Complexity-Guided Math Reasoning with ReAct Agents Using Normalized Compression Distance

Oleg Roshka
Project Contributor
oleg.roshka@proton.me

June 1, 2025

Abstract

Large Language Models (LLMs) have shown promise in mathematical reasoning, particularly when augmented with tools within frameworks like ReAct (Reason and Act). However, their multi-step solutions can sometimes become inefficient or deviate due to unconstrained generation. This paper introduces **NCD-Guided Reasoning**, a novel approach that leverages Normalized Compression Distance (NCD) to provide a real-time, domain-agnostic signal of cognitive complexity for an LLM agent's step-by-step reasoning process. We integrate these NCD signals into a ReAct agent, guided by a complexity model (a VAE-Diffusion model in our final experiments), to steer its problem-solving trajectory. We present experimental results on the GSM8K dataset using Gemma2-9B, comparing a baseline ReAct agent with our NCD-guided agent. The NCD-guided agent demonstrates a modest, though not statistically significant, improvement in accuracy, a reduction in solution steps for successful problems, and a notable decrease in the variance of solution complexity and step count, albeit at the cost of increased latency. This work demonstrates that NCD-guidance can act as a risk-control mechanism, leading to more predictable and robust, if not immediately faster, LLM-based mathematical reasoners.

1 Introduction

Recent advancements in Large Language Models (LLMs) have enabled significant progress in complex reasoning tasks, including mathematics [1, 2]. Frameworks like ReAct [8], which combine chain-of-thought reasoning with tool usage, have further improved performance by allowing models to interact with external environments, such as calculators or code interpreters. Despite these successes, LLM-generated solutions often suffer from inefficiencies: they might take unnecessarily convoluted paths, get stuck in repetitive loops, or produce overly verbose reasoning that doesn't contribute to the final answer. This can lead to increased latency, higher computational costs, and potentially more errors.

A key challenge is the lack of an intrinsic mechanism for LLMs to gauge the ”reasonableness” or ”complexity” of their own generated steps in real-time. While humans often have an intuitive sense of when a solution path is becoming too complex or too simplistic for a given problem, LLMs typically lack this self-awareness.

This project introduces **NCD-Guided Reasoning**, a method to imbue ReAct-style agents with a sense of their step-by-step cognitive complexity using Normalized Compression Distance (NCD) [4]. NCD provides a universal, parameter-free measure of similarity (and thus, complexity change) between sequences of text. We hypothesize that by monitoring local (step-to-step) and global (entire history) NCD-derived complexity signals, an agent can learn to navigate the problem space more effectively.

We aim to:

- Collect and analyze complexity profiles (NCD local, NCD global) from LLM agent trajectories on mathematical reasoning tasks (initially GSM8K [1]).

- Train models (e.g., a VAE-Diffusion model) to learn the dynamics of "successful" complexity trajectories.
- Integrate these learned complexity models into a ReAct agent to guide its step generation, through a step-gating mechanism (discard/retry based on predicted step appropriateness).
- Evaluate whether this complexity guidance improves solution accuracy, efficiency, and overall reasoning quality.

This report details our approach, including data analysis from Gemma2-9B on GSM8K, the setup for training complexity models, and the experimental evaluation of NCD-guided agents.

2 Problem Statement

The central problem is the unconstrained nature of multi-step reasoning in LLM agents, which can lead to suboptimal solution paths. Current ReAct agents typically generate steps based solely on the LLM's internal knowledge and the immediate context, without an explicit mechanism to evaluate if a generated step represents productive progress or a detour. This can manifest as:

- **Over-complexity:** Taking too many small, redundant steps, or introducing unnecessarily complex calculations.
- **Under-complexity (Oversimplification):** Making large leaps in reasoning that are hard to follow or error-prone, skipping crucial intermediate steps.
- **Getting Stuck:** Repeating similar reasoning patterns without making progress.

