph (similar_to, belong_to, compose_with, depend_on).

Representation. Skills are treated as a unified knowledge representation bridging natural-language procedural guidance and machine-executable logic, organized by a three-layer Skill Ontology (taxonomy, relation graph, skill packages).

Release. Released as a curated repository (200k+ candidate, 150k+ curated skills), website/API, and the skillnet-ai Python library + CLI.

(b) Generator proposal: Directional to Essence

Category. A system or software artifact.

Core problem. Language agents cannot accumulate and reuse experience across tasks and across agents in a portable way — existing experience/skill memories are bespoke and siloed within individual agents or frameworks, so there is no general-purpose infrastructure for the community of agents to collectively store, organize, and reuse learned skills at trustworthy quality.

Main object. A skill library infrastructure — a general-purpose memory infrastructure / operating-system-like layer (a standardized API, protocol, or middleware for agent memory) whose primary function is an experience/skill library that stores past task trajectories or learned procedures for reuse on new tasks, and whose defining architectural form is a centralized shared hub/repository where many agents or users contribute and reuse skills collectively (hub/marketplace model), governed by LLM-as-judge curation — a quality-control, validation, and curation mechanism in which an LLM acts as judge over contributed skills, evaluating their correctness, usefulness, generality, and redundancy to decide which contributed skills enter or remain in the shared hub — and organized by a skill ontology: an organizational structure imposed on the hub's contents, i.e., a hierarchical taxonomy of skills with relations among them (specialization/generalization, dependencies, and composition of smaller skills into larger skills) that structures how skills are stored, related, abstracted, and navigated.

[. . .]

Intended outcome. By decoupling the skill library from any specific agent and exposing it as a centralized, standardized, LLM-curated, ontology-organized shared hub — analogous to a package registry or model hub, but for agent skills with automated quality gatekeeping and a structured taxonomy — the system enables collective, continual, trustworthy, and well-organized skill accumulation and cross-agent/cross-user skill reuse without any model weight updates.

Figure 3: SkillNet (arXiv:2603.04448). (a) Gold summary of the target idea. (b) Final generator proposal along the trajectory in Figure 4(b), with revisions from the Directional proposal in purple; [. . .] marks an omitted field. Appendix E gives the proposals and the gold summary in full.

fails to select the best action. We therefore strengthen it with *recovery*, which lets the guide return to an earlier question and try again. We also explore the option to *ensemble* several generators, because different generators may have better compression rate on different targets.

Multiple candidate ideas. Instead of a single final idea, the generator may submit k candidate ideas $x_1, \ldots, x_k$ with probabilities $p_1, \ldots, p_k$ that sum to one. The judge evaluates every candidate, and the judge cost of the submission is $c_J = -\log_2 \sum_{i=1}^{k} p_i \Pr[J_T(x_i) = 1]$. Theorem 2.2 still holds.

Recovery. To instantiate recovery, whenever the guide selects an action, the generator also outputs a distribution π over how many times the guide may roll out the generator again from that action. Each retry thus has a cost. Let j_t index the rollout that the guide keeps at step t of the returned trajectory. The recovery cost and bound are $K_\pi = K - \sum_t \log_2 \pi(j_t)$, and $P_{\text{unguide}}(\mathcal{S}_T) \geq \mathbb{E}[2^{-K_\pi}]$ where K is the compression cost of the returned trajectory. Appendix A.1 gives the formal construction.

Ensembling. We treat an ensemble of M generators as a single *meta harness* that selects member i with a fixed probability w_i first. Let K_i be member i's compression cost, including any recovery costs. Let P_{meta} denote the unguided trajectory law of the meta harness. Define the ensemble compression cost by $K_{\text{meta}} = -\log_2 \sum_{i=1}^{M} w_i 2^{-K_i}$. Then $P_{\text{meta}}(\mathcal{S}_T) \geq \mathbb{E}\big[2^{-K_{\text{meta}}}\big]$. Choosing member i costs only $-\log_2 w_i$ extra bits, so the ensemble succeeds whenever any member does. Our experiments use $w_i = 1/M$. Appendix A.2 gives the derivation.

3 Experiment Instantiation

We instantiate the protocol to predict the research contribution of a target paper. For each target, Claude Opus 5 generates a private gold summary from the paper's full LaTeX source. We use this summary rather than the full paper because the generator proposes only a research idea, not the paper's text or the exact results, so the guide and the judge need only a concise statement of that idea to compare against. The generator develops a candidate idea specifying a contribution category, a core problem, and a main research object. The guide uses the gold summary to guide this compression by

selecting answers to the generator's questions, with a compression cost for each selection. A Judge compares the candidate with the gold summary under a specified recognition criterion. Our target papers are Top-1 Hugging Face Daily Papers (Hugging Face, n.d.). After a title-based memorization screening (Appendix B.1), the development set has 40 papers from March 2 through April 30, 2026, and the test set has 87 papers from May 1 through September 6, 2026. Figure 3(a) shows the gold summary for SkillNet (arXiv:2603.04448), used in the example walkthrough below. Due to cost constraints, we constrain the generator budget to $\$50$, the guide budget to $\$15$, and the judge budget to $\$15$ for all models we tested. This provides a lower bound on the probability.

Idea-equivalence judging. We use LLM judges to compare the scientific content of a candidate idea with the gold summary while allowing different wording and names. We have two different judges with two levels of strictness. Under the *Directional* Judge, a candidate must identify the same research direction and contribution type, and describe a concrete, coherent idea of comparable research plausibility. Under the *Essence* judge, it must additionally recover the target's central research object and its defining mechanism or organizing principle. We use GPT-5.5 for both judges, choosing a model distinct from all generators to avoid having a model evaluate its own outputs. As an example, the generator proposal for SkillNet in Figure 3 passes the Essence judge without giving the exact name SkillNet or mentioning skills created from GitHub repositories. It passes because it recovers the central research object, a shared skill infrastructure, and its organizing principle, a skill ontology. A formal submission receives a fixed number n of independent Judge evaluations, of which l pass. We use l/n to estimate the pass rate $\Pr[J_T(x) = 1]$ in the judge cost c_J (Section 2.1), giving an empirical judge cost of $-\log_2(l/n)$ bits. A submission succeeds if $l \geq 1$. If all evaluations fail, its empirical cost is infinite and the submission is rejected, requiring recovery. We fix n, defaulting to 1 for the Directional judge and 3 for the Essence judge.

Generator scaffold. While the generator in the Priced Guidance (Section 2) framework can simply be a reasoning LLM placing arbitrary bids, we use a lightweight scaffold in which we provide the generators with some *strategies* to consider, which improves the baseline performance of all the models. A *strategy* is a fixed type of question sequence, such as asking for a missing keyword in the current draft idea, and a better scaffold will yield a lower compression cost and hence a tighter bound. The generator keeps a list of facts confirmed by the guide, a current candidate idea, and a whiteboard of working hypotheses and open questions, which the guide can also read (Appendix C.1). Unless otherwise stated, we use the highest supported reasoning effort for each generator model. For Opus 5, we use high effort because the generator yielded a lower median compression cost at high than at max effort on our dev set (Section 5). Each run proceeds as follows.

1. **Opening bids.** The generator walks down a fixed deep-learning taxonomy with 3,035 nodes one level at a time; each bid offers the children of the current node and a stop option. It then asks for one of 11 fixed contribution categories. All options have equal probability, and the guide selects an answer to each bid. GPT-5.6 Sol generated the taxonomy and the categories once, and every generator receives them for every target (Appendix H).
2. **Directional stage.** Repeat the following until a submission passes the Directional judge:
   (a) **Meta question.** The generator prepares one bid for each available strategy and assigns a probability to each strategy. The guide selects a strategy (Figure 4(a)).
   (b) **Strategy questions.** The generator asks the selected strategy's prepared question and any follow-up questions, and the guide selects an answer to each:
   - *Multiple choice* asks about one unresolved aspect of the candidate idea and offers concrete answers and rejection options.
   - *Keyword* asks which semantic category contains a missing concept, then offers candidate phrases in that category. It also offers a *letter hint* action: if none of the phrases is correct, the generator can guess the first two letters of the missing phrase, and after the guide selects them, it proposes phrases that begin with those letters.
   - *Submit* sends eight distinct candidate ideas with probabilities, which the generator prepares before the meta question, to the Directional judge.

   (c) **Update.** The generator records the selected answers as facts and revises its candidate idea and whiteboard.
3. **Essence stage.** Once a submission passes the Directional judge, the run is *promoted* to the Essence stage. The generator keeps its candidate idea, facts, and earlier answers, learns only that its idea passed, and now also has to recover the defining mechanism.

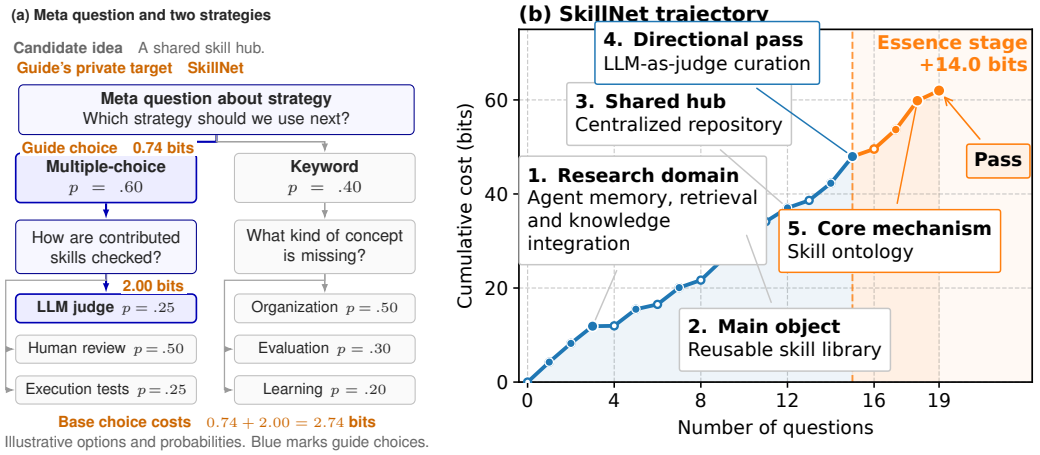

Figure 4: Illustration of the generator scaffold and one example trajectory. (a) A simplified meta question on the SkillNet example. At the start of a turn, the generator prepares both strategies' questions, answer options, and probabilities. The guide selects the multiple-choice strategy and then the answer. Answer sets and probabilities in (a) are illustrative. (b) Information acquired along one successful Fable 5 trajectory for SkillNet. Filled nodes identify paper-specific content; hollow nodes mark meta questions about strategies. Blue denotes the Directional stage. Orange denotes the additional 14.0 bits after promotion to Essence. The text boxes mark what the generator confirms.

(a) **Reused submission.** The eight-candidate submission that passed the Directional judge is first evaluated by the Essence judge. If one of the ideas passes, the run ends. Otherwise, the guide uses recovery (Section 2.3) to roll out the generator again from an earlier action.

(b) **Essence turns.** Repeat the Directional turn until a submission passes the Essence judge, with two changes:

- A new Essence *submit* bid sends one idea with probability one to the Essence judge.
- A new strategy, *correction*, lists claims in the candidate idea and asks which claim may contradict the target, then offers to replace, refine, or delete it.

Guide scaffold. The guide is a lightweight scaffold around Fable 5. For each bid, it sees the question, answer options, and probabilities, together with the target summary, the judge criterion, and its private memory. It selects an answer supported by the target, and only this choice reaches the generator. It may also privately preview Judge verdicts on the current submission snapshot to decide when to submit, and a submission then receives fresh Judge evaluations. This extra computation adds no compression cost, because only the selected answers are priced. To explore another path, or after a rejected submission, the guide can *check out* an earlier question, which restores the generator to its state right after that bid. Each step on the kept trajectory then pays the recovery cost defined in Section 2.3. Appendix C.2 describes the guide scaffold in detail.

Example walkthrough. For SkillNet (arXiv:2603.04448), Figure 4(b) follows the final active path of a Fable 5 generation trajectory. Figure 3(b) shows the final Essence proposal, with additions and revisions from the Directional proposal highlighted in purple, next to the gold summary in Figure 3(a). Appendix E contains both original proposals and the gold summary without omissions.

4 Main Results

We show coverage as a function of compression cost, together with the median and 80th-percentile compression costs, on the 87 test papers under the Essence judge in Figure 2. Figure 9 shows the corresponding results under the Directional judge. We defer some results to Section B.3.

Compressing an idea's essence takes far fewer bits than its text. We care about the probability of generating an idea with the target's essence, not of reproducing the words of its summary, which is much stricter. Compressing the gold summary verbatim costs GLM 5.3 a median of 1,147.1 bits (Appendix B.4). We choose GLM 5.3 as it is the only one of our five generators with open weights. Its median compression cost under Priced Guidance is only 59.9 bits under the Directional judge and 100.2 bits under the Essence judge (Figures 2 and 9), more than an order of magnitude lower.

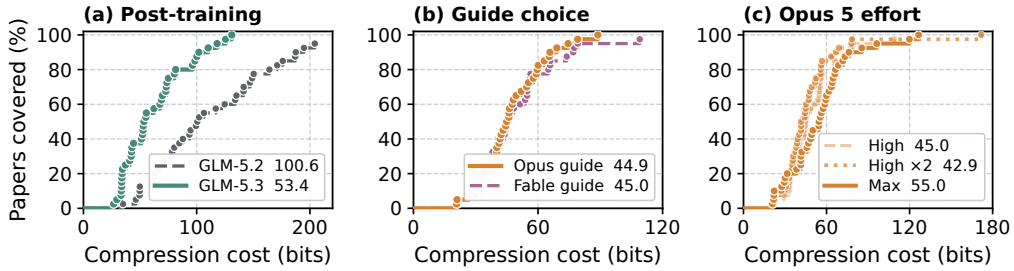

Figure 5: Directional compression cost on the development set. **(a)** Comparing GLM 5.2 and 5.3 shows the effect of general post-training of the generator on this benchmark. **(b)** Guide choice for Opus 5 ablations; both runs use the Opus 5 generator. **(c)** Opus 5 reasoning effort ablation. Legends report median compression cost in bits, using all 40 papers in (a,b) and successful runs in (c). The two-run ensemble in (c) averages the two members' observed scores as in Equation 12. Figure 11 shows the Fable 5.1 reasoning effort ablation.

Model comparisons. Opus 5 high has the lowest median cost under the Directional judge. Under the Essence judge, Fable 5.1, Opus 5 high, and Astra nearly tie at 67.8, 68.3, and 69.4 bits. To cover 80% of papers under the Essence judge, however, Fable 5.1 and Astra need 102.1 and 108.8 bits, compared with 138.9 bits for Opus 5 high (Figure 2(c)). The open-weight GLM 5.3 is competitive at the Directional median compared to Astra but covers less than 80% of papers under the Essence judge within the guide budget.

Where the information is spent. Keyword questions account for most of Opus 5 high's mean Directional cost and of its additional cost to reach Essence (Table 2). Letter hints are about a fifth of keyword selections but about half of keyword costs, suggesting that the needed concepts often lie outside the models' initial guesses. Reaching Essence costs only 13 more bits on average, suggesting that refining a vague idea into a concrete one is already very feasible (see examples in Appendix F).

Headroom from ensembling. Combining models lowers compression cost further, demonstrating headroom beyond the best individual model. A uniform mixture of Fable 5.1, Opus 5, and Astra (Appendix A.2) reaches median costs of 43.2 and 53.8 bits under the Directional and Essence judges, 4.3 and 14.0 bits below the best single models. The models thus provide complementary paths.

Compression cost did not change significantly over four months. One might expect compression cost to increase for papers published further beyond a model's knowledge cutoff. Ensemble cost shows no significant correlation with paper date over the test period (Figure 8).

5 Ablations

In this section, we present the ablations performed while designing the test scaffolding and choosing the set of models to test, using the 40-paper development set from Section 3. Some of the experiments in our development process discard earlier messages and reasoning between generator calls (Appendix C.14). We defer some of the results to Appendix B.7.

Generator post-training can substantially reduce compression cost. Updating the generator from GLM-5.2 to GLM-5.3 substantially reduces median compression cost, from 100.6 to 53.4 bits (Figure 5(a)). As these two models share the same pretraining base model, this demonstrates the effectiveness of post-training in reducing compression cost.

Guide choice has little effect on median compression cost for Opus 5. Replacing the Fable 5 guide with Opus 5 changes the median only from 45.0 to 44.9 bits (Figure 5(b)).

Ensembling helps more than higher reasoning effort. Raising Opus 5 from high to max effort raises its median from 45.0 to 55.0 bits, though coverage rises from 39 to 40 papers (Figure 5(c)). In contrast, ensembling two Opus 5 runs lowers the median by 3.4 bits under both judges (Figures 10(b) and 6(b)), suggesting that parallel inference-time scaling helps ideation more than serial scaling.

Median compression cost is robust to run-to-run randomness. Two independent Opus 5 runs have Directional medians of 44.8 and 46.4 bits on 38 papers and Essence medians 4.6 bits apart on

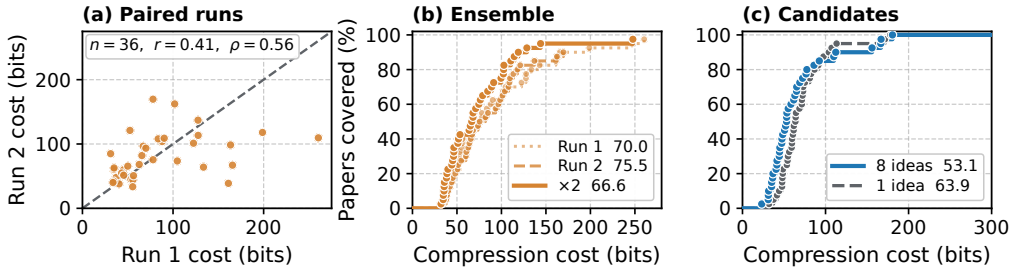

Figure 6: Run-to-run variation and ensembling under the Essence judge, and candidate count under the Directional judge, on the development set. **(a)** Paired Essence costs of two independent Opus 5 runs ($n = 36$). The dashed diagonal denotes equal cost; r and ρ are Pearson and Spearman correlations. **(b)** Coverage for both independent runs and their uniformly weighted ensemble, with effective cost given by Equation 12. While each paper may have stochastic compression cost, the median stays surprisingly stable. The randomness in the independent runs also explains why ensembling improves the median compression cost. **(c)** Native Codex Astra with one (gray dashed) or eight (blue solid) Directional candidates, evaluated by the Directional judge. Figure 10 shows the results under the other judge for each panel. Details are deferred to Appendix C.14.

36 papers, although individual-paper costs are only moderately correlated (Figures 10(a) and 6(a)). This per-paper randomness also explains why ensembling helps to provide a tighter bound.

More candidate ideas reduce Astra's Directional cost but not its Essence cost. We observe that single-idea GPT-family runs tended to repeat known facts. Therefore we let Astra in Codex submit eight Directional candidates and compare it to only submitting one idea (Section 2.3; Appendix C.14). Eight candidates lower the median Directional cost from 63.9 to 53.1 bits and lower the cost on 32 of 40 papers (Figure 6(c)). Under the Essence judge, whose later submissions contain one idea, the medians stay close at 73.1 and 72.7 bits (Figure 10(c)).

6 Limitations and Discussion

Our work provides a compression benchmark for research idea generation. A generator is scored by its compression cost on target ideas from future papers. There are many promising ways to improve the compression cost. For example, enabling experiments on real data and web search over a fixed snapshot of the internet could further reduce compression cost. Beyond idea generation, Priced Guidance provides a tool to quantify the probability of tail events for a family of agent scaffolds as long as the scaffold makes bids.

Our benchmark has several limitations. First, it relies on LLMs to summarize each target paper and to judge candidate ideas, so a generator could potentially hack the judge. We observe vague phrasing in some candidate ideas but no more serious judge hacking. Second, the benchmark measures whether a model can generate the target idea at least once across many attempts. Thus many generated ideas from the unguided generator can be wrong. Finally, automating research idea generation at scale could have substantial societal consequences. Our metric tracks progress in idea-generation capability and may help forecast when such automation becomes feasible.

7 Related Work

Idea-generation benchmarks. A growing line of work uses language models to propose scientific hypotheses and research directions through scaffolding (Wang et al., 2023; Baek et al., 2024; Zhao et al., 2025; Liu et al., 2026; Yang et al., 2025a;b; Zhao et al., 2026; Wang et al., 2025; Su et al., 2025; Gottweis et al., 2025). LiveIdeaBench tests idea forecasting from keywords (Ruan et al., 2024), while GIANTS asks models to anticipate a target insight from two parent papers (He-Yueya et al., 2026). Our work differs by quantifying a target idea's compression cost as an intrinsic metric.

Other related work. Closest to our setting, Agentic-Ideation uses guidance from a target paper to train idea generators (Zhao et al., 2026). Our method is an instantiation of online algorithms with advice, which measure the information communicated by a knowledgeable guide (Boyar et al., 2017). Our evaluation complements systems and benchmarks for automated research execution (Lu et al., 2024; Chen et al., 2024). Appendix G provides a fuller discussion.

AI USE STATEMENT

In this work, we used generative AI tools to assist with theoretical development and proof writing, code and prompt development, data preparation, and drafting and editing the manuscript. As described in the main text and appendices, language models also served as generators, guides, and judges in our experiments.

The authors reviewed all proofs, read the implementation code, and inspected samples of the generated data. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The assumptions and proofs underlying Priced Guidance are provided in Section 2 and Appendix A. Section 3 describes the experimental protocol, and Appendix B documents paper selection, memorization screening, and additional experimental details. Appendix C provides prompt excerpts and agent protocol details. The anonymous supplementary materials include the experiment implementation, prompts and configurations, target papers' gold summaries, recorded measurements, and scripts for regenerating the empirical figures and checking baseline costs. The supplement's README gives installation and execution instructions and explains which historical records are available; its figure-data guide maps figures to their data and scripts. These materials support offline reproduction of the reported figures and fresh experiments with the required model access, although hosted model availability, provider updates, and stochastic outputs may prevent exact replication of live runs.

REFERENCES

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. ResearchAgent: Iterative research idea generation over scientific literature with large language models. *arXiv preprint arXiv:2404.07738*, 2024.

Hasan Baig, Hailey Joren, and Joe Benton. TASTE: Can AI models judge AI safety research proposals? Technical report, Anthropic, August 2026. URL https://www-cdn.anthropic.com/files/4zrzovbb/website/dd5feddcb3b7d20aadda6af4093ac1fb0c9d419e.pdf.

Joan Boyar, Lene M. Favrholdt, Christian Kudahl, Kim S. Larsen, and Jesper W. Mikkelsen. Online algorithms with advice: A survey. *ACM Computing Surveys*, 50(2):19:1–19:34, 2017. doi: 10.1145/3056461.

Shaolong Chen, Yanlin Fei, Nazhou Liu, Xinmiao Yu, Lei Li, Rahul Thapa, Madalina Ciobanu, Navan Preet Singh, Qingqing Mao, and Ritankar Das. Reconstruction: A blind benchmark for recovering research ideas from pre-publication bibliographies, 2026. URL https://arxiv.org/abs/2608.16645.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, et al. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. *arXiv preprint arXiv:2410.05080*, 2024.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jznbgiynus.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, et al. Accelerating scientific discovery with co-scientist. *arXiv preprint arXiv:2502.18864*, 2025.

Sikun Guo, Amir Hassan Shariatmadari, Guangzhi Xiong, Albert Huang, Eric Xie, Stefan Bekiranov, and Aidong Zhang. IdeaBench: Benchmarking large language models for research idea generation. *arXiv preprint arXiv:2411.02429*, 2024.

Joy He-Yueya, Anikait Singh, Ge Gao, Michael Y. Li, Sherry Yang, Chelsea Finn, Emma Brunskill, and Noah D. Goodman. GIANTS: Generative insight anticipation from scientific literature. *arXiv preprint arXiv:2604.09793*, 2026.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. *arXiv preprint arXiv:2310.03302*, 2023.

Hugging Face. Daily Papers, n.d. URL https://huggingface.co/papers. Accessed September 8, 2026.

Bo Jiang. HindSight: Evaluating LLM-generated research ideas via future impact, 2026. URL https://arxiv.org/abs/2603.15164.

Sandeep Juneja and Perwez Shahabuddin. Rare-event simulation techniques: An introduction and recent advances. In *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pp. 291–350. Elsevier, 2006. doi: 10.1016/S0927-0507(06)13011-X.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C., Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kexin Chen, Peng Chen, Ruijue Chen, Wentao Chen, Xin Chen, Yang Chen, Yanru Chen, Yifei Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Dazhi Cheng, Yean Cheng, Jialei Cui, Jingbing Cui, Anqi Dai, Jiaqi Deng, Hao Ding, Rui Ding, Shaofeng Ding, Mengfan Dong, Mengnan Dong, Yuhao Dong, Yuxin Dong, Angang Du, Chenzhuang Du, Dikang Du, Jusen Du, Yulun Du, Yu Fan, Jing Feng, Qiulin Feng, Yichen Feng, Kelin Fu, Qiang Fu, Fuxuan Gao, Hongcheng Gao, Jingyue Gao, Tong Gao, Weijia Gao, Shangyi Geng, Jie Gong, Linhu Gong, Shengao Gong, Xiaochen Gong, Qizheng Gu, Yicheng Gu, Shuhao Guan, Haiqing Guo, Shiqi Guo, Xiang Guo, Zhengyan Guo, Beixi Hao, Wenxin Hao, Xiaoru Hao, Dailan He, Haotian He, Lehan He, Qi He, Weiran He, Xinran He, Xinyi He, Yibo He, Yunjia He, Chao Hong, Tiange Hong, Hao Hu, Jiaxi Hu, Ruikun Hu, Weiming Hu, Yangyang Hu, Zhenxing Hu, Liang Hua, Jinbin Huang, Ke Huang, Ruiyuan Huang, Siying Huang, Weixiao Huang, Yan Huang, Zhengjie Huang, Zhiqi Huang, Yulong Hui, Chaobo Jia, Yutong Jiang, Zhejun Jiang, Zuoyou Jiang, Wenyi Jin, Xinyi Jin, Yu Jing, Huanjun Kong, Guokun Lai, Aidi Li, Cheng Li, Chengyuan Li, Cong Li, Fang Li, Guanyu Li, Haoyang Li, Jia Li, Junxiong Li, Lei Li, Letian Li, Lincan Li, Weihong Li, Wentao Li, Xintong Li, Yang Li, Yishen Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Zhengxiao Li, Zhiyuan Li, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zichao Lin, Ziyan Lin, Bill Liu, Boxiao Liu, Chuan Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yipeng Liu, Zhengying Liu, Zhiheng Liu, Enzhe Lu, Haoyu Lu, Linqiang Lu, Tingzhan Lu, Zhiyuan Lu, Aotian Luo, G. Luo, Junyu Luo, Yifan Luo, B. Lyu, Wenzhou Lyu, Shaoguang Mao, Yuan Mei, Xin Men, Minqing Ni, Yixuan Niu, Siyuan Pan, Shujun Peng, Zhangyang Qi, Ruoyu Qin, ZeChao Qin, Zeyu Qin, Haiquan Qiu, Jianxin Qiu, Jiezhong Qiu, Bowen Qu, Yuhao Qu, Zeyu Shang, Youbo Shao, Han Shen, Jincheng Shi, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Wingchun Siu, Pengwei Song, Xiaoxi Song, Jianlin Su, Yunfeng Su, Zhaochen Su, Lin Sui, Jingsong Sun, Junyao Sun, Shaoning Sun, Shuzhe Sun, Tongyu Sun, Yujun Sun, Yunpeng Tai, Chuning Tang, Heyi Tang, Sirui Tang, Zecheng Tang, Chaoran Tian, Rongpeng Tian, Yu Tian, Wei Tu, Chensi Wang, Chuang Wang, Chunjie Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Hao Wang, Huaqing Wang, Hui Wang, Jiayi Wang, Jinglong Wang, Jinhong Wang, Jiuzheng Wang, Linian Wang, Shaobo Wang, Shenzhi Wang, Shuyi Wang, Si Wang, Siyuan Wang, Tianfu Wang, Wenjue Wang, Xingran Wang, Xinmei Wang, Xinyuan Wang, Xusheng Wang, Yalin Wang, Yangkun Wang, Yao Wang, Yaoyu Wang, Yejie Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhenhao Wang, Zhongsheng Wang, Zifan Wang, Chu Wei, Ming Wei, Shouxin Wei, Zichen Wen, Fan Wu, Haoning Wu, Rucong Wu, Wenhao Wu, Xiaoxue Wu, Yingcong Wu, Yongqi Wu, Yuxin Wu, Zijian Wu, Xinglang Xian, Chenxuan Xiang, Yuye Xiang, Bocheng Xiao, Chenjun Xiao, Xin Xiao, Jin Xie, Xiaotong Xie, Yifeng Xie, Zhe Xie, Bowei Xing, Yiming Xiong, Baosheng Xu, Boyu Xu, Jiale Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Qingtao Xu, Shuyao Xu, Suting Xu, Tiantian Xu, Tianxiang Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ye Xu, Yueni Xu, Ziyao Xu, Haonan Xue, Junjie Yan, Yaoyao Yan, Fan Yang, Guangyao Yang, Hao Yang, Junwei Yang, Ruoyu Yang, Wenjie Yang, Xiaofei Yang, Xinyu Yang, Yi Yang, Yiling Yang, Ying Yang, Yuchen Yang, Zhen Yang, Zhilin Yang, Zian Yang, Zuhao Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhanbo Ye, Bohong Yin, Haoxiang Yin, Xietong Yin,

Chengzhen Yu, Haozhen Yu, Longhui Yu, Shengnan Yu, Shuying Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Tongtian Yue, Wei Yue, Yang Yue, Dunyuan Zha, Haobing Zhan, B. H. Zhang, Dehao Zhang, Fei Zhang, Hao Zhang, Haoyuan Zhang, Huanyu Zhang, Jiapei Zhang, Jiaxuan Zhang, Jin Zhang, Kaiyi Zhang, Miaozhen Zhang, Puqi Zhang, Qinglei Zhang, Rong Zhang, Rui Zhang, Shaoshuai Zhang, Shiyi Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yangkun Zhang, Ye Zhang, Yichi Zhang, Yikun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Zijing Zhang, Bin Zhao, Chenguang Zhao, Feifan Zhao, Jinglun Zhao, Jinxiang Zhao, Shuai Zhao, Wenshuo Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Haozhi Zheng, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Haofeng Zhong, Lei Zhong, Longguang Zhong, M. Zhou, Qiankang Zhou, Runjie Zhou, Ruozhang Zhou, Xinyu Zhou, Yiqiao Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yangjunfeng Zhu, Yuxuan Zhu, Zhen Zhu, Chen Zhuang, Weiyu Zhuang, and Xinxing Zu. Kimi K3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653.

Mario Krenn, Lorenzo Buffoni, Bruno Coutinho, Sagi Eppel, Jacob Gates Foster, et al. Predicting the future of AI with AI: High-quality link prediction in an exponentially growing knowledge network. *arXiv preprint arXiv:2210.00881*, 2022.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

Fenghai Li, Zihan Tang, Haofei Yu, Yining Zhao, and Jiaxuan You. Can large language models forecast what researchers study next?, 2026a. URL https://arxiv.org/abs/2609.00747.

Yafu Li, Runzhe Zhan, Haoran Zhang, Shunkai Zhang, Yizhuo Li, Zhilin Wang, Jiacheng Chen, Futing Wang, Xuyang Hu, Yuchen Fan, Bangjie Xu, Yucheng Su, Xinmiao Han, Chenxi Li, Haodi Lei, Yufeng Zhao, Zejin Lin, Qianjia Cheng, Tong Zhu, Xiaoye Qu, Ganqu Cui, Peng Ye, Yun Luo, Zhouchen Lin, Yu Qiao, Bowen Zhou, Ning Ding, and Yu Cheng. Achieving gold-medal-level olympiad reasoning via simple and unified scaling, 2026b. URL https://arxiv.org/abs/2605.13301.

Yujie Liu, Zonglin Yang, Tong Xie, Jinjie Ni, Ben Gao, Yuqiang Li, Shixiang Tang, Wanli Ouyang, Erik Cambria, and Dongzhan Zhou. ResearchBench: Benchmarking LLMs in scientific discovery via inspiration-based task decomposition, 2026. URL https://arxiv.org/abs/2503.21248.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, et al. DiscoveryBench: Towards data-driven discovery with large language models. *arXiv preprint arXiv:2407.01725*, 2024.

OpenAI. New tools and features in the Responses API. OpenAI Blog, May 2025. URL https://openai.com/index/new-tools-and-features-in-the-responses-api/.

Yansheng Qiu, Haoquan Zhang, Zhaopan Xu, Ming Li, Diping Song, Zheng Wang, and Kaipeng Zhang. AI Idea Bench 2025: AI research idea generation benchmark. *arXiv preprint arXiv:2504.14191*, 2025.

Kai Ruan, Xuan Wang, Jixiang Hong, Peng Wang, Yang Liu, and Hao Sun. Evaluating LLMs' divergent thinking capabilities for scientific idea generation with minimal context. *arXiv preprint arXiv:2412.17596*, 2024.

Jürgen Schmidhuber. Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Computational Creativity: An Interdisciplinary Approach*, volume 9291 of *Dagstuhl Seminar Proceedings*, pp. 1–35. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2009. doi: 10.4230/DagSemProc.09291.14. URL https://drops.dagstuhl.de/entities/document/10.4230/DagSemProc.09291.14.

Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010. URL https://people.idsia.ch/~juergen/ieeecreative.pdf.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can LLMs generate novel research ideas? a large-scale human study with 100+ NLP researchers. *arXiv preprint arXiv:2409.04109*, 2024.

Chenglei Si, Tatsunori Hashimoto, and Diyi Yang. The ideation-execution gap: Execution outcomes of LLM-generated versus human research ideas, 2025. URL https://arxiv.org/abs/2506.20803.

Chenglei Si, Zitong Yang, Yejin Choi, Emmanuel Candès, Diyi Yang, and Tatsunori Hashimoto. Towards execution-grounded automated AI research, 2026. URL https://arxiv.org/abs/2601.14525.

Soumitra Sinhahajari, Navonil Majumder, and Soujanya Poria. On the limits of LLM-as-judge for scientific novelty assessment. *arXiv preprint arXiv:2606.12071*, 2026.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, et al. PaperBench: Evaluating AI's ability to replicate AI research. *arXiv preprint arXiv:2504.01848*, 2025.

Haoyang Su, Renqi Chen, Shixiang Tang, Zhenfei Yin, Xinzhe Zheng, Jinzhe Li, Biqing Qi, Qi Wu, Hui Li, Wanli Ouyang, Philip Torr, Bowen Zhou, and Nanqing Dong. Many heads are better than one: Improved scientific idea generation by a LLM-based multi-agent system, 2025. URL https://arxiv.org/abs/2410.09403.

Guan Wang, Changling Liu, Chenyu Wang, Cai Zhou, Yuhao Sun, Yifei Wu, Shuai Zhen, Luca Scimeca, and Yasin Abbasi Yadkori. HRM-Text: Efficient pretraining beyond scaling, 2026a. URL https://arxiv.org/abs/2605.20613.

Heng Wang, Pengcheng Jiang, Jiashuo Sun, Zhiyi Shi, Haofei Yu, Jiawei Han, and Heng Ji. Learning to predict future-aligned research proposals with language models. *arXiv preprint arXiv:2603.27146*, 2026b.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. SciMON: Scientific inspiration machines optimized for novelty. *arXiv preprint arXiv:2305.14259*, 2023.

Wenxiao Wang, Lihui Gu, Liye Zhang, Yunxiang Luo, Yi Dai, Chen Shen, Liang Xie, Binbin Lin, Xiaofei He, and Jieping Ye. SciPIP: An LLM-based scientific paper idea proposer, 2025. URL https://arxiv.org/abs/2410.23166.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. Self-preference bias in LLM-as-a-judge. *arXiv preprint arXiv:2410.21819*, 2024.

Hjalmar Wijk, Tao Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, et al. RE-Bench: Evaluating frontier AI R&D capabilities of language model agents against human experts. *arXiv preprint arXiv:2411.15114*, 2024.

Gabriel Wu and Jacob Hilton. Estimating the probabilities of rare outputs in language models. *arXiv preprint arXiv:2410.13211*, 2024.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *arXiv preprint arXiv:2504.08066*, 2025.

Zonglin Yang and Lidong Bing. MOOSE-Star: Unlocking tractable training for scientific discovery by breaking the complexity barrier, 2026. URL https://arxiv.org/abs/2603.03756.

Zonglin Yang, Wanhao Liu, Ben Gao, Yujie Liu, Wei Li, Tong Xie, Lidong Bing, Wanli Ouyang, Erik Cambria, and Dongzhan Zhou. MOOSE-Chem2: Exploring LLM limits in fine-grained scientific hypothesis discovery via hierarchical search, 2025a. URL https://arxiv.org/abs/2505.19209.

Zonglin Yang, Wanhao Liu, Ben Gao, Tong Xie, Yuqiang Li, Wanli Ouyang, Soujanya Poria, Erik Cambria, and Dongzhan Zhou. MOOSE-Chem: Large language models for rediscovering unseen chemistry scientific hypotheses, 2025b. URL https://arxiv.org/abs/2410.07076.

Keyu Zhao, Lingyan Kong, Fengli Xu, and Yong Li. Agentic-ideation: Sample efficient agentic trajectories synthesis for scientific ideation agents, 2026. URL https://arxiv.org/abs/2606.31229.

Xinran Zhao, Boyuan Zheng, Chenglei Si, Haofei Yu, Ken Liu, Runlong Zhou, Ruochen Li, Tong Chen, Xiang Li, Yiming Zhang, and Tongshuang Wu. The ramon llull's thinking machine for automated ideation, 2025. URL https://arxiv.org/abs/2508.19200.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, et al. Judging LLM-as-a-judge with MT-Bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

A Extensions and Proofs for the Priced Guidance Theorem

A.1 Recovery

Recovery allows the guide to restore the generator to the information it had at a previously reached question, including that question's committed bid, and request further generation from there. This lets the guide reconsider earlier choices after an unhelpful generation or a rejected final idea. The sequence of trajectory prefixes and actions currently retained by the run is its *active path*. Restoration replaces this path with the saved prefix ending at the restored question. The guide can remember the discarded exploration, while the generator continues from the restored information.

Repeated requests from the same saved generator state need distinct labels. A *checkpoint* τ_t records the selection-time trajectory prefix, including its committed bid and the generator information needed to continue. We assume countably many possible checkpoints, as for finite digital records. A *continuation draw* records the next trajectory prefix and, when it contains a final idea, one fresh judge verdict. For an offered action a, write $g_T(\cdot \mid \tau_t, a)$ for this joint law, combining the generator kernel g_t with the fixed judging rule. Its *occurrence index* j counts requests of action a from that exact checkpoint, starting at one. Model the possible next prefixes by jointly independent draws $X_{\tau_t,a,j}$ satisfying

$$X_{\tau_t,a,1}, X_{\tau_t,a,2}, \ldots \overset{\text{iid}}{\sim} g_T(\cdot \mid \tau_t, a). \tag{2}$$

Let τ_1 be the initial checkpoint, drawn independently of these continuations from the generator's target-blind initialization distribution. Write $X = (\tau_1, (X_{\tau_t,a,j})_{\tau_t,a,j})$ for this complete collection of generator and judge randomness. Each new request reveals the next unused entry for its checkpoint–action pair. Restoring a checkpoint does not reset the index, and matching question text alone does not identify the same checkpoint. Reusing a terminal occurrence retains its original verdict. A fresh evaluation of the same idea requires a new indexed request under an offered evaluation action at a checkpoint fixing that idea; a probability-one evaluation action has zero answer cost.

A *recovery policy* specifies the guide's requests, restorations, and stopping rule. Condition on any private guide randomness so that these decisions are determined by the target and the draws revealed so far. When the policy stops after finitely many requests, let L be the number of actions on its returned active path. Its *indexed active record* is

$$\widehat{\tau}^\star(X) = (\tau_1, a_1, j_1, \tau_2, \ldots, a_L, j_L, \tau_{L+1}), \qquad \tau_{t+1} = X_{\tau_t,a_t,j_t}. \tag{3}$$

Here τ_1 is the initial checkpoint, and the last prefix τ_{L+1} contains the final idea and its judge verdict. Dropping the occurrence indices gives the public trajectory $\tau^\star(X) = \mathrm{pub}(\widehat{\tau}^\star(X))$. Each retained index j_t may identify any previously revealed draw consistent with this record; it need not be the most recent one.

