ModeX: Evaluator-Free Best-of-N Selection for Open-Ended Generation

Anonymous ACL submission

Abstract

Selecting a single high-quality output from multiple stochastic generations remains a fundamental challenge for large language models (LLMs), particularly in open-ended tasks where no canonical answer exists. While Best-of-N and self-consistency methods show that aggregating multiple generations can improve performance, existing approaches typically rely on external evaluators, reward models, or exact string-match voting, limiting their applicability and efficiency. We propose **Mode Extraction** (**ModeX**), an evaluator-free Best-of-N selection framework that generalizes majority voting to open-ended text generation by identifying the *modal* output representing the dominant semantic consensus among generated texts. ModeX constructs a similarity graph over candidate generations and recursively applies spectral clustering to select a representative centroid, without requiring additional inference or auxiliary models. We further instantiate this selection principle as **ModeX-Lite**, an improved version of ModeX with early pruning for efficiency. Across open-ended tasks—including text summarization, code generation, and mathematical reasoning—our approaches consistently outperform standard single- and multi-path baselines, providing a computationally efficient solution for robust open-ended text generation.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks, from code generation to creative writing (Achiam et al., 2023; Team et al., 2023; Grattafiori et al., 2024; Yang et al., 2024a; Jiang et al., 2024). Despite this progress, reliably sampling a high-quality output from the model's inherently stochastic generation process remains a fundamental challenge, particularly for open-ended tasks where no canonical answer exists.

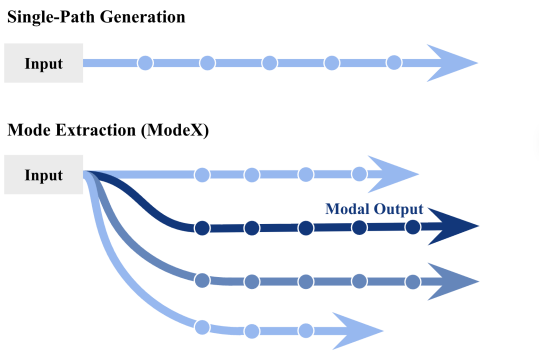

Figure 1: **Single Path Generation vs. Mode Extraction (ModeX).** While single-path text generation commits to a single trajectory, ModeX leverages the structural information across multiple generation paths to select a "modal" output.

Most LLM applications rely on *single-path generation*, in which the model commits to a single output trajectory token by token. This paradigm is inherently brittle: due to stochastic sampling, a single unfavorable token choice can trigger hallucinations or error propagation, even when the model's underlying distribution assigns substantial probability mass to correct or coherent outputs (Wang et al., 2023; Wei et al., 2022). A natural solution is therefore to sample *multiple* generation paths and select the best candidate.

Methods such as self-consistency and Best-of-N sampling demonstrate that aggregating multiple outputs can substantially improve performance, particularly on reasoning tasks (Wang et al., 2023; Hong et al., 2025; Snell et al., 2024). However, existing approaches typically rely on either (i) external evaluators such as reward models (Cobbe et al., 2021; Lightman et al., 2023b) or (ii) exact string-match–based voting schemes. Consequently, these methods are largely confined to closed-ended settings (*e.g.*, multiple-choice or short-answer tasks) and do not generalize naturally to open-ended text generation, where outputs may differ lexically yet remain semantically equivalent. These limitations motivate a central question: ***Can we***

select a single high-quality output from multiple generation paths without external evaluators or significant computational overhead?

To address this question, we propose Mode Extraction (**ModeX**), a Best-of-N selection framework that generalizes the principle of majority voting and self-consistency (Wang et al., 2023) to open-ended text generation. Rather than relying on an external evaluator, ModeX operates directly within the set of generated texts to identify a representative, high-quality solution. Concretely, ModeX builds a graph in which nodes correspond to generated sequences and edges encode pairwise lexical similarity. We then apply spectral clustering—leveraging the Fiedler vector (Fiedler, 1973) of the graph Laplacian—to isolate the dominant semantic cluster, and select its centroid as the final output.

Unlike standard voting schemes based on self-consistency, this procedure does not require exact string matches, predefined answer choices, or auxiliary scoring models. Our key insight is that high-quality generations may vary lexically yet tend to form coherent clusters in the semantic space, whereas hallucinations and erroneous outputs are more likely to manifest as sparse outliers (Lin et al., 2024). Consequently, the most reliable output is often not the most extreme or longest response, but the *modal* one: the generation that best represents the dominant semantic consensus among samples (Figure 1).

Additionally, we show that the efficiency of ModeX can be further improved through early pruning of generation paths. We introduce **ModeX-Lite**, a practical extension that periodically applies modal selection and pruning during generation. By identifying non-representative trajectories at early stages, ModeX-Lite retains the robustness benefits of multi-path aggregation while incurring minimal computational overhead, enabling efficient and reliable generation in practice. Through extensive experiments on text summarization, code generation, and mathematical reasoning, we demonstrate that our methods consistently outperform standard single- and multi-path baselines in both reliability and efficiency. We summarize our contributions as follows:

1. We propose ModeX, an evaluator-free Best-of-N selection framework that generalizes majority voting to open-ended generation, *without requiring external evaluators or expensive computation*.
2. We further introduce ModeX–Lite, an efficiency-improved variant that remains effective across a wide range of any open-ended generation tasks.
3. We conduct extensive experiments on three open-ended generation tasks, showing state-of-the-art performance among evaluator-free approaches. We provide theoretical justifications of our approach, offering a principled Best-of-N selection framework for modern LLMs.

2 Discovering the Mode of Text

Can a single high-quality output be selected from multiple text generation paths without relying on reward models or external verifiers? To address this question, we draw inspiration from the principles of *majority voting* and *self-consistency*, which have been widely adopted in multi-agent LLM frameworks for question answering (Wang et al., 2023; Du et al., 2023; Benedikt Kaesberg et al., 2025). These approaches rest on the premise that, as the number of sampled agents or generation trajectories increases, the aggregated response more faithfully reflects the underlying *modal* belief of the LLM (Choi et al., 2025). In tasks with a finite answer space (*e.g.* multiple-choice question answering), simple voting schemes can therefore reliably recover the modal answer.

Extending this idea to open-ended text generation, however, introduces a fundamental challenge: when the output space becomes infinitely large, the notion of majority or mode is no longer directly countable. In this section, we tackle the problem of identifying the modal generation in such open-ended tasks. We first introduce Mode Extraction (ModeX), a graphical framework that enables principled mode approximation over multiple generated trajectories (Section 2.1), and then qualitatively verify the effectiveness of this approach (Section 2.2).

2.1 Mode Extraction (ModeX)

ModeX's approach to selecting the "mode" of the generated responses proceeds in three steps: (1) adjacency matrix construction, (2) graph spectral clustering, and (3) centroid selection. A visual overview is provided in Figure 2, and the corresponding pseudocode is presented in Algorithm 1 of Appendix F.

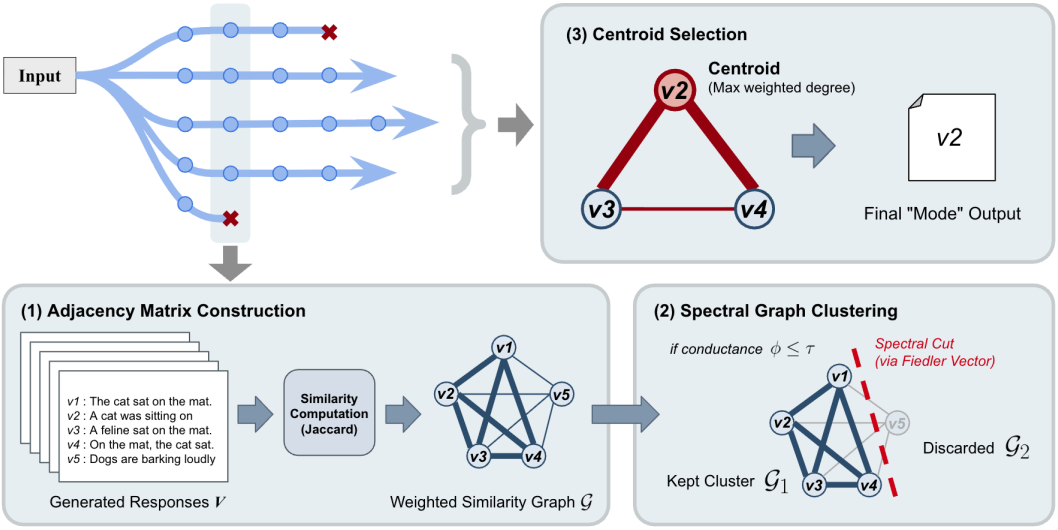

Figure 2: **Overview of the ModeX framework.** In standard ModeX, (1) adjacency matrix construction and (2) spectral graph clustering are iterated recursively as long as $\phi \leq \tau$. Then (3) centroid selection is performed. In the ModeX-Lite variant, (1) $\rightarrow$ (2) is performed only once without recursion for each pruning interval.

