Concept Attractors in LLMs and their Applications

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large language models (LLMs) often map semantically related prompts to similar internal representations at specific layers, even when their surface forms differ widely. We show that this behavior can be explained through Iterated Function Systems (IFS), where layers act as contractive mappings toward concept-specific Attractors. We leverage this insight and develop simple, training-free methods that operate directly on these Attractors to solve a wide range of practical tasks, including **language translation**, **hallucination reduction**, **guardrailing**, and **synthetic data generation**. Despite their simplicity, these Attractor-based interventions match or exceed specialized baselines, offering an efficient alternative to heavy fine-tuning, generalizable in scenarios where baselines underperform.

1 Introduction

Consider three distinct concepts: the Lord of the Rings universe, the Python programming language, and 19th-century romantic literature. When prompts from these concepts are given to a large language model (LLM) such as Llama 3.1 [2], we see an interesting phenomenon. For each concept, despite lexical variations among its prompts, their intermediate representations appear to collapse to distinct regions at *specific layers* – at which layer this happens varies based on the concept. For instance, prompts such as "Who is Gandalf the Grey?" and "What is the significance of Mount Doom?" share minimal similarity on the surface, yet their representations converge to nearly identical locations at layer 24. We see a similar behavior for Python-related queries such as "Help me implement a binary search tree in Python" versus "How can I find the longest non-repeating substring in Python?" and for prompts for the same genre in literature: "Discuss themes in Pride and Prejudice" and "Any easy way to recognize Byron's poetry?". Such a semantic collapse has been variously reported in some recent results. For instance, [3] notes that transformer models develop a structured latent representations that encode *belief states*. Separately, [4] suggests that due to the internal dynamics of the model, representations converge to "stable" configurations. From a more practical perspective, [5, 6, 7] showed that transformers and LLMs shape their latent space according to the underlying task. These findings, while restricted to smaller model and/or for specific contexts, cumulatively support the general idea of representation collapse.

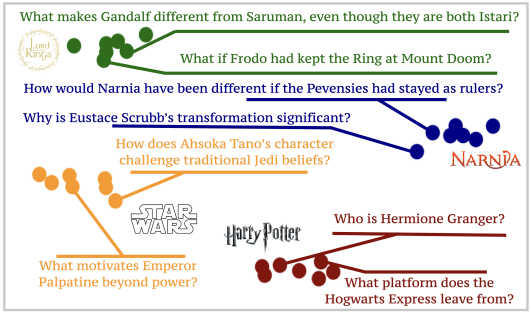


Figure 1: A t-sne[1] plot of the latent representations of Llama3.1-8B for $7 \times 4 = 28$ different prompts, seven each, for the Lord of the Rings universe, Narnia, Star Wars, and Harry Potter. Although the prompts explore different aspects of the universes and share almost no common keywords, we observe a clear clustering based on the different worlds.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

A natural question is whether this concept-specific collapse is implied as a property of some underlying dynamical system already studied in the literature, and if so, what guidance can these existing results provide? Specifically, can we obtain strategies for important downstream use-cases? If $p_1, \cdots, p_n$ are a set of prompts related to a specific concept $\mathcal{C}$, we conjecture that the layers of our model may be acting like a dynamical system that maps semantically related inputs to proximal regions, regardless of their form at the "surface". In other words, the full sequence of layers (leading up to where the representations collapse), if viewed as a unit, implements an iterative (contractive) mapping process to an *Attractor set*, one for each concept. We will see shortly that – to the extent that our hypothesis holds – how existing results are consistent with this view of the collapse phenomena.

Contributions. We show that viewing the LLMs through the lens of Iterated Function Systems [8, 9] offers a meaningful (or at worst, plausible) explanation for both the layer-specific concept clustering and the subsequent generative process. The main practical benefit is that for a wide-variety of downstream tasks, which are often handled piecemeal in the literature, we can obtain a generic scheme that operates under the assumption that operating with the Attractors alone is *sufficient*. We demonstrate that careful interventions on Attractors can provide us lightweight, *training-free* solutions to a wide array of problems, from **programming language translation** and **guardrailing**, to **hallucination reduction** and **synthetic data generation**. Despite the simplicity as well as limited data/compute needs, these solutions turn out to be comparable to existing specialized approaches.

2 Iterated Function Systems and LLMs

There is mounting evidence that large language models (LLMs) possess emergent capabilities beyond simple rote memorization and statistical pattern matching [10]. Among the many phenomena observed in these models – from in-context learning [11] to compositional reasoning [12, 13] – we focus on a particular representation-convergence property. Our scope is specifically the collapse phenomena at specific intermediate layers. To understand this behavior through the lens of dynamical systems, we hypothesize that LLMs implicitly implement a collection of Iterated Function Systems (IFS) during forward propagation through the layers (Fig. 2).

2.1 LLMs implement Iterated Function Systems?

Empirically, we see that for prompts p_i, p_j in each concept $\mathcal{C}$, there exists a layer l where:

$$\lim_{l \to l_{\mathcal{C}}} \frac{1}{n^2} \sum_{i,j=1}^{n} |h_l(p_i) - h_l(p_j)| \ll \frac{1}{n^2} \sum_{i,j=1}^{n} |h_0(p_i) - h_0(p_j)| \tag{1}$$

with h_l denoting the implicit transformation by the LLM up to layer l. This "squashing" of inter-prompt distances suggests a contractive mapping process is taking place through the layers. Our hypothesis is that this can be understood via the framework of Iterated Function Systems (IFS) [8, 9].

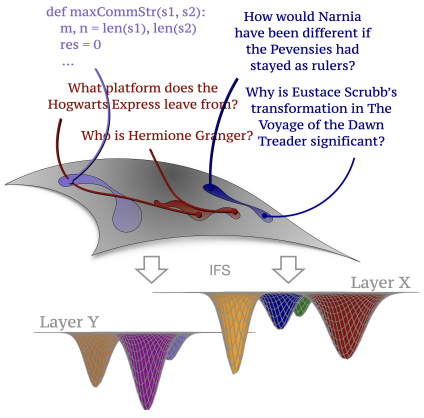


Figure 2: An LLM can be viewed as an IFS that transforms the non-linear manifold of texts into a well-behaving collection of Attractors.