Our project aims to address this by enabling an agent to monitor its own "cognitive effort" via NCD. The specific goals are to:

1. Develop methods to compute and represent step-wise complexity (local NCD) and cumulative trajectory complexity (global NCD) in real-time during an agent's operation.
2. Train a complexity model (e.g., VAE-Diffusion model) that, given a partial trajectory and its complexity features, can predict:
   a. The likely NCD values of the subsequent step.
   b. The overall probability of the current reasoning path leading to a successful solution.
3. Integrate this complexity model into a ReAct agent to guide step generation, for instance, by gating (accepting/discarding/retrying) candidate steps based on their predicted impact on success probability or deviation from typical complexity patterns.
4. Quantitatively evaluate the impact of this NCD-guidance on problem-solving accuracy, solution length (#steps), and computational latency against a baseline unguided agent.

The input to our system is a math problem and access to an LLM and tools. The output is a solved problem with a reasoning trace, hopefully achieved with improved accuracy and/or efficiency.

3 Literature Review and Background

Our work builds upon several key areas:

1. **ReAct Framework:** The "Reason and Act" paradigm proposed by Yao *et al.*[8] demonstrated the power of interleaving chain-of-thought reasoning with tool use for LLMs. Our work directly extends this framework by adding a layer of complexity-awareness.
2. **Chain-of-Thought (CoT) Reasoning:** Wei *et al.*[7] showed that prompting LLMs to produce step-by-step reasoning improves performance on complex tasks. Our complexity signals aim to make these chains more robust and efficient.
3. **Normalized Compression Distance (NCD):** Introduced by Li *et al.*[4], NCD is a parameter-free, feature-free measure of similarity between two data objects based on how well they compress together versus separately. It approximates Kolmogorov complexity [3] and has been used in diverse fields. Its application to real-time monitoring of LLM reasoning steps is novel. The formula is $NCD(x, y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))}$, where $C(\cdot)$ is the compressed size.
4. **Self-Correction and Reflection in LLMs:** Researchers have explored enabling LLMs to critique and refine their own outputs [6, 5]. Our proposed reflection phase, informed by complexity analysis,

aligns with this direction. The step-gating mechanism is a more immediate form of self-correction.

5. **Sequential Decision Making & Planning:** LLM agents operate as sequential decision-makers. Our complexity model aims to provide a heuristic or value function to guide this process, akin to state evaluation in reinforcement learning, but learned from complexity signals rather than explicit rewards alone.

While specific prior work on using NCD for real-time LLM step guidance is limited, the principles of monitoring agent behavior and using meta-cognitive signals are established in AI research.

4 Dataset and Initial Analysis

4.1 Primary Dataset: GSM8K

For initial development and evaluation, we are using the GSM8K dataset [1], which consists of grade school math word problems. This dataset is widely used for benchmarking LLM reasoning and CoT abilities. Its problems typically require 2-8 reasoning steps. We have used the official training split for data collection (for training the complexity model) and a subset of the official test split for evaluation.

4.2 Data Collection for Complexity Model Training

We have collected trajectories by running baseline ReAct agents using Google's Gemma2-9B-Instruct-Tuned ('google/gemma-2-9b-it') and Meta's Llama3-8B-Instruct-Tuned ('meta-llama/Meta-Llama-3-8B-Instruct') on the GSM8K training set. For each step, we logged the thought, action, observation, local NCD, and global NCD. Each trajectory was marked for correctness. These trajectories, particularly successful ones, were used to train the VAE-Diffusion complexity model.

4.3 Preliminary Data Visualisations from Training Data

Initial analysis of the collected trajectories from the GSM8K training set provided insights into the relationship between complexity, solution length, and correctness, motivating the use of NCD for guidance. All NCD global values presented are the raw compressed lengths using zlib. (Note: Figures in this subsection are illustrative from the initial analysis phase using training data, specific model comparisons may vary from the final test results shown later.)

