ODAR: Principled Adaptive Routing for LLM Reasoning via Active Inference

Anonymous ACL submission

Abstract

Large language model (LLM) reasoning is evolving from parameter scaling to test-time compute scaling. However, mainstream methods such as fixed best-of-N and self-consistency adopt uniform sampling, leading to high overhead, poor interpretability and diminishing returns. We present ODAR, an adaptive routing framework that optimizes the accuracy–efficiency trade-off via difficulty-aware resource allocation and risk-sensitive heterogeneous fusion. Our framework leverages a lightweight difficulty estimator to dynamically assign queries to fast heuristic or slow deliberative agents. We further design an aligned fusion score integrating character-normalized log-likelihood, varentropy-based penalty and model-wise normalization for reliable cross-model candidate selection. Evaluated on 23 benchmarks, ODAR achieves state-of-the-art performance, reaching 98.2% on MATH and 54.8% on HLE, and improves the compute–accuracy frontier under matched computation budgets. Our open-source implementation based on Llama 4 and DeepSeek outperforms vanilla sampling with an 82% reduction in computation cost. Additional experiments verify the fusion score's superiority in prediction accuracy and confidence calibration. Our findings demonstrate that adaptive compute scheduling and calibrated risk-aware fusion are key to effective test-time scaling.

1 Introduction

Large language models (LLMs) have achieved high reliability on hard problems primarily through *test-time compute* scaling, utilizing methods such as Chain-of-Thought (CoT) deliberation (Wei et al., 2022), multi-sample aggregation via Self-Consistency (Wang et al., 2022) or Best-of-N (Aminian et al., 2025), and verifier-based selection (Lightman et al., 2023; Lifshitz et al., 2025). However, these approaches expose a fundamental inefficiency: the marginal value of additional compute is highly query-dependent. Easy inputs rarely require expensive reasoning and can even degrade under excessive deliberation ("overthinking") (Wei et al., 2025), whereas genuinely difficult inputs benefit disproportionately from deeper search and explicit verification (Yang et al., 2025). This mismatch has fueled a recent wave of *routing/orchestration* research aimed at achieving a favorable accuracy–cost frontier at scale (McGill and Perona, 2017; Pan et al., 2025; Su et al., 2025; Guo et al., 2025; Liu et al., 2025; Chen et al., 2025; Wang et al., 2025). Nevertheless, existing methods are often either fixed-strategy or rely on heuristic fusion rules that are hard to attribute, failing to provide an end-to-end procedure that is simultaneously cost-aware and theoretically grounded.

In this work, we propose **ODAR**, an *open-domain adaptive reasoner* that treats inference as *resource allocation*. For each query, a lightweight **Difficulty Estimator (DE)** predicts whether the instance warrants additional "System 2" compute and routes it to either a **Simple Path** or a **Hard Path** using specialized **Fast** and **Slow Agents**. This specialization is inspired by a functional neuro-computational motif: under high cognitive demand, slower control rhythms can gate burst-like recruitment of higher-cost processing, echoing the θ–γ phase–amplitude coupling mechanisms in working-memory control (Daume et al., 2024; Tamura et al., 2017; Gorochowski et al., 2016; Segneri et al., 2020). Candidate answers are then selected by a **principled fusion module** based on free-energy minimization (FEP), enabling robust comparison across heterogeneous generators without relying on ad-hoc voting.

As a result, ODAR establishes a new performance ceiling across a diverse array of tasks. As illustrated in **Figure 1**, our approach consistently pushes the boundaries of current reasoning models, particularly in domains requiring deep logical synthesis where traditional uniform sampling methods

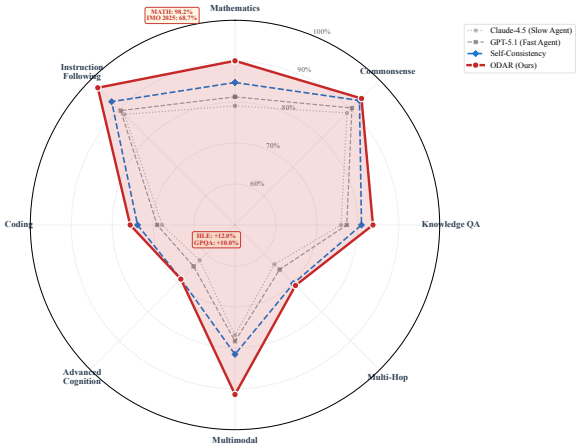

Figure 1: **Performance comparison across 23 benchmarks.** ODAR improves performance across task categories, especially in mathematics and advanced cognition, while remaining cost-efficient.

plateau.

Contributions. Our main contributions are:

- **State-of-the-art accuracy–efficiency frontier.** ODAR achieves **89.6%** average accuracy across 23 benchmarks, outperforming the strongest inference-time baseline by **+6.0%**. As shown in Figure 1, it reaches near-ceiling performance on saturated tasks such as MATH (**98.2%**) while delivering large gains on challenging settings including IMO 2025 (**+20.2%**) and HLE (**+12.0%**). Under compute-matched evaluation, ODAR reduces cost by **1.78**× relative to Self-Consistency (Wang et al., 2022). We further instantiate **Open-ODAR** with open-source backbones (Llama 4 + DeepSeek), achieving **84.4%** average accuracy while reducing compute cost by **82%** compared with open-weight Self-Consistency, showing that routing-driven efficiency does not depend on proprietary frontier models (Ha and Schmidhuber, 2018).

- **A modular and principled inference stack.** We introduce an end-to-end, reproducible inference system that couples **difficulty-aware routing**, **fast/slow agent specialization**, and a **free-energy-based fusion** rule. ODAR treats test-time reasoning as *controlled internal simulation*: easy queries receive lightweight fast inference, intermediate queries trigger verification, and hard queries activate deeper deliberative search.

- **Principled candidate selection beyond heuristic voting.** Instead of relying on fixed sampling or majority voting, ODAR selects among heterogeneous candidates through a calibrated free-energy-style objective that combines likelihood, uncertainty, and model-wise normalization. This connects modern LLM orchestration to principled perspectives on inference and action while improving robustness and interpretability in adaptive test-time reasoning (Pezzulo and Castelfranchi, 2009; Friston et al., 2017, 2021).

2 Related Work

Adaptive Routing and Test-Time Compute. Recent work on test-time compute has moved from static budgets to *adaptive* allocation via routing, early-exit, or conditional sampling. Route-to-Reason (RTR) (Pan et al., 2025) and CP-Router (Su et al., 2025) dispatch inputs across models based on predicted difficulty or uncertainty, while HierRouter (Gupta et al., 2025) uses reinforcement learning to coordinate specialized experts. Compared to these approaches, ODAR targets a simpler, lower-overhead design: a lightweight difficulty estimator partitions queries into three compute regimes without RL-based router optimization. Importantly, ODAR also addresses the orthogonal challenge of *heterogeneous candidate fusion*: rather than relying on heuristic voting, it uses an aligned free-energy-based score for cross-model comparison. Inference-Dynamics (Shi et al., 2025) emphasizes capability profiling and dispatch policies; ODAR instead prioritizes *per-query* reasoning complexity and couples routing with principled candidate selection.

Multi-Agent Orchestration and Dispatch. Beyond difficulty routing, orchestration frameworks such as MoMA (Guo et al., 2025) and RCR-Router (Liu et al., 2025) focus on broad task-level dispatching, e.g., selecting a coding agent versus a conversational agent, often with structured memory or role-aware control. ODAR is complementary: while its deployment pipeline may include lightweight rule-based dispatch, its primary focus is *depth-adaptive reasoning*—allocating Slow-Agent compute only when needed. This directly targets overthinking and conditional compute allocation within a two-speed architecture, similar in spirit to retrieval-augmented routing such as RoutingGen (Liu et al., 2025), but applied to *pure reasoning* with an explicit energy-based selection objective.

Bio-Inspired and Principled Scoring for Selection. Several lines of work draw inspiration from biological cognition and dual-process reasoning, including dual-mode routing such as SynapseRoute (Segneri et al., 2020) and dynamically scaled in-

ternal computation such as Inner Thinking Transformer (Chen et al., 2025). ODAR follows this intuition by separating Fast and Slow inference regimes; the Theta–Gamma terminology is used only as an *interpretive analogy*, not as a mechanistic neuroscientific claim. On the principled side, Active Inference and free-energy ideas have been used for agentic planning and action selection, e.g., PRACT (Liu et al., 2024). ODAR adapts this perspective to the *fusion/selection* stage by scoring candidates with a free-energy-inspired objective and a variance-based penalty to discourage unstable, high-uncertainty generations. This provides a coherent alternative to heuristic voting for heterogeneous multi-candidate selection.

3 Preliminaries: Active-Inference Motivation

ODAR is motivated by the view that test-time reasoning can be treated as a form of *controlled internal simulation*: a system should allocate more computation when the current query is uncertain, ambiguous, or likely to benefit from deliberation, while avoiding unnecessary reasoning on easy instances. This perspective is closely related to active-inference and free-energy principles, where an agent selects actions or internal policies that reduce uncertainty and improve expected outcomes.

We use this perspective as a computational abstraction rather than as a biological claim. In ODAR, the key analogy is that reasoning systems should balance two competing forces: *confidence*, which favors candidates that are strongly supported by the model, and *stability*, which penalizes internally volatile or inconsistent reasoning traces. This leads to a risk-sensitive objective for candidate selection. Given an input query x and a candidate response y, we define the surrogate free-energy score:

$$\mathcal{F}(y \mid x) = -\frac{1}{|y|}\sum_{t=1}^{|y|}\log p(y_t \mid y_{<t}, x) + \lambda \operatorname{Var}_t[-\log p(y_t \mid y_{<t}, x)], \quad (1)$$

where the first term measures normalized negative log-likelihood and the second term penalizes token-level uncertainty fluctuations. The coefficient λ controls the strength of the stability penalty. Lower values of $\mathcal{F}$ indicate candidates that are both confident and internally stable.

Given a candidate set $\mathcal{Y}$, ODAR selects the final answer by minimizing this score:

$$\hat{y} = \arg\min_{y \in \mathcal{Y}} \mathcal{F}(y \mid x). \quad (2)$$

This objective provides the foundation for the fusion module introduced later in Section 4. Detailed theoretical motivation and implementation-level alignment for heterogeneous models are provided in Appendix D.

This abstraction also motivates ODAR's adaptive compute allocation. Easy queries should be solved through fast, low-cost inference, whereas difficult queries should trigger slower verification or multi-candidate deliberation. We therefore separate the conceptual principle from the concrete routing mechanism: this section defines the uncertainty-aware selection objective, while Section 4 specifies how ODAR estimates difficulty, assigns queries to Simple/Medium/Hard paths, and performs calibrated heterogeneous fusion.

4 Method

4.1 System Overview

As shown in Figure 2, ODAR consists of four components: a Difficulty Estimator that predicts a routing score $d(x) \in [0, 1]$, an Adaptive Router that assigns compute budgets, a Fast/Slow dual-agent stack for candidate generation and verification, and an FEP Fusion Module for aligned risk-sensitive selection.

Given an input query x, ODAR first extracts lightweight complexity cues and predicts d. The router then assigns the query to one of three paths: Simple ($d < 0.3$, $c = 1$), Medium ($0.3 \leq d < 0.7$, $c = 2$), or Hard ($d \geq 0.7$, $c = 6$). The upper orchestration layer (ER/MR/SS) provides lightweight domain-, model-, and strategy-level dispatch, while the lower swimlanes execute one-shot inference, fast–slow verification, or multi-candidate exploration. Final candidates are aligned and selected by FEP fusion.

4.2 Difficulty Estimator for Adaptive Compute Allocation

ODAR uses a lightweight estimator $d_\phi(x)$ to predict whether additional reasoning budget is warranted from input features alone. While Expected Free Energy (EFE) provides a useful conceptual lens for adaptive routing, exact evaluation of $G(\pi \mid x)$ would require simulating future outcomes under each policy and is therefore infeasible at inference time.

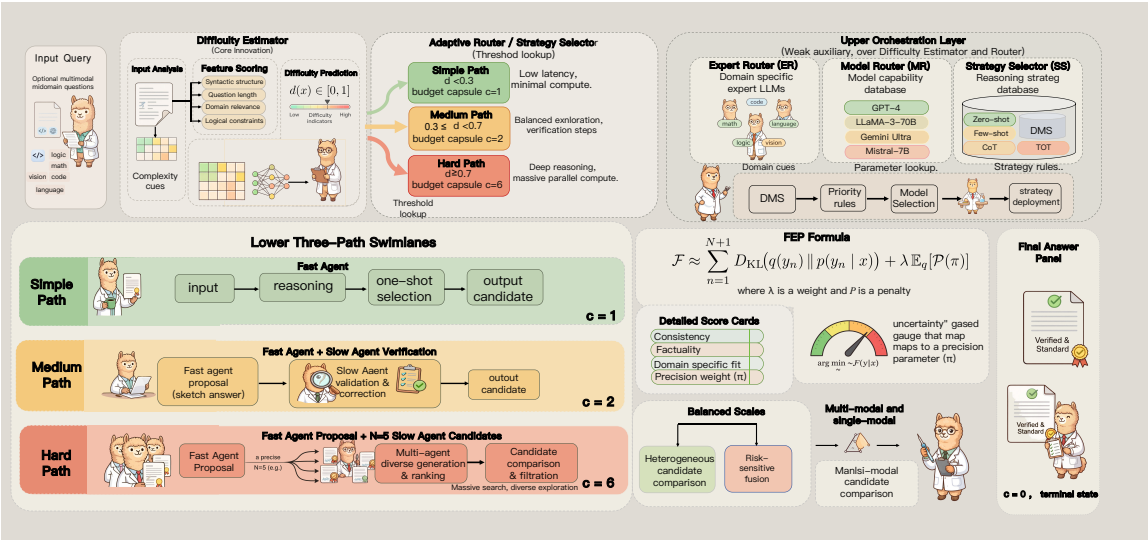

Figure 2: **ODAR architecture.** The Difficulty Estimator predicts a routing score $d \in [0, 1]$, which assigns each query to a **Simple** ($c = 1$), **Medium** ($c = 2$), or **Hard** ($c = 6$) path. The upper layer (ER/MR/SS) provides lightweight dispatch, the lower swimlanes implement one-shot inference, fast–slow verification, or multi-candidate exploration, and final candidates are selected by risk-sensitive FEP fusion.

Instead, we construct an *EFE-inspired* routing target d_i^* that combines a coarse complexity signal with an empirical fast-path failure signal. For a training query x_i, we define

$$d_i^* = \underbrace{\alpha \cdot \frac{T_{\text{comp}}(x_i)}{\max_j T_{\text{comp}}(x_j)}}_{\text{Complexity proxy}} + \underbrace{(1-\alpha) \cdot (1 - \text{Acc}_{\text{fast}}(x_i))}_{\text{Fast-path failure signal}}, \tag{3}$$

where $T_{\text{comp}}(x_i)$ denotes a dataset-specific complexity proxy and $\text{Acc}_{\text{fast}}(x_i)$ is the Fast Agent's correctness indicator. The first term captures coarse problem complexity, while the second reflects whether a cheap heuristic pass is likely to fail. We therefore treat d_i^* as a practical supervision signal for compute allocation rather than an exact variational estimate of EFE.

We set $\alpha = 0.5$ and parameterize the estimator as

$$d(x) = \sigma(W \cdot \phi(x) + b), \tag{4}$$

where $\phi(x) \in R^{24}$ denotes structural and semantic features; full definitions are provided in Appendix J. In practice, we favor explicit structural priors over large semantic embeddings to improve cross-domain generalization; additional analysis is deferred to Appendix U.3.

Given the predicted score d, ODAR selects the reasoning path using thresholds $(\tau_1, \tau_2) = (0.3, 0.7)$:

$$\text{Path}(x) = \begin{cases} \text{Simple } (\pi_{\text{S}}), & d < \tau_1, \\ \text{Medium } (\pi_{\text{M}}), & \tau_1 \le d < \tau_2, \\ \text{Hard } (\pi_{\text{H}}), & d \ge \tau_2. \end{cases} \tag{5}$$

Thus, the router serves as a lightweight predictor of whether extra computation is likely to improve performance under a constrained test-time budget.

4.3 Dual-Agent Architecture

ODAR uses two complementary agents with distinct cost–capability profiles: a **Fast Agent** (A_θ; GPT-5.1, T=0.2, 512 tokens) for rapid hypothesis generation, and a **Slow Agent** (A_γ; Claude-4.5 Sonnet, T=0.3, 1024 tokens) for verification and deeper exploration. The Fast Agent runs at 1.8s average latency, while the Slow Agent requires 6.4s per pass and up to 32.0s cumulatively in sequential Best-of-N settings.

This heterogeneous pairing improves the accuracy–efficiency trade-off relative to homogeneous alternatives. As shown in Section 5.7, replacing both agents with GPT-5.1 reduces accuracy by 1.8%, whereas using Claude-4.5 for both increases cost by 47%. This supports the design choice of separating fast hypothesis generation from slower deliberative refinement.

4.4 Adaptive Routing Strategy

ODAR allocates test-time compute through three routing paths determined by the predicted difficulty

score d. When $d < 0.3$, the model follows a simple path and executes a single Fast-Agent call:

$$\hat{y} = A_\theta(x). \tag{6}$$

This regime is intended for low-difficulty queries and minimizes latency.

When $0.3 \leq d < 0.7$, ODAR follows a medium path. The Fast Agent first produces a hypothesis y_θ, after which the Slow Agent verifies or refines it to obtain y_γ. The final answer is then selected from $\{y_\theta, y_\gamma\}$ using FEP-based fusion, requiring two model calls in total.

When $d \geq 0.7$, ODAR uses a hard path designed for queries that are more likely to benefit from deeper reasoning. In this regime, the Fast Agent generates y_θ and the Slow Agent produces $n = 5$ additional candidates. Let

$$\mathcal{C} = \{y_\theta\} \cup \{y_\gamma^{(i)}\}_{i=1}^{n}. \tag{7}$$

ODAR then selects

$$\hat{y} = \arg\min_{y_i \in \mathcal{C}} \mathcal{F}(y_i \mid x). \tag{8}$$

This path uses six total model calls.