(1) Adjacency matrix construction. In closed-ended tasks (*e.g.*, multiple-choice question answering), majority voting can be viewed as the problem of identifying the largest cluster of identical responses. This perspective naturally admits a graph-theoretic formulation. Specifically, consider a graph $\mathcal{G} = (V, E)$, where each node $v \in V$ represents a generated response, and an edge $e \in E$ connects nodes that correspond to the same answer. Under this construction, responses selecting the same choice form a clique, and the answer associated with the largest clique corresponds to the majority. For instance, given five responses in which three select option "A" and two select "B," the three "A" responses form the largest clique, and it is selected as the voted answer.

For open-ended generation, exact equivalence between responses is no longer well-defined, and the notion of a hard clique requires relaxation. Thus, we define edges based on *response similarity*. Concretely, we construct a weighted adjacency matrix $A \in \mathbb{R}^{|V|\times|V|}$, where each entry measures the similarity between a pair of responses:

$$A_{i,j} = s_1(v_i, v_j) + s_2(v_i, v_j) + s_3(v_i, v_j), \quad (1)$$

with $v_i, v_j \in V$ denoting two generated responses. Here, s_1, s_2, and s_3 correspond to Jaccard similarity computed over unigram, bigram, and trigram sets, respectively. This construction yields a weighted graph where stronger edges indicate higher lexical overlap, allowing a soft generalization of voting to open-ended texts. Comparison with an embedding similarity-based adjacency matrix is in Appendix E.

(2) Graph spectral clustering. To identify a dominant group of mutually consistent responses, we next perform clustering over the graph nodes. A key challenge is that the number of coherent groups among generated responses is *a priori* unknown. Rather than fixing the number of clusters, we adopt a hierarchical spectral clustering approach that recursively partitions the graph into two subgraphs.

Specifically, given the weighted adjacency matrix A and the corresponding degree matrix D, we compute the Fiedler vector (Fiedler, 1973), defined as the solution to the following problem:

$$f = \underset{u^\top \mathbf{1}=0, \|u\|_2=1}{\arg\min} \; u^\top (D - A) u, \quad (2)$$

where $L = D - A$ denotes the graph Laplacian. The Fiedler vector provides a continuous relaxation of the minimum cut objective and captures the most salient bipartition of the graph. Further explanation is in Appendix C for completeness.

We obtain a binary partition of the nodes by thresholding the entries of the Fiedler vector:

$$c_i = \begin{cases} 1, & \text{if } f_i \geq 0, \\ 0, & \text{otherwise}, \end{cases} \quad (3)$$

which induces a split of the vertex set $V = V_1 \cup V_2$. To determine whether this partition corresponds to a meaningful separation, we evaluate the quality of the cut using the *conductance ratio* (Sinclair, 1992). The conductance of the resulting cut $(\mathcal{G}_1, \mathcal{G}_2)$ is:

$$\phi(\mathcal{G}_1, \mathcal{G}_2) = \frac{\sum_{i\in V_1} \sum_{j\in V_2} w_{ij}}{\min\left(\sum_{i\in V_1} d_i, \; \sum_{i\in V_2} d_i\right)}, \quad (4)$$

where w_{ij} denotes the edge weight between nodes i and j, and d_i is the weighted degree of node

Chosen vs. Rejected

[Chosen] Scientists propose that life on Earth originated around 3.8 billion years ago, with early DNA-like fragments, capable of self-assembly, potentially guiding the evolution into complex life forms 4 billion years ago. Research from the University of Milan and the University of Colorado Boulder indicates RNA's crucial role, as DNA fragments can naturally bond to form longer chains under certain conditions. This finding supports the idea that RNA could have acted as a template for early life, predating more complex molecules like DNA and proteins.

[Rejected] Scientists theorize that life on Earth began evolving about 4 billion years ago. Recent research by scientists from the University of Milan and University of Colorado Boulder suggests that DNA-like fragments present 4 billion years ago may have come with self-evolving 'instructions', leading to the formation of complex life. These fragments, potentially guided by their innate properties, evolved into longer chemical chains, possibly filling the gap between the simplest life forms and more advanced structures. The study supports the idea that these early DNA-like molecules could assemble and form longer chains spontaneously under appropriate chemical conditions, paving the way for the development of life.

[Target] Scientists say early DNA-like fragments guided their own growth. They claim the process can drive the formation of chemical bonds. These connect short DNA chains to form long ones for life to evolve. This self-assembly capability has been shown to take place in RNA.

Figure 3: **Qualitative Examination.** In the text summarization task, "rejected" samples often miss keywords, include incorrect or less precise information, and contain repetitive and verbose text, whereas samples "chosen" by our method are overall concise.

i. A lower conductance indicates a stronger separation between the two subgraphs. Following the partition, we select the cluster containing the larger number of vertices; in the case of a tie, we select the cluster with the larger total edge weight. We recursively apply this bipartitioning procedure until no further split yields a sufficiently low-conductance cut, *i.e.*, when $\phi(\mathcal{G}_1, \mathcal{G}_2) \geq \tau$, at which point the recursion terminates. In our experiments, we set the conductance threshold to $\tau = 0.8$ and analyze its effect in Section 5.1.

(3) Centroid selection. Once the recursive spectral clustering procedure terminates, we obtain a final cluster of mutually consistent LLM outputs. To extract a single representative response from this cluster, we select its *centroid*, defined as the node that is most strongly connected to all other nodes in the cluster. Formally, let $\tilde{A} \in \mathbb{R}^{n \times n}$ denote the adjacency matrix induced by the final cluster, where n is the number of nodes in it. We define the centroid as the node with the maximum weighted degree:

$$v_c = \arg\max_{i \in \{1,\ldots,n\}} \sum_{j=1}^{n} \tilde{A}_{ij}. \tag{5}$$

Intuitively, this criterion selects the response that exhibits the highest overall similarity to other cluster members, and thus best represents the shared structure of the cluster. The output corresponding to the selected centroid is interpreted as an approximation to the "modal" generation among the original set of $|V|$ sampled outputs.

2.2 Qualitative Examination

To assess whether ModeX indeed selects a representative/modal output, we qualitatively compare the responses that are ultimately "chosen" with those that are not selected (*i.e.*, "rejected"). Figure 3 presents a representative example from the CNN/DailyMail text summarization benchmark (Hermann et al., 2015; See et al., 2017). Across multiple samples, we observe that rejected summaries often omit important keywords, include imprecise or erroneous details, or exhibit repetitive and verbose phrasing. These artifacts reflect idiosyncratic variations specific to individual generation paths and are less characteristic of an average response. In contrast, the selected summaries are consistently concise and focused, capturing the key information of the source document. These observations confirm that our approach is capable of identifying a representative output among multiple candidates, approximating the "modal" generation. Additional qualitative examples are provided in Appendix G, and theoretical discussions are in Section 5.3.

