

VecCISC: Improving Confidence-Informed Self-Consistency with Reasoning Trace Clustering and Candidate Answer Selection

James Petullo* and Sonny George and Dylan Cashman and Nianwen Xue

Computer Science Department, Brandeis University

jamespetullo@brandeis.edu

*Corresponding author

Abstract

A standard technique for scaling inference-time reasoning is Self-Consistency, whereby multiple candidate answers are sampled from an LLM and the most common answer is selected. More recently, it has been shown that weighted majority voting (e.g. Confidence-Informed Self Consistency (CISC)), which assigns a confidence value to each candidate answer and chooses the answer with the largest accumulated score, tends to be more accurate on a wide range of popular benchmarks. In practice, weighted majority voting necessitates calling a critic LLM on each candidate’s reasoning trace to produce the answer’s confidence score. This secondary series of LLM calls greatly increases the overhead and cost of weighted majority voting, despite its potential performance benefits. To reduce this expense, we propose VecCISC, a lightweight, adaptive framework that uses a measure of semantic similarity to filter reasoning traces that are semantically equivalent to others, degenerate, or hallucinated, thus decreasing the number of candidate answers that must be evaluated by the critic. To ensure adequate experimental thoroughness, we evaluate VecCISC on five challenging, widely-adopted datasets spanning the domains of mathematics, chemistry, biology, commonsense reasoning, and the humanities. Our results demonstrate that VecCISC reduces the total token usage by 47%, while maintaining or exceeding the accuracy of CISC.

1 Introduction

Scaling inference-time reasoning has become an effective and popular approach to improving the performance of LLMs on reasoning tasks (Snell et al., 2024; Liu et al., 2025; Parashar et al., 2025; Zhang et al., 2025a). At its core, scaling inference-time reasoning involves sampling multiple outputs from an LLM and, from them, producing a final answer. The underlying assumption is that a larger set of diverse reasoning traces will lead to a greater

number of potential answers, thus improving coverage of the solution search space. The foundational technique for inference scaling is Self-Consistency (Wang et al., 2022), whereby an LLM is called multiple times for a given prompt, and the most common answer is returned. Since these methods can be costly, efforts have been made to decrease sampling costs while maintaining accuracy (Chen et al., 2024). These approaches include early stopping (Huang et al., 2025; Li et al., 2024c; Aggarwal et al., 2023), mixture of models (Chen et al., 2025; Zhang et al., 2025b), and query routing (Ong et al., 2024). Recently, it has been shown that confidence-informed Self-Consistency (CISC) (Taubenfeld et al., 2025) can be utilized in conjunction with weighted majority voting to improve the accuracy of Self-Consistency. CISC, and the broader concept of LLM confidence calibration, belongs to the “think twice” paradigm (Li et al., 2024a), whereby an LLM first generates an answer and an associated reasoning trace, and then a critic LLM is prompted to assess the answer-trace pair. The critic produces a score that reflects its confidence that the answer is correct. These scores are then used as part of a weighted majority vote to select the final answer. Throughout this paper, we use the term “think twice” to refer to any method that necessitates calling a critic for confidence estimation.

However, a significant drawback to “think twice” approaches is that the inference overhead increases significantly, as each sample requires at least one additional LLM call, often with a token-heavy prompt. Furthermore, naive CISC makes no distinction between hallucinated, degenerate samples and higher quality outputs, running the critic LLM on all regardless.

As such, we introduce VecCISC, a flexible, conceptually straightforward candidate answer aggregation framework that reduces the number of samples needed to be evaluated by the critic LLM

through clustering candidate answers via embeddings of their corresponding reasoning traces. By adding only minimal overhead through the application of an embedding model, our clustering approach filters redundant traces that possess the same semantic meaning (see Appendix A.3) and can aid in isolating answers that contain flawed, hallucinated, and degenerate reasoning (see Appendix A.4). After clustering, a representative answer from each cluster is selected to be passed to the critic LLM, thus significantly reducing the number of calls that have to be made to the latter. In sum, our contributions are as follows:

- We introduce VecCISC, a novel method to reduce the cost of the “think twice” paradigm by utilizing reasoning trace clustering to eliminate redundant and low quality samples from the pool to be analyzed;
- We empirically demonstrate the effectiveness of our approach by performing extensive experiments on a range of models and datasets, with VecCISC achieving an average total token budget reduction of 47% across the entire inference pipeline, all while maintaining and even exceeding the performance of CISC;
- We compare the performance of VecCISC with two different clustering approaches against random candidate selection of K samples, demonstrating the efficacy of our proposed clustering approach;
- We will release our code publicly to support further research.

2 Methodology

This section details each step in the VecCISC pipeline, including embedding generation from answer reasoning traces, clustering, and candidate answer selection. The overall procedure is summarized in Figure 1, and a comparison of VecCISC’s methodology to that of Self-Consistency and CISC can be found in Figure 2.

2.1 Sampling

Given a sampling budget n , a large language model LLM_{gen} , and a question q , we first sample n reasoning trace-answer pairs from LLM_{gen} :

$$(r_1, a_1), \dots, (r_n, a_n) \sim LLM_{gen}(q) \quad (1)$$

The samples are then added to the answer candidate pool P :

$$P = \{(r_1, a_1), \dots, (r_n, a_n)\} \quad (2)$$

2.2 Embedding Generation

For each reasoning trace in the pool, a text embedding is produced via an embedding model Emb :

$$P = \{(r_i, Emb(r_i), a_i) \mid (r_i, a_i) \in P\}, \\ Emb(r_i) \in \mathbb{R}^d \quad (3)$$

Where d is the embedding dimension.

These embeddings enable the pipeline to capture semantic details in each reasoning trace that allow for clustering and outlier detection in subsequent steps.

2.3 Grouping By Answer

To preserve the set of distinct candidate answers, we group embeddings by answer before clustering. That is, for each unique answer a present in the sampled outputs:

$$G_a = \{(r_i, Emb(r_i)) \mid (r_i, Emb(r_i), a_i) \in P, \\ a_i = a\} \quad (4)$$

Where G_a is the group of embedded reasoning traces that lead to answer a .

2.4 Clustering

Next, we apply a clustering algorithm $Cluster$ to each reasoning trace group G_a :

$$CL_a = Cluster\left(\{Emb(r_i) \mid (r_i, Emb(r_i)) \in G_a\}, \\ \min(K, |G_a|)\right) \quad (5)$$

Where K is a hyperparameter controlling the number of clusters to be formed. If K exceeds the number of samples in G_a , then $|G_a|$ clusters are created instead. In our experiments, we analyzed KMeans and Hierarchical Agglomerative (HAC) clustering, as both methods allow for a distinct number of clusters to be specified. For both clustering algorithms, we performed a grid search on a holdout set from each (*dataset, model*) combination to choose K . The hyperparameters selected

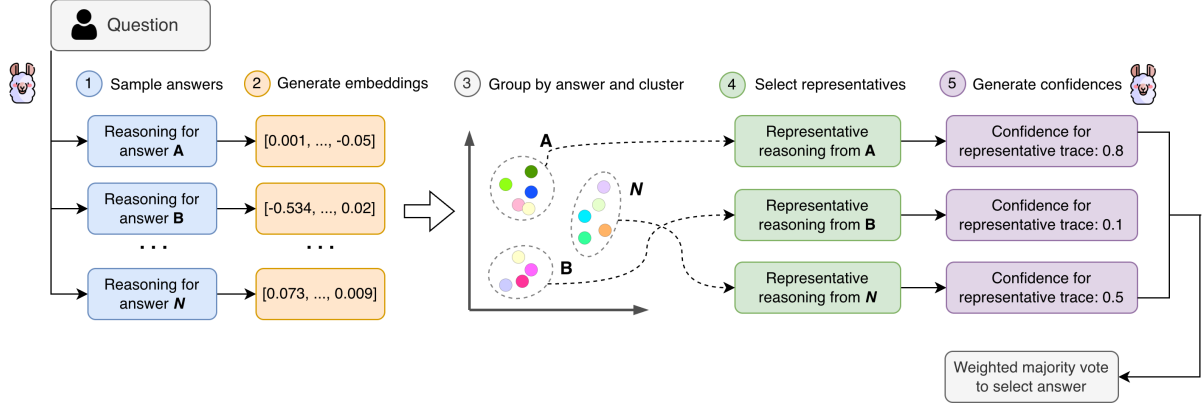


Figure 1: Overview of the VecCISC pipeline. Embeddings of the sampled reasoning traces are clustered within each unique answer returned from the LLM. A representative trace is chosen from each cluster, and these representatives are then used to generate the confidence scores for the answers. Lastly, weighted majority vote is performed to choose the final answer.

and further technical details are listed in Appendix A.