An IFS is defined as a finite set of contractive mappings on a complete metric space. The collective action of these mappings, defined by the Hutchinson operator [9] is:

$$\mathcal{F}(\mathbf{S}) = \bigcup_{i=1}^{N} f_i(\mathbf{S}) \tag{2}$$

and induces a compact invariant set i.e., $\mathcal{F}(\mathbf{S}^*) = \mathbf{S}^*$, which is called the Attractor of the IFS. More generally, for any initial non-empty compact set $\mathbf{S}_0 \in \mathbb{X}$, the sequence $\{\mathbf{S}_0, \mathbf{S}_1 \coloneqq \mathcal{F}(\mathbf{S}_0), \mathbf{S}_2 \coloneqq \mathcal{F}(\mathbf{S}_1), \cdots\}$ converges to $\mathbf{S}^*$ in the Haussdorf metric. More generally, an Attractor in a dynamical system is a closed invariant set toward which trajectories from a wide class of initial conditions evolve asymptotically within its basin of attraction, and may take the form of fixed points, periodic orbits, tori, or other Attractors characterized by sensitive dependence on initial conditions [8].

Dynamical systems often exhibit Attractors—sets toward which trajectories converge. Simple systems satisfying Banach's fixed-point conditions [14] converge to a single point, while others yield more complex structures like limit cycles or strange Attractors [15]. We hypothesize that the iterative application of layer transformations in an LLM induces concept-specific invariant sets –semantic Attractors ($\mathbf{A}_l^{\mathcal{C}}$) for each concept $\mathcal{C}$– within the latent space at layer l. These compact regions characterize specific concepts, with convergence potentially occurring at different depths depending on the concept.

Once a sequence's representation enters $\mathbf{A}_l^{\mathcal{C}}$, it is further processed by the remaining layers and output matrix W_{out} to yield a token distribution. Each Attractor may have an invariant measure $\mu_l^{\mathcal{C}}$, describing the distribution of states within it under stochastic dynamics (e.g., varied inputs aligned with concept $\mathcal{C}$). While $\mu_l^{\mathcal{C}}$ is useful for tasks like *synthetic data generation*, it does not directly define next-token probabilities in autoregressive inference, which depend on the specific input-driven state.

The attractors, $\mathbf{A}_l^{\mathcal{C}}$, are linked to the LLM's operational prefill and decode stages. During prefill, the LLM's composed layer transformations guide initial representations of an input prompt, $h_0(p)$, towards $\mathbf{A}_l^{\mathcal{C}}$, with the representation $h_l(p)$ landing within this attractor to give the initial semantic context. Then, during decode, each incremental update to the context (by newly generated tokens) is processed by these same underlying layer dynamics. For coherent generation aligned with concept $\mathcal{C}$, the evolving sequence representation at layer l is continually guided towards or kept within the basin of attraction of $\mathbf{A}_l^{\mathcal{C}}$. Thus, $\mathbf{A}_l^{\mathcal{C}}$ acts like a stabilizing latent structure.

Collage theorem. Our operational model takes the transformation performed by the LLM for a concept and approximates it by repeatedly iterating a single affine contractive map [16], $\phi_{\text{eff}} = M_{\text{eff}}V + t_{\text{eff}}$ (with V as a placeholder hidden representation), suggesting that the overall transformation, for a specific concept, can be roughly approximated by an iterated affine dynamics. We want to estimate the parameters (i.e., the matrix M_{eff} and vector t_{eff}) and the number of iterations `iter`, that best reproduce the observed mapping (Figure 2). This is achieved by minimizing the discrepancy between the LLM's observed states at the Attractor layer and the states predicted by iterating ϕ_{eff} from the initial prompt representations:

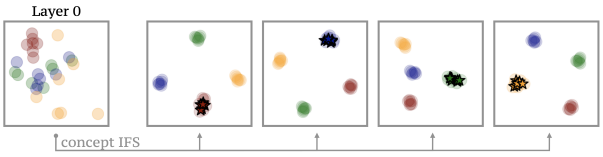


Figure 3: 4 different concepts in layer 0 (before any application of the underlying IFS, and one of the contractions of the underlying IFS we recover by solving the inverse problem for each concept separately. The circles correspond to the true vectors as obtained from the LLM in layer 24 and the stars correspond to the application of the contractions to the points in layer 0.

$$\min_{M_{\text{eff}},t_{\text{eff}},\texttt{iter}} \sum_{j=1}^{N} \mathcal{D}\left(h_l(p_j), \phi_{\text{eff}}^{\texttt{iter}}(h_0(p_j))\right) \tag{3}$$

subject to M_{eff} being contractive (e.g., its operator norm $|M_{\text{eff}}|_{op} < 1$). We apply this `iter` times, and $\mathcal{D}$ is a suitable distance metric. This single map ϕ_{eff} defines a simple Iterated Function System (IFS). The unique Attractor of this 1-map IFS is its fixed point, V^* to which all trajectories $\phi_{\text{eff}}^k(V)$ (for any initial V) converge as k grows. The observed empirical set $\mathbf{A}^{\mathcal{C}}$ is then interpreted as the collection of states reached after `iter` applications of ϕ_{eff} starting from the initial set S_0. If, as empirical evidence for many concepts suggests, this 1-map model provides a good first-order approximation, then $\mathbf{A}^{\mathcal{C}}$ would be expected to lie in the vicinity of V^*. The Collage Theorem [8] states that if $\mathbf{A}^{\mathcal{C}}$ is indeed close to the true Attractor V^* of our fitted ϕ_{eff}, then $\mathbf{A}^{\mathcal{C}}$ should be well "collaged" by ϕ_{eff} itself; i.e., $d\big(\mathbf{A}^{\mathcal{C}}, \phi_{\text{eff}}(\mathbf{A}^{\mathcal{C}})\big)$ should be small. While the iterated single affine map is simple, for concepts whose empirical Attractors $\mathbf{A}^{\mathcal{C}}$ exhibit more complex geometries (e.g., disjoint sets or intricate fractal structures not well approximated by convergence to a single point), a richer effective IFS comprising multiple affine maps might be necessary. This would involve finding ϕ's and an iteration count $\texttt{iter}'$ that minimize $d\left(\mathbf{A}^{\mathcal{C}}, \mathcal{F}^{\texttt{iter}'}(S_0)\right)$, where $\mathcal{F}$ is the Hutchinson operator for the candidate set of ϕ's. Alternatively, one could model the geometry of $\mathbf{A}^{\mathcal{C}}$ directly by finding an IFS whose intrinsic Attractor matches $\mathbf{A}^{\mathcal{C}}$, by minimizing the collage error. These approaches are more involved but grounded in IFS theory.