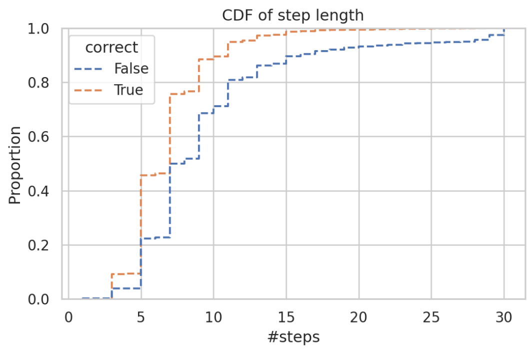


Figure 1: Illustrative Cumulative Distribution Function (CDF) of solution length (#steps) for correct (True) and incorrect (False) solutions from early analysis on GSM8K training data. Correct solutions tend to be shorter.

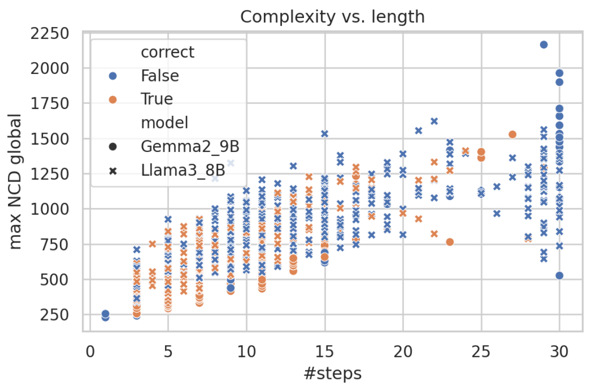


Figure 2: Max NCD global vs. solution length (#steps) from GSM8K test runs, comparing Base Gemma2-9B and Diffusion-Guided Gemma2-9B. Shows a general trend of increasing complexity with length. Markers differentiate correctness and model.

5 Technical Approach: NCD-Guided ReAct

Our core technical approach involves augmenting the ReAct agent with a complexity model trained on the NCD profiles of successful trajectories.

5.1 NCD Calculation

We use standard compressors (zlib, bz2, lzma, configurable) for NCD calculations [4].

- **Local NCD (NCD_{local}):** $NCD(\text{step}_{i-1}, \text{step}_i)$. Measures the complexity jump or novelty of the current step relative to the immediately preceding one.
- **Global NCD (NCD_{global}):** $C(\text{history}_i)$, the compressed length of the entire trajectory history up to and including step i. This serves as a proxy for cumulative cognitive effort.

These are computed for each 'ThoughtAction' and 'Observation' step. For training the complexity model, global NCD values are log-scaled ($log(1 + x)$) to manage their potentially large range.

5.2 Complexity Model: VAE-Diffusion

While initial explorations considered Transformer-based architectures, the experiments and results reported in this paper utilize a VAE-Diffusion model for complexity prediction and guidance.

- **Input Features per Step (x_t):** Similar to the Transformer approach, each step t in a trajectory is represented by a feature vector including step type, NCD measures, and normalized time step.
- **Model Architecture:** A Variational Autoencoder (VAE) combined with a Denoising Diffusion Probabilistic Model (DDPM) is trained to learn the latent distribution of "good" complexity trajectories from successful reasoning chains. The VAE encoder maps trajectories to a latent space, and the DDPM (acting as a denoiser on the latent variables) models this distribution.
- **Prediction Targets:** The model is trained to predict the likelihood of a partial trajectory belonging to the distribution of successful trajectories. This is used to score candidate steps.
- **Training:** The VAE-Diffusion model is trained on sequences extracted from successful trajectories.

This model provides a score indicating how "characteristic" a candidate step (and the resulting trajectory) is of successful problem-solving attempts.

5.3 Agent Integration: Step Gating (Discard/Retry)

The VAE-Diffusion complexity model (referred to as the "diffusion scorer") is integrated into the ReAct agent:

1. The ReAct agent generates candidate steps.
2. Before execution, each candidate is evaluated by the diffusion scorer. This involves forming a hypothetical feature sequence for the trajectory including the candidate.
3. The scorer provides a "success likelihood" or "appropriateness" score.
4. **Gating Decision:** If the score is below a threshold, the candidate step is discarded. The system makes multiple calls (e.g., four) to the complexity model per step to evaluate candidates.
5. **Retry Mechanism:** If a step is rejected, the agent may attempt to regenerate it, potentially with a modified prompt.