We choose KMeans and HAC over Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996), another popular choice for clustering embeddings, as DBSCAN requires a distance threshold D to define clusters. D specifies the maximum distance between two vectors V_1 and V_2 for them to be considered part of the same neighborhood. Thus, V_1 and V_2 are clustered together if $dist(V_1, V_2) \leq D$. In high-dimensional language model embedding spaces, small changes to D can drastically alter cluster assignments, making threshold selection both difficult and prone to producing suboptimal clusters.

2.5 Cluster Representative Selection

Upon creation of the clusters, we then pick a representative from each cluster $C_i \in CL_a$. To do so, we first compute the centroid u_i of C_i :

$$U_i = \frac{1}{|C_i|} \sum_{e \in C_i} e \quad (6)$$

Finally, we choose the reasoning trace $r_{rep,i} \in C_i$ that is closest to the centroid:

$$r_{rep,i} = \arg \min_{r \in C_i} \cos_sim(\text{Emb}(r), u_i) \quad (7)$$

Here, we use the cosine similarity function as our definition of distance, as it emphasizes the angle θ between vectors instead of the magnitude, thus making it more suitable for high-dimensional

spaces requiring the preservation of semantic overlap (Li et al., 2024c). Cosine similarity is defined in Equation 8:

$$\cos_sim(\text{emb}_1, \text{emb}_2) = \frac{\text{emb}_1 \cdot \text{emb}_2}{\|\text{emb}_1\| \cdot \|\text{emb}_2\|} \quad (8)$$

Selecting the reasoning trace closest to the centroid follows from the observation that the least semantically deviant trace would be the least likely to contain anomalous reasoning errors. At the end of the selection process, each answer now possesses a set of representative traces:

$$\text{Rep}_a = \{r_{rep,1}, \dots, r_{rep,k}\}, \quad (9)$$

where $k = |\text{Clusters}_a|$, $k \leq |G_a|$

2.6 Confidence Scoring

For each unique answer a and its associated representative traces $r_i \in \text{Rep}_a$, we derive a new prompt query $q'_i = [\text{instruction}] \circ q \circ r_i \circ a$, where \circ is the concatenation operator and $[\text{instruction}]$ prompts the critic LLM to reflect on the original question, reasoning trace, and answer, and produce a confidence score in the range of 0 to 1:

$$c_{a,i} = LLM_{critic}(q'_i) \quad (10)$$

Following (Taubenfeld et al., 2025), we normalize each confidence score using softmax:

$$\hat{c}_{a,i} = \frac{\exp\left(\frac{c_{a,i}}{T}\right)}{\sum_{j=1}^K \exp\left(\frac{c_{a,j}}{T}\right)} \quad (11)$$

Where T is a tunable hyperparameter of softmax that controls the smoothness of the normalization

distribution. As with the cluster number K , we perform a grid search to find T . Please see Appendix A for more details and T values from our experiments.

We select the final answer by using a weighted majority vote:

$$A_{\text{final}} = \arg \max_a \sum_{(a_j, \hat{c}_{a,j}) \in P} 1[a_j = a] \cdot \hat{c}_{a,j} \quad (12)$$

3 Experiments

3.1 Datasets

We evaluated the performance of VecCISC on five popular QA datasets:

- **AQuA_Rat** (Ling et al., 2017): A multiple choice QA dataset consisting of algebraic word problems. We sampled 1000 questions from the training set.
- **CommonsenseQA** (Talmor et al., 2019): A dataset that requires commonsense reasoning capabilities to solve problems across a spectrum of different domains. We sampled 1000 problems from the validation set.
- **ARC-Challenging** (Clark et al., 2018): A subset of the most challenging questions in the original ARC dataset, including grade-school level multiple choice science questions. We sampled 1000 problems from the test set.
- **MMLU-Pro** (Wang et al., 2024): A more challenging set of questions from the original MMLU dataset, designed to assess reasoning faculty across disciplines including math, science, economics, and history. We sampled 1000 problems from the test set.
- **GPQA** (Rein et al., 2023): A multiple-choice dataset of very challenging questions crafted by experts in biology, physics, and chemistry. We used all 546 questions in the dataset.

3.2 Models

For our experiments, we selected five widely used open and closed source LLMs: GPT-4o mini (OpenAI, 2024a), Llama 3.1 8B (AI, 2024), Llama 3.3 70B Versatile (Meta Platforms, 2024), Qwen 2.5 7B (Qwen et al., 2024), and Mistral 7B v0.1 (AI, 2023). Model hyperparameters can be found in Appendix A.

3.3 Embeddings

For all experiments, we used OpenAI’s text-embedding-3-small model (OpenAI, 2024b) for the embedding model Emb . This model was chosen to balance cost and general purpose applicability across a plethora of different reasoning tasks. As our goal is to design a lightweight, adaptive framework that will be performant regardless of reasoning trace composition, text-embedding-3-small was most suitable for both our task’s scope and budget limitations.

3.4 Baselines

We compare VecCISC to both standard Self-Consistency and CISC. In particular, we apply three distinct flavors of methodology: **VecCISC (random)**, **VecCISC + KMeans** and **VecCISC + HAC**. Instead of clustering, **VecCISC (random)** randomly samples K reasoning traces from each answer-grouped set, G_a , thus acting as an ablation for our method. All prompts were zero-shot. For the sake of experimental validity and to ensure a proper apples-to-apples comparison, we reimplemented both Self-Consistency and CISC, executing them on the same set of sampled questions used to evaluate the various VecCISC configurations. Following (Taubenfeld et al., 2025), we selected verbal confidence as the confidence extraction method, whereby the critic LLM is asked to rate its confidence in the range of 0 to 1.

3.5 Metrics

3.5.1 Budget Call Reduction

We report the reduction in the number of calls made to LLM_{critic} with VecCISC, as opposed to CISC:

$$\text{Reduction} = 100 \times \frac{\# \text{ of critic calls in VecCISC}}{\# \text{ of calls in CISC}} \quad (13)$$

We computed the reduction for each $(dataset, model)$ combination by running our pipeline on each question in the dataset with its corresponding model and recorded the average across all questions and experiments.

3.5.2 Token Usage Reduction

In addition, we report the reduction in token usage exhibited by VecCISC as opposed to CISC. Within VecCISC, we compare two cluster representative selection strategies: our proposed minimum centroid (min-centroid) approach and random cluster