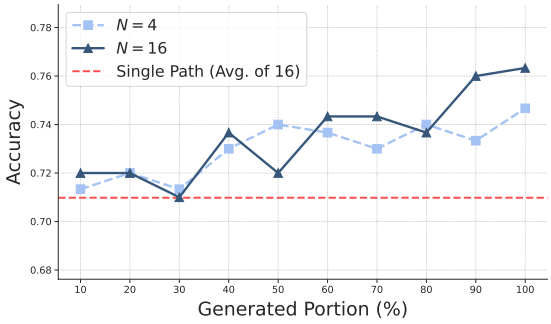

Figure 4: **Math reasoning accuracy at various stages of text generation.** Our mode selection approach consistently identifies high-quality samples early in the trajectory, maintaining high accuracy even with partial outputs.

3 Practical Extension: ModeX-Lite

Building on our principled selection framework that exploits the relational structure among multiple generated outputs, we now present a practical and computationally efficient extension. Transformer-based architectures naturally support parallel sequence generation (Vaswani et al., 2017), enabling multiple generation trajectories concurrently. This parallelism allows us to generate multiple candidate texts and identify a representative output among them without incurring substantial computational overhead.

Observation. Using our textual mode selection approach (Section 2), we observe that high-quality outputs can often be distinguished at early stages of the generation process. As illustrated in Figure 4 for the math reasoning task with Qwen-7B, high-quality candidates are identifiable based on partial generations, *even when less than 50% of the full trajectory has been produced.* This indicates that non-representative paths tend to diverge early, enabling them to be identified and pruned before generation is complete.

Motivated by this observation, we further introduce **ModeX-Lite**, a generation strategy that periodically prunes non-representative text paths at fixed intervals of T steps (T=100 unless stated otherwise). At each pruning interval, we apply graph spectral clustering to the partially generated trajectories, retaining only the most representative subset. To ensure computational efficiency, spectral clustering is performed only once per pruning interval without recursion, and centroid selection is deferred until generation terminates. This design balances the benefits of multi-path aggregation with practical computational efficiency. For clarity, we illustrate the complete procedure in Algorithm 2 and Figure 2.

4 Experiments

4.1 Setup

Tasks and Models. We test on three representative open-ended tasks: text summarization with CNN/DailyMail (Hermann et al., 2015; See et al., 2017), code generation with HumanEval (Chen et al., 2021), and mathematical reasoning with Math-500 (Lightman et al., 2023a). Details on tasks, models, reward models, and metrics are in Appendix A due to limited space.

Baselines. We compare our method against four baselines: (1) *Single Path* reports the performance of standard single-path generation, averaged across 16 independent runs; (2) *Self Refine* (Madaan et al., 2023) iteratively modifies an output four times, as performance is typically known to saturate by this point; (3) *LLM Judge* (Zheng et al., 2023) employs a separate LLM to select the best output out of either 4 or 16 candidates; (4) Perplexity selects the output with the lowest average uncertainty; (5) Self-Certainty (Kang et al., 2025) chooses the output with the lowest negative log likelihood; (6) *Best-of-N* serves as the gold-standard reference, utilizing reward models to choose the best among $N = 16$ samples. Prompt templates are in Appendix B.

4.2 Experimental Results

ModeX consistently outperforms baselines. As shown in Table 1, our method achieves consistently strong performance across all evaluated datasets. In particular, applying ModeX to Qwen with N=16 generation paths improves the mean *Single-Path* baseline from 69.89% to **78.66%** on the code generation task Pass@1 metric. Moreover, ModeX outperforms LLM Judge with 16 candidates by significant margins, and even sometimes surpasses the gold standard Best-of-N that requires external evaluators. This demonstrates that our evaluator-free selection mechanism is more effective than approaches that rely on an LLM to rank or verify multiple outputs. When comparing with the latest approach self-certainty (Kang et al., 2025), ModeX shows generally superior performance across tasks. Overall, these results indicate that ModeX effectively harnesses the benefits of ensemble generation, yielding substantial gains without introducing additional supervision.

More compute does not surpass Single Path performance. Despite consuming roughly 4× the computational resources of standard text

Table 1: **Main results.** Performances of ModeX and ModeX-Lite on three task benchmarks—CNN/DailyMail (text summarization), HumanEval (code generation), and Math-500 (math reasoning)—are presented. Single Path reports the "mean ± std" across 16 independent runs. Note that the code generation of Llama reports the performance of CodeLlama-7b-Instruct.

		Text Summarization				Code Generation		Math Reasoning
Model	**Method**	Rouge-1	Rouge-2	Rouge-L	BLEU	Pass@1	BLEU	Accuracy
Qwen	Single Path	32.95 ± 0.36	10.47 ± 0.22	20.17 ± 0.28	3.37 ± 0.18	69.89 ± 3.59	7.92 ± 0.50	70.98 ± 1.74
	Self-refine	29.76	10.07	18.22	3.04	26.22	1.83	68.67
	LLM Judge (N=4)	32.91	10.54	20.09	3.19	70.12	7.23	71.67
	LLM Judge (N=16)	32.68	10.16	19.72	3.22	65.24	7.52	74.67
	Perplexity BoN (N=16)	34.28	11.24	21.06	3.92	73.17	8.18	**78.00**
	Self-Certainty BoN (N=16)	32.29	10.32	19.32	3.21	55.49	5.43	67.00
	ModeX (N=4)	33.41	10.81	20.40	3.53	67.07	8.02	74.00
	ModeX (N=8)	34.26	11.39	21.08	3.59	71.34	**8.56**	74.67
	ModeX (N=16)	34.28	11.24	21.06	3.92	75.61	8.45	**78.00**
	ModeX-Lite (N=4)	34.15	11.11	21.13	3.47	73.17	8.12	72.67
	ModeX-Lite (N=8)	35.21	12.04	21.83	4.05	76.22	8.42	74.67
	ModeX-Lite (N=16)	**35.78**	**12.35**	**21.89**	**4.36**	**78.66**	8.29	75.33
	Best-of-16 (Gold Standard)	33.46	10.64	20.49	3.26	–	–	82.00
Llama	Single Path	33.97 ± 0.49	12.15 ± 0.22	21.30 ± 0.34	4.41 ± 0.17	18.29 ± 15.22	4.94 ± 1.97	38.75 ± 1.98
	Self-refine	23.97	8.83	15.28	2.75	3.05	1.71	39.00
	LLM Judge (N=4)	34.33	12.55	21.48	4.62	12.80	3.72	37.33
	LLM Judge (N=16)	34.54	12.57	21.60	4.67	7.32	3.14	38.67
	Perplexity BoN (N=16)	34.41	12.45	21.88	4.73	**33.54**	5.81	48.00
	Self-Certainty BoN (N=16)	32.42	11.77	20.06	4.12	4.27	1.37	27.33
	ModeX (N=4)	35.01	12.75	22.04	4.75	23.78	6.20	43.00
	ModeX (N=8)	35.26	12.97	22.13	4.65	27.44	6.39	43.67
	ModeX (N=16)	**35.79**	**13.35**	22.70	5.13	32.32	7.35	**49.33**
	ModeX-Lite (N=4)	35.28	12.87	22.02	4.63	20.12	5.65	39.00
	ModeX-Lite (N=8)	34.46	12.60	21.77	4.40	26.22	6.56	42.33
	ModeX-Lite (N=16)	35.57	13.22	**22.80**	**5.26**	29.88	**7.77**	45.33
	Best-of-16 (Gold Standard)	35.68	13.02	22.25	4.90	–	–	63.00

generation, *Self Refine* fails to surpass our ModeX approaches. In fact, we observe that the refinement process can cause performance to drop significantly below the original Single Path baseline. This suggests that simply scaling inference compute via self-correction is not effective without a selection mechanism to filter out error propagation, compared to parallel text generation.

Increasing the number of generation paths generally improves performance. We further investigate the effect of the number of generation paths, N, on overall performance, as summarized in Table 1. While both the *LLM Judge* baseline and ModeX variants can in principle benefit from the larger search space induced by additional generation paths, ModeX exhibits substantially more consistent and scalable gains as N increases. In the math reasoning task with Llama, increasing the number of paths from N=4 to N=16 yields only a marginal +1.34 percentage-point improvement in accuracy for the *LLM Judge* baseline. In contrast, ModeX-Lite leverages the same increase in paths to achieve a +7.33 percentage-point gain. These results indicate that merely generating more candidates is insufficient; instead, a principled, structure-aware selection strategy is essential to effectively exploit the diversity of the generation space.

