SEMIE: SEMANTIC ENTROPY-INFORMED DECODING

Benjamin Patrick Evans
JP Morgan AI Research
London, UK
benjamin.x.evans@jpmorgan.com

Sumitra Ganesh
JP Morgan AI Research
New York, USA

Leo Ardon
JP Morgan AI Research
London, UK

ABSTRACT

Inference time techniques, such as beam search and best-of-n, typically allocate a fixed amount of computation throughout LLM generations. However, recent entropy-informed decoding methods demonstrate that the shape of the output distribution provides a principled signal for more sample-efficient (adaptive) generations. Even so, the existing entropy-based decoding methods operate at the *token-level*, wasting computation across lexically distinct but semantically equivalent continuations. In this work, we introduce *Semantic Entropy-Informed Decoding (SEMIE)*, a Monte Carlo-based decoding strategy that operates over *semantic continuations* rather than tokens, adaptively branching in regions of high semantic information gain (measured based on semantic entropy) to improve sample efficiency. This semantic entropy provides a measure of uncertainty at higher-level abstractions (e.g., at the level of topics, strategies, or reasoning paths), rather than the underlying tokens. We prove theoretically that adaptive allocation based on semantic entropy yields strictly lower semantic regret than both fixed-width and token-level entropy-informed decoding under equal compute. Empirically, SEMIE attains superior accuracy–compute trade-offs across a range of LLMs and benchmarks, consistently outperforming best-of-n, beam search, and token-level adaptive branching, under generation equated comparisons.

1 INTRODUCTION

Large language models (LLMs) frequently face prompts for which multiple distinct *meanings* constitute valid continuations. In such settings, uncertainty is inherently semantic: the model may be unsure which topic to pursue, which reasoning strategy to apply, or which interpretation of the prompt is intended. Many inference-time techniques, including best-of-n, majority voting, and beam search, do not dynamically allocate computation efficiently under semantic uncertainty, instead using fixed sampling rules.

Entropy-informed decoding Evans et al. (2026) partially addresses this issue by using the entropy of the next-token distribution to adaptively set the branching factor, adjusting the computation. However, token-level entropy conflates two fundamentally different sources of variation: lexical variation (paraphrases) and semantic variation (distinct meanings) Farquhar et al. (2024), which may hamper the branching decisions, leading to inefficient exploration and redundant branching.

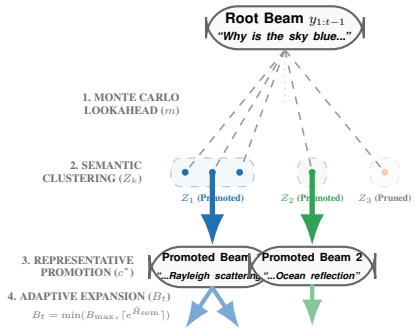


Figure 1: **SEMIE Expansion** The search process (1) draws lookahead samples, (2) groups them into semantic clusters Z_k, (3) promotes max-weight representatives c^* as new beams, and (4) adaptively expands or prunes the tree based on semantic entropy $\hat{H}_{\text{sem}}$.

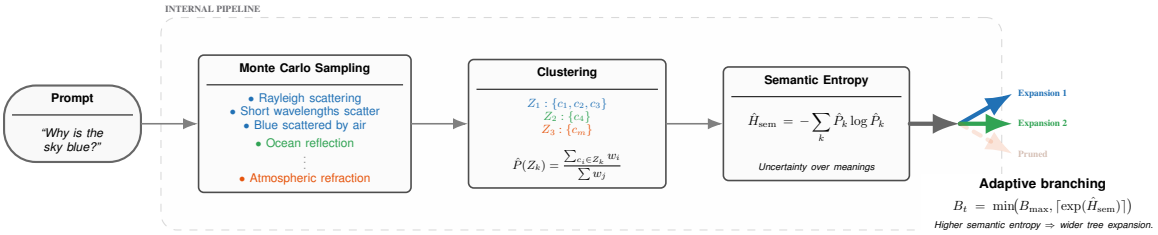


Figure 2: **Overall process** Dynamic branching based on semantic entropy $\hat{H}_{\text{sem}}$ estimated through Monte Carlo sampling and clustering, allowing for more targetted expansions by effectively pruning paths with low semantic likelihood.

These observations motivate a central question:

Can uncertainty over meanings better drive inference time compute allocation?

We address this question by introducing *Semantic Entropy-Informed Decoding (SEMIE)*, adapting compute based on semantic entropy, allowing targetted exploration of topic-level or strategy-level alternatives at inference time, and demonstrate that this outperforms the existing decoding strategies (including beam search, best-of-n) across compute budgets.

2 Background: Entropy-Informed Decoding

Let x denote a prompt, $y_{1:t}$ a partial generation, v a token, $\mathcal{V}$ the vocabulary, and T the maximum generation length. At inference time, we may generate multiple such continuations $\mathbf{y}$. To control the number of continuations to generate, standard token-level entropy-informed decoding computes:

$$H_t^{\text{token}} = -\sum_{v \in \mathcal{V}} P(v \mid x, y_{1:t}) \log P(v \mid x, y_{1:t}) \tag{1}$$

and uses H_t^{token} to determine the amount of exploration at step t, i.e., the number of continuations. This approach implicitly assumes that token-level entropy reflects uncertainty over important outcomes, allocating more compute in these regions. However, many tokens or sequences may encode the same meaning. As a result, token entropy systematically overestimates epistemic uncertainty by treating paraphrases as distinct possibilities. This motivates redefining entropy over *semantic equivalence classes* of continuations, to better drive adaptive generation.

3 Latent Semantic Entropy

While existing entropy-informed decoding operates over the token level, for many tasks (such as reasoning), we are interested in uncertainty over *meanings*. We formalize this distinction using a latent semantic variable framework.