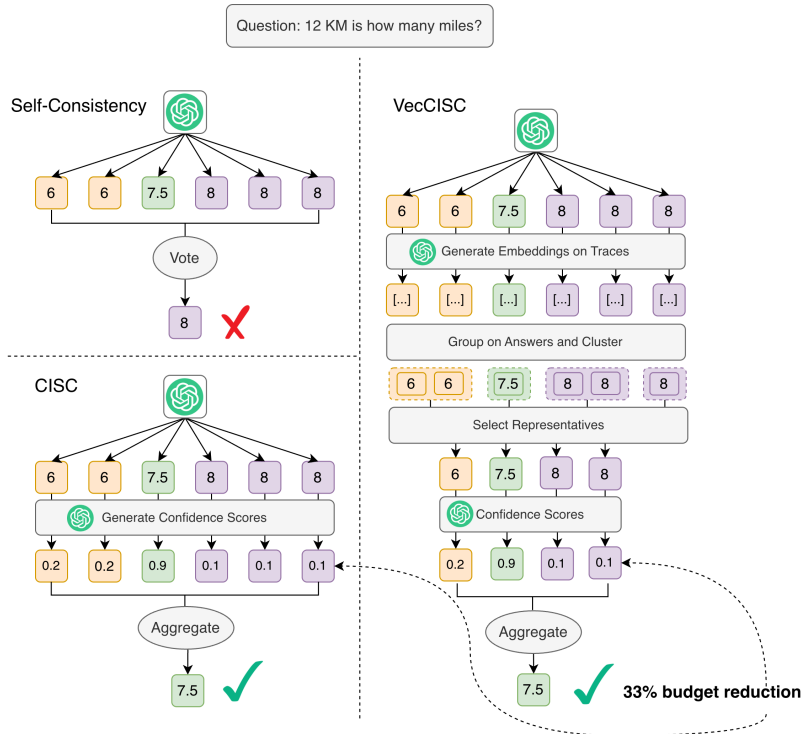


Figure 2: Comparison of VecCISC to Self-Consistency (SC) and CISC. While CISC represents an improvement upon SC via incorporation of confidence scores and weighted majority voting, the inference overhead costs of CISC increase twofold over SC. By incorporating language embeddings, clustering, and representative trace selection, the confidence scorer LLM_{critic} can be run on a smaller set of higher quality reasoning traces while still preserving relative performance.

value selection (rand-trace), the latter acting as an ablation of the former.

3.5.3 Accuracy

Lastly, we report the accuracy attained by VecCISC and its variants, CISC, and Self-Consistency (SC) on each dataset and model. KMeans clustering and random trace sampling are non-deterministic, thus, we ran each method 10 times for every $(dataset, model)$ combination and reported both the best and average.

4 Results and Discussion

Our budget reduction results can be found in Tables 1 and 2, while the token reduction results can be found in Tables 3 and 4. In addition, performance results can be found in Table 5.

4.1 Call Budget Reduction

VecCISC + KMeans and **VecCISC + HAC** significantly reduces the number of calls to LLM_{critic} , while achieving comparable or improved performance over CISC. Across all models and datasets, **VecCISC + KMeans** averages a 34.68% reduction

in the number of critic calls (Table 1), while **VecCISC + HAC** averages a 30.2% reduction in the number of critic calls (Table 2). Across the entire pipeline, including both the Self-Consistency sampling and CISC component, **VecCISC + KMeans** averages a 17.34% total LLM call reduction and **VecCISC + HAC** averages a 15.1% total call reduction.

4.2 Token Usage Reduction

By reducing the number of critic calls made, VecCISC variants reduce the number of tokens utilized by LLM_{critic} . Our proposed cluster representative selection (min-centroid) records the lowest token usage rates as opposed to random representative selection (rand-trace) on 15 of the 25 model-dataset combinations (60%) on **VecCISC + KMeans**, as shown in Table 3. On **VecCISC + HAC**, min-centroid records the lowest token usage rates on 17 of the 25 model-dataset combinations (68%), as shown in Table 4. These results demonstrate that our min-centroid approach leads to the selection of higher quality, lower-token reasoning traces as opposed to purely random reasoning trace selection from each cluster. On average, **VecCISC +**

Dataset	Budget	GPT-4o mini	Llama 3.1 8B	Qwen2.5 7B	Mistral 7B	Llama3.3 70B
AQuA-RAT	20	5.660 _(-71.72%)	13.45 _(-32.77%)	14.78 _(-26.09%)	17.05 _(-14.74%)	2.600 _(-86.99%)
CommonsenseQA	20	13.81 _(-30.95%)	12.70 _(-36.50%)	12.45 _(-37.76%)	15.54 _(-22.31%)	11.79 _(-41.02%)
ARC-Challenging	20	13.34 _(-33.29%)	14.80 _(-26.00%)	10.94 _(-45.30%)	15.65 _(-21.73%)	13.31 _(-33.43%)
MMLU-Pro	20	12.28 _(-38.61%)	14.81 _(-25.94%)	12.81 _(-35.93%)	17.47 _(-12.66%)	13.18 _(-34.10%)
GPQA	20	10.56 _(-47.19%)	13.88 _(-30.57%)	16.41 _(-17.93%)	10.03 _(-49.86%)	17.27 _(-13.63%)

Table 1: **VecCISC + KMeans** confidence evaluator call budget reductions, averaged across 10 runs. **min-centroid** cluster candidate selection was used for all experiments. **VecCISC + KMeans** records a 34.68% reduction in LLM_{critic} calls.

Dataset	Budget	GPT-4o mini	Llama 3.1 8B	Qwen2.5 7B	Mistral 7B	Llama3.3 70B
AQuA-RAT	20	13.67 _(-31.64%)	13.45 _(-32.77%)	14.78 _(-26.09%)	17.05 _(-14.74%)	8.100 _(-59.52%)
CommonsenseQA	20	11.89 _(-40.56%)	12.70 _(-36.50%)	15.23 _(-23.83%)	15.54 _(-22.31%)	11.79 _(-41.02%)
ARC-Challenging	20	10.39 _(-48.07%)	14.80 _(-26.00%)	10.94 _(-45.30%)	15.65 _(-21.73%)	13.31 _(-33.43%)
MMLU-Pro	20	12.28 _(-38.61%)	15.56 _(-22.18%)	12.81 _(-35.93%)	18.02 _(-9.890%)	11.29 _(-43.55%)
GPQA	20	10.56 _(-47.19%)	16.00 _(-20.02%)	16.41 _(-17.93%)	18.81 _(-5.940%)	17.95 _(-10.24%)

Table 2: **VecCISC + HAC** confidence evaluator call budget reductions, averaged across 10 runs. **min-centroid** cluster candidate selection was used for all experiments. **VecCISC + HAC** records a 30.2% reduction in LLM_{critic} calls.

Dataset	Method	GPT 4o-mini	Llama3.1 8B	Qwen2.5 7B	Mistral 7B	Llama3.3 70B
AQuA-RAT	CISC	13042.0	15480.9	11674.3	9317.2	14448.1
	VecCISC (rand-trace)	8914.5	<u>10500.6</u>	<u>8566.6</u>	<u>7652.0</u>	<u>6289.3</u>
	VecCISC (min-centroid)	8915.4	10483.7	8562.6	7649.6	6280.7
CommonsenseQA	CISC	11086.2	10932.4	10629.9	7691.9	13108.5
	VecCISC (rand-trace)	<u>6546.9</u>	6611.4	8044.4	<u>5663.9</u>	7747.4
	VecCISC (min-centroid)	6546.7	<u>6612.5</u>	<u>8044.7</u>	5663.7	<u>7748.2</u>
ARC-Challenging	CISC	12245.4	12292.1	11602.7	8764.0	14715.5
	VecCISC (rand-trace)	<u>6367.2</u>	<u>8913.2</u>	6327.2	6733.4	<u>9863.1</u>
	VecCISC (min-centroid)	6365.6	8912.3	<u>6327.5</u>	<u>6734.1</u>	9856.5
MMLU-Pro	CISC	16975.1	18496.6	16474.9	13926.1	20810.4
	VecCISC (rand-trace)	<u>10173.7</u>	<u>13616.7</u>	<u>10060.1</u>	11140.1	<u>11840.2</u>
	VecCISC (min-centroid)	10172.4	13614.9	10059.9	<u>11143.5</u>	11834.7
GPQA	CISC	18594.2	19659.5	17937.0	14324.6	25235.3
	VecCISC (rand-trace)	9574.2	<u>15224.9</u>	14419.0	<u>13262.6</u>	22386.5
	VecCISC (min-centroid)	<u>9577.6</u>	15211.4	<u>14422.6</u>	13260.5	<u>22389.4</u>

Table 3: **VecCISC + KMeans** token reductions with cluster candidate selection variants, averaged across 10 runs. In each column, the best results are in **bold**, and the second-best results are underlined. **VecCISC + KMeans** records a 36.2% reduction in tokens expended on LLM_{critic} calls.