5 Discussions

In this section, we provide a deeper analysis of ModeX and ModeX-Lite. We first examine the impact of key design choices and hyperparameter sensitivity (Section 5.1), and provide a complexity analysis demonstrating the computational efficiency of our approach (Section 5.2). We also formalize the theoretical connection between our graph-based selection mechanism and modal approximation (Section 5.3).

5.1 Sensitivity Analysis

We analyze the sensitivity of ModeX-Lite to three key design choices: (a) graph partitioning objective, (b) spectral threshold τ (Eq. (4)), and (c) pruning frequency T. In the top panel of Figure 5, we benchmark our conductance criterion (varying $\tau \in \{0.5, \ldots, 0.8\}$) against an alternative, Normalized Cut (Shi and Malik, 2000):

$$\phi'(\mathcal{G}_1, \mathcal{G}_2) = \sum_{i \in V_1} \sum_{j \in V_2} w_{ij} \left(\frac{1}{\sum_{i \in V_1} d_i} + \frac{1}{\sum_{i \in V_2} d_i} \right),$$

where V_1 and V_2 are the set of nodes in subgraphs $\mathcal{G}_1$ and $\mathcal{G}_2$. In the bottom panel, we examine the impact of the pruning frequency $T \in \{100, \ldots, 500\}$. Overall, we observe that performance is remarkably robust to hyperparameter variations; our method shows relatively stable performance across design choices,

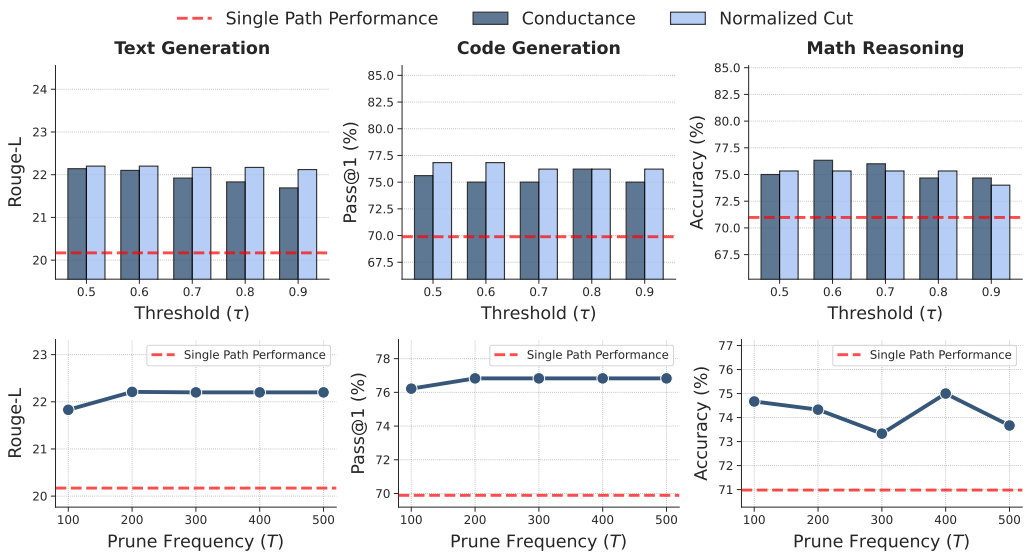

Figure 5: **Sensitivity analysis.** ModeX-Lite shows performance consistently above the single-path baseline in all settings.

Table 2: **Complexity and Latency Analysis.** L: Sequence length (# of tokens), N: Number of paths, k: Refinement iterations, L_{judge}: Length of judge output. We assume parallel generation for $N > 1$. Latency reports the per sample wall time measured on CNN/DailyMail with Qwen-7B.

Method	Complexity	Latency (s)
Single Path	$\mathcal{O}(L)$	5.5
Self-Refine	$\mathcal{O}(kL)$	31.7
LLM Judge	$\mathcal{O}(NL + NL_{judge})$	10.7
Best-of-N	$\mathcal{O}(NL + N \cdot C_{\text{RM}})$	11.1
ModeX-Lite ($N = 4$)	$\mathcal{O}(NL + N^2)$	7.2
ModeX-Lite ($N = 16$)	$\mathcal{O}(NL + N^2)$	9.1

and consistently yields significant improvements over the single-path baseline (red dashed line) across all tested configurations.

5.2 Complexity Analysis

We assess the efficiency of ModeX-Lite by comparing its computational complexity and empirical latency against standard baselines (Table 2). While single-path generation scales linearly with sequence length ($\mathcal{O}(L)$), baseline strategies often introduce significant overhead: *Self-Refine* suffers from sequential dependency ($\mathcal{O}(kL)$), and *LLM Judge* requires a computationally expensive second inference pass, and Best-of-N may require auxiliary reward model passes ($\mathcal{O}(C_{RM})$) which can be expensive in real-world applications without ground-truth labels to evaluate the outputs. In contrast, ModeX's complexity is dominated by the parallel generation of N trajectories ($\mathcal{O}(NL)$). The subsequent selection step—spectral clustering—scales as $\mathcal{O}(N^2)$, which is negligible in practice ($N \ll L$) and requires no neural re-evaluation. Empirically, this architectural difference translates into substantial latency gains. As shown in Table 2, *Self-Refine* incurs a massive latency penalty (31.7s) due to its serial nature. ModeX ($N = 16$) achieves a **3.5**$\times$ speedup (9.1s) over this baseline while maintaining robust performance. Also, compared to the *LLM Judge* (10.7s), our method is faster because its selection mechanism is "evaluator-free", deriving the optimal path solely from the relational structure of texts. With $N = 4$, ModeX-Lite (7.2s) adds only minor overhead to the *Single Path* baseline (5.5s).

5.3 Theoretical Analysis

To formally justify ModeX's graph-based selection mechanism, we model the text generation process as sampling from a high-dimensional probability distribution. We show that under mild assumptions, our two-step process, spectral clustering $\rightarrow$ centroid selection, corresponds to identifying the *modal region* of the distribution and then estimating the *mode* (peak density) within that region.

Setup. Let $\mathcal{X}$ be the space of all possible generated texts. Let $p(x)$ be the probability density function defined over $\mathcal{X}$ by the LLM given a specific prompt. We observe a set of N i.i.d. samples $V = \{v_1, v_2, \ldots, v_N\}$ drawn from $p(x)$. Our goal is to identify the sample $v^* \in V$ that is closest to the true mode of the distribution:

$$v^* \approx \arg\max_{x \in \mathcal{X}} p(x) \tag{6}$$

Our approach rests on the hypothesis that the generation process draws samples from a potentially multi-modal distribution $p(x)$. For instance, in multiple-choice tasks, distinct modes typically emerge around competing options like 'A'

and 'B'. We therefore address mode identification in two steps: first, isolating a coherent, high-density region (via spectral clustering), and second, estimating the point of maximum density within that region (via degree centrality).

Theorem 1. (Spectral Clustering Isolates Modal Components) *Consider a distribution $p(x)$ supported on a disjoint union of manifolds $\mathcal{M}_1 \cup \mathcal{M}_2$ (representing distinct semantic modes) separated by a region of low density. As $N \to \infty$, the spectral bipartition based on the Fiedler vector converges to the geometric cut that separates $\mathcal{M}_1$ and $\mathcal{M}_2$ with minimum probability flow.*
Proof. See Appendix D.

While the Fiedler vector produces a binary partition, our recursive framework naturally generalizes to distributions with $K > 2$ modes. We view the clustering as a hierarchical decomposition of the probability space: each spectral cut splits the current set of samples into two disjoint sets of semantic manifolds. By recursively applying this bipartition until the conductance criterion is met, we effectively isolate a single dominant mode from the original mixture of K modes.