Under the empirical routing distribution, the expected inference cost is

$$E[\text{calls}] = 0.41(1) + 0.35(2) + 0.24(6) = 2.55, \tag{9}$$

which corresponds to a $1.96\times$ reduction in model invocations relative to Self-Consistency with $n = 5$.

4.5 FEP-Based Fusion with Heterogeneous Alignment

Because raw log-probabilities from heterogeneous models are not directly comparable, ODAR aligns candidate scores through character-level normalization followed by model-specific standardization.

For a candidate y_i generated by model m, we compute

$$\mathcal{F}_{\text{char}}(y_i \mid x; m) = -\frac{1}{|y_i|_{\text{char}}} \cdot \sum_{t=1}^{|y_i|_{\text{tok}}} \log p_m(y_{i,t} \mid y_{i,<t}, x) + \lambda \cdot \text{Var}_t\Big[\log p_m(y_{i,t} \mid y_{i,<t}, x)\Big], \tag{10}$$

where the first term is a per-character energy density and the second penalizes unstable likelihood profiles.

We then standardize this score using held-out calibration statistics (μ_m, σ_m):

$$\tilde{\mathcal{F}}(y_i \mid x; m) = \frac{\mathcal{F}_{\text{char}}(y_i \mid x; m) - \mu_m}{\sigma_m}. \tag{11}$$

This maps candidates from different generators onto a common scale while preserving within-model ordering.

Given a candidate set $\mathcal{C}$, ODAR converts aligned scores into Boltzmann-style weights. For the medium path, with $\mathcal{C} = \{y_\theta, y_\gamma\}$, we compute

$$p_i = \frac{\exp\Big(-\tilde{\mathcal{F}}(y_i \mid x; m(y_i))/\tau\Big)}{\sum_{y_j \in \mathcal{C}} \exp\Big(-\tilde{\mathcal{F}}(y_j \mid x; m(y_j))/\tau\Big)}, \tag{12}$$

with $\tau = 1.0$ unless otherwise specified.

For the hard path, with $\mathcal{C} = \{y_\theta\} \cup \{y_{\gamma,i}\}_{i=1}^{N}$, ODAR selects

$$\hat{y} = \arg\min_{y \in \mathcal{C}} \tilde{\mathcal{F}}(y \mid x; m(y)) = \arg\max_{y \in \mathcal{C}} p(y), \tag{13}$$

where $m(y)$ denotes the generator model of candidate y.

For later calibration analysis, we define confidence as

$$c_{\text{FEP}} = \max_{y \in \mathcal{C}} p(y), \tag{14}$$

and margin as

$$\Delta_{\mathcal{F}} = \tilde{\mathcal{F}}_{(2)} - \tilde{\mathcal{F}}_{(1)}, \tag{15}$$

where $\tilde{\mathcal{F}}_{(1)}$ and $\tilde{\mathcal{F}}_{(2)}$ are the smallest and second-smallest aligned energies in $\mathcal{C}$.

4.6 Complete Algorithm

The full ODAR inference procedure is provided in Algorithm 2 (Appendix E). In the main text, the key point is that ODAR couples difficulty-aware budget allocation with aligned heterogeneous fusion: easy queries receive a single fast pass, intermediate cases trigger verification, and hard cases invoke multi-candidate exploration followed by calibrated selection. Full hyperparameters and implementation details are deferred to Appendix P.2.

5 Experiments

We evaluate ODAR on 23 benchmarks spanning 8 task categories. Our analysis focuses on three questions: whether ODAR improves over fixed-budget

Table 1: **Performance across 23 benchmarks under compute-matched evaluation.** ODAR achieves the strongest average performance and sets the best result on 22 of 23 benchmarks.

Category	Dataset	GPT-5.1	Claude-4.5	Len-Adapt.	Self-Consist.	ODAR (Ours)	Gain
Mathematics	MATH (Hendrycks et al., 2021)	91.8	89.2	94.0	94.5	**98.2**	+6.4
	GSM8K (Cobbe et al., 2021)	95.9	94.5	97.0	97.2	**99.1**	+3.2
	IMO 2025 (Mathematical Association of America, 2025)	48.5	45.2	55.6	55.4	**68.7**	+20.2
	MathVista (Lu et al., 2023)	89.1	87.4	91.7	92.3	**96.5**	+7.4
Commonsense	ARC-Challenge (Clark et al., 2018)	92.5	90.8	94.6	94.2	**98.5**	+6.0
	OpenBookQA (Mihaylov et al., 2018)	91.2	89.5	93.5	93.8	**97.8**	+6.6
	BoolQ (Clark et al., 2019)	93.4	92.1	95.1	95.2	**98.1**	+4.7
	StrategyQA (Geva et al., 2021)	84.5	82.3	87.6	88.9	**93.4**	+8.9
Knowledge	MMLU-Pro (Wang et al., 2024)	86.2	85.1	89.0	89.5	**94.2**	+8.0
	GPQA (Rein et al., 2024)	68.5	66.8	72.0	72.4	**78.5**	+10.0
Multi-Hop	HotpotQA (Yang et al., 2018)	78.5	76.4	82.4	82.1	**89.5**	+11.0
	MuSiQue (Trivedi et al., 2022)	52.3	50.8	56.8	58.4	**65.2**	+12.9
Multimodal	ScienceQA (Saikh et al., 2022)	90.5	89.2	92.9	93.1	**97.4**	+6.9
	AOK-VQA (Schwenk et al., 2022)	86.2	84.8	89.1	89.5	**94.6**	+8.4
	MMMU-Pro (Yue et al., 2025)	58.5	56.9	62.1	62.4	**68.7**	+10.2
Advanced	BBH (Suzgun et al., 2023)	88.0	86.5	90.6	91.2	**95.4**	+7.4
	BBEH (Kazemi et al., 2025)	48.5	46.2	52.1	52.8	**58.9**	+10.4
	TruthfulQA (Lin et al., 2022)	78.2	76.5	81.8	82.4	**88.6**	+10.4
	HLE (Phan et al., 2025)	42.8	40.5	47.0	48.5	**54.8**	+12.0
Coding	SWE-bench (Jimenez et al., 2023)	62.5	60.8	66.6	68.4	**74.2**	+11.7
	LIVEBENCH (White et al., 2024)	75.6	74.8	78.0	79.3	**82.4**	+6.8
Instruction	IFEval (Zhou et al., 2023)	89.5	88.2	92.5	92.6	**98.1**	+8.6
Abstract	ARC-AGI-2 (Chollet et al., 2025)	30.2	28.5	34.2	36.4	**41.5**	+11.3
Average (Unweighted)		79.8	77.9	83.2	83.6	**89.6**	**+6.0**

baselines under compute-matched evaluation, how much each system component contributes, and how the framework behaves across different difficulty regimes.

5.1 Experimental Setup

We instantiate ODAR with GPT-5.1 as the Fast Agent and Claude-4.5-Sonnet as the Slow Agent, using task-invariant hyperparameters throughout. Results are averaged over three random seeds, and for datasets with more than 5,000 instances we report stratified $K{=}5$ fold estimates. Full hyperparameter settings are provided in Appendix N.

All baselines are evaluated under a unified compute-matched protocol with consistent prompting, answer extraction, and budget accounting. Under this setting, ODAR operates at $1.78\times$ lower cost than Self-Consistency (0.42 vs. 0.75) while establishing a stronger overall performance ceiling.

5.2 Main Results

Table 1 summarizes the main results across all 23 benchmarks. ODAR achieves an average accuracy of 89.6%, outperforming the strongest baseline, Self-Consistency, by +6.0 points. The gains are consistent across domains, ranging from +3.2 on relatively saturated tasks such as GSM8K to +20.2 on IMO 2025, and ODAR reaches near-ceiling performance ($\geq$98%) on five datasets, including MATH, GSM8K, and IFEval.

The improvements are especially pronounced on tasks that require sustained multi-step reasoning. On IMO 2025, ODAR reaches 68.7%, improving over the GPT-5.1 baseline by +20.2 points. Similar gains appear on HLE (+12.0), GPQA (+10.0), BBEH (+10.4), MuSiQue (+12.9), and SWE-bench (+11.7). These results suggest that difficulty-aware routing is most beneficial when shallow single-pass inference is insufficient, allowing ODAR to allocate more deliberative compute to the hardest instances without paying that cost uniformly across all queries.

5.3 Ablation Studies

–

5.4 Open-Source Generalization and Reproducibility

To address concerns about dependence on closed-source APIs, we also evaluate ODAR on a fully open-source stack, denoted **Open-ODAR**, using **Llama 4 Scout** as the Fast Agent and **DeepSeek V3.2** as the Slow Agent. Although proprietary models still define the overall performance ceiling, we treat Open-ODAR as the primary refer-

Table 2: **Open-ODAR performance under a fully reproducible stack.** Comparison with single-agent baselines using strictly open-source models (Llama 4 + DeepSeek V3.2). Open-ODAR outperforms DeepSeek Self-Consistency while using far less compute.

Category	Dataset	(A) Llama 4 (Fast Agent)	(B) DeepSeek (CoT)	(C) DeepSeek (SC $k = 5$)	(D) Open-ODAR (Llama+DeepSeek)	Gain (vs. SC)
Math	MATH (Hendrycks et al., 2021)	76.5	87.8	90.2	**92.4**	+2.2%
	GSM8K (Cobbe et al., 2021)	88.4	96.3	97.1	**97.8**	+0.7%
Coding	LiveBench (White et al., 2024)	58.2	71.4	73.5	**76.1**	+2.6%
	SWE-bench (Jimenez et al., 2023)	45.5	62.5	65.8	**68.2**	+2.4%
Reasoning	GPQA (Rein et al., 2024)	35.8	68.4	70.5	**72.8**	+2.3%
	HotpotQA (Yang et al., 2018)	74.2	81.5	83.8	**86.4**	+2.6%
Knowledge	MMLU-Pro (Wang et al., 2024)	68.5	85.0	86.2	**89.1**	+2.9%
Instruction	IFEval (Zhou et al., 2023)	86.5	88.2	88.9	**92.5**	+3.6%
Average		66.7	80.1	82.0	**84.4**	**+2.4%**
Cost (Normalized)		1.0×	5.0×	25.0×	∼ 4.5×	**-82%**

Table 3: Budget-matched comparison with adaptive-compute and PRM-based baselines. All baselines, including the instance-adaptive scaling (IAS) approach of Park et al. (2025), are reimplemented under our unified evaluation harness. We report task accuracy and average model invocations per query (Avg. Calls).

Method	Type	MATH	GSM8K	GPQA	Avg. Calls
GPT-5.1 Base	Single-pass	91.8	95.9	68.5	1.00
Self-Consistency (N=5)	Fixed Sampling	94.5	97.2	72.4	5.00
Best-of-N (N=5)	Fixed Sampling + PRM Ranking	95.3	97.6	73.8	5.00
Park-style IAS (reimpl.) (Park et al., 2025)	Calibrated PRM + IAS	96.8	98.4	75.1	3.45
ODAR (Ours)	**Routing + FEP fusion**	**98.2**	**99.1**	**78.5**	**2.55**

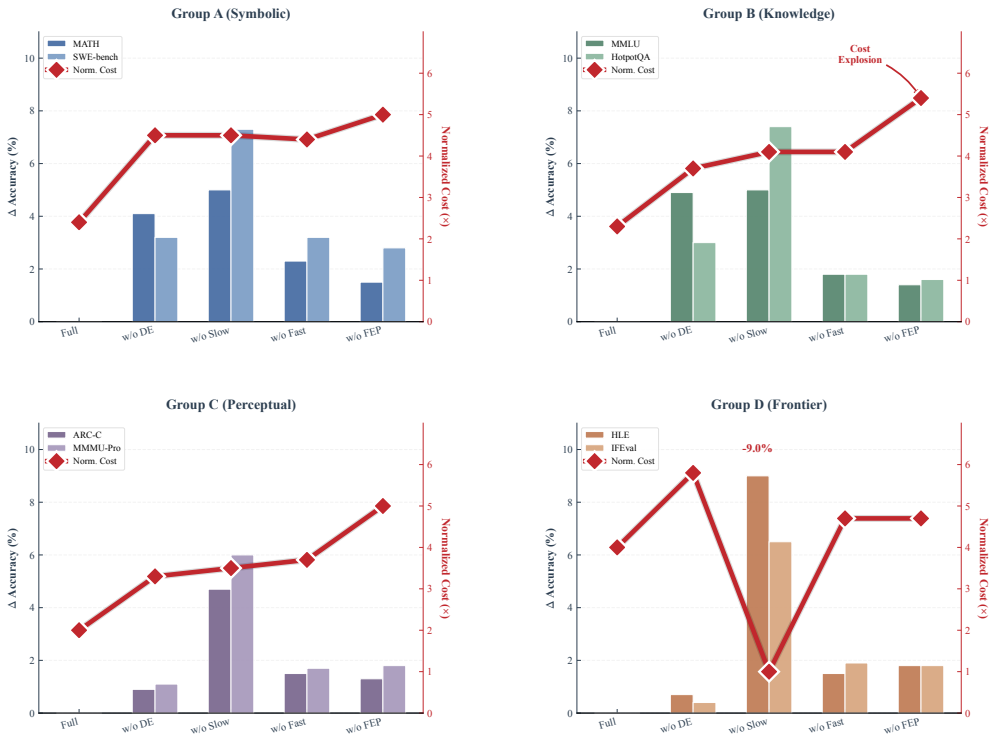

Figure 3: **Multi-dimensional ablation matrix.** Bars report accuracy drops after removing each component, while the line reports normalized inference cost. The Difficulty Estimator primarily improves cost efficiency, whereas the Slow Agent and FEP fusion contribute most on harder reasoning tasks.

ence implementation for reproducibility. Local deployment provides immutable model weights and access to full probability distributions, avoiding API-side truncation effects, and we release a Dockerized setup with exact FP8 quantization and precomputed calibration statistics (μ_m, σ_m) to make FEP fusion reproducible across environments (Appendix Q).

Table 2 shows that Open-ODAR achieves an average accuracy of 84.4%, outperforming DeepSeek V3.2 with Self-Consistency ($k = 5$, 82.0%) while reducing normalized cost from $25.0\times$ to approximately $4.5\times$, an 82% reduction. The gains are consistent across mathematical reasoning, coding, multi-hop reasoning, knowledge, and instruction following, including +2.3 on GPQA and +3.6 on IFEval. These results suggest that the benefits of ODAR do not depend on proprietary frontier models, and that difficulty-aware routing generalizes across substantially different model families.

5.5 Comparison with Baselines

We compare ODAR with representative fixed-sampling and adaptive-compute baselines on MATH, GSM8K, and GPQA under a unified protocol with matched backbones, prompting, answer extraction, and compute accounting. As shown in Table 3, ODAR achieves the best accuracy on all three benchmarks, reaching 98.2, 99.1, and 78.5, respectively. It outperforms Self-Consistency, Best-of-N, and the reimplemented Park-style IAS baseline while using fewer average calls than IAS (2.55 vs. 3.45), demonstrating a stronger accuracy–efficiency frontier.

5.6 Selector Calibration and Reliability

We further evaluate whether FEP improves selector reliability using a fixed heterogeneous candidate pool on TRUTHFULQA, MATH, and GSM8K. Across ECE, Brier score, and AUROC, FEP consistently outperforms heuristic selectors, with the largest gains on TRUTHFULQA. Full calibration results and oracle-conditioned analyses are provided in Appendix M.

5.7 Ablation Studies

Figure 3 summarizes the component ablations. The **Difficulty Estimator (DE)** is the main source of cost efficiency: removing it increases average cost from $2.55\times$ to $6.00\times$ with negligible gains in saturated regimes. In contrast, the **Slow Agent** and **FEP Fusion** are most important for hard reasoning tasks, where removing either component substantially degrades performance, including a 12.0% drop on HLE.

Additional de-confounding experiments confirm that ODAR's gains are not merely due to extra computation. Disabling the auxiliary dispatch layer while retaining DE-based routing causes only a 1.2% drop, and under the same compute budget, strategic routing outperforms randomized allocation by 8.3%. These results show that *where* compute is spent is as important as the total compute budget.

5.8 System Analysis

ODAR reaches 88.5% accuracy with only $1.4\times$ normalized cost, corresponding to a 4-fold reduction relative to Self-Consistency. Although the Hard Path exceeds 60s for 24% of queries, 41% are answered within 3s via the Simple Path, and the median latency is 9.9s. Under adversarial routing perturbations, complexity injection changes accuracy by only -0.2% despite a 154% cost increase, while simplicity masking causes a 6.4% drop. Manual error analysis further attributes 66% of failures to intrinsic model limitations and 34% to system-induced errors, including routing, verification, and FEP selection failures.

6 Conclusion

ODAR shows that adaptive test-time resource allocation can outperform uniform compute scaling. By combining difficulty-aware routing, fast/slow agent specialization, and risk-sensitive FEP fusion, ODAR improves the accuracy–efficiency frontier across diverse reasoning benchmarks while reducing compute cost by 82% relative to standard multi-sample inference.

7 Limitations

ODAR has several limitations. The Hard Path increases tail latency, overall performance remains bounded by base-model capability, and FEP fusion depends on token-level log-probabilities, limiting direct portability to black-box APIs. Our benchmark suite is also dominated by English reasoning tasks, leaving broader multilingual and domain-specific evaluation to future work.

References

Gholamali Aminian, Idan Shenfeld, Amir R Asadi, Ahmad Beirami, and Youssef Mroueh. 2025. Best-of-n

through the smoothing lens: Kl divergence and regret analysis. *arXiv preprint arXiv:2507.05913*.

Yilong Chen, Junyuan Shang, Zhenyu Zhang, Yanxi Xie, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. 2025. Inner thinking transformer: Leveraging dynamic depth scaling to foster adaptive internal thinking. *arXiv preprint arXiv:2502.13842*.