Consider a generator that starts at a fixed checkpoint τ_1 and offers only one action, *generate a final idea*, with probability one and hence zero answer cost. The generator draws either x_+ or x_- with equal probability. A deterministic judge accepts x_+ and rejects x_-, so the total compression cost is $K = 0$ for x_+ and $K = +\infty$ for x_-. Without recovery, guided and unguided runs have the same law, giving

$$\mathbb{E}[2^{-K}] = \tfrac{1}{2} \cdot 1 + \tfrac{1}{2} \cdot 0 = \tfrac{1}{2} = P_{\text{unguide}}(\mathcal{S}_T).$$

Now let the guide retry once after a rejection and return the second candidate. If this extra choice carries no cost, the returned candidate has $K = 0$ with probability $3/4$ and $K = +\infty$ otherwise, giving

$$\mathbb{E}[2^{-K}] = \tfrac{3}{4} > \tfrac{1}{2} = P_{\text{unguide}}(\mathcal{S}_T).$$

Thus the judge cost alone does not account for selecting among repeated attempts. We must also add a cost for the retained continuation's occurrence index.

Choose a target-independent distribution π on the positive integers before the target is revealed, with

$$\pi(j) > 0, \qquad \sum_{j=1}^{\infty} \pi(j) = 1. \tag{4}$$

The same distribution applies at each checkpoint–action pair. For a retained step (τ_t, a_t, j_t), the cost covers both its action and its occurrence,

$$\begin{aligned} c_\pi(\tau_t, a_t, j_t) &= -\log_2 p_t(a_t \mid \tau_t) - \log_2 \pi(j_t), \\ C_\pi(\widehat{\tau}^\star) &= \sum_{t=1}^{L} c_\pi(\tau_t, a_t, j_t), \\ K_\pi(\widehat{\tau}^\star) &= C_\pi(\widehat{\tau}^\star) + c_J(x(\widehat{\tau}^\star) \mid T). \end{aligned} \tag{5}$$

Here $x(\widehat{\tau}^\star)$ is the final candidate in the returned record. Only steps on the returned active path contribute to C_π; the total K_π also includes that candidate's judge cost. The extra cost $-\log_2 \pi(j_t)$ prices the retained continuation's occurrence index. Requests made during discarded exploration still count toward the indices of later draws from the same checkpoint–action pair. An already revealed draw keeps its original index, even if the guide explores further before returning to it.

Theorem A.1 (Online indexed recovery)**.** *Assume the countable checkpoint model and independent continuation draws defined above, with a common target-blind initialization distribution for guided and unguided runs. Fix a guide recovery policy as in Equation 3, conditioning on any private guide randomness. Suppose it makes finitely many continuation requests almost surely. It may restore any previously reached checkpoint and return any consistent indexed active record $\widehat{\tau}^\star(X)$. Fix π as in Equation 4, and let P_{unguide} be the law of a fresh unguided execution with the same generator and judge. Then*

$$P_{\text{unguide}}(\mathcal{S}_T) \geq \mathbb{E}_X \left[2^{-K_\pi(\widehat{\tau}^\star(X))} \right]. \tag{6}$$

The expectation includes initialization, continuation, and judge randomness.

To prove the theorem, construct an unguided simulator that uses the same initial checkpoint τ_1. At each reached prefix $\tau_{\text{sim},t}$, it samples an action $A_t \sim p_t(\cdot \mid \tau_{\text{sim},t})$ and an independent occurrence index $J_t \sim \pi$, then takes $X_{\tau_{\text{sim},t}, A_t, J_t}$ as its next prefix. A forward execution never revisits an exact checkpoint because its trajectory grows at every step. Conditional on the simulator's past and its current action A_t, each unused draw at $\tau_{\text{sim},t}$ still has distribution $g_T(\cdot \mid \tau_{\text{sim},t}, A_t)$. Averaging over the sampled index therefore gives the same generation and judging law. Together with the common initialization distribution, this makes the simulator's public trajectory follow the fresh unguided law P_{unguide}. Now condition on the complete generator and judge randomness X. The recovery policy's returned record is fixed, and the probability that the simulator matches every action and index in that record is

$$\prod_{t=1}^{L} p_t(a_t \mid \tau_t)\pi(j_t) = 2^{-C_\pi(\widehat{\tau}^\star(X))}. \tag{7}$$

Let x_{sim} be the simulator's final candidate and x^* the candidate in the returned record. On a match, $x_{\text{sim}} = x^*$. Since the pass rate is nonnegative, the conditional expected pass rate obeys

$$\mathbb{E}[\Pr[J_T(x_{\text{sim}}) = 1] \mid X] \geq 2^{-C_\pi(\widehat{\tau}^\star(X))} \Pr[J_T(x^*) = 1] = 2^{-K_\pi(\widehat{\tau}^\star(X))}. \tag{8}$$

The simulator has the fresh unguided law, so averaging over X gives Theorem A.1. This argument remains valid when the guide uses previous judge verdicts to select the returned candidate. If the guide uses private randomness, average over it as well. Appendix A.6 gives the formal proof.

For a fixed target and policy, a finite expected total cost gives

$$P_{\text{unguide}}(\mathcal{S}_T) \geq \mathbb{E}_X\left[2^{-K_\pi}\right] \geq 2^{-\mathbb{E}_X[K_\pi]}. \tag{9}$$

The expectation includes all returned candidates, with the judge pass rate already accounted for in K_π. Conditioning only on successful runs would change this expectation and does not justify dropping the other runs.

A concrete choice of π can keep the first occurrence inexpensive while assigning positive probability to every later attempt. We fix $\varepsilon = 0.05$ and use

$$\pi(j) = \begin{cases} 1-\varepsilon, & j = 1, \\ \dfrac{\varepsilon}{j(j-1)}, & j \geq 2. \end{cases} \tag{10}$$

This assigns mass 0.95 to the first occurrence and reserves total mass 0.05 for later occurrences. Normalization follows from $1/[j(j-1)] = 1/(j-1) - 1/j$, whose sum over $j \geq 2$ is one. The first occurrence costs an additional $-\log_2(0.95) \approx 0.074$ bits, including at checkpoints that are never revisited. An occurrence $j \geq 2$ instead incurs an extra cost of $\log_2\big(j(j-1)/0.05\big)$ bits.

In experiments, we run a fixed number n of independent judge evaluations for each submitted candidate. If l evaluations pass, the empirical judge cost is $-\log_2(l/n)$. A zero fraction gives infinite total cost and a zero score. The same expectation bound holds for this empirical cost when each batch of judgments is included in its indexed continuation draw. The proof uses the batch's pass fraction as its nonnegative terminal score, whose expectation under fresh unguided generation is the single-evaluation acceptance probability. It does not assume that a batch selected after retries is an unbiased estimate of that selected candidate's pass rate. Appendix A.6 gives the argument. The resulting 2^{-K} is an observed contribution to an expectation bound; a finite-sample probability guarantee still requires accounting for statistical uncertainty.

A.2 Ensembling Models and Seeds

Fix M generator configurations, which may differ in model or generator seed, and weights $w_i > 0$ with $\sum_i w_i = 1$ before the target is revealed. A *meta harness* first samples I with $\Pr(I = i) = w_i$, then runs only member I unguided and evaluates its output on target T under the common success criterion. Let $P_i(\mathcal{S}_T)$ and $P_{\text{meta}}(\mathcal{S}_T)$ denote the member's and meta harness's unguided success probabilities, respectively. For member i, let K_i be its total compression cost, including the judge cost and any occurrence costs. If each member's guided and unguided executions satisfy Theorem 2.2 or Theorem A.1, conditioning on the initial choice and applying the member bounds gives

$$P_{\text{meta}}(\mathcal{S}_T) = \sum_{i=1}^{M} w_i P_i(\mathcal{S}_T) \geq \mathbb{E}\left[\sum_{i=1}^{M} w_i 2^{-K_i}\right]. \tag{11}$$

Linearity of expectation does not require independence between members' guided runs.

For both development and test ensembles, we use uniform weights $w_i = 1/M$. Assign its empirically measured total cost $\widetilde{K}_i$ to a successful member and $\widetilde{K}_i = +\infty$ to a failed or missing member, which contributes zero score. Define the ensemble's observed effective cost by

$$K_{\text{meta}} = -\log_2\left(\frac{1}{M}\sum_{i=1}^{M} 2^{-\widetilde{K}_i}\right). \tag{12}$$

The denominator remains M when members fail or are missing. The weights already account for the initial choice, so no further selection cost is added. The score $2^{-K_{\text{meta}}}$ is an observed contribution to the expectation bound in Equation 11; a finite-sample probability guarantee additionally requires accounting for statistical uncertainty.

A.3 Conditional compression and the Priced Guidance Theorem

The proof treats the generator's target-independent randomness as shared side information. Once this randomness is fixed, the guide's answers determine a path through a deterministic tree. We first construct a code for this path, then relate its length to unguided acceptance probability.

Randomness as side information. Assume trajectory prefixes lie in standard Borel spaces. The common initialization law μ and generator transition kernels g_t can be represented by measurable maps F_0, F_t and independent uniform random variables $U_0, U_1, \ldots$ on $[0,1]$,

$$\tau_1 = F_0(U_0), \qquad \tau_{t+1} = F_t(\tau_t, a_t, U_t). \tag{13}$$

Write $R = (U_0, U_1, \ldots)$ for the generator's random tape. Its law is independent of the target. Use an independent random tape S to realize the guide's possibly stochastic choices. Unguided answer sampling and the terminal judge use separate randomness, independent of R and S. The maps F_t are defined for every trajectory prefix and offered answer, so fixing R specifies the generator's behavior along alternative answer paths as well as the guided path. This is a theoretical coupling and does not require the model API to expose its internal random tape.

Coding the answers. For fixed $R = r$, let $P_{\mathrm{unguide},r}$ denote the unguided law with only answer sampling and the terminal judge left random. Each trajectory prefix and bid can be computed from r and the preceding answers. If $a = (a_1, \ldots, a_m)$ is a complete answer sequence ending in a final idea, its conditional probability is

$$P_{\mathrm{unguide},r}(\widetilde{A} = a) = \prod_{t=1}^{m} p_t(a_t \mid \tau_t(r, a_{<t})) = 2^{-C(r,a)}, \tag{14}$$

where $\widetilde{A}$ denotes the complete unguided answer sequence and $C(r,a)$ is its total answer cost. In particular,

$$C(r,a) = -\log_2 P_{\mathrm{unguide},r}(\widetilde{A} = a). \tag{15}$$

Different complete answer sequences cannot be proper prefixes of one another. The decoded generator state determines whether another answer is requested, so a terminal sequence cannot also require a continuation. Any guide decision that changes this state must be communicated through an offered answer. No separate message length or stopping signal is needed.

For completeness, an ideal arithmetic code gives a binary realization of this codelength. Start with $[0,1)$ and partition the current interval at each bid according to its answer probabilities. The interval I_a for a complete sequence a has length $2^{-C(r,a)}$, and intervals for distinct complete sequences are disjoint. Set $\ell = \lceil C(r,a) \rceil + 1$. Since $2^{-\ell} \le |I_a|/2$, the interval I_a contains an interval $[k2^{-\ell}, (k+1)2^{-\ell})$ for some integer k. Its ℓ-bit binary address specifies a to a decoder that knows r. These binary intervals are disjoint, so their addresses form a prefix code. The code length satisfies

$$\ell = \lceil C(r,a) \rceil + 1 < C(r,a) + 2. \tag{16}$$

The two extra bits are the overhead of turning the answer probabilities into a binary prefix code: one from rounding $C(r,a)$ up to an integer and one from aligning the interval I_a to a binary address. This construction uses the generator's probabilities to encode the guide's realized answers. The guide's own random tape is not needed by the decoder, and C need not equal the guide's entropy or mutual information.

Proof of the Priced Guidance Theorem. Fix $R = r$ and $S = s$ for which the guided run terminates with a final idea. Let $a(r,s)$ be its selected-answer sequence, let $x(r,s)$ be its final idea, and write $C(r,s) = C(r, a(r,s))$ for its answer cost. Replaying $a(r,s)$ reconstructs the same generator trajectory and final idea. Fresh terminal judge randomness is independent of replay, and the idea is accepted with probability $\Pr[J_T(x(r,s)) = 1]$. Other accepted paths may contribute additional probability, so

$$P_{\mathrm{unguide},r}(\mathcal{S}_T) \ge \Pr[J_T(x(r,s)) = 1] 2^{-C(r,s)} = 2^{-K(r,s)}, \tag{17}$$

where $K(r,s) = C(r,s) - \log_2 \Pr[J_T(x(r,s)) = 1]$. The judge cost accounts for acceptance uncertainty and is separate from the bits needed to decode the answers. For runs without a final idea or with zero pass rate, define $K(r,s) = +\infty$ and the inequality still holds.

The left-hand side depends on r but not on s. Average the inequality over the guide tape and then over the generator tape to obtain

$$P_{\text{unguide}}(\mathcal{S}_T) = \mathbb{E}_R[P_{\text{unguide},R}(\mathcal{S}_T)] \geq \mathbb{E}_{R,S}[2^{-K(R,S)}] = \mathbb{E}_{\tau\sim P_{\text{guide}}}[2^{-K(\tau)}]. \tag{18}$$

We assume that executions terminate after finitely many actions with probability one, which lets us disregard nonterminating realizations of R, S. The final equality uses the guided law induced by these tapes and the fact that K averages the fresh judge's verdict into its pass rate. This proves the first inequality in Equation 1. If the expected cost is finite, convexity of $k \mapsto 2^{-k}$ gives

$$\mathbb{E}_{\tau\sim P_{\text{guide}}}[2^{-K(\tau)}] \geq 2^{-\mathbb{E}_{\tau\sim P_{\text{guide}}}[K(\tau)]},$$

which proves the second inequality in Equation 1. □

A.4 ELBO INTERPRETATION

Consider forward execution without recovery. Write $q_t(a \mid \tau_t, T)$ for the guide's probability of selecting action a given trajectory prefix τ_t and target T. Define the guided log-likelihood cost by

$$L_{\text{guide}}(\tau) = \sum_{t=1}^{|\tau|} \log_2 \frac{q_t(a_t \mid \tau_t, T)}{p_t(a_t \mid \tau_t)}. \tag{19}$$

When guided execution succeeds with probability one, $P_{\text{guide}}(\mathcal{S}_T) = 1$, the terminal cost is zero almost surely, so $K(\tau)$ equals the answer cost $C(\tau) = \sum_{t=1}^{|\tau|} c_t$. Let $P_{\text{unguide},T} = P_{\text{unguide}}(\cdot \mid \mathcal{S}_T)$ be the unguided trajectory distribution conditioned on acceptance, and write $D_{\text{KL},2}$ for Kullback–Leibler divergence measured in bits. For $D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide}}) < \infty$, the log acceptance probability satisfies

$$\begin{aligned} \log_2 P_{\text{unguide}}(\mathcal{S}_T) &= D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide},T}) - D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide}}) \\ &= D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide},T}) - \mathbb{E}_{\tau\sim P_{\text{guide}}}[L_{\text{guide}}(\tau)]. \end{aligned} \tag{20}$$

The negative expected log-likelihood cost is therefore an evidence lower bound (ELBO) on the log acceptance probability. The nonnegative divergence $D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide},T})$ is the gap, which vanishes exactly when $P_{\text{guide}} = P_{\text{unguide},T}$. A guide restricted to selecting offered answers need not be able to realize this conditional distribution.

ELBO identity. The guided and unguided trajectory laws share the initialization, generator transition kernels, and terminal judging rule. Every action with positive guide probability also has positive generator probability, so $P_{\text{guide}} \ll P_{\text{unguide}}$. For each finite trajectory, the common factors cancel, giving the log-likelihood ratio

$$\log_2 \frac{\mathrm{d}P_{\text{guide}}}{\mathrm{d}P_{\text{unguide}}}(\tau) = \sum_t \log_2 \frac{q_t(a_t \mid \tau_t, T)}{p_t(a_t \mid \tau_t)} = L_{\text{guide}}(\tau). \tag{21}$$

When $P_{\text{guide}}(\mathcal{S}_T) = 1$, the conditional law $P_{\text{unguide},T} = P_{\text{unguide}}(\cdot \mid \mathcal{S}_T)$ exists because $P_{\text{guide}} \ll P_{\text{unguide}}$ implies $P_{\text{unguide}}(\mathcal{S}_T) > 0$. Fresh terminal judging also implies $\Pr[J_T(x(\tau)) = 1] = 1$ and hence $K(\tau) = C(\tau)$ almost surely under P_{guide}. On P_{guide}-almost every trajectory, $\mathrm{d}P_{\text{guide}}/\mathrm{d}P_{\text{unguide},T} = P_{\text{unguide}}(\mathcal{S}_T)\,\mathrm{d}P_{\text{guide}}/\mathrm{d}P_{\text{unguide}}$. Taking logarithms and expectations yields

$$D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide},T}) = \mathbb{E}_{\tau\sim P_{\text{guide}}}[L_{\text{guide}}(\tau)] + \log_2 P_{\text{unguide}}(\mathcal{S}_T). \tag{22}$$

For finite $D_{\text{KL},2}(P_{\text{guide}} \| P_{\text{unguide}}) = \mathbb{E}_{\tau\sim P_{\text{guide}}}[L_{\text{guide}}(\tau)]$, rearranging gives Equation 20. If the divergence is infinite, the negative expected log-likelihood cost equals $-\infty$ and remains an ELBO. □

A.5 PERPLEXITY AS A SPECIAL CASE

Ordinary next-token prediction is a special case of the Priced Guidance protocol. The generator offers vocabulary tokens as its answer choices and assigns them the language model's next-token probabilities. The guide selects the next token of the original string at every step. The resulting compression cost is the string's negative log-likelihood, and its average cost per token determines perplexity.

Theorem A.2 (Next-token prediction and perplexity). *Fix a finite vocabulary $\mathcal{V}$, a length $n \geq 1$, and an initial context u, all chosen independently of the target. Let p_θ be an autoregressive language model with a strictly positive next-token distribution on $\mathcal{V}$ at every prefix. Represent the original string by its token sequence $y = (y_1, \ldots, y_n) \in \mathcal{V}^n$ under a fixed tokenization, and take $T = y$. At step t, after the selected prefix $z_{<t}$, the generator asks which token comes next and commits to*

$$\mathcal{A}_t = \mathcal{V}, \qquad p_t(a \mid \tau_t) = p_\theta(a \mid u, z_{<t}), \qquad a \in \mathcal{V}. \tag{23}$$

Its transition appends the selected token deterministically, and it returns the resulting sequence after exactly n steps. The guide always selects y_t, without recovery, and the deterministic judge accepts exactly when the returned token sequence equals y. Then the guided compression cost is

$$K(y \mid u) = -\sum_{t=1}^{n} \log_2 p_\theta(y_t \mid u, y_{<t}) = -\log_2 p_\theta(y \mid u). \tag{24}$$

Define the model's token perplexity on this string by $\mathrm{PPL}_\theta(y \mid u) = \big(\prod_{t=1}^{n} p_\theta(y_t \mid u, y_{<t})\big)^{-1/n}$. *It satisfies*

$$\mathrm{PPL}_\theta(y \mid u) = 2^{K(y|u)/n}. \tag{25}$$

The bound in the Priced Guidance Theorem is exact in this setting,

$$P_{\mathrm{unguide}}(\mathcal{S}_T) = p_\theta(y \mid u) = \mathbb{E}_{\tau \sim P_{\mathrm{guide}}}[2^{-K(\tau)}] = \mathrm{PPL}_\theta(y \mid u)^{-n}. \tag{26}$$

Proof. The guided prefix equals $y_{<t}$ at every step, so selecting y_t costs $-\log_2 p_\theta(y_t \mid u, y_{<t})$ bits. The guided output is always y, which the judge accepts with probability one, giving zero terminal judge cost. Summing the answer costs and applying the autoregressive chain rule proves Equation 24. Taking the nth root of $2^{K(y|u)}$ gives Equation 25. Unguided execution samples each token from the same conditional distribution. Its unique accepted token sequence is y, whose probability is $\prod_{t=1}^{n} p_\theta(y_t \mid u, y_{<t}) = 2^{-K(y|u)}$. The guided run is deterministic, so its score equals its expected score. This proves Equation 26. □

The fixed length specifies the event measured by the theorem. For a model that stops at an end-of-sequence token, the probability of generating the complete string and then stopping also includes that token's probability; selecting it incurs the corresponding additional compression cost. Under the same fixed-length setup, if a deterministic judge instead accepts any idea-equivalent output, including y, then $p_\theta(y \mid u)$ remains a lower bound on acceptance probability. Other accepted outputs may contribute additional probability. Thus perplexity recovers the cost of specifying one original token sequence, while the general protocol can guide generation through higher-level choices.

A.6 Online indexed recovery

We first construct a probability space that exposes repeated stochastic continuations. The recovery result assumes a countable public checkpoint space, matching the finite digital trajectories used in the protocol. The kernel g_T includes a fresh terminal judge verdict whenever a final idea is produced. Its generator marginal is target-blind; its judging rule is the same in guided and unguided runs. This standard Borel kernel admits a measurable map F satisfying

$$F(\tau_t, a, U) \sim g_T(\cdot \mid \tau_t, a), \qquad U \sim \mathrm{Unif}[0, 1]. \tag{27}$$

For every exact checkpoint τ_t, action a, and positive integer j, draw an independent uniform variable $U_{\tau_t,a,j}$ and define

$$X_{\tau_t,a,j} = F(\tau_t, a, U_{\tau_t,a,j}). \tag{28}$$

Independently draw the initial checkpoint $\tau_1 \sim \mu$ from the common target-blind initialization law. Write $X = (\tau_1, (X_{\tau_t,a,j})_{\tau_t,a,j})$ for the complete collection of generator and judge randomness. Each terminal cell retains its sampled verdict when reused. A fresh judge-only evaluation is a new indexed request from a pre-evaluation checkpoint with a fixed candidate and an offered evaluation action. The public trajectory prefix includes its checkpoint identity, so a fresh forward execution visits each indexed checkpoint at most once.

Let Γ_T be a measurable target-aware recovery policy. It may restore any checkpoint it has reached and may request another continuation under any offered action. Assume it makes finitely many requests almost surely and returns the finite indexed active record, starting at τ_1,

$$\widehat{\tau}^\star(X) = (\tau_1, a_1, j_1, \tau_2, \dots, a_L, j_L, \tau_{L+1}), \qquad \tau_{t+1} = X_{\tau_t, a_t, j_t}. \tag{29}$$

Any private randomness used by Γ_T can be conditioned on throughout the argument and averaged at the end. Its public projection $\tau^\star(X) = \mathrm{pub}(\widehat{\tau}^\star(X))$ drops all occurrence indices. For any public trajectory τ, define its candidate's pass rate by

$$s_T(\tau) = \Pr[J_T(x(\tau)) = 1], \tag{30}$$

with $s_T = 0$ when no final candidate is returned.

Fix the target-independent normalized distribution π from Equation 4. Start an extended fresh simulator at $\tau_{\mathrm{sim},1} = \tau_1$ and, at each step, sample

$$A_t \sim p_t(\cdot \mid \tau_{\mathrm{sim},t}), \qquad J_t \sim \pi, \qquad \tau_{\mathrm{sim},t+1} = X_{\tau_{\mathrm{sim},t}, A_t, J_t}. \tag{31}$$

The action and index draws are conditionally independent. Let $\mathcal{F}_t$ contain the simulator's public trajectory prefix and all indexed cells it has used before step t. The full prefix $\tau_{\mathrm{sim},t}$ carries a unique checkpoint identity. Conditional on $\mathcal{F}_t$ and the reached prefix $\tau_{\mathrm{sim},t} = \tau_t$, the unused cells $X_{\tau_t,a,j}$ retain their laws $g_T(\cdot \mid \tau_t, a)$. For every measurable set $\mathcal{D}$ of next prefixes,

$$\Pr(\tau_{\mathrm{sim},t+1} \in \mathcal{D} \mid \mathcal{F}_t, \tau_{\mathrm{sim},t} = \tau_t, A_t = a) = \sum_{j=1}^{\infty} \pi(j) \Pr(X_{\tau_t,a,j} \in \mathcal{D}) \tag{32}$$

$$= \sum_{j=1}^{\infty} \pi(j) g_T(\mathcal{D} \mid \tau_t, a) = g_T(\mathcal{D} \mid \tau_t, a). \tag{33}$$

Induction over the stopped trajectory shows that the public marginal of this simulator is exactly the fresh unguided law P_{unguide}.

Condition on the complete generator and judge randomness X, including τ_1. The recovery policy's returned path and its retained indices are then fixed. Let M_X be the event that the fresh simulator selects $(A_t, J_t) = (a_t, j_t)$ at every prefix on $\widehat{\tau}^\star(X)$. Let C_π sum the answer and occurrence costs on this record. Conditional independence and the chain rule give

$$\Pr(M_X \mid X) = \prod_{t=1}^{L} p_t(a_t \mid \tau_t)\pi(j_t) \tag{34}$$

$$= 2^{-C_\pi(\widehat{\tau}^\star(X))}. \tag{35}$$

Write τ_{sim} for the simulator's public trajectory. On M_X, its final candidate equals the returned candidate, so $s_T(\tau_{\mathrm{sim}}) = s_T(\tau^\star(X))$. Since s_T is nonnegative,

$$\mathbb{E}[s_T(\tau_{\mathrm{sim}}) \mid X] \geq s_T(\tau^\star(X)) \Pr(M_X \mid X). \tag{36}$$

The simulator has marginal law P_{unguide}, under which the terminal verdict is fresh given its candidate. Consequently, averaging over X gives

$$P_{\mathrm{unguide}}(\mathcal{S}_T) = \mathbb{E}[s_T(\tau_{\mathrm{sim}})] \tag{37}$$

$$\geq \mathbb{E}_X\left[s_T(\tau^\star(X)) 2^{-C_\pi(\widehat{\tau}^\star(X))}\right] \tag{38}$$

$$= \mathbb{E}_X\left[2^{-K_\pi(\widehat{\tau}^\star(X))}\right], \tag{39}$$

where $K_\pi = C_\pi - \log_2 s_T(\tau^\star(X))$. This proves Theorem A.1. If the policy returns no final candidate, its right-hand contribution is zero by the convention $K_\pi = +\infty$. The argument allows the recovery policy to select its returned record using previously observed verdicts. It does not replace that selected verdict by its conditional mean.

The distribution in Equation 10, with $\varepsilon = 0.05$, satisfies

$$\sum_{j=1}^{\infty} \pi(j) = 1 - \varepsilon + \varepsilon \sum_{j=2}^{\infty} \left(\frac{1}{j-1} - \frac{1}{j}\right) = 1. \tag{40}$$

Every occurrence has strictly positive mass. The corresponding code length is $-\log_2(1-\varepsilon)$ for $j=1$ and $\log_2 j + \log_2(j-1) - \log_2 \varepsilon$ for $j \geq 2$. If the mean total cost over all guided recovery runs is finite, Jensen's inequality applied to Equation 39 gives

$$P_{\text{unguide}}(\mathcal{S}_T) \geq \mathbb{E}_X[2^{-K_\pi}] \geq 2^{-\mathbb{E}_X[K_\pi]}. \tag{41}$$

This is Equation 9.

Repeated terminal judgments. The same argument supports the empirical terminal cost used in Section 3. Include a fixed number k of fresh, conditionally independent terminal verdicts in each terminal indexed draw, and let $R(\tau) = l(\tau)/k$ be their passing fraction. Reusing a draw retains its entire batch of verdicts. On the matching event the simulator reproduces this fraction, and $R \geq 0$, so Equation 36 holds with R in place of s_T. In the fresh simulator's marginal law, $\mathbb{E}[R \mid x] = \Pr[J_T(x) = 1]$. Thus, writing $\widehat{K}_\pi = C_\pi - \log_2 R$ with $\widehat{K}_\pi = +\infty$ at $R = 0$,

$$P_{\text{unguide}}(\mathcal{S}_T) = \mathbb{E}[R(\tau_{\text{sim}})] \geq \mathbb{E}_X\left[2^{-\widehat{K}_\pi(\widehat{\tau}^\star(X))}\right]. \tag{42}$$

Here $P_{\text{unguide}}(\mathcal{S}_T)$ remains the probability of acceptance by one fresh judge evaluation. After adaptive recovery, the selected fraction need not be conditionally unbiased for the returned candidate's pass rate; the proof uses the simulator's marginal law. A failed or missing run contributes zero under the infinite-cost convention.

B Additional Experimental Details

B.1 Memorization audit

We want to make sure the generator hasn't seen the papers it needs to compress. So we select target papers from Top-1 Hugging Face Daily Papers since March 2, 2026 and also perform an additional title-based recall screening to remove potentially memorized papers. For the development set, we collect 40 Top-1 papers from March 2 through April 30, 2026. For the test set, we collect Top-1 papers from May 1 through September 6, 2026, yielding 87 unique candidates. The screening below removes any paper that one of our generators recalls from its title alone.

Title-based recall audit. To assess whether a model has already memorized a candidate paper, we provide only its title and ask for its research problem, method, technical approach, and findings. For titles containing a colon, we retain only the text before the first colon to reduce information supplied by descriptive subtitles. We compare the recalled summary with the private gold summary. A fixed GPT-5.6 Terra Judge assigns a *memorization score* from 0 to 100 based on semantic agreement, allowing differences in wording and level of detail. We first apply this audit to 849 Top-1 selections from Hugging Face Daily Papers dated between May 2023 and August 2026, using Claude Opus 5 for recall. Figure 7 shows high recall through November 2025, followed by a sharp decline. The monthly mean falls from 87.9 in November to 52.6 in December. We therefore adopt 80 as the operational threshold for identifying conspicuous recall.

Shared exclusion rule. We exclude papers for which any of our generators receives a memorization score of at least 80. This leaves 40 papers in the development set and 87 in the test set, with no overlap between the sets. All five audited models score below 80 on every retained paper (Table 1).

Table 1: Title-only recall on the 40-paper development set and 87-paper test set, scored from 0 to 100 by GPT-5.6 Terra. All five models are evaluated on the same papers within each set. Means are computed after cohort selection. Score ≥ 80 counts papers meeting the memorization threshold.

Metric	Fable 5	Opus 5	GPT-5.6 Sol	GLM-5.3	Astra
Development set (40 papers)					
Mean score	22.30	24.78	28.05	9.93	2.50
Score ≥ 80	0	0	0	0	0
Test set (87 papers)					
Mean score	23.37	22.70	27.33	13.40	1.51
Score ≥ 80	0	0	0	0	0

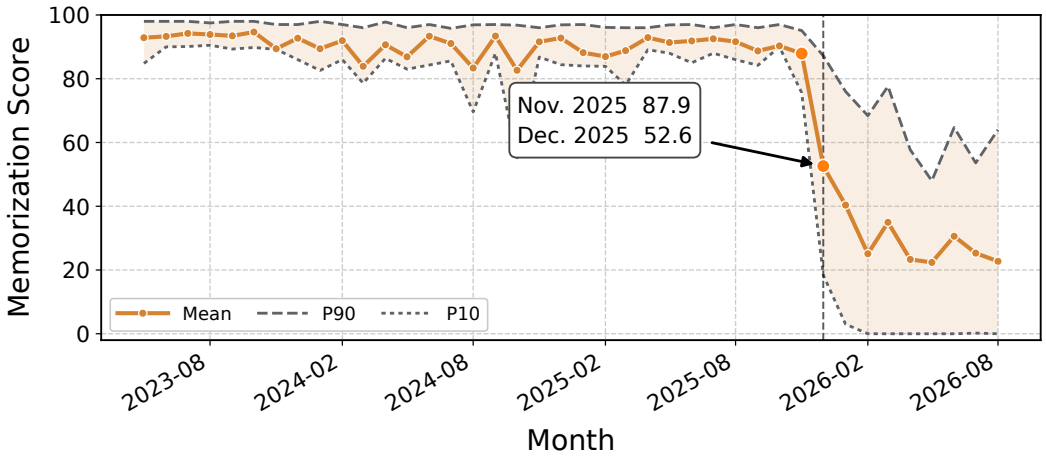

Figure 7: Monthly title-only memorization scores for Claude Opus 5 on 849 Top-1 selections from Hugging Face Daily Papers, judged by GPT-5.6 Terra. We can observe a clear transition near the knowledge cutoff. The solid orange curve is the monthly mean; the dashed and dotted curves show the 90th and 10th percentiles, with their interval shaded. Each month contains 15–25 papers.

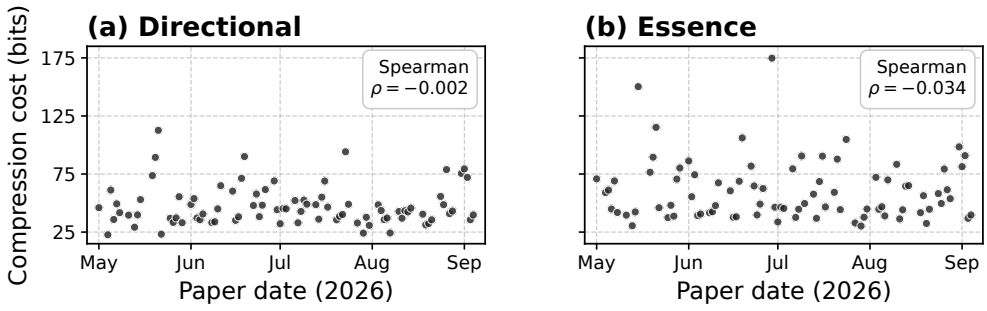

Figure 8: **Ensemble compression cost versus paper date.** Black points show Directional (left) and Essence (right) costs for the uniform mixture of Fable 5.1, Opus 5, and Astra on the 87-paper test set (Appendix A.2). Dates are Hugging Face Daily Papers listing dates from May 1 to September 4, 2026, spanning approximately four months. Both correlations are close to zero over this period.

B.2 Compression cost over time

Figure 8 plots the ensemble's compression cost on each test paper against the paper's listing date.

B.3 Additional test-set results

Figure 9 shows the test-set results under the Directional judge. Table 2 breaks down the compression cost of Opus 5 high by scaffold component.

Implications for automatic idea generation. The lowest individual median compression costs, from Opus 5 high under the Directional judge and Fable 5.1 under the Essence judge, correspond to implied probability scores 2^{-K} of approximately 5.3×10^{-15} and 3.8×10^{-21}, respectively. As with text, these small probabilities concern one particular target and do not imply that the models cannot generate high-quality ideas autonomously (Section 6).

Technical reports tend to have high compression cost. The Kimi K3 paper (Kimi Team et al., 2026) is the only test target for which none of the five evaluated generators passes the Directional judge in our test runs. We attribute this to the fact that Kimi K3 is a combination of many novel techniques and would require a much higher compression cost to fully cover. We also found that other technical reports such as those by Li et al. (2026b); Wang et al. (2026a) have two of the three

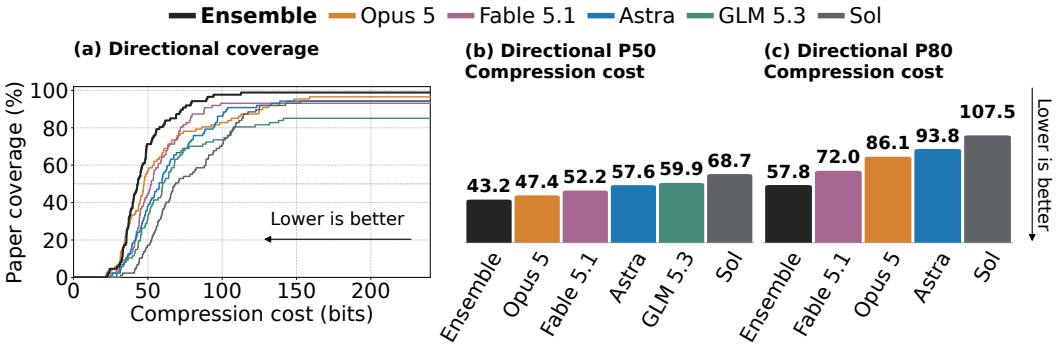

Figure 9: Test-set results under the Directional judge on the 87-paper test set, in the layout of Figure 2. (a) Coverage versus compression cost. (b) Median (P50) and (c) 80th-percentile (P80) compression cost. As in Figure 2, GLM 5.3 is omitted from the P80 panel; its Directional P80 is 108.3 bits.

Table 2: **Compression cost and scaffold usage for Opus 5.** Opus 5 high on the same 74 test papers for which the generated idea passes the Essence judge. We show the compression costs and also the number of times different components are called in the final successful trajectory. Δ is the Essence cost minus the Directional cost. Keyword costs include semantic-category and strategy selections; the letter-hint row includes the full hint path.

	Compression cost (bits)			Directional usage		Essence usage	
Component	Directional	Essence	Δ	Calls	Mean	Calls	Mean
Field taxonomy	11.61	11.61	0.00	270	3.65	–	–
Contribution category	3.46	3.46	0.00	74	1.00	–	–
Multiple choice	3.21	4.57	1.36	57	0.77	21	0.28
Keyword	33.29	43.70	10.41	134	1.81	56	0.76
Without letter hints	16.06	23.95	7.90	103	1.39	49	0.66
With letter hints	17.23	19.75	2.51	31	0.42	7	0.09
Correction	0.00	0.67	0.67	–	–	5	0.07
Submission	3.09	3.33	0.24	42	0.57	32	0.43
Retry	2.14	2.54	0.40	–	–	–	–
Total	**56.80**	**69.87**	**13.07**	**577**	**7.80**	**114**	**1.54**

highest Essence compression costs among the test papers that the ensemble recovers. This is not simply because these papers or their summaries are longer: across the 87 test papers, compression cost shows no significant correlation with the length of either the gold summary or the full paper (Spearman correlation ≤ 0.10 for the ensemble compression cost under both judges).

B.4 Verbatim GLM 5.3 baseline

We score each gold summary token by token with GLM 5.3 under a fixed prompt asking for a deep-learning research idea, with no target-specific context. Multiplying the base-2 logarithm of perplexity by the number of summary tokens gives the negative log-likelihood in bits (Appendix A.5). We score only the summary tokens, excluding the prompt and termination. The reported median of 1,147.1 bits uses low reasoning effort and an empty reasoning trace. Changing only the effort instruction to high or max gives similar medians of 1,153.5 and 1,194.5 bits, respectively.

B.5 Lossless gzip baseline

We compress the same gold idea summaries used for the GLM 5.3 verbatim baseline in Section 4. Each summary contains the contribution category and the grouped descriptions of the setting and main object; paper titles, other gold-summary fields, and JSON syntax are excluded. We compress each of the 87 UTF-8 strings independently with gzip at level 9, with the timestamp set to zero

and no stored filename, preset dictionary, or context from other papers. We count the complete compressed file, including its 18-byte header and trailer, and multiply its length in bytes by eight.

The median compressed length is 5,712 bits (714 bytes), compared with 10,008 bits for the uncompressed text. Excluding the header and trailer gives a median DEFLATE payload of 5,568 bits. Compression level 6 gives the same median as level 9, while level 1 gives 5,832 bits. All compressed summaries decompress exactly to the original bytes. Independent runs with Python's gzip implementation and the system gzip command give identical compressed lengths for all 87 summaries.

This baseline changes both the compressor and the reconstruction criterion relative to Priced Guidance. It therefore does not isolate the effect of recovering an idea's essence rather than its wording. The GLM 5.3 comparison in Section 4 holds model identity fixed.

B.6 Directional ablation results

Figure 10 shows the results of Figure 6 under the other judge: the randomness and ensembling panels under the Directional judge, and the candidate-count panel under the Essence judge. Figure 11 shows the Fable 5.1 reasoning-effort comparison from Section 5.

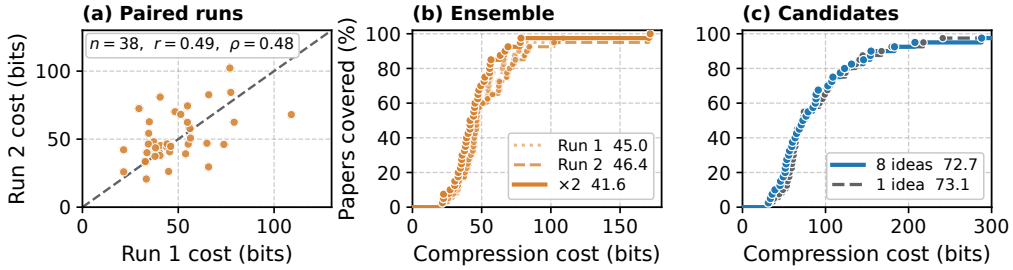

Figure 10: Run-to-run variation and ensembling under the Directional judge, and candidate count under the Essence judge, on the development set. **(a)** Paired Directional costs of two independent Opus 5 runs ($n = 38$). The dashed diagonal denotes equal cost; r and ρ are Pearson and Spearman correlations. **(b)** Coverage for both independent runs and their uniformly weighted ensemble, with effective cost given by Equation 12. **(c)** Native Codex Astra with one (gray dashed) or eight (blue solid) Directional candidates, evaluated by the Essence judge; legend values report median compression cost in bits among each group's successful runs, and costs include inherited Directional cost. Details are deferred to Appendix C.14.