Once the recursive spectral clustering terminates, we obtain a subgraph of $V' \subseteq V$ assumed to be drawn from a locally uni-modal component of the distribution. We now show that the degree centrality within this cluster identifies the mode:

Theorem 2. (Weighted Degree as KDE) *Given a set of samples V' drawn from a distribution, let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a symmetric similarity kernel (e.g., cosine or Jaccard similarity). The weighted degree $d(v_i) = \sum_{v_j \in V'} K(v_i, v_j)$ is proportional to the Kernel Density Estimator (KDE) of the underlying probability density $p(x)$.*
Proof. See Appendix D.

Consequently, our two-step process performs a *conditional mode estimation*: by first partitioning the graph to isolate the dominant cluster C (Theorem 1), the subsequent centroid selection identifies the sample $x = v^*$ that maximizes the conditional likelihood $p(x \mid x \in C)$ (Theorem 2), thereby recovering the specific mode of the dominant interpretation. In effect, this replaces the discrete frequency counting of exact matches in majority voting with continuous density estimation over semantic manifolds. ***This framework therefore constitutes a formal generalization of "majority voting" to open-ended generation tasks.***

6 Related Works

LLM Generation Strategy. A growing line of work has proposed enhanced generation strategies that go beyond standard single-path generation. One approach incorporates *reward models or external verifiers* at inference time to guide generation toward preferred outputs (Huang et al., 2025; Hung et al., 2025; Khanov et al., 2024; Deng and Raffel, 2023; Ouyang et al., 2022). Another line of research exploits *internal model signals* from internal representations and output embeddings (Manvi et al., 2024; Chuang et al., 2023). More recently, *multi-agent generation* frameworks have been introduced, in which multiple agents or experts collaborate during generation by alternately proposing tokens to produce a single output stream (Chakraborty et al., 2025). While effective, these approaches focus on refining a *single* generation path and often require additional models and coordination mechanisms.

Multi-Path Text Generation. A promising avenue for enhancing generation quality involves explicitly leveraging *multiple* generation trajectories. The standard approach, *Best-of-N* (BoN), samples independent candidates and selects the optimal output via an external reward model (Sun et al., 2024; Ouyang et al., 2022). While effective, BoN may incur high computational costs and relies heavily on the quality of the external evaluator. Alternative strategies have attempted to mitigate this via the notion of self-consistency (Hong et al., 2025; Yin et al., 2024; Wang et al., 2023; Chen et al., 2025), internal model signals (Liang et al., 2025; Kang et al., 2025), external reward models (Snell et al., 2024), or multi-agent collaboration (Wang et al., 2025; Chakraborty et al., 2025). Yet, most methods typically target exact-match answer aggregation, restricting their utility to closed-ended reasoning tasks. In this work, we bridge this gap by introducing a framework *applicable to any open-ended tasks* that functions *without external evaluators*.

7 Conclusion

We introduced ModeX and ModeX-Lite, a framework that generalizes majority voting to open-ended generation via graph spectral clustering. By identifying high-quality modal outputs without external evaluators, our method significantly outperforms standard baselines across diverse open-ended tasks. Ultimately, ModeX offers a principled, efficient, and robust generation strategy for LLMs.

Limitations

While ModeX offers a robust, inference-only selection mechanism, it relies on lexical Jaccard similarity to approximate semantic consensus; this metric may fail to recognize valid paraphrases that differ significantly in surface form, potentially causing the rejection of high-quality but lexically distinct outputs. Further investigation with embedding-based similarity measures may be useful. Moreover, the method rests on the assumption that the most frequent output is correct; in cases where the underlying model exhibits systematic bias or “mode collapse" towards a specific hallucination, our spectral clustering approach may inadvertently identify and reinforce this consensus on error. However, we view this as a systematic error of the target LLM itself, rather than a direct limitation of ModeX. Relevant future work to mitigate such corner cases is called for.

Ethical Considerations

This work aims to improve the consistency and reliability of LLMs without relying on costly external verification. We acknowledge that adopting multi-path generation strategies increases the aggregate energy consumption per query, contributing to a larger environmental footprint. We affirm that our experiments utilize public benchmarks and do not involve human subjects, and while improved reasoning capabilities could theoretically be misused, our focus remains on mitigating hallucinations and enhancing general model robustness.

Disclosure of LLM Usage

We used large language model (LLM) tools to polish portions of the writing, to assist in literature searches to check for relevant related work that we might have missed, and to check sanity of our theoretical claims.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Lars Benedikt Kaesberg, Jonas Becker, Jan Philip Wahle, Terry Ruas, and Bela Gipp. 2025. Voting or consensus? decision-making in multi-agent debate. *arXiv e-prints*, pages arXiv–2502.

Souradip Chakraborty, Sujay Bhatt, Udari Madhushani Sehwag, Soumya Suvra Ghosal, Jiahao Qiu, Mengdi Wang, Dinesh Manocha, Furong Huang, Alec Koppel, and Sumitra Ganesh. 2025. Collab: Controlled decoding using mixture of agents for llm alignment. In *The Thirteenth International Conference on Learning Representations*.

Jianhao Chen, Zishuo Xun, Bocheng Zhou, Han Qi, Hangfan Zhang, Qiaosheng Zhang, Yang Chen, Wei Hu, Yuzhong Qu, Wanli Ouyang, and 1 others. 2025. Do we truly need so many samples? multi-llm repeated sampling efficiently scales test-time compute. *arXiv preprint arXiv:2504.00762*.

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, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Hyeong Kyu Choi, Xiaojin Zhu, and Sharon Li. 2025. Debate or vote: Which yields better decisions in multi-agent large language models? In *Advances in Neural Information Processing Systems*.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*.

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*.

Haikang Deng and Colin Raffel. 2023. Reward-augmented decoding: Efficient controlled text generation with a unidirectional reward model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11781–11791.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.

Miroslav Fiedler. 1973. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In (See et al., 2017), pages 1693–1701.

Colin Hong, Xu Guo, Anand Chaanan Singh, Esha Choukse, and Dmitrii Ustiugov. 2025. Slim-sc: Thought pruning for efficient scaling with self-consistency. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 34488–34505.

James Y Huang, Sailik Sengupta, Daniele Bonadiman, Yi-an Lai, Arshit Gupta, Nikolaos Pappas, Saab Mansour, Katrin Kirchhoff, and Dan Roth. 2025. Deal: Decoding-time alignment for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26280–26300.

Chia-Yu Hung, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. 2025. Reward-guided tree search for inference time alignment of large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 12575–12593.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Zhewei Kang, Xuandong Zhao, and Dawn Song. 2025. Scalable best-of-n selection for large language models via self-certainty. *Advances in neural information processing systems*.

Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. 2024. Args: Alignment as reward-guided search. In *The Twelfth International Conference on Learning Representations*.

Zhenwen Liang, Ruosen Li, Yujun Zhou, Linfeng Song, Dian Yu, Xinya Du, Haitao Mi, and Dong Yu. 2025. Clue: Non-parametric verification from experience via hidden-state clustering. *arXiv preprint arXiv:2510.01591*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023a. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

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

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2024. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*, 2024.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, Yang Liu, and Yahui Zhou. 2025. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Rohin Manvi, Anikait Singh, and Stefano Ermon. 2024. Adaptive inference-time compute: Llms can predict if they can do better, even mid-generation. *arXiv preprint arXiv:2410.02725*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, and 1 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

Alistair Sinclair. 1992. Improved bounds for mixing rates of markov chains and multicommodity flow. *Combinatorics, probability and Computing*, 1(4):351–370.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. 2024. Fast best-of-n decoding via speculative rejection. *Advances in Neural Information Processing Systems*, 37:32630–32652.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Junlin Wang, WANG Jue, Ben Athiwaratkun, Ce Zhang, and James Zou. 2025. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

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.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024b. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Zhiyuan Zeng, Xiaonan Li, Tianxiang Sun, Cheng Chang, Qinyuan Cheng, Ding Wang, Xiaofeng Mou, and 1 others. 2024. Aggregation of reasoning: A hierarchical framework for enhancing answer selection in large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 609–625.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Appendix

Table of Contents

A Experimental Details **13**

A.1 Benchmark Details 13

A.2 Model Details 13

A.3 Metric Details 13

B Prompt Templates **13**

B.1 Task Prompts 13

B.2 Baseline Prompts 14

C Why the Second Eigenvector of the Laplacian Acts as a Clusterer? **14**

D Proofs **14**

E Similarity Function Comparison for Adjacency Matrix Construction **15**

F Algorithms **16**

G More Qualitative Examples **17**

A Experimental Details

A.1 Benchmark Details

Text Summarization. For the text summarization task, we evaluate on the CNN/DailyMail benchmark (Hermann et al., 2015; See et al., 2017), which is a dataset for abstractive text summarization. It was constructed from news articles from CNN and Daily Mail. We utilize the first 300 samples from the test split of version 3.0.0.