Does this perspective add to existing results? Several recent results have indirectly hinted at the IFS-like nature of the LLMs, and more generally transformers, for specific tasks, datasets, and

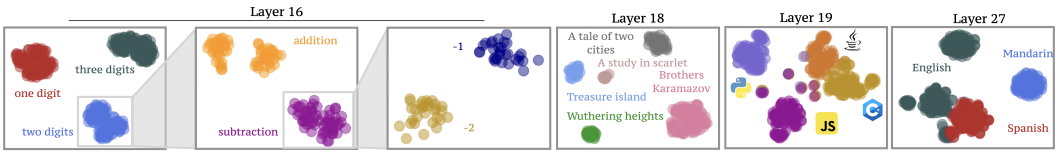


Figure 4: Attractors in Llama3.1-8B [2]. From the fractal structure of the task vectors in layer 16, to literature-based Attractors in layer 18 and programming-based in layer 19, the treatment of an LLM as an IFS allows us to recover (and use) them in multiple applications.

architectures. [4] describes how the intermediate layers of an LLM converge to different "Attractor" points/vectors as the context window of the LLM increases. The result in [17] examines the Attractors formed in the output layer of an LLM, discovering that paraphrasing results in 2-period cycles. The authors in [3] present evidence that transformers develop internal representations corresponding to "belief states" over hidden variables in the data-generating process. This phenomenon mirrors the behavior of an IFS, belief states in [3] can be viewed as specific points within concept Attractors that encode probabilistic information about possible continuations. Notice that the fractal structures reported in [3] arises naturally from known properties of IFS: systems whose repeated application to an initial set converges to a unique invariant set with so-called *self-similar* properties.

2.2 A preliminary investigation of Attractors

Before evaluating their practical utility, we first examine the nature of Attractors and their underlying IFS across various concepts and datasets as a sanity check.

Table 1: Top induced tokens of Attractors.

Concept	Tokens
Harry Potter	Harry, wizard, Hogwarts, magical, Voldermort, London, British, £
Lord of the Rings	Lord, Tolkien, Middle, Auckland, NZ
Narnia	Kingdom, Tolkien, British, Oxford, Aslan
Star Wars	Imperial, Star, galaxy, Galactic, Jedi, Empire, Skywalker, Force, powerful

Induced tokens. To understand what the Attractors represent, we average the vectors for each of the four fictional worlds from Fig. 1 to approximate their Attractor points, then project them to vocabulary space via the LLM's final linear layer. The top induced tokens (table 1) support our hypothesis, revealing meaningful associations—including tokens not present in the original texts, such as the pound symbol (£), filming locations (Auckland, NZ), or author connections (C.S. Lewis and J.R.R. Tolkien). This suggests the Attractors capture the underlying "essence" of each world, beyond surface-level content.

Different concepts, different layers. While for functional worlds, as in Figure 1, we see that the LLM forms clear Attractors in layer 24, this is not the case for all families of concepts, and not discussed in many existing results. We will see later that different families of concepts form Attractors in different layers. For example, we observe the same behavior in layer 19 for programming languages, in layer 27 for natural languages, and in layer 18 for literature books (Figure 4).

Same concept, multiple Attractors. Previously, we modeled each concept as a single Attractor (or Concept Vector) in the LLM's latent space. However, some concepts may decompose into multiple sub-concepts. For instance, English forms two distinct Attractors when combining datasets with different semantic styles (`https://www.manythings.org/anki/spa-eng.zip`, `https://huggingface.co/datasets/swaption2009/20k-en-zh-translation-pinyin-hsk`; see fig. 4). This fragmentation is even clearer in layer 16, where tasks produce multiple Attractors based on the number of digits per example.

A fractal-like structure in the Attractors. In fig. 4 (left), replicating the setup from [5], we observe a fractal-like structure in the ttractors derived from simple arithmetic tasks (e.g., adding 1, subtracting 2). At a high level, Attractors cluster by the number of digits in the examples. Zooming in, subclusters emerge based on task type (addition vs. subtraction), and further divisions align with specific values being added or subtracted. This hierarchical structure aligns with theoretical findings in [3], suggesting a fractal organization of Attractors in this setting.

LLMs and World Models. There is much discussion related to whether LLMs operate with an explicit, internal world model [18]. Based on the empirical analysis described so far, we find that there is at least partial evidence to support the idea that the models indeed harbor a *fuzzy* understanding

of the world, which is better expressed partially across many of these intermediate layers. In the subsequent section, we will focus on how we can better exploit this fuzzy world model of the LLMs and propose practical, training free solutions to a number of use cases.

3 Attractor for concept detection

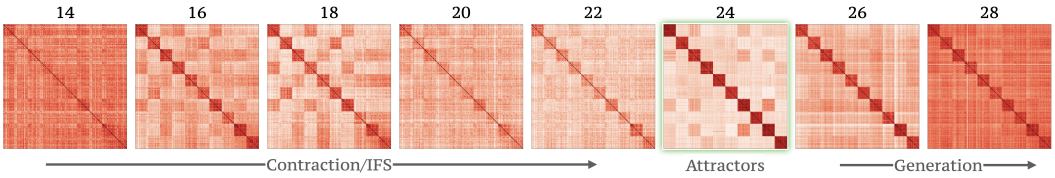
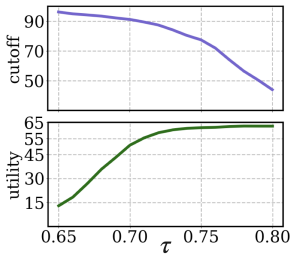


Figure 5: Cosine similarity between all prompts' from TOFU forget05 [19]. The first 20 rows/columns of each heatmap correspond to questions about the first author, the second 20 about the second author, and so on. The forming of author-based Attractors is apparent and it becomes clearer in layer 24.