KMeans reduces critic token usage by 36.2% (Table 3), while **VecCISC + HAC** records a 31.69% token usage reduction (Table 4). LLM_{critic} calls are the most token-heavy of the entire pipeline, accounting for 77% of all token usage, thus, at scale, these reductions represent significant cost savings. Averaging token consumption across the entire pipeline, both variants of our proposed methodology reduce token usage by 47%¹.

4.3 Accuracy

As shown in Table 5, **VecCISC + KMeans** and **VecCISC + HAC** consistently outperform CISC and Self-Consistency across nearly all models and datasets. **VecCISC + HAC** records the best average result on all (*dataset, model*) combinations except for (AQuA_RAT, Mistral 7B) and (CommonsenseQA, GPT 4o-mini), where it scores the second highest. On (AQuA_RAT, Mistral 7B) and (CommonsenseQA, GPT 4o-mini), **VecCISC + KMeans** performs the best. As expected, **VecCISC (random)**, which randomly samples candidate reasoning traces from each answer group G_a as opposed to clustering, significantly underperforms both CISC and Self-Consistency, thus demonstrating the efficacy of clustering reasoning traces to find an ideal representative candidate.

5 Related Work

5.1 Embeddings and Clustering for Scaling Inference Time Reasoning

Recent work has studied the application of LLM embeddings and clustering to improve model accuracy on reasoning and calibration benchmarks. (Knappe et al., 2024) uses embeddings of reasoning traces to directly compute a sample score for use in weighted majority voting. Although conceptually straightforward, this approach is unrelated to the “think twice” paradigm, relies upon a fleet of domain-specific, fine-tuned embedding models for different benchmarks, and exhibits inconsistent performance improvements, often dropping pipeline accuracy below baseline Self-Consistency. (Lee et al., 2025) utilizes an LLM’s hidden state as context embeddings to cluster reasoning traces in tree-of-thought (ToT) pipelines and improve the accuracy of LLM uncertainty estimation. Although lightweight in practice, their proposed method is

¹To ensure fair approximations across all models tested, we computed the number of tokens used in each prompt as $length(prompt)/4$.

only applicable to white-box LLMs, limiting its reach and accessibility, while not addressing the need for candidate trace selection for weighted majority voting. Lastly, (Abdaljalil et al., 2025) clusters embeddings of LLM responses to detect instances of hallucinations.

5.2 Think-twice Approaches

Utilizing an LLM as a critic of its own outputs (or the outputs of other LLMs) has been the topic of much investigation. In particular, efforts have been made to use LLM-as-a-judge approaches to improve the accuracy of standard self-consistency. (Li et al., 2024b) samples a set of unique answers from an LLM and then prompts a second LLM to generate a justification for each candidate answer, before using a third LLM call to produce a final confidence score. Unlike CISC, this methodology does not make use of the sample reasoning traces, choosing instead to analyze only the final answers themselves, overlooking key contextual clues in the provided answers’ reasoning traces. Lastly, (Wang et al., 2025) applies Monte Carlo Tree Search (MCTS) with an LLM-as-a-judge reward function to improve the accuracy of code-writing pipelines.

6 Conclusion

While the “think twice” paradigm can increase model performance on a variety of reasoning tasks, the additional inference overhead required can render it prohibitively expensive. In this work, we propose a novel framework for reducing the cost of model confidence estimation at inference time by clustering embeddings of candidate answer reasoning traces. Our experimental results demonstrate that choosing answer representatives from embedding clusters can significantly decrease inference costs during the critic component of the “think twice” approach without a commensurate decline in performance. Indeed, our results indicate that passing a smaller number of higher-quality reasoning traces to the critic can lead to small gains in accuracy. In this way, our methodology emerges as an effective, lightweight augmentation of pre-existing think-twice pipelines.

7 Limitations

Our study emphasizes the adaptability of our framework. To this end, we adopt a general text embedding model for our experiments. Pipelines for

Dataset	Method	GPT 4o-mini	Llama3.1 8B	Qwen2.5 7B	Mistral 7B	Llama3.3 70B
AQuA-RAT	CISC	13042.0	15480.9	11674.3	9317.2	14448.1
	VecCISC (rand-trace)	8914.0	<u>10499.9</u>	<u>8565.2</u>	<u>7652.3</u>	<u>6291.0</u>
	VecCISC (min-centroid)	<u>8915.4</u>	10483.7	8562.6	7649.6	6280.7
CommonsenseQA	CISC	11086.2	10932.4	10629.9	7691.9	13108.5
	VecCISC (rand-trace)	6546.6	6611.0	<u>8044.9</u>	<u>5664.7</u>	7746.3
	VecCISC (min-centroid)	<u>6546.7</u>	<u>6612.5</u>	8044.7	5663.7	<u>7748.2</u>
ARC-Challenging	CISC	12245.4	12292.1	11602.7	8764.0	14715.5
	VecCISC (rand-trace)	<u>6367.3</u>	<u>8914.3</u>	<u>6328.4</u>	<u>6735.3</u>	<u>9862.4</u>
	VecCISC (min-centroid)	6365.6	8912.3	6327.5	6734.1	9856.5
MMLU-Pro	CISC	16975.1	18496.6	16474.9	13926.1	20810.4
	VecCISC (rand-trace)	<u>10173.6</u>	<u>13616.6</u>	<u>10061.0</u>	11141.2	<u>11840.5</u>
	VecCISC (min-centroid)	10172.4	13614.9	10059.9	<u>11143.5</u>	11834.7
GPQA	CISC	18594.2	19659.5	17937.0	14324.6	25235.3
	VecCISC (rand-trace)	9573.1	<u>15225.0</u>	14421.6	<u>13261.4</u>	22382.6
	VecCISC (min-centroid)	<u>9577.6</u>	15211.4	<u>14422.6</u>	13260.5	<u>22389.4</u>

Table 4: **VecCISC + HAC** token reductions with cluster candidate selection variants, averaged across 10 runs. In each column, the best results are in **bold**, and the second-best results are underlined. **VecCISC + HAC** records a 31.69% reduction in tokens expended on $LLM_{critical}$ calls.