This mechanism aims to prune unpromising reasoning paths.

5.4 Self-Reflection (Phase 5 - Future Work)

While not the focus of the current results, a self-reflection phase, where the full trajectory and its NCD profile are critiqued by the LLM, remains a potential future extension.

6 Experimental Setup

6.1 Agents for Comparison

We evaluated two agent configurations using Google's Gemma2-9B-Instruct-Tuned LLM:

- **Base Gemma2-9B:** A standard ReAct agent without NCD-guidance.

- **Diffusion-Guided Gemma2-9B:** The ReAct agent augmented with the VAE-Diffusion complexity model for step gating, as described in Section 5.3.

6.2 Evaluation Test Set

Experiments were conducted on a randomly sampled subset of **300 problems** from the official GSM8K test set.

6.3 Metrics

The following metrics were used for evaluation:

- **Accuracy:** Percentage of correctly solved problems. Wilson score confidence intervals (95%) are reported. Statistical significance of accuracy differences is assessed using a 2-proportion z-test.
- **Efficiency:**
  - *a.* Average number of agent steps for successful solutions.
  - *b.* Average solution wall-clock latency.
- **Complexity Profiles:**
  - *a.* Mean maximum Global NCD (NCD_{global}) for successful solutions.
  - *b.* Distribution of NCD_{global} values and step counts.
  - *c.* Variance of NCD_{global} and step counts (Levene test for variance).
- **Reliability:** Success rate across different NCD buckets.

7 Results and Analysis

The NCD-guided agent, specifically the Diffusion-Guided Gemma2-9B, was compared against the Base Gemma2-9B agent on the 300-sample GSM8K test subset.

The Diffusion-Guided agent achieved a modest accuracy improvement of +1.7 percentage points (pp), which was not statistically significant (p=0.60) on this dataset size (Table 1). However, the guidance demonstrated other notable effects on the reasoning process.

Table 1: Performance Comparison on GSM8K Test Subset (300 Samples).

Metric	Base Gemma2-9B	Diffusion-Guided Gemma2-9B
Accuracy	80.0 % (240/300)	81.7 % (245/300)
95% CI (Wilson)	*75.0 – 84.4%*	*76.8 – 85.7%*
vs. Base (2-prop z-test)	–	*z=0.52, p=0.60 (n.s.)*
Mean Max NCD_{global} (successful)	498	485 (-2.6 %)
Mean #steps (successful)	6.5	5.7 (-12.3 %)
Wall-clock time (total)	3152 s	13 092 s ($\times$4.2)

Note: Mean Max NCD_{global} and Mean #steps are averaged over successful problems. Latency is total for 300 problems.

7.1 Efficiency and Latency Trade-off

While the guided agent reduced the average number of steps for successful solutions by approximately 0.8 steps (from 6.5 to 5.7), this came at a significant cost in wall-clock time. The Diffusion-Guided agent was about 4.2 times slower than the baseline (Figure 3). This increased latency is primarily due to the overhead of the diffusion scorer, which was called multiple times per reasoning step for candidate evaluation. As seen in Figure 3, the guided agent (orange) often operates at a higher latency for similar complexity levels compared to the baseline (blue).

Interestingly, guidance helps prune very high-complexity attempts. Figure 3 (and supported by scatter plots like Figure 2) shows that the guided agent has fewer correct completions at very high max NCD_{global} values (e.g., above 1200), where the baseline agent still attempts solutions but frequently fails. This suggests the guided agent spends its (longer) time on trajectories considered ”safer” by the complexity model.