Machine unlearning is a active research area, stemming from computer vision [20] where many widely used datasets include images of individuals who did not consent to their use. The training datasets of contemporary LLMs are also prompting concern about compliance with the Right to Be Forgotten [21] and similar regulations. Due to the size of these models, retraining or fine-tuning (e.g., [22, 23, 24, 25]) is often too costly. Moreover, since removal requests are continuous, efficient online unlearning is desirable. To evaluate unlearning in LLMs, Maini et al. [19] proposed the TOFU benchmark, where models must forget certain fictional authors while retaining performance on others and unrelated tasks.

Existing solutions. LLM unlearning methods fall into two main categories: (1) weight reversion and (2) guardrailing. *Weight reversion* seeks new parameters θ' close to those of a model trained without the forget set, θ^*. Early work [26] proposed lightweight fine-tuning to forget specific content (e.g., Harry Potter), but it does not scale to frequent or multi-instance requests. Recent PEFT-based methods [27, 28] improve efficiency but still require retraining and access to retention data, making them impractical for continuous unlearning. *Guardrailing* avoids changing model weights by intervening at input/output levels. While widely used, such techniques are typically shallow and vulnerable to jailbreaking [29, 30]. Hybrid approaches like Preference Optimization [19] use gradient ascent and placeholder outputs but still involve full model fine-tuning and retention data. Other methods (e.g., [31]) inject noise using concept classifiers, offering improved efficiency but still requiring training and retention data for each concept.

A training-free approach. We propose a train-free concept guardrailing method for LLMs that requires only data from the concept to be removed –no retention data needed– making it both computationally and data efficient. As shown in fig. 5, certain concepts (e.g., TOFU authors) form clear attractors in intermediate layer 24. We estimate each attractor by averaging hidden activations across the concept's samples. At inference, we compute the cosine similarity between the output's



Method	forget01		forget05		forget10		Train Free	No ret. data
	Utility ↑	Rouge ↓	Utility ↑	Rouge ↓	Utility ↑	Rouge ↓		
Original	62.67	97.67	62.67	97.67	62.67	97.67	-	-
Grad Asc	60.24	43.61	00.00	00.09	00.00	00.00	✗	✓
Grad Diff	60.59	44.80	32.44	01.85	58.23	00.32	✗	✗
Pref Opt	62.36	31.31	47.85	03.27	53.95	06.02	✗	✗
NPO	45.32	24.27	17.14	19.68	17.01	20.10	✗	✓
NPO-RT	48.96	26.55	54.14	28.93	49.97	23.80	✗	✗
ECO	62.57	03.32	62.57	07.62	62.35	06.94	✗	✗
Ours	62.67	00.48	61.20	10.33	61.34	19.54	✓	✓

Figure 6: (left) Model utility and cutoff percentage as a function of the threshold τ for TOFU forget10 [19]. Model utility determines the impact that the guardrailing has on the general answering abilities of the LLM while cutoff represents the percentage of questions regarding the forget set that are detected and guardrailed. (right) Model Utility and Forget Rouge of our train-free method compared to the typical (e.g., Gradient Ascent) and most recent trainable approaches (e.g., NPO [32] and ECO [33]). Although our approach is the only train-free approach and it requires no retention data, it is better than most baselines, offering also a greater control over the tradeoff of Model Utility vs Cutoff/Rouge, with the introduction of τ.

attractor and the stored one; if it exceeds a threshold τ, the response is blocked and replaced with a fixed message (e.g., "I cannot provide information about author X due to removal request <id>"). This requires only a single forward pass and no training.

Evaluation. Figure 6 (left) shows the cutoff percentage and the model's utility for different values of τ and for all 3 versions of the TOFU benchmark [19]. We can observe that even for the hardest version (forget10), the model's utility remains high while we enjoy a cutoff percentage of more than 90%. For specifically chosen values of τ, we show in Figure 6 (right) that our train-free approach is competitive with many heavier, trainable solutions. At the same time, the use of τ allows a finer control over the tradeoff of forgetting versus model utility.

4 Attractors for traversals

Treating the LLM as an IFS, and more generally a dynamical system, allows us to intervene on its trajectory and guide it towards specific Attractors. From a dynamical system perspective, if we assume that the LLM can be characterized from a function f such that $dx/dt = f(x)$, then, given a target Attractor y, we can modify the system as $dx/dt = f(x) + \lambda(y - x)$ and steer it towards another Attractor y, with λ being influenced by the underlying dynamics of the system (robustness to perturbations, distance of Attractors, etc.).

Such an approach, called *steering*, has been variously studied. We know that carefully chosen vectors can steer a model's behavior so that its output is less toxic, more poetic, etc. [34, 35, 36], essentially steering the model internally to different Attractors. However, many of these approaches require training the model itself or auxiliary smaller networks (e.g., [36, 37, 38]), while other works require carefully chosen data that satisfy some, more or less restrictive, assumptions (e.g., [35, 39, 40]).

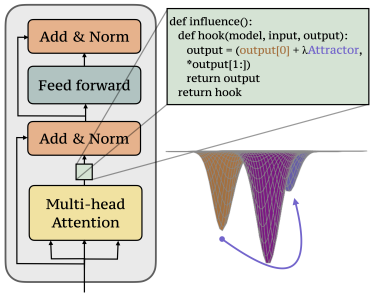


Figure 7: Influencing the dynamics of the LLM by adding the target Attractor. The only modification needed is the introduction of a forward hook on the appropriate layer.

Unlike methods requiring extensive retraining or retention data, we show that simply adding or subtracting Attractors at selected intermediate layers can influence LLM behavior across tasks –from detoxification to code translation– without these constraints. Surprisingly, in practice, the *before* Attractor is unnecessary, removing the need for retention data entirely. Despite requiring only a single forward pass over target data and no training, our approach matches the performance of more resource-intensive methods.

4.1 Drifting away from the toxicity Attractor

Method	Toxicity (↓)	Rouge (↑)	Train free	No ret. data
Original dataset	84.58	-	-	-
Llama2-7B	71.60	73.15	-	-
ICL	66.81	74.19	✓	✗
LoRA	49.94	57.32	✗	✗
ICV	39.54	65.97	✓	✗
Ours	**37.66**	61.27	✓	✓

ok - i can second this , that nicaraguan coffee is f*cking amazing.	this sh*t is why i moved to cali and refuse to look back.	lol sh*t i need change.	for example i don 't give a sh*t about your opinions , yet you still comment.
I can confirm that the coffee from Nicaragua is excellent.	This is why I moved to California and will not look back.	I need to change my life	I don't care about your opinions, but you still commented.