Code Generation. For the code generation task, we evaluate on the HumanEval benchmark (Chen et al., 2021), which contains 164 Python programming problems with a function signature, docstring, body, and several unit tests. We utilize the full dataset of the test split.

Mathematical Reasoning. For the mathematical reasoning task, we evaluate on the Math-500 benchmark (Lightman et al., 2023a), which contains 500 math questions, ranging six domains, including algebra, geometry, intermediate algebra, number theory, precalculus, and probability. We utilize the first 300 samples from the test split. Also, for precise evaluation, we adopt the evaluation protocol from Sheng et al. (2024)'s codebase.

A.2 Model Details

we evaluate on two model families: `Qwen2.5-7b-instruct` (Yang et al., 2024b) and `Llama3.1-8b-instruct` (Grattafiori et al., 2024). For the code generation task, we adopt `CodeLlama-7b-Instruct` (Roziere et al., 2023), instead of `Llama3.1-8b-instruct`. For the gold-standard Best-of-N, we adopt the `Skywork-Reward-V2-Qwen3-8B` (Liu et al., 2025) reward model for the text summarization task evaluation, and `Qwen2.5-Math-PRM-7B` (Zhang et al., 2025) for the math reasoning task. The code generation task does not currently have a good reward model for Best-of-N selection.

A.3 Metric Details

ROUGE-1 is a recall-oriented metric that measures the overlap of unigrams (individual words) between the generated text and a reference text. It assesses how much of the key content from the reference appears in the output.

ROUGE-2 is similar to ROUGE-1, but measures the overlap of bigrams (pairs of consecutive words). This captures some level of fluency and phrasing, rather than just isolated keywords.

ROUGE-L is based on the Longest Common Subsequence (LCS) between the generated text and the reference. Unlike ROUGE-1 or ROUGE-2, it does not require a fixed n-gram length. Instead, it identifies the longest sequence of words that appear in both texts in the same relative order (though not necessarily consecutively). This allows it to capture sentence-level structure and flow better than simple keyword matching.

BLEU is a precision-oriented metric that counts the overlap of n-grams (usually 1 to 4) between the generation and the reference, penalizing outputs that are too short (brevity penalty). It is widely used to assess how "natural" or close to a human reference the generation is. BLEU is adopted for both text summarization and code generation tasks, but the importance of this metric is lower for the latter task.

Pass@1 is a functional correctness metric often used in code generation or math reasoning. It measures the percentage of problems where the model's first single attempt is correct (*i.e.*, passes all unit tests or yields the correct final answer).

B Prompt Templates

B.1 Task Prompts

Text Summarization

Summarize the following in less than 5 sentences:
`<TEXT TO SUMMARIZE>`
First, briefly state your step-by-step reasoning. Then, make sure to provide your summary after stating "# Answer #"

Code Generation

Complete the following code:
`<BEGINNING OF CODE>`
First, briefly state your step-by-step reasoning. Then, make sure to provide ONLY your COMPLETE code after stating "# Code #"

Mathematical Reasoning

<QUESTION>
First, briefly state your step-by-step reasoning. Then, state your final answer in \\boxed{{}} at the very end of your response, just like: "final answer: \\boxed{{123}}".

B.2 Baseline Prompts

Self Refinement

Previous response:
<OUTPUT FROM LAST ITERATION>
Question:
<TASK>
Instructions: Review your previous response above and provide an improved, refined version. Consider:
1. Accuracy and correctness
2. Clarity and completeness
3. Logical reasoning
4. Better explanations or solutions
Provide your refined response. First, briefly state your step-by-step reasoning. Then,
<TASK-SPECIFIC INSTRUCTIONS>

LLM Judge

Question:
<TASK>
Below are N different responses from different agents:
Response 1 (from Agent 1)
<RESPONSE 1>

Response 2 (from Agent 2)
<RESPONSE 2>

⋮

Response N (from Agent N)
<RESPONSE N>

Instructions: Review all the responses above and select the BEST response based on:
1. Accuracy and correctness
2. Clarity and completeness
3. Quality of reasoning
4. Overall quality
Your response should be ONLY the number (1, 2, 3, etc.) corresponding to the best response. For example, if you think Response 2 is the best, respond with just "2".

C Why the Second Eigenvector of the Laplacian Acts as a Clusterer?

Let $G = (V, E)$ be a graph with adjacency matrix A and degree matrix D, and define the unnormalized Laplacian $L = D - A$. For any real-valued function $x \in \mathbb{R}^{|V|}$ defined over the vertices, the quadratic form of the Laplacian is

$$x^\top L x = \frac{1}{2} \sum_{i,j} a_{ij}(x_i - x_j)^2.$$

This quantity measures the smoothness of x over the graph: it is small when adjacent nodes (i, j) have similar values of x_i and x_j. Thus, minimizing $x^\top L x$ encourages x to vary smoothly along edges.

Since L is positive semidefinite, its eigenvalues satisfy

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n,$$

with the first eigenvector $\mathbf{u}_1 = \mathbf{1}$ (one vector) corresponding to the trivial case of no variation across the graph. The second smallest eigenvector, known as the *Fiedler vector* $\mathbf{u}_2$, solves the constrained optimization problem

$$\min_{x \perp \mathbf{1},\, \|x\|=1} x^\top L x.$$

It represents the smoothest nontrivial variation over the graph—that is, the direction along which the graph can be most naturally divided into two weakly connected components. Nodes with similar $\mathbf{u}_2$ values are strongly connected, whereas nodes with dissimilar $\mathbf{u}_2$ values are weakly connected. Partitioning the graph by thresholding $\mathbf{u}_2$ (e.g., by its median or sign) therefore yields two clusters that approximately minimize the *graph cut* objective, effectively acting as a binary graph clusterer.

D Proofs

Proof of Theorem 1. Let $\mathcal{G}$ be the similarity graph constructed from samples V. The objective of the spectral cut is to find a partition (V_1, V_2) that minimizes the probability flow defined by the conductance ϕ:

$$\phi(V_1, V_2) = \frac{\text{cut}(V_1, V_2)}{\min(\text{vol}(V_1), \text{vol}(V_2))} \tag{7}$$

where $\text{cut}(V_1, V_2) = \sum_{u \in V_1, v \in V_2} A_{uv}$. In the limit of large N, the graph Laplacian converges to the Laplace-Beltrami operator on the underlying data manifold. The Cheeger's Inequality states that the second smallest eigenvalue λ_2 (associated with the Fiedler vector) bounds the conductance:

$$\frac{\lambda_2}{2} \leq \phi^* \leq \sqrt{2\lambda_2} \tag{8}$$

Table 3: **Adjacency matrix similarity function comparison.** We compare our n-gram based design choice with embedding cosine similarity-based computation.

	Text Summarization				Code Generation		Math Reasoning
Method	Rouge-1	Rouge-2	Rouge-L	BLEU	Pass@1	BLEU	Accuracy
Single Path	32.95 ± 0.36	10.47 ± 0.22	20.17 ± 0.28	3.37 ± 0.18	69.89 ± 3.59	7.92 ± 0.50	70.98 ± 1.74
ModeX—cosine (N=16)	33.17	10.25	20.26	2.97	75.00	8.20	71.33
ModeX—n-gram (N=16)	34.28	11.24	21.06	3.92	75.61	8.45	78.00

If the distribution has two distinct modes separated by a "valley" of low probability (low similarity), the edges bridging these regions will have low weights ($A_{uv} \to 0$). This creates a "bottleneck," resulting in a near-zero conductance ϕ^*. Consequently, the Fiedler vector cut will optimally slice through this low-density valley, isolating the high-density clusters $\mathcal{M}_1$ and $\mathcal{M}_2$. This ensures that subsequent mode estimation (Theorem 1) is performed *within* a single coherent semantic cluster, preventing the selection of an incoherent "average" that lies in the low-probability valley between modes.