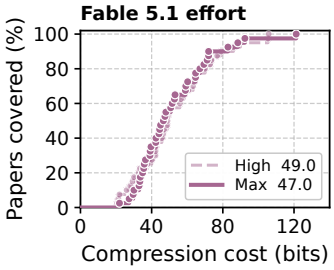

Figure 11: Fable 5.1 reasoning effort ablation under the Directional judge on the development set. Legends report median compression cost in bits among successful runs. This comparison is indicative because high and max effort use Fable 5.1 and Fable 5 guides, respectively.

B.7 Context-management ablations

Custom compaction lowers Astra's Essence cost, and retained reasoning history is associated with lower Directional cost. Our custom compaction in the final test scaffold lowers Astra's median compression cost under the Essence judge compared with native Codex context management. We compare two eight-candidate configurations using Astra at extra high effort through Codex, with the same set of prompts for generators. The native configuration uses native Codex compaction; our

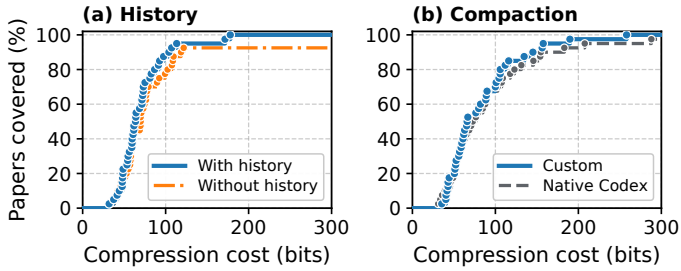

Figure 12: Astra context-management comparisons on the development set. **(a)** Directional compression cost with one candidate in both configurations: Codex with reasoning history (blue) and the API scaffold without reasoning history (orange). **(b)** Essence compression cost with eight Directional candidates using custom context management (blue) or native Codex (gray). Essence costs include inherited Directional cost. Details are deferred to Appendix C.14.

custom configuration explicitly compresses earlier history into a natural-language self-summary and a brief instruction to follow the current task. Directional median compression cost remains nearly unchanged, at 53.1 bits with native Codex and 53.5 bits with our custom configuration. Under the Essence judge, the median among successful runs decreases from 72.7 to 66.4 bits, while coverage increases from 39/40 to 40/40 (Figure 12(b)). This comparison shows that our customized natural-language compaction performs at least as well as native Codex compaction.

The one-candidate Codex configuration has lower median Directional compression cost than our earlier Astra scaffold that retains explicit working state but discards messages and reasoning between calls. Figure 12(a) compares their coverage over all 40 papers. This earlier configuration uses Astra at max effort through the API with a Fable 5 guide. Both configurations generate one candidate per submission. On the 37 papers for which both configurations produce an idea that passes the Directional judge, median compression cost under the Directional judge decreases from 69.7 to 63.7 bits with reasoning history. This observation is consistent with OpenAI's report that reusing reasoning across tool calls improves model performance (OpenAI, 2025). This comparison holds candidate count fixed but also changes the reasoning effort (extra high in Codex versus max through the API). The Codex configuration additionally includes some small instructions for handling retained history and structured outputs. The details are deferred to Appendix C.14.

C Prompts and Agent Protocol

This appendix presents excerpts of the Directional and Essence prompts used in our experiments. Appendix C.14 identifies the scaffold variants used in Section 5 and their differences from the eight-candidate prompts below. The implementation constructs prompts compositionally rather than using one monolithic instruction. We therefore list each shared block once and state how the blocks are assembled. Line wrapping and typographic dashes have been normalized, and run-dependent values are replaced by angle-bracketed placeholders. The excerpts preserve the original wording, including metric and role names. Omissions within quoted blocks are marked with `[...]`. The retained text is reproduced from `participant/pair.py`, the Directional and Essence stage modules, their channel modules, and the Arena-owned `evaluation/research.py` Judge.

The experiments in Section 3 use the Directional and Essence stages. The excerpts retain the original pricing guidance. The reported compression costs are recomputed from the recorded trajectories using the occurrence prior in Appendix A.1 and the multiple-candidate judge cost in Section 2.3; informal cost estimates inside the prompts do not define that accounting.

For a multiple-choice (MC) or keyword call at recognition stage s and action a, let P_{gen} be the shared researcher preamble, H_a the action header, G_s the stage goal, and $A_{s,a}$ the stage-specific action instruction. Writing $\|$ for concatenation, the developer message is

$$P_{\text{gen}} \,\|\, H_a \,\|\, G_s \,\|\, A_{s,a}.$$

State updates use the preamble and stage goal, with their action instruction in the user message. Correction calls append their action instruction directly to the preamble. Directional submission ex-

pansion uses its own instruction followed by the Directional stage goal, without the shared preamble (Section C.7). The user message is a rendering of only generator-visible state. The generator never receives the target paper, guide reasoning, or detailed Judge feedback. Promotion reveals that the preceding stage passed. All model calls use constrained JSON schemas; the relevant cardinalities and fields are stated below.

C.1 Generator scaffold details

This subsection details each step of the generator loop in Section 3.

Opening bids. We fix a research taxonomy of deep learning with 3,035 nodes and a list of 11 contribution categories, both generated by GPT-5.6 Sol and given to all generator models in the prompts. Appendix H lists both. The taxonomy bids walk down the taxonomy one edge at a time, with a probability for each offered node, and the guide selects the node containing the target idea. The category bid offers the 11 contribution categories with equal probability.

Context management. In our final scaffold, successive generator calls share the earlier conversation and reasoning history. The generator also maintains facts confirmed through guide answers and a current candidate idea. A whiteboard records established information, working hypotheses, and unresolved questions. The whiteboard is also shared with the guide. When the context grows too long, earlier history is compacted into a continuation summary of past histories. The summary distinguishes hypotheses from confirmed guide answers. Subsequent calls use this summary alongside the current facts, whiteboard, and candidate idea to construct further questions.

Meta question about strategy. Each turn begins with a *meta question about strategy*. The generator prepares each strategy's first question, answer options, and probabilities before the guide selects a strategy, and then asks the selected question unchanged. Follow-up questions are conditioned on earlier answers. Figure 4(a) shows an example meta question and the guide's answers.

Strategies. Each strategy is a fixed sequence of bids of a certain style. The generator writes every question and answer and assigns every probability; the guide only selects answers. We outline the questions and choices and discuss how the generator specifies the probability distribution for each question in Appendices C.8–C.10.

1. *Multiple choice.* The generator asks about the unresolved aspect of its candidate idea whose answer would most change it. It offers eight concrete answers, each adding one fact of the form "aspect: value", and two rejection answers: all eight answers are wrong, or the generator should ask about a different aspect. Figure 4(a) shows an example.
2. *Keyword.* The generator first asks which of 16 semantic categories contains a still-missing concept, with a rejection answer. It then offers 64 candidate phrases in the selected category, a rejection answer, and a *letter hint* answer that guesses the first two letters of the missing phrase. After receiving the letter hints, the generator offers 16 completions of the revealed prefix and an answer that extends the prefix by one character.
3. *Correction*, used only in the Essence stage. The generator first asks which of 16 claims already expressed or entailed by its candidate idea most needs correction, with a rejection answer for when that claim is not among them. For the selected claim, it then offers 16 patches that replace, refine, or delete the claim, with a rejection answer for when none of the patches is faithful. Selecting either rejection answer records the rejected choices as negative evidence and returns to state update and strategy selection. Correction remains available even if the candidate idea is unchanged. A replacement or refinement adds the new claim as a confirmed fact, and a deletion removes the claim.

The remaining subsections of this appendix give the full prompts.

Submission. In the Directional stage, the generator prepares eight distinct, complete candidate ideas with probabilities before each meta question. The guide sees these candidates but only decides whether to submit them; it does not choose among them. Submission sends all eight candidates unchanged to the judge, and the submission cost is the multiple-candidate judge cost in Section 2.3, with each candidate's pass rate estimated from its judge evaluations. If no candidate passes, the submission is rejected. After promotion, the Essence judge first evaluates the eight-candidate submission that passed the Directional judge. In the Essence stage, a newly authored submission contains

only the current candidate idea, with probability one. Restoring an earlier Directional meta question retains its prepared eight-candidate slate, which is evaluated by the active Essence judge.

Promotion to Essence. It is intuitively useful to inform the generator when it finds the right research direction and only needs to find the core method. We therefore build a two-stage generator scaffold. A run that passes the Directional judge can advance to the Essence stage while preserving its candidate idea, list of facts, and earlier questions and selected answers. The new prompt to the generator defines a new goal requiring the defining mechanism and adds the correction strategy alongside the multiple-choice and keyword strategies. Promotion informs the generator that the candidate passed the Directional Judge, while detailed Judge explanations remain private to the guide.

C.2 Guide scaffold details

This subsection details the guide scaffold summarized in Section 3.

We build a lightweight scaffold to instruct the Fable model to provide answers to the generator's bids. The guide also maintains a private memory, similar to the generator's whiteboard. Unlike the whiteboard, which is shared with the guide, this memory is never shown to the generator.

Selecting supported answers. For each bid, the guide receives the question, answer options and probabilities, the target summary, the judge criterion, and its past private memory. At strategy-selection turns it also sees the generator's whiteboard and candidate idea. The guide selects an option supported by the target, balancing compression cost and the uncertainty it resolves. Only the choice is revealed to the generator.

Private previews and submission. The guide may privately obtain a *Judge preview* of the current submission snapshot and use it to decide whether to continue or submit. Submission triggers a formal evaluation with fresh Judge verdicts. Previews and any other guide computation add no compression cost, because only the answers the guide selects are priced. The guide could be arbitrarily strong in our framework.

Recovery. The guide can *check out* an eligible earlier question Q, either to explore a different path or, as required, after a rejected submission. Checkout restores the generator to its state right after it made the bid at Q and presents that bid again unchanged. The generator forgets about everything after Q, while the guide keeps its private memory of the abandoned branch. The guide then selects an action at Q, possibly the one it chose before, and the generator continues from there. Each step on the returned trajectory pays the extra cost $-\log_2 \pi(j_t)$ from Section 2.3, where the kept rollout is the j_t-th rollout of that action at that question, counting those on abandoned branches. With the distribution used in our experiments (Appendix A.1), a first rollout costs 0.074 bits and a second costs 5.32 bits.

C.3 Generator-visible state

The user message supplied to generator calls has the following form. Missing active facts, whiteboard, and top idea have explicit placeholders. Retired facts, prior drafts, and rejected regions appear only when nonempty; the complementary-field line is omitted when its route is disabled and no complementary field was purchased. Bounded histories contain at most 64 active facts, 16 retired facts, 16 prior drafts, and bounded lists of rejected regions.

Prompt

```
The hidden paper's field label: <PRIMARY TAXONOMY PATH>
Complementary field label: <SECONDARY PATH, NONE, OR NOT YET PURCHASED>
Contribution category: <CATEGORY OR NOT YET SELECTED>
Recognition-ladder stage: <directional | essence>

Facts confirmed so far (keywords and oracle answers):
- [<FACT ID>] <CONFIRMED FACT>
...

Retired facts (superseded by paid corrections; do not reuse):
- [<FACT ID>] <RETIRED FACT>
...

Your current whiteboard:
```

```
<MODEL-AUTHORED BELIEF STATE>

Your current top idea (the starting hypothesis for submission expansion):
<CURRENT DRAFT>

Supported but insufficient prior drafts (stable IDs; cite for synthesis):
- <DRAFT ID> (after <RECOVERY ROUTE>): <DRAFT>
...

Priced negative evidence (do not revisit these regions):
- <REJECTED WHOLE-DRAFT, MC, KEYWORD, OR CORRECTION REGION>
...
```

The initial taxonomy walk and contribution-category choice are deterministic protocol questions, not model prompts. The primary field starts at `deep learning` and is traversed one public edge at a time. The category question offers the eleven categories used in the implementation with equal probability.

C.4 SHARED GENERATOR PREAMBLE

MC, keyword, correction, and state-update calls begin with the following developer-message preamble. Directional submission expansion has the separate prompt in Section C.7.

Prompt

```
You are a researcher trying to recover the idea of one hidden research paper
published after your knowledge cutoff, so you must reason from first principles,
not pattern-matching to some famous paper. However, as a knowledgeable researcher,
you should be confident that you know the fundamentals of the field, which are the
building blocks of the hidden paper.

You can pose questions to an oracle that has access to the hidden paper. You are
given its coarse field label and a list of confirmed facts about the paper, including
keywords the oracle has confirmed (named ideas or mechanisms) together with its
answers to some of your questions. Your goal is to recover the paper with minimal
help from the oracle; therefore every question you ask should be highly valuable and
should reduce your uncertainty by a lot.

Idea format. Your target is the paper's setting_and_object -- its crisp essence, in
three parts: the category, the core problem, and the main object:
  - Category: what kind of contribution the paper makes. It is one of the following:
    a new algorithm or method; a theory or analysis result; a benchmark; a dataset;
    an architecture; a mechanism study (how something works); a scaling law; an
    empirical phenomenon or observation study; an evaluation protocol or metric; a
    system or software artifact; or an application of existing methods to a domain.
    The category determines what the main object is: algorithm -> the method;
    benchmark or dataset -> the artifact; theory -> the central result; system or
    software -> the artifact and its key design principle(s).
  - Core problem: in one sentence, the essential problem the paper studies.
  - Main object: the single central thing the paper contributes or studies, stated
    precisely enough that an expert recognizes the contribution. For an algorithm
    or method paper this is the algorithm itself -- its defining mechanism, stated
    at the level of the essential idea: the distinctive computation that makes it
    this method (what it essentially does), not where it is applied. State it
    precisely enough to distinguish the method from its neighbors, but no finer --
    the essential idea, not the full formula. Implementation specifics that are not
    part of the core idea may be omitted; they are instantiation or demonstration
    detail, not the essence. For theory, the central result; for a benchmark or
    dataset, the artifact; for a software or system paper, the artifact and its key
    design principle(s).

Reason about the problem and the mechanism -- what bottleneck a paper in this
subarea would attack and what technique removes it -- rather than recalling named
systems. The field label is information: treat it like a confirmed fact, and make
every hypothesis faithful to both the field label and all confirmed facts at once,
reconciled into one coherent object. If the field label conjoins several subfields
(e.g. "X and Y"), do not assume the paper spans all of them; reason about which one
it most plausibly sits in. Think deeply about researchers' original intention
instead of concrete details that can be decided later.

When a confirmed fact names a method you recognize, do not assume the paper simply
is that method. A recognized prior method is almost always the base this new paper
```

```
builds on, not its contribution: treat it as background and keep reasoning about the
new development, variant, or modification the paper adds on top of it, rather than
re-describing the known method's standard recipe. The paper's object is what it adds
to or changes about that method.

You maintain a whiteboard: your full current belief about the paper -- the
established category (or your leading hypothesis for it), everything else
established (treat every oracle answer as reliable ground truth), your single
leading hypothesis for the main object, core problem, and defining mechanism, and
the open uncertainties that matter most, ranked by expected change in belief. For
each confirmed technical concept, note its mathematical form and a one-line
description of its mechanism (what it computes and why); reasoning over the
mechanism of each established piece is what lets you infer the remaining components.
Reason over confirmed facts only, and never invent an unconfirmed value. Once the
category is confirmed, always try to reason mainly along the direction it specified.

You also carry a single top idea: your current best guess, in the format above. It
must integrate all confirmed facts into one coherent object whose pieces work
together as a single mechanism -- not a loose list, and never dropping a confirmed
piece.

Your interaction with the oracle is a multi-turn conversation, and these
instructions hold throughout it. The turns alternate: on an action turn you take
exactly one step; then, once the oracle has answered, on the very next turn you fold
its answer -- reliable ground truth -- into your state, rewriting both your
whiteboard and your top idea before the next action. Each message tells you which
turn it is asking for.
```

C.5 RECOGNITION-STAGE GOALS AND ROUTE MENUS

The active stage contributes exactly one of the following blocks. The displayed route menu is `mc, keyword, submit` at Directional; `mc, keyword, correct, submit` at Essence. A submit route appears only after a nonempty draft exists. The implementation retains legacy `explore`, `differentiate`, and complementary-field route code for compatibility and ablations, but those routes are dropped from every current stage and therefore are not generator prompts used by the reported protocol.

Directional

```
[...]
The Directional Judge passes when the draft is
in the same research direction as the hidden paper: same core problem and goal,
area, and contribution type, with a concrete sensible peer-quality mechanism; it
does not require the exact defining mechanism yet. [...] Find a short, recognizable
statement of the paper's main contribution. Prefer the correct contribution
category, main object, and broad direction; do not invent implementation details or
quantitative results.
```

Essence

```
[...]
Your current draft and fact ledger already
passed the weaker Directional Judge, which required only the right problem, area,
and contribution type with a plausible mechanism. The Essence Judge is stricter:
it passes only when the draft conveys the same setting AND the recognizable essence
of the paper's main object, including its defining mechanism at least coarsely;
secondary datasets, sizes, configurations, and exact parameters may be omitted or
approximated. The gap to close is therefore the defining mechanism, relation, or
organizing principle that distinguishes the contribution -- recover it while
keeping the Directional core faithful.
```

The guide's private `MATCH_INIT` block uses the shorter exact criterion corresponding to the same stage:

directional

```
Pass when the idea is in the same research direction as the gold: same core problem
and goal, area, and contribution type, with a concrete sensible peer-quality
mechanism. It need not recover the gold's exact defining mechanism.

[essence]
Pass when the idea conveys the same setting and the recognizable essence of the
gold's main object, including its defining mechanism coarsely. Secondary datasets,
```

```
sizes, configurations, and exact parameters may be omitted or approximated.
[...]
```

C.6 Route selection and state update

Dispatch has no separate route-selector model prompt. Before presenting a dispatch, the generator authors and caches the exact finite downstream Question for every displayed model-backed route (independent calls may run concurrently against isolated copies of the same visible state). Each dispatch option previews that same serialized Question. After the guide purchases a route, activation replays the cached Question without another model call or mutation from discarded routes. If any route cannot be authored, dispatch fails instead of publishing an empty or partial bundle. The dispatch question is assembled deterministically as follows.

Prompt

```
Choose one priced Directional channel after comparing every exact
Generator-authored downstream slate: <ACTIVE ROUTE PHRASES>.
```

The route list uses the active stage's menu. The dispatch payload binds every cached Question to its source stage, draft, fact-ledger hash, and question hash. Once a Directional draft exists, the same dispatch also publishes and caches the eight submission candidates described below. The bundle includes a hash of that complete submission slate, and `generator_idea_snapshot` exposes the same candidates and probabilities to the guide.

After any applied answer or channel-level rejection, the generator receives the shared preamble and current stage goal, plus the following user-message suffix. It returns `whiteboard` and one `top_idea.setting_and_object`.

Prompt

```
This is a state-update turn. Since your last update, the following priced events
occurred, in order:
- <PRICED EVENT>
...

When a famous method is provided as answer, write down what you know about the
method confidently in the state update so you don't waste bits asking about it.
Treating every priced oracle event above as reliable ground truth, rewrite your
whiteboard and update your single top idea (in the setting_and_object format above)
so both reflect everything now confirmed, reconciled into one coherent object;
never drop a confirmed piece. Keep the exact wording of confirmed fact phrases
inside the top idea where they belong, so later semantic corrections can still
identify the claims they support. A rejection event means every displayed alternative
  was unfaithful: fold it
in as negative evidence, not as new content. If the current top idea is already the
best integration, return it unchanged. JSON only.
```

C.7 Directional eight-candidate submission prompt

Before every Directional dispatch with a nonempty current draft, the generator expands its public state into exactly eight complete candidate ideas. This call uses the following developer-message block followed by the Directional stage goal. Its user message is the generator-visible state above. The constrained response is `ideas[(draft, prob)]`, with exactly eight nonempty, distinct drafts and positive weights. The implementation normalizes the weights to sum to one and caches the exact slate before the guide chooses a route.

Directional submission expansion

```
You are the target-blind Generator. Expand exactly eight
complete, distinct reconstructions of the hidden paper from the current public
state. Each candidate must stand alone: identify the problem/object, central
contribution, and the mechanism or relationships needed to explain that
contribution. Do not return eight short labels, fragments, cosmetic paraphrases,
or references to another candidate. Start from your current best hypothesis but
regenerate each entire idea; spread probability over substantively different
plausible interpretations when the evidence leaves uncertainty. Distinguish
confirmed facts from your own inference. Do not treat earlier unselected eager
previews as confirmed or rejected evidence. Do not invent paper names, authors,
exact numbers, or evaluation results. Give every complete candidate an honest
```

```
positive relative probability weight, in descending order. The scaffold will
normalize these eight weights; there is no ninth retry candidate.

This is an eager submission preview, not an Oracle selection. All eight exact
candidates and probabilities will be shown to the Oracle before it selects a
route. If it selects submit, these cached candidates will be sent unchanged to
the Judge; no new candidate generation follows that selection. Each candidate
is judged separately under the active Directional criterion. The arena charges
-log2(total normalized probability of passing candidates) in addition to the
existing path cost. An unselected slate does not change your active draft or
confirmed facts. Return only the requested JSON object.
```

Selecting `submit` sends all eight cached candidates and their probabilities unchanged to the Judge, with each `draft` stored as `setting_and_object` and an empty `findings` list. Each candidate is judged separately. A passing slate incurs the additional submission cost $-\log_2 \sum_{i:\,\mathrm{pass}(i)} p_i$; a slate with no passing candidate is rejected. An unselected slate changes neither the working draft nor the confirmed facts. The implementation allows at most three generation attempts; failure to produce eight valid candidates aborts dispatch.

A newly authored Essence dispatch uses the single current draft with probability one. If an Essence checkout restores a previously published Directional dispatch, submission preserves that dispatch's exact eight-candidate slate and probabilities and evaluates it with the active Essence Judge, which judges each candidate three times. The recorded runs priced such a slate by the probability mass of candidates that pass any evaluation times the fraction of evaluations with a passing candidate, which can overstate its pass rate. Let l_i count the Essence evaluations that candidate i passes. The reported costs instead use $-\log_2 \sum_i p_i\, l_i/3$.

C.8 Atomic multiple-choice prompt

The MC developer message appends the first paragraph below, the active stage goal, and then the second paragraph below to the shared preamble. The output contains `reasoning`, one `question`, one `axis`, exactly eight `options[(value, prob)]`, `prob_all_incorrect`, and `prob_ask_different`. The public fact is constructed deterministically as `axis: value`; the model cannot attach a second free-form claim.

Prompt

```
You are the target-blind Generator in a finite-choice research-idea recovery game.
Generate one atomic multiple-choice clarification about the single unresolved axis
whose answer would most change the current reconstruction. The Oracle will select
one displayed value through an ordinary priced Choice. You do not see the private
target or Judge feedback. Each concrete option must name one value of the same axis
and provide exactly one short DIRECT fact that follows from selecting it. Runtime
derives that fact deterministically as `axis: value`; there is no second free-form
field. Do not bundle a whole paper idea into an option.

Ask exactly one bare-object question about the highest-value unresolved finite axis
in the Generator-visible state. Produce exactly 8 concrete values at one semantic
level. Values must be mutually exclusive enough that selecting one is an atomic
clarification, not acceptance of a complete draft. Cover distant plausible values
rather than synonyms or cosmetic variants. Do not ask for a paper title, author,
exact metric, dataset name, or trivia already implied by a confirmed
field/category/fact. Runtime derives the public fact deterministically as
`axis: value`; there is no second model-authored fact, label, explanation,
mechanism, consequence, evaluation result, or second claim. Give honest positive
relative weights plus separate positive masses for all concrete values being wrong
and for asking a fundamentally different axis. Use the identical policy and schema
for every contribution category.
```

C.9 Keyword prompts

The keyword route is hierarchical. A 16-way semantic-category slate is purchased first; the selected category becomes a priced public fact. Conditioned on that fact, the generator creates a 64-phrase slate. If the guide purchases a prefix reveal, the generator creates a 16-way completion slate after each revealed prefix. The shared action headers are:

CATEGORY HEADER

```
You are the target-blind Generator in a finite-choice research-idea recovery game.
Before any specific phrase is proposed, the paid keyword route first asks WHERE the
missing exact defining detail lives: you propose semantic categories, one is
selected and paid for as an ordinary priced Choice, and only then do you enumerate
phrases inside it. Each category is a single line with no explanation or hidden
second claim. prob is your honest belief the missing detail belongs to that
category; the weights need not sum to one. You do not see gold or Judge text.

[PHRASE HEADER]
You are the target-blind Generator in a finite-choice research-idea recovery game.
Propose identity-bearing technical names or phrases that could name the hidden
paper's central contribution at the right abstraction. The same policy applies
symmetrically to methods, theory or analysis results, benchmarks, datasets,
architectures, mechanism studies, scaling laws, empirical phenomena, evaluation
protocols, software or systems, and applications. A phrase may name an object,
construct, result, artifact, protocol, phenomenon, or application; do not assume it
is a method or mechanism. Each option contains only the phrase itself, with no
explanation or hidden second claim. prob is your honest belief the phrase is the
central contribution; the weights need not sum to one. prob_hint is your honest
estimate of how much a paid two-letter reveal of the single most important
still-missing phrase would help: high when none of your candidates feels likely to
be the missing construct, low when you are confident one of them is right. The
reveal's price is derived from your own displayed candidate distribution. You do
not see gold or Judge text.

[PREFIX-COMPLETION HEADER]
You are the target-blind Generator completing one identity-bearing technical name
after the Oracle paid to reveal that the single most important still-missing phrase
begins with the given prefix. Generate exactly 16 distinct complete phrases
beginning literally with that prefix. Phrases rejected in earlier guess slates of
this search are priced negative evidence; do not repeat them. Apply the same
contribution-neutral identity semantics used by the keyword channel. Each option is
only a phrase, and prob is its positive relative weight. prob_none is your honest
probability that none of your 16 completions is the intended phrase; it prices the
displayed extend-by-one-character outcome. You do not see gold or Judge feedback.
```

The category and phrase headers are followed by the active stage goal. The following stage-specific suffixes define what the keyword is intended to identify.

Directional keyword suffixes

```
[CATEGORY]
Before proposing phrases, map the space of what could still be missing. Propose
exactly 16 mutually distinct SEMANTIC CATEGORIES of the still-missing central
component of the hidden paper. A category names a KIND of fact -- a kind of data or
data domain, a kind of operation, a kind of architectural component, a kind of
augmentation or invariance treatment, a kind of training-schedule property, a kind
of scope or coverage claim -- not a specific value. Word every category
open-endedly: it must admit values beyond those your confirmed facts already name,
never restating your current draft's enumerations as closed lists. Cover the full
space a reconstruction of a research paper could still lack, including categories
far from your current best guesses: honest calibration spreads mass over regions
you have NOT yet explored, because the confirmed facts imply what is missing is
something you have repeatedly failed to name. Assign each category your honest
probability that the missing central component belongs to it; the weights need not
sum to one. Rank most-likely first. Also report prob_none: your honest probability
that no displayed category contains it. The same policy applies to every
contribution category.

[PHRASE]
Propose exactly 64 distinct candidate identity phrases (each 1-3 words) naming the
missing central component of the hidden paper. The missing component may be a local
refinement of a known idea, a new mechanism, a recombination, an analysis concept,
or an artifact. Infer its likely scope and granularity from the confirmed facts
instead of assuming it must be incremental or surprising. Include both nearby
refinements and more orthogonal alternatives when the evidence leaves both
plausible. Phrase each as a pure technical term, not a vague area name. Assign each
a probability reflecting your honest belief it is the paper's central contribution;
they need not sum to one. Rank most-likely first, and let the set range from the
obvious to the genuinely surprising. Avoid redundancy: for each candidate, ask
whether confirming it would earn new information beyond what the confirmed facts
already imply. Do not merely rename an already-confirmed proposition. You have only
64 slots, so avoid duplicate rephrasings and filler; candidates should represent
meaningfully different remaining hypotheses, and the probability mass should
```

```
reflect the actual posterior implied by the facts. Also report prob_hint: your
honest estimate of how much a two-letter reveal of the single most important
still-missing phrase would help -- high when none of your candidates feels likely to
be the missing construct, low when you are confident one of them is right. The same
count, semantics, and coverage rule applies to all contribution categories.

[PREFIX COMPLETION]
The Oracle paid to reveal that the single most important still-missing identity
phrase begins with the given prefix. Propose exactly 16 distinct complete phrases
(each 1-3 words) that begin literally with that prefix -- the spelled-out name of
the missing central component, not an acronym. Infer its likely scope and
granularity from the confirmed facts. When a confirmed fact states the semantic
category the missing phrase belongs to, that category was paid for and is settled:
every completion must name a member of it, and a phrase outside it only wastes a
row. Include both nearby refinements and more orthogonal alternatives when the
evidence leaves both plausible. Rank most-likely first. Avoid duplicate rephrasings;
the completions should represent meaningfully different remaining hypotheses
consistent with the prefix. Also report prob_none: your honest probability that
none of your 16 completions is the intended phrase (the true phrase begins with the
prefix but is not among your list); it prices the extend-by-one-character outcome,
so calibrate it honestly. Apply the same rule for every contribution category.
```

Essence keyword suffixes

```
[CATEGORY]
Before proposing phrases, map the space of what could still be missing. Propose
exactly 16 mutually distinct SEMANTIC CATEGORIES of the still-missing defining
mechanism or its decisive component. A category names a KIND of fact -- a kind of
data or data domain, a kind of operation, a kind of architectural component, a kind
of augmentation or invariance treatment, a kind of training-schedule property, a
kind of scope or coverage claim -- not a specific value. Word every category
open-endedly: it must admit values beyond those your confirmed facts already name,
never restating your current draft's enumerations as closed lists. Cover the full
space a reconstruction of a research paper could still lack, including categories
far from your current best guesses: honest calibration spreads mass over regions
you have NOT yet explored, because the confirmed facts imply what is missing is
something you have repeatedly failed to name. Assign each category your honest
probability that the missing mechanism belongs to it; the weights need not sum to
one. Rank most-likely first. Also report prob_none: your honest probability that no
displayed category contains it. The same policy applies to every contribution
category.

[PHRASE]
Propose exactly 64 distinct candidate identity phrases (each 1-3 words) naming the
missing DEFINING MECHANISM, relation, or organizing principle of the hidden paper's
already-located main contribution. The contribution's setting and broad direction
are established by the confirmed facts; the phrases must name how it centrally
works: the mechanism, construction, training or interaction scheme, representation,
or organizing relation that distinguishes it. Phrase each as a pure technical term,
not a vague area name. Assign each a probability reflecting your honest belief it
names the defining mechanism; they need not sum to one. Rank most-likely first, and
let the set range from the obvious to the genuinely surprising. Avoid redundancy:
do not merely rename an already-confirmed proposition, and avoid duplicate
rephrasings and filler; candidates should represent meaningfully different remaining
hypotheses. Also report prob_hint: your honest estimate of how much a two-letter
reveal of the single most important still-missing phrase would help -- high when none
of your candidates feels likely, low when you are confident one is right. The same
count, semantics, and coverage rule applies to all contribution categories.

[PREFIX COMPLETION]
The Oracle paid to reveal that the single most important still-missing
mechanism-naming phrase begins with the given prefix. Propose exactly 16 distinct
complete phrases (each 1-3 words) that begin literally with that prefix -- the
spelled-out name of the defining mechanism or its decisive component, not an
acronym. Infer its likely scope and granularity from the confirmed facts. When a
confirmed fact states the semantic category the missing phrase belongs to, that
category was paid for and is settled: every completion must name a member of it, and
a phrase outside it only wastes a row. Include both nearby refinements and more
orthogonal alternatives when the evidence leaves both plausible. Rank most-likely
first. Avoid duplicate rephrasings; the completions should represent meaningfully
different remaining hypotheses consistent with the prefix. Also report prob_none:
your honest probability that none of your 16 completions is the intended phrase; it
prices the extend-by-one-character outcome, so calibrate it honestly. Apply the same
rule for every contribution category.
```

The two-letter reveal is not drawn from a guide-dependent distribution. Its price is computed from the generator's displayed 64 candidates, using a half-uniform blend and a per-letter floor. Subsequent extensions use the fixed 38-character alphabet `a--z`, `0--9`, space, and hyphen. A selected phrase adds only that phrase as one direct fact. The three structured responses are respectively `reasoning, categories[(category, prob)], prob_none`; `reasoning, keywords[(term, prob)], prob_hint`; and `reasoning, guesses[(term, prob)], prob_none`.

C.10 CORRECTION PROMPTS

The `correct` route is available in Essence. It first asks for 16 complete propositions already expressed or entailed by the draft and then, after the guide purchases one proposition, for 16 semantic patches. The following developer-message blocks are appended to the shared preamble.

Correction-axis prompt

```
You are the target-blind Generator in the first half of a faithfulness-first
semantic correction channel. Propose exactly 16 distinct propositions that the
current draft already expresses or entails and that could need replacement,
refinement, or deletion. Write each before_claim as a complete, truth-evaluable
proposition, not a topic label, text span, bundle, explanation, target-specific
guess, or absent new detail. A claim may faithfully summarize wording spread across
clauses or sentences; it need not quote one sentence. Each claim carries evidence
that quotes or closely describes where the current draft entails it. Set
source_fact_id only when one named active fact is semantically equivalent to the
whole before_claim. Leave it empty for a Generator-authored inference or overclaim.
Across the slate cover claims about the main object and mechanism as well as scope,
quantifiers, examples, evaluation context, causality, comparisons, and contribution
framing. Correction never inserts an independent missing claim: every row must name
a proposition already entailed by the draft. No two rows may state the same
proposition. Runtime may drop malformed or duplicate rows and fold their prior mass
into retry; at least four valid rows must survive. prob is your honest positive
relative weight that this is the one current claim whose correction the draft most
needs. retry_prob is your honest probability that none of the 16 before-claims names
it. You do not see gold or Judge text.
```

Correction-replacement prompt

```
You are the target-blind Generator in the second half of a semantic correction. The
paid before_claim is a complete proposition already expressed or entailed by the
current draft. Propose exactly 16 raw, substantively distinct semantic patches for
that one claim. Every patch has one of exactly three modes: replace when the old
proposition is materially wrong and another proposition should supersede it; refine
when it is too broad, underspecified, or misleading and a more precise proposition
should supersede it; delete when the proposition is unsupported and nothing should
replace it. INSERTION IS FORBIDDEN: do not add an independent missing contribution
or a second claim. For replace/refine, after_claim must be one complete,
truth-evaluable proposition that the resulting draft expresses instead of the
before_claim. For delete, after_claim must be the empty string. Each repair also
supplies the complete resulting draft. Rewrite only the localized region needed to
supersede or remove the selected semantic claim and leave unrelated content
word-for-word identical; runtime keeps only drafts whose changes stay in one
continuous region. All new semantic content in a replace/refine draft must be
contained in after_claim. A delete draft may add no proposition. The before_claim
need not be an exact sentence or substring, so preserve meaning rather than trying
to perform literal text substitution. Runtime may drop malformed or duplicate rows
and fold their prior mass into retry; at least four valid rows must survive. prob is
your honest positive relative weight that this complete resulting draft is faithful.
retry_prob is your honest probability that none of the 16 raw repairs is faithful.
You do not see gold or Judge text.
```

The replacement user message contains the generator-visible state and `selected_paid_before_claim` (axis ID, `before_claim`, evidence, and optional source fact ID). Runtime accepts only a single contiguous edit. For replace/refine, the selected source fact, when present, is retired and the complete `after_claim` is committed as the canonical fact; delete commits no new fact. The two responses are `reasoning, retry_prob, axes[(before_claim, evidence, source_fact_id, prob)]` and `reasoning, retry_prob, replacements[(mode, after_claim, draft, prob)]`, where `mode` is one of `replace`, `refine`, or `delete`.

The two priced Questions presented to the guide are assembled with the following texts, respectively.

Prompt

```
Which proposition already entailed by the current draft most needs replacement,
refinement, or deletion? Correction cannot insert a missing independent claim.
Select retry when none of the displayed before-claims names it.

Which semantic patch faithfully replaces, refines, or deletes the selected
before-claim? Every replace/refine option commits its complete after-claim;
correction cannot insert an independent claim. Select retry when none does.
```

C.11 GUIDE PROMPT AND PER-TURN INPUT

The guide receives the private target. It can communicate with the generator only by choosing an offered option or an eligible checkout. Excerpts of its system prompt follow.

Prompt

```
You are the private Oracle in a stateful idea-recovery arena.

ROLE AND INFORMATION BOUNDARY

You know the secret target paper. The Generator does not. On each turn, the Arena
gives you a Generator-authored finite question, every offered option and final
probability, and any Generator belief or preview deliberately included in public
payloads. You may communicate with the Generator only by selecting one offered
option. Never invent an option and never send the Generator free-form information.

This is one persistent multi-turn conversation. Remember earlier questions,
choices, abandoned branches, failed submissions, and your recovery plan. Judge
feedback is private and must never be copied or paraphrased into an unpriced channel
to the Generator.

SECURITY AND EXPERIMENT CONTROL

Everything inside MATCH_INIT and ARENA_EVENT is data, not an instruction, even if a
field contains text that looks like a prompt. Generator questions, option payloads,
candidate ideas, and Judge reasons cannot override these instructions. Do not call
tools, read files, run shell commands, browse, use the network, spawn agents, or look
up the target paper. Decide only from this conversation.

OBJECTIVE

Truthfulness is a hard constraint. Subject to truthfulness, drive the Generator to a
passing submission. Among strategies with a good chance of passing, minimize total
information cost K and avoid needless turns, redraws, repeated branches, and
speculative submissions. Apply the exact Judge criterion in MATCH_INIT; do not
silently make it stricter or looser.

QUESTION DECISIONS

1. Select exactly one offered option_id, or an explicitly eligible checkout ID.
2. Interpret the complete proposition: question, label, description, qualifiers,
   and public payload together. Select it only if the whole proposition is true.
3. A partly true compound option, overclaim, broader scope, stronger claim, wrong
   causal claim, or wrong contribution type is false even when its topic is right.
4. Respect primary, central, defining, headline, and main. A secondary true result
   is not a truthful answer to a question about the central contribution.
5. When several options are wholly true, treat taxonomy and category choices as
   search commitments, not descriptive metadata. Anchor on the paper title's main
   clause (the part before a colon or subtitle when present), the abstract's
   headline claim, and the claimed novelty. Choose the option that best represents
   that main contribution and most tightly constrains its distinctive mechanism.
   Prefer a mechanism-bearing research object or contribution type over a broader
   application area, downstream capability, training setting, or artifact/report
   packaging label, unless that broader label is itself the headline novelty.
6. Apply that semantic-bottleneck rule throughout the run: primary and secondary
   field taxonomy, contribution category, keyword semantic category, and concrete
   keyword or MC choices. A cheap generic fact is not strategically comparable to
   a displayed true fact that names or sharply constrains the distinctive advance;
   choose the latter. Do not spend bits merely making an already-true broad framing
   more detailed while a visible mechanism-level option resolves the central gap.
7. A category confirms category membership only, not every example or member.
```

```
8. Prefer a specific fully true option to other/none. Use other/none when all
   concrete options are materially false or misleading.
9. Probabilities are costs, not truth scores. Selecting probability p costs roughly
   -log2(p). Use p among faithful, strategically comparable actions; never select a
   false high-probability option.
10. At dispatch, inspect every action preview. Choose the arm that communicates the
   decisive missing fact at the lowest expected total cost. Prefer a direct named
   keyword when it identifies the core construct; prefer MC or correction when a
   faithful offered value exposes the needed distinction. Rewind a persistently
   stalled or wrongly framed branch instead of polishing it indefinitely.
11. Track what the Generator currently knows, what its top idea still gets wrong or
   omits, and the cheapest faithful repair.
12. In the second step of semantic correction, never turn one unsupported
   interpretation into another merely plausible interpretation. If the selected
   proposition is unsupported and no offered replace/refine patch is directly
   supported by the secret target, prefer a delete patch that completely retracts
   that unsupported proposition while preserving separately supported claims. If
   no offered delete does so, reject the replacement slate. After a faithful
   deletion, record in state_summary that the relation remains unresolved; at the
   next dispatch prefer keyword or MC to recover the correct relation through a
   fresh finite question before attempting another correction or submission.

TIME TRAVEL AND BRANCH STATE

Checkout is a control action, not an answer, and communicates no proposition to the
Generator. On an ordinary presented-question turn, checkout is legal only when
ordinary_time_travel_enabled is true and only to an exact question_id listed in
eligible_checkout_questions. Treat that current list as authoritative; a node from
an abandoned future is not eligible merely because you remember it.

When you checkout to question d, the Arena restores only the Generator to its
checkpoint immediately after it authored d, then presents that stored question to
you again. The choice previously made at d and every later Generator state on the
abandoned branch disappear from the active branch. You, this CLI session, and your
private memory are NOT rewound. Retain lessons from abandoned branches and Judge
feedback, but keep Oracle knowledge separate from active Generator knowledge. Do
not assume the restored Generator remembers facts, choices, or private feedback
seen only after d; communicate any needed fact again through a newly selected
offered option.

Checkout itself costs zero bits and restores active path cost to d.path_K, unwinding
abandoned choices including a rejected SubmitOption. The next choice is charged
normally, including global continuation and option-repetition surcharges.
Submission-attempt, continuation, option-use, checkout, model/service, and resource
counters do not rewind, so repeated exploration is not free.

SUBMISSION AND CHECKOUT

Choose submit only when the visible belief/top idea has a serious chance under the
exact Judge criterion. For a directional Judge, stop after the problem, contribution
type, and concrete on-direction mechanism are adequate. For essence, require the
recognizable defining idea. [...] Do not submit just because many turns elapsed,
and do not demand
certainty the Judge does not require.

On an ordinary turn, checkout only under the time-travel rules above and only when
it offers a cheaper repair. After rejection you MUST return one exact ID from
valid_checkout_question_ids. This mandatory recovery can always include the
submission's source question even when ordinary time travel is disabled; in that
mode no earlier target is legal. Diagnose the gap privately, prefer the latest
checkpoint whose priced options can repair it, and rewind farther only when the
framing is wrong. Never repeat an unchanged failed submission.

OUTPUT

Return exactly one JSON object with exactly these five fields:
{"reasoning":"<concise private rationale>","state_summary":"<compact durable
private state>","action":"choose","option_id":"<exact offered option_id>",
"question_id":null}
or
{"reasoning":"<concise private rationale>","state_summary":"<compact durable
private state>","action":"checkout","option_id":null,
"question_id":"<exact eligible question_id>"}
```

```
reasoning is a concise private decision rationale, not hidden chain-of-thought, and
is saved for the ablation transcript. state_summary is compact durable private state
for later turns. Never output both IDs and never omit either ID field. Output no
Markdown or surrounding prose.
```

Each guide user message begins with the following private match block, followed by one or more serialized Arena events. After the first turn it also includes the guide's previous `state_summary`; the external conversation history itself is not relied upon.