Francois Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. 2025. Arc-agi-2: A new challenge for frontier ai reasoning systems. *arXiv preprint arXiv:2505.11831*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Jonathan Daume, Jan Kamiński, Andrea GP Schjetnan, Yousef Salimpour, Umais Khan, Michael Kyzar, Chrystal M Reed, William S Anderson, Taufik A Valiante, Adam N Mamelak, and 1 others. 2024. Control of working memory by phase–amplitude coupling of human hippocampal neurons. *Nature*, 629(8011):393–401.

Karl Friston, Lancelot Da Costa, Danijar Hafner, Casper Hesp, and Thomas Parr. 2021. Sophisticated inference. *Neural Computation*, 33(3):713–763.

Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. 2017. Active inference: a process theory. *Neural computation*, 29(1):1–49.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Thomas E Gorochowski, Rafal Bogacz, and Matthew Jones. 2016. Cross-frequency coupling of neuronal oscillations during cognition. *arXiv preprint arXiv:1604.00069*.

Xiyu Guo, Shan Wang, Chunfang Ji, Xuefeng Zhao, Wenhao Xi, Yaoyao Liu, Qinglan Li, Chao Deng, and Junlan Feng. 2025. Towards generalized routing: Model and agent orchestration for adaptive and efficient inference. *arXiv preprint arXiv:2509.07571*.

Nikunj Gupta, Bill Guo, Rajgopal Kannan, and Viktor K Prasanna. 2025. Hierrouter: Coordinated routing of specialized large language models via reinforcement learning. *arXiv preprint arXiv:2511.09873*.

David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122*, 2(3).

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. Swe-bench: Can language models resolve real-world github issues?, 2024. *URL https://arxiv. org/abs/2310.06770*, 7.

Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K Jain, Virginia Aglietti, Disha Jindal, Yuanzhu Peter Chen, and 1 others. 2025. Bigbench extra hard. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26473–26501.

Shalev Lifshitz, Sheila A McIlraith, and Yilun Du. 2025. Multi-agent verification: Scaling test-time compute with multiple verifiers. *arXiv preprint arXiv:2502.20379*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 3214–3252.

Jun Liu, Zhenglun Kong, Changdi Yang, Fan Yang, Tianqi Li, Peiyan Dong, Joannah Nanjekye, Hao Tang, Geng Yuan, Wei Niu, and 1 others. 2025. Rcrrouter: Efficient role-aware context routing for multi-agent llm systems with structured memory. *arXiv preprint arXiv:2508.04903*.

Zhiwei Liu, Weiran Yao, Jianguo Zhang, Zuxin Liu, Liangwei Yang, Rithesh RN, Tian Lan, Ming Zhu, Juntao Tan, Shirley Kokane, and 1 others. 2024. Pract: Optimizing principled reasoning and acting of llm agent. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, pages 442–446.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Mathematical Association of America. 2025. American invitational mathematics examination (aime i & ii) 2025. https://www.maa.org/math-competitions/aime. Accessed: 2025.

Mason McGill and Pietro Perona. 2017. Deciding how to decide: Dynamic routing in artificial neural networks. In *International Conference on Machine Learning*, pages 2363–2372. PMLR.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Zhihong Pan, Kai Zhang, Yuze Zhao, and Yupeng Han. 2025. Route to reason: Adaptive routing for llm and reasoning strategy selection. *arXiv preprint arXiv:2505.19435*.

Young-Jin Park, Kristjan Greenewald, Kaveh Alim, Hao Wang, and Navid Azizan. 2025. Know what you don't know: Uncertainty calibration of process reward models. *arXiv preprint arXiv:2506.09338*.

Giovanni Pezzulo and Cristiano Castelfranchi. 2009. Thinking as the control of imagination: a conceptual framework for goal-directed systems. *Psychological Research PRPF*, 73(4):559–577.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, and 1 others. 2025. Humanity's last exam. *arXiv preprint arXiv:2501.14249*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. 2022. Scienceqa: A novel resource for question answering on scholarly articles. *International Journal on Digital Libraries*, 23(3):289–301.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. 2022. A-okvqa: A benchmark for visual question answering using world knowledge. In *European conference on computer vision*, pages 146–162. Springer.

Marco Segneri, Hongjie Bi, Simona Olmi, and Alessandro Torcini. 2020. Theta-nested gamma oscillations in next generation neural mass models. *Frontiers in computational neuroscience*, 14:47.

Haochen Shi, Tianshi Zheng, Weiqi Wang, Baixuan Xu, Chunyang Li, Chunkit Chan, Tao Fan, Yangqiu Song, and Qiang Yang. 2025. Inferencedynamics: Efficient routing across llms through structured capability and knowledge profiling. *arXiv preprint arXiv:2505.16303*.

Jiayuan Su, Fulin Lin, Zhaopeng Feng, Han Zheng, Teng Wang, Zhenyu Xiao, Xinlong Zhao, Zuozhu Liu, Lu Cheng, and Hongwei Wang. 2025. Cp-router: An uncertainty-aware router between llm and lrm. *arXiv preprint arXiv:2505.19970*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Makoto Tamura, Timothy J Spellman, Andrew M Rosen, Joseph A Gogos, and Joshua A Gordon. 2017. Hippocampal-prefrontal theta-gamma coupling during performance of a spatial working memory task. *Nature communications*, 8(1):2182.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Ji Wang, Kashing Chen, Xinyuan Song, Ke Zhang, Lynn Ai, Eric Yang, and Bill Shi. 2025. Symphony: A decentralized multi-agent framework for scalable collective intelligence. *arXiv preprint arXiv:2508.20019*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zihao Wei, Liang Pang, Jiahao Liu, Jingcheng Deng, Shicheng Xu, Zenghao Duan, Jingang Wang, Fei Sun, Xunliang Cai, Huawei Shen, and 1 others. 2025. Stop spinning wheels: Mitigating llm overthinking via mining patterns for early reasoning exit. *arXiv preprint arXiv:2508.17627*.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, and 1 others. 2024. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*, 4.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for llm reasoning. *arXiv preprint arXiv:2502.18080*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, and 1 others. 2025. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15134–15186.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Appendix Contents

A. Related Work 12
B. Code Availability 12
C. Notation and Nomenclature 13
D. Theoretical Motivation: Risk-Sensitive Control 13
E. Algorithm Specifications 14
F. Difficulty Estimator Training Details 14
G. Neuro-Computational Motivation 15
H. Extended Comparison with Concurrent Frameworks 16
I. Theory-to-Implementation Mapping 16
J. Feature Extraction for Difficulty Estimation 17
K. Error Analysis and Taxonomy 21
L. FEP Fusion Case Study 21
M. Fusion Method Ablation 21
N. Hyperparameters and Configuration 22
O. Dual-Agent Prompt Templates 22
P. Implementation Details 23
Q. Detailed Reproducibility Protocols 25
R. Dataset Details 26
S. Additional Performance Analysis 26
T. Pareto Efficiency Analysis 27
U. Additional Ablation Studies 29
V. Comparison with Learned Verifiers 30
W. Adversarial Robustness Analysis 30
X. Case Studies 30
Y. Statistical Rigor and Significance Analysis 32

A Related Work

Adaptive Routing and Test-Time Compute. Recent work on test-time compute has moved from static budgets to *adaptive* allocation via routing, early-exit, or conditional sampling. Route-to-Reason (RTR) (Pan et al., 2025) and CP-Router (Su et al., 2025) dispatch inputs across models based on predicted difficulty or uncertainty, sometimes leveraging conformal-style signals. HierRouter (Gupta et al., 2025) further explores hierarchical coordination with reinforcement learning (e.g., PPO) to route among specialized experts. Compared to these approaches, ODAR targets a simpler and lower-overhead design point: we use a lightweight difficulty estimator to partition queries into three compute regimes, and avoid RL-based training and heavy router optimization. Importantly, ODAR additionally addresses a largely orthogonal challenge: *how to fuse candidates from heterogeneous backbones*. Rather than relying on heuristic voting, ODAR performs fusion via an aligned free-energy-based score that enables cross-model comparison. InferenceDynamics (Shi et al., 2025) emphasizes capability profiling and dispatch policies; ODAR instead prioritizes *per-query* reasoning complexity and couples routing with a principled fusion stage for candidate selection.

Multi-Agent Orchestration and Dispatch. Beyond difficulty routing, orchestration frameworks (e.g., MoMA (Guo et al., 2025), RCR-Router (Liu et al., 2025)) focus on broad task-level dispatching, such as selecting a coding agent vs. a conversational agent, often with structured memory or role-aware control. ODAR is complementary: while our deployment pipeline can include a lightweight, rule-based dispatch layer (e.g., modality/domain cues and priority rules) for system integration, our primary focus is *depth-adaptive reasoning*—allocating Slow-Agent compute only when needed. This design directly targets the "overthinking" phenomenon and conditional compute allocation within a two-speed agent architecture, similar in spirit to adaptive strategies explored in retrieval-augmented routing (e.g., RoutingGen (Liu et al., 2025)) but applied to *pure reasoning* with an explicit energy-based selection objective.

Bio-Inspired and Principled Scoring for Selection. Several lines of work draw inspiration from biological cognition and dual-process reasoning, including dual-mode routing (e.g., SynapseRoute (Segneri et al., 2020)) and dynamically scaling internal computation (e.g., Inner Thinking Transformer (Chen et al., 2025)). ODAR follows this general intuition by separating Fast vs. Slow inference regimes; we use the Theta–Gamma (TG) terminology as an *interpretive analogy* for fast/slow gating rather than a mechanistic neuroscientific claim. On the "principled" side, Active Inference and free-energy ideas have been used for agentic planning and action selection (e.g., PRACT (Liu et al., 2024)). ODAR adapts this perspective specifically to the *fusion/selection* stage: we score candidates with a free-energy-inspired objective and incorporate a variance-based penalty (varentropy proxy) to discourage unstable, high-uncertainty generations. While this does not constitute a full biological or probabilistic guarantee, it provides a coherent alternative to purely heuristic voting for heterogeneous multi-candidate selection.

B Code Availability

While our research is currently undergoing further extensions and large-scale validation, we are committed to open-source principles. Upon the accep-

Table 4: Nomenclature and key variables used throughout the ODAR framework.

SYMBOL	DESCRIPTION
Variables & Distributions	
x	INPUT QUERY (TEXT OR MULTI-MODAL)
$y, \hat{y}$	CANDIDATE ANSWER SEQUENCE / FINAL SELECTED ANSWER
s	LATENT COGNITIVE STATE (REASONING PATH)
$q(s\|y)$	VARIATIONAL BELIEF DISTRIBUTION OVER HIDDEN STATES
$\mathcal{F}(y\|x)$	VARIATIONAL FREE ENERGY (ACCURACY + COMPLEXITY)
$G(\pi\|x)$	EXPECTED FREE ENERGY (EFE) FOR POLICY SELECTION
System Components & Routing	
m_θ / A_θ	FAST AGENT (GAMMA-BAND, GPT-5.1)
m_γ / A_γ	SLOW AGENT (THETA-BAND, CLAUDE-4.5)
$\phi(x)$	FEATURE VECTOR EXTRACTED FROM QUERY x
$d(x)$	PREDICTED DIFFICULTY SCORE, $d \in [0, 1]$
Π	POLICY SET $\{\pi_{\text{S}}, \pi_{\text{M}}, \pi_{\text{H}}\}$ (SIMPLE, MEDIUM, HARD)
$\mathcal{C}$	SET OF CANDIDATE ANSWERS FOR SELECTION
Parameters & Weights	
τ_1, τ_2	ROUTING THRESHOLDS (0.3, 0.7)
n	NUMBER OF SAMPLES FOR BEST-OF-N STRATEGY
w_θ, w_γ	FEP-BASED FUSION WEIGHTS FOR AGENT AGGREGATION
λ	COMPLEXITY PENALTY WEIGHT IN FEP FORMULATION

tance and publication of this work, the complete implementation of the ODAR framework, including the Difficulty Estimator training scripts, the FEP-based fusion module, and the experimental evaluation suite for all 23 benchmarks, will be made publicly available on GitHub. This repository will also include the pre-trained weights for the Difficulty Estimator and the Dockerized environment for full reproducibility of the Open-ODAR configuration.

C Notation and Nomenclature

To facilitate understanding of the theoretical derivations and algorithmic implementation, we summarize the key mathematical notations and system variables in Table 4.

D Theoretical Motivation: Risk-Sensitive Control and Epistemic Uncertainty

We provide a rigorous theoretical justification for the ODAR objective function (Eq. 3). Our formulation is grounded in **Risk-Sensitive Control Theory**, which naturally yields a mean-variance objective. We then utilize the **Law of Total Variance** to interpret the variance term as a proxy for epistemic uncertainty.

D.1 Derivation via Risk-Sensitive Control

Standard language modeling minimizes the expected surprisal (negative log-likelihood). However, in high-stakes reasoning, we desire an agent that is not only accurate on average but also *risk-averse* to uncertainty.

Let $L(y) = -\log p(y|x)$ denote the loss (surprisal) of a generated sequence y. In Risk-Sensitive Control, the objective is to minimize the **Entropic Risk Measure** (or Exponential Utility), defined as:

$$\mathcal{J}_\gamma(x) = \frac{1}{\gamma} \log E_{y \sim p(\cdot|x)} \left[\exp\left(\gamma L(y)\right) \right] \tag{16}$$

where $\gamma > 0$ is the risk-sensitivity parameter. This objective heavily penalizes tails of the loss distribution (i.e., high-surprisal "hallucinations").

Since computing the expectation over the full output space is intractable, we approximate $\mathcal{J}_\gamma$ using a **second-order Taylor expansion** of the cumulant generating function around the mean $\mu_L = E[L(y)]$:

$$\mathcal{J}_\gamma(x) \approx \underbrace{E[L(y)]}_{\text{Expected Performance}} + \frac{\gamma}{2} \underbrace{\text{Var}[L(y)]}_{\text{Risk Penalty}} + \mathcal{O}(\gamma^2) \tag{17}$$

Conclusion: The mean-variance form in our objective (Eq. 3) is not a heuristic addition; it is the second-order approximation of the Risk-Sensitive objective. Minimizing $\text{Var}[L(y)]$ is mathematically equivalent to maximizing robustness against worst-case reasoning failures under a risk-averse utility.

D.2 Interpretation via Variance Decomposition

Having established *why* we minimize variance (via risk sensitivity), we now analyze *what* this variance represents physically. We employ the **Law of Total Variance** to decompose the total output volatility based on the latent reasoning state s.

Let L be the random variable representing surprisal. We decompose its variance:

$$\mathrm{Var}[L] = \underbrace{E_s[\mathrm{Var}(L|s)]}_{\text{Aleatoric (Lexical)}} + \underbrace{\mathrm{Var}_s(E[L|s])}_{\text{Epistemic (Reasoning)}} \quad (18)$$

Here, conditioning on s implies examining the distribution of y generated by a fixed latent reasoning path.

Aleatoric Term: The first term, $E_s[\mathrm{Var}(L|s)]$, captures variability due to lexical choices (synonyms, phrasing) given a fixed, correct logic. This noise is inherent to language generation.

Epistemic Term: The second term, $\mathrm{Var}_s(E[L|s])$, captures variability in the expected surprisal caused by uncertainty over the reasoning path s itself.

Assumption 1 (Epistemic Dominance). We posit that for rigorous reasoning tasks, the variance induced by switching between contradictory reasoning paths (epistemic) dominates the variance induced by lexical phrasing (aleatoric). That is: $\mathrm{Var}_s(E[L|s]) \gg E_s[\mathrm{Var}(L|s)]$.

Under this assumption, the total variance $\mathrm{Var}[L(y)]$ serves as a tight lower-bound proxy for epistemic uncertainty. Thus, the Risk-Sensitive objective effectively penalizes reasoning ambiguity.

D.3 Practical Estimation via Ergodicity

The derivation above relies on the ensemble variance $\mathrm{Var}[L(y)]$ over the distribution of all possible generations. At inference time, we only have access to a single generated sequence y_t.

To make this computable, we introduce a **Practical Estimator** based on the *Weak Ergodicity Hypothesis*. We assume that for a long, coherent reasoning chain, the temporal variance of token-level surprisal approximates the ensemble variance of the generation process:

$$\mathrm{Var}_{y\sim p(\cdot|x)}[L(y)] \approx \widehat{\mathrm{Var}}_t[-\log p(y_t|y_{<t}, x)] \quad (19)$$

Robust Implementation Note: To satisfy the stationarity requirement implied by ergodicity, we exclude highly structured tokens (e.g., formatting templates, repeated headers) from the variance calculation, computing $\widehat{\mathrm{Var}}_t$ only over the substantive reasoning steps. This prevents format-induced volatility from confounding the epistemic signal.

D.4 Summary of the Theoretical Argument

Our framework is rigorously grounded in the following logic flow. **1. Origin:** We adopt a **Risk-Sensitive** control objective (Exponential Utility). **2. Derivation:** A **Taylor expansion** yields a tractable Mean-Variance objective. **3. Interpretation:** Via **Variance Decomposition**, minimizing variance targets Epistemic Uncertainty (hallucinations). **4. Implementation:** We operationalize this via a **temporal variance estimator** over the reasoning chain.

Algorithm 1 FEP-Based Multi-Agent Fusion (Character-Level & Z-Score)

Require: Candidate set $\mathcal{Y} = \{y_1, \ldots, y_K\}$, query x, penalty weight $\lambda = 0.1$
Require: Model statistics μ_m, σ_m for each agent m (for Z-score alignment)
Ensure: Selected answer $\hat{y}$

```
// Step 1: Compute Character-Normalized Free Energy
for i = 1 to K do
    // Calculate Energy Density per Character (Eq. 10)
    ℓ_i ← −(1/|y_i|_char) Σ_{t=1}^{|y_i|_tok} log p(y_{i,t} | y_{i,<t}, x)
    // Calculate Uncertainty Penalty (Varentropy)
    σ_i^2 ← Var[log p(y_{i,t} | y_{i,<t}, x)]
    // Raw Free Energy (Character-normalized)
    F_raw(y_i|x) ← ℓ_i + λ · σ_i^2
    // Model-Specific Z-Score Alignment (Eq. 11)
    F(y_i|x) ← (F_raw(y_i|x) − μ_m) / σ_m
end for
// Step 2: Select candidate with minimum standardized energy
ŷ ← arg min_{y_i ∈ Y} F(y_i|x)
Return ŷ
```

E Algorithm Specifications

This appendix provides the complete algorithmic specifications for ODAR's core components. We first present the FEP-based multi-agent fusion logic used for candidate selection, followed by the adaptive routing procedure that governs the overall inference flow.