3.1 Semantic State Space

We postulate the existence of a latent semantic variable $z \in \mathcal{Z}$ that represents the underlying intent or meaning of a continuation. We define a surjective mapping $\phi : \mathcal{V}^* \to \mathcal{Z}$ (where $|\mathcal{Z}| \leq |\mathcal{V}|$) that maps a sequence of tokens to its semantic equivalence class.

The *Semantic Entropy* is the entropy of the push-forward measure induced by ϕ on the predictive distribution:

$$H_{\text{sem}}(y_{1:t}) = H(Z) = -\sum_{z \in \mathcal{Z}} P(z \mid x, y_{1:t}) \log P(z \mid x, y_{1:t}) \tag{2}$$

Lemma 3.1 (Denoising via Data Processing)**.** *Let Y be the random variable representing the next L tokens under the model conditional on prefix $(x, y_{1:t})$. Then $H(Z \mid x, y_{1:t}) \leq H(Y \mid x, y_{1:t})$ with equality iff ϕ is bijective on the support of Y.*

Proof. This follows from the standard property of entropy that the entropy of a deterministic function of a random variable cannot exceed the entropy of the original variable Cover (1999) □

This inequality provides the theoretical justification for SEMIE: H_{sem} filters out "lexical noise" (paraphrases) that contributes to H_{token} without adding semantic information.

3.2 Estimating Latent Modes via Clustering

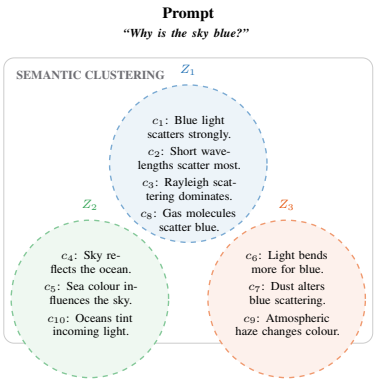


Figure 3: **Semantic clustering.** Monte Carlo continuations $\{c_i\}_{i=1}^m$ are grouped into semantically related clusters Z_k. Each Z_k can contain a variable number of continuations.

Since the latent semantic variable $\mathcal{Z}$ is intractable to marginalize exactly, we approximate it via Monte Carlo (MC) sampling. Let $\mathcal{C} = \{c_1, \ldots, c_m\}$ denote m i.i.d. continuations sampled from $P(\cdot \mid x, y_{1:t})$.

We treat each continuation as a noisy realization of an underlying semantic intent and approximate the latent space $\mathcal{Z}$ by clustering sampled continuations into semantic equivalence classes. Following Farquhar et al. (2024), two continuations are assigned to the same cluster if they express the same meaning (visualised in Figure 3), operationalized via a semantic similarity criterion such as bidirectional entailment.

Let $\{Z_1, \ldots, Z_K\}$ denote the resulting partition of $\mathcal{C}$ into K disjoint semantic clusters.

Cluster probability estimation. We aggregate probability mass within each semantic cluster to estimate its probability mass. For each sampled continuation $c_i = (c_{i,1}, \ldots, c_{i,L})$, we compute its cumulative probability weight: $w_i = \exp(\sum_{j=1}^{L} \log P(c_{i,j} \mid x, y_{1:t}, c_{i,<j}))$.. The probability of a semantic mode Z_k is then estimated as: $\hat{P}(Z_k) = \frac{\sum_{c_i \in Z_k} w_i}{\sum_{\ell=1}^{K} \sum_{c_j \in Z_\ell} w_j}$. approximating the marginal probability mass assigned by the model to each semantic equivalence class.

Semantic entropy. The latent semantic entropy is computed over the induced cluster distribution:

$$\hat{H}_{\text{sem}} = -\sum_{k=1}^{K} \hat{P}(Z_k) \log \hat{P}(Z_k). \tag{3}$$

Overall, clustering compresses the high-variance output space of surface realizations into a lower-dimensional latent space of semantic modes, preserving uncertainty about meaning while discarding syntactic and lexical variability.

4 Semantic Entropy-Informed Decoding (SEMIE)

We propose SEMIE, which utilizes $\hat{H}_{\text{sem}}$ to dynamically modulate the search breadth during the decoding process, adapting based on uncertainty over meanings. Unlike token-based methods that expand based on lexical uncertainty, SEMIE expands based on semantic ambiguity, allowing more targetted expansions.

4.1 Algorithm

The full procedure is detailed in Algorithm 1, and visualised in Figure 2. The key idea behind the proposed approach is to set the branching factor B_t at step t based on the *effective number of semantic modes*, defined by the semantic perplexity: $N_{\text{eff}}(Z) := \exp\left(\hat{H}_{\text{sem}}\right)$, mapping the semantic clusters into an expansion tactic:

$$B_t = \min\left(B_{\max}, \lceil \alpha \cdot N_{\text{eff}}(Z) \rceil\right). \tag{4}$$

This formulation (under $\alpha = 1$) ensures that if the model is distributed across K equally likely semantic paths, the search will branch exactly K times (when $K \leq B_{\max}$), expanding each path. In cases of high lexical variation but low semantic uncertainty ($H_{\text{sem}} \to 0$), the decoding collapses to a greedy path ($B_t = 1$), effectively ignoring the token-level noise arising from multiple semantically equivalent options. In Section 4.2, we explain why this is *principally grounded*.