7.2 Complexity Distribution and Predictability

NCD-guidance led to a more predictable reasoning behavior by reducing variance in complexity and step counts (Levene test p ¡ 0.01 for both max NCD_{global} and #steps). The histograms in Figure 4 show that both agents concentrate successful solutions (True) in a similar NCD_{global} range (approx. 300-550). However, for incorrect solutions (False), the guidance shifts the distribution leftward and reduces its variance. For instance, the 95th percentile of NCD_{global} for failed trajectories dropped from around 1340 for the baseline to 1100 for the guided

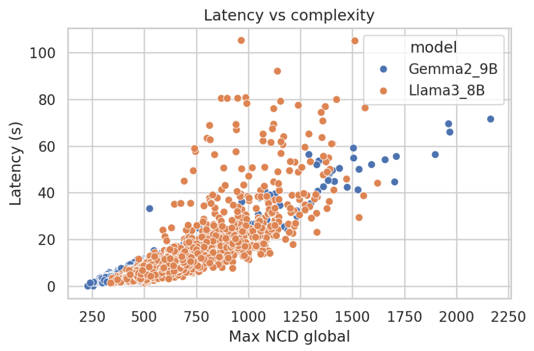


Figure 3: Latency vs. Maximum NCD global for Base Gemma2-9B and Diffusion-Guided Gemma2-9B. The guided agent shows higher latency for comparable complexity but avoids some high-complexity failure modes.

agent.

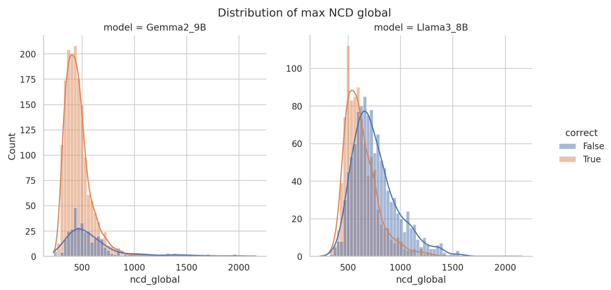
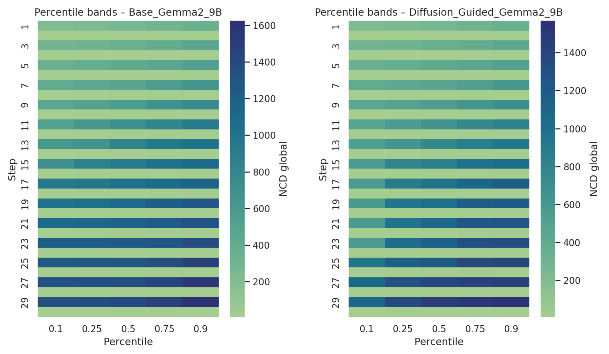


Figure 4: Distribution of maximum global NCD for correct and incorrect solutions, comparing Base Gemma2-9B (left) and Diffusion-Guided Gemma2-9B (right). Guidance tends to tighten the distribution for incorrect solutions.

The percentile band visualizations (whisker plots, Figure 5) further illustrate this. The Diffusion-Guided agent (Figure 5b) maintains the 0.9 quantile of NCD_{global} approximately 200 NCD units below the baseline (Figure 5a) for a significant portion of the trajectory horizon. However, the mean NCD_{global} traces over steps are nearly indistinguishable between the two agents (Figure 6). This suggests the guidance primarily acts as a risk filter, pruning high-complexity outliers rather than drastically lowering the average complexity explored.



(a) Base Gemma2-9B (b) Guided Gemma2-9B

Figure 5: Percentile bands of NCD global values at each step index for Base and Diffusion-Guided Gemma2-9B. Shows the spread and evolution of trajectory complexity. Guidance tends to constrain the upper percentiles.

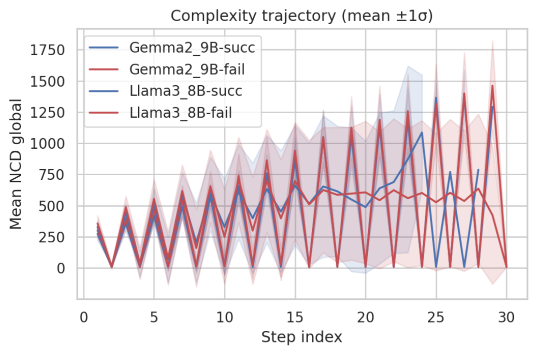


Figure 6: Mean NCD global (trajectory complexity) vs. step index for successful and failed trajectories (Base and Diffusion-Guided Gemma2-9B), showing mean $\pm 1\sigma$. Mean traces are similar, but variance differs.