Figure 8: (left) Toxicity score and Rouge on ParaDetox. Although our lightweight approach requires no training or even retention data, it is reducing significantly the toxicity while maintaining the textual quality. (right) Toxic examples and the modified passages according to our method.

Multiple works have shown that careful manipulation of the activations across the LLM's layers allows us to control its behavior, and a common application is toxicity reduction. We note that these ideas impose one or more restrictive requirements on the data format, such as the need for retention data, or even the existence of paired data [35, 34]. Here, we check whether the estimation of the toxicity Attractor alone allows us steer the generation away from it and thereby, reducing the toxicity content of the LLM's output. No additional assumptions on the data are needed. Using the ParaDetox [41] dataset, we obtain a single vector estimate of the toxicity Attractor on layer 16 and, then, during generation, we subtract this value from each token's activation on layer 16, essentially discouraging the generation to converge to the toxicity Attractor. Although we only require the

toxicity Attractor/vector, our targeted approach performs better than many of the existing (but more restrictive) solutions.

Evaluation. In Figure 8, we show that our approach, without any need for training/retention data, performs similar as ICV [35] which needs a PCA projection of the differences between paired samples. We also appear to perform better than LoRA fine-tuning or the more lightweight In-Context Learning [11]. To assess both the reduction in toxicity as well as any potential drop in the quality of the generated text, we report both Toxicity [42], as well as the Rouge score [43]. Our approach is one of the few training free methods and the only one that requires no retention data. We find that relaxing these requirements does not lead to a performance drop, instead a performance gain. Finally, we should note that there are practical benefits of our lightweight approach.

4.2 Switching language Attractor on the fly

Py → JS

Method	x/5 Score (↑)	Train free	No ret. data
ICV	1.01 ± 0.10	✓	✗
LR	1.22 ± 0.56	✗	✗
DM	2.61 ± 1.24	✓	✗
Logistic	3.83 ± 1.28	✗	✗
RFM	4.34 ± 1.09	✗	✗
Ours	$\mathbf{4.56 \pm 0.69}$	✓	✓

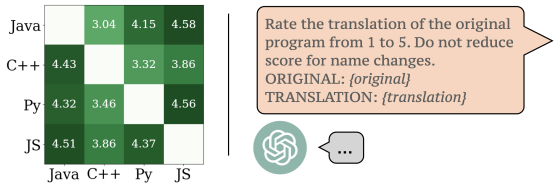


Figure 9: (left) LLM as a transpiler. For all pairs of the four considered languages, switching the Attractor to the target language can successfully make the LLM act as a transpiler without any specific such instructions or retention data. (right) Using o4 to judge the quality of the generated translations.

LLMs are extremely capable at code comprehension and composition [44, 45, 46]. Other than use as a code-generation assistant, an important use case is as a transpiler, especially for programming languages with limited support. Typically, the approach involves a data-intense stage of fine-tuning on code-specific data (e.g., [47, 48]). Some recent works have evaluated the limits of zero/few-shot transpiling in LLMs [49, 36].

As we showed in Figure 4, programming languages form Attractors on layer 19 of Llama3.1-8B. Here, we examine whether we can use these Attractors and use the LLM as a transpiler. Given only code block in an input language and without any specialized instructions, can we translate it to another, target language? Using 100 solutions of LeetCode problems in Python, Java, C++, and Javascript, we obtain an estimate of the corresponding Attractors in layer 19. Assuming that the model, provided with a solution in language X converges to the corresponding Attractor X, we examine the effect on the generation process if we traverse the Attractor space and move to the Attractor of another language Y.

Evaluation. To evaluate the quality of the generated code, we use o4-judge to provide us with a score of the quality of the generated code in the target language. As shown in fig. 9, we can successfully repurpose the LLM as a transpiler without any demonstrations (zero-shot) as well as no other relevant information in the prompt. We achieve impressive results for all pairs of the 4 considered languages. We do not require any retention data, additional training, or an increase in the inference time. We obtain a score better than other simple, train-free approaches (e.g., Difference of Means (DM) [36] and ICV [35]) as well as approaches that involve training auxiliary classifiers (e.g., RFM, LR [36]).

4.3 Remaining on the visual Attractor

Hallucinations in LLMs are a well-known challenge [54, 55, 56], and they worsen in Vision-Language Models (VLMs) due to the fading memory effect—where the model's attention to visual input diminishes during generation, "forgetting" essentially the visual input [57, 58]. We hypothesize that this results from a shift between Attractors: although VLMs initially align with an Attractor encoding visual input, the LLM backbone tends to drift toward a text-only Attractor due to its pretraining. To counter this, we propose adding the initial visual Attractor vector (computed at the first generation step) to the hidden state at each subsequent step, reinforcing visual grounding. Unlike before (fig. 7), our method dynamically calculates and maintains the visual Attractor throughout generation on each generation.

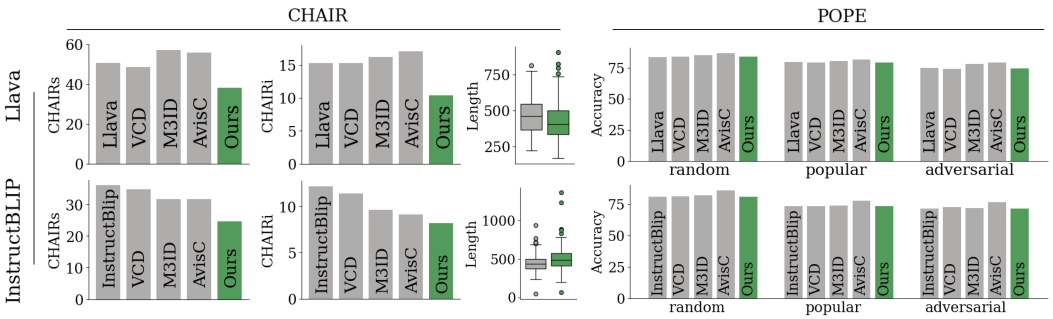


Figure 10: CHAIR [50] and POPE [51] on Llava-1.5 [52] and InstructBLIP [53]. While our approach maintains performance on the discriminative questions (POPE) it significantly reduces the hallucinations in the generative tasks (CHAIR), without affecting the length of the generated descriptions.