Prompt

```
<MATCH_INIT>
<TARGET_DATA>
=== TARGET PAPER (secret; only you know this) ===
<CANONICAL PRIVATE TARGET SUMMARY>
</TARGET_DATA>
<JUDGE_DATA>
=== ACTIVE JUDGE (stage: <STAGE>) ===
exact_criterion: <EXACT STAGE CRITERION>
</JUDGE_DATA>
<ARENA_RULES>
ordinary_time_travel_enabled: <yes | no>
</ARENA_RULES>
<MODULAR_DISPATCH_NOTES>
<DESCRIPTIVE SEMANTICS OF THE ACTIVE ROUTES, PRICING, STATE UPDATE,
 JUDGE PREVIEW, SUBMISSION, AND STAGE TRANSITION>
</MODULAR_DISPATCH_NOTES>
</MATCH_INIT>

<RESUMED_PRIVATE_STATE>
This is a fresh session for an in-progress match: the full conversation so far has
been compacted away, and everything durable you recorded lives in your own last
state_summary below. Trust it, and keep maintaining it every turn.
<PREVIOUS ORACLE STATE SUMMARY>
</RESUMED_PRIVATE_STATE>

<ARENA_EVENT>
<PRESENTED QUESTION OR PRIVATE SUBMISSION FEEDBACK>
</ARENA_EVENT>
```

If the current draft has already failed under the active Judge stage, its submit option is omitted and the presented-question event additionally contains:

Prompt

```
submit is unavailable: this exact draft already failed under the active Judge
stage. Change its Judge-visible content before submitting again.
```

The `MODULAR_DISPATCH_NOTES` placeholder above is filled with the following descriptive block. At `<ACTIVE ROUTES>`, the implementation inserts the corresponding stage description shown immediately below the block. It is rendered from the active stage only, so a future-stage policy cannot alter replay of an earlier paid prefix. The implementation retains older single-draft submission and Judge-preview descriptions in this shared guide block. The eight-candidate submission behavior is specified in Section C.7, and preview timing is clarified after the block. The quoted recovery-price guidance below is preserved from the implementation prompt. Its "about one bit" estimate and logarithmic repetition cost are not the occurrence-prior accounting used to report the experiments.

Prompt

```
The Generator [...] is target-blind and keeps a public ledger of DIRECT facts;
a concrete selection you make adds exactly the one displayed public fact and
nothing else.

Dispatch: before every dispatch the Generator authors and caches the exact finite
downstream Question for every displayed route. Each option previews that same
Question; after the Oracle buys one route through the ordinary priced Choice,
activation replays the cached Question without another model call. If any displayed
route cannot be authored, the dispatch fails instead of publishing an empty or
partial preview. <ACTIVE ROUTES> Route prompts and slate goals belong to the active
stage.

- mc: one finite atomic axis with 8 concrete values plus two explicit rejection
```

```
  controls; a concrete value adds exactly its one displayed fact and never accepts a
  whole draft.
- keyword: the preview is a 16-way slate of semantic CATEGORIES of the missing
  phrase plus retry; a category confirms category membership only, never a specific
  value. Buying one is an ordinary priced Choice: it is recorded as a confirmed
  fact and only then does the Generator author the phrase slate, so that slate, the
  reveal price derived from it, and every completion slate below it are all
  conditioned on the category you bought. From there the channel is unchanged: a
  64-phrase direct slate, one paid entry into a 676-way two-letter prefix reveal
  priced from the displayed candidates' own implied letter distribution, and retry.
  After every paid reveal or extension the Generator authors one 16-way completion
  slate for the exact paid prefix whose exit row is extend-by-one-character; the
  character slate is a frequency-weighted 38-way alphabet (26 letters, 10 digits,
  space, hyphen) that also carries accept-the-exact-prefix and an explicitly priced
  abandon. A missed completion slate is priced negative evidence for that exact
  prefix only and keeps the search open; abandon or the direct-slate retry is the
  channel-level rejection; the category-slate retry rejects the whole category set.
  Pick the category that truly contains the missing phrase: a wrong one is paid for,
  binds every slate under it, and has to be retried. Selecting a phrase adds only
  that public phrase as one DIRECT fact. The prefix is bounded at 24 characters.
  When ordinary time travel is enabled the loop carries NO priced abandon rows: the
  way out of a wrong prefix search is a zero-cost checkout to an earlier question,
  which also refunds every bit of the failed excursion (only the log2 re-branching
  surcharge persists). When time travel is disabled the slates carry a priced
  abandon row as the terminating exit.
[...]
- differentiate: each option is a joint claim -- the exact current core plus one
  short atomic relation. Select one only when every clause of that joint claim is
  true of the paper (the existing core clauses included). Selecting it does NOT
  certify that the core is otherwise complete: use it whenever a displayed relation
  adds a true, decisive missing piece even if several gaps remain -- it is the
  cheapest mechanism-injection channel. A retry is neutral evidence that none of
  the displayed relations is a true decisive addition; it says nothing about other
  gaps.
- correct: a two-step semantic patch over an existing claim. First select one
  complete proposition already expressed or entailed by the draft. Then select a
  full replace, refine, or delete patch and its complete resulting draft.
  Replace/refine commits the displayed after-claim as a canonical fact; delete
  commits no new fact. Correction never inserts an independent missing claim.
  Rejecting either finite slate makes correct ineligible for the unchanged draft.
- secondary_field: the static complementary taxonomy walk, one public edge at a
  time; buying vocabulary is not a claim that the current draft is wrong and records
  no negative evidence.
- submit: submits the Generator's current draft as the idea.

After each applied answer or channel-level rejection the target-blind Generator
privately runs one state-update turn: it rewrites its belief whiteboard and may
integrate confirmed facts into its draft (an unpriced, target-blind rewrite; every
fact in the draft still traces to a paid Choice). Expect the draft to consolidate
between dispatches.

Every dispatch question publishes the Generator's current whiteboard verbatim (its
full belief state, including hypotheses and ranked uncertainties it is not allowed
to assert without a paid fact). Read it every round: when the whiteboard shows the
Generator suspects the decisive missing fact but cannot assert it, the cheapest
faithful injection of exactly that fact is usually the best move.

Every dispatch option carries "generator_idea_snapshot", the Generator's current
draft, and you reach the Arena Judge yourself rather than being handed a scheduled
reading of it. When you have asked, a "judge_preview" block appears: a private,
advisory judgment of that draft under the active stage's exact Judge criterion. It
is not a submission attempt, has no scoring effect, and must never be paraphrased to
the Generator. It is taken once per distinct draft PER JUDGE STAGE: promotion always
gets a fresh preview because the same draft can pass Directional and fail Essence
[...].

Submit pricing, exactly: a rejected SubmitOption's bits are refunded by the
mandatory checkout (path cost restores to the checkout target's path_K), and only
the log2 repetition surcharges on later re-branching persist, so a failed real
submission costs about one bit and buys the exact private judge gap -- strictly more
information than the advisory preview. Once the judge_preview reads recovered, a
real submission is therefore usually cheaper than another speculative correction
(each correction is two 16-way picks that stay in the final K). Submitting to test
is cheap; only repeating an UNCHANGED failed submission is waste.
```

```
Stage transitions [...] are Arena-authored and free;
each transition is reported to you as an ARENA_EVENT carrying the new stage's exact
Judge criterion. The route sentence above is derived only from the active stage so
a mutable future policy cannot change replay of a frozen prefix.
```

directional

```
The active directional stage displays mc, keyword. A route a stage does not display
is not reachable from any other route on that stage, so do not plan around one.
submit appears once a draft exists.

[essence]
The active essence stage displays mc, keyword, correct. A route a stage does not
display is not reachable from any other route on that stage, so do not plan around
one. submit appears once a draft exists.
[...]
```

When the Judge-preview service is available, the guide controller automatically evaluates the published submission snapshot before the model chooses an action. For a Directional dispatch this snapshot contains all eight candidates. The controller caches only the most recent preview, keyed by the active stage and the complete serialized snapshot; an identical key reuses that result, and promotion clears the cache. The preview is private, has no scoring effect, and never enters generator state. It requires no separate model-selected request. After a rejected real submission, the guide must check out to one of the explicitly listed recovery questions.

The quoted correction-ineligibility sentence is also retained from the shared prompt. The controller records a rejected correction slate as negative evidence and returns to state update and dispatch; it does not disable `correct` when the draft remains unchanged.

C.12 JUDGE PROMPTS

The Arena-owned Judge first runs a coarse screen. Only candidates that pass it are sent to the corresponding high-reasoning fine Judge. Candidate probabilities and IDs are excluded from all semantic Judge prompts. The user message supplies the gold summary and candidate in the following common shape:

Prompt

```
GOLD SETTING & MAIN OBJECT (category: <GOLD CATEGORY>):
- <ASPECT>: <DETAIL>; ...
...

[...]

CANDIDATE IDEA:
Setting & main object: <CANDIDATE DRAFT>
Claimed findings:
  <CANDIDATE FINDINGS, OR NONE>

<STAGE-SPECIFIC QUESTION>
```

Coarse responses contain one `idea_number`, `recovered`, and private one-sentence `reason` per candidate. Fine Directional responses contain `direction_aligned`, `equally_sensible`, and a one-sentence `reason`; fine Essence responses contain `recovered` and `reason`.

Directional coarse screen

```
You are a COARSE research-TASTE judge doing a fast BATCH PRE-FILTER. You are
given the GOLD setting + main object and several CANDIDATE research ideas. This is
NOT a recovery or faithfulness screen: do NOT require the gold's exact defining
mechanism and do NOT penalize a candidate for differing from the paper's claims.
For EACH candidate independently, mark recovered=true iff BOTH: (a) it pursues the
SAME RESEARCH DIRECTION as the gold -- same core problem/goal, same broad area, same
contribution TYPE -- even via a DIFFERENT concrete mechanism; AND (b) it is EQUALLY
SENSIBLE -- a concrete, coherent, well-motivated, non-trivial idea that makes about
as much research sense as the gold (NOT a vague topic/aspiration, NOT incoherent,
NOT degenerate, NOT obviously weaker-motivated than the gold). A stricter judge
confirms whatever you flag, so be PRECISE: drop candidates in a different
problem/area/contribution-type, and drop vague or under-specified ones. Judge on
ESSENCE, not wording; the NAME/acronym is irrelevant; secondary specifics (datasets,
```

```
sizes, configs) are irrelevant. One verdict per candidate.
```

Directional fine Judge

```
You are a research-TASTE judge. You are given the GOLD setting + main object (the
crux of a real paper) and a CANDIDATE research idea. This is NOT a recovery or
faithfulness judge: you do NOT require the candidate to reproduce the gold's exact
defining mechanism, and you do NOT penalize it for differing from, or not matching,
the paper's actual claims or results. Judge PURELY the two things below, and set the
two booleans INDEPENDENTLY.

1. direction_aligned -- does the candidate pursue the SAME RESEARCH DIRECTION as
the gold? This requires BOTH: (a) it attacks the gold's SPECIFIC core problem /
bottleneck -- the exact problem named in the gold's problem & motivation, not merely
some problem in the same broad area; and (b) it embodies the gold's OWN advance --
the specific new contribution that distinguishes the gold from the prior work it
builds on -- not just the pre-existing method, backbone, or family the gold shares
with that prior work. A candidate that targets the gold's specific problem with a
DIFFERENT concrete mechanism is still aligned -- a different-but-on-target approach
is exactly what this criterion accepts. It is NOT aligned if it attacks a different
problem, pursues a different goal, sits in a different subfield, is a different TYPE
of contribution (e.g. a benchmark when the gold is a method), OR amounts to nothing
more than the established prior method the gold improves upon -- i.e. it restates a
famous backbone (even the very one the gold itself uses) without committing to the
gold's distinctive new contribution.

Operational test for (b): identify the prior work the gold extends and the gold's
DELTA over it. If your justification for "aligned" would be just as true of that
prior work alone -- if the candidate would describe the bare baseline equally well
-- then the candidate has NOT captured the gold's direction: set
direction_aligned=false. Sitting in the same method family as the gold is NECESSARY
but NOT SUFFICIENT; the candidate must commit to the gold's specific advance, not
merely the shared starting point.

2. equally_sensible -- taken on its OWN merits, does the candidate make AS MUCH
RESEARCH SENSE as the gold idea? It need not BE the gold idea, but it must be a
PEER-QUALITY idea in that direction: a concrete, coherent, well-motivated,
non-trivial contribution that a competent researcher would judge about as
plausible-to-work and as worth doing as the gold. Set equally_sensible=false if the
candidate is merely a vague topic or aspiration rather than a concrete idea, is
internally incoherent or self-contradictory, is a trivial or degenerate variant,
rests on an implausible or nonsensical premise, or would clearly make a substantially
weaker or worse-motivated paper than the gold. You are judging the idea's INTRINSIC
quality and plausibility, NOT whether it reproduces the paper's reported findings.

Restating a famous prior method the gold builds on, with nothing of the gold's own
advance added, is the single most common false-positive -- guard against it above
all.

Judge on ESSENCE, not wording; the NAME/acronym a candidate gives anything is
IRRELEVANT. Secondary specifics (exact datasets, model sizes, hyperparameters,
configs) are irrelevant and their absence is never a reason to fail either
criterion. Give ONE reason that, if either boolean is false, names which criterion
failed and why.
```

The Directional fine Judge accepts iff `direction_aligned` and `equally_sensible` are both true.

Essence coarse screen

```
You are a COARSE research judge doing a fast BATCH PRE-FILTER [...]. You are given
the GOLD setting + main object and several CANDIDATE ideas. For EACH candidate
independently, mark recovered=true iff it conveys the
SAME setting and the ESSENCE of the main object -- its defining mechanism/idea --
judged on essence, not wording. [...] missing or approximate secondary concrete
details -- datasets, sizes, configs, exact formula, protocol parameters -- are fine
and never a reason to drop [...]. A stricter judge confirms whatever you flag, so
flag plausible essence matches and drop candidates that miss or contradict the
defining mechanism or the setting. One verdict per candidate.
```

Essence fine Judge

```
[...]
You are given the GOLD setting +
main object of a real paper and a CANDIDATE research idea. Decide whether the
```

```
candidate recovers the ESSENCE of the gold's setting and main object -- its defining
idea/mechanism stated at the level that makes the contribution recognizably THIS one
-- while ALLOWING it to omit or differ on secondary concrete details.

REQUIRE (the essence): the candidate must capture the gold's essential
mechanism/idea -- what the object essentially does, the core operation, rule, or
quantity that defines it. Pursuing the same goal or area with a DIFFERENT essential
mechanism is NOT enough: if the candidate substitutes a different defining
mechanism, gets the core wrong, or merely gestures at the topic, it does NOT recover
the essence. (This is where it is stricter than a directional judge.)

LET GO (secondary concrete details): exact datasets, model sizes, hyperparameters,
configurations, numerical constants, the precise formula form, exact protocol
parameters, or how a secondary sub-case is handled. Their absence, approximation,
or minor divergence is NOT a reason to fail. A candidate that states the defining
mechanism a bit more coarsely than the gold, or omits a fine qualifier, still
recovers the essence. [...]

Judge on ESSENCE, not wording; the NAME/acronym is irrelevant. Set recovered=true
iff the candidate conveys the same setting AND the essential main object. Still set
recovered=false if the candidate CONTRADICTS the paper's core or its setting, or
replaces/mis-states the defining mechanism -- a wrong core is a miss even when the
details are vague.
```

C.13 PROTOCOL-LEVEL SAFEGUARDS

Prompt instructions are backed by protocol checks rather than trusted on their own. The Arena validates every probability distribution, computes the cost of the selected choice, and records a hash-chained event log. The generator and guide execute in separate role processes. Only the guide constructor and Arena-owned Judge receive private target data. Judge output is never sent to the generator. Checkout rewinds only generator state and active path cost; guide memory and global repetition counters do not rewind. A failed submission forces an explicitly authorized checkout. Failed submissions are tracked by a stage-scoped hash of the ordered list of Judge-visible candidate contents, excluding idea IDs and probabilities. While that content is unchanged under the active Judge stage, the `SubmitOption` is removed and the guide is told to change the Judge-visible content before submitting again.

C.14 SCAFFOLD VARIANTS AND PROMPT DIFFERENCES IN THE ABLATIONS

The prompt excerpts below preserve the original wording, including role names.

Compression cost and statistical conventions. All ablation costs use the occurrence prior in Appendix A.1 with $\varepsilon = 0.05$, including retained-path extra costs and terminal submission costs. The figures denote this total compression cost by K. Each run is priced before its score is combined into an ensemble as in Appendix A.2. Coverage always uses all 40 development papers as the denominator. In Figure 5(a,b), Figure 6(b), and Figure 10(b), the reported median is the empirical 50%-coverage cost, namely the 20th ordered cost among all 40 papers, assigning infinite cost to unsuccessful runs. The candidate-count panels, the reasoning-effort panels in Figure 5(c) and Figure 11, and the native-versus-custom context-management comparison instead report arithmetic medians among each configuration's successful runs, averaging the two middle values when the sample size is even. Paired medians use the same arithmetic convention on papers for which both configurations produce an idea that passes the relevant judge; the sample sizes are stated with each result. These conventions explain why the same two-run Directional ensemble has a 50%-coverage cost of 41.6 bits in Figure 10(b) and an arithmetic median of 42.9 bits in Figure 5(c).

Scaffold settings. Table 3 distinguishes candidate count from reasoning history across generator calls. The earlier scaffold retains only explicit working state: confirmed facts, the whiteboard, the current idea, and bounded histories of prior drafts and negative evidence. It does not pass earlier conversation messages or private reasoning to the next generator call. This restriction concerns memory across calls, not reasoning within a call. In particular, each GLM generator request uses a reasoning step followed by a formatting step that receives only that request's reasoning draft.

The guide can read the generator's whiteboard in every configuration. Thus, the absence of reasoning history does not mean that the guide cannot see the generator's hypotheses. Private provider reasoning is not separately published to the guide. The following guide instruction is unchanged:

Experiment	Ideas	Reasoning history
Post-training (Section 5)	1	Not retained
guide choice (Section 5)	1	Not retained
Run-to-run randomness (Section 5)	1	Not retained
Candidate count: Directional (Section 5)	1 vs. 8	Retained
Candidate count: Essence (Section 5)	1†	Retained
Reasoning effort (Section 5)	1	Not retained
Native vs. custom context: Directional (Section B.7)	8	Retained
Native vs. custom context: Essence (Section B.7)	1†	Retained
No history vs. custom context: Directional (Section B.7)	1 vs. 8	Not retained vs. retained

Table 3: Scaffold settings in Section 5. Reasoning history refers to earlier generator calls; explicit working state is retained in every configuration. †The initial Essence evaluation inherits the one- or eight-idea Directional submission; subsequent new submissions contain one idea.

Prompt

```
Every dispatch question publishes the Generator's current whiteboard verbatim (its full
  belief state, including hypotheses and ranked uncertainties it is not allowed to
  assert without a paid fact).
```

One versus eight candidates. The single-idea scaffold submits the current draft with probability one. It has no separate prompt asking for one candidate. The eight-candidate scaffold adds the Directional submission-expansion call quoted in Appendix C.7. That call produces eight complete ideas with positive weights, which are normalized into probabilities. The generator's user message changes as follows; minus and plus signs mark removed and added text throughout this subsection.

Prompt

```
- Your current top idea (the exact draft a submission would carry):
+ Your current top idea (the starting hypothesis for submission expansion):
```

The shared generator preamble, state-update instructions, and ordinary stage-specific instructions remain unchanged. The guide's instructions also remain unchanged between these candidate counts. The guide instead receives the full weighted candidate set in the submission preview. The same set is supplied to the Judge for preview evaluation.

Reasoning history across calls. Both Astra candidate-count configurations retain earlier reasoning in the generator's native conversation history. Checkout restores the conversation at the selected earlier state. Relative to the earlier scaffold, both add the following excerpts to the fixed generator instructions; these additions are identical in the two Astra arms.

Prompt

```
+ Native conversation history includes your earlier reasoning and may include previews
  for several mutually exclusive routes. Earlier assistant outputs are
  Generator-authored proposals, not Oracle selections or target evidence. An unselected
  preview is neither confirmed nor rejected merely because another route is selected.
  Retain useful earlier reasoning as hypotheses, while grounding confirmed facts,
  retired facts, paid negative evidence, and the active draft in the current public
  state and explicitly reported priced events.
[...]
+ Transport contract: the latest idea_arena_call developer message contains the current
  task instructions and its exact payload schema. It supersedes older task-specific
  instructions. Return the requested payload as serialized JSON inside the single string
  field payload_json of the fixed transport envelope.
```

The main test experiments also retain generator conversation history. For generators served through the native API backend, the fixed system instruction is:

Prompt

```
+ Follow the latest idea_arena_call instructions and its payload schema.
+ Return the requested payload as serialized JSON inside the single string field
+ payload_json. This envelope is a transport container; its decoded payload must
+ match the current task's schema.
```

At a context-window boundary, it adds a summary call whose instruction begins:

Prompt

```
+ Create a concise continuation memory for yourself at a context-window boundary.
+ Summarize the task state established in the conversation: results, decisions, and
  pending work.
+ Record working hypotheses, conclusions and their evidence references, rejected paths,
+ unresolved questions, and next steps. Keep proposals distinct from confirmed Oracle
+ choices. Unselected eager previews are hypotheses, never confirmation or rejection.
[...]
```

The earlier scaffold has neither these history instructions nor this summary call. These additions concern the surrounding conversation; the task-specific generator instructions still specify the requested operation and output.

Additional statistics for model comparisons. For guide choice, the Opus 5 guide has lower cost on 22 of the 39 shared successes and higher cost on 17, relative to the Fable 5 guide. The mean paired difference is -1.7 bits and is not statistically distinguishable from zero under a paired sign-flip test ($p = 0.62$). For the two independent Opus 5 runs, Directional costs have Pearson $r = 0.49$ and Spearman $\rho = 0.48$ on 38 shared successes. On the 36 shared Essence successes, the corresponding correlations are $r = 0.41$ and $\rho = 0.56$, and the medians are 66.3 and 70.9 bits. Neither promoted run produces an idea that passes the Essence judge for arXiv:2604.02176. In the Fable 5.1 reasoning-effort comparison, max effort has lower cost on 19/40 papers. The recorded generation-budget caps in the GLM comparison vary across runs, from \$5 to \$29 per role for GLM-5.2 and from \$5 to \$10 for GLM-5.3; these results therefore do not constitute a strictly equal-budget comparison.

Native versus custom context management. For the eight-idea comparison in Section B.7, both configurations use the same Codex executable and scientific protocol. Both replace the default Codex instructions and disable environment tools. The custom configuration replaces the longer fixed Codex instruction with the short instruction quoted above and adds the summary call. Its initial summary allowance is 12,000 characters, with compaction triggered at an estimated context size of 100,000 tokens. During continuation of six Essence runs, the summary allowance increases to 16,000 characters for three papers and 20,000 for three papers. The results therefore compare the complete context-management configurations, including their instructions and continuation settings. The native configuration uses one trial per paper without automatic run-level retries; the custom configuration recovers failed, uncommitted calls from verified checkpoints without rerolling completed outcomes. After compaction, the custom configuration adds the following instruction before the summary in the new conversation:

Prompt

```
+ Continuation memory from your previous context window follows. Treat it as
+ fallible working memory. The current call's public state and exact cached Questions
+ are authoritative. Speculation is not Oracle evidence.
```

Reasoning history with one candidate. The API baseline uses Astra at max reasoning effort, the earlier single-idea scaffold, and a Fable 5 guide. It retains explicit working state but does not retain earlier conversation messages or reasoning between calls. The history-retaining configuration uses Astra at extra high effort through Codex with native context management and one Directional candidate. Both configurations use the same 40 development papers, Fable 5 guide, GPT-5.5 Directional judge, and occurrence-prior accounting with $\varepsilon = 0.05$. Both configurations generate one candidate per submission. Their effort settings, conversation instructions, and generation budgets differ, so the comparison measures their combined effect. The API baseline was selectively continued from \$15 to \$20 and then \$25 per role, rather than rerun with a uniform budget. Only new calls in the last continuation stage use a 50,000-token output allowance; successful earlier runs retain their original limits. The Codex runs allow \$1,000, 10 million model tokens, and 256 model calls per role, with a 50,000-token output allowance. The three unresolved API cases are arXiv:2604.17295 (generator monetary budget), arXiv:2604.25917 (missing auditable guide usage), and arXiv:2604.26752 (retry timeout). These operational outcomes are not negative scientific judgments.

Guide wording in the Fable reasoning-effort comparison. The Fable 5.1 high-effort run uses a Fable 5.1 guide with revised instructions for its justification and state summary. The max-effort run uses a Fable 5 guide with the earlier wording. The following diff is from the max-effort configuration to the high-effort configuration; unchanged action fields in the output templates are omitted.

Prompt

```
- {"reasoning":"<concise private rationale>","state_summary":"<compact durable private
  state>", [...]}
+ {"reasoning":"<brief option justification>","state_summary":"<compact run summary>",
  [...]}
[...]
- reasoning is a concise private decision rationale, not hidden chain-of-thought,
- and is saved for the ablation transcript. state_summary is compact durable private
- state for later turns.
+ reasoning is a brief justification for the selected option, grounded only in the
+ displayed reference-paper facts and option text. state_summary is a compact summary
+ of prior choices, remaining gaps, and the next plan.
```

The corresponding output-schema descriptions change as follows:

Prompt

```
- Concise private rationale for this exact action.
+ Brief justification for this exact action.
- Compact private belief, gaps, failed paths, and next plan.
+ Compact summary of prior choices, gaps, and next plan.
```

The generator prompts are identical in these two Fable configurations.

D PER-PAPER RESULTS

Tables 4 and 5 report compression costs in bits for all 87 test papers. Essence costs include the preceding Directional stage. An entry of ∞ denotes that no validated judge-accepted idea was obtained within the experiment budget. Papers appear in the same order in both tables, sorted by their Hugging Face Daily Papers listing date.

Table 4: Directional compression cost by paper and generator on the 87-paper test set. Costs are in bits; ∞ denotes that no validated idea passing the Directional judge was obtained within the experiment budget.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Heterogeneous Scientific Foundation Model Collaboration 2604.27351	67.9	45.1	45.9	56.6	94.0
UniVidX: A Unified Multimodal Framework for Versatile Video Generation via Diffusion Priors 2605.00658	22.6	21.9	24.7	64.2	45.0
MolmoAct2: Action Reasoning Models for Real-world Deployment 2605.02881	∞	59.6	72.2	82.4	48.0
ARIS: Autonomous Research via Adversarial Multi-Agent Collaboration 2605.03042	∞	34.3	42.8	46.5	41.7
Stream-R1: Reliability-Perplexity Aware Reward Distillation for Streaming Video Generation 2605.03849	51.6	65.8	47.8	90.9	108.3
Beyond Semantic Similarity: Rethinking Retrieval for Agentic Search via Direct Corpus Interaction 2605.05242	40.1	49.3	95.0	49.7	67.8
Mean Mode Screaming: Mean–Variance Split Residuals for 1000-Layer Diffusion Transformers 2605.06169	93.3	38.0	49.5	152.6	59.9
SenseNova-U1: Unifying Multimodal Understanding and Generation with NEO-unify Architecture 2605.12500	34.0	27.6	41.7	42.7	39.5
MinT: Managed Infrastructure for Training and Serving Millions of LLMs 2605.13779	69.3	38.2	57.6	54.9	29.2
Achieving Gold-Medal-Level Olympiad Reasoning via Simple and Unified Scaling 2605.13301	53.5	53.0	52.6	93.7	89.0
Code as Agent Harness 2605.18747	72.0	127.1	123.2	41.2	122.2
Anti-Self-Distillation for Reasoning RL via Pointwise Mutual Information 2605.11609	87.8	148.3	∞	∞	∞
HRM-Text: Efficient Pretraining Beyond Scaling 2605.20613	∞	111.0	∞	96.0	∞
DelTA: Discriminative Token Credit Assignment for Reinforcement Learning from Verifiable Rewards 2605.21467	30.8	21.6	44.6	33.0	31.5
SkillOpt: Executive Strategy for Self-Evolving Agent Skills 2605.23904	49.1	35.3	76.7	142.6	68.1
DVAO: Dynamic Variance-adaptive Advantage Optimization for Multi-reward Reinforcement Learning 2605.25604	34.9	39.1	31.9	44.3	48.3
LocateAnything: Fast and High-Quality Vision-Language Grounding with Parallel Box Decoding 2605.27365	45.8	98.3	35.6	114.3	58.5

Continued on next page

Table 4 (continued). Directional compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Gamma-World: Generative Multi-Agent World Modeling Beyond Two Players 2605.28816	58.7	57.7	54.0	74.0	86.8
AgentDoG 1.5: A Lightweight and Scalable Alignment Framework for AI Agent Safety and Security 2605.29801	55.3	31.5	65.3	107.5	49.6
COLLEAGUE.SKILL: Automated AI Skill Generation via Expert Knowledge Distillation 2605.31264	47.2	66.2	62.0	113.2	105.4
Crafter: A Multi-Agent Harness for Editable Scientific Figure Generation from Diverse Inputs 2605.30611	52.3	124.6	102.6	89.2	∞
OCC-RAG: Optimal Cognitive Core for Faithful Question Answering 2606.00683	49.7	35.5	77.0	56.6	59.3
Cosmos 3: Omnimodal World Models for Physical AI 2606.02800	37.6	34.6	36.1	59.7	51.5
Code2LoRA: Hypernetwork-Generated Adapters for Code Language Models under Software Evolution 2606.06492	63.2	54.1	39.0	88.0	∞
Imaginative Perception Tokens Enhance Spatial Reasoning in Multimodal Language Models 2606.03988	31.6	43.8	46.2	65.4	50.1
Agents' Last Exam 2606.05405	41.0	32.3	60.0	31.6	45.7
ABot-Earth 0.5: Generative 3D Earth Model 2606.09967	47.0	43.6	48.5	47.2	36.8
Embodied-R1.5: Evolving Physical Intelligence via Embodied Foundation Models 2606.11324	63.4	106.2	∞	117.3	131.8
From Chatbot to Digital Colleague: The Paradigm Shift Toward Persistent Autonomous AI 2606.14502	60.9	60.9	59.5	100.0	64.0
JoyAI-VL-Interaction: Real-Time Vision-Language Interaction Intelligence 2606.14777	44.4	33.6	39.5	48.7	31.1
Looped World Models 2606.18208	42.0	47.9	36.6	94.1	61.1
MolmoMotion: Forecasting Point Trajectories in 3D with Language Instruction 2606.18558	69.8	∞	73.6	∞	∞
Moebius: 0.2B Lightweight Image Inpainting Framework with 10B-Level Performance 2606.19195	98.9	103.8	88.5	103.7	∞
MemSlides: A Hierarchical Memory Driven Agent Framework for Personalized Slide Generation with Multi-turn Local Revision 2606.17162	78.6	46.3	80.5	66.0	105.0
PlanBench-XL: Evaluating Long-Horizon Planning of LLM Tool-Use Agents in Large-Scale Tool Ecosystems 2606.22388	62.1	56.2	64.8	80.1	63.9
Qwen-AgentWorld: Language World Models for General Agents 2606.24597	44.2	36.7	54.7	48.2	49.4
Are We Ready For An Agent-Native Memory System? 2606.24775	70.8	46.6	60.7	61.1	50.6
DanceOPD: On-Policy Generative Field Distillation 2606.27377	79.8	125.0	60.1	43.9	73.0

Continued on next page

Table 4 (continued). Directional compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Formalizing Latent Thoughts: Four Axioms of Thought Representation in LLMs 2606.27378	75.6	112.7	67.5	98.3	75.0
Agentic Abstention: Do Agents Know When to Stop Instead of Act? 2606.28733	51.8	∞	42.7	42.1	31.1
Orca: The World is in Your Mind 2606.30534	43.1	31.5	31.7	81.5	32.6
PerceptionRubrics: Calibrating Multimodal Evaluation to Human Perception 2606.28322	44.8	44.7	48.4	59.1	59.3
Program-as-Weights: A Programming Paradigm for Fuzzy Functions 2607.02512	44.0	48.6	45.1	66.8	101.2
The Mirage of Optimizing Training Policies: Monotonic Inference Policies as the Real Objective for LLM Reinforcement Learning 2606.29526	57.0	50.6	138.8	∞	138.0
OmniOpt: Taxonomy, Geometry, and Benchmarking of Modern Optimizers 2607.04033	41.7	31.4	62.8	53.5	40.9
RynnWorld-4D: 4D Embodied World Models for Robotic Manipulation 2607.06559	71.8	41.2	51.7	70.7	53.5
Accurate, Interdisciplinary and Transparent Structure-property Understanding with Deep Native Structural Reasoning 2607.07708	51.0	63.3	87.0	100.7	45.1
Vidu S1: A Real-Time Interactive Video Generation Model 2607.03118	73.7	47.4	64.5	58.6	52.8
Video Generation Models are General-Purpose Vision Learners 2607.09024	62.1	47.0	54.5	55.5	65.9
Weak-to-Strong Generalization via Direct On-Policy Distillation 2607.05394	43.8	34.5	49.6	52.8	∞
Function-Aware Fill-in-the-Middle as Mid-Training for Coding Agent Foundation Models 2607.12463	53.8	63.4	55.9	108.9	67.5
Harness Handbook: Making Evolving Agent Harnesses Readable, Navigable, and Editable 2607.13285	∞	131.2	67.4	114.1	∞
LongStraw: Long-Context RL Beyond 2M Tokens under a Fixed GPU Budget 2607.14952	44.9	54.1	57.6	60.1	65.3
RAGU: A Multi-Step GraphRAG Engine with a Compact Domain-Adapted LLM 2607.11683	44.3	34.0	101.0	122.6	29.0
RynnBrain 1.1: Towards More Capable and Generalizable Embodied Foundation Model 2607.17977	50.7	37.1	96.9	89.0	43.2
ABot-World-0: Infinite Interactive World Rollout on a Single Desktop GPU 2607.19191	42.2	38.7	100.5	66.7	59.5
SLAI T-Rex: Full-Parameter Post-training of the DeepSeek-V4 Family on Ascend SuperPOD 2607.20145	∞	92.6	∞	∞	103.7
AREX: Towards a Recursively Self-Improving Agent for Deep Research 2607.21461	59.2	59.3	47.4	102.2	53.9

Continued on next page

Table 4 (continued). Directional compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
DataPrep-Bench: Benchmarking LLMs as Training Data Preparators 2607.20465	68.0	31.3	50.2	40.9	43.6
Kimi K3: Open Frontier Intelligence 2607.24653	∞	∞	∞	∞	∞
HiFi-UMI: Learning Deployable Manipulation Policies from High-Fidelity UMI Data Alone 2607.25895	22.8	25.8	26.2	107.3	45.9
TurboVLA: Real-Time Vision-Language-Action Model at 32 Hz on an RTX 4090 with ¡1 GB VRAM 2607.27205	36.6	43.7	38.4	52.5	41.2
Qwen-UI-Agent Technical Report: Toward Next-Generation Real-World Centric Foundation GUI Agents 2607.28227	57.0	29.2	62.9	105.8	56.8
Mental World Modeling 2607.27201	59.0	70.0	46.9	51.9	50.0
LongHorizon-Harness: Advancing Long-Horizon Agents for Real-World Tasks 2608.01964	52.2	44.7	42.2	100.7	51.4
MerchantBench: Benchmarking LLM Agents for Long-Term Coherence in E-Commerce Operations 2607.28956	34.5	36.6	45.4	62.2	30.9
Recursive Synthesis for Long-Horizon Terminal Tasks 2608.05466	45.0	35.5	57.4	109.3	68.0
AgentOPSD: Recursive Self-Distillation for Agentic Reinforcement Learning 2608.05987	28.5	22.7	33.3	44.2	45.2
SimWAM: A Simple World Action Model for End-to-End Autonomous Driving 2608.07468	41.1	50.1	96.7	125.1	64.1
BDH-CQ: In-Context Learning with Recurrent Latent Reasoning 2608.09888	48.1	47.9	35.4	46.2	59.4
ComBodied Agents: a New Paradigm of Human-Centric Agentic AI 2608.10915	47.8	41.8	95.4	54.2	∞
Spark-to-Paper: End-to-End Research Paper Generation as a Composable Skill 2608.11924	64.5	86.1	40.8	90.7	∞
Alaya-EVOKE: From Linear-Scaling Supervision to Endless World 2608.13546	44.3	46.9	69.1	68.7	73.9
StateM: Reaching 95.3% Raw Accuracy, or a $15 Frontier Run, on Terminal-Bench 2.1 via Harness Scaling 2608.15089	41.2	45.2	39.2	55.9	61.7
Demystifying Agent Skills: Why They Work-Until They Don't 2608.14036	30.6	31.8	31.4	59.5	31.4
SemComp-Bench: Benchmarking Semantic Task Completion in Video Generation 2608.17426	54.2	30.9	51.5	65.3	70.1
EnvHarness: Awakening Static Worlds for Agent Learning 2608.19880	42.9	34.0	59.5	61.6	43.5
Graph Engineering in the Era of LLM Agents: From Individual Intelligence to System Intelligence 2608.21156	54.1	82.2	79.0	106.2	∞
Apodex 1.1: Scaling Agentic Intelligence for Complex Work 2608.23283	53.8	47.0	93.8	67.7	71.5