7.3 Step Economy

The NCD-guidance improved step economy. The CDF of step length (Figure 7, using results data) shows that by the 10th step, 75

7.4 Reliability Analysis

The reliability curve (Figure 9), plotting success rate against binned max NCD_{global}, shows that guidance gen-

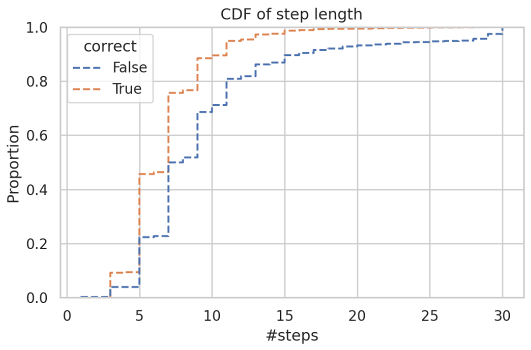


Figure 7: Cumulative Distribution Function (CDF) of solution length (#steps) for correct (True) and incorrect (False) solutions from GSM8K test runs.

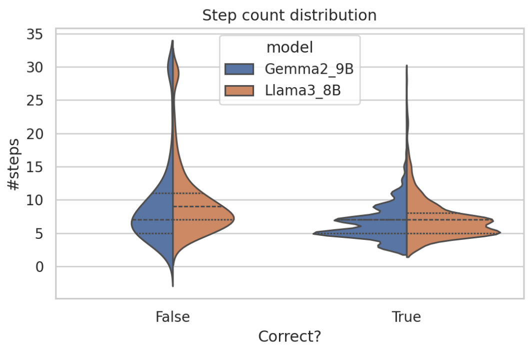


Figure 8: Step count distribution for correct and incorrect solutions, comparing Base Gemma2-9B and Diffusion-Guided Gemma2-9B. Guidance reduces median steps and curtails long failure trajectories.

erally helps or maintains performance across complexity bins, with one exception. For problems in the 350 $\leq NCD < 450$ range, guidance slightly improved accuracy (+2pp). However, in the 450 $\leq NCD < 550$ range, it led to a decrease (-8pp), suggesting the guided agent might sometimes over-prune viable, shorter candidates it deems "too simple" or atypical. For very high complexity (NCD $\geq$ 950), both agents perform poorly, but guidance offers a slight edge. Overall, the guided system exhibits a flatter slope, indicating more graceful degradation in performance as problem complexity (measured by NCD) increases.

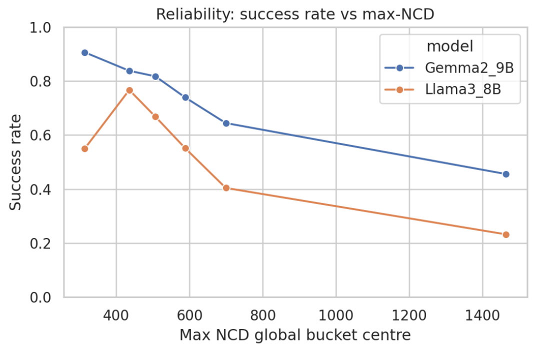


Figure 9: Success rate vs. binned Maximum NCD global for Base Gemma2-9B and Diffusion-Guided Gemma2-9B. Guidance offers more graceful degradation but can over-prune in certain low-mid complexity bins.