Evaluation. Compared to other train-free approaches (e.g., [57, 59, 60]), our algorithm does not lead to an increase in inference time, since it does not require multiple forward passes. Despite its simplicity, the results are strong, leading to a significant reduction in the hallucination rate of two widely used VLMs (InstructBLIP [53] and Llava-1.5 [52]), as shown in Figure 10 (CHAIR). Our modification also does not affect the general abilities of the VLM, resulting in a similar (or slightly improved) performance on discriminative questions.

5 Attractors perturbation for data generation

Recent works have suggested that LLMs can be used for generation of new samples, similar to a (usually small) real dataset. Multiple works have explored different ways that LLMs can be prompted successfully to generate accurate samples, as well as different multi-step ways that can further improve the quality of the newly generated samples [61]. Others have highlighted the difficulty of designing proper prompts and the tedious trial and error required, and proposed a minimal fine-tuning of an LLM so that it serves as an Autoencoder that can produce new samples via high temperature sampling [62].

Limitations of Temperature sampling.LLM output variability is often controlled via Temperature (and its related parameters top-K and top-P), introducing stochasticity into generation. However, even at high randomness, outputs often remain limited and lack diversity when generating text similar to existing data [63, 64, 61]. Usually, this problem is addressed by a more carefully tuned input prompt or the use of many different ones. But such an approach is not easily scalable and not applicable to tasks involving large synthetic data generation.

Specifically, a common way to address the lack of diversity (beyond the capability of temperature sampling alone) is to perform multiple forward passes with different prompts/instructions, while keeping the same sample from the original data. Several proposals show that carefully tuned instructions/prompts can help the model generate different, more diverse types of synthetic samples [63, 64, 61]. However, this requires tedious and careful design of the prompts in a non-automated way, with multiple rounds of trial-and-error in some cases. It is also problematic when we consider large, diverse datasets that do not adhere to a single "type", like the ones we will examine here (e.g., BoolQ [65]).

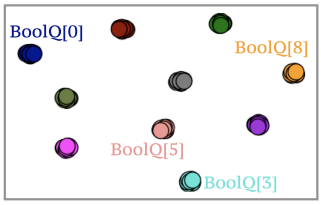


Figure 11: Sample-based Attractors for different generation instructions. Each Attractor corresponds to one sample from BoolQ[65].

5.1 Attractor perturbations: Replicating the effect of multiple tailored instructions

Similar to the experiments reported so far, we can examine if there are sample-wise Attractors for multiple different and diverse instructions. Is there a layer where we can observe a "collapse" based on the different samples? The answer is yes, and we show this in Figure 11. We curated a list of 10 different instructions tailored to BoolQ dataset, and we observed that on layer 16, for the same sample, all trajectories converge to the same Attractor, forming essentially sample-wise Attractors. Based on this observation, we examine whether we can replicate the effect that multiple different prompts

	BoolQ		AG	
	Qwen2.5-0.5B	GPTNeo-1.3B	Qwen2.5-0.5B	GPTNeo-1.3B
No train	38.47	38.53	-	-
No augmentation	62.54	62.17	30.66	23.46
Temp sampling	64.16(±2.98)	64.80(±2.80)	82.91(±2.58)	50.96(±19.45)
Ours	**69.28**(±**0.88**)	**67.77**(±**1.86**)	**85.64**(±**0.59**)	**72.06**(±**7.51**)

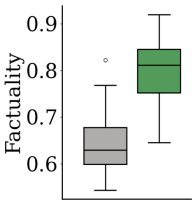


Figure 12: (left) Test-set accuracy on BoolQ [65] and AG [66] when trained with synthetic datasets generated through temperature sampling and our approach. In all cases, our dataset results in a more generalizable model with better performance. (right) Factuality of generated facts about popular figures with temperature sampling (gray) and our approach (green). We observe a more than 20% increase in the factuality on average.

have on the generated data – simply by perturbing the Attractor estimates we obtain using a single instruction. Using only a single (and perhaps simple) prompt/instructions, can we generate multiple, diverse samples without the need to increase the temperature and deal with corrupted, non-sensical samples? As we will show shortly, such simple, train-free and tuning-free approach can result in better quality data, and we check this in both direct and indirect ways.

Estimating the quality of the generated data. Here, we consider textual two datasets: BoolQ [65] and AG [66]. Although both datasets are relatively large/diverse, to limit the influence of the original train set on the final results and better assess each generation method, we consider a minimal version for each dataset, with only 100 samples. Based on these 100 samples, we prompt the model (Llama3.1-8B) to generate new synthetic samples, following both approaches (the typical temperature sampling and ours). One common way to assess the quality of the generated data is by fine-tuning a smaller LLM on the newly obtained version of the data [61] (called an indirect evaluation). Here, we consider Qwen2.5-0.5B [67] and GPTNeo-1.3B [68] as the small LLMs that we will finetune on the synthetic collections. In fig. 12 (left) we show the accuracy obtained on the real test set by training each model with each version of the dataset. The improved quality of our method is clear and we find that it leads to better results in all cases.

Estimating the factuality of the generated data. Besides the indirect comparison described above, we can also examine the quality of the generated samples directly. Following [69], we prompt the model to generate facts for a collection of randomly selected celebrities and historical figures. To evaluate the factuality of each fact, we uses o4-judge by prompting it to output true or false for each of the generated facts. In fig. 12 (right), we show that the factuality of the generated samples is much lower using temperature sampling. Using Attractors, we achieve an absolute increase of 20% on average. A detailed improvement for each person separately can be found on the appendix.