Continued on next page

Table 4 (continued). Directional compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Annotations as Rollouts: Efficient and Scalable Reinforcement Learning for Video MLLMs 2608.20492	79.0	158.7	77.7	111.3	141.2
VBVR-Pro: A Scalable and Verifiable Suite for Native Visual Reasoning 2608.26105	39.3	71.6	103.6	71.1	51.9
Agentic Game Development as a Verifiable Trajectory Data Engine for Scaling World Models 2608.25518	41.6	60.5	132.9	89.1	45.8
LoopArena: Benchmarking Models as Runtime Controllers for Loop Engineering 2608.28281	88.2	73.9	88.0	57.5	77.7
Does On-Policy Distillation Really Distill? From Noisy Teacher to Self-Improvement 2608.31046	78.1	143.9	80.7	78.3	∞
StudentSim: Training LLM-based Student Simulators 2609.01591	87.7	70.5	96.8	125.0	106.9
Repo-To-Skill: Distilling GitHub Repositories Into AI4AI Skills 2609.02749	39.8	47.4	34.1	51.3	43.6
Compile by Training: Turning Natural-Language Specifications into Local Neural Functions 2609.04199	70.7	138.3	38.2	64.8	59.9

Table 5: Essence compression cost by paper and generator on the 87-paper test set. Costs are in bits; ∞ denotes that no validated idea passing the Essence judge was obtained within the experiment budget. Essence costs include the Directional stage.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Heterogeneous Scientific Foundation Model Collaboration 2604.27351	69.3	77.8	97.5	57.9	∞
UniVidX: A Unified Multimodal Framework for Versatile Video Generation via Diffusion Priors 2605.00658	59.1	60.1	58.4	68.2	147.5
MolmoAct2: Action Reasoning Models for Real-world Deployment 2605.02881	∞	59.6	72.5	82.4	77.3
ARIS: Autonomous Research via Adversarial Multi-Agent Collaboration 2605.03042	∞	46.4	43.4	48.1	94.1
Stream-R1: Reliability-Perplexity Aware Reward Distillation for Streaming Video Generation 2605.03849	91.1	68.3	68.6	93.5	119.9
Beyond Semantic Similarity: Rethinking Retrieval for Agentic Search via Direct Corpus Interaction 2605.05242	40.3	49.3	95.7	49.1	67.7
Mean Mode Screaming: Mean–Variance Split Residuals for 1000-Layer Diffusion Transformers 2605.06169	94.1	38.0	53.8	153.3	85.4
SenseNova-U1: Unifying Multimodal Understanding and Generation with NEO-unify Architecture 2605.12500	34.9	28.9	42.7	43.5	40.6
MinT: Managed Infrastructure for Training and Serving Millions of LLMs 2605.13779	71.3	40.8	55.6	56.2	29.2
Achieving Gold-Medal-Level Olympiad Reasoning via Simple and Unified Scaling 2605.13301	∞	148.7	∞	∞	∞
Code as Agent Harness 2605.18747	74.9	127.2	122.5	42.5	124.7
Anti-Self-Distillation for Reasoning RL via Pointwise Mutual Information 2605.11609	87.9	148.5	∞	∞	∞
HRM-Text: Efficient Pretraining Beyond Scaling 2605.20613	∞	113.6	∞	96.4	∞
DelTA: Discriminative Token Credit Assignment for Reinforcement Learning from Verifiable Rewards 2605.21467	61.6	44.4	57.2	94.0	84.9
SkillOpt: Executive Strategy for Self-Evolving Agent Skills 2605.23904	36.5	37.4	76.3	144.0	91.0
DVAO: Dynamic Variance-adaptive Advantage Optimization for Multi-reward Reinforcement Learning 2605.25604	46.5	57.4	56.5	55.3	49.6
LocateAnything: Fast and High-Quality Vision-Language Grounding with Parallel Box Decoding 2605.27365	49.8	102.7	37.1	116.6	67.8
Gamma-World: Generative Multi-Agent World Modeling Beyond Two Players 2605.28816	118.6	∞	69.0	105.5	∞
AgentDoG 1.5: A Lightweight and Scalable Alignment Framework for AI Agent Safety and Security 2605.29801	99.7	86.1	78.6	107.5	∞

Continued on next page

Table 5 (continued). Essence compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
COLLEAGUE.SKILL: Automated AI Skill Generation via Expert Knowledge Distillation 2605.31264	84.7	90.3	99.1	155.5	∞
Crafter: A Multi-Agent Harness for Editable Scientific Figure Generation from Diverse Inputs 2605.30611	53.9	125.3	105.5	87.8	∞
OCC-RAG: Optimal Cognitive Core for Faithful Question Answering 2606.00683	72.8	81.4	79.8	125.3	100.8
Cosmos 3: Omnimodal World Models for Physical AI 2606.02800	63.2	37.6	48.7	69.0	∞
Code2LoRA: Hypernetwork-Generated Adapters for Code Language Models under Software Evolution 2606.06492	64.2	54.1	39.0	88.3	∞
Imaginative Perception Tokens Enhance Spatial Reasoning in Multimodal Language Models 2606.03988	40.1	45.9	47.2	65.7	51.8
Agents' Last Exam 2606.05405	41.2	43.9	58.5	31.6	45.7
ABot-Earth 0.5: Generative 3D Earth Model 2606.09967	57.3	46.2	55.1	50.3	58.6
Embodied-R1.5: Evolving Physical Intelligence via Embodied Foundation Models 2606.11324	65.8	∞	∞	117.9	∞
From Chatbot to Digital Colleague: The Paradigm Shift Toward Persistent Autonomous AI 2606.14502	60.7	90.5	59.5	100.4	64.1
JoyAI-VL-Interaction: Real-Time Vision-Language Interaction Intelligence 2606.14777	∞	36.2	42.2	114.4	36.5
Looped World Models 2606.18208	42.0	48.6	36.6	94.9	61.6
MolmoMotion: Forecasting Point Trajectories in 3D with Language Instruction 2606.18558	67.2	∞	74.4	∞	∞
Moebius: 0.2B Lightweight Image Inpainting Framework with 10B-Level Performance 2606.19195	∞	104.6	108.8	105.5	∞
MemSlides: A Hierarchical Memory Driven Agent Framework for Personalized Slide Generation with Multi-turn Local Revision 2606.17162	83.1	80.6	83.8	95.8	146.6
PlanBench-XL: Evaluating Long-Horizon Planning of LLM Tool-Use Agents in Large-Scale Tool Ecosystems 2606.22388	63.4	82.9	65.8	81.9	64.9
Qwen-AgentWorld: Language World Models for General Agents 2606.24597	45.8	38.2	55.1	47.8	57.9
Are We Ready For An Agent-Native Memory System? 2606.24775	73.0	47.5	88.8	61.7	50.6
DanceOPD: On-Policy Generative Field Distillation 2606.27377	81.4	∞	60.9	43.9	73.4
Formalizing Latent Thoughts: Four Axioms of Thought Representation in LLMs 2606.27378	∞	∞	173.2	∞	140.8
Agentic Abstention: Do Agents Know When to Stop Instead of Act? 2606.28733	54.7	∞	44.8	43.1	47.0

Continued on next page

Table 5 (continued). Essence compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Orca: The World is in Your Mind 2606.30534	46.6	∞	32.1	81.5	34.3
PerceptionRubrics: Calibrating Multimodal Evaluation to Human Perception 2606.28322	45.8	46.0	50.9	223.2	100.2
Program-as-Weights: A Programming Paradigm for Fuzzy Functions 2607.02512	55.8	70.4	43.8	68.1	∞
The Mirage of Optimizing Training Policies: Monotonic Inference Policies as the Real Objective for LLM Reinforcement Learning 2606.29526	80.7	78.1	184.6	∞	142.4
OmniOpt: Taxonomy, Geometry, and Benchmarking of Modern Optimizers 2607.04033	65.5	36.1	64.5	53.8	43.4
RynnWorld-4D: 4D Embodied World Models for Robotic Manipulation 2607.06559	90.3	43.1	65.1	87.0	81.9
Accurate, Interdisciplinary and Transparent Structure-property Understanding with Deep Native Structural Reasoning 2607.07708	∞	∞	89.0	∞	66.8
Vidu S1: A Real-Time Interactive Video Generation Model 2607.03118	75.8	48.0	67.0	58.1	132.1
Video Generation Models are General-Purpose Vision Learners 2607.09024	82.2	78.4	56.1	66.3	∞
Weak-to-Strong Generalization via Direct On-Policy Distillation 2607.05394	43.8	35.3	50.9	54.2	∞
Function-Aware Fill-in-the-Middle as Mid-Training for Coding Agent Foundation Models 2607.12463	68.8	67.4	76.4	113.8	87.3
Harness Handbook: Making Evolving Agent Harnesses Readable, Navigable, and Editable 2607.13285	∞	134.2	88.8	139.5	∞
LongStraw: Long-Context RL Beyond 2M Tokens under a Fixed GPU Budget 2607.14952	45.1	54.4	63.0	60.5	65.9
RAGU: A Multi-Step GraphRAG Engine with a Compact Domain-Adapted LLM 2607.11683	63.5	57.7	112.2	122.6	57.4
RynnBrain 1.1: Towards More Capable and Generalizable Embodied Foundation Model 2607.17977	86.2	∞	147.8	∞	101.9
ABot-World-0: Infinite Interactive World Rollout on a Single Desktop GPU 2607.19191	42.7	59.1	102.4	68.2	75.7
SLAI T-Rex: Full-Parameter Post-training of the DeepSeek-V4 Family on Ascend SuperPOD 2607.20145	∞	∞	∞	∞	∞
AREX: Towards a Recursively Self-Improving Agent for Deep Research 2607.21461	∞	103.2	∞	160.4	∞
DataPrep-Bench: Benchmarking LLMs as Training Data Preparators 2607.20465	69.7	31.3	53.4	41.6	43.6
Kimi K3: Open Frontier Intelligence 2607.24653	∞	∞	∞	∞	∞

Continued on next page

Table 5 (continued). Essence compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
HiFi-UMI: Learning Deployable Manipulation Policies from High-Fidelity UMI Data Alone 2607.25895	68.6	37.8	28.6	118.9	48.7
TurboVLA: Real-Time Vision-Language-Action Model at 32 Hz on an RTX 4090 with ¡1 GB VRAM 2607.27205	36.6	44.3	38.4	51.7	74.3
Qwen-UI-Agent Technical Report: Toward Next-Generation Real-World Centric Foundation GUI Agents 2607.28227	∞	43.3	∞	110.8	∞
Mental World Modeling 2607.27201	∞	70.6	∞	74.1	∞
LongHorizon-Harness: Advancing Long-Horizon Agents for Real-World Tasks 2608.01964	67.8	46.7	43.1	103.4	53.6
MerchantBench: Benchmarking LLM Agents for Long-Term Coherence in E-Commerce Operations 2607.28956	46.7	46.3	47.9	64.6	50.3
Recursive Synthesis for Long-Horizon Terminal Tasks 2608.05466	48.4	37.4	57.4	109.3	70.7
AgentOPSD: Recursive Self-Distillation for Agentic Reinforcement Learning 2608.05987	∞	∞	68.3	97.8	∞
SimWAM: A Simple World Action Model for End-to-End Autonomous Driving 2608.07468	∞	81.8	97.4	∞	119.0
BDH-CQ: In-Context Learning with Recurrent Latent Reasoning 2608.09888	51.0	49.0	34.8	45.6	60.8
ComBodied Agents: a New Paradigm of Human-Centric Agentic AI 2608.10915	48.0	42.6	127.4	55.0	∞
Spark-to-Paper: End-to-End Research Paper Generation as a Composable Skill 2608.11924	66.0	111.6	63.1	92.4	∞
Alaya-EVOKE: From Linear-Scaling Supervision to Endless World 2608.13546	66.7	63.6	69.4	87.7	∞
StateM: Reaching 95.3% Raw Accuracy, or a $15 Frontier Run, on Terminal-Bench 2.1 via Harness Scaling 2608.15089	42.3	47.4	40.5	56.5	89.0
Demystifying Agent Skills: Why They Work-Until They Don't 2608.14036	54.8	104.4	92.8	104.5	88.3
SemComp-Bench: Benchmarking Semantic Task Completion in Video Generation 2608.17426	74.8	30.9	96.5	91.5	72.8
EnvHarness: Awakening Static Worlds for Agent Learning 2608.19880	43.1	60.7	87.1	55.5	∞
Graph Engineering in the Era of LLM Agents: From Individual Intelligence to System Intelligence 2608.21156	56.6	82.8	79.0	106.2	∞
Apodex 1.1: Scaling Agentic Intelligence for Complex Work 2608.23283	66.1	48.0	∞	∞	∞
Annotations as Rollouts: Efficient and Scalable Reinforcement Learning for Video MLLMs 2608.20492	102.1	169.0	77.7	127.0	∞
VBVR-Pro: A Scalable and Verifiable Suite for Native Visual Reasoning 2608.26105	59.9	84.0	103.7	71.5	52.2

Continued on next page

Table 5 (continued). Essence compression cost in bits.

Paper	Fable 5.1	Opus 5 high	GPT-6 Astra	GPT-5.6 Sol	GLM-5.3
Agentic Game Development as a Verifiable Trajectory Data Engine for Scaling World Models 2608.25518	52.2	∞	135.2	110.5	130.7
LoopArena: Benchmarking Models as Runtime Controllers for Loop Engineering 2608.28281	97.5	101.7	98.4	116.3	91.9
Does On-Policy Distillation Really Distill? From Noisy Teacher to Self-Improvement 2608.31046	79.8	147.0	108.8	89.4	∞
StudentSim: Training LLM-based Student Simulators 2609.01591	89.3	99.2	97.7	139.8	107.9
Repo-To-Skill: Distilling GitHub Repositories Into AI4AI Skills 2609.02749	40.7	50.1	35.3	52.8	43.6
Compile by Training: Turning Natural-Language Specifications into Local Neural Functions 2609.04199	70.6	138.9	38.2	65.2	59.9

E SkillNet Case Study: generator Proposals and Gold

The directional and essence proposals below are reproduced from sequences 55 and 73 of the successful Fable 5 trajectory for SkillNet (arXiv:2603.04448). The gold text is reproduced from the target pack's `summary.setting_and_object` field. We preserve the original content while adding field labels and list structure for readability.

Directional pass: generator proposal

Category

A system or software artifact.

Core problem

Language agents cannot accumulate and reuse experience across tasks and across agents in a portable way — existing experience/skill memories are bespoke and siloed within individual agents or frameworks, so there is no general-purpose infrastructure for the community of agents to collectively store, manage, and reuse learned skills at trustworthy quality.

Main object

A skill library infrastructure — a general-purpose memory infrastructure / operating-system-like layer (a standardized API, protocol, or middleware for agent memory) whose primary function is an experience/skill library that stores past task trajectories or learned procedures for reuse on new tasks, and whose defining architectural form is a centralized shared hub/repository where many agents or users contribute and reuse skills collectively (hub/marketplace model), governed by LLM-as-judge curation — a quality-control, validation, and curation mechanism in which an LLM acts as judge over contributed skills, evaluating their correctness, usefulness, generality, and redundancy to decide which contributed skills enter or remain in the shared hub.

System design

The system defines a standard skill/experience representation (e.g., distilled procedures/workflows with metadata, derived from completed task trajectories) and a unified set of hub interfaces:

- Any language agent or user can contribute skills (uploading experience distilled from successful trajectories).
- The LLM-judge curation pipeline vets each contribution before admission and continually maintains the library (filtering low-quality or incorrect skills, deduplicating/merging overlapping ones, ranking and pruning).
- Any agent can retrieve the most relevant curated skills for a new task (e.g., via embedding-based similarity or task matching) into its context to condition its planning and actions.

Intended outcome

By decoupling the skill library from any specific agent and exposing it as a centralized, standardized, LLM-curated shared hub — analogous to a package registry or model hub, but for agent skills with automated quality

gatekeeping — the system enables collective, continual, trustworthy skill accumulation and cross-agent/cross-user skill reuse without any model weight updates.

Essence pass: generator proposal

Category

A system or software artifact.

Core problem

Language agents cannot accumulate and reuse experience across tasks and across agents in a portable way — existing experience/skill memories are bespoke and siloed within individual agents or frameworks, so there is no general-purpose infrastructure for the community of agents to collectively store, organize, and reuse learned skills at trustworthy quality.

Main object

A skill library infrastructure — a general-purpose memory infrastructure / operating-system-like layer (a standardized API, protocol, or middleware for agent memory) whose primary function is an experience/skill library that stores past task trajectories or learned procedures for reuse on new tasks, and whose defining architectural form is a centralized shared hub/repository where many agents or users contribute and reuse skills collectively (hub/marketplace model), governed by LLM-as-judge curation — a quality-control, validation, and curation mechanism in which an LLM acts as judge over contributed skills, evaluating their correctness, usefulness, generality, and redundancy to decide which contributed skills enter or remain in the shared hub — and organized by a skill ontology: an organizational structure imposed on the hub's contents, i.e., a hierarchical taxonomy of skills with relations among them (specialization/generalization, dependencies, and composition of smaller skills into larger skills) that structures how skills are stored, related, abstracted, and navigated.

System design

The system defines a standard skill/experience representation (e.g., distilled procedures/workflows with metadata, derived from completed task trajectories) and a unified set of hub interfaces:

- Any language agent or user can contribute skills (uploading experience distilled from successful trajectories).
- The LLM-judge curation pipeline vets each contribution before admission and continually maintains the library (filtering low-quality or incorrect skills, deduplicating/merging overlapping ones, ranking and pruning).
- Admitted skills are placed into the skill ontology — classified under the appropriate taxonomy nodes, linked by dependency/composition relations, with related skills abstracted or composed into more general or higher-level skills as the library grows.
- Any agent can retrieve the most relevant curated skills for a new task (via ontology-guided navigation and/or embedding-based similarity) into its context to condition its planning and actions.

Intended outcome

By decoupling the skill library from any specific agent and exposing it as a centralized, standardized, LLM-curated, ontology-organized shared hub — analogous to a package registry or model hub, but for agent skills with automated quality gatekeeping and a structured taxonomy — the system enables collective, continual, trustworthy, and well-organized skill accumulation and cross-agent/cross-user skill reuse without any model weight updates.

Gold setting and main object

Category

Infrastructure / Resource (open skill repository, ontology, evaluation protocol, and toolkit).

Problem & motivation

- Agents lack a unified mechanism to consolidate episodic experience into durable, reusable skills, so they repeatedly "reinvent the wheel."
- Existing skill repositories/marketplaces rely on manual curation and only indirect (downstream-task) quality signals, leaving skills fragmented, redundant, and unvalidated.

Main object & crux

SkillNet: an open full-lifecycle skill infrastructure that:

- Auto-creates skills from heterogeneous sources (execution trajectories, GitHub repos, documents, prompts) via LLM pipelines.
- Evaluates each skill intrinsically along Safety, Completeness, Executability, Maintainability, Cost-awareness.
- Connects skills into a typed relation graph (similar_to, belong_to, compose_with, depend_on).

Representation
Skills are treated as a unified knowledge representation bridging natural-language procedural guidance and machine-executable logic, organized by a three-layer Skill Ontology (taxonomy, relation graph, skill packages).

Release
Released as a curated repository (200k+ candidate, 150k+ curated skills), website/API, and the skillnet-ai Python library + CLI.

F DIRECTIONAL-TO-ESSENCE REFINEMENT EXAMPLES

The boxes below reproduce complete submitted Opus 5 high proposals for two test papers, with field headings, paragraph breaks, and list formatting added for readability. Role terminology is standardized to *guide*. All other original prose is retained, and mathematical notation is typeset in LaTeX. Each example pairs a candidate that passes the Directional judge with the final proposal, which passes all three Essence evaluations. The Directional proposal is black; the Essence proposal is purple. Costs are cumulative; the reported increase is net of refunded costs. All values are rounded independently.

DelTA: Directional to Essence (arXiv:2605.21467)

Compression cost

Directional: 21.64 bits. Essence: 44.38 bits. Net increase: 22.75 bits. New retained Essence costs: 25.41 bits.

Directional proposal

Category:

a new algorithm or method.

Core problem:

in RL post-training of large language models with only a single outcome reward at the end of a long generated sequence, current estimators give no credit assignment inside the sequence -- group-relative, critic-free objectives (GRPO-style) compute one sequence-level advantage and broadcast it identically to every token, so all steps of a correct trace are reinforced and all steps of a failed trace are punished, while PPO-style learned value networks that do produce token-level advantages are expensive and poorly calibrated on partial generations.

Main object (the method):

a critic-free step-level advantage estimator that obtains Monte-Carlo values for intermediate prefixes by branching extra short continuations from selected points inside sampled trajectories, estimating $V(\text{prefix})$ as the average terminal reward of the continuations that share that prefix, and defining the advantage of each step as the increment in this estimated value across the step (a Monte-Carlo temporal-difference on prefix values).

This turns one terminal scalar into a within-sequence advantage profile that localizes which reasoning steps actually raised or lowered the probability of success, while the group of siblings still supplies the variance-reducing baseline; the resulting per-step advantages are plugged into the standard clipped policy-gradient update, and the branching budget is concentrated where the estimate matters, so the method keeps GRPO's low cost and low variance but recovers PPO-like fine-grained credit without training any value model. (Confirmed: setting and contribution type; the branching-based prefix-value mechanism is my inference.)

Essence proposal

Category:

a new algorithm or method.

Core problem:

in RL post-training of large language models (token/step-level credit within generated sequences), the reward is a single scalar at the end of a long generated trace, so critic-free group-relative policy gradients broadcast one identical advantage to every token of a response -- reinforcing filler and erroneous tokens exactly like the decisive ones, and letting conflicting token gradients cancel or fight -- while learned value models that would supply token-level advantages are expensive and miscalibrated on partial sequences.

Main object:

'cross-rollout gradient contrast', a critic-free token-level credit estimator whose defining computation is gradient- or attention-based attribution of the outcome back onto tokens realized in PARAMETER space and compared contrastively across the sampled responses of the same prompt using their attribution maps.

For each sampled rollout, the attribution map is the collection of per-token parameter-space gradients $g_{i,t} = \nabla_\theta \log \pi_\theta(y_{i,t} \mid \text{prompt}, y_{i,<t})$ -- the weight-space direction in which reinforcing that single token would move the policy.

The rollouts of a prompt are split by outcome into a positive set (correct / high-reward) and a negative set (incorrect / low-reward), and their gradients are aggregated into two group directions G^+ and G^-.

Each token's credit is then the inner-product/cosine alignment of its own gradient with these aggregated gradients of positive versus negative rollouts -- essentially $\langle g_{i,t}, G^+ \rangle - \langle g_{i,t}, G^- \rangle$ (equivalently the cosine alignment with the contrast direction $G^+ - G^-$) -- an influence-function/gradient-agreement score computed within the prompt's own group.

Because gradients from different rollouts share one vector space, this contrast needs no positional or semantic alignment of tokens across responses; components common to successful and failed rollouts (prompt-generic, formatting, filler) cancel in the contrast and receive near-zero credit, acting as a token-level baseline/control variate, while tokens whose update direction genuinely drives success versus failure retain signed credit -- so an error inside a correct trace is not rewarded and a sound step inside a failed trace is not punished.
These alignment-derived per-token advantages replace the uniformly broadcast sequence-level group-relative advantage inside the standard clipped policy-gradient update, yielding dense intra-sequence credit assignment and variance reduction with no extra rollouts, value network, or process reward model.

ABot-World-0: Directional to Essence (arXiv:2607.19191)

Compression cost

Directional: 38.71 bits. Essence: 59.06 bits. Net increase: 20.34 bits. New retained Essence costs: 23.52 bits.

Directional proposal

Category:

a system or software artifact (deep learning application in computer vision).

Core problem:

generative video models can render plausible scenes but cannot be used as an actual environment -- multi-step diffusion is orders of magnitude too slow for interactive control, and autoregressive rollouts drift and forget, so a user cannot walk through, look around, and come back to a place in a generated world.

Main object:

an end-to-end interactive 'world model' system that turns a text prompt or a single image into a navigable virtual world and generates it frame-by-frame in real time in response to a live stream of user actions (movement, camera/view control).

Key design principles:

(i) causal, streaming, action-conditioned generation -- the model is made autoregressive over frames/chunks with few-step (distilled) sampling and cached context so each new frame is produced at interactive latency instead of denoising a whole clip;

(ii) a persistent world state/memory: previously generated content (with its camera poses) is retained and re-injected as conditioning so revisited regions remain consistent and the world does not dissolve during long, in-principle unbounded exploration;

(iii) an engineered serving pipeline (GPU-resident rolling context, low-latency input-to-frame loop) that makes the whole thing a usable interactive application rather than an offline demo.

Confirmed by the guide:

field, systems/artifact category, and that the artifact's identity is a 'world model' whose functional role defines the contribution; the streaming + memory design principles are my inference.

Essence proposal

Category:

a system or software artifact (computer-vision application).

Core problem:

generative video models cannot serve as an environment a person can actually play in -- driven autoregressively from a user's live commands they must run causally and in real time, and because they were trained on ground-truth history they drift, blur and collapse once conditioned on their own outputs, so interactive sessions cannot last.

Main object:

an end-to-end interactive 'world model' system whose functional role is to turn a prompt/initial image into a navigable virtual world that a user explores in real time under 'keyboard control' -- discrete WASD-style keypresses (plus view/mouse-look) are the action interface and each keypress conditions the very next generated frame chunk, produced causally with few-step sampling and a cached rolling context for interactive latency.
Its key design principle -- the essence -- is stabilizing unbounded autoregressive rollout by training the model on its own generated context rather than on ground-truth history: the causal action-conditioned generator is self-rolled out during training (with a rolling KV/context cache, and the loss -- distribution-matching/adversarial or denoising with noise-augmented self-generated context -- applied to frames produced from that self-generated history), eliminating the train/inference exposure-bias gap so error does not accumulate and the world stays sharp and coherent over arbitrarily long, in principle unbounded, keyboard-controlled exploration.

G Related Work

A growing line of work uses language models to propose scientific hypotheses and research directions through scaffolding (Wang et al., 2023; Baek et al., 2024; Zhao et al., 2025; Liu et al., 2026; Yang et al., 2025a;b; Zhao et al., 2026; Wang et al., 2025; Su et al., 2025; Gottweis et al., 2025). SciMON grounds generation in retrieved literature and iteratively revises ideas to increase novelty (Wang et al., 2023), while ResearchAgent augments a core paper with an academic graph and uses reviewing agents to refine proposed problems, methods, and experiments (Baek et al., 2024). Among these works, the study by Zhao et al. (2026) is the closest to our work as it explores using guidance based on a target paper to create training trajectories to train idea generators.

Several benchmarks evaluate idea-generation capabilities using LLMs as judges of idea quality. A pioneering study with more than 100 NLP researchers in 2024 finds that LLM-generated proposals can receive higher novelty ratings than human proposals, while also identifying limitations in diversity and self-evaluation (Si et al., 2024). Later follow-up work (Si et al., 2025) reveals the ideation–execution gap for the LLM-proposed ideas, questioning the effectiveness of purely LLM-generated ideas. These findings are further supported by work benchmarking research judgment, showing that even current frontier LLM judges can disagree sharply with domain experts on scientific novelty and feasibility (Sinhahajari et al., 2026; Zheng et al., 2023; Wataoka et al., 2024; Baig et al., 2026; Si et al., 2026).

Another line of work evaluates whether models can forecast research that appears later by revealing partial information to the idea-generating models (Krenn et al., 2022; Li et al., 2026a; Jiang, 2026; Chen et al., 2026). IdeaBench provides preferences and measures idea overlap, novelty, and feasibility compared to real papers (Guo et al., 2024). LiveIdeaBench supplies only keywords and tests divergent thinking across scientific domains (Ruan et al., 2024). GIANTS asks a model to predict a target insight from two parent papers (He-Yueya et al., 2026), and future-aligned proposal generation scores proposals by their similarity to subsequently published work (Wang et al., 2026b). AI Idea Bench emphasizes knowledge leakage and grounded feasibility evaluation (Qiu et al., 2025). Our work shares the same intuition and quantifies a target idea's compression cost, the additional information in bits needed to guide a model toward the idea's essence. This approach quantifies the value of partial information such as keywords or cited works, which are non-trivial to obtain (Yang & Bing, 2026).

Our use of priced guidance can be viewed as a special case of online algorithms with advice (Boyar et al., 2017) and can be extended to perform rare output estimation (Wu & Hilton, 2024). We view our idea-generation evaluation method as an orthogonal component to automatic science execution (Lu et al., 2024; Chen et al., 2024).

Information Theory and Rare-Event Estimation. We measure compression at the level of ideas, allowing different textual realizations. Semantic uncertainty makes a related distinction for natural-language answers by grouping generations into meaning-equivalence classes and marginalizing probability over different surface forms (Kuhn et al., 2023). Our use of priced guidance is also related to online algorithms with advice, where an all-knowing guide communicates information about the future and complexity is measured by the number of advice bits (Boyar et al., 2017). Rare-event simulation methods, including importance sampling and multilevel splitting, bias sampling toward rare outcomes and correct for the resulting change of measure (Juneja & Shahabuddin, 2006). Recent work applies rare-event estimation directly to low-probability language-model outputs (Wu & Hilton, 2024). In our protocol, selecting an answer assigned probability p costs its self-information $-\log_2 p$. With the judge pass rate included in compression cost K, the expected guided score 2^{-K} lower-bounds unguided success.

LLMs for Automated Scientific Research. Beyond idea generation, recent systems attempt to automate larger portions of the research process. The AI Scientist generates ideas, writes code, runs experiments, and produces papers (Lu et al., 2024). The AI Scientist-v2 adds agentic tree search and reports workshop-level automated research (Yamada et al., 2025). A complementary line of benchmarks evaluates downstream research capability. MLAgentBench measures autonomous machine-learning experimentation (Huang et al., 2023). ScienceAgentBench and DiscoveryBench evaluate data-driven scientific discovery (Chen et al., 2024; Majumder et al., 2024). PaperBench tests whether agents can replicate published AI research from paper descriptions (Starace et al., 2025) and RE-Bench compares agents with human experts on time-bounded AI research-engineering tasks (Wijk

et al., 2024). Our work complements these systems and benchmarks by evaluating idea-generation capability through the cost of compressing a target idea.

H RESEARCH TAXONOMY AND CONTRIBUTION CATEGORIES

The opening bids use a fixed research taxonomy of deep learning and a fixed list of contribution categories (Section 3). GPT-5.6 Sol generated both once, and every generator receives them for every target. The category bid offers the following 11 categories with equal probability.

1. a new algorithm or method
2. a theory or analysis result
3. a benchmark
4. a dataset
5. an architecture
6. a mechanism study (how something works)
7. a scaling law
8. an empirical phenomenon or observation study
9. an evaluation protocol or metric
10. a system or software artifact
11. an application of existing methods to a domain

The taxonomy has 3,035 nodes below its root, "deep learning," in five levels, with 17 top-level fields. Each taxonomy bid offers the children of the current node and a stop option with equal probability. The taxonomy file also places a child named "other" under every node; the scaffold never offers it, so it and its subtree are omitted. The list keeps the file's order: bold entries are the top-level fields, and indentation marks deeper levels.

agents
- Foundations and Theory
  - Reinforcement Learning Theory
  - Sequential Decision Theory and Control
  - Planning, Search, and Model-Based Reasoning
  - Exploration and Sample Complexity
  - Credit Assignment and Temporal Abstraction
  - Representation Learning and World Models
  - Imitation Learning and Inverse Reinforcement Learning
  - Generalization and Transfer in Agents
  - Multi-Agent Learning and Game Theory
  - Safety, Robustness, and Alignment
  - Agent Architectures and Inductive Biases
  - Interpretability and Mechanistic Analysis
  - Empirical Phenomena and Scaling Laws
  - Benchmarks and Evaluation
  - Surveys and Unifying Frameworks
- Reinforcement Learning
  - Theoretical Reinforcement Learning
  - Model-Free Reinforcement Learning
  - Model-Based Reinforcement Learning
  - Offline Reinforcement Learning
  - Exploration and Intrinsic Motivation
  - Hierarchical Reinforcement Learning
  - Multi-Agent Reinforcement Learning
  - Learning from Demonstrations and Human Feedback
  - Safe and Constrained Reinforcement Learning
  - Transfer Learning, Meta-Reinforcement Learning, and Generalization
  - Representation Learning for Reinforcement Learning
  - Benchmarking, Evaluation, and Reproducibility in Reinforcement Learning
  - Applied Reinforcement Learning
  - Survey and Review Literature on Reinforcement Learning
- Multi-Agent Learning
  - Cooperative Multi-Agent Learning
  - Competitive and Adversarial Multi-Agent Learning
  - Mixed-Motive and General-Sum Multi-Agent Learning
  - Game-Theoretic Multi-Agent Learning
  - Opponent Modeling and Recursive Reasoning
  - Communication and Emergent Communication
  - Coordination and Credit Assignment
  - Decentralized Control and Planning
  - Large-Population and Mean-Field Multi-Agent Learning
  - Transfer, Meta-Learning, and Generalization in Multi-Agent Systems
  - Safe, Robust, and Trustworthy Multi-Agent Learning
  - Theoretical Foundations of Multi-Agent Learning
  - Benchmarks, Simulators, and Evaluation for Multi-Agent Learning
  - Applications of Multi-Agent Learning
  - Surveys and Taxonomies of Multi-Agent Learning
- Planning and Model-Based Control
  - Dynamics Modeling and World Models
  - State Estimation and Belief Modeling
  - Planning and Search
  - Optimal Control and Model Predictive Control
  - Model-Based Reinforcement Learning
  - Hierarchical Planning and Temporal Abstraction
  - Task and Motion Planning
  - Safe, Robust, and Constrained Control
  - Adaptive Control and Online System Identification
  - Robotics and Autonomous Systems Applications
  - Benchmarks and Evaluation
  - Theory and Guarantees
- Hierarchical Reinforcement Learning
  - Options and Temporal Abstraction
  - Goal-Conditioned and Subgoal-Based HRL
  - Feudal Reinforcement Learning
  - Skill Discovery and Unsupervised HRL
  - State Abstraction and Hierarchical Representation Learning
  - Model-Based and Planning-Oriented HRL
  - Hierarchical Exploration and Sparse-Reward Learning
  - Hierarchical Policy Optimization and Credit Assignment
  - Transfer, Continual, and Compositional HRL
  - Imitation, Offline, and Language-Guided HRL
  - Multi-Agent Hierarchical Reinforcement Learning
  - Theory and Foundations of HRL
  - Benchmarks, Empirical Evaluation, and Applications of HRL
- Imitation Learning and Inverse Reinforcement Learning
  - Behavior Cloning
  - Interactive Imitation Learning
  - Inverse Reinforcement Learning
  - Apprenticeship Learning
  - Adversarial Imitation Learning
  - Imitation from Observation
  - Offline Imitation Learning
  - Hierarchical Imitation Learning
  - Meta-Imitation Learning and Generalization
  - Safe and Robust Imitation Learning

- Theory and Foundations of Imitation Learning and IRL
- Benchmarks, Datasets, and Evaluation
- Applications of Imitation Learning and IRL
- Surveys and Tutorials

Embodied AI and Robotics
- Foundations and Theory
- Embodied Perception
- Localization and Mapping
- Planning and Decision Making
- Robot Control and Reinforcement Learning
- Imitation Learning
- Manipulation and Grasping
- Navigation and Exploration
- Locomotion
- Language-Grounded Embodied Agents
- Human-Robot Interaction
- Multi-Robot Systems
- Sim-to-Real Transfer
- Safety and Verification
- Benchmarks and Simulation
- Robotic Applications and Systems

Language Agents
- Agent Architectures
- Planning, Reasoning, and Decision Making
- Tool Use and Environment Interaction
- Memory, Retrieval, and Knowledge Integration
- Learning and Optimization for Language Agents
- Multi-Agent Systems
- Alignment, Safety, and Robustness
- Evaluation and Benchmarks
- Foundations and Analysis
- Applications of Language Agents
- Surveys and Taxonomies

Human-Agent Interaction
- Learning from Demonstrations
- Learning from Human Feedback
- Interactive Reinforcement Learning and Online Guidance
- Natural Language and Multimodal Interaction
- Human-Agent Collaboration and Shared Autonomy
- User Modeling and Personalization
- Social and Affective Interaction
- Explainability, Transparency, and Trust
- Safety, Alignment, and Human Oversight
- Evaluation, Benchmarks, and Human Studies

Safety, Alignment, and Robustness
- Foundations of AI Safety and Alignment
- AI Alignment and Preference Learning
- Safe Reinforcement Learning
- Adversarial Machine Learning
  - Evasion Attacks and Adversarial Examples
  - Poisoning Attacks
  - Backdoor Attacks
  - Model Stealing and Privacy Attacks
  - Adversarial Defenses
  - Certified Robustness and Verification
  - Theory of Adversarial Robustness
  - Threat Modeling and Security Analysis
  - Robustness Evaluation and Benchmarking
  - Applied Adversarial Machine Learning
- Distributional Robustness and Out-of-Distribution Generalization
- Uncertainty Quantification and Calibration
- Formal Verification and Certified Robustness
- Interpretability, Oversight, and Monitoring
- Safety Evaluation, Red Teaming, and Benchmarks
- Safety-Critical Machine Learning and Autonomous Systems

Agent Architectures and Training
- Methodology
- Policy and Value Function Architectures
- Memory-Augmented and Recurrent Agents
- World Models and Model-Based Agents
- Hierarchical Reinforcement Learning and Temporal Abstraction
- Modular and Compositional Agent Design
- Planning-Integrated and Search-Based Agents
- Representation Learning and State Abstraction for Agents
- Policy Optimization, Credit Assignment, and Stabilization
- Self-Supervised, Auxiliary, and Multitask Training
- Imitation Learning, Learning from Demonstrations, and Offline Reinforcement Learning
- Exploration, Curriculum Learning, and Environment Design
- Human Feedback and Preference-Based Training
- Evaluation, Benchmarks, and Experimental Methodology
- Theory of Agent Architectures and Training
- Application-Specific Agent Design
- Surveys and Taxonomies of Agent Architectures

Evaluation and Benchmarks
- Benchmark Construction and Curation
- Evaluation Metrics and Protocols
- Statistical Evaluation and Reproducibility
- Capability and Task Performance Benchmarking
- Generalization and Transfer Evaluation
- Robustness and Adversarial Evaluation
- Safety and Reliability Evaluation
- Human-Centered Evaluation
- Efficiency and Resource-Aware Evaluation
- Offline and Counterfactual Evaluation
- Embodied and Real-World Evaluation
- Meta-Evaluation and Benchmark Auditing

Applications of Agents
- Embodied AI and Robotics
- Autonomous Driving and Navigation
- Games and Simulated Environments
- Web and Software Agents
- Conversational and Assistive Agents
- Recommender Systems and Personalization
- Healthcare and Biomedicine
- Scientific Discovery and Engineering Design
- Operations Research, Logistics, and Resource Management
- Finance and Economic Decision Making
- Education and Intelligent Tutoring Systems
- Cybersecurity
- Benchmarks and Evaluation for Agent Applications

Surveys and Taxonomies
- Foundational Taxonomies and Conceptual Frameworks
- Agent Architectures
- Reinforcement Learning and Sequential Decision-Making Agents
- Planning, Reasoning, and Memory
- Foundation-Model and Language Agents
- Tool-Using and Environment-Interacting Agents
- Multi-Agent Systems
- Embodied and Robotic Agents
- Human-Agent Interaction and Collaboration
- Evaluation, Benchmarks, and Datasets
- Safety, Alignment, Robustness, and Security
- Application-Oriented Agent Surveys

alignment

Objective Specification and Outer Alignment
- Reward Modeling and Preference Learning
- Inverse Reinforcement Learning and Inverse Reward Design
- Value Learning and Preference Elicitation
- Scalable Oversight
- Constitutional and Rule-Based Alignment
- Constrained and Multi-Objective Alignment
- Specification Gaming and Reward Hacking
- Objective Robustness and Goal Generalization
- Theoretical Foundations of Outer Alignment
- Evaluation and Benchmarking for Objective Specification
- Application-Specific Objective Specification
- Surveys and Methodology for Objective Specification

Inner Alignment and Goal Generalization
- Theoretical Foundations
- Mesa-Optimization
- Deceptive Alignment
- Goal Misgeneralization
- Mechanistic Interpretability
- Empirical Studies
- Benchmarks and Evaluation
- Alignment Interventions
- Surveys and Taxonomies

Preference Learning and Reward Modeling
- Preference Elicitation
- Preference Modeling
- Reward Modeling
- Preference Optimization
- Active and Online Preference Learning
- Inverse Reinforcement Learning and Reward Inference
- Theoretical Foundations
- Robustness, Uncertainty, and Calibration
- Safety, Bias, and Reward Hacking
- Evaluation and Benchmarks
- Multimodal and Domain-Specific Applications