7.5 Discussion Summary

The NCD-guided approach, using a diffusion scorer, acts as a complexity-aware risk control mechanism. It leads to:

- **Safer, more predictable reasoning:** Evidenced by narrower complexity distributions for failures (Figure 4), constrained upper NCD quantiles during trajectory generation (Figure 5), and reduced variance in max-NCD and step counts. This suggests guidance prunes high-risk, overly complex branches.
- **Modest, statistically insignificant accuracy uplift:** A +1.7pp gain was observed, but a larger sample size would be needed to confirm significance. The current data (p=0.60) is consistent with this difference being due to noise.
- **Efficiency trade-off:** While mean steps for successful solutions decreased, the computational overhead of the guidance mechanism led to a significant increase in latency (Figure 3). This highlights the need for optimizing the scorer's efficiency (e.g., through batching, caching, or model distillation).
- **Potential for over-pruning:** The drop in reliability for the 450-550 NCD bin suggests the model might

occasionally discard valid, efficient solutions. This could be addressed by refining the scoring mechanism or adding guard-bands.

The method provides enhanced interpretability through post-hoc complexity trace analysis and demonstrates a path towards more stable LLM reasoning.

8 Next Steps and Future Work

Based on these findings, future work will focus on:

1. **Optimizing Guidance Efficiency:** Reducing the latency of the diffusion scorer through techniques like model distillation, quantization, batching, or developing more lightweight complexity models.
2. **Refining Gating Strategy:** Addressing the over-pruning in certain NCD ranges, perhaps by incorporating a "too-simple" guard-band or combining NCD-guidance with cheaper heuristics.
3. **Larger Scale Evaluation:** Evaluating on a larger dataset to achieve greater statistical power for accuracy improvements.
4. **Exploring Transformer-based Guidance:** Completing the development and comparison with Transformer-based complexity models as initially proposed.
5. **Broader Applications:** Testing the NCD-guidance approach on other LLMs, more complex mathematical datasets (e.g., MATH, SVAMP), and potentially other reasoning tasks beyond mathematics.
6. **Advanced Reflection Mechanisms:** Implementing and evaluating the self-reflection phase (Phase 5) based on NCD profiles.

9 Conclusion

This project introduced NCD-Guided Reasoning, a novel method for enhancing LLM-based mathematical reasoning by integrating real-time complexity signals. We developed and evaluated an NCD-guided ReAct agent using a VAE-Diffusion based complexity model with a step-gating mechanism, comparing it against a baseline Gemma2-9B agent on a 300-sample GSM8K test subset.

Our NCD-guided agent achieved a marginal accuracy improvement of +1.7 percentage points (p=0.60) and a reduction in the average number of steps for correct solutions (5.7 vs. 6.5). More significantly, the guidance led to a substantial reduction in the variance of maximum NCD and step counts, indicating more predictable and stable reasoning behavior by pruning high-risk, overly complex solution paths. However, these benefits came with a considerable increase in latency (approximately 4.2x) due to the computational overhead of the current diffusion-based guidance mechanism.

These findings suggest that NCD-guidance is a promising approach for complexity-aware risk control in LLM agents, fostering more stable and interpretable reasoning. While the direct accuracy boost was modest and not statistically significant in this study, the ability to make agent behavior more predictable is a valuable step. Future work will focus on improving the computational efficiency of the guidance, refining the gating strategy to avoid over-pruning, and conducting larger-scale evaluations to further validate accuracy improvements.

References

[1] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training Verifiers to Solve Math Word Problems. In *arXiv preprint arXiv:2110.14168*, 2021. 1, 3

[2] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. *arXiv preprint arXiv:2103.03874*, 2021. 1

[3] A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Problemy Peredachi Informatsii*, 1(1):3–11, 1965. English translation in International Journal of Computer Mathematics, 2(1-4):157-168, 1968. 2

[4] M. Li, X. Chen, X. Li, B. Ma, and P. M. Vitányi. The Similarity Metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004. 1, 2, 4

[5] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, S. Krishna, G. Neubig, and A. Yazdanbakhsh. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2

[6] N. Shinn, B. Labash, and A. Gopinath. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2

[7] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[8] S. Yao, J. Zhao, D. Yu, N. Du, Y. Tsvetkov, Y. LeCun, and S. Singh. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 2