Algorithm 1 SEMIE

Input: prompt x, language model M, semantic unit length L, lookahead samples m, maximum branching factor $B_{\max}$
Initialize active beams $\mathcal{B}_0 \leftarrow \{y_{1:0}\}$
for $t = 1$ to T **do**
  # Step 1: Monte Carlo Lookahead
  Initialize sample set $\mathcal{C}_t \leftarrow \emptyset$
  for each prefix $y \in \mathcal{B}_{t-1}$ **do**
    Draw m i.i.d. continuations $c_i \sim P_M(\cdot \mid x, y)$ of length L
    // Add (y, c_i) to $\mathcal{C}_t$ for all $i = 1, \ldots, m$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(y, c_1), \ldots, (y, c_m)\}$
  end for
  # Step 2: Semantic Abstraction
  Partition $\mathcal{C}_t$ into semantic clusters $\mathcal{Z}_t = \{Z_1, \ldots, Z_{K_t}\}$ under bidirectional entailment
  Compute sequence weights w_i for all $(y, c_i) \in \mathcal{C}_t$
  Aggregate probability mass within each cluster to obtain $\{\hat{P}(Z_k)\}_{k=1}^{K_t}$
  Estimate semantic entropy: $\hat{H}_{\text{sem}}(t) = -\sum_{k=1}^{K_t} \hat{P}(Z_k) \log \hat{P}(Z_k)$
  # Step 3: Adaptive Semantic Branching
  Compute effective number of semantic modes $N_t = \exp(\hat{H}_{\text{sem}}(t))$
  Set branching factor $B_t = \min(B_{\max}, \lceil \alpha \cdot N_t \rceil)$
  Select the B_t clusters with highest probability mass: $\{Z_{(1)}, \ldots, Z_{(B_t)}\}$
  # Step 4: Sample Reuse and Beam Promotion
  Initialize next beam set $\mathcal{B}_t \leftarrow \emptyset$
  for each selected cluster $Z_{(k)}$ **do**
    Choose representative $(y, c^*) \in Z_{(k)}$ with maximal w_i
    Promote beam: $\mathcal{B}_t \leftarrow \mathcal{B}_t \cup \{y \,\|\, c^*\}$
  end for
end for
Output: final beam set $\mathcal{B}_T$

Table 1: Computational cost comparison. L is the per-step generation length, m is the number of rollouts, B_t is the branching factor at step t, and D is the expected remaining generation depth. For SEMIE, the lookahead cost $m \cdot L$ is incurred once per decision and amortized over future steps.

Method	Branching Width	Lookahead Cost	Amortized Future Cost
Greedy	1	0	$D \cdot L$
Beam Search	B	0	$B \cdot D \cdot L$
EDEN (Token)	$\approx \exp(H_{\text{token}})$	0	$\exp(H_{\text{token}}) \cdot D \cdot L$
SEMIE	$\approx \exp(H_{\text{sem}})$	$m \cdot L$	$\exp(H_{\text{sem}} \cdot D \cdot L$

4.2 Decoding as Semantic Information Gain

Rather than viewing decoding as a simple path-finding exercise, we frame SEMIE as an *Active Information Acquisition* process. We treat the unknown optimal semantic continuation z^* as a latent variable to be uncovered.

Each lookahead sample $c \in \mathcal{C}$ is an observation intended to reduce our uncertainty about the distribution of z^*. The resulting *Semantic Information Gain* (SIG) is the expected reduction in Shannon entropy of the latent semantic state:

$$\text{SIG}(\mathcal{C}) = H(Z \mid x, y_{1:t}) - \mathbb{E}_{c \in \mathcal{C}}[H(Z \mid x, y_{1:t}, c)]. \tag{5}$$

This provides the "why" for branching: by setting $B_t \propto \exp(H_{\text{sem}})$, SEMIE branches exactly when the prefix provides insufficient information to collapse the semantic distribution. To determine *how*

much to branch, we associate each semantic mode z with a value $V(z)$. The expected loss from selecting a suboptimal mode defines the *semantic regret*:

$$\mathcal{R}_{\text{sem}}(t) = \mathbb{E}[V(z^*) - V(z_{\text{selected}})]\,. \tag{6}$$

4.3 Optimality of Semantic Branching

We show that adaptive semantic branching is sample-optimal under the SIG and regret framework, under a monotone entropy–gap assumption (Evans et al., 2026), whereby increasing entropy $H(t)$ implies a smaller gap Δ_t between the top two semantic modes, and thus a greater sample requirement to distinguish them.

Theorem 4.1 (Optimality of Semantic Entropy-Adaptive Branching)**.** *Let $H_{\text{sem}}(t) = H(Z \mid x, y_{1:t})$ denote the semantic entropy at decoding step t, and define the effective number of semantic modes as $N_t := \exp\big(H_{\text{sem}}(t)\big)$.*

At step t, suppose:

1. *Each semantic mode Z_k has a true value $V(Z_k)$, and let*

$$\Delta_t := V(Z^*) - V(Z_{(2)}) > 0$$

   denote the gap between the best and second-best semantic modes.

2. *Each Monte Carlo sample assigned to a mode yields an unbiased estimate of $V(Z_k)$ with independent sub-Gaussian noise of variance proxy σ^2.*

3. *A total sampling budget $\mathcal{B}$ is available and may be allocated adaptively across decoding steps.*

Then allocating a dynamic per step branching factor $B_t \asymp N_t$ and proportionally allocating samples to match B_t minimizes an upper bound on cumulative semantic regret

$$\mathcal{R}_{\text{sem}} = \sum_{t=1}^{T} \mathbb{E}\big[V(Z^*) - V(Z_{\text{selected}})\big] \tag{7}$$

compared to any fixed branching B_{fixed} under the same total computational budget.

Proof sketch. Fix a decoding step t. Estimating the optimal semantic mode is equivalent to a best-arm identification problem whose effective complexity is N_t and gap Δ_t.