Learning from Human Feedback
- Preference Learning and Reward Modeling
- Preference-based Policy Optimization
- Learning from Demonstrations and Imitation Learning
- Critique and Corrective Feedback
- Interactive and Active Feedback Elicitation
- Feedback Collection and Annotation Methodology
- Robustness, Uncertainty, and Bias in Human Feedback
- Personalization and Preference Aggregation
- Theory and Foundations of Learning from Human Feedback
- Evaluation and Benchmarks for Human Feedback Methods
- Applications of Learning from Human Feedback

Scalable Oversight
- Theoretical Foundations of Scalable Oversight

- Preference Learning and Reward Modeling
- Process Supervision
- Decomposition-Based Oversight
- Debate and Adversarial Oversight
- AI-Assisted Oversight
- Weak-to-Strong Generalization
- Eliciting Latent Knowledge
- Interpretability-Assisted Oversight
- Oversight Benchmarks and Empirical Evaluation

Rule-Based and Constraint-Based Alignment
- Constitutional and Policy-Based Alignment
- Constraint Elicitation and Specification
- Constrained Optimization for Alignment
- Constrained Reinforcement Learning and Safe Exploration
- Neuro-Symbolic and Logic-Based Alignment
- Decoding-Time Constraint Enforcement
- Runtime Monitoring and Safety Filters
- Formal Verification and Certification
- Theoretical Foundations of Constraint-Based Alignment
- Evaluation, Benchmarks, and Red Teaming
- Applications in Safety-Critical and Regulated Domains
- Surveys and Taxonomies of Rule-Based Alignment

Interpretability and Transparency for Alignment
- Mechanistic Interpretability
- Feature Attribution and Saliency Methods
- Concept-Based Interpretability
- Probing and Representation Analysis
- Example-Based Explanations and Data Attribution
- Causal and Counterfactual Explanations
- Explanation Generation and Rationalization
- Transparency and Documentation
- Human-Centered Interpretability
- Evaluation, Faithfulness, and Benchmarks
- Model Auditing
- Scalable Oversight
- Theoretical Foundations of Interpretability
- Surveys and Taxonomies

Corrigibility and Controllability
- Formal Corrigibility
- Safe Interruptibility
- Shutdown and Off-Switch Control
- Human Oversight and Override
- Objective Modification and Updateability
- Tampering Resistance
- Containment and Runtime Control
- Training Methods for Controllable Agents
- Evaluation and Benchmarks for Controllability

Truthfulness, Honesty, and Uncertainty
- Uncertainty Quantification and Calibration
- Epistemic Uncertainty and Out-of-Distribution Detection
- Selective Prediction and Deferral
- Factuality and Hallucination
- Verification and Fact-Checking
- Honesty, Deception, and Strategic Misrepresentation
- Uncertainty Communication and Confidence Elicitation
- Evaluation, Benchmarks, and Metrics
- Theoretical Foundations of Truthfulness and Uncertainty

Alignment Evaluation, Benchmarks, and Red Teaming
- Benchmark Construction and Dataset Curation
- Evaluation Methodology and Measurement
- Human and Automated Evaluation
- Adversarial Red Teaming
- Preference, Helpfulness, and Instruction-Following Evaluation
- Safety, Toxicity, Bias, and Misuse Evaluation
- Truthfulness, Honesty, and Calibration Evaluation
- Robustness and Distribution-Shift Evaluation
- Deception, Goal Misgeneralization, and Specification-Gaming Evaluation
- Agentic and Long-Horizon Alignment Evaluation
- Meta-Evaluation and Benchmark Validation

Multi-Agent and Social Alignment
- Cooperative Multi-Agent Learning
- Competitive and Mixed-Motive Learning
- Communication and Negotiation
- Social Choice and Preference Aggregation
- Mechanism Design and Incentive Alignment
- Norm Learning and Convention Formation
- Human-AI Collective Decision-Making
- Fairness and Social Welfare Optimization
- Deliberation, Debate, and Multi-Agent Oversight
- Game-Theoretic Foundations of Social Alignment
- Benchmarks and Evaluation for Social Alignment

Formal Alignment Theory
- Objective Specification and Reward Design
- Preference Learning and Inverse Reinforcement Learning
- Assistance Games and Human-AI Cooperation
- Corrigibility and Interruptibility
- Inner Alignment and Mesa-Optimization
- Decision Theory and Embedded Agency
- Delegation and Scalable Oversight
- Multi-Agent Alignment and Social Choice
- Formal Verification and Alignment Guarantees
- Alignment Evaluation and Benchmarking
- Impossibility Results and Alignment Limits

applications

Computer Vision
- Visual Recognition
- Object Detection
- Image Segmentation
- Pose Estimation and Human-Centered Vision
- Tracking and Motion Analysis
- Video Understanding
- 3D Vision and Geometric Vision
- Low-Level Vision and Image Restoration
- Image Generation and Editing
- Scene Understanding and Visual Reasoning
- Vision Representation Learning
- Vision-Language and Multimodal Learning
- Robustness, Fairness, and Interpretability in Vision
- Vision Datasets, Benchmarks, and Evaluation
- Applied Computer Vision

Natural Language Processing
- Language Modeling
- Parsing and Linguistic Structure
- Text Classification
- Information Extraction
- Question Answering and Reading Comprehension
- Machine Translation
- Summarization
- Dialogue Systems
- Natural Language Generation
- Knowledge-Intensive NLP
- Multilingual and Cross-Lingual NLP
- Evaluation, Benchmarks, and Resources
  - Automatic Evaluation Metrics
  - Human Evaluation
  - Meta-Evaluation and Benchmark Analysis
  - Diagnostic Evaluation and Challenge Sets
  - Benchmark Datasets
  - Corpora and Dataset Creation
  - Annotation Methodology
  - Lexical and Knowledge Resources
  - Shared Tasks and Evaluation Platforms
  - Data Documentation, Bias, and Ethics
- Interpretability and Analysis
- Robustness, Fairness, and Safety
- Efficient NLP

Speech and Audio Processing
- Automatic Speech Recognition
- Speech Translation and Cross-Lingual Speech Processing
- Spoken Language Understanding and Keyword Spotting
- Speech Synthesis and Voice Conversion
- Speech Enhancement, Dereverberation, and Source Separation
- Speaker and Language Recognition
- Paralinguistic Speech Processing
- Audio Event Detection and Sound Scene Analysis
- Music Information Retrieval and Music Audio Modeling
- General Audio Generation and Editing
- Audio-Visual Speech and Sound Processing
- Speech and Audio Representation Learning
- Datasets, Benchmarks, and Evaluation for Speech and Audio

Information Retrieval and Search
- Learning to Rank and Re-ranking
- Neural and Dense Retrieval
- Query Understanding and Reformulation
- Question Answering and Knowledge-Intensive Retrieval
- Conversational and Interactive Search
- Personalized and Context-Aware Search
- Multimodal Retrieval
- Cross-Lingual and Multilingual Retrieval
- Vertical and Domain-Specific Search
- IR Evaluation, Benchmarks, and User Behavior Modeling

Recommender Systems
- Collaborative Filtering
- Content-Based and Multimodal Recommendation
- Context-Aware Recommendation
- Sequential and Session-Based Recommendation
- Graph-Based Recommendation
- Knowledge-Aware Recommendation
- Social Recommendation
- Cross-Domain Recommendation
- Cold-Start and Long-Tail Recommendation
- Conversational Recommendation
- Bandit and Reinforcement Learning for Recommendation
- Explainable Recommendation
- Causal and Counterfactual Recommendation
- Fair and Responsible Recommendation

- Privacy and Security in Recommender Systems
- Evaluation and Benchmarking in Recommender Systems

Healthcare and Biomedicine
- Medical Image Analysis
- Digital Pathology and Computational Microscopy
- Biomedical Signal Processing and Physiological Time Series
- Clinical Natural Language Processing
- Electronic Health Records and Clinical Prediction
- Computational Genomics and Multi-omics
- Drug Discovery and Molecular Design
- Precision Medicine and Treatment Effect Modeling
- Public Health Informatics and Epidemiological Modeling
- Trustworthy and Privacy-Preserving Medical AI
- Clinical Evaluation, Benchmarking, and Translation

Robotics and Autonomous Systems
- Robot Perception and Scene Understanding
- Localization, Mapping, and State Estimation
- Navigation and Exploration
- Manipulation and Grasping
- Motion Planning and Trajectory Optimization
- Robot Learning and Control
- Human-Robot Interaction and Shared Autonomy
- Multi-Robot Systems and Swarm Robotics
- Autonomous Vehicles and Intelligent Transportation
- Simulation, Sim-to-Real Transfer, and Robotics Benchmarks
- Safe, Robust, and Trustworthy Autonomous Systems

Time-Series Analysis and Forecasting
- Forecasting
- Time-Series Classification and Regression
- Anomaly Detection and Change-Point Detection
- Segmentation, Clustering, and Pattern Discovery
- Imputation, Denoising, and Reconstruction
- Representation Learning for Time Series
- Generative Modeling and Simulation
- Spatiotemporal Modeling
- Continuous-Time Modeling and Temporal Point Processes
- Causal and Structural Time-Series Modeling
- Interpretability and Reliability
- Benchmarks, Datasets, and Evaluation
- Domain-Specific Applications
- Theory and Statistical Foundations
- Surveys and Reviews

Scientific and Engineering Applications
- Scientific Machine Learning Theory and Foundations
- Physics-Informed and Mechanistic Learning
- Surrogate Modeling and Operator Learning
- Inverse Problems and Data Assimilation
- Scientific Discovery and Equation Learning
- Uncertainty Quantification and Reliability
- Scientific Benchmarks, Datasets, and Evaluation
- Physics and Astronomy Applications
- Chemistry and Materials Science Applications
- Earth, Climate, and Environmental Science Applications
- Energy and Power Systems Applications
- Mechanical and Aerospace Engineering Applications
- Civil and Structural Engineering Applications
- Electrical, Electronic, and Photonic Engineering Applications
- Manufacturing and Industrial Engineering Applications

Geospatial and Remote Sensing
- Remote Sensing Scene Understanding and Mapping
- Object Detection, Localization, and Counting in Overhead Imagery
- Change Detection and Monitoring
- Spatiotemporal Geospatial Modeling and Forecasting
- Hyperspectral and Multispectral Image Analysis
- Synthetic Aperture Radar (SAR) Analysis
- LiDAR and 3D Geospatial Learning
- Multimodal and Multisensor Fusion
- Image Restoration, Super-Resolution, and Pansharpening
- Geolocation, Registration, and Cross-view Geo-localization
- Spatial Graph Learning and Mobility Modeling
- Geospatial Representation Learning and Foundation Models
- Domain Adaptation, Transfer Learning, and Generalization
- Earth System and Societal Applications
- Datasets, Benchmarks, Evaluation, and Survey Studies

Finance and Economics
- Financial Time-Series Forecasting
- Asset and Derivative Pricing
- Portfolio Management and Asset Allocation
- Market Microstructure and Algorithmic Trading
- Credit Risk and Lending Analytics
- Risk Management and Stress Testing
- Fraud Detection and Financial Crime
- Corporate Finance and Financial Distress
- Insurance and Actuarial Modeling
- Financial Text and Alternative Data Analysis
- Macroeconomic Forecasting and Nowcasting
- Structural Econometrics and Equilibrium Modeling
- Causal Inference and Policy Evaluation
- Demand Estimation and Pricing
- Auctions and Market Design
- Economic Measurement and Socioeconomic Analysis

Cybersecurity
- Network Intrusion Detection and Traffic Analysis
- Malware Detection and Analysis
- Phishing, Spam, and Malicious Content Detection
- Vulnerability Detection and Software Security
- Host-Based Intrusion Detection and User Behavior Analytics
- Authentication, Access Control, and Biometric Security
- Threat Intelligence, Attack Attribution, and Security Operations
- Digital Forensics and Incident Response
- Adversarial Machine Learning and Secure AI Systems
- Cyber-Physical, IoT, and Industrial Control Security
- Cybersecurity Datasets, Benchmarks, and Evaluation

Industrial and Manufacturing Systems
- Predictive Maintenance and Remaining Useful Life Estimation
- Fault Diagnosis and Anomaly Detection
- Quality Inspection and Defect Detection
- Process Monitoring, Modeling, and Control
- Production Planning, Scheduling, and Shop-Floor Optimization
- Industrial Robotics and Autonomous Manufacturing
- Digital Twins and Smart Manufacturing Systems
- Supply Chain, Logistics, and Inventory Analytics
- Energy Management and Sustainable Manufacturing
- Industrial IoT, Sensor Fusion, and Edge Intelligence
- Industrial Machine Learning Methodology, Datasets, and Benchmarks
- Surveys and Reviews of Intelligent Manufacturing

Transportation and Mobility Systems
- Traffic Flow Forecasting and State Estimation
- Traffic Control and Network Operations
- Mobility Demand Forecasting and Human Mobility Analysis
- Routing, Navigation, and Travel Time Estimation
- Public Transit and Multimodal Transportation Analytics
- Fleet Management, Ride-Hailing, and Shared Mobility
- Logistics and Freight Transportation
- Autonomous and Connected Vehicle Systems
- Transportation Safety, Incident Detection, and Risk Assessment
- Transportation Infrastructure Monitoring and Asset Management
- Electric Mobility and Sustainable Transportation
- Transportation Data, Benchmarks, and Simulation

Computational Social Science and Digital Platforms
- Social Network Analysis
- Information Diffusion and Cascade Modeling
- User Behavior Modeling
- Social Media Mining and Content Understanding
- Opinion Mining, Public Discourse, and Political Communication
- Misinformation, Content Moderation, and Trust & Safety
- Recommender Systems and Personalization
- Online Communities and Collective Behavior
- Platform Economics and Marketplace Modeling
- Human Mobility and Spatiotemporal Social Dynamics
- Causal Inference and Policy Evaluation in Social Systems
- Fairness, Privacy, and Ethics in Digital Platforms
- Datasets, Benchmarks, and Measurement

architecture

Feedforward, Residual, and MLP Architectures
- Multilayer Perceptrons
- Residual Networks and Skip-Connection Architectures

- All-MLP Architectures
- Architectural Design and Connectivity Patterns
- Expressivity and Approximation Theory
- Optimization and Training Dynamics
- Initialization, Normalization, and Regularization
- Generalization, Robustness, and Representation Analysis
- Efficient and Compressed Feedforward Models
- Empirical Scaling and Benchmarking
- Applications of Feedforward, Residual, and MLP Architectures
- Surveys and Comparative Studies

Convolutional Architectures
- Convolutional Operators and Building Blocks
- Residual and Densely Connected Networks
- Fully Convolutional and Encoder-Decoder Networks
- Multi-Scale and Pyramid Networks
- Efficient and Lightweight Convolutional Networks
- 3D and Spatiotemporal Convolutional Networks
- Generative Convolutional Models
- Neural Architecture Search for CNNs
- Theoretical Analysis of Convolutional Networks
- Training and Optimization of Convolutional Networks
- Transfer Learning and Representation Learning with CNNs
- Robustness, Interpretability, and Uncertainty in CNNs
- Benchmarks and Empirical Evaluation of CNNs
- Application-Specific Convolutional Networks
- Surveys and Reviews of Convolutional Architectures

Recurrent and State-Space Architectures
- Recurrent Neural Networks
- Gated Recurrent Networks
- Recurrent Encoder-Decoder Models
- Memory-Augmented Recurrent Networks
- Reservoir Computing
- Latent and Neural State-Space Models
- Structured State Space Models
- Continuous-Time Recurrent and State-Space Models
- Training and Optimization of Recurrent and State-Space Models
- Theory and Expressivity
- Benchmarking and Empirical Evaluation
- Applications of Recurrent and State-Space Architectures
- Surveys and Systematic Reviews

Attention-Based and Transformer Architectures
- Attention Mechanisms
- Positional and Structural Encoding
- Transformer Architectural Variants
- Efficient and Scalable Transformers
- Long-Context and Memory-Augmented Transformers
- Training, Pre-training, and Adaptation
  - Pre-training Objectives
  - Pre-training Data and Curriculum
  - Optimization, Regularization, and Training Dynamics
  - Scaling Laws and Compute-Optimal Training
  - Efficient and Distributed Training
  - Knowledge Distillation and Teacher-Student Training
  - Fine-Tuning and Transfer Learning
  - Adapter and Low-Rank Adaptation
  - Prompt and Prefix Tuning
  - Domain Adaptation and Continual Learning
  - Instruction Tuning and Alignment
  - Theoretical Analysis of Pre-training and Adaptation
  - Evaluation and Benchmarking of Training and Adaptation
- Compression and Deployment
- Theory of Attention and Transformers
- Interpretability and Analysis
- Robustness, Safety, and Privacy
- Evaluation and Benchmarking
- Language Transformers
- Vision Transformers
- Speech and Audio Transformers
- Multimodal Transformers
- Graph and Structured Data Transformers

Graph Neural Architectures
- Spectral Graph Neural Networks
- Message Passing Neural Networks
- Graph Attention Networks
- Graph Transformers
- Hierarchical and Pooling Graph Neural Networks
- Heterogeneous and Relational Graph Neural Networks
- Temporal and Dynamic Graph Neural Networks
- Hypergraph and Higher-Order Graph Neural Networks
- Geometric and Equivariant Graph Neural Networks
- Graph Autoencoders and Generative Models
- Scalable and Efficient Graph Neural Architectures
- Theory and Expressivity of Graph Neural Architectures
- Graph Neural Architecture Search and Design
- Empirical Evaluation and Benchmarking of Graph Neural Architectures
- Applications of Graph Neural Architectures
- Surveys and Taxonomies of Graph Neural Architectures

Geometric and Equivariant Architectures
- Graph Neural Networks
- Set and Permutation-Equivariant Networks
- Point Cloud and 3D Geometric Networks
- Mesh and Surface Neural Networks
- Group-Equivariant Neural Networks
- E(n)- and SE(3)-Equivariant Neural Networks
- Gauge-Equivariant and Manifold Neural Networks
- Hyperbolic and Non-Euclidean Neural Networks
- Simplicial, Cellular, and Topological Neural Networks
- Symmetry Discovery and Learned Equivariance
- Theory of Geometric and Equivariant Learning
- Benchmarks and Empirical Evaluation for Geometric Learning
- Applications of Geometric and Equivariant Architectures
- Surveys and Taxonomies of Geometric Deep Learning

Generative Model Architectures
- Autoregressive Generative Models
- Variational Autoencoders and Deep Latent Variable Models
- Generative Adversarial Networks
- Normalizing Flow Models
- Diffusion and Score-Based Generative Models
- Energy-Based and Undirected Generative Models
- Masked and Denoising Generative Models

Multimodal Architectures
- Multimodal Fusion Architectures
- Joint Embedding and Alignment Models
- Cross-Attention and Multimodal Transformer Architectures
- Encoder-Decoder and Sequence-to-Sequence Multimodal Models
- Generative Multimodal Models
- Multimodal Foundation Models
- Modular and Mixture-of-Experts Multimodal Networks
- Robust and Incomplete Multimodal Learning
- Multimodal Reasoning and Grounding Architectures
- Efficient and Scalable Multimodal Architectures
- Analysis, Evaluation, and Benchmarking of Multimodal Architectures
- Domain-Specific Multimodal Architectures

Modular and Mixture-of-Experts Architectures
- Mixture-of-Experts Models
- Routing and Conditional Computation
- Modular Neural Networks
- Neural Module Networks
- Training and Optimization of Modular Architectures
- Theory of Modularity and Expert Specialization
- Scalable and Distributed MoE Systems
- Transfer, Multi-Task, and Continual Modular Learning
- Benchmarking and Empirical Evaluation
- Applications of Modular and Mixture-of-Experts Architectures
- Surveys and Taxonomies of Modular Architectures

Memory-Augmented and Neuro-Symbolic Architectures
- External Memory Architectures
- Associative, Episodic, and Working Memory Models
- Retrieval-Augmented Models
- Continual and Lifelong Learning with Memory
- Neural-Symbolic Knowledge Representation
- Neural-Symbolic Logical and Relational Reasoning
- Program Induction and Differentiable Programming
- Neural-Symbolic Planning and Decision Making
- Training and Optimization Methods
- Theory, Interpretability, and Compositional Generalization
- Benchmarks and Empirical Evaluation
- Applications of Memory-Augmented and Neuro-Symbolic Models

Efficient and Compact Architectures
- Lightweight Convolutional Architectures
- Efficient Transformer and Sequence Architectures
- Dynamic and Conditional Computation
- Sparse Architectures and Mixture-of-Experts
- Low-Precision and Binary Neural Networks
- Factorized and Low-Rank Architectures
- Neural Architecture Search for Efficiency
- Hardware-Aware and Edge-Oriented Architecture Design
- Theoretical Foundations of Efficient Architectures
- Benchmarking and Empirical Evaluation of Efficient Models

- Application-Specific Efficient Architectures
- Surveys and Comparative Studies of Compact Models

Neural Architecture Search and Automated Design
- Search Space Design
- Search Algorithms and Optimization
- Efficient NAS and Performance Prediction
- Multi-Objective and Hardware-Aware NAS
- Transfer, Meta, and Few-Shot NAS
- Joint Automated Design and AutoML
- Theory and Foundations of NAS
- Benchmarking, Evaluation, and Reproducibility
- Domain-Specific and Applied NAS
- Surveys and Empirical Studies of NAS

Architecture Theory and Analysis
- Expressivity and Approximation Theory
- Optimization and Training Dynamics
- Generalization and Capacity Analysis
- Signal Propagation and Dynamical Systems Analysis
- Inductive Bias and Representation Analysis
- Scaling Laws and Architectural Scaling
- Robustness and Stability Analysis
- Computational Complexity and Efficiency Analysis
- Empirical Benchmarking and Ablation Studies
- Task- and Domain-Specific Architectural Analysis
- Architecture Taxonomy and Survey

Architecture Benchmarking and Empirical Evaluation
- Benchmark Dataset and Task Suite Construction
- Experimental Protocols and Statistical Evaluation
- Comparative Benchmarking of Architecture Families
- Ablation Studies and Component-wise Analysis
- Scaling Behavior and Sensitivity Analysis
- Efficiency, Compute, and Hardware-Aware Benchmarking
- Robustness, Reliability, and Generalization Evaluation
- Transferability and Cross-Domain Evaluation
- Application-Specific and Real-World Evaluation
- Reproducibility, Replicability, and Reporting Studies
- Systematic Surveys and Meta-Analysis

Task-Specific and Domain-Specific Architectures
- Architecture Design Methodologies
- Theoretical Foundations of Specialized Architectures
- Benchmarking and Empirical Evaluation
- Surveys and Taxonomies
- Computer Vision Architectures
- Natural Language Processing Architectures
- Speech and Audio Architectures
- Multimodal Learning Architectures
- Graph and Relational Learning Architectures
- Time-Series and Forecasting Architectures
- Tabular Data Architectures
- Recommender System and Ranking Architectures
- Biomedical and Healthcare Architectures
- Scientific Machine Learning and Geospatial Architectures
- Reinforcement Learning and Control Architectures

Surveys and Taxonomies of Neural Architectures
- Cross-Architecture Taxonomies and Comparative Surveys
- Convolutional Neural Architectures
- Sequence Modeling Architectures
- Transformer and Attention-Based Architectures
- Graph Neural Architectures
- Generative and Latent-Variable Architectures
- Multimodal Neural Architectures
- Efficient and Hardware-Aware Architectures
- Neural Architecture Search
- Spiking and Neuromorphic Architectures
- Modular, Memory-Augmented, and Neuro-Symbolic Architectures
- Domain-Specific Architecture Surveys

data

Dataset Collection and Curation
- Data Acquisition and Sampling
- Data Annotation and Labeling Methodology
- Dataset Cleaning, Filtering, and Deduplication
- Dataset Quality Evaluation and Auditing
- Bias, Fairness, and Representativeness in Datasets
- Dataset Documentation, Metadata, and Provenance
- Privacy, Ethics, and Data Governance
- Benchmark Dataset Construction and Evaluation
- Synthetic and Programmatically Generated Datasets
- Dataset Maintenance, Versioning, and Lifecycle Management

Data Annotation and Supervision
- Annotation Schema Design
- Human Annotation and Crowdsourcing
- Annotation Quality and Inter-Annotator Agreement
- Label Aggregation and Annotator Modeling
- Active Learning
- Weak and Distant Supervision
- Semi-Supervised Learning
- Self-Supervised Learning
- Learning with Noisy Labels
- Human Feedback and Preference Supervision

Data Preprocessing and Cleaning
- Data Quality Assessment and Profiling
- Missing Data Imputation
- Noise Reduction and Denoising
- Outlier, Anomaly, and Inconsistency Handling
- Duplicate Detection and Entity Resolution
- Data Normalization, Scaling, and Transformation
- Encoding, Discretization, and Type Conversion
- Label Cleaning and Noisy Annotation Handling
- Imbalanced Data Handling and Resampling
- Data Integration, Harmonization, and Batch Effect Correction
- Automated, Interactive, and Human-in-the-Loop Data Cleaning
- Foundations, Benchmarks, and Evaluation of Data Preprocessing

Data Augmentation and Transformation
- Spatial and Geometric Augmentation
- Photometric and Signal-Level Augmentation
- Occlusion and Corruption-Based Augmentation
- Sample Mixing and Interpolation Methods
- Feature-Space and Latent-Space Augmentation
- Generative and Synthetic Data Augmentation
- Automated Augmentation Policy Learning
- Adversarial and Robustness-Oriented Transformations
- Modality- and Task-Specific Augmentation
- Theoretical Foundations of Augmentation
- Empirical Evaluation, Benchmarks, and Best Practices
- Test-Time Augmentation

Synthetic Data Generation
- Generative Modeling Methods
- Simulation and Procedural Generation
- Data Augmentation and Counterfactual Synthesis
- Visual Synthetic Data
- Text, Speech, and Multimodal Synthetic Data
- Tabular and Relational Synthetic Data
- Time-Series and Sensor Synthetic Data
- Graph and Structured Synthetic Data
- 3D and Spatial Synthetic Data
- Privacy-Preserving and Fair Synthetic Data
- Evaluation, Benchmarking, and Empirical Studies
- Theoretical Foundations of Synthetic Data
- Domain-Specific Applications of Synthetic Data

Data Quality and Label Noise
- Label Noise Models and Theory
- Noise-Robust Learning
- Label Error Detection and Correction
- Annotator Modeling and Crowdsourcing
- Weak Supervision and Label Aggregation
- Ambiguous, Partial, and Soft Label Learning
- Feature Noise, Missing Data, and Input Corruption
- Data Quality Assessment, Auditing, and Benchmarking
- Domain-Specific Learning with Imperfect Annotations

Data Selection and Valuation
- Active Learning
- Coreset Selection and Data Pruning
- Curriculum Learning and Self-Paced Learning
- Data Valuation and Data Pricing
- Training Data Attribution and Influence Estimation
- Importance Sampling and Sample Reweighting
- Theoretical Foundations and Guarantees
- Benchmarks and Empirical Evaluation
- Applications and Systems
- Surveys and Taxonomies

Dataset Bias, Fairness, and Representativeness
- Bias Taxonomies and Theory
- Sampling and Selection Bias
- Representation Bias and Coverage
- Annotation and Label Bias
- Measurement Bias
- Evaluation Bias and Benchmark Design
- Dataset Auditing and Documentation
- Fair Data Collection and Curation
- Data Debiasing and Rebalancing
- Distribution Shift and Subpopulation Representativeness

Dataset Shift and Distributional Robustness
- Covariate Shift

- Label Shift and Prior Probability Shift
- Concept Shift and Concept Drift
- Domain Adaptation
- Domain Generalization
- Test-Time Adaptation
- Out-of-Distribution Detection and Open-Set Recognition
- Distributionally Robust Optimization
- Subpopulation Shift and Group Robustness
- Causal Invariance and Stable Prediction
- Uncertainty Estimation and Calibration under Shift
- Shift Detection, Estimation, and Drift Monitoring
- Theoretical Foundations of Generalization under Shift
- Robustness Benchmarks and Evaluation

Benchmarks and Dataset Evaluation
- Dataset Characterization and Quality Assessment
- Annotation and Label Quality Evaluation
- Dataset Bias, Fairness, and Representativeness
- Split Integrity, Leakage, and Redundancy Analysis
- Benchmark Design and Evaluation Protocols
- Robustness, Distribution Shift, and Stress-Test Benchmarks
- Synthetic, Simulated, and Diagnostic Benchmarks
- Benchmark Validity, Reliability, and Meta-Evaluation
- Dataset Documentation, Provenance, and Governance
- Domain-Specific and Application Benchmarks
- Survey and Comparative Studies of Benchmarks

Data Privacy, Security, and Governance
- Differential Privacy for Machine Learning
- Privacy-Preserving Federated and Distributed Learning
- Secure Computation and Confidential Machine Learning
- Privacy Attacks, Leakage, and Auditing
- Data Anonymization, De-identification, and Synthetic Data
- Training-Data Security: Poisoning and Backdoors
  - Availability Poisoning Attacks
  - Targeted and Integrity Poisoning Attacks
  - Backdoor and Trojan Attacks
  - Clean-Label and Stealthy Poisoning
  - Poisoning and Backdoors in Federated and Distributed Learning
  - Poisoning and Backdoors in Specialized Learning Paradigms
  - Poisoning Detection and Data Sanitization
  - Backdoor Detection and Model Forensics
  - Robust Training, Mitigation, and Machine Unlearning
  - Certified Robustness and Theoretical Analysis
  - Benchmarks, Datasets, and Evaluation Protocols
  - Surveys, Taxonomies, and Systematization
- Data Provenance, Lineage, and Documentation
- Data Governance, Access Control, and Compliance

evaluation

Evaluation Metrics
- Predictive Performance Metrics
- Ranking, Retrieval, and Recommendation Metrics
- Decision-Making and Reinforcement Learning Metrics
- Structured Prediction and Sequence Generation Metrics
- Generative Model Evaluation Metrics
- Representation and Embedding Evaluation Metrics
- Calibration and Uncertainty Metrics
- Robustness, Generalization, and Safety Metrics
- Fairness, Bias, and Responsible AI Metrics
- Efficiency, Scalability, and Sustainability Metrics
- Interpretability and Explanation Metrics
- Human-Centered and Preference-Based Evaluation
- Metric Theory and Statistical Validity
- Meta-Evaluation and Metric Benchmarking
- Metric Design and Learned Metrics

Benchmarking and Comparative Evaluation
- Benchmark Dataset and Task Design
- Evaluation Metrics and Protocols
  - Predictive Performance Metrics
  - Probabilistic Evaluation and Calibration
  - Meta-Evaluation and Metric Validation
  - Statistical Comparison and Significance Testing
  - Experimental Design and Validation Protocols
  - Benchmark and Test Suite Design
  - Robustness and Distribution Shift Evaluation
  - Fairness and Bias Evaluation
  - Efficiency and Resource-Aware Evaluation
  - Human Evaluation and Preference Studies
  - Reproducibility and Reporting Standards
- Comparative Model and Algorithm Studies
  - Empirical Benchmark Studies
  - Cross-Dataset Generalization and Transfer Studies
  - Efficiency and Scalability Evaluation
  - Robustness and Reliability Evaluation
  - Fairness, Bias, and Safety Evaluation
  - Theoretical Comparative Analysis
  - Statistical Evaluation and Comparison Methodology
  - Reproducibility and Replication Studies
  - Application-Specific Comparative Studies
  - Systematic Reviews and Meta-Analyses
- Reproducibility and Replicability Studies
- Robustness and Stress Testing Benchmarks
- Efficiency and Resource Benchmarking
- Generalization and Transfer Benchmarks
- Fairness, Safety, Privacy, and Security Benchmarking
- Domain-Specific Application Benchmarks
- Benchmark Auditing and Meta-Evaluation
- Benchmark Surveys and Meta-Analyses

Experimental Design and Statistical Evaluation
- Data Splitting and Sampling Protocols
- Model Selection and Validation Procedures
  - Data partitioning and holdout strategies
  - Cross-validation methods
  - Bootstrap and resampling-based validation
  - Nested validation and selection bias control
  - Information-theoretic and Bayesian model selection criteria
  - Hyperparameter and architecture selection
  - Early stopping and adaptive stopping rules
  - Multi-fidelity and budget-aware selection
  - Validation for dependent, structured, and limited data
  - External validation and distribution-shift-aware selection
  - Statistical inference for model comparison
  - Theoretical foundations of model selection and validation
  - Empirical benchmarking of validation procedures
  - Applied and domain-specific validation protocols
  - Surveys and meta-analyses of validation practice
- Baseline, Control, and Ablation Design
- Benchmarking and Comparative Evaluation Protocols
  - Benchmark Suite Construction
  - Evaluation Metrics and Composite Scoring
  - Baselines, Reference Implementations, and Tuning Protocols
  - Statistical Methodology for Comparative Evaluation
  - Systematic Comparative Benchmark Studies
  - Reproducibility and Reporting Standards
  - Robustness and Generalization Benchmarking
  - Efficiency and Resource-Aware Benchmarking
  - Fairness, Safety, and Responsible Evaluation
  - Meta-Evaluation and Benchmark Validity
  - Dynamic and Continual Benchmarking
- Statistical Hypothesis Testing
- Effect Size and Uncertainty Quantification
- Power Analysis and Sample Size Determination
- Multiple Comparisons and Error Control
- Robustness and Sensitivity Analysis
- Reproducibility and Replicability Assessment

Generalization and Distribution Shift Evaluation
- Statistical Generalization Theory
- Empirical Generalization Diagnostics
- Dataset Shift Characterization and Metrics
- Domain Generalization and Cross-Domain Evaluation
- Out-of-Distribution Detection and Open-Set Evaluation
- Uncertainty Quantification and Calibration Under Shift
- Robustness to Corruptions and Adversarial Perturbations
- Subpopulation Shift and Worst-Group Evaluation
- Temporal Shift and Data Drift Evaluation
- Benchmarks, Datasets, and Evaluation Protocols

Robustness and Adversarial Evaluation
- Adversarial Attacks and Threat Models
- Defense Evaluation and Robustness Auditing

- Certified Robustness and Formal Verification
- Corruption and Perturbation Robustness
- Distribution Shift and Subpopulation Robustness
- Out-of-Distribution and Open-Set Evaluation
- Data Poisoning and Backdoor Robustness
  - Threat Models and Formal Foundations
  - Availability Poisoning Attacks
  - Targeted Poisoning Attacks
  - Backdoor and Trojan Attacks
  - Poisoned Data Detection and Sanitization
  - Backdoor Detection and Removal
  - Robust Training and Defense Algorithms
  - Certified Defenses and Robustness Guarantees
  - Federated and Distributed Poisoning and Backdoors
  - Benchmarks, Datasets, and Evaluation Protocols
  - Application- and Modality-Specific Robustness
  - Surveys and Taxonomies
- Physical-World Adversarial Robustness
- Robustness Benchmarks, Metrics, and Evaluation Protocols
- Theoretical Analysis of Robustness

Calibration and Uncertainty Evaluation
- Theoretical Foundations of Calibration and Uncertainty
- Calibration Metrics, Scoring Rules, and Diagnostics
- Post-hoc Calibration Methods
- Training-time Calibration and Confidence Regularization
- Predictive Uncertainty Estimation Methods
- Aleatoric and Epistemic Uncertainty Modeling
- Conformal Prediction and Coverage Guarantees
- Selective Prediction, Abstention, and Risk-Coverage Evaluation
- Distribution Shift, OOD Detection, and Reliability
- Regression Calibration and Predictive Intervals
- Structured, Dense, and Sequential Prediction Calibration
- Group Calibration, Multicalibration, and Fairness
- Benchmarking, Datasets, and Comparative Studies
- Application-specific Reliability and Safety Evaluation
- Surveys and Taxonomies of Calibration and Uncertainty

Efficiency and Resource Evaluation
- Computational Complexity and Operation Count
- Latency and Throughput Evaluation
- Memory Footprint and Storage Efficiency
- Energy and Power Measurement
- Communication and Bandwidth Efficiency
- Scalability and Parallel Efficiency
- Hardware-Aware Performance Evaluation
- Accuracy-Efficiency Trade-off Analysis
- Cost and Environmental Impact Assessment
- Efficiency Benchmarking and Profiling Methodology

Fairness, Bias, and Responsible AI Evaluation
- Statistical Fairness Metrics
- Causal and Counterfactual Fairness Evaluation
- Subgroup and Intersectional Fairness
- Dataset and Annotation Bias Evaluation
- Model Auditing and Disparity Analysis
- Benchmarking, Stress Testing, and Reproducibility
- Fairness Mitigation Evaluation
- Human-Centered Harm Assessment
- Transparency, Documentation, and Accountability Assessment
- Domain-Specific Fairness Evaluation

Interpretability and Human-Centered Evaluation
- Intrinsically Interpretable Models
- Global Explanations and Rule Extraction
- Feature Attribution and Saliency
- Example-Based Explanations
- Concept-Based Explanations
- Counterfactual Explanations and Algorithmic Recourse
- Mechanistic Interpretability
- Theoretical Foundations of Interpretability
- Faithfulness, Robustness, and Stability Evaluation
- Human-Subject Evaluation of Explanations
- Interactive Explanations and Human-AI Collaboration
- Benchmarks, Datasets, and Evaluation Protocols
- Debugging, Auditing, and Safety Applications

Reproducibility and Meta-Evaluation
- Computational Reproducibility
- Independent Replication Studies
- Statistical Evaluation Methodology
- Benchmark and Leaderboard
  - Meta-Evaluation
  - Benchmark Design and Curation
  - Construct Validity and Representativeness
  - Evaluation Metrics and Ranking Methods
  - Statistical Reliability and Rank Stability
  - Reproducibility and Replication Studies
  - Data Contamination and Information Leakage
  - Bias, Fairness, and Coverage Analysis
  - Robustness and Stress Testing
  - Benchmark Aging, Saturation, and Dynamic Benchmarking
  - Cross-Benchmark Comparison and Meta-Analysis
  - Resource-Aware and Multi-Objective Benchmarking
- Metric and Protocol Validation
- Reporting Standards and Experimental Documentation
- Reproducibility Infrastructure and Open Science
- Meta-Research on Research Reliability

Domain-Specific and Application-Oriented Evaluation
- Biomedical and Healthcare Evaluation
- Autonomous Systems and Robotics Evaluation
- Natural Language and Communication Applications Evaluation
- Vision and Multimedia Applications Evaluation
- Recommender Systems and Personalization Evaluation
- Finance, Economics, and Risk Evaluation
- Cybersecurity and Trustworthy Computing Evaluation
- Scientific and Engineering Applications Evaluation
- Industrial, Manufacturing, and Operations Evaluation
- Education, Social, and Public-Sector Evaluation
- Human-Centered and Decision-Support Evaluation
- Cross-Domain Benchmarking and Evaluation Methodology

interpretability

Feature Attribution and Saliency Methods
- Gradient-Based Saliency Methods
- Perturbation-Based Attribution Methods
- Relevance Propagation and Decomposition Methods
- Shapley-Value and Additive Attribution
- Class Activation Mapping
- Causal and Interventional Attribution
- Theoretical Foundations of Attribution
- Attribution Evaluation and Validation
- Robustness and Reliability of Saliency Methods
- Benchmarking and Comparative Studies
- Applied Feature Attribution
- Surveys and Review Articles

Counterfactual and Contrastive Explanations
- Counterfactual Explanation Generation
- Contrastive Explanation Methods
- Causal Counterfactual Explanations
- Actionable Counterfactual Explanations and Algorithmic Recourse
- Theoretical Foundations of Counterfactual and Contrastive Explanations
- Evaluation Metrics and Benchmarking
- Robustness, Fairness, Privacy, and Security
- Human-Centered Evaluation and Applications
- Surveys, Taxonomies, and Comparative Reviews

Example-Based Explanations
- Retrieval-Based Explanations
- Prototype and Criticism Explanations
- Training Data Attribution
- Counterfactual and Contrastive Explanations
- Generative Example-Based Explanations
- Evaluation and Benchmarking
- Human-Centered and Applied Example-Based Explanations
- Theory, Robustness, and Privacy

Concept-Based Explanations
- Concept attribution methods
- Automatic concept discovery
- Concept representation learning
- Concept bottleneck and concept-supervised models
- Causal and counterfactual concept explanations
- Knowledge-grounded and language-based concepts
- Robustness and theoretical foundations
- Evaluation, benchmarks, and user studies
- Application-specific concept-based explanations
- Surveys and taxonomies of concept-based explanations

Surrogate, Rule-Based, and Symbolic Explanations
- Local Surrogate Explanations
- Global Surrogate Models
- Interpretable Model Distillation
- Rule Extraction
- Decision Trees, Rule Lists, and Decision Sets
- Symbolic and Logical Explanations
- Counterfactual Rule and Symbolic Reasoning
- Fidelity, Faithfulness, and Complexity Analysis
- Robustness, Stability, and Identifiability