Dataset	Method	GPT 4o-mini	Llama3.1 8B	Qwen2.5 7B	Mistral 7B	Llama3.3 70B
AQuA-RAT		Best / Average		Best / Average		
	SC Baseline	83.5 / 83.5	82.6 / 82.6	<u>84.5 / 84.5</u>	30.7 / 30.7	86.6 / 86.6
	CISC	84.0 / <u>84.0</u>	<u>82.9 / 82.9</u>	85.5 / 85.5	31.3 / 31.3	<u>87.6 / 87.6</u>
	VecCISC (random)	83.7 / 82.9	69.9 / 66.5	82.6 / 82.0	28.1 / 27.2	87.3 / 86.8
	VecCISC + KMeans	84.6 / 84.3	83.0 / 82.9	85.5 / 85.5	32.2 / 31.7	87.7 / 87.2
	VecCISC + HAC	<u>84.3 / 84.3</u>	83.0 / 83.0	85.5 / 85.5	<u>31.5 / 31.5</u>	<u>87.6 / 87.6</u>
CommonsenseQA	SC Baseline	83.3 / 83.3	76.9 / 76.9	<u>82.5 / 82.5</u>	<u>67.8 / 67.8</u>	<u>86.0 / 86.0</u>
	CISC	<u>83.6 / 83.6</u>	<u>77.3 / 77.3</u>	83.0 / 83.0	68.0 / 68.0	86.2 / 86.2
	VecCISC (random)	83.0 / 82.2	56.8 / 54.7	79.4 / 78.4	50.9 / 49.4	84.9 / 84.4
	VecCISC + KMeans	83.7 / 83.7	77.4 / 77.4	83.0 / 82.8	68.0 / 68.0	86.2 / 86.2
	VecCISC + HAC	<u>83.6 / 83.6</u>	77.4 / 77.4	83.0 / 83.0	68.0 / 68.0	86.2 / 86.2
	ARC-Challenging	SC Baseline	95.7 / 95.7	91.0 / 91.0	<u>92.9 / 92.9</u>	71.3 / 71.3
CISC		<u>96.0 / 96.0</u>	91.1 / 91.1	93.2 / 93.2	71.3 / 71.3	96.3 / 96.3
VecCISC (random)		95.6 / 95.2	75.5 / 74.8	90.9 / 90.2	54.9 / 52.5	96.0 / 95.6
VecCISC + KMeans		96.1 / 96.0	91.1 / 91.1	93.2 / 93.2	71.3 / 71.3	96.3 / 96.3
VecCISC + HAC		96.1 / 96.1	91.1 / 91.1	93.2 / 93.2	71.3 / 71.3	96.3 / 96.3
MMLU-Pro		SC Baseline	64.6 / 64.6	56.0 / 56.0	59.3 / 59.3	27.8 / 27.8
	CISC	65.9 / <u>65.9</u>	57.6 / 57.6	60.2 / 60.2	28.2 / <u>28.2</u>	<u>71.3 / 71.3</u>
	VecCISC (random)	64.6 / 63.4	47.1 / 45.7	55.6 / 54.6	21.8 / 20.8	68.4 / 67.7
	VecCISC + KMeans	66.2 / 66.1	57.9 / 57.7	61.0 / 60.7	28.4 / 28.3	71.4 / 71.3
	VecCISC + HAC	<u>66.1 / 66.1</u>	<u>57.8 / 57.8</u>	<u>60.9 / 60.9</u>	<u>28.3 / 28.3</u>	<u>71.3 / 71.3</u>
	GPQA	SC Baseline	39.9 / 39.9	34.8 / 34.8	<u>37.4 / 37.4</u>	27.1 / 27.1
CISC		40.5 / 40.5	<u>35.0 / 35.0</u>	38.1 / 38.1	<u>28.4 / 28.4</u>	61.7 / 61.7
VecCISC (random)		<u>40.7 / 39.2</u>	33.9 / 30.5	38.1 / 35.4	28.8 / 27.1	58.6 / 57.3
VecCISC + KMeans		41.4 / 39.9	35.7 / 34.5	38.1 / 38.1	<u>28.4 / 25.7</u>	61.7 / 61.6
VecCISC + HAC		40.5 / 40.5	35.7 / 35.7	38.1 / 38.1	<u>28.4 / 28.4</u>	61.7 / 61.7

Table 5: Comparison between baselines and our methods across different LLMs and datasets. In each column, the best results are in **bold**, and the second-best results are underlined. VecCISC’s clustering approach maintains or improves upon CISC’s performance across models and datasets, with **VecCISC + HAC**’s average performance accounting for the most consistent method. Experiments on each ($dataset, model$) combination were performed 10 times for each method, and the best and average scores were recorded.

more specialized problems might benefit from applying custom embedding models, fine-tuned to capture key semantic representations. As our work demonstrates performance gains and budget improvements across a wide range of different tasks, we defer the application of fine-tuned embedding models for domain-specific problems to future research. In addition, the choice of hyperparameters K and T are important in ensuring that the pipeline’s accuracy on the datasets does not fall below that of standard Self-Consistency. While our hyperparameter search process is fundamentally data-driven, future research should investigate adaptive approaches for choosing K and T in the absence of analogous data *a priori*.

References

- Samir Abdaljalil, Hasan Kurban, Parichit Sharma, Erchin Serpedin, and Rachad Atat. 2025. [Sindex: Semantic inconsistency index for hallucination detection in llms](#).
- Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. 2023. [Let’s sample step by step: Adaptive-consistency for efficient reasoning and coding with llms](#).
- Meta AI. 2024. Llama 3.1 8b: Multilingual instruction-tuned large language model. <https://ai.meta.com/blog/meta-llama-3-1/>. Model card retrieved: Accessed October 20, 2025.
- Mistral AI. 2023. Mistral 7b v0.1: A 7-billion-parameter large language model. <https://huggingface.co/mistralai/Mistral-7B-v0.1>. Accessed: 2025-10-20.
- Jianhao Chen, Zishuo Xun, Bocheng Zhou, Han Qi, Hangfan Zhang, Qiaosheng Zhang, Yang Chen, Wei Hu, Yuzhong Qu, Wanli Ouyang, and Shuyue Hu. 2025. [Do we truly need so many samples? multi-llm repeated sampling efficiently scales test-time compute](#).
- Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. 2024. [Are more llm calls all you need? towards scaling laws of compound inference systems](#).
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. [Think you have solved question answering? try arc, the ai2 reasoning challenge](#). *arXiv:1803.05457v1*.
- Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. [The curious case of neural text degeneration](#).
- Chengsong Huang, Langlin Huang, Jixuan Leng, Jiacheng Liu, and Jiaxin Huang. 2025. [Efficient test-time scaling via self-calibration](#).
- Anil K. Jain and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall.
- Tim Knappe, Ryan Li, Ayush Chauhan, Kaylee Chhua, Kevin Zhu, and Sean O’Brien. 2024. [Semantic self-consistency: Enhancing language model reasoning via semantic weighting](#).
- Sungjae Lee, Hoyoung Kim, Jeongyeon Hwang, Eunhyeok Park, and Jungseul Ok. 2025. [Efficient latent semantic clustering for scaling test-time computation of llms](#).
- Huayang Li, Tian Lan, Zihao Fu, Deng Cai, Lemao Liu, Nigel Collier, Taro Watanabe, and Yixuan Su. 2023. [Repetition in repetition out: Towards understanding neural text degeneration from the data perspective](#).
- Moxin Li, Wenjie Wang, Fuli Feng, Fengbin Zhu, Qifan Wang, and Tat-Seng Chua. 2024a. [Think twice before trusting: Self-detection for large language models through comprehensive answer reflection](#).
- Moxin Li, Wenjie Wang, Fuli Feng, Fengbin Zhu, Qifan Wang, and Tat-Seng Chua. 2024b. [Think twice before trusting: Self-detection for large language models through comprehensive answer reflection](#).
- Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. 2024c. [Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning](#).
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.
- Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. 2025. [Can 1b llm surpass 405b llm? rethinking compute-optimal test-time scaling](#).
- Inc. Meta Platforms. 2024. Llama 3.3 70b versatile. <https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct>. Large language model.
- Daniel Müllner. 2011. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*.
- Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. [Routellm: Learning to route llms with preference data](#).
- OpenAI. 2024a. Gpt-4o mini: Advancing cost-efficient intelligence. <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>. Accessed: 2025-10-20.

- OpenAI. 2024b. `text-embedding-3-small`: A highly efficient embedding model. <https://platform.openai.com/docs/models/text-embedding-3-small>. Accessed: 2025-10-20.
- Shubham Parashar, Blake Olson, Sambhav Khurana, Eric Li, Hongyi Ling, James Caverlee, and Shuiwang Ji. 2025. [Inference-time computations for llm reasoning and planning: A benchmark and insights](#).
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, and 25 others. 2024. [Qwen2.5 technical report](#).
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. [Gpqa: A graduate-level google-proof q&a benchmark](#).
- Lior Rokach and Oded Maimon. 2005. *Clustering Methods*. Springer.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. [Scaling llm test-time compute optimally can be more effective than scaling model parameters](#).
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. [CommonsenseQA: A question answering challenge targeting commonsense knowledge](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.
- Amir Taubenfeld, Tom Sheffer, Eran Ofek, Amir Feder, Ariel Goldstein, Zorik Gekhman, and Gal Yona. 2025. [Confidence improves self-consistency in llms](#).
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. [Self-consistency improves chain of thought reasoning in language models](#).
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhui Chen. 2024. [Mmlu-pro: A more robust and challenging multi-task language understanding benchmark](#).
- Yutong Wang, Pengliang Ji, Chaoqun Yang, Kaixin Li, Ming Hu, Jiaoyang Li, and Guillaume Sartoretti. 2025. [Mcts-judge: Test-time scaling in llm-as-a-judge for code correctness evaluation](#).
- Junchi Yao, Shu Yang, Jianhua Xu, Lijie Hu, Mengdi Li, and Di Wang. 2025. [Understanding the repeat curse in large language models from a feature perspective](#).
- Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenye Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, Irwin King, Xue Liu, and Chen Ma. 2025a. [A survey on test-time scaling in large language models: What, how, where, and how well?](#)
- Yiqun Zhang, Hao Li, Chenxu Wang, Linyao Chen, Qiaosheng Zhang, Peng Ye, Shi Feng, Daling Wang, Zhen Wang, Xinrun Wang, Jia Xu, Lei Bai, Wanli Ouyang, and Shuyue Hu. 2025b. [The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants](#).