By standard sub-Gaussian concentration and a union bound over the effective support of Z, if n_t total samples are allocated at step t, the probability of selecting a suboptimal semantic mode is bounded by

$$\Pr(\text{error}_t) \;\le\; N_t \exp\!\left(-c\,\frac{n_t}{N_t}\,\Delta_t^2\right),$$

for a constant $c > 0$, assuming samples are distributed roughly evenly across modes.

Equivalently, to ensure $\Pr(\text{error}_t) \le \varepsilon_t$, it suffices that $n_t \;\gtrsim\; \frac{N_t \log(N_t/\varepsilon_t)}{\Delta_t^2}$. Thus, the minimal sample complexity required at step t scales with $\log N_t$, and increases when the semantic entropy is high or the value gap Δ_t is small.

A fixed branching policy B_{fixed} allocates the same effective number of candidate modes at every step, and therefore either: (i) over-allocates samples in low-entropy steps (small N_t), or (ii) under-allocates samples in high-entropy steps (large N_t), incurring excess regret.

In contrast, choosing a branching factor $B_t \asymp N_t$ ensures that the sampling effort tracks the intrinsic semantic complexity of the decoding step, avoiding wasting computation where semantic uncertainty is low.

Summing the per-step expected regret bounds over $t = 1, \ldots, T$ yields a cumulative semantic regret bound that is asymptotically minimal under the total budget constraint $\mathcal{B}$, establishing the claimed optimality. □

Corollary 4.2 (Semantic branching factor)**.** *By Lemma 3.1,*

$$H_{\text{sem}}(t) \leq H_{\text{token}}(t) \implies B_{sem} \approx e^{H_{sem}} \leq e^{H_{token}} \approx B_{token}.$$

Hence, SEMIE requires strictly fewer or equal branches than token-level EID while preserving semantic coverage.

Thus, SEMIE achieves the same semantic coverage as token-level EID with fewer samples (where token-level EID already requires fewer than fixed width beam search), or equivalently, achieves lower Semantic Regret for the same expansion cost by avoiding redundant exploration of lexically distinct but semantically equivalent paraphrases.

One might argue that estimating H_{sem} incurs an overhead of m samples due to the Monte Carlo lookahead. We show that under standard generation lengths, SEMIE remains computationally superior due to sample reuse, as we progress the "beam" to the end of the best Monte Carlo rollout.

Corollary 4.3 (Amortized Semantic Efficiency)**.** *Let C_{step} denote the computational cost of advancing a single beam by one semantic unit of length L tokens.*

- ***Token-level EID.*** *The expected per-step cost scales as $B_{\text{token}} \cdot C_{\text{step}}$ since each lexical branch must be independently extended.*
- ***SEMIE.*** *At step t, SEMIE incurs an additional lookahead cost of $m \cdot C_{\text{step}}$ to estimate semantic entropy. However, the continuations used for entropy estimation are reused to initialize the next beam, so the marginal cost of advancing the selected branches is reduced.*

As a result, the amortized *per-step cost of SEMIE over a remaining generation horizon of expected depth D follows*

$$\underbrace{m \cdot C_{\text{step}}}_{\textit{Lookahead cost}} + \underbrace{D \cdot B_{\text{sem}} \cdot C_{\text{step}}}_{\textit{Expansion cost}},$$

where the initial lookahead cost is incurred once but its benefit is realized over all future steps. SEMIE is computationally more efficient than token-level EID whenever

$$\underbrace{(B_{\text{token}} - B_{\text{sem}})\, D}_{\textit{Excess Expansions}} \gtrsim \underbrace{m}_{\textit{Monte Carlo rollouts}}, \tag{8}$$

i.e., when the reduction in redundant lexical branching over the remaining horizon outweighs the one-time cost of semantic lookahead.

Since B_{token} scales with lexical entropy and may remain large even when semantic uncertainty is low, while B_{sem} tracks the effective number of semantic alternatives, the LHS grows linearly with generation length, while m is constant.

Therefore, for long-form generation ($D \gg 1$), SEMIE reduces total compute by pruning semantically redundant branches early while preserving coverage of genuinely distinct reasoning paths.

In Section 5, we additionally show that this is the case empirically, with SEMIE requiring far fewer expansions than the comparison approaches.

4.4 Relation to Existing Decoding Methods

SEMIE provides a paramaterised approach to adaptive decoding. Several widely used decoding methods arise as degenerate cases under specific parameter choices.

Proposition 4.4 (SEMIE Strictly Generalizes Common Decoding Methods)**.** *Let SEMIE be parameterized by a semantic equivalence mapping ϕ, branching parameters $(\alpha, B_{\max})$, and m Monte Carlo rollouts of length L. Then the following decoding methods arise as special cases:*

A ***Beam search.*** *If $L = 1$ and ϕ is bijective on the support of the next-token distribution so that $H_{\text{sem}} = H_{\text{token}}$, and the branching factor is fixed as $B_t = B_{\max}$ (through choosing α sufficiently large that $\lceil \alpha e^{H_{\text{sem}}} \rceil \geq B_{\max}$)) for all t, then SEMIE reduces exactly to classical fixed-width beam search.*

B ***Token-level entropy-informed decoding (EDEN).*** *If $L = 1$ and ϕ is bijective, then the semantic clusters each contain a single token, and SEMIE's branching rule $B_t = \lceil \alpha e^{H_{\text{sem}}} \rceil$ reduces to the token-level entropy-informed rule $B_t = \lceil \alpha e^{H_{\text{token}}} \rceil$ used in EDEN.*

C ***Best-of-n sampling.*** *If $B_{\max} = 1$, the number of Monte Carlo rollouts is set to $m = n$, and the rollout length spans the full generation horizon ($L = T$), then SEMIE generates n independent full-length continuations and selects the highest-probability representative, recovering classical best-of-n decoding.*

Thus, SEMIE strictly generalizes beam search, token-level entropy-informed decoding, and best-of-n, while additionally enabling semantically informed adaptive branching.