6 Conclusion

This work is based on the hypothesis that the evolution of hidden representations of prompts in Large Language Models (LLMs), specifically their convergence to distinct internal representations (for semantically related prompts), can be understood through the framework of Iterated Function Systems (IFS). We check that LLM layers progressively map inputs towards concept-specific "Attractors" in their latent space. Building on this perspective, we evaluated a range of simple, training-free ideas that directly manipulate these identified Attractors. On a diverse set of practical tasks, including machine unlearning (guardrailing against specific concepts), guiding LLM generation for tasks like code translation and toxicity reduction, mitigating hallucinations in vision-language models, and improving the diversity and factuality of synthetic data generation, we find that our proposal offers surprisingly strong performance. It is computationally efficient and there is no need of re-training or fine-tuning, and offers a clear and promising direction for evaluating applicability in other use-cases.

Impact & Limitations. A single modeling of the LLMs (as IFS) can lead to multiple solutions for very different problems. We believe that these ideas can help solve many more practical problems that LLMs face or expand their capabilities. One limitation of our work is that we require direct access to the hidden activations of the model, for both estimating and manipulating the identified "concept Attractors". Our methods cannot typically be implemented through standard API calls to LLMs, which usually only provide black-box input/output functionality. Due to compute constraints, we focused on evaluating our methods on models with up to 8B parameters but it will be interesting to check if a similar Attractor phenomena holds for much larger models.

References

[1] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

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

[3] Adam Shai, Paul M. Riechers, Lucas Teixeira, Alexander Gietelink Oldenziel, and Sarah Marzen. Transformers represent belief state geometry in their residual stream. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=YIB7REL8UC.

[4] Jesseba Fernando and Grigori Guitchounts. Transformer dynamics: A neuroscientific approach to interpretability of large language models. 2025. URL https://doi.org/10.48550/arXiv.2502.12131.

[5] Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL https://aclanthology.org/2023.findings-emnlp.624/.

[6] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. Once: Boosting content-based recommendation with both open- and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 452–461, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703713. doi: 10.1145/3616855.3635845. URL https://doi.org/10.1145/3616855.3635845.

[7] Oscar Skean, Md Rifat Arefin, and Ravid Shwartz-Ziv. Does representation matter? exploring intermediate layers in large language models. In *Workshop on Machine Learning and Compression, NeurIPS 2024*, 2024. URL https://openreview.net/forum?id=FN0tZ9pVLz.

[8] Michael Barnsley. *Fractals everywhere*. Academic Press Professional, Inc., USA, 1988. ISBN 0120790629.

[9] John E. Hutchinson. Fractals and self similarity. *Indiana University Mathematics Journal*, 30(5):713–747, 1981. ISSN 00222518, 19435258. URL http://www.jstor.org/stable/24893080.

[10] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

[11] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, et al. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128, 2024.

[12] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HtqnVSCj3q.

[13] Zhaoyi Li, Gangwei Jiang, Hong Xie, Linqi Song, Defu Lian, and Ying Wei. Understanding and patching compositional reasoning in LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics ACL 2024*, pages 9668–9688, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-acl.576.

[14] Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3(1):133–181, 1922.

[15] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Chapman and Hall/CRC, 2024.

[16] Randall Balestriero and Richard G. Baraniuk. Mad max: Affine spline insights into deep learning. *Proceedings of the IEEE*, 109(5):704–727, 2021. doi: 10.1109/JPROC.2020.3042100.

[17] Zhilin Wang, Yafu Li, Jianhao Yan, Yu Cheng, and Yue Zhang. Unveiling attractor cycles in large language models: A dynamical systems view of successive paraphrasing. *arXiv preprint arXiv:2502.15208*, 2025.

[18] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[19] Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. Tofu: A task of fictitious unlearning for llms, 2024.

[20] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. Machine unlearning: A survey. *ACM Comput. Surv.*, 56(1), August 2023. ISSN 0360-0300. doi: 10.1145/3603620. URL https://doi.org/10.1145/3603620.

[21] Jean-Marie Chenou and Roxana Radu. The "right to be forgotten": Negotiating public and private ordering in the european union. *Business & Society*, 58(1):74–102, 2019. doi: 10.1177/0007650317717720. URL https://doi.org/10.1177/0007650317717720.

[22] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gn0mIhQGNM.

[23] Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, PRANAY SHARMA, and Sijia Liu. Model sparsity can simplify machine unlearning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 51584–51605. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a204aa68ab4e970e1ceccfb5b5cdc5e4-Paper-Conference.pdf.

[24] Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[25] Min Chen, Weizhuo Gao, Gaoyang Liu, Kai Peng, and Chen Wang. Boundary unlearning: Rapid forgetting of deep networks via shifting the decision boundary. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7766–7775, June 2023.

[26] Ronen Eldan and Mark Russinovich. Who's harry potter? approximate unlearning for llms. 2023.

[27] Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. Towards safer large language models through machine unlearning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1817–1829, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.107. URL https://aclanthology.org/2024.findings-acl.107/.

[28] Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5716–5731, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.310. URL https://aclanthology.org/2024.acl-long.310/.

[29] Haibo Jin, Andy Zhou, Joe D. Menke, and Haohan Wang. Jailbreaking large language models against moderation guardrails via cipher characters. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 59408–59435. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/6d56bc83ae9a4fafdce050bb36f04174-Paper-Conference.pdf.

[30] Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151*, 2024.

[31] Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. Large language model unlearning via embedding-corrupted prompts. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 118198–118266. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/d6359156e0e30b1caa116a4306b12688-Paper-Conference.pdf.

[32] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=MXLBXjQkmb.

[33] Chris Yuhao Liu, Yaxuan Wang, Jeffrey Flanigan, and Yang Liu. Large language model unlearning via embedding-corrupted prompts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=e5icsXBD8Q.

[34] Yu Li, Han Jiang, Chuanyang Gong, and Zhihua Wei. Destein: Navigating detoxification of language models via universal steering pairs and head-wise activation fusion. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=jq2kNXigPP.

[35] Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32287–32307. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bx.html.

[36] Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Aggregate and conquer: detecting and steering llm concepts by combining nonlinear predictors over multiple layers, 2025. URL https://arxiv.org/abs/2502.03708.

[37] Joris Postmus and Steven Abreu. Steering large language models using conceptors: Improving addition-based activation engineering. *arXiv preprint arXiv:2410.16314*, 2024.

[38] Ruixuan Huang. Steering llms' behavior with concept activation vectors, September 2024. Draft manuscript. Available on LessWrong forum.

[39] Zhuohan Gu, Jiayi Yao, Kuntai Du, and Junchen Jiang. Llmsteer: Improving long-context llm inference by steering attention on reused contexts. *arXiv preprint arXiv:2411.13009*, 2024.