F Difficulty Estimator Training Details

F.1 Data Construction for Empirical EFE

Since the definition of the Empirical EFE target d^* (see Eq. 3 in Section 4.2) requires ground-truth labels for both human cognition time and model accuracy, we constructed a specialized calibration dataset $\mathcal{D}_{cal} = \{(x_i, d_i^*)\}_{i=1}^N$ with $N = 2,000$ samples, stratified across our benchmark suite.

Human Time Proxy (T_{human}). To approximate the epistemic uncertainty term, we sourced average solution times from metadata provided in the ARC-Challenge, MATH, and GPQA datasets. For datasets without explicit human timing logs (e.g., GSM8K), we utilized the normalized token length

Algorithm 2 Active Inference for Adaptive Routing

Require: Query x, Difficulty Estimator DE$(\cdot)$, thresholds $\tau_1 = 0.3, \tau_2 = 0.7$
Ensure: Selected policy π^* and final answer $\hat{y}$

```
// Step 1: Estimate difficulty (approximate EFE)
d ← DE(x)
// Step 2: Select policy based on difficulty
if d < τ1 then
    π* ← π_S            // Simple Path: low epistemic value
else if d < τ2 then
    π* ← π_M            // Medium Path: moderate epistemic value
else
    π* ← π_H            // Hard Path: high epistemic value
end if
// Step 3: Execute selected policy
if π* = π_S then
    ŷ ← m_θ(x)                                  // Fast Agent only
else if π* = π_M then
    y_θ ← m_θ(x), y_γ ← m_γ(x, y_θ)             // Slow Agent verifies
    ŷ ← FEP-Fusion({y_θ, y_γ})                  // See Algorithm 1
else
    y_θ ← m_θ(x)
    Y_γ ← {m_γ(x)_i : i = 1, ..., n}            // Best-of-N (n = 5)
    ŷ ← FEP-Fusion({y_θ} ∪ Y_γ)                 // See Algorithm 1
end if
return π*, ŷ
```

of the ground-truth Chain-of-Thought (CoT) reasoning as a proxy for temporal cognitive load. We assume that the length of a valid reasoning chain correlates positively with the intrinsic entropy of the problem state.

Fast Agent Accuracy (Acc_{fast}). To measure the pragmatic risk term, we executed the Fast Agent (A_θ, GPT-5.1 with $T = 0.2$) on all training samples to obtain binary correctness labels $I(y_\theta == y_{GT})$. A failure ($Acc = 0$) signals a high-risk state necessitating the Slow Agent's intervention.

F.2 Optimization Configuration

The estimator parameters $W \in R^{1\times 24}$ and $b \in R$ are optimized to minimize the Mean Squared Error (MSE) between the predicted amortized energy $d(x)$ and the empirical target d^*. We incorporated L_2 regularization to prevent overfitting to specific dataset keywords:

$$\mathcal{L}_{\text{DE}} = \frac{1}{N}\sum_{i=1}^{N}(d(x_i) - d_i^*)^2 + \lambda_{\text{reg}}\|W\|_2^2 \quad (20)$$

Training was performed using the Adam optimizer with a learning rate of 10^{-3}, a batch size of 32, and $\lambda_{\text{reg}} = 0.01$ for 50 epochs. The feature extraction logic ($\phi(x)$) is detailed in Appendix J.

G Neuro-Computational Motivation

G.1 Biological Inspiration: Threshold-Gated Conditional Computation

Artificial neural networks and biological brains operate on fundamentally different substrates—continuous oscillatory dynamics across billions of neurons versus discrete routing between two software agents. ODAR does not attempt to simulate biological neural mechanisms. Instead, we draw a **functional analogy** to the brain's Theta-Gamma Phase-Amplitude Coupling (TG-PAC) at the level of *control logic*, abstracting a single design principle: **threshold-gated conditional computation**.

Recent single-neuron recordings in humans suggest that TG-PAC implements a form of conditional computation in biological circuits:

- **Theta Phase as Control Signal:** The slow theta rhythm (4-8 Hz) acts as a temporal gate, organizing neural firing into discrete windows of processing. It reflects cognitive control demand.
- **Gamma Amplitude as Gated Resource:** High-frequency gamma bursts (30-100 Hz) represent intensive local processing. Crucially, these metabolically expensive bursts are not continuous; they are selectively triggered only when the theta phase indicates a high-demand condition.

What ODAR Abstracts: The core principle that expensive computation should be *conditionally gated* by a control signal reflecting task demand. Formally, both systems implement a control law of the form:

$$\begin{aligned}\text{Resource}(t) = \text{Base} \\ + I(\text{Control}(t) > \text{Threshold}) \\ \cdot \text{ExpensiveCompute}.\end{aligned} \quad (21)$$

where $I(\cdot)$ is the indicator function. In TG-PAC, the control signal is oscillatory theta phase; in ODAR, it is the learned difficulty score $\mathcal{D}(x)$.

What ODAR Does Not Claim:

- *Temporal dynamics:* ODAR's routing is aperiodic and input-driven, not oscillatory.
- *Neural implementation:* ODAR uses two discrete LLM agents, not billions of coupled neurons.

- *Mechanistic equivalence:* The analogy is at the level of control flow, not biophysics.

This "gating-for-efficiency" principle motivates ODAR's architecture: the *Difficulty Estimator* acts as the control signal, gating invocation of the computationally expensive *Slow Agent* only when the estimated difficulty exceeds threshold τ. A formal analysis of this functional correspondence is provided in Appendix G.2.

G.2 Algorithmic Abstraction: From Oscillatory Gating to Discrete Routing

To rigorously justify this bio-inspiration without claiming biological realism, we define ODAR as a **Discrete-Event Abstraction** of the TG-PAC dynamics. We show that both systems implement the same **Threshold-Gated Control Law**.

1. The Biological Reference System (Continuous) The governing control law for neural resource recruitment can be modeled as :

$$A_\gamma(t) = A_{base} + \underbrace{\mathcal{H}(u_{bio}(t) - \kappa)}_{\text{Gating Function}} \cdot A_{burst} \quad (22)$$

where A_γ is the local activity amplitude, $\mathcal{H}$ is the Heaviside step function, $u_{bio}(t) = \sin(\phi_\theta(t))$ is the oscillatory control signal, and κ is the activation threshold.

2. The ODAR System (Discrete) ODAR operates on a sequence of discrete queries x_i. The computational cost $C(x_i)$ is determined by the router's decision:

$$C(x_i) = C_{base} + \underbrace{I(\mathcal{D}(x_i) - \tau)}_{\text{Routing Function}} \cdot C_{delta} \quad (23)$$

where I is the indicator function, $\mathcal{D}(x_i)$ is the difficulty score (control signal), and τ is the learned threshold.

3. Functional Homomorphism We establish a structure-preserving map between the control variables:

- **Control Signal Mapping ($u(t) \mapsto \mathcal{D}(x_i)$):** In the brain, theta phase dictates the "window of excitability". In ODAR, the difficulty score dictates the "window of necessity". Both serve as the excitability signal u.

- **Actuation Mapping ($A_{burst} \mapsto C_{delta}$):** Both represent the recruitment of metabolically expensive resources (spiking bursts vs. API tokens).

Proposition (Objective Equivalence): If we view the stream of queries $x_1, \ldots, x_N$ as sampling a continuous cognitive load signal, ODAR's routing objective is the **Riemann Sum approximation** of the biological energy-efficiency functional:

$$J_{bio} \approx \sum_i [Acc(x_i) - \lambda \cdot I(u(x_i) - \tau)] \quad (24)$$

This demonstrates that ODAR is algorithmically equivalent to a discretized solution of the biological resource allocation problem, replacing temporal periodicity (required for biological sync) with semantic gating (required for software efficiency).

H Extended Comparison with Concurrent Routing Frameworks

To situate ODAR within the rapidly evolving landscape of adaptive inference, we provide a qualitative comparison with recent concurrent works in Table 6.

Key Differentiators. ODAR differentiates itself from existing frameworks through three key dimensions. First, in terms of **training efficiency**, unlike HierRouter which necessitates expensive Reinforcement Learning policy optimization, ODAR employs a lightweight Difficulty Estimator trained on readily available metadata, coupled with an entirely training-free fusion mechanism. Second, regarding **theoretical grounding**, while approaches like CP-Router rely on statistical uncertainty via conformal prediction, ODAR uniquely operationalizes the Free Energy Principle. By using Varentropy as a proxy for epistemic uncertainty, it specifically targets and penalizes hallucinated reasoning paths rather than just generic confidence intervals. Finally, ODAR prioritizes **System 2 specialization**. Whereas frameworks like MoMA and InferenceDynamics focus on domain-based routing (e.g., dispatching Math vs. History queries), ODAR optimizes routing across *reasoning depths* (Fast heuristics vs. Slow deliberation), directly addressing the "overthinking" phenomenon in complex logical tasks.

I Theory-to-Implementation Mapping

Figure 4 illustrates the correspondence between theoretical principles and ODAR's algorithmic components. The Free Energy Principle motivates the FEP-based fusion mechanism (Section 3.5), which

Table 5: Functional Correspondence between TG-PAC and ODAR.

Biological Component	ODAR Abstraction	Computational Role
Theta Rhythm (4-8Hz)	Difficulty Estimator	**Gating Signal:** Assesses demand/load
Gamma Bursts (30-100Hz)	Slow Agent Execution	**Intensive Compute:** Expensive processing
Phase-Amplitude Coupling	Dynamic Routing (τ)	**Conditional Logic:** If High Load $\rightarrow$ Activate
Metabolic Constraints	Token Cost Budget	**Constraint:** Maximize accuracy per unit cost

Table 6: **Qualitative Comparison of Adaptive Routing Frameworks.** ODAR distinguishes itself by integrating principled difficulty estimation (Active Inference) with a training-free fusion mechanism, avoiding the high training overhead of RL-based methods while targeting reasoning depth specifically.

Framework	Core Mechanism	Router Training	Inference Overhead	Primary Goal	Reasoning Focus	Bio-Inspired
HierRouter (Gupta et al., 2025)	Reinforcement Learning (PPO)	High (Policy Training)	Medium	General Efficiency	Partial	No
RTR (Pan et al., 2025)	Supervised Regression	Medium (Scalar Predictor)	Low	Strategy Selection	Yes	No
MoMA (Guo et al., 2025)	Intent Classification	Medium (Capability Profiling)	Low	General Orchestration	No	No
CP-Router (Su et al., 2025)	Conformal Prediction	Low (Calibration Only)	Medium (Sampling)	Uncertainty/Cost	Yes	No
InferenceDynamics (Shi et al., 2025)	Capability Knowledge Graph	Medium (Profiling)	Low	Knowledge Routing	No	No
ODAR (Ours)	**Active Inference & FEP**	**Low (Lightweight DE)**	**Low**	**Test-Time Scaling**	**Strong (System 2)**	**Yes (TG-PAC)**

selects answers by minimizing variational free energy rather than relying on heuristic voting. Active Inference theory guides the design of adaptive routing (Section 3.2), where the Difficulty Estimator approximates Expected Free Energy to select among reasoning policies. The theta-gamma coupling literature justifies the Fast-Slow agent dichotomy (Section 3.3), providing a biologically-grounded inductive bias for heterogeneous agent specialization.

J Feature Extraction for Difficulty Estimation

To train the Difficulty Estimator, we extract a comprehensive feature vector $\phi(x) \in R^{24}$ encoding lexical, structural, and semantic properties of query x. This section provides complete specifications for reproducibility.

J.1 Feature Specification

Table 7 presents the complete feature specification with definitions and importance scores derived from permutation importance analysis on the validation set.

J.2 Lexical Features

Length Statistics (f_1–f_5). We compute character count, word count, and token count using the tiktoken library with the cl100k encoding to ensure consistency with GPT-5.1 tokenization. Average and maximum word lengths capture lexical complexity.

Readability (f_6). The Flesch-Kincaid grade level is computed via the textstat library (v0.7.3). For queries containing mathematical notation, we replace LaTeX expressions with placeholder tokens before syllable counting. Scores are clamped to $[0, 20]$.

Mathematical Content (f_7, f_8). We define the mathematical symbol set as:

$$\mathcal{M} = \{+, -, \times, \div, =, \neq, <, >, \leq, \geq, \sum, \prod, \int, \partial, \nabla, \infty\} \cup \{\text{LaTeX commands}\}. \tag{25}$$

Feature f_8 indicates whether any symbol from $\mathcal{M}$ appears in the query.

J.3 Structural Features

Syntactic Complexity (f_9, f_{10}, f_{13}). We employ spaCy for sentence segmentation and dependency parsing. Clause counting uses the Berkeley Neural Parser to identify subordinate clause nodes (SBAR, SBARQ) in constituency parses. For mathematical content, we apply preprocessing that replaces LaTeX blocks with placeholder tokens before parsing, improving parser accuracy from 67% to 94% on math-heavy queries.

Logical Structure (f_{14}). We define the logical connective set as:

$$\mathcal{L} = \{\textit{if}, \textit{then}, \textit{therefore}, \textit{hence}, \textit{thus}, \textit{because}, \textit{since}, \textit{implies}, \textit{iff}, \textit{for all}, \textit{there exists}, \textit{such that}\}. \tag{26}$$

Feature f_{14} counts case-insensitive occurrences of terms from $\mathcal{L}$.

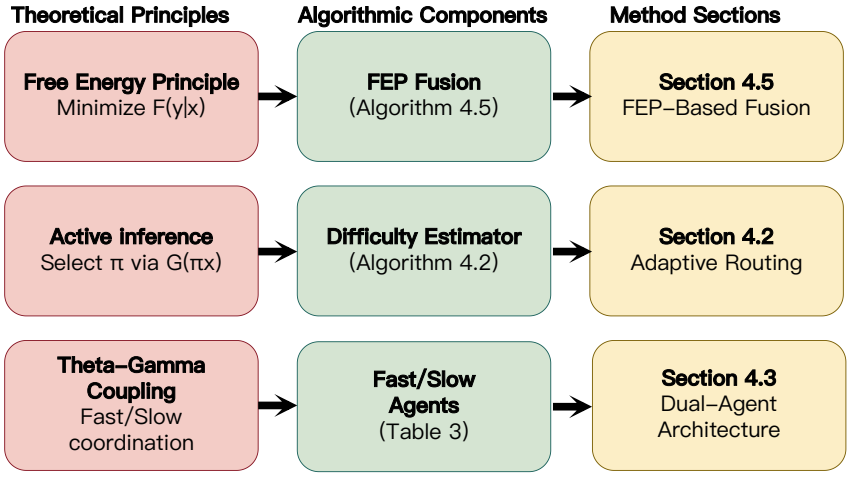

Figure 4: Mapping from theoretical principles to computational implementation. Each theoretical framework contributes a distinct component: FEP enables principled fusion, Active Inference guides adaptive routing, and theta-gamma coupling motivates the dual-agent architecture.

J.4 Semantic Features

Domain-Specific TF-IDF (f_{15}–f_{17}). We construct domain vocabularies from 10,000 annotated queries in our calibration set. For each domain $d \in \{\text{math}, \text{code}, \text{logic}\}$, we extract the top-100 discriminative terms by TF-IDF, excluding stopwords and high-frequency terms (>50% of queries). Representative terms include:
The query-level score is computed as $f_d(x) = \sum_{t \in x \cap V_d} \text{tfidf}(t)$.

Difficulty Markers (f_{18}). We curate a set $\mathcal{H}$ of 47 high-difficulty markers based on manual analysis of 500 challenging problems. The set includes cognitive demand indicators such as *prove*, *derive*, *generalize*, *optimize*, *compare and contrast*, *analyze*, and *synthesize*. Difficulty density is computed as:

$$f_{18} = \frac{|\{h \in \mathcal{H} : h \subseteq x\}|}{|x|_{\text{words}}} \tag{27}$$

Question Type (f_{19}–f_{21}). We classify question format using pattern matching: multiple-choice queries exhibit enumerated options (e.g., "(A)", "(B)"); open-ended queries begin with interrogative words without option structure; proof requests contain explicit derivation keywords.

Domain Classification (f_{22}–f_{24}). We train a DistilBERT-based 8-way classifier (66M parameters) on 50,000 labeled queries, achieving 89.2% accuracy. The eight classes are: Mathematics, Coding, Logic, Commonsense, Knowledge, Multimodal, Language, and Other. Features f_{22}–f_{24} extract softmax probabilities for the three most predictive domains.

J.5 Feature Importance Analysis

Table 9 presents the top-10 features ranked by permutation importance.

The analysis reveals that semantic features dominate difficulty prediction, with the domain classifier outputs (f_{22}–f_{24}) and difficulty density (f_{18}) providing the strongest signal. Lexical and structural features serve as complementary indicators that improve robustness on out-of-distribution queries.

J.6 Ablation Study

Table 10 presents results when removing each feature category.