Proof sketch. Case (A) follows by Lemma 3.1, which ensures $H_{\text{sem}} = H_{\text{token}}$ when ϕ is bijective; with $B_t = B_{\max}$ recovering standard beam expansion. Case (B) follows since single-token continuations yield singleton clusters (when ϕ is bijective), so SEMIE's entropy estimator matches token entropy exactly. Case (C) follows directly from Algorithm 1: when $B_{\max} = 1$ the search collapses to choosing a single beam, and when $L = T$ and $m = n$ this representative is selected from n independent full-sequence rollouts. □

5 EXPERIMENTS

We evaluate SEMIEs ability to improve generations by utilising (semantic) information-driven expansions, disentangling distinct continuations from surface-level lexical noise.

The experiments test the following hypothesis: Semantic-aware adaptive branching reduces redundant exploration while encouraging *semantic* coverage, and in doing so, achieves strong quality–compute trade-offs.

5.1 SETUP

Methods SEMIE is compared to a range of widely used state-of-the-art inference-time techniques, including: **Best-of-N**: Generating N decoding paths, **Beam Search**: Fixed-width heuristic search, **Token-level EID (EDEN)**: Adaptive branching driven by token-level entropy H_{token}. This provides a range of methods, each with configurable computation amounts for fair evaluations. In each case, the resulting candidate selected is the one with the highest (length-penalised) cumulative log probability.

Underlying models A range of open source models are tested, including LFM (LFM2.5-1.2B-Instruct) Amini et al. (2025), Llama (Llama-3.2-3B-Instruct) Grattafiori et al. (2024), and Granite (granite-4.0-1b) Mishra et al. (2024), covering a multitude of model families and sizes, to ensure that the results are robust and model-agnostic.

Datasets We compare across examples from four commonly used datasets, spanning math (GSM8k Cobbe et al. (2021) and MATH-500 Lightman et al. (2024)), science (SciBench, Wang et al. (2024)), and programming (HumanEval, Chen et al. (2021)), giving a breadth of verifiable domains.

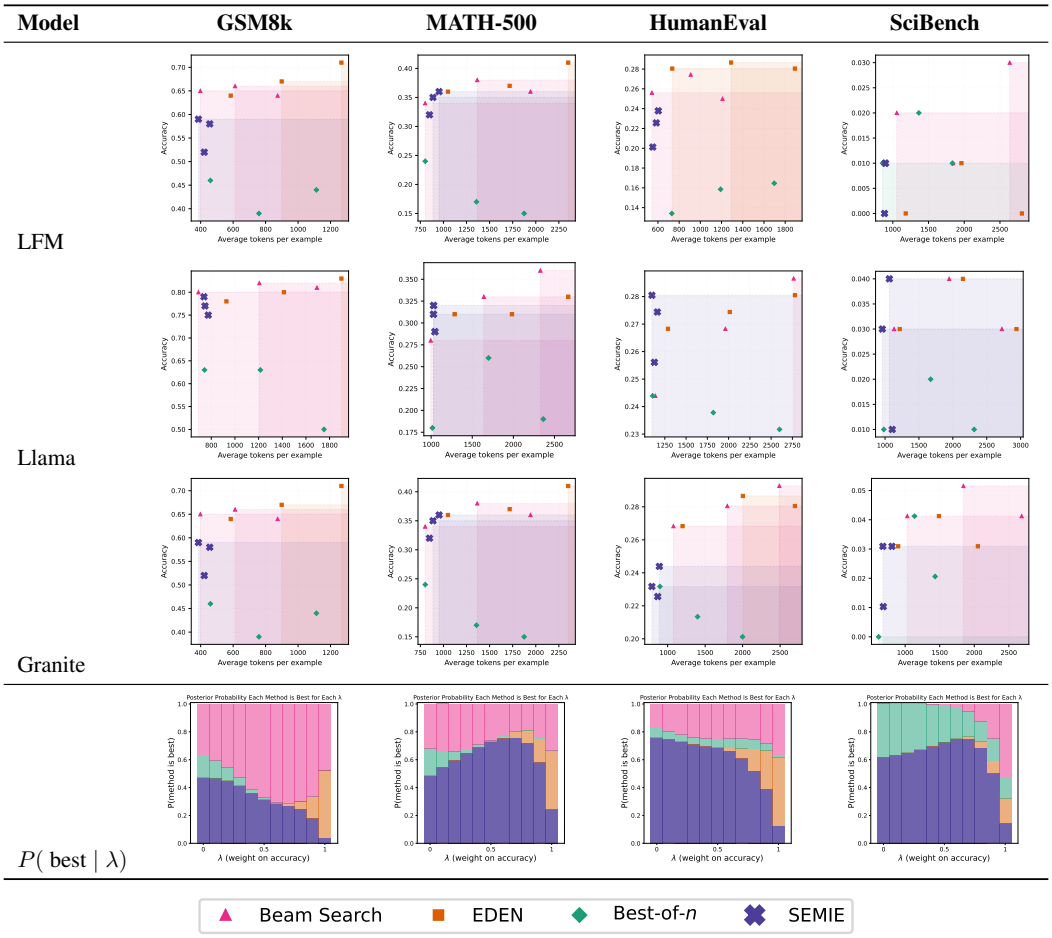
5.1.1 EVALUATION METRICS

Compute budget Computation is controlled using a global budget $G \in \mathbb{N}, G \geq 1$, defined as the maximum allowed factor of overhead in total generated tokens relative to greedy decoding (here, $G \in [3, 5, 7]$). Greedy decoding produces at most T tokens; thus each method is parameterised ($N = G$, $B = G$, $B_{\max} = G$) so that it generates no more than GT tokens per example. This provides an implementation-agnostic notion of compute that depends only on the number of tokens generated. Methods are then compared based on the *actual* number of generated tokens, with G just to control the upper bound for each method.