Prompt for Candidate Answer Sampling from the Base LLM (LLM_{gen})

Instructions

Given a multiple-choice question and set of answer options for the question, choose the correct answer from the list of options.

Question and Answer Options

{question}

{options}

Output

Produce the correct answer's ID in a JSON object and include your reasoning for why you chose that particular answer.

```
```json
{"answer":<PUT THE CORRECT ANSWER ID HERE>, "reasoning": <PUT YOUR REASONING HERE>}
```

Let's think step by step.

### Prompt for Confidence Scoring LLM ( $LLM_{critic}$ )

#### ## Instructions

You are a reviewer of answers given to multiple-choice questions. Given a multiple-choice question, a set of answer options for the question, a given answer, and the reasoning that is associated with the given answer, rate your confidence that the given answer is correct.

To rate your confidence, provide a value on a scale of 0 to 1. The more confident you are that the provided answer is correct for the question, the closer to (or equal) to 1 the confidence score should be.

#### ## Question and Answer Options

**{question}**

**{options}**

#### ## Given Answer and Provided Reasoning

Given answer: **{answer}**

Reasoning for the given answer: **{reasoning}**

#### ## Output

Produce the confidence score in a JSON object.

```
```json
{"confidence":<PUT THE CONFIDENCE SCORE HERE>}
```

Let's think step by step.

A Appendix

A.1 Model Hyperparameters

For all our experiments, we set $n = 20$ as our sampling budget. The temperature of LLM_{gen} was set to 0.7, while the temperature of LLM_{critic} was set to 0.3. We gave LLM_{gen} , the baseline question answering model, a higher temperature to encourage exploration and novelty in reasoning and answer generation, while our critic was given a lower temperature to ensure more deterministic behavior.

A.2 Cluster Hyperparameters and Implementation Details

For both KMeans and Hierarchical Agglomerative Clustering, we performed a grid search over the range of 1 to 20 (the maximum budget allowance) to choose K , the number of clusters to be formed within each answer reasoning trace group. The grid search was performed on a 20% holdout set from each ($dataset, model$) combination. We utilized the KMeans and HAC implementations from the Scikit-learn package (Pedregosa et al., 2011). For KMeans, only the $n_cluster$ parameter was altered. For HAC, we selected the ‘‘cosine’’ distance metric and the ‘‘average’’ linkage method. Average linkage is commonly found to produce more balanced and robust clusters compared to single and complete linkage. Single linkage measures the minimum pairwise distance between clusters, which makes it highly sensitive to noise and prone to ‘‘chaining’’ effects that merge dissimilar points through intermediate samples. Complete linkage, however, uses the maximum pairwise distance, often fragmenting elongated or irregular clusters. Average linkage mitigates both extremes by considering the mean pairwise distance between cluster members, leading to clusters that are less sensitive to outliers and better reflect the overall structure of the data (Müllner, 2011; Rokach and Maimon, 2005; Jain and Dubes, 1988). The hyperparameter heatmaps for T , K (VecCISC + KMeans), and K (VecCISC + HAC) can be found in Figures 3, 4, and 5, respectively.

A.3 Case Study 1: Clustering Traces With Similar Semantic Meaning

To demonstrate the application of clustering on LLM reasoning traces, we present a case study on a question from the MMLU-Pro dataset with traces sampled from GPT 4o-mini. The question is as

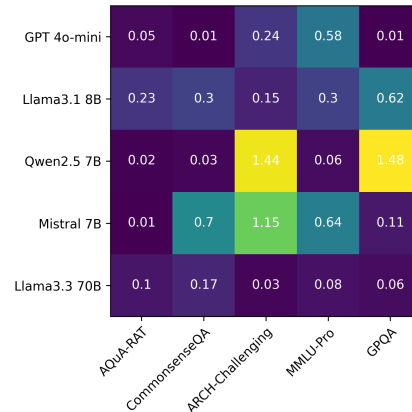


Figure 3: Heatmap of T temperature values. To find T , a grid search was performed across the range $[0, 5]$ for each dataset and model.

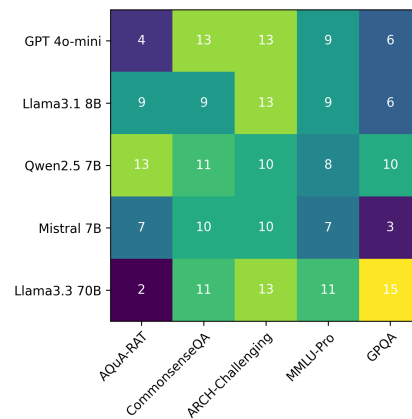


Figure 4: Heatmap of values for K used in **VecCISC + KMeans**. To find K , a grid search was performed across the range $[0, 20]$ for each dataset and model.

follows: ‘‘What sport is featured in the 1996 movie ‘Kingpin’?’’, with the given options ‘‘A) chess B) ice hockey C) baseball D) wrestling E) basketball F) bowling G) boxing H) golf I) tennis J) football’’. In keeping with our methodology, we grouped all reasoning traces that produce the correct answer ‘‘F’’ and applied HAC with a cluster number of 9. The trace clusters are presented below. As expected, traces with similar semantic details, phrasing, and logical reasoning steps are clustered together.

Cluster ID: 1

of Traces: 5

Trace 1: *The movie ‘Kingpin’ is a comedy film released in 1996 that revolves around the sport of bowling. It follows the story of a former bowling champion who tries to mentor a promising bowler.*

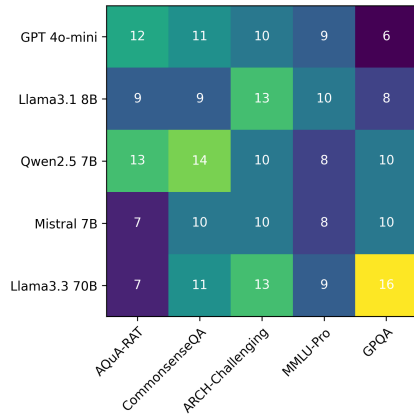


Figure 5: Heatmap of values for K used in **VecCISC + HAC**. To find K , a grid search was performed across the range $[0, 20]$ for each dataset and model.

From the given options, the only sport that matches the theme of the movie is: F) bowling Therefore, the correct answer is F, and the reasoning is based on the fact that the entire plot of 'Kingpin' is centered around bowling.

Trace 2: The movie 'Kingpin' is a comedy film released in 1996 that centers around the sport of bowling. The plot follows a washed-up professional bowler who mentors a promising young talent. Given the options provided, the correct sport featured in 'Kingpin' is bowling. Now, looking at the answer options, the only one that matches the sport in the movie is: F) bowling Thus, I will select this option.

Trace 3: The movie 'Kingpin' is a comedy that centers around the sport of bowling. The film follows a former bowling prodigy who attempts to coach a promising young bowler in order to win a bowling tournament. Given the options provided, the correct sport featured in 'Kingpin' is bowling. Therefore, the correct answer is F) bowling.

Trace 4: The movie 'Kingpin,' released in 1996, is primarily centered around the sport of bowling. The plot follows a former professional bowler who tries to mentor a promising bowling prodigy. Given the context of the film and its focus on bowling as the central sport, the correct answer is F) bowling. Thus, the answer ID for bowling is F.