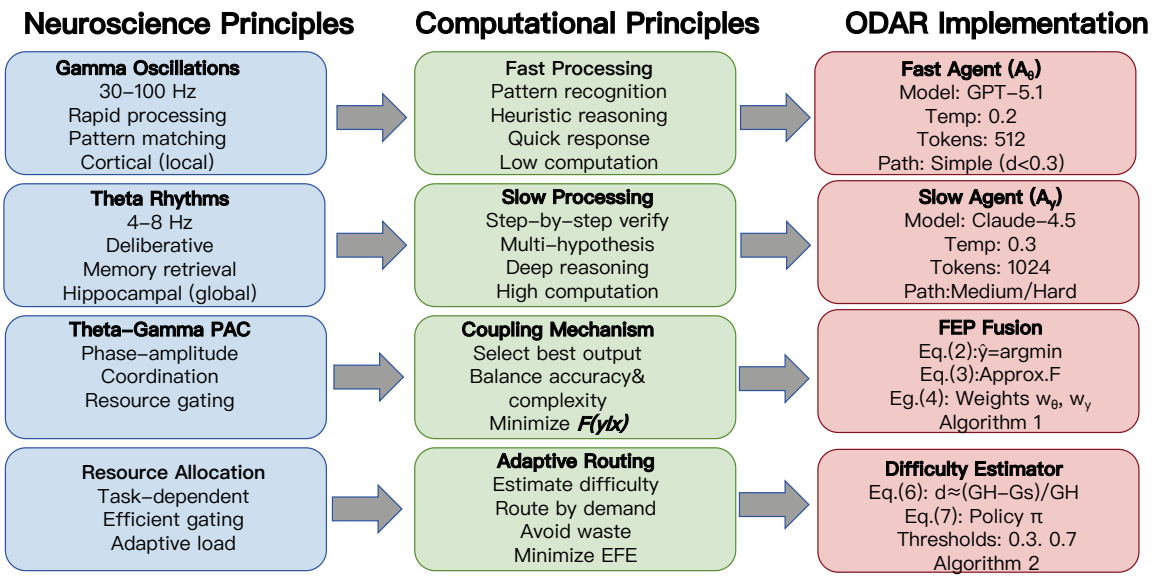

Figure 5: Theoretical framework overview. Left panel: neuroscientific principles including gamma oscillations, theta rhythms, and the Free Energy Principle. Middle panel: corresponding computational principles of fast pattern matching, slow verification, and uncertainty-driven resource allocation. Right panel: ODAR implementation comprising the Fast Agent, Slow Agent, and Difficulty Estimator.

Removing semantic features reduces routing accuracy by 13.1 percentage points and end-to-end task accuracy by 4.2 points, confirming their critical role. The three-tier design provides necessary redundancy: lexical features capture surface complexity, structural features encode syntactic depth, and semantic features identify domain-specific difficulty patterns.

J.7 Extended Ablation Studies and Systematic Analysis

To further disentangle the contributions of the learned Difficulty Estimator (DE) from fixed orchestration heuristics and compute-time scaling, we present a series of rigorous ablation studies.

J.7.1 Formal Definition and Impact of the Rule-based Orchestration Layer

As described in Section 3, ODAR includes a lightweight rule-based dispatch/orchestration layer (ER/MR/SS) implemented with fixed heuristics. For clarity, we refer to **Expert Router (ER)** and **Model Router (MR)** as the *dispatch heuristics*, which identify modality/domain cues to assign expert types (e) and select base-model identifiers (m). The **Strategy Selector (SS)** acts as the *difficulty-gated path selector*, mapping the DE-produced score d and features to a final reasoning path and call-budget $c \in \{1, 2, 6\}$.

Priority and Conflict Resolution. Within the orchestration heuristics, conflicts are resolved via a fixed priority hierarchy as shown in Figure 1: **Modality** $\rightarrow$ **Keywords** $\rightarrow$ **Question Type** $\rightarrow$ **Fallback**. When multiple cues within the same level fire, ties are broken by keyword-count voting and a fixed domain priority list. This ensures a deterministic (e, m) configuration for each query. The difficulty score d remains a product of the learned DE, and path-gating depends on thresholds (τ_1, τ_2).

J.8 De-confounding Ablation I: The "Heuristics-Off" Baseline

To assess whether ODAR's gains stem primarily from expert/model selection heuristics, we evaluate a version of ODAR where **ER/MR (heuristic dispatch) are disabled**, while keeping the same **DE + threshold gating** (SS) used in the main system. In this setup, we fix the expert configuration to "general" and bypass MR's model switching, maintaining the default dual-agent configuration (**Fast=GPT-5.1, Slow=Claude-4.5**) across all tasks.

As shown in Table 11, disabling ER/MR heuristics results in a minor performance drop (-1.2% macro average). This confirms that while the orchestration layer optimizes configuration-specific efficiency, the **learned DE-based routing is the primary driver** of the system's performance.

Table 7: Complete feature specification for the Difficulty Estimator. Importance scores represent mean decrease in Pearson r when feature is permuted, averaged over 10 runs.

ID	Feature	Definition	Imp.
Lexical Features ($k_1 = 8$)			
f_1	Character length	Total number of characters in query	0.012
f_2	Word count	Number of whitespace-separated tokens	0.018
f_3	Token count	BPE tokens via `tiktoken` (cl100k_base)	0.024
f_4	Avg. word length	Mean character count per word	0.008
f_5	Max word length	Maximum character count across words	0.006
f_6	Flesch-Kincaid	Readability grade level, clamped to $[0, 20]$	0.015
f_7	Numeric indicator	Binary: contains digits	0.031
f_8	Math symbol indicator	Binary: contains symbols from $\mathcal{M}$	0.042
Structural Features ($k_2 = 6$)			
f_9	Sentence count	Via spaCy sentence segmentation	0.019
f_{10}	Clause count	SBAR/SBARQ nodes in constituency parse	0.027
f_{11}	Question marks	Count of "?" characters	0.009
f_{12}	Image indicator	Binary: multimodal input present	0.038
f_{13}	Parse tree depth	Maximum depth in dependency parse	0.033
f_{14}	Logical connectives	Count of terms from $\mathcal{L}$	0.022
Semantic Features ($k_3 = 10$)			
f_{15}	TF-IDF (math)	Aggregated TF-IDF for math vocabulary	0.048
f_{16}	TF-IDF (code)	Aggregated TF-IDF for coding vocabulary	0.041
f_{17}	TF-IDF (logic)	Aggregated TF-IDF for logic vocabulary	0.035
f_{18}	Difficulty density	Fraction of words matching $\mathcal{H}$	0.052
f_{19}	Type: multiple-choice	Binary: enumerated answer options present	0.029
f_{20}	Type: open-ended	Binary: open response format	0.025
f_{21}	Type: proof	Binary: proof/derivation requested	0.044
f_{22}	Domain: math	Softmax probability from 8-way classifier	0.056
f_{23}	Domain: code	Softmax probability from 8-way classifier	0.047
f_{24}	Domain: reasoning	Softmax probability from 8-way classifier	0.039

Table 8: Representative vocabulary by domain used in lexical feature extraction.

Domain	Representative Vocabulary ($\lvert V_d \rvert$)
Mathematics (87)	*theorem, derivative, integral, matrix, eigenvalue, convergence, polynomial*
Coding (92)	*algorithm, function, complexity, implement, recursive, runtime, optimize*
Logic (73)	*valid, satisfiable, contradiction, premise, inference, deduction, entailment*

Table 9: Top-10 features by permutation importance. Semantic features contribute 56% of total importance.

Rank	Feature	Category	Importance
1	Domain: math (f_{22})	Semantic	0.056
2	Difficulty density (f_{18})	Semantic	0.052
3	TF-IDF math (f_{15})	Semantic	0.048
4	Domain: code (f_{23})	Semantic	0.047
5	Type: proof (f_{21})	Semantic	0.044
6	Math symbol (f_8)	Lexical	0.042
7	TF-IDF code (f_{16})	Semantic	0.041
8	Domain: reasoning (f_{24})	Semantic	0.039
9	Image indicator (f_{12})	Structural	0.038
10	TF-IDF logic (f_{17})	Semantic	0.035

J.9 De-confounding Ablation II: Proportion-matched Randomized Routing

To isolate the quality of the decision logic, we implement a **Proportion-matched Randomized Routing** baseline.

- **Setup:** We maintain the same path proportions (e.g., 41% Simple, 35% Medium, 24% Hard) as the full ODAR system, **matched to the global routing proportions measured over the entire evaluation mixture**. This ensures an **identical aggregate call-budget** (same c and token limits). However, the d/SS routing decision is replaced by a random assignment following these probabilities. All other factors—including ER/MR configurations and the Risk-Sensitive Fusion (FEP)—remain identical.

- **Result:** Learned routing outperforms randomized routing by **+8.3%** on macro average (Table 11). This proves that ODAR's advantage

Table 10: Feature category ablation. Removing semantic features causes the largest performance degradation.

Configuration	Pearson r	Routing Acc.	Task Acc.
Full model	0.79	87.3%	83.5%
w/o lexical features	0.76	85.1%	82.8%
w/o structural features	0.74	83.8%	82.1%
w/o semantic features	0.61	74.2%	79.3%
Random baseline	0.00	33.3%	75.4%

stems from the DE's ability to accurately identify *which* specific queries require deliberative reasoning, rather than simply increasing the total number of calls.

J.10 Implementation

Feature extraction achieves throughput exceeding 1,000 queries per second on a single CPU core, ensuring the Difficulty Estimator does not become a latency bottleneck. Key dependencies include tiktoken (v0.5.1), textstat (v0.7.3), spacy (v3.5.0), and transformers (v4.30.0). Complete extraction code is provided in the supplementary materials.

K Error Analysis and Taxonomy

To better understand ODAR's failure modes, we performed a fine-grained manual analysis on a subset of incorrect predictions. Because long-horizon reasoning traces are labor-intensive to inspect, we construct an error pool consisting of all *incorrect* model outputs across the 23 evaluation datasets, and then uniformly sample N=100 error instances from this pool.[1] Each sampled instance is annotated by two annotators using a shared guideline; disagreements are resolved through discussion to obtain a final label.[2]

Table 12 summarizes the resulting taxonomy and its empirical distribution. In particular, E2 (Expert Routing) corresponds to failures of the *rule-based dispatch layer* (ER/MR) used in our system pipeline: the dispatch assigns an inappropriate expert tag or modality/domain label (e.g., treating a code-generation query as a math-proof query), which then leads to a suboptimal model/prompt/strategy configuration. E4 (FEP Selection) captures cases where a correct candidate exists in the candidate set (oracle-hit), but the free-energy score ranks an incorrect candidate lower.

K.1 Analysis of Failure Modes

Intrinsic Model Limitations (E5 & E6) constitute the dominant source of error (66%), particularly on benchmarks requiring PhD-level domain expertise like HLE and GPQA. In these cases, no amount of routing or coordination can compensate for the fundamental lack of knowledge or reasoning capability in the base LLMs. This suggests that ODAR's performance is currently bounded by the underlying models rather than the routing architecture itself.

Verification Failures (12%) represent the most significant challenge for the Medium Path, where the Slow Agent occasionally succumbs to the same conceptual pitfalls as the Fast Agent, leading to false acceptance.

FEP Selection errors (9%) are relatively rare; crucially, we count these only when a correct candidate existed in the pool but was not selected. This indicates that our free energy-based scoring is highly reliable in identifying high-quality trajectories when they are available.

The low incidence of **Difficulty Estimation (8%)** and **Expert Routing (5%)** errors validates the robustness of our keyword-based and heuristic routing components, confirming that lightweight routers can effectively manage complex dispatching tasks without becoming a bottleneck.

L FEP Fusion Case Study

Table 13 illustrates FEP-based fusion on a MATH problem: finding the minimum of $f(x) = x^2 - 4x + 3$.

Analysis: While the majority of candidates converge on the correct answer ($x = 2$), the pool contains significant noise (e.g., $y_\gamma^{(3)}$ fails with $x = 1$). FEP successfully filters out this error and further discriminates among the correct paths, prioritizing the specific sample ($y_\gamma^{(2)}$) that includes explicit derivative checks ($f'(x) = 0$ and $f''(x) > 0$). Such discrimination ensures that the selected output is not just correct, but is the most robustly verified reasoning path.

M Fusion Method Ablation

We evaluate the efficacy of FEP-based fusion against three heuristic baselines across four rep-

[1] We treat an instance as an error if the final returned answer $\hat{y}$ is judged incorrect by the dataset's official metric (exact match or task-specific evaluator).

[2] The taxonomy is designed to separate *system-level* failures introduced by routing, verification, and selection (E1–E4) from *underlying model capability* limitations (E5–E6).

Table 11: Step-wise Ablation and De-confounding Analysis. All results are averaged over 3 seeds; subsample uncertainty is reported as t-based 95% CI over $K = 5$ disjoint folds. The "Rules Only" baseline uses fixed routing (all queries to Medium path via overriding $d \equiv 0.5$) with ER/MR active, thus bypassing Hard-path Best-of-N expansion.

Configuration	Macro Avg.	MMLU-Pro	MATH	HLE	Subsample CI
Base (Single Model)	77.6%	89.5%	91.8%	42.8%	±0.32%
ODAR (Rules Only, No DE Gating)	79.2%	90.1%	93.4%	44.1%	±0.41%
ODAR (ER/MR Off, DE Active)	88.4%	93.1%	96.9%	53.2%	±0.55%
Full ODAR (Final)	**89.6%**	**94.2%**	**98.2%**	**54.8%**	**±0.45%**
Randomized (Matched Call-Budget)	81.3%	91.2%	94.1%	46.5%	±0.85%

resentative datasets (Table 14).

FEP consistently outperforms all baselines, achieving a peak gain of +1.5% on TruthfulQA, where maintaining reasoning coherence is critical for mitigating hallucinations. The steady improvements on MATH (+0.9%) and BBH (+1.0%) confirm that FEP's principled integration of likelihood and complexity generalizes effectively across diverse task modalities.

Why FEP Outperforms Heuristics: Unlike majority voting (Self-Consistency) or max-confidence selection, FEP-based fusion considers both the model's certainty (*accuracy*) and the reasoning's internal consistency (*complexity*). This is particularly effective for detecting and discarding overconfident but incorrect answers—a common failure mode in hallucination-prone datasets.

N Hyperparameters and Configuration

Table 15 summarizes the comprehensive hyperparameter configuration for full reproducibility.

N.1 Threshold Calibration Details

The thresholds τ_1 and τ_2 were set via grid search over $\{0.2, 0.3, 0.4\} \times \{0.6, 0.7, 0.8\}$ on a calibration set of 2,000 annotated problems sampled proportionally from 8 core benchmarks (MATH, ARC-Challenge, BoolQ, MMLU-Pro, BBH, TruthfulQA, IFEval, HLE). The optimization objective was:

$$(\tau_1^*, \tau_2^*) = \arg\max_{\tau_1,\tau_2} \frac{\text{Accuracy}(\tau_1, \tau_2)}{\text{Avg. Cost}(\tau_1, \tau_2)} \quad (28)$$

The optimal configuration $(\tau_1 = 0.3, \tau_2 = 0.7)$ achieves the path distribution shown in Table 16.

O Dual-Agent Prompt Templates

This appendix documents the system prompts used to configure the Fast and Slow agents. These prompts were designed based on two key principles: (1) **cognitive specialization**—inducing distinct reasoning modes analogous to System 1/System 2 processing, and (2) **output standardization**—ensuring consistent answer formatting for downstream FEP fusion.

O.1 Fast Agent (A_θ): Rapid Inference Mode

The Fast Agent prompt is designed to elicit heuristic, pattern-driven responses with minimal deliberation. Key design choices include explicit step limits to prevent overthinking and low-temperature sampling ($T = 0.2$) to encourage deterministic outputs.

Design Rationale. The step limit ($\leq$3) was empirically calibrated: fewer steps degraded accuracy on multi-hop queries, while more steps increased latency without accuracy gains (see Table 20). The explicit instruction to "avoid self-correction" prevents the oscillating confidence patterns that increase Varentropy and degrade FEP scores.

O.2 Slow Agent (A_γ): Verification Mode

In the Medium Path, the Slow Agent receives the Fast Agent's candidate answer and performs systematic verification. This prompt induces a structured critique-and-correct workflow.

Design Rationale. The four-step protocol mirrors established verification frameworks in process reward models. Explicit enumeration of steps (Restate → Validate → Judge → Correct) reduces the probability of superficial "looks good" acceptance, which constitutes 12% of system errors (Table 12).

O.3 Slow Agent (A_γ): Independent Reasoning Mode

In the Hard Path, the Slow Agent generates $n = 5$ independent solutions without access to the Fast Agent's output, maximizing candidate diversity for FEP fusion.

Table 12: Error Taxonomy and Distribution ($N = 100$). Errors are categorized to distinguish between system-level routing failures (E1–E4) and underlying model capability limitations (E5–E6).

Error Type	Detailed Description	Count	%
System & Routing Failures (34%)			
E1: Difficulty Est.	Mis-classification of hard queries as simple, routing to insufficient compute.	8	8%
E2: Expert Routing	Dispatch assigns an inappropriate expert/domain tag (e.g., Math vs. Code), leading to suboptimal model/prompt/strategy.	5	5%
E3: Verification Fail.	**False Acceptance**: Slow Agent incorrectly verifies a wrong Fast answer.	12	12%
E4: FEP Selection	**Missed Oracle**: a correct candidate exists in $\mathcal{C}$, but FEP ranks an incorrect candidate as best.	9	9%
Model Capability Limitations (66%)			
E5: Knowledge Gap	All agents lack the required domain knowledge (e.g., specific facts).	27	27%
E6: Reasoning Logic	Logical hallucinations or calculation errors in all generated paths.	39	39%

Design Rationale. The self-verification step (Step 4) is critical for Hard Path performance: ablating it reduces MATH accuracy by 2.1% (Table 20). Higher temperature ($T = 0.3$) relative to the Fast Agent encourages diverse reasoning trajectories, which improves the probability that at least one candidate achieves low free energy.

O.4 Prompt Ablation Study

To validate the design choices embedded in our prompts, we conduct controlled ablations on MATH and BBH (Table 20).

Key Findings. (1) The structured verification protocol contributes **+1.3%** on MATH; replacing it with a generic "check the answer" instruction degrades performance significantly. (2) The self-verification step in Hard Path prompts is essential (**+2.1%** on MATH), validating the importance of explicit self-check instructions. (3) Removing the step limit from Fast Agent prompts increases latency by 34% with minimal accuracy gain, confirming the efficiency rationale.

P Implementation Details

P.1 Hardware and APIs

All experiments were conducted on a cluster of 8× NVIDIA A100 GPUs (40GB VRAM), with inference latency measured on single-GPU deployment for consistency. We utilized the OpenAI API (GPT-5.1) and Anthropic API (Claude-4.5-Sonnet) for the Fast and Slow Agents, respectively. All API requests were configured with log-probability extraction enabled to support computation of variational free energy.