Proof of Theorem 2. The Kernel Density Estimator $\hat{p}(x)$ for a distribution $p(x)$ given samples $V' = \{v_j\}_{j=1}^N$ is defined as:

$$\hat{p}(x) = \frac{1}{Nh} \sum_{j=1}^{N} K\left(\frac{x - v_j}{h}\right) \tag{9}$$

where h is a bandwidth parameter and K is the kernel. In our graphical formulation, the edge weight A_{ij} is defined by the similarity $S(v_i, v_j)$, which is the Jaccard similarity measure. Assuming S behaves as a kernel function (where $S(v_i, v_j) \approx K(v_i, v_j)$), the weighted degree of a node v_i is:

$$d(v_i) = \sum_{j=1}^{N} A_{ij} = \sum_{j=1}^{N} S(v_i, v_j) \tag{10}$$

Multiplying and dividing by the normalization constants, we observe:

$$d(v_i) \propto \frac{1}{N} \sum_{j=1}^{N} K(v_i, v_j) \approx \hat{p}(v_i) \tag{11}$$

Thus, the weighted degree $d(v_i)$ is a direct proxy for the local probability density around v_i.

$$v_{\text{centroid}} = \arg\max_{v_i \in V'} d(v_i) \equiv \arg\max_{v_i \in V'} \hat{p}(v_i) \tag{12}$$

Since V' represents a coherent (unimodal) cluster, the sample with the maximum empirical density $\hat{p}(v_i)$ is the consistent estimator for the mode of that cluster.

E Similarity Function Comparison for Adjacency Matrix Construction

In Table 3, we compare our n-gram-based similarity matrix construction with the embedding cosine similarity-based approach. Specifically, we retrieve the last token embedding of each output, and compute the cosine similarity between the generated samples:

$$A_{i,j} = \frac{e_i \cdot e_j}{||e_i|| \times ||e_j||}, \tag{13}$$

where e_i, e_j refers to the retrieved embeddings for sample i and j. Overall, the ModeX–cosine outperforms the Single Path baseline, but is consistently worse than ModeX–n-gram.

F Algorithms

Algorithm 1 Mode Extraction (ModeX)

Require: LLM $\mathcal{F}$; Number of text paths N; Similarity function $\mathrm{Sim}(\cdot,\cdot)$; Spectral clustering routine $\mathrm{SpecCluster}(\cdot)$; Cluster evaluator $\mathrm{CutCriterion}(\cdot,\cdot)$; Spectral threshold τ.

Ensure: Selected response r_{i^*}

Initialize active index set $S \leftarrow \{1, 2, \ldots, n\}$.
Initialize response set $R \leftarrow \{r_i\}_{i \in S}$, where $r_i \leftarrow \mathcal{F}(\text{prompt}_i)$
Initialize A, where $A_{ij} \leftarrow \mathrm{Sim}(r_i, r_j) \quad \forall\, r_i, r_j \in R$ ▷ (1) adjacency matrix construction
while $|S| > 1$ **do**
  $(C_1, C_2) \leftarrow \mathrm{SpecCluster}(A)$ ▷ (2) graph spectral clustering
  if $\mathrm{CutCriterion}(C_1, C_2) < \tau$ **then**
    if $|C_1| \neq |C_2|$ **then**
      $S \leftarrow \arg\max_{C \in \{C_1, C_2\}} |C|$
    else
      $S \leftarrow \arg\max_{C \in \{C_1, C_2\}} \sum_{i \in C, j \in [N]} A_{ij}$
    end if
    Update A, R to include only $i, j \in S$
  else
    Terminate clustering
  end if
end while
$i^* \leftarrow$ maximum degree node index of A ▷ (3) centroid selection
return r_{i^*}

Algorithm 2 ModeX-Lite

Require: LLM $\mathcal{F}$; Input prompt x; Initial number of paths N; Pruning interval T

Ensure: Final response y

Initialize N generation paths: $X^{(0)} \leftarrow \{x^{(i)} = x\}_{i=1}^{N}$
$t \leftarrow 0$
while not all remaining paths have generated an `EOS` token **do**
  $X^{(t+1)} \leftarrow \mathcal{F}\big(X^{(t)}\big)$ ▷ one-step parallel generation
  if $(t+1) \bmod T = 0$ **then**
    $X^{(t+1)} \leftarrow \textsc{ClusterAndPrune}\big(X^{(t+1)}\big)$ ▷ adjacency construction & spectral clustering
  end if
  $t \leftarrow t + 1$
end while
$y \leftarrow \textsc{SelectCentroid}\big(X^{(t)}\big)$ ▷ centroid selection
return y

G More Qualitative Examples

Chosen vs. Rejected (CNN/DailyMail)

[Chosen] The article discusses Tiger Woods' performance at the Masters, focusing on his struggle with the short game while showing improvement in driving. Despite his challenges, particularly with his chipping, the galleries remained supportive. Woods showcased moments of his former greatness but was far from his peak performance, dealing with an ongoing crisis of confidence and physical ailments. His game demonstrated both his cussedness and vulnerability, indicating mixed progress towards regaining his competitive form.

[Rejected] Tiger Woods experienced a challenging start to his first round at the Masters, showcasing both flashes of brilliance and persistent flaws. Despite working on his short game, Woods exhibited inconsistent driving, which overshadowed his typically strong chipping skills. His performance highlighted a mix of improvement and decline, with Woods making heroic saves and committing occasional errors. A notable moment included his stunning drive on the seventh hole, reminiscent of his past greatness, juxtaposed with misses like a mis-hit approach to the tenth hole. The crowd, though supportive, recognized the signs of an older Tiger Woods. While showing promise and determination, Woods' performance suggested a player still adjusting to physical limitations and potentially dealing with psychological challenges.

Chosen vs. Rejected (CNN/DailyMail) (CNN/DailyMail)

[Chosen] George Boyd is the Barclays Premier League's most tireless player, having covered 210.5 miles this season, a feat equivalent to running between Burnley's Turf Moor and Crystal Palace's Selhurst Park. Boyd beats fellow player Christian Eriksen in the "running marathon." Burnley, sitting near the relegation zone, boasts three players in the top 10 hardest-working runners, collectively covering 2,172.3 miles.

[Rejected] George Boyd of Burnley has set a new milestone as the hardest working player in the Barclays Premier League this season, having run 215 miles. This distance equals the journey from Turf Moor to Selhurst Park, with Boyd leading the list just ahead of Tottenham's Christian Eriksen. Burnley, currently 18th and fighting for survival, has three players in the top 10 running enthusiasts. Key other featured players include Stewart NZonzi, Jordan Henderson, and Daryl Janmaat. Boyd, Arfield, and Jones lead a total running distance of 2,172.3 miles for Burnley. The ranking highlights efforts from lower-ranked teams, with only a few from top-half clubs making the list. Boyd aims to use his running for Burnley's benefit in upcoming matches against Tottenham.

[Target] Burnley winger George Boyd has run the furthest in the Premier League. His 210.5 miles this season beats Christian Eriksen from Tottenham. Burnley have run an incredible 2,172.3 miles between them this season. Players from Stoke, Hull, Liverpool and West Ham feature in the top 10.

Chosen vs. Rejected (CNN/DailyMail)

[Chosen] The article discusses the rapid growth in the number of millionaires in various African cities, particularly in Nigeria's Lagos and others like Luanda, Dar es Salaam, and Accra. This new affluent class is driving the demand for luxury goods, with sales of high-end products growing significantly in these regions. Cities like Nigeria are now rivaling traditional wealth centers for luxury consumption. Market experts attribute this growth to rising incomes and population. These wealthy consumers, often younger than their Western counterparts, prefer well-known global brands that symbolize status and success. Meanwhile, some homegrown African businesses see this trend as an opportunity to establish themselves in the luxury market, leveraging the continent's rich cultural heritage and craftsmanship.