Trace 5: The movie 'Kingpin,' which was released in 1996, is primarily centered around the sport of bowling. The plot revolves around a former bowling champion who mentors a promising young bowler, and much of the film's humor and storyline is tied to bowling competitions and the bowling culture. Given the options provided, the only sport

that directly relates to the movie is: F) bowling Therefore, the correct answer is F.

Cluster ID: 2

of Traces: 5

Trace 6: The 1996 movie 'Kingpin' is a comedy film that centers around the sport of bowling. The film features characters who are involved in the world of competitive bowling, highlighting the sport's culture and eccentricities. Now, looking at the answer options provided: - A) chess: Not applicable, as chess is not a sport featured in the movie. - B) ice hockey: Not applicable, as ice hockey is not related to the film. - C) baseball: Not applicable, as baseball is not featured in the movie. - D) wrestling: Not applicable, as wrestling is not the focus of the film. - E) basketball: Not applicable, as basketball is not part of the storyline. - F) bowling: This is the correct answer, as it is the main sport featured in 'Kingpin.' - G) boxing: Not applicable, as boxing is not part of the film. - H) golf: Not applicable, as golf is not related to the movie. - I) tennis: Not applicable, as tennis is not a focus of the film. - J) football: Not applicable, as football is not featured in the movie. Given this analysis, the correct answer is F) bowling.

Trace 7: The 1996 movie 'Kingpin' is a comedy that centers around bowling. The main characters are involved in the sport of bowling, which is a key element of the film's plot and humor. Now let's analyze the answer options: A) chess - This is not related to the movie. B) ice hockey - This sport is not featured in 'Kingpin.' C) baseball - This is not the focus of the movie. D) wrestling - This sport is not the main theme of 'Kingpin.' E) basketball - This sport does not play a role in the film. F) bowling - This is the correct answer, as 'Kingpin' revolves around bowling. G) boxing - This is not featured in the movie. H) golf - This sport is not related to the movie. I) tennis - This is not relevant to 'Kingpin.' J) football - This sport does not relate to the film. Given that the main focus of 'Kingpin' is bowling, the correct answer is F) bowling.

Trace 8: The question asks about the sport featured in the 1996 movie 'Kingpin'. To answer this, I recall that 'Kingpin' is a comedy film that revolves around the sport of bowling. The main characters are involved in bowling competitions, and the storyline heavily features bowling culture. Now, looking at the answer options provided: A) chess - Not related to the movie. B) ice hockey -

Not related to the movie. C) baseball - Not related to the movie. D) wrestling - Not related to the movie. E) basketball - Not related to the movie. F) bowling - This is the sport featured in 'Kingpin'. G) boxing - Not related to the movie. H) golf - Not related to the movie. I) tennis - Not related to the movie. J) football - Not related to the movie. Given that 'bowling' is explicitly mentioned as the sport in the movie, the correct answer is clearly option F.

Trace 9: The question asks about the sport featured in the 1996 movie 'Kingpin.' To answer this, I need to recall what 'Kingpin' is about. The film 'Kingpin' is primarily centered around the sport of bowling. It follows the story of a former bowling champion who mentors a promising young bowler. The comedic elements and plot revolve around bowling competitions and the characters' interactions within that context. Now, reviewing the answer options: A) chess - not related to the movie B) ice hockey - not related to the movie C) baseball - not related to the movie D) wrestling - not related to the movie E) basketball - not related to the movie F) bowling - directly related to the movie G) boxing - not related to the movie H) golf - not related to the movie I) tennis - not related to the movie J) football - not related to the movie From this analysis, the correct answer is F) bowling.

Trace 10: The movie 'Kingpin' is primarily centered around the sport of bowling. It features characters who are involved in the bowling scene, including competitions and personal rivalries. The film is well-known for its comedic portrayal of the world of bowling. Now, let's look at the answer options: A) chess - not related to the movie B) ice hockey - not related to the movie C) baseball - not related to the movie D) wrestling - not related to the movie E) basketball - not related to the movie F) bowling - this is the correct sport featured in the movie G) boxing - not related to the movie H) golf - not related to the movie I) tennis - not related to the movie J) football - not related to the movie Given this analysis, the correct answer is F) bowling.

Cluster ID: 3
of Traces: 2

Trace 11: The 1996 movie 'Kingpin' is a comedy film that centers around the sport of bowling. The storyline follows a former bowling champion who tries to coach a promising Amish bowler. Given the options presented: A) chess - not relevant to

the film B) ice hockey - not relevant to the film C) baseball - not relevant to the film D) wrestling - not relevant to the film E) basketball - not relevant to the film F) bowling - directly relevant to the film G) boxing - not relevant to the film H) golf - not relevant to the film I) tennis - not relevant to the film J) football - not relevant to the film The correct answer is F) bowling.

Trace 12: The 1996 movie 'Kingpin' is a comedy that revolves around the sport of bowling. The film follows a former bowling champion who discovers a promising bowling prodigy and attempts to mentor him. Now, looking at the options provided: - A) chess - B) ice hockey - C) baseball - D) wrestling - E) basketball - F) bowling - G) boxing - H) golf - I) tennis - J) football Among these options, the sport that is clearly related to the plot of 'Kingpin' is F) bowling.

Cluster ID: 4
of Traces: 2

Trace 13: The movie 'Kingpin' is a comedy that revolves around the sport of bowling. It features a former professional bowler and his journey to redemption through the sport, along with comedic antics. The other options, such as chess, ice hockey, baseball, wrestling, basketball, boxing, golf, tennis, and football, are not central to the plot of the movie. Based on this understanding, the correct answer is bowling.

Trace 14: The movie 'Kingpin,' released in 1996, is a comedy that revolves around the sport of bowling. It features characters who are bowlers and delves into the dynamics of competitive bowling. The other options listed, such as chess, ice hockey, baseball, wrestling, basketball, boxing, golf, tennis, and football, are not the focus of this film. Given this information, the correct answer is F) bowling.

Cluster ID: 5
of Traces: 1

Trace 15: The movie 'Kingpin,' released in 1996, is primarily about the sport of bowling. The plot follows a former bowling champion who takes a promising young bowler under his wing. Given this context, the correct answer from the provided options is F) bowling.

Cluster ID: 6

of Traces: 1

Trace 16: *The movie 'Kingpin,' released in 1996, revolves around the sport of bowling. The plot centers on a washed-up bowler who becomes a mentor to a young bowling prodigy, showcasing various aspects of the bowling culture. Now, let's analyze the options: A) chess - This is not related to the plot. B) ice hockey - Not relevant to 'Kingpin.' C) baseball - Not the sport featured in the movie. D) wrestling - While it is a sport, it's not the focus of 'Kingpin.' E) basketball - Not the sport in question. F) bowling - This is the correct answer, as it is the main sport featured in the movie. G) boxing - Not relevant to 'Kingpin.' H) golf - Not related to the movie's theme. I) tennis - Not part of the plot. J) football - Not the sport featured in the movie. Based on this analysis, the sport featured in 'Kingpin' is unequivocally bowling.*

Cluster ID: 7
of Traces: 2

Trace 17: *The 1996 movie 'Kingpin' is a comedy that centers around the sport of bowling. The film features characters participating in bowling tournaments and showcases various aspects of the sport. Given this information, the correct answer from the provided options is: F) bowling.*

Trace 18: *The 1996 movie 'Kingpin' is a comedy that revolves around bowling. The main characters participate in a bowling tournament, and the film is centered around the sport of bowling. Given the answer options, the only sport that relates to bowling is option F.*

Cluster ID: 8
of Traces: 1

Trace 19: *To determine the correct answer to the question about the sport featured in the 1996 movie 'Kingpin', we need to consider what the film is about. 1. The movie 'Kingpin' is a comedy directed by the Farrelly brothers, and it revolves around the sport of bowling. 2. The plot follows a former bowling champion who becomes a mentor to a promising young bowler. 3. The film features various bowling tournaments and characters who are involved in the bowling community. Given these points, the answer that corresponds to the sport featured in 'Kingpin' is clearly bowling. Now, looking at the answer options provided: A)*

chess B) ice hockey C) baseball D) wrestling E) basketball F) bowling G) boxing H) golf I) tennis J) football The correct option is F) bowling.

