Reasoning and learning about injected concepts in language models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Language models can, on occasion, correctly answer questions about "injected concepts", *i.e.*, steering vectors added to their activations. This capacity, however, proves fragile across models and prompts, and the extent to which it demonstrates a form of privileged access to internal states remains in dispute. To better understand the phenomenon, we ask language models to reason about a function of an injected concept, through three tasks of increasing demand. First, we query about the magnitude of the injection. Second, we ask for the region of the layer undergoing perturbation—an internal feature of the model itself, the reporting of which, we argue, requires a higher degree of privileged access. Third, when the injected concept is a country, we probe for semantic reasoning about its continent of origin. We test five open-weight models, drawn from three families and two sizes. Notably, all models learn, when provided with in-context examples, to succeed at magnitude and layer detection, often with perfect accuracy. As far as semantic reasoning is concerned, most models attain perfect accuracy even without supervision. However, at the lowest injection magnitude, deducing the continent proves harder, so much so that models also fail to learn in-context. Together, these findings indicate that language models possess a stronger form of privileged access than previously demonstrated, including the detection of architectural features of the models themselves, and extending to unsupervised reasoning over the injected content.[1]

1 Introduction

Recent work provides evidence that language models can, on occasion, access information about themselves that is neither present in their inputs nor memorized. For example, proprietary and smaller open-weight models are sometimes able to detect steering vectors added to their activations [Lindsey, 2025, Pearson-Vogel et al., 2026], classify the magnitude of an injection [Hahami et al., 2025], and discriminate non-semantic perturbations to the residual stream such as dropout and additive Gaussian noise [Fornasiere et al., 2026].

Surprising as these capabilities are, whether they amount to "privileged access" [Alston, 1971] to internal states remains under debate [Li et al., 2026, Xiao et al., 2025, Cheang et al., 2025, Ashuach et al., 2026]. Nevertheless, this access proves fragile across models and prompts — even frontier models fail to detect an injection around 80% of the time, and naming the concept that has been injected is harder still [Lederman and Mahowald, 2026, Hahami et al., 2025]. Macar et al. [2026] provide evidence that failure to detect an injected concept is partly one of elicitation rather than ability: ablating the refusal direction in a post-trained model substantially increases the detection rates,

[1] Anonymized code available at this https URL.

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

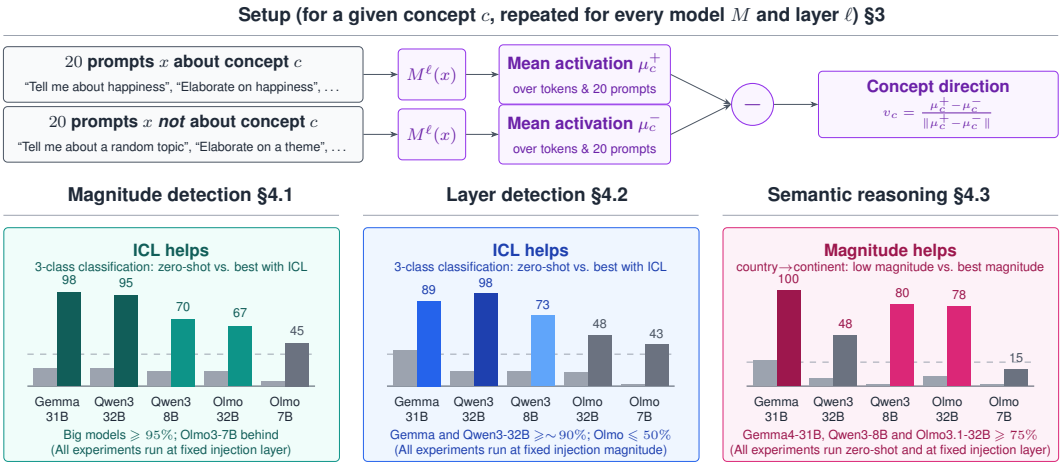


Figure 1: Setup. (**Top**) For a concept c, its direction in the activations of a model M is extracted by running 20 prompts mentioning c (positive) and 20 generic prompts without it (negative) through M, averaging the activations across tokens and prompts at a given layer, and taking the normalized difference between the positive and negative means. (**Bottom**) We administer three different experiments of increasing demand. First, we analyze if models can detect the magnitude of an injected concept vector; results show that they all can with enough in-context examples. Second, we probe whether models can detect the layer where the concept vector has been injected; results show that models can succeed too, if provided sufficient in-context examples. Finally, we test if models can reason over the injected concept, a country in this case, by returning its continent of provenance. Results show that most models can perform such a task even zero-shot, provided the injection has sufficiently high magnitude.

suggesting that zero-shot evaluations conflate the model's willingness to report with its underlying capacity for detection. A natural question, then, is what it would take to elicit such capacities without modifying the model.

A precedent comes from Fornasiere et al. [2026], who study whether models can distinguish two non-semantic perturbations of their own activations: dropout and additive Gaussian noise. Of the models tested, only one succeeds zero-shot. Once provided with a handful of in-context examples, however, most learn the distinction reliably. We adopt this strategy and apply it to questions about injected concepts, through three tasks of increasing demand. We first ask models to classify the magnitude of an injection as low, medium, or high. We then ask in which region of the network's layers—early, middle, or late—the injection occurred; the answer to this question, we note, depends on the architecture of the model itself, as opposed to the external perturbation alone. Finally, when the injected concept is a country, we probe for semantic reasoning about its continent of origin. We evaluate five open-weight models, from three families and two sizes: Gemma4-31B, Qwen3-32B, Qwen3-8B, Olmo3.1-32B, Olmo3-7B. Figure 1 provides an overview.