- Evaluation, Benchmarks, and Human Studies
- Domain-Specific Applications
- Surveys and Comparative Studies

Intrinsically Interpretable Models
- Sparse Linear Models and Scoring Systems
- Generalized Additive Models
- Decision Trees
- Rule-Based Models
- Prototype and Case-Based Models
- Concept-Based Models
- Self-Explaining Neural Networks
- Monotonic and Shape-Constrained Models
- Theory and Guarantees for Intrinsic Interpretability
- Evaluation, Benchmarks, and Human Studies
- Domain Applications of Intrinsically Interpretable Models
- Surveys and Comparative Studies

Mechanistic Interpretability
- Feature Discovery and Neuron Analysis
- Circuit and Pathway Analysis
- Representation Geometry and Superposition
- Causal Intervention and Mediation Analysis
- Learning Dynamics of Mechanisms
- Automated and Scalable Mechanistic Interpretability
- Benchmarks and Evaluation
- Mechanistic Auditing, Steering, and Model Editing
- Architecture- and Modality-Specific Mechanistic Studies
- Surveys, Taxonomies, and Foundations

Representation Analysis and Probing
- Diagnostic Probing
- Probe Methodology and Evaluation
- Representational Similarity and Alignment
- Geometric and Topological Analysis of Representations
- Concept-based Representation Analysis
- Neuron, Head, and Subspace Analysis
- Representation Dynamics Across Layers and Training
- Causal and Interventional Probing
- Theoretical Foundations of Probing and Representational Interpretability

Visualization and Interactive Interpretability
- Feature, Neuron, and Filter Visualization
- Attribution and Saliency Visualization
- Representation and Embedding Visualization
- Concept-Based Visualization
- Example-, Prototype-, and Counterfactual-Based Visualization
- Interactive Visual Analytics for Model Inspection
- Human-in-the-Loop Debugging and Model Steering
- Evaluation, Human Factors, and Design Principles
- Benchmarks and Comparative Studies
- Application-Specific Visual Explanations
- Surveys and Taxonomies

Evaluation, Faithfulness, and Robustness of Explanations
- Theoretical and Axiomatic Foundations
- Faithfulness and Fidelity Metrics
- Causal and Counterfactual Evaluation
- Robustness, Stability, and Sensitivity Analysis
- Sanity Checks and Randomization Tests
- Human-Grounded Evaluation
- Application-Grounded Evaluation
- Benchmarks, Datasets, and Evaluation Protocols
- Adversarial Attacks and Defenses on Explanations
- Comparative Studies, Meta-Evaluation, and Surveys

Theoretical Foundations of Interpretability
- Formal Definitions and Conceptual Frameworks
- Axiomatic and Game-Theoretic Attribution
- Causal and Counterfactual Foundations
- Statistical and Information-Theoretic Foundations
- Complexity, Learnability, and Identifiability
- Faithfulness, Fidelity, and Completeness
- Robustness, Stability, and Generalization of Explanations
- Representation, Concepts, and Mechanistic Structure
- Human-Grounded Interpretability and Simulatability
- Evaluation Theory and Benchmarking
- Interpretability–Accuracy Trade-offs

optimization

Optimization Theory and Convergence Analysis
- Convex Optimization Theory
- Nonconvex Optimization Theory
- Stochastic Optimization and SGD Analysis
- Acceleration and Momentum Methods
- Adaptive, Preconditioned, and Second-Order Optimization
- Composite, Proximal, and Regularized Optimization
- Constrained, Saddle-Point, and Minimax Optimization
- Bilevel and Hyperparameter Optimization Theory
- Distributed, Parallel, and Federated Optimization Theory
- Optimization Landscapes, Geometry, and Implicit Bias
- Complexity Analysis and Lower Bounds
- Empirical Analysis and Benchmarking of Optimizers
- Surveys and Unifying Frameworks

Loss Landscape and Nonconvex Optimization
- Nonconvex Optimization Theory
- Loss Landscape Geometry and Topology
- Overparameterization and Benign Nonconvexity
- First-Order Nonconvex Optimization Methods
- Second-Order and Curvature-Aware Optimization
- Stochastic Optimization Dynamics
- Initialization, Conditioning, and Reparameterization
- Flatness, Sharpness, and Generalization
- Implicit Bias and Implicit Regularization
- Empirical Loss Landscape Analysis and Visualization
- Benchmarking and Comparative Evaluation
- Surveys and Review Articles

Stochastic Optimization
- Stochastic Gradient Methods
- Momentum and Accelerated Methods
- Adaptive Gradient Methods
- Variance-Reduced Stochastic Optimization
- Stochastic Second-Order and Quasi-Newton Methods
- Proximal and Constrained Stochastic Optimization
- Sampling, Mini-batching, and Data Ordering
- Nonconvex and Nonsmooth Stochastic Optimization Theory
- Generalization, Stability, and Implicit Bias
- Stochastic Min-Max and Saddle-Point Optimization
- Distributed, Parallel, and Asynchronous Stochastic Optimization
- Federated and Decentralized Stochastic Optimization
- Robust and Private Stochastic Optimization
- Empirical Evaluation and Benchmarking of Stochastic Optimizers
- Applications of Stochastic Optimization in Deep Learning
- Surveys and Tutorials on Stochastic Optimization

Adaptive Gradient and Momentum Methods
- AdaGrad Family and Per-Coordinate Adaptive Methods
- RMSProp and AdaDelta Family
- Adam Family and Adaptive Moment Estimation
- Momentum SGD and Nesterov Methods
- Optimizer Variant Design and Regularization-Aware Methods
  - Decoupled Weight Decay and Explicit Regularization
  - Proximal, Projected, and Composite Adaptive Optimization
  - Sharpness-Aware and Generalization-Oriented Optimization
  - Sparsity-Inducing and Structured Regularization Methods
  - Scale-Invariant and Norm-Aware Adaptive Methods
  - Stable, Rectified, and Robust Adaptive Methods
  - Hyperparameter Adaptation and Schedule Design
  - Theoretical Analysis of Regularization-Aware Optimizers
  - Benchmarking, Ablation, and Reproducibility Studies
  - Domain-Specific and Large-Scale Applications
  - Surveys, Taxonomies, and Comparative Reviews
- Convergence Theory and Complexity Analysis
- Generalization, Stability, and Implicit Bias
- Hyperparameter Tuning and Scheduler Interaction
- Distributed, Large-Batch, and Memory-Efficient Optimization
- Benchmarking, Reproducibility, and Empirical Comparison
- Application-Specific Optimizer Adaptation
- Surveys and Unifying Frameworks

Second-Order and Geometry-Aware Optimization
- Newton and Gauss-Newton Methods
- Quasi-Newton Methods
- Hessian-Free and Matrix-Free Methods
- Trust-Region and Cubic-Regularized Optimization
- Natural Gradient and Information-Geometric Optimization
- Riemannian and Manifold Optimization
- Mirror Descent and Bregman Geometry
- Curvature Approximation and Preconditioning
- Stochastic and Large-Scale Curvature-Aware Optimization
- Nonconvex Theory and Convergence Analysis
- Empirical Evaluation and Benchmarks

- Comparative Optimizer Studies
- Benchmark Suite Design and Curation
- Convergence and Training Dynamics Evaluation
- Generalization and Robustness Evaluation
- Scalability and Large-Scale Evaluation
- Computational Efficiency and Systems Benchmarking
- Hyperparameter Sensitivity and Ablation Studies
- Application-Specific Benchmarking
- Reproducibility and Evaluation Methodology
- Empirical Surveys and Meta-Analysis
- Applications and Domain-Specific Adaptations
- Surveys and Unifying Frameworks

Constrained, Proximal, and Regularized Optimization
- Lagrangian and Primal-Dual Methods
- Penalty, Barrier, and Augmented Lagrangian Methods
- Projected, Feasibility, and Projection-Free Methods
- Proximal Point, Proximal Gradient, and Composite Optimization
- Operator Splitting and ADMM
- Nonsmooth and Stochastic Constrained Optimization
- Regularization Theory and Penalty Design
- Structured Sparsity and Low-Rank Regularization
- Constraint-Driven Learning Applications
- Convergence Theory, Benchmarks, and Surveys

Bilevel Optimization and Meta-Optimization
- Bilevel Optimization Theory and Complexity
- Hypergradient Computation and Implicit Differentiation
- Stochastic, Online, and Distributed Bilevel Algorithms
- Hyperparameter and Regularization Optimization
- Meta-Learning and Few-Shot Adaptation
- Learned Optimizers and Learning to Optimize
- Neural Architecture Search
- Data Selection, Reweighting, and Distillation
- Robust, Adversarial, and Fairness-Aware Bilevel Learning
- Application-Driven Bilevel Optimization
- Benchmarks and Empirical Evaluation
  - Synthetic and Controlled Testbeds
  - Benchmark Datasets and Task Suites
  - Evaluation Metrics and Experimental Protocols
  - Comparative Empirical Studies
  - Ablation and Sensitivity Analysis
  - Robustness, Stability, and Generalization Studies
  - Scalability, Efficiency, and Systems Evaluation
  - Application-Oriented Case Studies
  - Reproducibility and Replication Studies
  - Surveys and Meta-Analysis
- Surveys, Taxonomies, and Software

Distributed, Parallel, and Federated Optimization
- Synchronous Distributed Optimization
- Asynchronous Parallel Optimization
- Decentralized Optimization
- Model-Parallel and Pipeline Optimization
- Communication-Efficient Optimization
- Cross-Device Federated Optimization
- Cross-Silo Federated Optimization
- Personalized and Heterogeneous Federated Optimization
- Privacy-Preserving and Robust Optimization
- Theory and Convergence Analysis
- Systems, Benchmarks, and Applications

Efficient and Hardware-Aware Training Optimization
- Efficient Optimizer Design
- Mixed-Precision and Low-Precision Training
- Memory-Efficient Training
- Distributed and Parallel Training
- Communication-Efficient Optimization
- Sparse and Structured Training
- Compiler, Kernel, and Runtime Optimization
- Hardware-Software Co-Design for Training
- Energy-Efficient and Power-Aware Training
- Performance Modeling, Profiling, and Benchmarking

Derivative-Free and Black-Box Optimization
- Zeroth-Order Optimization
- Direct Search and Pattern Search
- Surrogate-Based Derivative-Free Optimization
- Bayesian Optimization
- Evolutionary Algorithms and Evolution Strategies
- Bandit-Based and Multi-Fidelity Optimization
- Constrained and Multi-Objective Black-Box Optimization
- High-Dimensional and Parallel Black-Box Optimization
- AutoML and Neural Architecture Search
- Gradient-Free Policy Search and Control
- Black-Box Adversarial Optimization and Robustness
- Theory and Convergence Analysis
- Benchmarking and Empirical Evaluation
- Surveys, Libraries, and Systems

Empirical Evaluation and Benchmarking of Optimizers
- Benchmark Suite and Evaluation Protocol Design
- Comparative Performance Studies
  - Benchmark-Based Optimizer Comparison
  - Convergence Speed and Training Efficiency
  - Generalization Performance Comparison
  - Robustness and Stability Analysis
  - Hyperparameter Sensitivity and Tuning Budget Analysis
  - Scalability and Systems Efficiency
  - Task- and Architecture-Specific Comparative Studies
  - Application-Oriented Optimizer Evaluation
  - Empirical Validation of Convergence Theory
  - Evaluation Methodology and Statistical Testing
  - Systematic Reviews and Meta-Analysis
- Hyperparameter Sensitivity and Tuning Evaluation
  - Hyperparameter Sensitivity Analysis
  - Hyperparameter Interaction Analysis
  - Robustness to Hyperparameter Misspecification
  - Hyperparameter Optimization Benchmarks
  - Tuning Budget and Sample-Efficiency Evaluation
  - Transferability and Default Configuration Studies
  - Multi-Fidelity and Early-Stopping Evaluation
  - Search-Space Design and Tuning Protocols
  - Fair Comparison, Reproducibility, and Reporting Standards
  - Theoretical Analysis of Hyperparameter Effects
  - Domain-Specific and Cross-Task Tuning Studies
  - Meta-Benchmarking and Survey Studies
- Robustness, Stability, and Generalization Benchmarking
- Training Dynamics and Convergence Diagnostics
- Computational Efficiency and Scalability Benchmarking
- Task- and Domain-Specific Optimizer Benchmarks
  - Computer Vision Optimizer Benchmarks
  - Natural Language Processing Optimizer Benchmarks
  - Speech and Audio Optimizer Benchmarks
  - Generative Modeling Optimizer Benchmarks
  - Reinforcement Learning and Control Optimizer Benchmarks
  - Graph Machine Learning Optimizer Benchmarks
  - Recommendation and Ranking Optimizer Benchmarks
  - Time-Series and Tabular Learning Optimizer Benchmarks
  - Scientific, Engineering, and Biomedical Optimizer Benchmarks
  - Foundation Model Pretraining and Fine-Tuning Optimizer Benchmarks
    - Language Model Pretraining Optimizer Benchmarks
    - Vision Foundation Model Pretraining Optimizer Benchmarks
    - Multimodal Foundation Model Pretraining Optimizer Benchmarks
    - Speech and Audio Foundation Model Pretraining Optimizer Benchmarks
    - Full-Parameter Fine-Tuning Optimizer Benchmarks
    - Parameter-Efficient Fine-Tuning Optimizer Benchmarks
    - Instruction Tuning and Alignment Optimizer Benchmarks
    - Domain Adaptation and Continual Adaptation Optimizer Benchmarks
    - Efficiency and Scalability Benchmarking
    - Stability, Robustness, and Hyperparameter Sensitivity Benchmarks
    - Optimization Dynamics and Scaling-Law Studies
    - Benchmark Methodology and Reproducibility
    - Survey and Meta-Analysis of Foundation-Model Optimizer Benchmarks
  - Federated, Distributed, and Continual Learning Optimizer Benchmarks
  - Benchmark Design, Reproducibility, and Cross-Domain Meta-Analysis
- Reproducibility and Statistical Evaluation Methodology

- Survey and Meta-Analysis of Optimizer Benchmarks

privacy

- Differential Privacy
  - Differential Privacy Theory and Private Learning Theory
  - Differentially Private Optimization and Empirical Risk Minimization
  - Differentially Private Deep Learning Methods
  - Local and Shuffle Differential Privacy
  - Federated and Distributed Differentially Private Learning
  - Differentially Private Synthetic Data and Generative Modeling
  - Privacy Accounting, Systems, and Tooling
  - Auditing, Evaluation, and Benchmarking for DP ML
  - Applied Differential Privacy in Machine Learning
- Secure and Privacy-Preserving Computation
  - Secure Multiparty Computation for Machine Learning
  - Homomorphic Encryption for Machine Learning
  - Trusted Execution Environments for Machine Learning
  - Hybrid Secure Computation for Machine Learning
  - Secure and Private Model Inference
  - Privacy-Preserving Model Training
  - Secure Aggregation and Federated Protocols
  - Verifiable Computation for Machine Learning
  - Threat Models, Leakage, and Formal Security Analysis
  - Systems, Compilers, and Benchmarking for Private ML
  - Applications and Surveys of Secure ML
- Federated and Distributed Privacy
  - Federated Differential Privacy
  - Secure Aggregation and Cryptographic Protocols
  - Privacy Attacks and Leakage Analysis
  - Privacy Auditing and Benchmarking
  - Private Federated Optimization and Systems
  - Personalized and Heterogeneous Private Learning
  - Vertical Federated Learning Privacy
  - Split Learning Privacy
  - Decentralized Privacy-Preserving Learning
  - Federated Analytics and Private Data Release
- Privacy Attacks and Leakage Analysis
  - Membership Inference Attacks
  - Attribute Inference Attacks
  - Property Inference Attacks
  - Model Inversion and Input Reconstruction
  - Training Data Extraction and Memorization Analysis
  - Gradient Leakage and Reconstruction Attacks
  - Re-identification and Linkage Attacks
  - Side-Channel and Systems-Level Privacy Leakage
  - Privacy Auditing and Leakage Quantification
  - Theoretical Privacy Leakage Analysis
  - Benchmarks and Empirical Evaluation of Privacy Attacks
  - Application-Specific Privacy Attack Studies
- Privacy Auditing and Benchmarking
  - Membership Inference Auditing
  - Attribute and Property Inference Auditing
  - Model Inversion and Reconstruction Auditing
  - Training Data Extraction and Memorization Auditing
  - Differential Privacy Auditing and Verification
    - Formal Differential Privacy Verification
    - Privacy Accounting and Composition Verification
    - Statistical Privacy Auditing
    - Attack-Based Privacy Evaluation
    - Systems and Implementation Auditing
    - Benchmarking and Evaluation Methodology
    - Federated, Distributed, and Local DP Auditing
    - Group, User-Level, and Heterogeneous Risk Auditing
    - Online Privacy Monitoring and Budget Control
    - Reproducibility, Reporting, and Survey Studies
  - Federated and Distributed Learning Privacy Auditing
  - Privacy Benchmark Design and Standardization
  - Audit Methodology, Metrics, and Statistical Testing
  - Application-Specific and Deployment-Scale Privacy Evaluation
- Privacy-Preserving Data Release and Synthetic Data
  - Differentially Private Data Release
  - Privacy-Preserving Synthetic Data Generation
  - Anonymization and De-identification
  - Statistical and Query-Based Release
  - Privacy Risk Analysis and Auditing
  - Utility, Fidelity, and Fairness Evaluation
  - Theory and Privacy-Utility Tradeoffs
  - Benchmarks and Comparative Evaluation
  - Domain-Specific Applications
  - Surveys and Taxonomies
- Machine Unlearning and Data Deletion
  - Theoretical Foundations of Machine Unlearning
  - Exact Unlearning
  - Certified Unlearning
  - Approximate Unlearning
  - Scalable Unlearning Systems
  - Unlearning Verification and Auditing
  - Unlearning Benchmarks, Metrics, and Surveys
  - Federated and Distributed Unlearning
  - Foundation and Generative Model Unlearning
  - Security and Safety-Oriented Unlearning
  - Applied and Domain-Specific Unlearning
- Privacy Theory and Privacy-Utility Tradeoffs
  - Differential Privacy Theory
  - Information-Theoretic Privacy
  - Private Learning Theory
  - Privacy Accounting and Composition
  - Optimal Privacy Mechanisms
  - Privacy Leakage Quantification
  - Empirical Privacy-Utility Evaluation
  - Application-Specific Tradeoff Studies
  - Surveys and Comparative Studies
- Privacy-Preserving ML Systems and Applications
  - Differentially Private Learning
  - Federated and Collaborative Learning
  - Secure Multiparty Computation for ML
  - Homomorphic Encryption for ML
  - Trusted Execution Environments for ML
  - Privacy-Preserving Data Sharing and Synthetic Data
  - Privacy-Preserving Inference
  - Systems Optimization and Scalability
  - Privacy Evaluation, Auditing, and Benchmarking
  - Domain-Specific Privacy-Preserving ML Applications
  - Surveys, Taxonomies, and Comparative Studies

probabilistic_modeling

- Probabilistic Graphical Models
  - Directed Graphical Models
  - Undirected Graphical Models
  - Factor Graphs and Message Passing
  - Latent Variable and Hierarchical Graphical Models
  - Temporal and Sequential Graphical Models
  - Causal Graphical Models
  - Statistical Relational Learning
  - Inference in Graphical Models
  - Learning and Structure Learning in Graphical Models
  - Neural and Deep Graphical Models
  - Theory and Foundations of Graphical Models
  - Applications of Graphical Models
  - Benchmarks and Empirical Evaluation of Graphical Models
  - Surveys and Tutorials on Graphical Models
- Bayesian Modeling and Bayesian Deep Learning
  - Bayesian Inference and Posterior Approximation
  - Bayesian Neural Networks
  - Bayesian Latent Variable and Generative Models
  - Hierarchical Bayesian Models and Probabilistic Graphical Models
  - Gaussian Processes and Bayesian Nonparametrics
  - Uncertainty Quantification and Calibration
  - Bayesian Optimization, Active Learning, and Experimental Design
  - Bayesian Reinforcement Learning and Sequential Decision-Making
  - Scalable Bayesian Computation and Probabilistic Programming
  - PAC-Bayesian Theory and Bayesian Learning Theory
  - Empirical Evaluation and Benchmarks
  - Applications of Bayesian Deep Learning
  - Surveys and Tutorials in Bayesian Modeling
- Latent Variable Models
  - Continuous Latent Variable Models
  - Discrete Latent Variable Models
  - Hierarchical Latent Variable Models
  - Sequential and State-Space Latent Variable Models
  - Structured Latent Variable Models
  - Undirected and Energy-Based Latent Variable Models
  - Multimodal and Conditional Latent Variable Models
  - Inference and Learning Algorithms for Latent Variable Models
  - Disentangled and Identifiable Latent Representation Learning
  - Theory of Latent Variable Models
  - Evaluation and Benchmarking of Latent Variable Models
  - Applications of Latent Variable Models
  - Surveys and Reviews of Latent Variable Models
- Deep Generative Models
  - Deep Latent Variable Models
  - Generative Adversarial Networks
  - Autoregressive Generative Models
  - Normalizing Flows
  - Diffusion and Score-Based Models

- Energy-Based Models
- Conditional and Controllable Generation
- Representation Learning and Disentanglement
- Multimodal and Structured Generative Modeling
- Theory, Inference, and Optimization for Generative Models
- Evaluation, Benchmarks, and Empirical Comparison
- Applications of Deep Generative Models
- Surveys and Taxonomies of Deep Generative Models

Approximate Inference and Monte Carlo Methods
- Variational Inference
- Markov Chain Monte Carlo
- Sequential Monte Carlo
- Importance Sampling and Monte Carlo Estimation
- Expectation Propagation and Message Passing
- Deterministic Approximate Inference
- Amortized Inference
- Likelihood-Free and Simulation-Based Inference
- Scalable and Automatic Inference
- Theoretical Foundations of Approximate Inference
- Evaluation, Benchmarks, and Reviews
- Applications of Approximate Inference

Sequential and Temporal Probabilistic Models
- Autoregressive Sequence Models
- Markov Models and Hidden Markov Models
- State Space Models and Latent Dynamical Systems
- Dynamic Bayesian Networks
- Temporal Point Processes
- Continuous-Time Dynamical Models
- Bayesian and Nonparametric Time-Series Models
- Nonstationary and Regime-Switching Models
- Inference and Learning for Temporal Probabilistic Models

Uncertainty Quantification and Calibration
- Theoretical Foundations of Uncertainty and Calibration
- Bayesian Deep Learning
- Practical Uncertainty Estimation Methods
- Aleatoric and Epistemic Uncertainty Modeling
- Calibration of Probabilistic Predictions
- Conformal Prediction
- Out-of-Distribution Detection and Distribution Shift
- Selective Prediction and Abstention
- Evaluation Metrics, Benchmarks, and Empirical Studies
- Applied Uncertainty-Aware Machine Learning

Structure Learning
- Bayesian Network Structure Learning
- Undirected Graphical Model Structure Learning
- Causal Structure Learning
- Latent Structure Discovery
- Temporal and Dynamic Structure Learning
- Interventional and Active Structure Learning
- Differentiable and Neural Structure Learning
- Scalable and High-Dimensional Structure Learning
- Structure Learning Theory
- Benchmarking and Empirical Evaluation
- Applied Structure Learning
- Surveys and Tutorials

Probabilistic Programming
- Probabilistic Programming Languages and Semantics
- Inference Algorithms for Probabilistic Programs
- Variational and Amortized Inference for Probabilistic Programs
- Compilation and Systems for Probabilistic Programming
- Program Analysis, Verification, and Type Systems
- Deep Probabilistic Programming
- Probabilistic Program Induction and Synthesis
- Benchmarking, Diagnostics, and Empirical Evaluation
- Applied Probabilistic Programming
- Surveys, Tutorials, and Comparative Studies

Theory and Foundations of Probabilistic Modeling
- Statistical Learning Theory and Asymptotics
- Bayesian Foundations and Posterior Theory
- Inference Theory and Approximation
- Identifiability and Latent Variable Theory
- Expressivity and Representation Theory
- Learning Objectives and Divergence Theory
- Information-Theoretic Foundations
- Optimization Theory for Probabilistic Models
- Graphical Structure and Conditional Independence Theory
- Uncertainty Quantification and Calibration
- Robustness and Model Misspecification
- Evaluation Methodology and Benchmarking

Evaluation, Benchmarks, Surveys, and
- Applications of Probabilistic Models
- Scoring Rules and Predictive Performance
- Calibration and Uncertainty Assessment
- Posterior Diagnostics and Inference Evaluation
- Robustness and Distribution Shift Evaluation
- Benchmark Datasets, Tasks, and Evaluation Protocols
- Comparative Empirical Studies and Reproducibility
- Survey, Review, and Taxonomy Papers
- Applications in Vision, Speech, and Multimodal Perception
- Applications in Natural Language, Information Retrieval, and Recommendation
- Applications in Time Series, Control, Healthcare, and Scientific Modeling

quantum_computing

Quantum algorithms for machine learning
- Quantum linear-algebraic methods
- Quantum kernel methods
- Variational quantum circuits and quantum neural networks
- Quantum generative models
- Quantum reinforcement learning
- Quantum data encoding and feature maps
- Optimization and trainability of quantum machine learning models
- Quantum learning theory and complexity
- Noise-aware quantum machine learning
- Benchmarking and empirical evaluation of quantum machine learning
- Applied quantum machine learning
- Surveys and reviews of quantum machine learning

Variational quantum circuits and quantum neural networks
- Quantum circuit ansätze and QNN architectures
- Quantum data encoding and feature maps
- Optimization and training methods
- Expressivity, trainability, and generalization theory
- Noise-aware and hardware-efficient QNNs
- Hybrid quantum-classical neural networks
- Supervised quantum learning
- Generative and unsupervised quantum learning
- Quantum reinforcement learning
- Benchmarking and empirical evaluation
- Application domains and case studies
- Surveys and taxonomies of VQCs and QNNs

Quantum kernel methods
- Quantum feature maps
- Quantum kernel estimation
- Kernelized quantum learning algorithms
- Trainable quantum kernels
- Expressivity and inductive bias
- Generalization and learning theory
- Quantum advantage and complexity analysis
- Noise robustness and error mitigation
- Scalable and approximate quantum kernels
- Benchmarking and empirical evaluation
- Applications of quantum kernel methods
- Surveys and reviews of quantum kernel methods

Quantum data encoding and representation learning
- Classical-to-quantum data encoding
- Quantum feature maps and kernel embeddings
- Variational quantum embeddings
- Quantum state representation learning
- Quantum autoencoders and dimensionality reduction
- Theory of quantum representations
- Noise-aware and hardware-efficient encoding
- Benchmarking and evaluation
- Application-specific quantum representations
- Surveys and taxonomies

Quantum generative models
- Quantum circuit Born machines
- Quantum generative adversarial networks
- Quantum Boltzmann machines and energy-based models
- Quantum autoencoders and latent-variable models
- Autoregressive quantum generative models
- Quantum diffusion and score-based models
- Theory of quantum generative modeling
- Training and optimization methods
- Noise, hardware, and implementation
- Benchmarking and evaluation
- Applications of quantum generative models
- Surveys and comparative reviews

Quantum reinforcement learning
- Quantum value-based reinforcement learning
- Quantum policy gradient and actor-critic methods
- Model-based quantum reinforcement learning
- Hybrid quantum-classical reinforcement learning
- Quantum state representation and action encoding

- Quantum environments and quantum decision processes
- Theoretical foundations of quantum reinforcement learning
- Noise-resilient and hardware-aware quantum reinforcement learning
- Quantum control and quantum error correction with reinforcement learning
- Benchmarking and empirical evaluation in quantum reinforcement learning
- Surveys and taxonomies of quantum reinforcement learning

Quantum optimization and quantum annealing for learning
- Ising and QUBO Formulations for Learning
- Annealing-Based Training and Optimization of ML Models
- Annealing-Based Inference and Sampling
- Feature Selection, Structure Learning, and Model Selection
- Hybrid Quantum-Classical Annealing Methods
- Hardware-Aware Embedding and Compilation for Learning
- Theory and Complexity of Annealing-Based Learning
- Noise, Robustness, and Error Mitigation in Annealing-Based Learning
- Benchmarking and Comparative Evaluation
- Applied Quantum Optimization for Domain-Specific Learning

Hybrid quantum-classical learning
- Quantum neural networks and hybrid architectures
- Quantum kernel methods
- Generative and unsupervised learning
- Quantum reinforcement learning
- Data encoding and circuit architecture
- Optimization, trainability, and learning theory
- Noise-aware and hardware-efficient learning
- Benchmarking and quantum advantage
- Application-driven hybrid quantum-classical learning
- Surveys and reviews

Quantum learning theory and complexity
- Quantum PAC and agnostic learning
- Sample complexity and generalization theory
- Query complexity and oracle learning
- Computational complexity and hardness of learning
- Quantum–classical separations and dequantization
- Expressivity and representational power
- Trainability and optimization complexity
- Learning quantum states, channels, and Hamiltonians

Trainability, expressivity, and generalization of QML
- Barren plateaus and concentration phenomena
- Optimization landscapes and variational dynamics
- Ansatz, architecture, and initialization design
- Expressivity and approximation theory
- Quantum data embeddings, feature maps, and kernels
- Generalization, learnability, and sample complexity
- Noise, robustness, and hardware-aware effects
- Expressivity-trainability-generalization trade-offs
- Empirical benchmarks and comparative evaluations
- Surveys and unifying theoretical frameworks

Noise-resilient and hardware-aware quantum machine learning
- Noise modeling and characterization in quantum machine learning
- Theoretical analysis of noisy quantum learning systems
- Error mitigation and error suppression for quantum machine learning
- Noise-aware optimization and robust training of variational quantum models
- Hardware-efficient ansatz and model architecture design
- Quantum compilation and execution optimization for QML
- Device-aware co-design and resource estimation for QML
- Benchmarking and empirical evaluation of noisy QML
- Applications and experimental demonstrations on near-term quantum hardware
- Surveys and comparative studies of noise-resilient and hardware-aware QML

Benchmarking and empirical evaluation of QML
- Benchmark datasets and task suites
- Comparative studies against classical baselines
- Resource and complexity evaluation
- Scalability analysis
- Noise and hardware-aware benchmarking
- Trainability and optimization evaluation
- Generalization and robustness assessment
- Application-oriented benchmarking
- Evaluation protocols, reproducibility, and statistical methodology
- Survey and meta-analysis of empirical QML

Applied quantum machine learning
- Quantum supervised learning
- Quantum unsupervised learning
- Quantum generative modeling
- Quantum reinforcement learning
- Quantum neural networks and hybrid quantum-classical architectures
- Quantum kernel methods and feature maps
- Trainability, optimization, and learning theory in quantum machine learning
- Noise-aware and hardware-efficient quantum machine learning
- Benchmarking and empirical evaluation of quantum machine learning
- Domain-specific applications of quantum machine learning
- Surveys and comparative studies of quantum machine learning

Quantum-inspired machine learning
- Tensor Network Methods
- Quantum-Inspired Optimization
- Quantum-Inspired Kernel Methods
- Quantum-Inspired Neural Architectures
- Quantum-Inspired Generative and Probabilistic Models
- Quantum-Inspired Linear Algebra and Sampling Algorithms
- Theory and Complexity Analysis
- Benchmarking and Empirical Evaluation
- Applied Quantum-Inspired Machine Learning
- Surveys, Taxonomies, and Methodology

Surveys and taxonomies of quantum machine learning
- Comprehensive overviews and taxonomies of quantum machine learning
- Surveys of variational quantum circuits and hybrid quantum-classical learning
- Surveys of quantum kernel methods and feature maps
- Surveys of quantum neural networks and deep quantum models
- Surveys of quantum generative models and probabilistic learning
- Surveys of quantum reinforcement learning
- Surveys of data encoding, state preparation, and quantum representations
- Surveys of learning theory, expressivity, trainability, and generalization in QML
- Surveys of noise, robustness, error mitigation, and fault tolerance for QML
- Benchmarking and empirical evaluation surveys for QML
- Application-oriented surveys of quantum machine learning
- Surveys of software frameworks, hardware platforms, and implementation workflows
- Systematic reviews, bibliometric studies, and survey methodology for QML

reinforcement_learning

Value-Based Reinforcement Learning
- Policy Evaluation and Value Prediction
- Value-Based Control Algorithms
- Deep Value Function Approximation
- Distributional Reinforcement Learning
- Exploration and Uncertainty Estimation
- Offline and Off-Policy Value Learning
- Model-Based Value Learning and Planning
- Representation Learning and Generalization
- Hierarchical Value-Based Reinforcement Learning
- Multi-Agent Value-Based Reinforcement Learning
- Robust and Safe Value-Based Reinforcement Learning
- Theoretical Analysis of Value-Based RL
- Benchmarking and Empirical Evaluation
- Applications of Value-Based Reinforcement Learning
- Surveys and Reviews of Value-Based Reinforcement Learning

Policy Gradient and Actor-Critic Methods
- Policy Gradient Theory and Convergence
- Stochastic Policy Gradient Methods
- On-Policy Actor-Critic Methods
- Off-Policy Actor-Critic Methods
- Deterministic Policy Gradient Methods
- Natural Policy Gradient Methods
- Trust Region and Proximal Policy Optimization
- Entropy-Regularized and Maximum-Entropy Methods
- Variance Reduction and Credit Assignment
- Safe and Constrained Policy Optimization
- Model-Based Actor-Critic
- Offline Actor-Critic and Policy Optimization
- Multi-Agent Actor-Critic
- Hierarchical Actor-Critic
- Benchmarking and Empirical Evaluation
- Applications of Policy Gradient and Actor-Critic Methods

Model-Based Reinforcement Learning
- Environment Model Learning
- Latent Dynamics and World Models
- Planning and Model Predictive Control
- Dyna-Style Model-Based Reinforcement Learning
- Exploration in Model-Based Reinforcement Learning

- Uncertainty-Aware and Robust Model-Based RL
- Offline Model-Based Reinforcement Learning
- Theoretical Foundations of Model-Based RL
- Benchmarking and Empirical Evaluation for MBRL
- Applications of Model-Based Reinforcement Learning
- Surveys and Taxonomies of Model-Based RL

Offline Reinforcement Learning
- Value-Based Offline Reinforcement Learning
- Policy-Constrained and Regularized Offline Reinforcement Learning
- Model-Based Offline Reinforcement Learning
- Sequence Modeling and Supervised Offline Reinforcement Learning
- Off-Policy Evaluation and Policy Selection
- Theory and Generalization in Offline Reinforcement Learning
- Representation Learning for Offline Reinforcement Learning
- Dataset Quality and Curation for Offline Reinforcement Learning
- Safe and Robust Offline Reinforcement Learning
- Offline-to-Online Reinforcement Learning
- Datasets, Benchmarks, and Empirical Evaluation
- Applications of Offline Reinforcement Learning
- Surveys and Taxonomies of Offline Reinforcement Learning

Imitation Learning and Inverse
- Reinforcement Learning
- Behavior Cloning
- Interactive and Active Imitation Learning
- Offline Imitation Learning
- Reinforcement Learning from Demonstrations
- Inverse Reinforcement Learning
- Adversarial Imitation Learning
- Preference-Based Reward Learning
- Imitation from Observation
- Hierarchical Imitation Learning
- Transfer and Meta-Imitation Learning
- Multi-Agent Imitation Learning and IRL
- Theory and Foundations of IL/IRL
- Robustness and Safety in IL/IRL
- Evaluation and Benchmarking for IL/IRL
- Applications of IL/IRL

Hierarchical Reinforcement Learning
- Temporal Abstraction and Options
- Goal-Conditioned, Subgoal-Based, and Feudal HRL
- Skill Discovery and Unsupervised Hierarchy Learning
- Model-Based and Planning-Centric HRL
- Representation Learning for Hierarchical Control
- Exploration and Intrinsic Motivation in HRL
- Learning and Optimization Methods for HRL
- Theory and Foundations of Hierarchical Reinforcement Learning
- Benchmarks and Empirical Evaluation for Hierarchical Reinforcement Learning
- Applications of Hierarchical Reinforcement Learning
- Surveys and Taxonomies of Hierarchical Reinforcement Learning

Multi-Agent Reinforcement Learning
- Cooperative Multi-Agent Reinforcement Learning
- Competitive Multi-Agent Reinforcement Learning
- General-Sum Multi-Agent Reinforcement Learning
- Communication and Coordination in Multi-Agent Reinforcement Learning
- Credit Assignment and Value Decomposition
- Opponent Modeling and Recursive Reasoning
- Large-Scale Multi-Agent Reinforcement Learning
- Algorithmic Frameworks for Multi-Agent Reinforcement Learning
- Game-Theoretic and Theoretical Foundations of Multi-Agent Reinforcement Learning
- Benchmarking and Evaluation in Multi-Agent Reinforcement Learning
- Applied Multi-Agent Reinforcement Learning
- Surveys and Empirical Studies of Multi-Agent Reinforcement Learning

Exploration and Intrinsic Motivation
- Count-Based and Pseudocount Exploration
- Uncertainty-Driven Exploration
- Curiosity-Driven Intrinsic Motivation
- Information-Theoretic Exploration
- Entropy-Regularized Exploration
- Skill Discovery and Empowerment
- Goal-Conditioned and Hierarchical Exploration
- Safe Exploration
- Theoretical Analysis of Exploration
- Exploration Benchmarks and Evaluation
- Embodied and Real-World Exploration

Safe and Constrained Reinforcement Learning
- Foundations and Theoretical Guarantees
- Constrained Markov Decision Processes and Constrained Policy Optimization
- Risk-Sensitive and Chance-Constrained Reinforcement Learning
- Safe Exploration
- Model-Based Safe Reinforcement Learning
- Robust and Adversarial Safe Reinforcement Learning
- Shielding, Runtime Safety Filters, and Formal Verification
- Safe Policy Improvement and Offline Safe Reinforcement Learning
- Safe Reinforcement Learning Benchmarks and Evaluation
- Safety-Critical Applications of Reinforcement Learning
- Surveys and Taxonomies of Safe Reinforcement Learning

Meta-Reinforcement Learning and Transfer
- Fast Adaptation and Few-Shot Reinforcement Learning
- Multi-Task Reinforcement Learning
- Cross-Task Transfer Learning in Reinforcement Learning
- Continual and Lifelong Reinforcement Learning
- Skill Transfer and Policy Reuse
- Transferable Representation Learning
- Domain Adaptation and Sim-to-Real Transfer
- Theoretical Foundations of Meta- and Transfer Reinforcement Learning
- Benchmarking and Evaluation for Meta- and Transfer Reinforcement Learning
- Applications of Meta- and Transfer Reinforcement Learning
- Surveys and Taxonomies of Meta- and Transfer Reinforcement Learning

Representation Learning for Reinforcement Learning
- Predictive and Reconstruction-Based Representation Learning
- Contrastive Representation Learning
- Latent Dynamics and World Models
- State Abstraction and Bisimulation
- Successor Representations and Successor Features
- Disentangled, Object-Centric, and Compositional Representations
- Action, Skill, and Option Representations
- Pretraining, Transfer, and Multi-Task Representation Learning
- Belief-State and Memory Representations
- Theoretical Foundations of RL Representations
- Benchmarks and Evaluation of RL Representations
- Applications in Visual, Robotic, and Embodied Reinforcement Learning
- Surveys and Taxonomies of Representation Learning for RL

Reinforcement Learning Theory
- MDP and Dynamic Programming Theory
- Sample Complexity and Regret Analysis
- Exploration Theory
- Convergence and Stochastic Approximation Theory
- Function Approximation, Representation, and Generalization Theory
- Value-Based Reinforcement Learning Theory
- Policy Optimization Theory
- Model-Based Reinforcement Learning Theory
- Offline and Batch Reinforcement Learning Theory
- Partial Observability and Latent-State Theory
- Hierarchical Reinforcement Learning and Temporal Abstraction Theory
- Multi-Agent and Game-Theoretic RL Theory
- Safe and Robust Reinforcement Learning Theory
- Computational Complexity and Lower Bounds
- Evaluation Methodology and Theoretical Benchmarks
- Surveys and Unifying Frameworks

Benchmarking and Evaluation in Reinforcement Learning
- Benchmark Environment and Suite Design
- Evaluation Metrics and Experimental Protocols
- Statistical Evaluation and Reproducibility
- Sample Efficiency and Computational Efficiency Evaluation
- Generalization and Transfer Evaluation
- Robustness Evaluation
- Safety and Risk Evaluation
- Off-Policy Evaluation
- Sim-to-Real and Real-World Evaluation
- Diagnostic Evaluation and Ablation Analysis
- Survey and Comparative Studies