Cluster ID: 9
of Traces: 1

Trace 20: *The 1996 movie 'Kingpin' is a comedy that revolves around the sport of bowling. The film follows a washed-up professional bowler who becomes a mentor to a young bowling prodigy. Given the options provided, the correct answer is F) bowling.*

A.4 Case Study 2: Isolating Degenerate Traces

The presence of text degeneration in LLM reasoning traces is an extensively studied (Yao et al., 2025; Holtzman et al., 2019; Li et al., 2023) phenomenon and common occurrence when sampling from LLMs, especially at higher temperatures. Degenerate traces exhibit repeated blocks of frequently nonsensical and meaningless content, often exceeding the model's output window. As such, these traces are much more expensive for a critic to evaluate. Our proposed methodology isolates such traces by selecting a candidate representative from each cluster, defined as the trace closest to the centroid of its sibling trace embeddings in the cluster. To illustrate this in practice, we present a case study on a question from the ARC-Challenging dataset, with answers sampled from Llama 3.3 70B Versatile. The question is as follows: "Fossil bones and teeth of dinosaurs have been researched for the last century. Recent discoveries of fossilized dinosaurs have also revealed details of soft tissues, such as skin. Which is best for a scientist to do when reporting research on dinosaurs now?" Below is a cluster of three traces sampled from the LLM for this question. The last trace contains text degeneration (highlighted in red), as the content switches from a question-relevant explanation to pseudocode gibberish. In particular, reasoning trace #1 is identified by our pipeline as the candidate representative, thus avoiding the erroneous trace #3.

Cluster ID: 1
of Traces: 3

Trace 1: *To answer this question, let's consider the implications of each option in the context of*

scientific research on dinosaurs.

A) *Exclude research on teeth or bones:* This option suggests ignoring a significant portion of existing data on dinosaurs. Teeth and bones have been the primary sources of information about dinosaurs for a long time and are crucial for understanding their evolution, diet, and behavior. Excluding this information would not be beneficial for comprehensive research.

B) *Predict what the next discovery will be:* While predictions can be interesting and might guide future research directions, they are speculative and do not contribute directly to the reporting of current research findings. Predictions are not a substitute for analyzing and reporting actual data.

C) *Analyze new data as it becomes available:* This option is about incorporating the latest findings into the research. As new discoveries are made, such as details of soft tissues like skin, analyzing this new data alongside existing knowledge (like that from fossil bones and teeth) can provide a more complete understanding of dinosaurs. This approach allows for the refinement of theories and a more accurate representation of dinosaur biology and behavior.

D) *Delete earlier reports that were missing the new findings:* Deleting earlier reports would result in the loss of historical data and the context of how our understanding of dinosaurs has evolved over time. Earlier reports, even if they lack the new findings, are still valuable for understanding the progression of knowledge and can serve as a baseline to show how new discoveries have changed our understanding.

Considering these points, the most appropriate action for a scientist reporting research on dinosaurs now would be to analyze new data as it becomes available. This ensures that all relevant information, both old and new, is considered to provide the most accurate and comprehensive understanding of dinosaurs.

Trace 2: To answer this question, let's consider the implications of each option in the context of scientific research on dinosaurs.

A) *Exclude research on teeth or bones:* This option suggests ignoring a significant portion of the data that has been collected over the last century. Teeth and bones have been the primary sources of information about dinosaurs for a long time, and excluding them would not be beneficial for a comprehensive understanding of these creatures.

B) *Predict what the next discovery will be:* While

predicting future discoveries can be an interesting speculative exercise, it does not contribute directly to the reporting of current research findings. Scientific reports should be based on evidence and data that have already been collected and analyzed.

C) *Analyze new data as it becomes available:* This option is about incorporating the latest findings into the ongoing research. As new discoveries are made, such as details of soft tissues like skin, analyzing this new data allows scientists to refine their understanding of dinosaurs. This approach ensures that research reports are as up-to-date and accurate as possible.

D) *Delete earlier reports that were missing the new findings:* Deleting earlier reports because they lack new findings would result in the loss of historical research data. Earlier reports still contain valuable information about dinosaurs, even if they do not include the latest discoveries. It's more productive to build upon previous research rather than discard it.

Given these considerations, the best approach for a scientist reporting research on dinosaurs is to analyze new data as it becomes available. This ensures that their reports are current, comprehensive, and based on the latest evidence.

Trace 3: To answer this question, let's consider the implications of each option in the context of scientific research on dinosaurs, particularly focusing on the integration of new discoveries with existing knowledge.

A) *Exclude research on teeth or bones:* This option suggests ignoring a significant portion of dinosaur research that has been foundational to our understanding of these creatures. Teeth and bones have been crucial in identifying species, understanding evolutionary relationships, and reconstructing dinosaur biology. Excluding this research would be counterproductive and not contribute to a comprehensive understanding of dinosaurs.

B) *Predict what the next discovery will be:* While predictions can be interesting and might guide future research directions, they are speculative and do not directly contribute to the reporting of current research findings. Predictions might not always be accurate and could distract from the solid, evidence-based reporting of scientific discoveries.

C) *Analyze new data as it becomes available:* This option emphasizes the importance of incorporating new findings into our existing knowledge base. As new details about dinosaurs are discovered, such as those con-

cerning soft tissues, analyzing dataInjected
contaminants MAV(dateTime contaminants
itInjected_both—from: by is a is series series
should are(dateTime continually continu-
ally MAV_both(dateTime(dateTime(dateTime
Basel To(dateTime the the PSI exposi-
tion.visitInsn.visitInsn contaminants exposi-
tion from PSI Basel ToastrBritainBuilder-
Factory.visitInsn MAVRODUCTION field
MAVroscope MAV MAV/slider Succ(Size Toastr
Succ(dateTimeInjected contaminants(dateTime
—— RODUCTIONRODUCTIONBritain-
BuilderFactory Toastr PSI_both exposition-
roscopeRODUCTION Basel_bothroscope
PSI PSI—from(SizeBuilderFactory Succ
MAVroscope—fromInjected—from(dateTime
—— BritainBritainexternalAction-
Code(SizeroscopeBuilderFactory Succ Succ
exposition(dateTimeroscope MAV contami-
nants—from Succ SuccBuilderFactory Suc-
cexternalActionCode(Size expositionroscope
Succroscope MAVRODUCTIONexternalAction-
Code SuccBuilderFactoryInjected contaminants
Succ PSI(dateTimeBuilderFactory PSI Toastr con-
taminants PSIIInjected —— .visitInsn(dateTime
Size—from.visitInsnexternalActionCode": " Toastr
PSI contaminantexternalActionCode exposition
PSI PSIBuilderFactory MAV Succ PSImpilerizr-
roscopecroscope PSI Succ SuccInjectedInjected
PSIRODUCTIONInjectedRODUCTIONBuilder-
Factory PSIexternalActionCode—from Succ ——
(Size PSIRODUCTIONBritain(dateTime PSI
Britain PSI—fromBuilderFactory exposition con-
taminantexternalActionCode ToastrBuilderFac-
tory Basel BaseexternalActionCode MAVBuilder-
Factory PSI exposition SuccRODUCTION.visitInsn
Succ Succ Succ Succ Succ/slider THAT toLeidleer
'IRODUCTION ToastrBritain SuccPsi PSI [
categories contaminants_both.visitInsnroscope
contaminants.visitInsn Succ Basil PSI Psi Basil]
PsiRODUCTION MAVexternalActionCodeInject-
edBritain apologize_both SuccexternalActionCode
PSI Toastr Basel.visitInsn Toastr MAV Britain
expositionBuilderFactory(dateTime(Size Suc-
croscope —— PSI_both exposition Toastr
Toastr PSI Succ —— expositionroscopeBri-
tain(dateTimeroscopeInjectedBuilderFactory(dateTime
Toastr PSI Succ/slider ToastreexternalAction-
Code(dateTime Succ forgotten Britain Succ her
winds ...