Accuracy Each tasks has a verifiable accuracy, used to compare the performance of each approach.

The two-objectives are thus: minimise token usage and maximise accuracy.

Table 2: Token–accuracy frontiers across benchmarks and models for $G \in [3, 5, 7]$. For each model (rows) and benchmark (columns), we show the accuracy–token trade-off for all decoding methods. The ideal operating point lies in the top-left corner of each plot (lowest tokens, highest accuracy). Shaded regions indicate the areas in which each method dominates others. Below the frontier plots, we additionally report **p(best)**, the average posterior probability (over $\lambda \in [0, 1]$) that each method is optimal under a scalarized accuracy–compute objective. This provides a preference-agnostic summary of overall performance across methods.



5.2 RESULTS

Table 3: Preference-agnostic robustness scores $\overline{P}(M)$, showing the average posterior probability that each method is optimal under a randomly chosen accuracy–compute preference $\lambda \in [0, 1]$.

	Best-of-n	EDEN	Beam Search	**SEMIE**
$\overline{P}(M)$	10.165%	6.39%	29.67%	**53.78** %

The resulting frontiers are visualised in Table 2. Across the board, the proposed approach consistently achieves strong performance, balancing token generations with resulting accuracy, sitting at or near the Pareto frontier in all cases, maintaining an ideal position relative to the optimal in the top left (lowest tokens, highest accuracies).

To quantify method-level performance, we compute for each method the Bayesian posterior probability of being optimal under accuracy–compute preferences. Following Section B, we evaluate a

scalarized utility function:

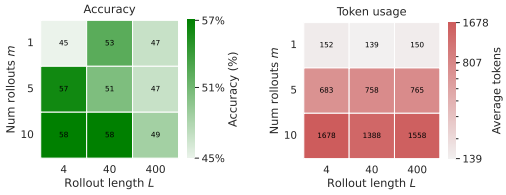
$$U_\lambda = \underbrace{\lambda \,\text{accuracy}}_{\text{Accuracy component}} + \underbrace{(1-\lambda)(1-\text{normalized tokens})}_{\text{Compute component}}$$

over a preference grid $\lambda \in [0, 1]$, and identify the method with the highest utility in each posterior sample. The resulting probabilities $P(M \text{ optimal} \mid \lambda)$ provide a preference-aware view of method performance: for any accuracy–compute trade-off λ, they give the probability that method M is the optimal decoding strategy, reported in the final row of Table 2. Importantly, SEMIE particularly excels in the lower compute regions, as SEMIE essentially approximates the wider searches performed by the comparisons with fewer samples required. As the focus shifts away from token efficiency ($\lambda \to 1$), we expectedly, see beam search and/or EDEN begin to become the best methods, as these perform the most indepth searches.

To quantify across preferences, we additionally report the scalar summary $\overline{P}(M) = \frac{1}{|\Lambda|}\sum_\lambda P_M(\text{ best } \mid \lambda)$, representing the posterior probability that method M is the optimal decoding strategy for a randomly selected accuracy–compute preference in Table 3, providing a single preference-agnostic robustness score for each method. Since four methods are compared, chance performance corresponds to 25%; scores above this level indicate that a method is meaningfully competitive, while scores exceeding 50% imply that the method is more likely to be optimal than all alternatives combined. SEMIE achieves the highest score of all methods *across preferences* of **53.78**%, higher than all the alternatives combined.

Additionally, across the upper bound compute budgets G, the number of tokens with SEMIE does not vary drastically, indicating that the semantic clustering is finding similar paths and keeping expansions targeted, reducing redundant expansions even as the maximum branching factor increases. This shows higher branching factors could be utilised under similar token usage as the comparisons.

5.3 Ablations



(a) Accuracy (b) Token Usage

Figure 5: Monte Carlo Ablations: m and L

ϕ	Accuracy (↑)	Avg. Tokens (↓)	Efficiency (↑)
No Clustering	61%	1618	1.00×
Random Clustering	60%	1311	1.23×
Semantic Clustering	58%	**414**	**3.91**×

Table 4: Accuracy, average token usage, and relative token efficiency across clustering strategies. Semantic clustering yields a $\approx 4\times$ efficiency gain while maintaining competitive accuracy.

To analyse the stability of SEMIE, and the impact of the different components, we run targeted ablations on the key hyperparameters. For this, we use LFM, $G = 5$, and GSM8k.

Monte Carlo Rollouts We ablate two key hyperparameters of the MC rollouts (described in Section A): $m \in \{1, 5, 10\}$ and $L \in \{T/100, T/10, T/1\}$ corresponding to the the number of rollouts and the number of semantic decision / branching points.

Figure 5 (and Table 5) report the resulting accuracy and token usage as a function of (m, L). Increasing m yields modest accuracy improvements, but the token cost increases approximately linearly with m, leading to diminishing Pareto gains at high rollout counts. For example, similar accuracy can be achieved with ≈ 600 tokens with $m = 5$ as ≈ 1600 tokens with $m = 10$. Shorter rollout lengths L generally achieve higher accuracy while having negligible effect on token usage. For very large L, performance degrades as the procedure approaches best-of-N behaviour in the limit $L \to T$ (rightmost column, Figure 4a), as there are fewer adaptive decisions. Overall, these ablations show that SEMIE behaves smoothly and predictably across hyperparameters, demonstrating robustness. Moderate m and relatively low L (in comparison to T) provide the best accuracy–compute trade-off, empirically motivating the fixed choice ($m = 3, L = 5$) in Section 5.2.