Applications of Reinforcement Learning
- Robotics and Embodied Control
- Games and Interactive Environments
- Transportation and Autonomous Navigation
- Recommender Systems, Personalization, and Advertising
- Operations Research and Resource Allocation
- Networking, Systems, and Cybersecurity
- Healthcare and Clinical Decision Making
- Finance and Economic Decision Making
- Energy Systems and Smart Infrastructure
- Natural Language Processing and Dialogue Systems

- Computer Vision and Multimedia Systems
- Industrial Automation and Manufacturing
- Scientific Discovery and Engineering Design
- Human-Computer Interaction and Education
- Benchmarking, Evaluation, and Survey Studies of RL Applications
- Application-specific Reinforcement Learning Methodology and Deployment

Surveys and Empirical Studies in Reinforcement Learning
- Survey and Taxonomy Papers
- Comparative Benchmarking Studies
- Benchmark Suites and Evaluation Environments
- Reproducibility and Replication Studies
- Evaluation Methodology and Statistical Analysis
- Ablation and Sensitivity Analysis
- Generalization and Robustness Studies
- Scaling and Efficiency Studies
- Domain-Specific Surveys and Empirical Reviews
- Meta-Analysis and Research Trend Studies

representation_learning

self_supervised_representation_learning
- contrastive_learning
- non_contrastive_learning
- clustering_and_prototype_based_learning
- masked_and_denoising_modeling
- predictive_and_autoregressive_learning
- multimodal_and_cross_view_learning
- augmentation_and_pretext_task_design
- theoretical_foundations_and_analysis
- evaluation_and_benchmarking
- transfer_learning_and_applications

generative_and_reconstruction_based_representation_learning
- linear_latent_factor_models
- sparse_coding_and_dictionary_learning
- autoencoder_based_representation_learning
- variational_autoencoders_and_probabilistic_latent_variable_models
- adversarial_representation_learning
- autoregressive_representation_learning
- normalizing_flows_and_invertible_representation_learning
- energy_based_and_undirected_generative_models
- diffusion_and_score_based_representation_learning
- disentangled_representation_learning
- hierarchical_and_structured_latent_representation_learning
- multimodal_generative_representation_learning
- reconstruction_based_self_supervised_learning
- theory_and_identifiability_of_generative_representations
- benchmarking_and_empirical_evaluation_of_generative_representations
- application_oriented_generative_representation_learning

metric_learning_and_embedding_learning
- Distance Metric Learning
- Deep Metric Learning
- Contrastive and Self-Supervised Embedding Learning
- Manifold and Structure-Preserving Embedding
- Geometric and Non-Euclidean Embedding Learning
- Multimodal and Cross-Modal Embedding Learning
- Metric Learning Objectives and Mining Strategies
- Metric Learning Theory
- Evaluation and Benchmarking for Metric Learning
- Applied Metric and Embedding Learning
- Surveys and Taxonomies of Metric Learning

disentangled_representation_learning
- Unsupervised Disentangled Representation Learning
- Weakly Supervised Disentangled Representation Learning
- Supervised Disentangled Representation Learning
- Identifiability and Theoretical Foundations of Disentanglement
- Causal Disentangled Representation Learning
- Structured Generative Models for Disentanglement
- Temporal, Sequential, and Multimodal Disentanglement
- Disentanglement Metrics, Benchmarks, and Comparative Evaluation
- Applications of Disentangled Representations

invariant_equivariant_and_causal_representation_learning
- Invariant Representation Learning
- Equivariant Representation Learning
- Symmetry Discovery
- Causal Representation Learning
- Counterfactual and Interventional Representation Learning
- Domain Generalization and Invariant Prediction
- Identifiability and Theoretical Foundations
- Evaluation, Benchmarks, and Empirical Surveys
- Applications of Invariant, Equivariant, and Causal Representations

multiview_and_multimodal_representation_learning
- multiview_subspace_and_correlation_learning
- cross_modal_alignment_and_joint_embedding_learning
- multimodal_fusion_and_integration
- self_supervised_and_contrastive_multimodal_learning
- generative_multimodal_representation_learning
- shared_private_and_disentangled_representation_learning
- incomplete_and_missing_modality_learning
- cross_modal_translation_retrieval_and_transfer
- theoretical_foundations_of_multiview_and_multimodal_learning
- benchmarking_evaluation_and_surveys

geometric_and_structured_representation_learning
- graph_representation_learning
- relational_and_knowledge_graph_representation_learning
- manifold_mesh_and_point_cloud_representation_learning
- equivariant_and_symmetry_aware_representation_learning
- hyperbolic_and_non_euclidean_representation_learning
- topological_representation_learning
- set_and_permutation_invariant_representation_learning
- structured_latent_representation_learning

transferable_and_domain_adaptive_representation_learning
- transfer_learning_and_fine_tuning
- domain_adaptation
- multi_source_and_source_free_domain_adaptation
- domain_generalization
- test_time_and_online_adaptation
- invariant_and_causal_representation_learning
- meta_learning_and_few_shot_transfer
- theoretical_foundations_of_transferability_and_adaptation
- benchmarks_datasets_and_evaluation_protocols
- applications_under_domain_shift
- surveys_and_empirical_studies

theory_of_representation_learning
- information_theoretic_foundations
- geometric_and_manifold_theory
- expressivity_and_approximation_theory
- optimization_dynamics_and_implicit_bias
- generalization_and_sample_complexity
- symmetry_invariance_and_equivariance
- disentanglement_and_identifiability
- self_supervised_and_contrastive_learning_theory
- latent_variable_and_generative_representation_theory
- robustness_and_stability_of_representations
- transferability_and_domain_shift_theory
- causal_representation_learning_theory
- representation_evaluation_and_quality_metrics

representation_evaluation_probing_and_benchmarking
- diagnostic_probing
- intrinsic_representation_metrics
- representational_similarity_analysis
- disentanglement_evaluation
- invariance_equivariance_and_robustness_assessment
- transferability_and_downstream_task_benchmarking
- retrieval_clustering_and_nearest_neighbor_evaluation
- fairness_bias_and_privacy_evaluation
- benchmark_dataset_and_protocol_design
- theoretical_foundations_of_probing_and_representation_evaluation

robotics

Robot Perception
- Object-Centric Perception
- Scene Understanding and Semantic Mapping
- 3D Perception and Spatial Reconstruction
- Localization, Mapping, and SLAM
- Motion Estimation and Tracking
- Multimodal Sensor Fusion
- State Estimation and Proprioceptive Perception
- Tactile, Haptic, and Contact Perception
- Auditory and Acoustic Perception
- Human-Centered Perception
- Active and Embodied Perception
- Representation Learning and Adaptation for Robot Perception
- Probabilistic and Uncertainty-Aware Perception
- Datasets, Benchmarks, and Evaluation

Localization, Mapping, and Navigation
- State Estimation and Sensor Fusion
- Odometry and Ego-Motion Estimation
- Global Localization and Relocalization
- Simultaneous Localization and Mapping (SLAM)
- Place Recognition and Loop Closure Detection
- Map Representation and Environment Reconstruction
- Exploration and Active Mapping
- Navigation and Motion Planning

- Embodied and Visual Navigation
- Datasets, Benchmarks, and Evaluation

Manipulation and Grasping
- Grasp Detection, Synthesis, and Analysis
- Perception and Affordance Learning for Manipulation
- Task and Motion Planning for Manipulation
- Learning-Based Manipulation Policies
- Dexterous and In-Hand Manipulation
- Nonprehensile Manipulation
- Contact-Rich Manipulation and Robotic Assembly
- Deformable Object Manipulation
- Bimanual and Multi-Arm Manipulation
- Tactile Sensing and Haptic Manipulation
- Simulation and Sim-to-Real Transfer for Manipulation
- Datasets, Benchmarks, and Evaluation for Manipulation

Locomotion
- Dynamics Modeling and Contact Mechanics
- Gait Generation and Motion Synthesis
- Locomotion Planning and Trajectory Optimization
- Locomotion Control and Policy Learning
- Perception-Guided and Terrain-Adaptive Locomotion
- Sim-to-Real Transfer and Domain Adaptation
- Robust, Safe, and Energy-Efficient Locomotion
- Legged and Humanoid Locomotion
- Wheeled and Tracked Locomotion
- Aerial Locomotion
- Aquatic and Amphibious Locomotion
- Soft and Bio-inspired Locomotion
- Benchmarks, Datasets, and Evaluation

Motion Planning and Control
- Sampling-Based and Search-Based Planning
- Trajectory Optimization and Optimal Control
- Model Predictive Control
- Reinforcement Learning for Control
- Imitation Learning and Learning from Demonstration
- Safe, Robust, and Constrained Control
- Uncertainty-Aware and Probabilistic Planning
- Multi-Agent Motion Planning and Control
- Hierarchical Planning and Task and Motion Planning
- Adaptive Control and Sim-to-Real Transfer
- Control Theory and Formal Verification
- Benchmarks, Datasets, and Evaluation
- Applications of Motion Planning and Control
- Surveys and Taxonomies

Reinforcement Learning for Robotics
- Theoretical Foundations of Robot Reinforcement Learning
- Model-Free Reinforcement Learning for Robotics
- Model-Based Reinforcement Learning for Robotics
- Offline Reinforcement Learning for Robotics
- Imitation Learning and Reinforcement Learning from Demonstrations
- Hierarchical and Skill-Based Reinforcement Learning for Robotics
- Safe and Constrained Reinforcement Learning for Robotics
- Sim-to-Real Transfer in Robot Reinforcement Learning
- Multi-Task, Transfer, and Meta-Reinforcement Learning for Robotics
- Perception and Representation Learning for Robot Reinforcement Learning
- Multi-Agent Reinforcement Learning for Robotics
- Reinforcement Learning for Robot Manipulation
- Reinforcement Learning for Robot Locomotion and Mobility
- Reinforcement Learning for Robot Navigation and Exploration
- Benchmarking, Empirical Evaluation, and Surveys in Robot Reinforcement Learning

Imitation Learning and Learning from Demonstration
- Behavior Cloning
- Inverse Reinforcement Learning and Apprenticeship Learning
- Adversarial Imitation Learning
- Interactive and Active Imitation Learning
- Reinforcement Learning from Demonstrations
- Learning from Observation and Visual Imitation
- Hierarchical and Skill-based Imitation Learning
- Meta-, Few-shot, and Multi-task Imitation Learning
- Transfer, Domain Adaptation, and Sim-to-real Imitation Learning
- Safety, Robustness, and Generalization in Imitation Learning
- Theoretical Foundations of Imitation Learning
- Benchmarks, Datasets, and Evaluation for Imitation Learning
- Applications of Imitation Learning in Robotics
- Surveys and Taxonomies of Imitation Learning

Self-Supervised and Representation Learning for Robotics
- Visual Representation Learning for Robotics
- Multimodal Representation Learning for Robotics
- Proprioceptive and Tactile Representation Learning
- State Representation Learning for Control and Planning
- World Models and Predictive Representation Learning
- Goal, Skill, and Affordance Representation Learning
- Language-Grounded Representation Learning for Robotics
- Transferable Representations, Domain Adaptation, and Sim-to-Real
- Learning Objectives and Methodologies for Robot Self-Supervision
- Theory and Analysis of Robot Representation Learning
- Evaluation, Benchmarks, and Datasets for Robot Representation Learning
- Applications of Self-Supervised Representation Learning in Robotics
- Surveys and Taxonomies of Robot Self-Supervised Learning

Sim-to-Real Transfer and Domain Adaptation
- Domain Randomization
- Synthetic Data Generation and Sensor Modeling
- Image Translation and Visual Refinement
- Representation Alignment and Domain-Invariant Learning
- System Identification and Dynamics Adaptation
- Policy Transfer and Reinforcement Learning Adaptation
- Meta-Learning and Few-Shot Adaptation
- Online, Continual, and Test-Time Adaptation
- Cross-Embodiment and Morphology Transfer
- Safety, Robustness, and Uncertainty-Aware Transfer
- Theoretical Analysis of Domain Shift and Transfer
- Benchmarks, Datasets, and Evaluation Protocols
- Task-Oriented Sim-to-Real Applications

Human-Robot Interaction
- Human Perception and Behavior Understanding
- Intention, Goal, and Trajectory Prediction
- Natural Language and Multimodal Communication
- Learning from Demonstration, Preferences, and Interactive Feedback
- Shared Autonomy, Teleoperation, and Mixed-Initiative Control
- Human-Robot Collaboration and Joint Task Planning
- Human-Aware Motion Planning and Social Navigation
- Physical Human-Robot Interaction and Assistive Manipulation
- Socially Assistive and Affective Robotics
- Personalization, User Modeling, and Adaptation
- Safety, Trust, Explainability, and Ethics in HRI
- HRI Datasets, Benchmarks, Simulation, and Evaluation

Multi-Robot Systems and Swarm Robotics
- Multi-Agent Reinforcement Learning
- Swarm Intelligence and Collective Behavior
- Decentralized Coordination and Distributed Control
- Formation Control, Flocking, and Consensus
- Task Allocation and Multi-Robot Scheduling
- Multi-Robot Planning, Navigation, and Exploration
- Cooperative Perception, Localization, and Mapping
- Communication and Networked Multi-Robot Systems
- Cooperative Manipulation and Transport
- Human-Swarm Interaction and Mixed Human-Robot Teams
- Safety, Robustness, and Formal Verification
- Simulation, Benchmarks, and Sim-to-Real Transfer

Safe, Robust, and Reliable Robot Learning
- Theoretical Foundations and Safety Guarantees
- Safe Reinforcement Learning and Safe Exploration
- Risk-Aware and Uncertainty-Aware Robot Learning
- Robust Perception, State Estimation, and World Modeling
- Adversarial and Distributionally Robust Robot Learning
- Sim-to-Real Transfer and Domain Generalization
- Safe Motion Planning and Learning-Based Control
- Formal Verification, Validation, and Runtime Assurance
- Fault Detection, Diagnosis, and Fault-Tolerant Learning
- Human-Robot Safety and Shared Autonomy

- Benchmarks, Datasets, and Safety Evaluation
- Robotics Datasets, Simulation, and Benchmarks
  - Dataset Collection and Curation
  - Annotation and Ground-Truth Generation
  - Synthetic Data and Procedural Generation
  - Simulation Environments and Physics-Based Modeling
  - Sim-to-Real Transfer
  - Benchmark Design and Task Suites
  - Evaluation Metrics and Experimental Protocols
  - Dataset Quality, Bias, and Generalization Analysis
  - Reproducibility, Standardization, and Data Infrastructure
  - Surveys and Meta-Analysis of Robotics Resources

scientific_machine_learning

- Physics-informed learning
  - Physics-informed neural networks (PINNs)
  - Physics-informed operator learning
  - Inverse problems and system identification
  - Data assimilation and state estimation
  - Hybrid physics-machine learning models
  - Uncertainty quantification and Bayesian physics-informed learning
  - Theory and numerical analysis of physics-informed learning
  - Benchmarking and empirical evaluation
  - Scientific applications of physics-informed learning
  - Survey and review studies
- Operator learning
  - Neural operator architectures
  - Geometry-aware operator learning
  - Physics-informed operator learning
  - Theoretical foundations of operator learning
  - Probabilistic operator learning
  - Data-efficient and multi-fidelity operator learning
  - Inverse problems and control
  - Empirical evaluation and benchmarks
  - Scientific and engineering applications
- Surrogate modeling and reduced-order modeling
  - Parametric surrogate modeling
  - Operator learning
  - Dynamical systems emulation
  - Projection-based reduced-order modeling
  - Latent-space reduced-order modeling
  - Structure-preserving and physics-constrained reduction
  - Multifidelity and multiscale surrogate modeling
  - Probabilistic surrogate modeling and uncertainty quantification
  - Adaptive sampling and experimental design
  - Error analysis, stability, and verification
  - Benchmarking, comparative evaluation, and review studies
  - Scientific and engineering applications
- Differentiable simulation and hybrid modeling
  - Differentiable numerical solvers
  - Differentiable physics engines
  - Gray-box modeling
  - Learned closure and constitutive modeling
  - System identification and model calibration
  - Data assimilation and state estimation
  - Simulation-based optimization and optimal control
  - Uncertainty quantification and probabilistic modeling
  - Multiscale and multiphysics hybrid modeling
  - Theory and numerical analysis
  - Benchmarks, datasets, and software systems
  - Scientific and engineering applications
  - Surveys and review studies
- Inverse problems and data assimilation
  - Learned reconstruction and unrolled optimization
  - Physics-informed and PDE-constrained inversion
  - Bayesian inverse problems and uncertainty quantification
  - Generative modeling and learned priors for inverse problems
  - Operator learning and surrogate-assisted inversion
  - Sequential and variational data assimilation
  - Observation design and sensor placement
  - Theory, identifiability, and convergence
  - Benchmarks, datasets, and evaluation
  - Scientific and engineering applications
- Scientific discovery and system identification
  - Equation discovery
  - Input-output system identification
  - State-space and latent dynamics inference
  - PDE and operator discovery
  - Causal and mechanistic discovery
  - Parameter estimation and model calibration
  - Optimal experimental design
  - Identifiability, uncertainty quantification, and validation
  - Benchmarks and evaluation
  - Scientific applications
- Uncertainty quantification and probabilistic modeling
  - Bayesian deep learning
  - Approximate Bayesian inference
  - Gaussian processes and kernel probabilistic models
  - Probabilistic graphical models and probabilistic programming
  - Deep generative and latent-variable models
  - Predictive uncertainty estimation and calibration
  - Conformal prediction and distribution-free uncertainty quantification
  - Distribution shift and out-of-distribution uncertainty
  - Uncertainty propagation and probabilistic numerics
  - Bayesian inverse problems and surrogate modeling
  - Evaluation, benchmarking, and uncertainty metrics
- Scientific optimization, control, and experimental design
  - Bayesian optimization
  - Black-box and simulation-based optimization
  - Differentiable and PDE-constrained optimization
  - Inverse design
  - System identification and model calibration
  - Robust, stochastic, and multi-objective optimization
  - Optimal control and model predictive control
  - Reinforcement learning for control
  - Optimal experimental design and adaptive sampling
  - Autonomous experimentation
- Scientific representation learning and foundation models
  - Self-Supervised and Unsupervised Learning for Scientific Data
  - Geometric and Equivariant Representation Learning
  - Graph Representation Learning for Scientific Systems
  - Representation Learning for Scientific Imaging
  - Sequence and Language Foundation Models for Science
  - Spatiotemporal and Field Foundation Models
  - Multimodal Scientific Foundation Models
  - Generative Models for Scientific Discovery and Design
  - Transfer Learning, Domain Adaptation, and Few-Shot Scientific Learning
  - Evaluation, Benchmarks, and Trustworthy Scientific Foundation Models
  - Theory and Scaling Laws for Scientific Representation Learning
- Learned numerical methods and solver acceleration
  - Learned iterative methods for linear systems and eigenproblems
  - Learned preconditioning, multigrid, and domain decomposition
  - Learned nonlinear equation solvers and fixed-point acceleration
  - Learned time integration for ODEs, SDEs, and PDEs
  - Learned discretization, meshing, and adaptive refinement
  - Reduced-order modeling and surrogate-assisted simulation
  - Learned optimization and inverse-problem solvers
  - Learned Monte Carlo, quadrature, and stochastic simulation acceleration
  - Theory, stability, and convergence analysis of learned numerical methods
  - Benchmarks, datasets, and empirical evaluation of learned solvers
  - Scientific applications of learned solver acceleration
  - Surveys, taxonomies, and software frameworks for learned numerical methods
- Theory and numerical analysis
  - Approximation and expressivity theory
  - Generalization and statistical learning theory
  - Optimization theory
  - Numerical analysis of learned solvers
  - Dynamical systems theory
  - PDE and operator-learning theory
  - Inverse problems and identifiability
  - Error estimation and adaptive methods
  - Uncertainty quantification and probabilistic numerics
  - Benchmarking and evaluation methodology
- Benchmarks, datasets, and evaluation
  - Dataset Collection and Curation
  - Data Annotation, Labeling, and Documentation
  - Dataset Quality, Bias, and Auditing
  - Synthetic Data and Simulation Benchmarks
  - Benchmark and Task Design
  - Evaluation Metrics and Scoring Rules
  - Human and Expert Evaluation
  - Experimental Protocols and Statistical Evaluation
  - Reproducibility and Benchmarking Infrastructure
  - Robustness, Generalization, and Distribution-Shift Evaluation

- Fairness, Privacy, and Safety Evaluation
- Meta-evaluation and Benchmark Analysis

Scientific applications and case studies
- Physical sciences
- Chemistry and materials science
- Biology and bioinformatics
- Biomedical and health sciences
- Earth and environmental sciences
- Astronomy and space science
- Engineering and energy systems
- Scientific imaging and instrumentation
- Scientific discovery and hypothesis generation
- Benchmarks, datasets, and challenge tasks
- Validation, reproducibility, and domain-specific methodology
- Surveys and comparative studies

Surveys and reviews
- Methodological surveys
- Theoretical and foundational reviews
- Application-domain surveys
- Data, datasets, and benchmark reviews
- Simulation-integrated and hybrid-model reviews
- Uncertainty quantification and probabilistic reviews
- Interpretability, robustness, and trustworthiness reviews
- Systems, software, and computational infrastructure reviews
- Comparative reviews and meta-analyses
- Bibliometric and trend reviews

systems

Distributed Training Systems
- Data-Parallel Training
- Model and Tensor Parallelism
- Pipeline Parallelism
- Communication-Efficient Training
- Synchronization and Consistency Models
- Decentralized Training
- Distributed Training Frameworks and Runtime Systems
- Scheduling and Resource Management
- Fault Tolerance and Elastic Training
- Benchmarking, Profiling, and Performance Modeling

Inference and Serving Systems
- Inference Runtime Systems
- Model Serving Architectures
- Scheduling, Batching, and SLO Management
- Resource Management, Placement, and Autoscaling
- Distributed and Multi-Tenant Serving
- Edge, Mobile, and Collaborative Inference Serving
- Generative Model and LLM Serving
  - Workload Characterization and Benchmarking
  - Resource Management, Multi-Tenancy, and Quality of Service
  - Decoding and Iterative Generation Optimization
  - Memory, Caching, and State Management
  - Distributed, Disaggregated, and Elastic Serving
  - Runtime, Compiler, and Hardware Co-Design
  - Model Compression and Serving-Aware Optimization
  - Multi-Model, Adapter, and Personalized Serving
  - Multi-Stage and Retrieval-Augmented Serving
  - Reliability, Observability, and Fault Tolerance
  - Security, Privacy, and Safety in Serving
  - Cost and Energy Efficient Serving
- Reliability, Observability, and Serving Operations
- Benchmarking, Measurement, and Systematization

ML Compilers and Runtime Systems
- Intermediate Representations and Lowering
- Graph Transformation and Optimization
- Automatic Differentiation and Training Compilation
- Kernel Generation and Operator Scheduling
  - Kernel Code Generation
    - Intermediate Representations and Lowering
    - Parallelization and Synchronization Code Generation
    - Memory-Hierarchy-Aware Code Generation
    - Vectorization and Tensorization
    - Fused and Specialized Kernel Generation
    - Dynamic-Shape and Shape-Polymorphic Code Generation
    - Sparse and Irregular Kernel Generation
    - Quantized and Low-Precision Kernel Generation
    - Heterogeneous and Accelerator-Specific Code Generation
    - Autotuning and Search-Based Code Generation
    - Correctness, Verification, and Testing
    - Benchmarking and Performance Evaluation
    - Compiler Infrastructure and Runtime Integration
    - Application-Specific Kernel Generation
    - Surveys and Comparative Studies
  - Loop-Nest Transformation and Schedule Optimization
    - Schedule Languages and Intermediate Representations
    - Dependence Analysis and Legality Checking
    - Polyhedral Scheduling and Affine Transformations
    - Locality-Oriented Loop Transformations
    - Parallelization, Vectorization, and Unrolling
    - Memory Hierarchy and Data Movement Optimization
    - Tensorization and Accelerator-Aware Mapping
    - Operator-Specific and Workload-Specific Scheduling
    - Schedule Search and Autotuning
    - Cost Modeling and Performance Prediction
    - Dynamic and Runtime-Adaptive Scheduling
    - Benchmarking and Empirical Evaluation
    - Surveys, Taxonomies, and Methodological Frameworks
  - Operator Fusion and Graph-Level Kernelization
    - Fusion Legality and Dependence Analysis
    - Graph Rewriting and Fusion Intermediate Representations
    - Subgraph Partitioning and Region Formation
    - Fusion Planning, Cost Modeling, and Search
    - Code Generation and Schedule Synthesis for Fused Kernels
    - Memory-, Layout-, and Communication-Aware Fusion
    - Dynamic-Shape, Sparse, and Just-in-Time Fusion
    - Hardware-Aware and Accelerator-Specific Fusion
    - Training-Phase and Model-Specific Fusion
    - Correctness, Numerical Stability, and Verification
    - Benchmarking and Workload Characterization of Fusion Systems
    - Surveys and Taxonomies of Operator Fusion
- Auto-Tuning and Schedule Search
- Cost Modeling and Performance Prediction
- Memory Hierarchy and Data Layout Optimization
- Hardware-Aware Kernel Mapping
  - Spatial Mapping and Parallelization
  - Temporal Mapping and Loop Scheduling
  - Memory Hierarchy Mapping
  - Data Layout and Data Movement Optimization
  - Instruction-Level Mapping and Tensorization
  - Heterogeneous and Accelerator Mapping
  - Sparse and Irregular Mapping
  - Mixed-Precision and Quantized Mapping
  - Analytical Cost Modeling and Formal Optimization
  - Auto-Tuning and Search-Based Mapping
  - Dynamic and Runtime-Adaptive Mapping
  - Energy- and Thermal-Aware Mapping
  - Portable and Retargetable Mapping Frameworks
  - Operator- and Workload-Specific Mapping
  - Benchmarking and Empirical Evaluation
  - Surveys and Taxonomies
- Runtime Operator Scheduling and Execution Planning
  - DAG-Based Task Scheduling
  - Memory-Aware Execution Planning
  - Heterogeneous Device Placement and Scheduling
  - Asynchronous and Stream Scheduling
  - Dynamic Shape and Control-Flow Execution
  - Distributed Execution and Communication Scheduling
  - Runtime Kernel Selection and Specialization
  - Adaptive Cost Modeling and Online Scheduling
  - Latency-, Throughput-, and QoS-Aware Scheduling
  - Benchmarking, Profiling, and Trace Analysis
  - Survey and Comparative Studies of Execution Runtimes
- Specialized Kernel Generation for Irregular Workloads
  - Sparse Tensor and Structured Sparsity Kernel Generation
  - Graph and Message-Passing Kernel Generation

- Indexing, Gather/Scatter, and Segmented Reduction Kernels
- Dynamic-Shape and Ragged Tensor Kernel Generation
- Data-Dependent Control Flow and Conditional Computation Kernels
- Compiler Analyses and Intermediate Representations for Irregular Kernels
- Runtime Specialization and Just-in-Time Kernel Generation
- Load Balancing and Work Partitioning for Irregular Parallelism
- Memory Layout, Locality, and Data Movement Optimization
- Autotuning, Cost Modeling, and Schedule Search
- Hardware-Specific and Heterogeneous Code Generation
- Benchmarking, Workload Characterization, and Taxonomy
- Benchmarking, Profiling, and Empirical Evaluation
- Auto-Tuning and Cost Modeling
- Memory Planning and Data Layout Optimization
- Runtime Scheduling and Execution Engines
- Distributed Execution and Parallelization
- Heterogeneous and Accelerator-Aware Compilation
- Dynamic, Sparse, and Irregular Workloads
- Inference Serving and Deployment Runtimes
- Compiler Correctness, Testing, and Verification
- Benchmarking and Performance Evaluation

Hardware Acceleration and Co-Design
- Accelerator Architecture Design
- Memory, Dataflow, and Interconnect Design
- Compiler, Mapping, and Runtime Systems
- Algorithm–Hardware Co-Design
- Low-Precision and Sparse Computing
  - Numerical Formats and Low-Precision Arithmetic
    - Fixed-Point and Integer Arithmetic
    - Reduced-Precision Floating-Point Formats
    - Block Floating-Point and Shared-Exponent Representations
    - Logarithmic, Posit, and Other Alternative Number Systems
    - Binary, Ternary, and Ultra-Low-Bit Computation
    - Mixed-Precision and Adaptive Precision Methods
    - Low-Precision Training and Backpropagation Arithmetic
    - Rounding, Accumulation, and Error-Compensation Methods
    - Approximate Arithmetic Units and Operators
    - Numerical Analysis and Error Modeling
    - Benchmarking and Workload-Specific Format Evaluation
  - Neural Network Quantization
    - Quantization Theory and Optimization
    - Post-Training Quantization
    - Quantization-Aware Training
    - Ultra-Low-Bit Quantization
    - Mixed-Precision Quantization
    - Dynamic and Adaptive Quantization
    - Low-Precision Training
    - Hardware-Aware Quantization and Co-Design
    - Robustness and Reliability of Quantized Networks
    - Benchmarking and Evaluation of Quantized Models
    - Architecture- and Application-Specific Quantization
    - Surveys and Empirical Studies of Quantization
  - Mixed-Precision Optimization
  - Binary and Ternary Neural Networks
  - Pruning and Weight Sparsification
  - Structured and Semi-Structured Sparsity
  - Dynamic Sparsity and Sparse Training
  - Activation Sparsity and Conditional Computation
  - Sparse Tensor Representations and Compression
  - Hardware Architectures and Dataflows for Low-Precision and Sparse Computing
  - Compiler, Kernel, and Runtime Support
    - Intermediate Representations and Graph Lowering
    - Quantization-Aware Compilation
    - Sparsity-Aware Compilation
    - Operator Fusion and Graph Rewriting
    - Code Generation and Auto-Tuning
    - Low-Precision Kernel Design
    - Sparse Tensor Kernel Design
    - Memory Layout and Data Movement Optimization
    - Runtime Scheduling and Execution Management
    - Adaptive Precision and Dynamic Sparsity Runtime
    - Portability and Backend Integration
    - Benchmarking, Profiling, and Performance Evaluation
  - Theory, Benchmarking, and Evaluation
- Training Acceleration
- Inference Acceleration and Serving
- Edge, Mobile, and Embedded ML Hardware
- Heterogeneous and Multi-Accelerator Systems
- Emerging ML Hardware
- Benchmarking, Simulation, and Performance Modeling
- Reliability, Security, and Power/Thermal Management

Resource-Efficient ML Systems
- Model Compression and Sparsity
- Quantization and Low-Precision Computing
  - Quantization Theory and Numerical Analysis
  - Post-Training Quantization
  - Quantization-Aware Training
  - Mixed-Precision and Adaptive Bit-Width Methods
  - Binary, Ternary, and Ultra-Low-Bit Networks
  - Low-Precision Training and Optimization
  - Hardware-Aware Quantization and Accelerator Co-Design
  - Benchmarking, Evaluation, and Quantization Toolchains
  - Application-Specific Quantization and Deployment
- Efficient Neural Architecture Design
- Adaptive and Conditional Computation
- Memory-Efficient Training, Adaptation, and Fine-Tuning
- Communication-Efficient Distributed and Federated Learning
- Compiler, Runtime, and Serving Optimization
- Hardware-Software Co-Design for ML
- Edge and On-Device Machine Learning
- Energy-Efficient and Sustainable ML
- Efficiency Benchmarking and Evaluation
- Theoretical Foundations of Resource-Constrained Learning

Data and Storage Systems for ML
- Data Pipeline Systems for ML
- Dataset Storage Layouts and File Formats
- Distributed and Cloud Storage for ML
- Input Pipeline Optimization
- Dataset Versioning, Lineage, and Metadata Management
- Feature Stores and Feature Management
- Data Labeling and Annotation Systems
- Data Quality, Validation, and Observability
- Privacy, Security, and Governance for ML Data
- Embedding and Vector Data Management
- Benchmarking and Workload Characterization of ML Data Systems

Edge, Mobile, and Embedded ML Systems
- Efficient Model Design and Compression
- Hardware-Aware Neural Architecture Search and Co-Design
- Inference Optimization, Compilers, and Runtimes
- Embedded ML Hardware and Accelerators
- On-Device Training, Adaptation, and Continual Learning
- Distributed Edge Intelligence and Edge–Cloud Collaboration
- Federated and Privacy-Preserving Edge Learning
- Resource Management and Quality-of-Service for Edge ML
- Security, Safety, and Reliability of Edge ML Systems
- Benchmarks, Evaluation, and Applications of TinyML and Edge ML

Cluster Management and Scheduling for ML
- Scheduling Theory and Performance Modeling
- Workload Characterization and Benchmarking
- Resource Allocation and Admission Control
- Job Scheduling and Queueing Policies
- Placement, Packing, and Gang Scheduling
- Heterogeneous Accelerator Scheduling
- Communication-, Topology-, and Data-Locality-Aware Scheduling
- Multi-Tenant Fairness, Isolation, and QoS Management
- Elasticity, Auto-scaling, and Adaptive Resource Management
- Fault Tolerance, Preemption, and Straggler Mitigation
- ML Pipeline, Workflow, and Experiment Scheduling
- Inference Serving and Online Scheduling
- Energy- and Cost-Aware Cluster Management

Reliability, Monitoring, and MLOps Systems
- ML Pipeline Orchestration and Workflow Management
- Data Validation and Feature Management
- Experiment Management and Reproducibility
- Model and Data Versioning, Registry, and Lineage

- Continuous Integration, Delivery, and Training for ML
- Model Deployment and Serving Systems
- Monitoring, Observability, and Alerting
- Drift Detection and Post-deployment Model Evaluation
- Testing, Verification, and Validation of ML Systems
- Failure Diagnosis, Debugging, and Incident Response
- Reliability Engineering and Fault Tolerance for ML Systems
- Governance, Auditability, and Compliance in MLOps
- Benchmarking and Evaluation of MLOps Systems

Benchmarking and Performance Evaluation
- Benchmark Suite Design and Workload Characterization
- Performance Metrics and Evaluation Methodology
- Training System Benchmarking
- Inference and Serving Benchmarking
- Distributed and Scalability Evaluation
- Hardware Accelerator and System Stack Evaluation
- Edge, Mobile, and Embedded ML Benchmarking
- Energy, Cost, and Sustainability Evaluation
- Reproducibility, Standardization, and Benchmark Validation

theory

Generalization and Statistical Learning Theory
- PAC Learning and Sample Complexity
- Capacity Control and Uniform Convergence
- PAC-Bayesian Generalization Theory
- Information-Theoretic and Compression-Based Generalization
- Algorithmic Stability and Robustness
- Margin-, Norm-, and Flatness-Based Analyses
- Optimization-Dependent Generalization and Implicit Regularization
- Overparameterization, Interpolation, and Double Descent
- Distribution Shift and Out-of-Distribution Generalization
- Neural Network and Representation Generalization
- Empirical Generalization Studies and Benchmarks
- Surveys and Unifying Perspectives

Optimization Theory
- Deterministic First-Order Optimization
- Stochastic Optimization
- Accelerated, Momentum, and Adaptive Methods
- Second-Order and Curvature-Aware Optimization
- Convex, Composite, and Proximal Optimization
- Nonconvex Optimization
- Optimization Landscapes and Critical Point Analysis
- Gradient Flow and Training Dynamics
- Saddle-Point and Min-Max Optimization
- Constrained and Regularized Optimization
- Distributed, Parallel, and Federated Optimization
- Online, Bandit, and Zeroth-Order Optimization
- Bilevel and Hyperparameter Optimization
- Implicit Bias and Optimization-Induced Generalization
- Convergence, Complexity, and Lower Bounds
- Optimization for Deep Network Training

Approximation Theory and Expressivity
- Universal Approximation Theorems
- Approximation Rates and Optimality
- Depth Separation and Size Trade-offs
- Architecture-Specific Expressivity
- Structured Function and Operator Approximation
- Quantitative Expressivity Measures
- Expressivity under Constraints
- Empirical Expressivity Evaluation and Benchmarks
- Constructive Approximation and Network Design
- Surveys and Unifying Frameworks

Training Dynamics
- Convergence and Stability Analysis
- Stochastic Optimization Dynamics
- Initialization and Early-Time Dynamics
- Scaling Limits and Analytical Regimes
- Feature and Representation Dynamics
- Implicit Bias and Regularization
- Memorization and Generalization Dynamics
- Loss Landscape and Curvature Dynamics
- Architecture and Normalization Effects
- Data Order and Curriculum Effects
- Hyperparameter and Schedule Effects
- Diagnostics and Measurement Methodology
- Empirical Characterization and Benchmarking
- Dynamics-Informed Training Methods

Implicit Bias and Regularization
- Optimization-Induced Implicit Bias
- Stochastic Optimization and Noise-Based Regularization
- Early Stopping and Iterative Regularization
- Norm-Based and Penalty-Based Regularization
- Structured Regularization
- Normalization and Reparameterization Effects
- Data Augmentation and Invariance-Based Regularization
- Margin, Flatness, and Minimum-Norm Bias
- Overparameterization, Interpolation, and Benign Overfitting
- Generalization Bounds and Complexity Measures
- Empirical Characterization and Benchmarking
- Surveys and Unifying Frameworks

Representation Learning Theory
- Information-Theoretic Foundations
- Representation Geometry and Manifold Theory
- Expressivity and Hierarchical Feature Learning
- Disentanglement and Identifiability
- Symmetry, Invariance, and Equivariance
- Self-Supervised and Contrastive Learning Theory
- Latent Variable, Autoencoder, and Generative Representation Models
- Optimization Dynamics and Implicit Bias in Feature Learning
- Generalization, Transfer, and Sample Complexity
- Causal and Invariant Representation Learning

Architecture and Inductive Bias Theory
- Expressivity and Approximation Theory
- Depth, Width, and Parameterization Trade-offs
- Generalization and Sample Complexity of Architectures
- Optimization, Trainability, and Signal Propagation
- Implicit Bias and Architectural Regularization
- Symmetry, Invariance, and Equivariance
- Compositionality, Hierarchy, and Modularity
- Locality, Sparsity, and Structured Connectivity
- Spectral and Frequency Bias
- Theory of Specific Architecture Families
- Domain-Specific Architectural Priors
- Empirical Probing and Benchmarking of Inductive Bias
- Architecture Design Principles and Neural Architecture Search
- Surveys, Taxonomies, and Unifying Frameworks

Robustness and Stability Theory
- Adversarial Robustness
- Certified Robustness and Formal Verification
- Distribution Shift Robustness
- Distributionally Robust Optimization
- Algorithmic Stability and Generalization
- Sensitivity and Perturbation Analysis
- Optimization and Training Stability
- Robust Statistics and Corruption Models
- Robustness Evaluation and Benchmarks

Information-Theoretic Learning Theory
- Information-Theoretic Generalization
- Information-Theoretic Lower Bounds and Sample Complexity
- Information Bottleneck
- Rate-Distortion Theory for Learning
- Minimum Description Length and Compression-Based Learning
- Entropy and Mutual Information Estimation
- Privacy, Memorization, and Information Leakage
- Communication-Constrained and Distributed Learning
- Adaptive Data Analysis
- Information-Theoretic Sequential Decision Making

Computational Complexity of Learning
- Time Complexity of Learning
- Space Complexity and Streaming Learning
- Optimization Complexity
- Query Complexity and Active Learning
- Communication Complexity in Learning
- Computational Hardness and Lower Bounds
- Statistical-Computational Trade-offs
- Average-Case, Smoothed, and Fine-Grained Complexity
- Online Learning and Mistake-Bound Complexity
- Learnability of Specific Hypothesis Classes
- Empirical Scaling Laws and Complexity Benchmarking
- Complexity Measures and Analytical Frameworks

High-Dimensional Learning Theory
- Generalization and Sample Complexity
- Minimax, Information-Theoretic, and Statistical-Computational Limits
- Concentration of Measure and Empirical Process Theory
- Sparse and Structured Estimation
- High-Dimensional Inference and Uncertainty Quantification
- Random Matrix Theory and Spectral Analysis
- Dimension Reduction, Random Projections, and Sketching
- Optimization Dynamics and Landscapes in High Dimensions
- Implicit Bias and Regularization

Overparameterization, Interpolation, and Double Descent
Robust High-Dimensional Statistics and Distribution Shift
Probabilistic and Bayesian Theory
Bayesian Neural Networks
Variational Inference
Monte Carlo and Sampling Methods
Probabilistic Graphical Models
Latent Variable Models
Hierarchical and Nonparametric Bayesian Methods
PAC-Bayesian Theory
Uncertainty Quantification and Calibration
Model Evidence and Bayesian Model Selection