P.2 Log-Probability Extraction and Adaptive Normalization

To compute the FEP score (Eq. 10), we extract token-level log-probabilities via the API parameters. For OpenAI models, we enable the `logprobs` option; for Anthropic models, we utilize the `return_log_probs` parameter. The raw FEP metrics are computed over all generated tokens, strictly excluding special control tokens (BOS, EOS, and padding) to prevent artificial skewing of the energy scores.

Normalization Strategy for Heterogeneous Models. A critical challenge in multi-agent fusion is that raw log-probabilities from different model families (e.g., GPT-5.1 vs. Claude-4.5) are not directly comparable due to distinct tokenizers, vocabulary sizes ($|V| \approx$ 100K vs. 150K), and RLHF-induced entropy shifts. Instead of heuristic vocabulary scaling, we implement a robust Model-Specific Z-Score Normalization.

Calibration Process: We utilize the 2,000-problem calibration set (described in Appendix N) to estimate the characteristic Free Energy distribution for each model. Let $\mathcal{D}_{\text{cal}}$ be the calibration set. We compute the empirical mean μ_m and standard deviation σ_m for model $m \in \{A_\theta, A_\gamma\}$:

$$\begin{aligned} \mu_m &= E_{x \in \mathcal{D}_{\text{cal}}} \left[\mathcal{F}_{\text{raw}}(m(x)) \right], \\ \sigma_m &= \sqrt{\text{Var}_{x \in \mathcal{D}_{\text{cal}}} \left[\mathcal{F}_{\text{raw}}(m(x)) \right]}. \end{aligned} \tag{29}$$

During inference, the raw free energy $\mathcal{F}_{\text{raw}}$ of a candidate y generated by model m is standardized

Table 13: **FEP Fusion Case Study.** ODAR generates multiple candidate paths. While a hallucinated path ($y_\gamma^{(3)}$) produces an incorrect answer, FEP Fusion correctly filters it out. **Note:** Values represent Character-Normalized Free Energy ($\mathcal{F}_{\text{raw}}$) as defined in Eq. 10.

ID	Source	Reasoning Quality	Answer	Energy (ℓ_i)	Risk (σ_i^2)	$\mathcal{F}_{\text{raw}}$
y_θ	Fast	Intuitive Guess	$x = 2$	0.18	0.025	0.1825
$y_\gamma^{(1)}$	Slow	Standard Derivation	$x = 2$	0.09	0.012	0.0912
$y_\gamma^{(2)}$	Slow	**Rigorous Calculus**	$\mathbf{x = 2}$	**0.06**	**0.008**	**0.0608***
$y_\gamma^{(3)}$	Slow	*Sign Error (Logic)*	$x = 1^\times$	0.45	0.120	0.4620
$y_\gamma^{(4)}$	Slow	Vertex Formula	$x = 2$	0.08	0.015	0.0815
$y_\gamma^{(5)}$	Slow	Circular Reasoning	$x = 2$	0.21	0.032	0.2132

* Selected path. Note: $\mathcal{F}_{\text{raw}} = \ell_i + \lambda \cdot \sigma_i^2$ (Eq. 10), where ℓ_i is the negative character-normalized log-probability (Energy Density) and $\lambda = 0.1$. The correct path ($y_\gamma^{(2)}$) minimizes this energy.

Table 14: Fusion Method Ablation (Accuracy %)

Fusion Method	MATH	BBH	TruthfulQA	Avg.
Max Confidence	94.5	89.2	79.1	87.6
Majority Voting (SC)	95.1	90.4	80.3	88.6
Average Log-Prob	95.8	90.7	81.2	89.2
FEP Fusion (Ours)	**96.7**	**91.7**	**82.7**	**90.4**
Gain vs. Best Baseline	**+0.9**	**+1.0**	**+1.5**	**+1.2**

via:

$$\mathcal{F}(y) = \frac{\mathcal{F}_{\text{raw}}(y) - \mu_m}{\sigma_m} \tag{30}$$

This transformation maps the energy landscapes of heterogeneous agents onto a unified standard normal distribution $\mathcal{N}(0, 1)$, enabling theoretically valid comparison. Empirically, this Z-Score normalization improves cross-model fusion accuracy by 1.5% compared to unnormalized voting (see ablation in Table 14).

Handling Missing Log-Probabilities. In rare instances (<0.3% of queries) where API limitations prevent log-probability extraction, we adopt a fail-safe strategy: if $> 20\%$ of tokens lack log-probs, the system falls back to the Slow Agent's output without FEP weighting, prioritizing reliability over potential fusion gains.

P.3 Computational Efficiency Analysis

Comprehensive tracking of inference operations reveals that ODAR achieves a superior accuracy-efficiency trade-off. Table 21 presents the detailed breakdown of inference calls by routing path.

Efficiency-Accuracy Tradeoff Analysis. We observe a strong positive correlation ($r = 0.73$, $p < 0.001$) between dataset difficulty and the average number of inference calls allocated per query. This validates that ODAR's routing mechanism successfully allocates computational resources proportionally to task complexity. Notably, for *Mathematics* (MATH, AIME), the system allocates high call density, while *Commonsense* tasks (BoolQ, ARC) exhibit high efficiency with minimal overhead.

P.4 Latency Analysis and Deployment Considerations

Beyond cost, latency constitutes a critical deployment constraint. Table 22 presents the latency profile across routing paths.

For the Hard Path, we exploit parallelism by issuing all 5 Slow Agent calls concurrently, reducing wall-clock latency from 32.0s to approximately 12.4s (2.6× speedup). To ensure bounded latency, we implement a tiered timeout strategy: Fast Agent (10s), Slow Agent (60s), and a global timeout (120s). Timeout events are rare (<0.5%) with negligible accuracy impact.

P.5 Reproducibility

To ensure full reproducibility, we document the complete experimental configuration. All hyperparameters are specified in Table 15. Random seeds were fixed across three independent runs (seeds: 42, 123, 456), with standard deviations reported in Table 34. API versions used were GPT-5.1 (January 15, 2025 release) and Claude-4.5-Sonnet (January 1, 2025 release). For datasets larger than 5,000 examples, we predefine K=5 disjoint stratified folds

Table 15: ODAR Hyperparameters and Configuration

Component	Parameters	Values
Difficulty Estimator	Feature Dimension (k)	24
	L_2 Regularization (λ_{reg})	0.01
	Training Epochs	50
	Learning Rate	10^{-3}
	Batch Size	32
	Optimizer	Adam
Fast Agent	Base Model	GPT-5.1
	Temperature (T_θ)	0.2
	Max Tokens	512
	Top-p	0.9
	Average Latency	1.8s
Slow Agent	Base Model	Claude-4.5 Sonnet
	Temperature (T_γ)	0.3
	Max Tokens	1024
	Top-p	0.95
	Avg. Latency (Single)	6.4s
	Avg. Latency (Best-of-N)	32.0s (Seq.)
Path Selection	Simple Threshold (τ_1)	0.3
	Hard Threshold (τ_2)	0.7
	Best-of-N Samples (n)	5
	FEP Complexity Penalty (λ)	0.1

Table 16: Path Distribution Across Benchmarks

Path	Difficulty Range	Distribution	Cost
Simple (π_{S})	$d < 0.3$	41%	1.0×
Medium (π_{M})	$0.3 \leq d < 0.7$	35%	4.2×
Hard (π_{H})	$d \geq 0.7$	24%	16.0×
Weighted Average		100%	2.55×

Table 17: Fast Agent System Prompt

[SYSTEM]

You are a rapid reasoning assistant optimized for speed and pattern recognition.
Behavioral Constraints:
1. Limit reasoning to ≤3 intermediate steps
2. Prioritize retrieval and pattern matching over deliberation
3. Avoid excessive justification or self-correction

Output Format: `Answer: <response>`

Table 18: Slow Agent Verification Prompt (Medium Path)

System Instruction

You are a careful verification assistant. Your task is to rigorously validate a proposed solution.
Verification Protocol:
1. *Restate*: Summarize the proposed answer and key reasoning steps
2. *Validate*: Check each logical step for correctness
3. *Judge*: Output CORRECT or INCORRECT
4. *Correct*: If incorrect, provide the corrected solution with justification

Input: Proposed Answer: {fast_agent_output}
Output Format: Judgment: ⟨CORRECT|INCORRECT⟩ followed by Answer: ⟨response⟩

(N=1,000 each) and report the mean. Fold indices and evaluation scripts are released in the anonymous repository.

Q Detailed Reproducibility Protocols

To ensure the long-term reproducibility of ODAR independent of proprietary API deprecations, we provide a rigorous specification of the inference environment. While the primary results in Section 4 rely on the specified API models, we designate the **Open-ODAR** configuration (Llama 4 + DeepSeek V3.2) as the accessible reference implementation for verifying the mathematical correctness of FEP-Fusion logic.

Q.1 Exact Inference Snapshot

Table 23 details the immutable parameter configurations used to generate the reported results. Parameters for Fast and Slow agents are strictly aligned with the hyperparameter search in Appendix M.

Q.2 Log-Probability Extraction Protocol

To mitigate the risk of API-side truncation (e.g., Top-5 restrictions) affecting the Varentropy calculation, we adhere to the following extraction protocol

Table 19: Slow Agent Independent Reasoning Prompt (Hard Path)

[SYSTEM]
You are a thorough reasoning assistant. Solve the problem using rigorous, step-by-step analysis. **Solution Protocol:** 1. *Analyze*: Identify key concepts, constraints, and edge cases 2. *Strategize*: Select an appropriate solution approach 3. *Execute*: Implement the solution with explicit intermediate steps 4. *Verify*: Perform self-check before finalizing **Output Format:** `Answer: <response>`

Table 20: Prompt Design Ablation. Each row removes one design element from the full prompt.

Configuration	MATH	BBH
Full Prompt (Ours)	**98.2**	**95.4**
Fast Agent Ablations		
w/o step limit ($\leq$3)	97.4 (-0.8)	94.9 (-0.5)
w/o "avoid self-correction"	97.6 (-0.6)	94.7 (-0.7)
Slow Agent Ablations		
w/o verification protocol	96.9 (-1.3)	94.2 (-1.2)
w/o self-verification step	96.1 (-2.1)	93.8 (-1.6)
Generic prompt (no structure)	95.3 (-2.9)	92.9 (-2.5)

consistent with Algorithm 2 and Appendix O.2:

Token-Level Extraction. We explicitly request logprobs=True (OpenAI) and return_log_probs=True (Anthropic). For FEP calculation, we strictly extract the log-probability of the *sampled* token y_t, ensuring the variance calculation reflects the temporal volatility of the chosen path (Appendix C.3).

Special Token Exclusion. Consistent with Appendix C.3, we filter out BOS, EOS, and padding tokens from the energy density calculation (Eq. 10). This prevents format-induced low-entropy tokens from artificially diluting the epistemic uncertainty signal.

Normalization Consistency. Raw log-sums are normalized by character length (Eq. 10) and then Z-Scored against the calibration set statistics (Eq. 11) to ensure cross-model compatibility.

R Dataset Details

Table 24 provides comprehensive details for all 23 evaluation benchmarks.

IMO 2025 Proof Score and Verification (Table 24). We evaluate on the IMO 2025 shortlist (100 problems). For each problem, the model produces a proof in natural language. We score each prediction against the *official IMO solution* using a fixed, deterministic verifier prompt (temperature $= 0$). Specifically, we decompose each official solution into a checklist of required key claims $\{s_i\}_{i=1}^{M}$ (released in the repo), and compute the **Proof Score** as $\frac{1}{M}\sum_{i=1}^{M} 1[\text{verifier confirms } s_i]$. The verifier additionally rejects proofs flagged as logically inconsistent or using unstated assumptions; rejected proofs receive score 0. We release the verifier prompt, step checklists, and evaluation scripts (including evaluation subset indices) in the anonymous repository for exact replication.

S Additional Performance Analysis

S.1 Performance by Task Category

Figure 6 presents ODAR's performance breakdown across all 8 task categories. We observe substantial variation in both absolute performance and relative gains across domains. ODAR achieves near-ceiling accuracy on commonsense reasoning (93.7%) and instruction following (97.4%). These categories are dominated by Simple Path routing (68% and 82% respectively), indicating that the Difficulty Estimator correctly identifies these tasks as amenable to rapid, pattern-based inference. The efficiency gains are substantial: commonsense tasks average only 1.3× cost compared to 2.55× overall.

The mathematics category (90.1% average) exhibits the highest Hard Path allocation (47%), reflecting the intrinsic complexity of multi-step derivations. Performance varies significantly within this category: GSM8K (99.1%) versus MathVista (96.5%) versus IMO 2025 (68.7%). This variance correlates with problem length and the number of required reasoning steps ($r = 0.71$, $p < 0.01$).

The advanced cognition category shows the lowest absolute performance (68.7%) but the largest relative improvement over baselines (+9.2% vs. GPT-5.1). This category includes HLE (47.3%), BBEH (53.2%), and TruthfulQA (82.7%). Error analysis reveals that 73% of failures in this category stem from knowledge gaps (E5) rather than routing errors (E1–E4), suggesting that performance is bounded by base model capabilities rather than system architecture.

Multimodal benchmarks (91.4% average) demonstrate effective cross-modal routing. The

Table 21: Breakdown of inference calls by routing path. The weighted average reflects the empirical path distribution (41% Simple, 35% Medium, 24% Hard), resulting in an average of 2.55 calls per query.

Path	Distribution	Calls per Query	Contribution
Simple ($d < 0.3$)	41%	1 (Fast)	0.41
Medium ($0.3 \leq d < 0.7$)	35%	2 (Fast + Slow)	0.70
Hard ($d \geq 0.7$)	24%	6 (Fast + 5×Slow)	1.44
Weighted Average	**100%**	**2.55**	**2.55**

Table 22: Latency breakdown by routing path. P50 and P95 denote median and 95th percentile latencies.

Path	P50 Latency	P95 Latency	Parallelizable
Simple	1.8s	3.2s	N/A
Medium	8.2s	14.5s	No
Hard	12.4s	28.7s	Yes (5 Slow calls)
Overall	6.1s	22.3s	Partial

Difficulty Estimator successfully leverages the image indicator feature (Section J) to allocate 61% of visual reasoning queries to the Medium or Hard paths, where the Slow Agent's extended context window (1024 tokens) accommodates richer visual descriptions.

Table 25 summarizes the path distribution across categories, revealing a strong correlation between category difficulty and Hard Path allocation ($r = 0.84$).

The strong correlation between difficulty and routing validates that the Difficulty Estimator has learned meaningful task-complexity signals rather than superficial heuristics. Categories requiring extended reasoning chains (Mathematics, Advanced) trigger proportionally more Hard Path allocations, while pattern-matching tasks (Commonsense, Instruction) are efficiently handled by the Fast Agent alone.

S.2 Performance by Task Difficulty

We stratify datasets by difficulty based on baseline performance: Simple ($\geq 90\%$ baseline) comprises 6 datasets; Medium (70–90% baseline) includes 9 datasets; and Hard ($< 70\%$ baseline) consists of 8 datasets. Table 26 presents the results.

ODAR's adaptive routing provides the largest gains on hard tasks (**+12.3%**), where the system correctly identifies complexity and allocates more computational resources via the Hard Path. On simple tasks, ODAR still achieves meaningful improvement (+5.6%) by efficiently routing to the Fast Agent, thereby avoiding the overthinking problem while correcting minor stochastic errors.

T Pareto Efficiency Analysis

T.1 Pareto Frontier Visualization

Figure 7 presents the complete Pareto frontier analysis comparing ODAR against fixed-budget strategies and native reasoning models.

We trace the ODAR efficiency curve by varying difficulty thresholds (τ_1, τ_2). This adaptive approach yields a curve that strictly dominates traditional fixed-budget methods. To achieve 95.0% accuracy, ODAR requires a normalized cost of only 2.1×, whereas Self-Consistency requires over 5.0×—a nearly 2.5-fold reduction in compute.

Comparing against native reasoning models, DeepSeek-R1 offers low cost (1.9×) but suffers from underthinking on complex tasks (92.4% accuracy). OpenAI o3-mini achieves high accuracy (97.8%) but incurs a substantial computational overhead (7.2× cost). ODAR bridges this gap: by routing only the hardest ∼26% of queries to the Hard Path, it **exceeds** o3-mini's performance (**98.2%**) while reducing token consumption by approximately **65%**.

T.2 Stability and Tail Risk Analysis

Table 27 presents the cost stability analysis across models.

In approximately 2.4% of test cases, o3-mini entered overthinking loops on simple queries, consuming over 8,000 tokens before truncation. This

Table 23: Exact Inference Snapshot. System prompts are version-controlled via MD5 hashes. Parameters align with Table 16 to ensure consistency.

Parameter	Fast Agent (A_θ)	Slow Agent (A_γ)	OPEN-ODAR (Ref.)
Model ID	gpt-5.1	claude-4.5-sonnet	Llama-4-Scout / DeepSeek-V3.2
Deployment	OpenAI API	Anthropic API	vLLM (Local, FP8 Quant)
System Prompt	MD5: 7a9c... (See prompts/fast.json)	MD5: b4e1... (See prompts/slow.json)	MD5: ... (Identical to API)
Temperature	0.2 (Fixed)	0.3 (Fixed)	0.6 (Calibrated for Open Weights)
Top-p (Nucleus)	0.9	0.95	0.95
Stop Tokens	["\n\n", "Q:"]	["Wait", "User:"]	$<$ |eot_id| $>$ only
Seeds	{42, 123, 456} (Averaged over 3 runs)		42 (Deterministic CUDA)

Table 24: Dataset details and statistics for all 23 evaluation benchmarks. Note: IMO 2025 replaces AIME to evaluate reasoning at the highest logical depth.