[Rejected] The number of millionaires in African cities like Lagos, Luanda, Dar es Salaam, and Accra is rapidly increasing, with significant growth particularly noted in Lagos. This shift has transformed the consumption patterns of luxury goods, with wealthy Nigerians now able to enjoy prestigious brands locally. Factors such as rising incomes and slowing growth in other emerging markets are driving this trend. While international luxury brands remain dominant, there is a growing interest in local, artisanal products as Africa's affluent consumers become younger and look to luxury for social status. Despite the promising outlook, the future of luxury markets in Africa depends on economic reforms improving infrastructure and trading conditions. Additionally, there is a recognition of the potential for developing a strong domestic luxury industry by leveraging traditional craftsmanship and cultural heritage.

[Target] The city with most multimillionaires in Africa is Johannesburg. However a crop of new pretenders have been expanding their millionaire count.

Chosen vs. Rejected (CNN/DailyMail)

[Chosen] A 16-year-old student, Rebecca Rice, was allowed to bring her pet Labrador, Holly, into a GCSE French oral exam to help calm her nerves. This arrangement was made after Rebecca's teacher, Lorette Esteve, learned that no other methods were effective. Bodmin College in Cornwall, where the student attends, has implemented various strategies to reduce exam stress, such as bringing drinks, lucky items, or friends into the exam room.

[Rejected] A 16-year-old student, Rebecca Rice, was allowed to bring her 11-year-old Labrador, Holly, into a French GCSE oral exam to help manage her severe nerves. Following a discussion with her teacher, Lorette Esteve, who leads foreign languages at Bodmin College, it was decided that Holly's presence would assist Rebecca, as nothing else did. Rebecca felt her nerves settled significantly with Holly by her side during the five-minute exam. Esteve mentioned that the school employs various methods, such as offering drinks, lucky key rings, or friends in the room, to help students cope during exams.

[Target] Rebecca Rice, 16, was allowed to take her pet dog into five-minute exam. She told her teacher she struggles to control her nerves ahead of the GCSE. Teacher Lorette Esteve advised she bring along 11-year-old dog Holly. School has various tactics help reduce stress including lucky key rings.

Chosen vs. Rejected (CNN/DailyMail)

[Chosen] Dr. Xiao-Ping Zhai, based in Harley Street, London, uses traditional Chinese medicine to help women conceive. She offers methods like acupuncture and customized Chinese herb prescriptions, claiming to have helped over a thousand women get pregnant. Patients like Jane Parker successfully conceived using Dr. Zhai's techniques. Her methods are detailed in a BBC documentary and include daily herbal tea or vitamin tablets, costing up to £150 weekly. The program also explores other alternative treatments, including leech therapy, reflecting a trend towards traditional remedies among infertile couples unsatisfied with Western medicine.

[Rejected] Dr. Xiao-Ping Zhai, a prominent figure at The Zhai Clinic in London, uses traditional Chinese medicine to help infertile couples conceive. Opened in 1996, her clinic offers services such as acupuncture and customized Chinese herbal remedies, which she claims have helped over a thousand women become pregnant. The BBC documentary "Inside Harley Street" provides insight into Zhai's approach and features the journey of patient Jane Parker, who conceived her son at age 41 through the clinic's methods. While many seek these alternative treatments due to dissatisfaction with conventional approaches like IVF, other practices, such as leech therapy, are also explored for their purported benefits in various health issues. The documentary examines the growing trend of relying on non-traditional methods for overcoming infertility, reflecting the desire among patients for personalized and holistic care.

[Target] Around one in seven couples suffer with infertility. Dr Xiao-Ping Zhai offers help via with traditional Chinese medicine. Uses acupuncture and prescribes course of Chinese herbs.

Chosen vs. Rejected (CNN/DailyMail)

[Chosen] Australian cook Lyndey Milan collaborated with Aldi to create affordable Easter recipes that cost less than $6 per person. She developed meals like Greek lamb with salad and zucchini pilaf, herb-crusted salmon, and desserts such as hot cross bun and Easter egg bread & butter pudding. Milan stressed the cost-effectiveness, health benefits, and fun aspects of homemade cooking. Practical tips include using up all bought ingredients, purchasing seasonal produce, and freezing surplus foods to reduce waste and grocery expenses.

[Rejected] Lyndey Milan, an Australian chef known for her home cooking expertise, has collaborated with Aldi to create affordable Easter lunch recipes that cater to six people for under $6 per person. Utilizing fresh, seasonal produce from Aldi, Milan offers recipes like Greek lamb with salad and zucchini pilaf, herb-crusted salmon with pea puree, smashed potatoes, and carrots, along with a hot cross bun and Easter egg bread and butter pudding. She emphasizes tips for minimizing waste, reducing grocery expenses, and maximizing the use of fresh ingredients, suggesting that her dishes are not only budget-friendly but also tasty and easy to prepare.

[Target] Lyndey Milan created Easter lunch recipes for $6 per person from Aldi. Home cook icon trawled supermarket aisles to find inspiration for recipes. Result of experiment include Greek lamb and herb crusted salmon dishes . Hot cross bun and Easter egg bread & butter pudding came in at just $2 a person and Lyndey says it is sure to be a crowd pleaser.

Chosen vs. Rejected (HumanEval)

[Task]
Complete the following code:

```
def specialFilter(nums):
    """Write a function that takes an array of numbers as input and returns
    the number of elements in the array that are greater than 10 and both
    first and last digits of a number are odd (1, 3, 5, 7, 9).
    For example:
    specialFilter([15, -73, 14, -15]) => 1
    specialFilter([33, -2, -3, 45, 21, 109]) => 2
    """
```

Make sure to provide ONLY your COMPLETE code after stating "# Code # ".

[Chosen]

```
def specialFilter(nums):
    def is_odd_digit(n):
        return n % 2 != 0

    def has_odd_first_last_digits(num):
        str_num = str(abs(num))
        first_digit = int(str_num[0])
        last_digit = int(str_num[-1])
        return is_odd_digit(first_digit) and is_odd_digit(last_digit)

    count = 0
    for num in nums:
        if num > 10 and has_odd_first_last_digits(num):
            count += 1
    return count
```

[Rejected]

```
def specialFilter(nums):
    def is_odd_digit(n):
        while n > 0:
            digit = n % 10
            if digit not in [1, 3, 5, 7, 9]:
                return False
            n //= 10
        return True

    count = 0
    for num in nums:
        if num > 10 and is_odd_digit(abs(num)):
            count += 1

    return count
```

Chosen vs. Rejected (HumanEval)

[Task]
Complete the following code:

```
def f(n):
    """ Implement the function f that takes n as a parameter,
    and returns a list of size n, such that the value of the element at index
    i is the factorial of i if i is even
    or the sum of numbers from 1 to i otherwise.
    i starts from 1.
    the factorial of i is the multiplication of the numbers from 1 to i (1 * 2
     * ... * i).
    Example:
    f(5) == [1, 2, 6, 24, 15]
    """

 Make sure to provide ONLY your COMPLETE code after stating "# Code # ".
```

[Chosen]

```
def f(n):
    def factorial(x):
        if x == 0 or x == 1:
            return 1
        else:
            return x * factorial(x - 1)

    def sum_to_i(x):
        return x * (x + 1) // 2

    result = []
    for i in range(1, n + 1):
        if i % 2 == 0:
            result.append(factorial(i))
        else:
            result.append(sum_to_i(i))
    return result
```

[Rejected]

```
import math
def f(n):
    def factorial(i):
        return math.factorial(i)

    def sum_to_i(i):
        return i * (i + 1) // 2

    result = []
    for i in range(1, n + 1):
        if i % 2 == 0:
            result.append(factorial(i))
        else:
            result.append(sum_to_i(i))
    return result
```