[40] Bingqing Song, Boran Han, Shuai Zhang, Hao Wang, Haoyang Fang, Bonan Min, Yuyang Wang, and Mingyi Hong. Effectively steer llm to follow preference via building confident directions. *arXiv preprint arXiv:2503.02989*, 2025.

[41] Varvara Logacheva, Daryna Dementieva, Sergey Ustyantsev, Daniil Moskovskiy, David Dale, Irina Krotova, Nikita Semenov, and Alexander Panchenko. ParaDetox: Detoxification with parallel data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6804–6818, 2022. doi: 10.18653/v1/2022.acl-long.469. URL https://aclanthology.org/2022.acl-long.469/.

[42] A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*, 2017.

[43] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

[44] Chongzhou Fang, Ning Miao, Shaurya Srivastav, Jialin Liu, Ruoyu Zhang, Ruijie Fang, Asmita, Ryan Tsang, Najmeh Nazari, Han Wang, and Houman Homayoun. Large language models for code analysis: do llms really do their job? In *Proceedings of the 33rd USENIX Conference on Security Symposium*, SEC '24, USA, 2024. USENIX Association. ISBN 978-1-939133-44-1.

[45] Paul Denny, David H. Smith, Max Fowler, James Prather, Brett A. Becker, and Juho Leinonen. Explaining code with a purpose: An integrated approach for developing code comprehension and prompting skills. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ITiCSE 2024, page 283–289, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706004. doi: 10.1145/3649217.3653587. URL https://doi.org/10.1145/3649217.3653587.

[46] Nalin Wadhwa, Jui Pradhan, Atharv Sonwane, Surya Prakash Sahu, Nagarajan Natarajan, Aditya Kanade, Suresh Parthasarathy, and Sriram Rajamani. Core: Resolving code quality issues using llms. *Proc. ACM Softw. Eng.*, 1(FSE), July 2024. doi: 10.1145/3643762. URL https://doi.org/10.1145/3643762.

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

[48] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022. doi: 10.1126/science.abq1158. URL https://www.science.org/doi/abs/10.1126/science.abq1158.

[49] Sahil Bhatia, Jie Qiu, Niranjan Hasabnis, Sanjit A. Seshia, and Alvin Cheung. Verified code transpilation with LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=spwE9sLrfg.

[50] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[51] Kun Zhou Jinpeng Wang Wayne Xin Zhao Yifan Li, Yifan Du and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=xozJw0kZXF.

[52] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023.

[53] Wenliang Dai, Junnan Li, DONGXU LI, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 49250–49267. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/9a6a435e75419a836fe47ab6793623e6-Paper-Conference.pdf.

[54] Ariana Martino, Michael Iannelli, and Coleen Truong. Knowledge injection to counter large language model (llm) hallucination. In *European Semantic Web Conference*, pages 182–185. Springer, 2023.

[55] Robert Friel and Atindriyo Sanyal. Chainpoll: A high efficacy method for llm hallucination detection. *arXiv preprint arXiv:2310.18344*, 2023.

[56] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, 2025.

[57] Alessandro Favero, Luca Zancato, Matthew Trager, Siddharth Choudhary, Pramuditha Perera, Alessandro Achille, Ashwin Swaminathan, and Stefano Soatto. Multi-modal hallucination control by visual information grounding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. doi: 10.1109/CVPR52733.2024.01356.

[58] Shi Liu, Kecheng Zheng, and Wei Chen. Paying more attention to image: A training-free method for alleviating hallucination in lvlms. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXXIII*, volume 15141 of *Lecture Notes in Computer Science*, pages 125–140. Springer, 2024. doi: 10.1007/978-3-031-73010-8_8. URL https://doi.org/10.1007/978-3-031-73010-8_8.

[59] Sangmin Woo, Donguk Kim, Jaehyuk Jang, Yubin Choi, and Changick Kim. Don't miss the forest for the trees: Attentional vision calibration for large vision language models. *arXiv preprint arXiv:2405.17820*, 2024.

[60] Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[61] Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. On LLMs-driven synthetic data generation, curation, and evaluation: A survey. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.658. URL https://aclanthology.org/2024.findings-acl.658/.

[62] Giulia DeSalvo, Jean-Fracois Kagy, Lazaros Karydas, Afshin Rostamizadeh, and Sanjiv Kumar. No more hard prompts: Softsrv prompting for synthetic data generation. *arXiv preprint arXiv:2410.16534*, 2024.

[63] Saumya Gandhi, Ritu Gala, Vijay Viswanathan, Tongshuang Wu, and Graham Neubig. Better synthetic data by retrieving and transforming existing datasets. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6453–6466, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.385. URL https://aclanthology.org/2024.findings-acl.385/.

[64] Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. Best practices and lessons learned on synthetic data. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=0JaWBhh61C.

[65] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

[66] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NIPS*, 2015.

[67] Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

[68] Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL https://doi.org/10.5281/zenodo.5297715. If you use this software, please cite it using these metadata.

[69] Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. Fine-tuning language models for factuality. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=WPZ2yPag4K.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: Both the abstract and the introduction report all of our findings, both theoretically and practically.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: Limitations are discussed in the conclusions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

Justification: We do not make any theoretical claims and our work does not include any theorems or proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes] .

Justification: We have included an algorithmic sketch on our main paper and the full details about the hyperparameters on the supplement.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes] .

Justification: All models and datasets are publicly available. The code is also provided.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes] .

Justification: We provide details for all the datasets and models used, as well as the hyperparameter settings in each experiment.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes] .

Justification: When applicable, we include plots with the full distribution of the errors across multiple hyperparameters and their corresponding devation on the tables. The details of each metric used are provided on the supplementary material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes] .

   Justification: All details are provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes] .

   Justification: We acknowledge we have reviewed and conform to the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes] .

    Justification: The impact can be found on the conclusions.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: We present no new data or models in our work.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes] .

    Justification: All used assets are cited appropriately and accordingly.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes] .

    Justification: No new models or datasets are part of the contributions of this work. Our code is documented appropriately.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: There is no crowdsourcing in this work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: There is no crowdsourcing in this work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA] .

    Justification: LLMs were not used for anything beyond grammar checks.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.