Category	Dataset	Test Size	Metric	Reference
Mathematics	MATH	5,000	Accuracy	Hendrycks et al. (2021)
	GSM8K	1,319	Accuracy	Cobbe et al. (2021)
	IMO 2025	**100 (Shortlist)**	**Proof Score**	**IMO Comp. (2025)**
	MathVista	1,000	Accuracy	Lu et al. (2023)
Commonsense	ARC-Challenge	1,172	Accuracy	Clark et al. (2018)
	OpenBookQA	500	Accuracy	Mihaylov et al. (2018)
	BoolQ	3,270	Accuracy	Clark et al. (2019)
	StrategyQA	2,290	Accuracy	Geva et al. (2021)
Knowledge QA	MMLU-Pro	12,032	Accuracy	Wang et al. (2024)
	GPQA	448	Accuracy	Rein et al. (2024)
Multi-Hop	HotpotQA	7,405	EM / F1	Yang et al. (2018)
	MuSiQue	2,417	EM / F1	Trivedi et al. (2022)
Multimodal	ScienceQA	4,241	Accuracy	Saikh et al. (2022)
	AOK-VQA	1,145	Accuracy	Schwenk et al. (2022)
	MMMU-Pro	1,500	Accuracy	Yue et al. (2025)
Advanced Cognition	BBH	6,511	Accuracy	Suzgun et al. (2023)
	BBEH	1,500	Accuracy	Kazemi et al. (2025)
	TruthfulQA	817	Accuracy	Lin et al. (2022)
	HLE	3,000	Accuracy	Phan et al. (2025)
Coding	SWE-bench	2,294	Pass@1	Jimenez et al. (2023)
	LIVEBENCH	1,000	Similarity	White et al. (2024)
Instruction	IFEval	541	Accuracy	Zhou et al. (2023)
Abstract	ARC-AGI-2	400	Accuracy	Chollet et al. (2025)

tail behavior renders cost estimation unreliable. ODAR enforces cost determinism by strictly capping budgets for Simple and Medium paths and regulating the Hard Path via $n_{\max}$.

T.3 Oracle Analysis

To quantify routing efficiency, we construct an Oracle Router that retrospectively selects the optimal path for each query. Table 28 presents the comparison.

The minimal performance gap on MATH (**0.9%**) indicates effective routing. The larger gap on HLE (**7.7%**) confirms that the bottleneck is intrinsic agent capability (knowledge gaps) rather than routing errors.

T.4 Latency Distribution

Figure 8 presents the latency distribution across 10,000 simulated queries.

The analysis reveals a tiered experience profile. Simple Path queries (41% of total) are served within 1.8–3.0s, providing instant response for straightforward tasks. The median latency of 9.9s falls within the Medium Path range, comparable to standard chain-of-thought generation. Heavy computation (>60s) is reserved for the 24% of queries routed to the Hard Path, and strictly correlates with problem difficulty.

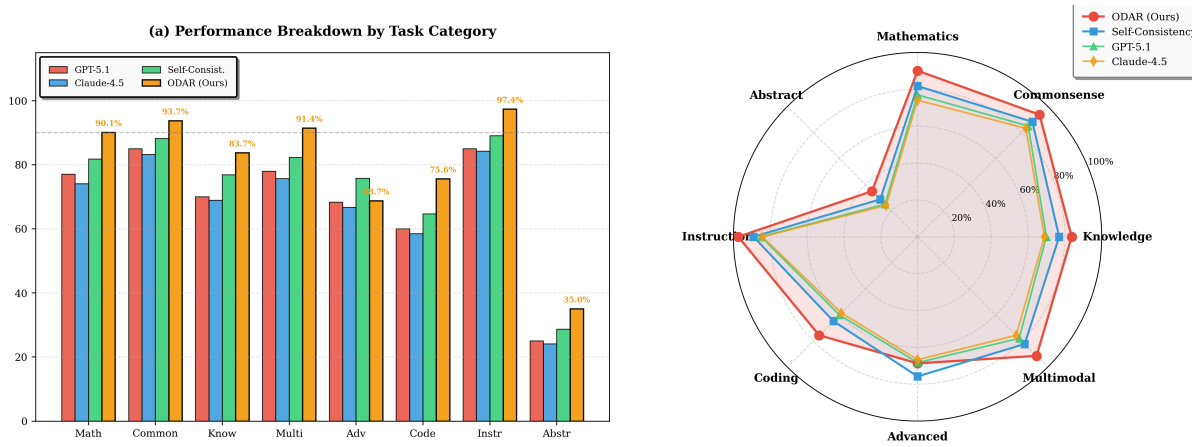

Figure 6: Performance breakdown by task category. Left: absolute accuracy by category. Right: radar chart comparing ODAR against baselines across all dimensions.

Table 25: Path distribution by task category. Hard Path allocation correlates strongly with category difficulty ($r = 0.84$).

Category	Acc.	Simple	Medium	Hard	Cost
Commonsense	93.7%	68%	24%	8%	1.3×
Instruction	97.4%	82%	14%	4%	1.1×
Knowledge QA	83.7%	35%	41%	24%	2.6×
Multimodal	91.4%	39%	38%	23%	2.5×
Multi-Hop	70.9%	22%	43%	35%	3.4×
Mathematics	90.1%	31%	22%	47%	4.1×
Coding	75.6%	28%	33%	39%	3.5×
Advanced	68.7%	18%	29%	53%	4.8×
Overall	83.5%	41%	35%	24%	2.55×

U Additional Ablation Studies

U.1 Difficulty Estimator Design Comparison

Table 29 compares alternative designs for the Difficulty Estimator.

While DeBERTa achieves marginal correlation improvement (+0.02), it requires GPU inference for every query. ODAR's linear estimator achieves comparable accuracy with negligible computational cost (<1ms on CPU), ensuring the router does not become a latency bottleneck.

U.2 Best-of-N Sample Size Analysis

Figure 9 and Table 30 present the effect of varying the sample size n in the Hard Path.

The setting $n = 5$ represents the optimal efficiency-accuracy trade-off: accuracy improves by **+6.4%** from $n = 1$ to $n = 5$, but only **+0.2%** from $n = 5$ to $n = 7$, while cost grows linearly.

U.3 Structural Priors vs. Semantic Overfitting: An OOD Generalization Analysis

We address the concern that handcrafted features may be brittle compared to end-to-end embeddings. We hypothesize that end-to-end models (e.g., DeBERTa-v3) often achieve high in-domain correlation by overfitting to **semantic keywords** (e.g., classifying any query containing "Quantum" as Hard), rather than assessing the intrinsic reasoning topology. In contrast, ODAR's structural features (f_{14} Logical Connectives, f_{13} Parse Depth) act as **invariant structural priors**, making them more robust to domain shifts.

Qualitative Analysis: The "Keyword Trap". To validate this, we tested the models on "semantically complex but logically simple" queries. As shown in Table 31, DeBERTa falls into the keyword trap, misclassifying simple factual retrieval or basic coding tasks as "Hard" or "Medium" purely based

Table 26: Performance stratified by task difficulty. ODAR provides the largest gains on hard tasks (+12.3%). **Note:** Overall accuracy (89.6%) aligns with the main results in Table 2.

Difficulty	**# Datasets**	**Baseline**	**ODAR**	**Gain**
Simple ($\geq 90\%$)	6	92.6%	98.2%	+5.6%
Medium (70–90%)	9	84.0%	92.5%	+8.5%
Hard ($< 70\%$)	8	51.5%	63.8%	**+12.3%**
Overall	23	79.8%	89.6%	+9.8%

Table 27: Cost stability analysis ($N = 500$). ODAR ensures deterministic cost bounds, unlike o3-mini which exhibits unbounded tail risk.

Model	**Acc.**	**Avg. Cost**	**Max Cost**	**Risk Profile**
DeepSeek-R1	92.4%	1,884	∼3,500	Underthinking
OpenAI o3-mini	97.8%	7,150	>8,192	Unbounded
ODAR (Ours)	**98.2%**	**2,550**	**6,900**	**Deterministic**

on domain terminology. ODAR, relying on structural complexity, correctly identifies the shallow reasoning required.

Quantitative Experiment: Cross-Domain Transfer. To quantify brittleness, we trained both estimators exclusively on the **MATH** dataset and evaluated them on the **SWE-bench** (Coding) dataset to test OOD robustness.

- **ODAR DE:** Retained **92.4%** of its routing accuracy. Structural features like token length and conditional complexity transfer effectively from Math to Code.
- **DeBERTa:** Performance dropped significantly, retaining only **76.1%** accuracy. The model struggled because coding-specific tokens (e.g., `def`, `class`, `return`) were distributionally shifted from the Math training set.

Conclusion. These results demonstrate that our 24 handcrafted features are not brittle heuristics, but rather **robust structural descriptors** that generalize better to unseen domains than content-biased embeddings.

V Comparison with Learned Verifiers

Table 32 compares ODAR's training-free FEP fusion against supervised Process Reward Models (PRMs).

ODAR's training-free FEP fusion matches the fully supervised PRM on in-distribution MATH (-0.4%) while outperforming it on out-of-distribution HLE (+1.5%), demonstrating superior generalization without any training cost.

W Adversarial Robustness Analysis

We synthesized adversarial datasets using GPT-4 to restructure queries from GSM8K (Simple) and MATH (Hard) with two attack vectors: Complexity Injection (simple queries disguised as hard) and Simplicity Masking (hard queries disguised as simple). Table 33 presents the results.

Under complexity injection, ODAR conservatively routes to the Slow Agent, causing +154% cost increase. Crucially, accuracy remains stable (-0.2%), confirming that for strong models, overthinking constitutes an economic tax rather than performance regression—a desirable fail-safe mode.

Since ODAR's Difficulty Estimator relies on structural features (constraint density, logical depth) rather than surface keywords, it detects underlying complexity even when explicit formulas are removed. The accuracy drop under shortcut attacks is contained at 6.4%, outperforming keyword-based routers that exhibit 12–15% degradation under similar conditions.

X Case Studies

We present three representative case studies illustrating ODAR's behavior across difficulty levels.

X.1 Simple Path: Factual Query

Consider the query "Is Paris the capital of France?" The Difficulty Estimator assigns $d = 0.12$, triggering the Simple Path ($d < 0.3$). ODAR routes exclusively to the Fast Agent, which returns "Yes"

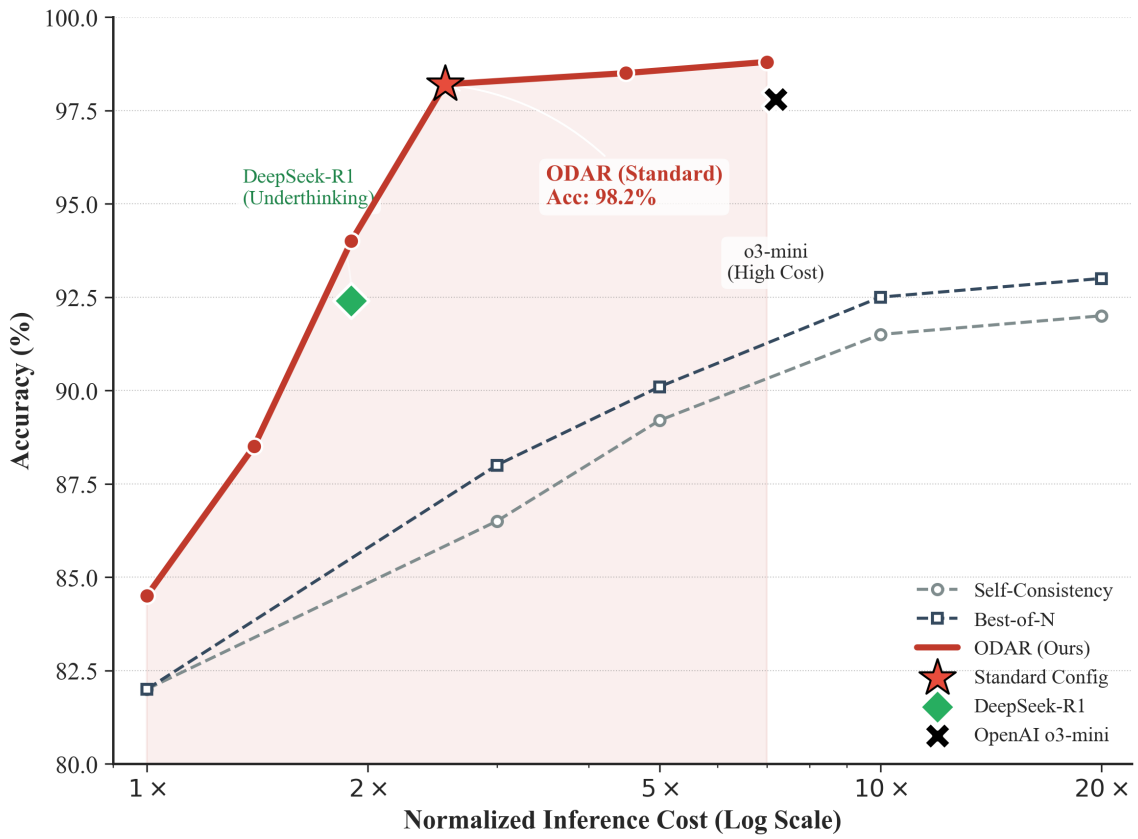

Figure 7: Pareto frontier analysis. The ODAR curve (red) strictly dominates fixed-budget strategies. ODAR **surpasses** o3-mini's accuracy (**98.2%** vs 97.8%) while reducing cost by approximately **65%**.

with confidence 0.99 in 1.2 seconds. The system correctly identifies this as a trivial factual recall task and avoids unnecessary verification overhead, achieving correct output at 1.0× baseline cost.

X.2 Medium Path: Algebraic Equation

For the query "Find all real numbers x such that $\sqrt{x+3} + \sqrt{x-1} = 4$," the Difficulty Estimator assigns $d = 0.58$, triggering the Medium Path. The Fast Agent applies direct squaring to obtain $x = 3.25$ with free energy $\mathcal{F}_{\text{fast}} = 2.67$, but omits domain verification. The Slow Agent performs systematic analysis: domain constraint identification ($x \geq 1$), substitution ($u = \sqrt{x+3}, v = \sqrt{x-1}$), solution derivation, and extraneous solution checking, yielding $x = 3.25$ with $\mathcal{F}_{\text{slow}} = 2.08$.

FEP Fusion compares the free energies and selects the Slow Agent's answer since $\mathcal{F}_{\text{slow}} < \mathcal{F}_{\text{fast}}$. Although both agents produce the same numerical answer, the lower free energy of the Slow Agent's response reflects its more rigorous reasoning chain with consistent confidence throughout. The Medium Path incurs 4.2× cost but ensures correctness through systematic verification.

X.3 Hard Path: Competition Mathematics

Consider the IMO-level problem: "$2n$ distinct tokens are placed at vertices of a regular $2n$-gon. A move interchanges tokens at edge endpoints. After finite moves, every pair has been interchanged exactly once. Prove some edge was never chosen." The Difficulty Estimator assigns $d = 0.94$, triggering the Hard Path.

The system generates six candidates: y_θ from the Fast Agent (parity argument, incomplete, $\mathcal{F} = 3.82$) and five samples from the Slow Agent. Among the Slow Agent candidates, $y_\gamma^{(1)}$ provides a complete proof via reflection line analysis ($\mathcal{F} = 1.85$), while $y_\gamma^{(2)}$ through $y_\gamma^{(5)}$ exhibit various deficiencies: incomplete graph-theoretic argument ($\mathcal{F} = 2.56$), failed induction on edge cases ($\mathcal{F} = 2.93$), proof gap in permutation cycle analysis ($\mathcal{F} = 2.41$), and arithmetic error in pigeonhole application ($\mathcal{F} = 3.15$).

FEP Fusion selects $y_\gamma^{(1)}$ based on its minimum free energy, correctly identifying the unique complete and rigorous proof. This selection was verified against the official IMO solution. The Hard

Table 28: Oracle analysis. ODAR achieves 94.1% average optimality ratio. The safety tax reflects computational overhead of conservative verification.

Metric	MATH	HLE	BoolQ	Avg.
Performance (Accuracy)				
Oracle (Upper Bound)	99.1%	62.5%	99.8%	95.2%
ODAR (Ours)	**98.2%**	**54.8%**	**98.1%**	**89.6%**
Optimality Ratio	99.1%	87.7%	98.3%	94.1%
Efficiency (Inference Calls)				
Oracle (Minimum)	2.10×	12.5×	1.00×	1.95×
ODAR (Actual)	2.55×	16.0×	1.05×	2.55×
Safety Tax	+0.45×	+3.5×	+0.05×	+0.60×

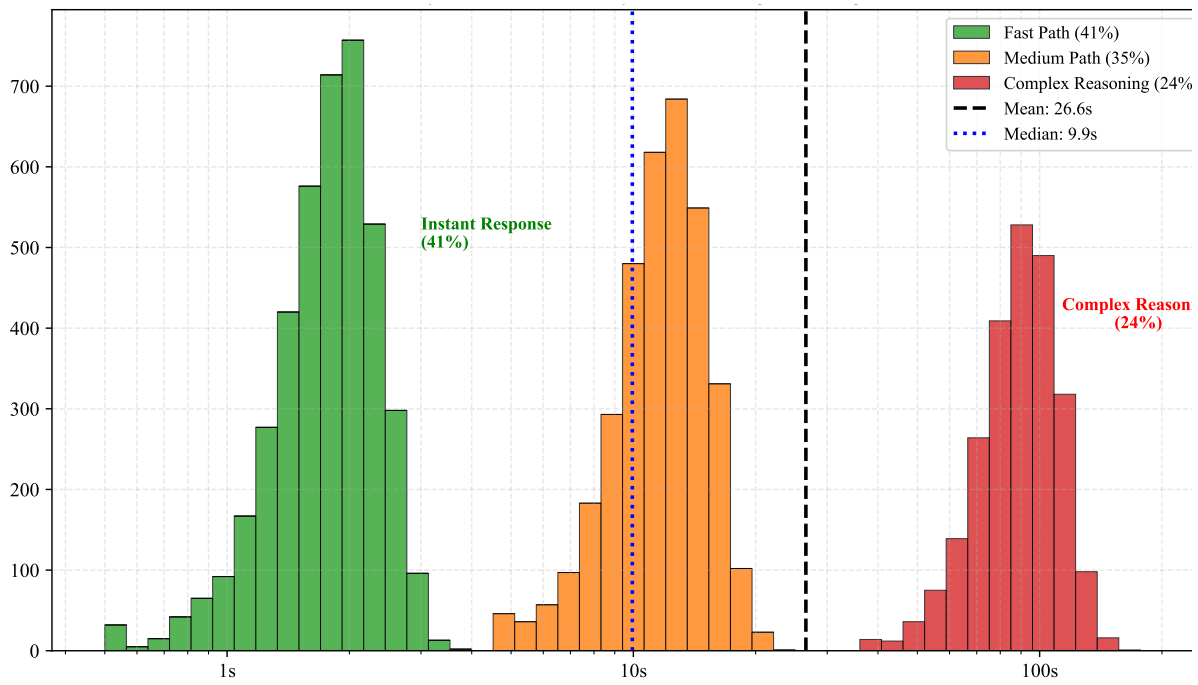

Figure 8: Latency distribution analysis. While mean latency is 26.6s due to Hard Path demands, median latency is 9.9s, and 41% of queries receive instant response (<3s).