Clustering approach ϕ We next examine the role of semantic clustering. Although SEMIE is designed to be agnostic to the particular clustering algorithm used, we adopt the established bidirectional entailment procedure of Farquhar et al. (2024). To assess the contribution of semantic structure itself, we compare this approach against (i) a no-clustering variant, where each input forms a singleton cluster (closer to token-level), and (ii) a randomised baseline, where cluster assignments are uniform at random.

All three approaches yield comparable accuracy (58–61%), but differ markedly in token efficiency (Table 4). Semantic clustering reduces the average tokens to ≈ 400, compared to ≈ 1300 and ≈ 1600 for random and no clustering, a 3–4$\times$ improvement. This demonstrates that semantic structure enables SEMIE to operate more sample efficiently without compromising predictive quality, a feature also seen in Table 2.

5.4 Summary

Overall, through targetted expansions, SEMIE consistently demonstrates good compute–quality trade-offs. By tracking semantic complexity rather than surface-level variation, SEMIE avoids wasting computation in regions of low semantic information gain, ultimately resulting in improved generations, empirically validating the theoretical advantages of semantic entropy for adaptive decoding.

6 Discussion

Semantic entropy separates topic-level variability from surface variability. By allocating computation over meanings rather than tokens, SEMIE reduces redundant exploration while preserving coverage of genuinely distinct continuations, *allowing compute to be targetted at the most impactful regions*. This is particularly important for reasoning, planning, and long-form generation, where semantic diversity is more meaningful than lexical diversity.

6.1 Related work

Motivated by the potential for test time scaling Chen et al.; Wu et al. (2025); Snell et al. (2025), there are many approaches to allocating additional compute at inference time to improve the output quality of LLMs Zhang et al. (2025). Here, we focused on one such strategy, *decoding*.

Best-of-N is the simplest of such approaches, generating multiple full continuations and selecting the best under some measure. Beam search based approaches Sutskever et al. (2014) allow more control at decoding time, branching at each token rather than generation full continuations up front, but is limited by the fixed branching factor. Extensions consider ad-hoc diversity penalties Vijayakumar et al. (2016; 2018), but are ultimately still limited by the fixed branching factor. Evans et al. (2026) introduce EDEN, an entropy-informed decoding framework that adapts the branching factor based on token-level predictive entropy to guide search. While effective for adaptive exploration, EDEN operates purely at the lexical level and treats all tokens as distinct, thereby conflating surface-level variation with genuine uncertainty over meaning. In contrast, Farquhar et al. (2024) propose semantic entropy as a post-hoc measure for hallucination detection and uncertainty estimation, validating the usefulness of semantic uncertainty, but do not consider its use for guiding the decoding process itself. SEMIE builds on these insights by integrating semantic entropy directly into the decoding loop, enabling adaptive exploration driven by uncertainty over latent semantic representations rather than token-level, improving generation quality.

7 Conclusion

We introduced an adaptive inference-time decoding strategy. By adapting expansions based on uncertainty over semantic modes (rather than tokens), our approach allocates computation more efficiently through focusing generation in regions of high semantic information gain, resulting in improved generation quality and simple test-time scaling. Through comparisons across multiple underlying models, methods, and datasets, SEMIE consistently achieves strong performance, outperforming the comparisons, providing strong generations across varying (customizable) token budgets.

DISCLAIMER

This paper was prepared for informational purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co. and its affiliates ("JP Morgan") and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

REFERENCES

Alexander Amini, Anna Banaszak, Harold Benoit, Arthur Böök, Tarek Dakhran, Song Duong, Alfred Eng, Fernando Fernandes, Marc Härkönen, Anne Harrington, et al. Lfm2 technical report. *arXiv preprint arXiv:2511.23404*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL `https://arxiv.org/abs/2107.03374`.

Yanxi Chen, Xuchen Pan, Yaliang Li, Bolin Ding, and Jingren Zhou. Provable scaling laws for the test-time compute of large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Benjamin Patrick Evans, Sumitra Ganesh, and Leo Ardon. Entropy-informed decoding: Adaptive information-driven branching, 2026. URL `https://openreview.net/forum?id=dzmh4xA4Pq`.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=v8L0pN6EOi`.

Mayank Mishra, Matt Stallone, Gaoyuan Zhang, Yikang Shen, Aditya Prasad, Adriana Meza Soria, Michele Merler, Parameswaran Selvam, Saptha Surendran, Shivdeep Singh, et al. Granite code models: A family of open foundation models for code intelligence. *arXiv preprint arXiv:2405.04324*, 2024.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search for improved description of complex scenes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.12340. URL https://ojs.aaai.org/index.php/AAAI/article/view/12340.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R. Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. SciBench: Evaluating College-Level Scientific Problem-Solving Abilities of Large Language Models. In *Proceedings of the Forty-First International Conference on Machine Learning*, 2024.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VNckp7JEHn.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, et al. A survey on test-time scaling in large language models: What, how, where, and how well? *arXiv preprint arXiv:2503.24235*, 2025.

A HYPERPARAMETERS

SEMIE introduces two Monte Carlo hyperparameters that govern semantic lookahead: (i) the number of rollouts m drawn per active beam, and (ii) the rollout length L, which determines the semantic granularity at which entropy is estimated. Both parameters are fixed to constants during decoding.

Number of Monte Carlo samples (m). The parameter m controls the exploratory budget allocated to estimating the distribution over semantic continuation modes. Because SEMIE draws m rollouts *per active beam*, the lookahead cost scales linearly with both m and the current beam width. Larger m provides a more accurate estimate of semantic entropy but increases compute proportionally. Unless otherwise stated, we use $m = 3$.