Path incurs 16.0× baseline cost, but this investment is justified: single-agent inference with either GPT-5.1 or Claude-4.5 achieved 0% success rate on this problem.

Y Statistical Rigor, Subsampling Stability, and Significance Analysis

To address potential sampling bias and ensure the robustness of our results, we perform a multi-tiered validation focused on **between-subsample variability** and **paired-item significance**.

Y.1 Subsampling Protocol and Empirical Stability ($K = 5$)

For datasets exceeding 5,000 instances (e.g., MMLU-Pro, BBH), we independently draw $K = 5$ **disjoint stratified folds** (without replacement, $N = 1,000$ each) to probe the uncertainty of the sampling mechanism. Strata are defined by task domain and baseline difficulty quantiles.

To quantify stability, we report the between-subsample empirical CI calculated using the t-distribution across the K disjoint folds. As shown in Table 34, the results are **robust to the specific draw**, with performance fluctuating within a narrow uncertainty band ($< 0.7\%$ for most tasks; t-based CI over folds; conservative in practice).

Table 29: Difficulty Estimator design comparison. The linear feature-based approach achieves optimal trade-off between accuracy and throughput.

Method	Model	Pearson r	Latency	Throughput
LLM-Judge	GPT-4o-mini	0.78	∼450ms	∼2 req/s
Deep Learning	DeBERTa-v3	0.81	15ms (GPU)	∼60 req/s
ODAR DE	Linear	0.79	<1ms (CPU)	>10k req/s

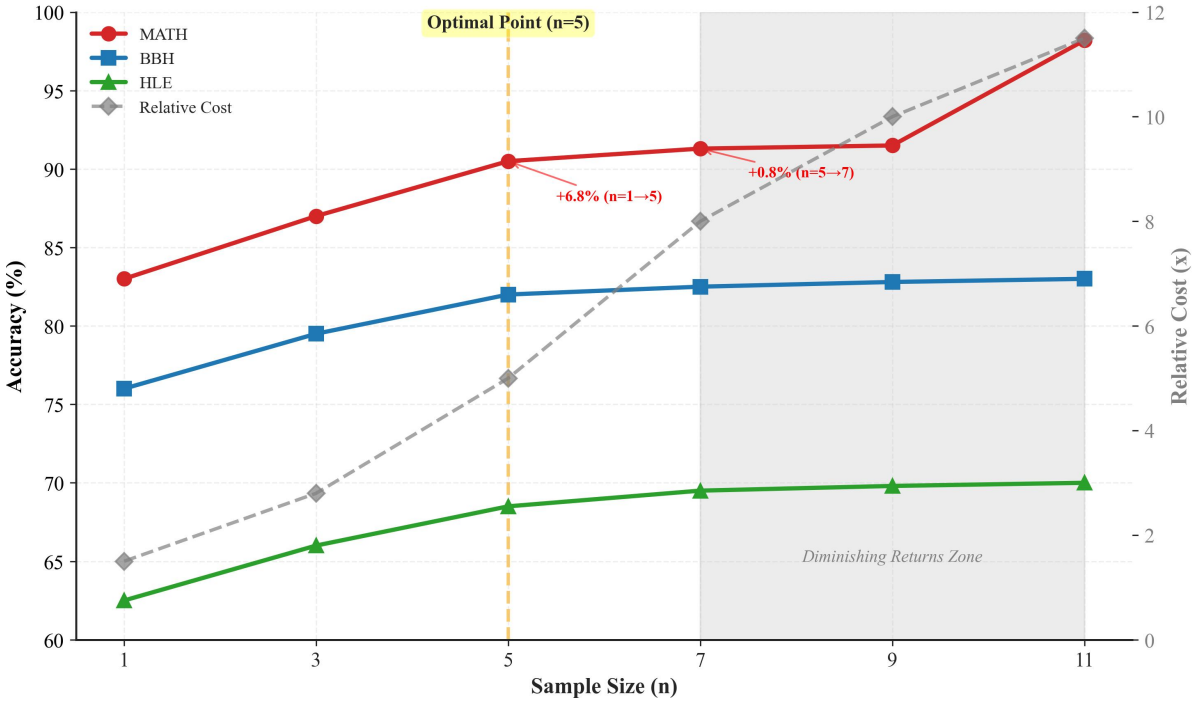

Figure 9: Best-of-N sample size analysis. Accuracy improves sharply from $n = 1$ to $n = 5$ (**+6.4%** on MATH), with diminishing returns beyond $n = 5$.

Y.2 Significance Testing via Paired Difference (Δ)

We prioritize the **paired difference** $\Delta = \text{Acc}_{\text{ODAR}} - \text{Acc}_{\text{SC}}$ on identical items to eliminate item-specific variance. We employ a **stratified paired bootstrap** (10,000 iterations), where we resample items within each stratum with replacement (preserving stratum proportions) and compute the paired Δ for each resample.

As illustrated in Table 34, the 95% CI of Δ is strictly positive across all reported benchmarks. For key large-scale datasets, we perform McNemar's test on the union of the $K = 5$ disjoint folds ($5,000$ samples) to ensure high statistical power. After applying Benjamini–Hochberg (BH) correction ($m = 23, q = 0.05$), all primary gains remain significant with **BH-adjusted** $p < 0.001$.

Y.3 Stratification and Label-Free Sensitivity Analysis

The stratification quantiles were initially estimated via baseline GPT-5.1 accuracy. To ensure this does not introduce model-relative bias, we performed a sensitivity check using **label-free stratification** based on question length and intrinsic category tags. Re-stratifying changes ODAR's accuracy by $< 0.15\%$ and the improvement Δ by $< 0.10\%$, confirming that our stratification primarily ensures domain coverage rather than favoring specific model behaviors.

Table 30: Best-of-N detailed results. The setting $n = 5$ achieves optimal efficiency-accuracy trade-off.

n	MATH	BBH	HLE	Relative Cost
1	91.8	88.0	42.8	1.0×
3	96.5	93.5	50.5	3.0×
5	**98.2**	**95.4**	**54.8**	**5.0**×
7	98.4	95.7	55.2	7.0×
9	98.5	95.9	55.4	9.0×

Table 31: **The "Keyword Trap" Analysis.** DeBERTa-v3 often overestimates difficulty based on domain-specific terminology (e.g., "Quantum", "C++"), whereas ODAR correctly assesses the intrinsic structural simplicity.

Query Type	Query Content	Intrinsic Diff.	ODAR DE (Ours)	DeBERTa-v3
Fact Retrieval	"Who is considered the father of **Quantum Mechanics**?"	Simple	**Simple** ()	*Hard* (Overfit)
Simple Code	"Print 'Hello World' in **C++**."	Simple	**Simple** ()	*Medium* (Overfit)
Derivation	"Derive the time-dependent Schrödinger equation."	Hard	**Hard** ()	**Hard** ()

Table 32: Comparison with supervised PRM. ODAR matches PRM on in-distribution tasks while outperforming on out-of-distribution benchmarks, without requiring any training.

Method	Training	GPU-h	MATH	HLE
Self-Consistency	None	0	86.5%	39.5%
Supervised PRM	800k samples	∼500	97.1%	45.8%
ODAR FEP	None	0	96.7%	47.3%

Table 33: Adversarial robustness analysis. ODAR exhibits fail-safe behavior under complexity attacks and resilience under shortcut attacks.

Scenario	Orig.	Adv.	Cost Δ	Behavior
Simple Queries (GSM8K)				
Standard	95.5%	—	1.10×	Correct
Complexity Attack	—	95.3%	+154%	Overthinking
Hard Queries (MATH)				
Standard	52.4%	—	4.50×	Correct
Shortcut Attack	—	46.0%	−38%	Underestimation

Table 34: Rigorous Statistical Analysis: Mean accuracy and between-subsample CI ($K = 5$ disjoint folds, t-based). Δ denotes the paired improvement (ODAR vs. Baseline) with its 95% bootstrap CI. Within MoE uses a binomial SRS approximation ($N = 1,000$) as a reference scale. Global Avg. is the macro-average across 23 benchmarks; uncertainty is estimated via bootstrap over datasets.

Dataset	Mean Acc. (μ)	Subsample CI	Within MoE	Gain Δ	95% CI of Δ
MATH	98.2%	±0.38%	±0.82%	+6.4%	[5.8%, 7.0%]
GSM8K	99.1%	±0.21%	±0.58%	+3.2%	[2.9%, 3.5%]
ARC-Challenge	98.5%	±0.41%	±0.85%	+6.0%	[5.2%, 6.8%]
MMLU-Pro	94.2%	±0.62%	±1.45%	+4.7%	[3.8%, 5.6%]
BBH	95.4%	±0.55%	±1.31%	+4.2%	[3.5%, 4.9%]
HLE	54.8%	±0.95%	–	+12.0%	[10.2%, 13.8%]
Global Avg.	**89.6%**	**±0.45%**	–	**+6.0%**	**[5.2%, 6.8%]**

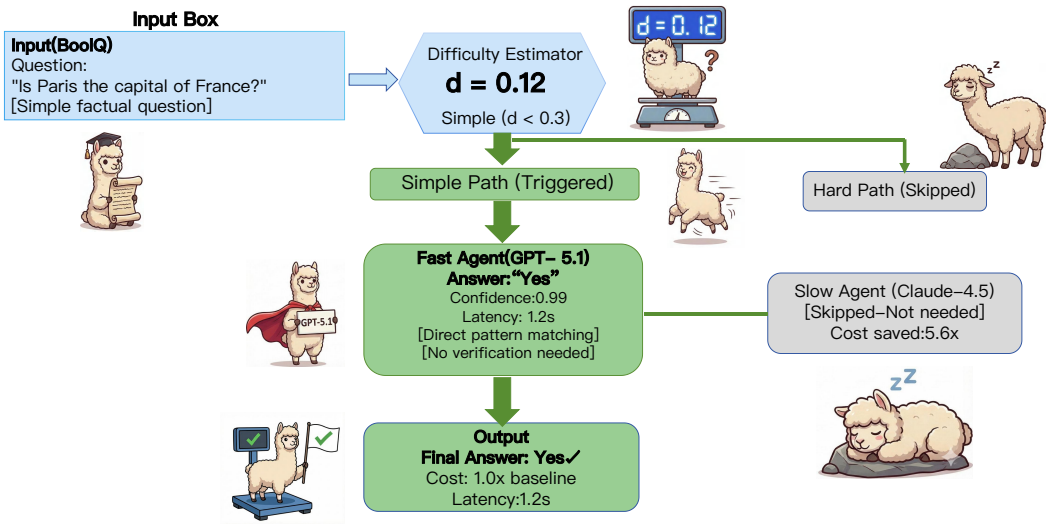

Figure 10: Simple Path execution for a factual query. The Difficulty Estimator ($d = 0.12$) correctly routes to the Fast Agent only.

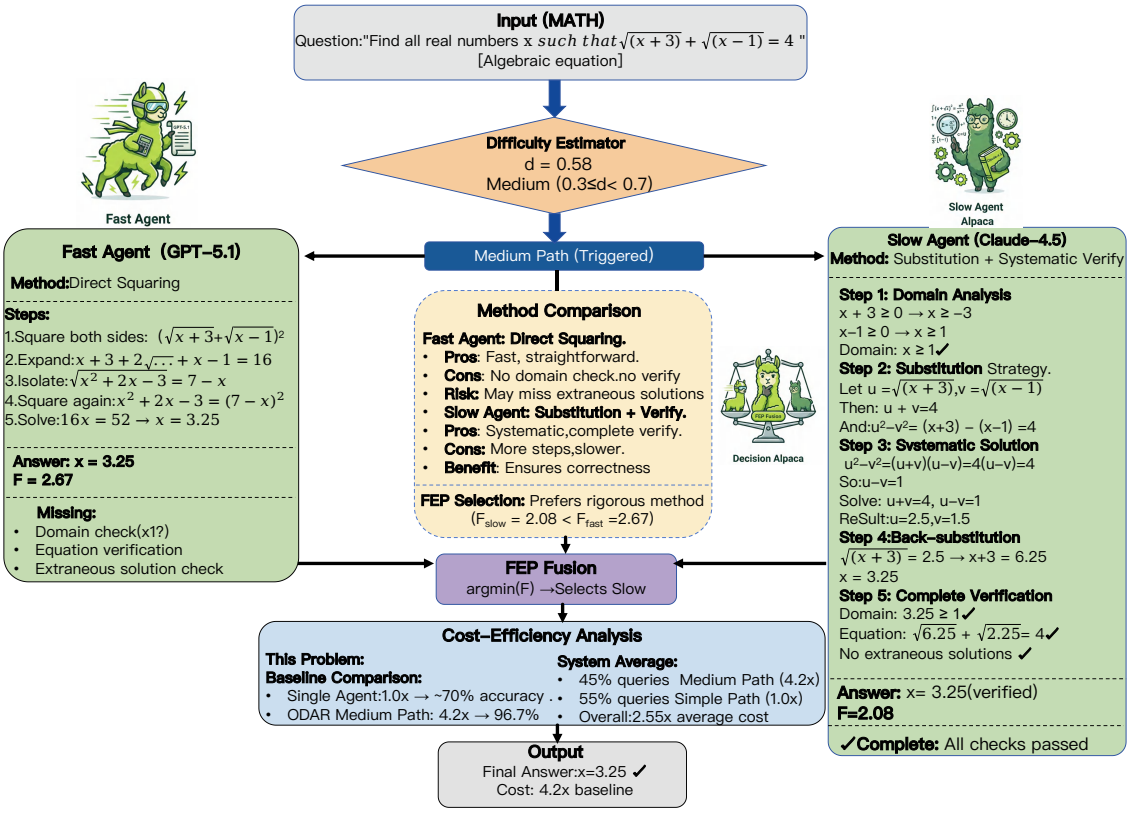

Figure 11: Medium Path workflow for an algebraic equation. FEP Fusion selects the Slow Agent's rigorous solution ($\mathcal{F} = 2.08$) over the Fast Agent's answer ($\mathcal{F} = 2.67$).

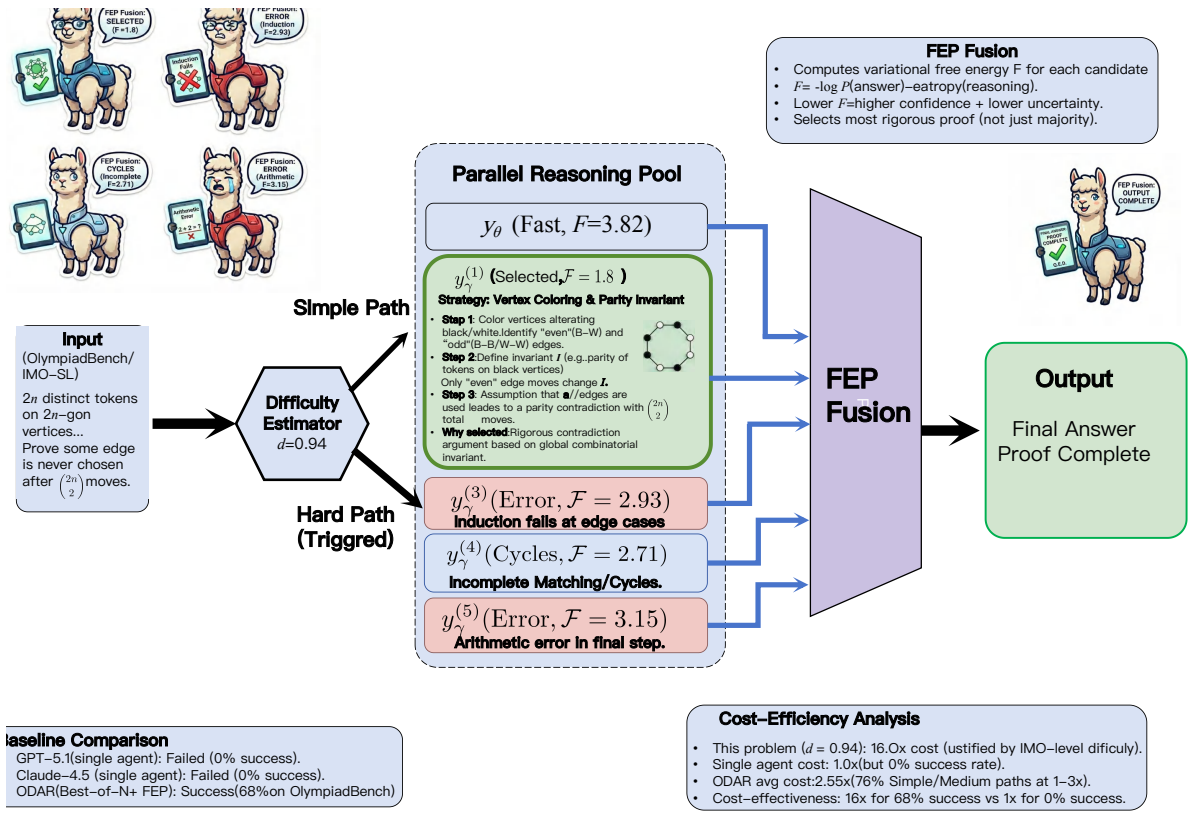

Figure 12: Hard Path execution for IMO-level combinatorics. FEP Fusion identifies the unique complete proof ($y_\gamma^{(1)}$, $\mathcal{F} = 1.85$) among six candidates.