Lookahead length (L). The rollout length L specifies the number of tokens generated during each semantic lookahead step. A generation of total length T therefore contains approximately T/L semantic decision points. Shorter rollouts (L small) yield finer-grained semantic estimates and behave closer to token-level entropy-guided decoding. Longer rollouts (L large) provide coarser semantic resolution; in the extreme $L \to T$ limit, SEMIE approaches a best-of-n strategy due to having only a single semantic decision point. Throughout the main experiments, we set $L = 5$, which provides a good balance: rollouts are long enough to reveal high-level semantic intent, yet short enough to ensure multiple decision points.

A.1 ABLATIONS

Table 5 reports ablation results on GSM8k (with LFM) over (m, L), covering $m \in \{1, 5, 10\}$ and $L \in \{4, 40, 400\}$. Token usage grows roughly linearly with m, while accuracy varies only moderately across settings, illustrating that SEMIE is robust to a wide range of hyperparameter choices.

Table 5: Ablations on Monte Carlo parameters (m, L).

m	L	Tokens	Accuracy (%)
1	4	152	45
	40	139	53
	400	150	47
5	4	683	57
	40	758	51
	400	765	47
10	4	1,678	58
	40	1,388	58
	400	1,558	49

B BAYESIAN ESTIMATION OF OPTIMALITY UNDER ACCURACY–COMPUTE TRADE-OFFS

In this section we describe our Bayesian procedure for estimating, for each decoding method, the probability of being optimal under a scalarized accuracy–compute trade-off encoded by a weighting parameter $\lambda \in [0, 1]$. This yields a posterior distribution over which method is preferred as the relative importance of accuracy versus compute varies.

Setup. Let $\mathcal{M}$ denote the set of methods under comparison (e.g. SEMIE, EDEN, beam search, best-of-n). As in the main text, each method M may be evaluated at a finite set of compute budgets $G \in \mathcal{G}$, giving a collection of *configurations*

$$\mathcal{C} = \{(M, G) : M \in \mathcal{M},\, G \in \mathcal{G}\}.$$

For each configuration $c \in \mathcal{C}$ and each example $i \in \{1, \ldots, n\}$, we observe a correctness indicator $z_{c,i} \in \{0, 1\}$ and a token count $t_{c,i} \in \mathbb{R}_+$. The empirical accuracy and mean token usage are

$$\hat{p}_c = \frac{1}{n} \sum_{i=1}^{n} z_{c,i}, \qquad \bar{t}_c = \frac{1}{n} \sum_{i=1}^{n} t_{c,i}.$$

B.1 Posterior over accuracy and compute

Accuracy. We model the true accuracy parameter p_c of configuration c via a Bernoulli–Beta model: $p_c \sim \mathrm{Beta}(1, 1)$ and $z_{c,i} \mid p_c \sim \mathrm{Bernoulli}(p_c)$. The posterior distribution is therefore

$$p_c \mid \{z_{c,i}\}_{i=1}^{n} \;\sim\; \mathrm{Beta}\big(k_c + 1,\, n - k_c + 1\big),$$

where $k_c = \sum_i z_{c,i}$.

Compute (token usage). We approximate the posterior distribution of the mean token usage $\bar{t}_c$ using a nonparametric bootstrap. For each Monte Carlo draw we resample indices $i_1, \ldots, i_n$ with replacement from $\{1, \ldots, n\}$ and compute the bootstrap mean $\tilde{t}_c = \frac{1}{n} \sum_{j=1}^{n} t_{c,i_j}$. This yields a sampling-based approximation to the posterior distribution of $\bar{t}_c$.

B.2 Posterior optimality under a scalarized objective

We evaluate each configuration using a scalarized utility function that trades off accuracy against normalized compute cost. For $\lambda \in [0, 1]$, define the utility

$$U_\lambda(c) \;=\; \lambda\, p_c \;+\; (1 - \lambda)\, \big(1 - \tilde{t}_c^{\,\mathrm{norm}}\big),$$

where $\tilde{t}_c^{\,\mathrm{norm}}$ is the posterior-sampled token usage of configuration c normalized to $[0, 1]$ across all configurations. Thus $\lambda = 1$ corresponds to prioritizing accuracy alone, while $\lambda = 0$ favors compute efficiency; intermediate values interpolate smoothly between these extremes.

For each Monte Carlo sample s:

1. For every $c \in \mathcal{C}$, draw $p_c^{(s)} \sim \mathrm{Beta}(k_c + 1, n - k_c + 1)$.
2. For every $c \in \mathcal{C}$, draw a bootstrap mean token count $\tilde{t}_c^{(s)}$.
3. Normalize the vector $\tilde{t}_c^{(s)}$ to $[0, 1]$ across all c.
4. For each λ in a discretization of $[0, 1]$, compute $U_\lambda^{(s)}(c)$ for every configuration.
5. For each method M, compute the best utility achieved by method M at trade-off λ:
$$U_\lambda^{(s)}(M) = \max_{c : M(c) = M} U_\lambda^{(s)}(c)$$
.
6. The method that achieves the largest $U_\lambda^{(s)}(M)$ is considered optimal for that sample and that value of λ.

Let $J_{M,\lambda}^{(s)}$ be the indicator that method M is optimal at weight λ in Monte Carlo sample s.

B.3 Posterior probability of optimality

For any fixed λ, the posterior probability that method M is the best is estimated as:

$$P(M \mid \lambda) \;\approx\; \frac{1}{S} \sum_{s=1}^{S} J_{M,\lambda}^{(s)}.$$

To obtain a preference-agnostic summary, we report in the main text the average probability of optimality across the entire range of trade-off weights:

$$\overline{P}(M) \;=\; \frac{1}{|\Lambda|} \sum_{\lambda \in \Lambda} P(M \mid \lambda),$$

where Λ is a uniform grid of values in $[0, 1]$. This quantity represents the Bayesian posterior probability that method M is optimal for a randomly selected accuracy–compute preference, providing a holistic summary of a method's robustness *across* preferences.