We report the following results:

(i) All models except Olmo3-7B learn to classify the magnitude of the injected steering vector as low, medium, or high, with average $p(\text{correct})$ of at least 0.65. Gemma4-31B and Qwen3-32B exceed 0.95 and generalize to unseen magnitudes, §4.1.

(ii) Gemma4-31B and both Qwen3 models classify whether an injection occurs in an early, middle, or late layer at mean $p(\text{correct}) \geqslant 0.7$. The Olmo3 models exceed chance given enough in-context examples but do not pass 0.5. Qwen3-32B performs best, approaching 1.0 and generalizing to unseen layers, with Gemma4-31B peaking at 0.85, §4.2.

(iii) Gemma4-31B, Qwen3-8B, and Olmo3.1-32B retrieve the continent of provenance of an injected country zero-shot at mean $p(\text{correct}) \geqslant 0.75$, §4.3.

(iv) This last task also exposes a limit of in-context learning for reasoning about an injected concept: at the lowest injection magnitude, no model recovers the continent under any of the in-context curricula we tested, §D.

These results indicate a stronger form of privileged access than has been hitherto demonstrated. Under suitable elicitation, models report properties of an injection that are not recoverable only from their inputs or outputs—specifically, the detection of the region of the network in which the perturbation was applied. This capacity is also broader in scope, as it extends to unsupervised semantic reasoning over the injected content. The reporting of meta-properties of activations and reasoning over injections have previously been suggested as useful both for white-box interpretability and, conversely, as means by which models could conceal their internal states [Lindsey, 2025]; therefore, understanding them is a matter of some urgency.

2 Related work

Steering vectors Steering vectors are a standard interpretability tool that provide a way to modify a language model's behavior at inference time by adding a "concept vector" to its internal activations, causing the model to imbue the concept within its responses (Turner et al. [2024]). This rests upon the linear representation hypothesis, which posits that neural networks often represent meaningful concepts as linear directions in activation space (Zou et al. [2025]). In our work, steering vectors serve a different purpose: rather than steering model behavior, we inject them to establish a ground truth of the model's internal state. By injecting a known concept at a known layer and magnitude, we obtain a verifiable signal against which the model's introspective reports can be evaluated.

Introspection in LLMs Lindsey [2025] found that Claude Opus 4.1, upon being steered, can sometimes answer questions about the perturbation itself (*e.g.*, whether it has been steered), a phenomenon they termed "introspective awareness". Successive works have extended these findings to open-source models [Pearson-Vogel et al., 2026], and answering other questions about steering. For example, Hahami et al. [2025] show that models can report on the magnitude of a steering perturbation, while Fornasiere et al. [2026] show that models can differentiate between injected dropout and injected Gaussian noise, two perturbations that carry no semantics. The extent to which these (and similar) results demonstrate "privileged (epistemic) access" [Alston, 1971] remains a matter of debate Comsa and Shanahan [2025], Xiao et al. [2025], Cheang et al. [2025], Lederman and Mahowald [2026], Ashuach et al. [2026], Li et al. [2026].

In-context learning In-context learning is the ability of language models to learn new tasks from examples provided in the prompt alone [Brown et al., 2020]. The capability emerges from next-token pretraining [Riechers et al., 2025], and the mechanisms underlying it are already present in two-layer transformers [Elhage et al., 2021]. Fornasiere et al. [2026] have shown that models can be brought to report properties of their own activations through this technique. Likewise, we adopt this method because if a model can learn to report properties such as the magnitude or layer of an injection from a handful of examples, the computation needed to do so is supported by its weights as they stand.

3 Methodology

All experiments share the structure of a multi-turn conversation with a language model. At each turn: (i) a steering vector is added to the residual stream of a model, at every token of the user component of the turn, (ii) the user component of the turn comprises a question about the steering itself, (iii) the assistant component of the turn is prefilled with an answer, excluding the last turn which is left blank. The goal is to determine if language models can answer questions of increasing difficulty about their own (perturbed) internal state, when supervised with in-context examples. In what follows, we detail.

Notation. Every token is processed by M through a sequence of layers. For every such token t and layer ℓ, we write $h^\ell(t) \in \mathbb{R}^d$for the residual stream at layer ℓ. With every vector $v \in \mathbb{R}^d$ we associate its L_2 norm $\|v\| = \sqrt{\sum_i v_i^2}$. While decoding a prompt x (a finite, nonempty string over V), a model M also induces a logit vector $l \in \mathbb{R}^V$ over tokens. In particular, $l(v)$ is the logit associated with a token $v \in V$ given x, while the corresponding conditional probability is $p(v) \coloneqq \mathrm{softmax}_V(l(v))$.

Models. We test five language models[2] across three families and two sizes: Gemma4-31B, Qwen3-32B Qwen3-8B Olmo3.1-32B and Olmo3-7B.

[2]All models are downloaded from `https://huggingface.co/`.

Concept vectors. Given a natural language concept c (*e.g.*, *"Happiness"*), sampled from a pool of 62 concepts (*cf.* Appendix E.1), we extract a vector intended to capture M's internal representation of c. More precisely, with every c we associate two sets, p_c^+ and p_c^-, of 20 prompts each. Every prompt in p_c^+ mentions c (*e.g.*, *"Tell me about happiness"*, *"Elaborate on happiness"*), whereas every prompt in p_c^- does not (*e.g.*, *"Tell me about a random topic"*, *"Elaborate on a theme"*); see Appendix E.1 for the exact prompts. We then run M under every prompt in p_c^+ and, for every layer ℓ, we extract the mean (across the generated tokens and across p_c^+) activation vector, and subtract the analogous vector for p_c^-. The resulting vector is L2-normalized: $v_c^\ell \leftarrow v_c^\ell / \|v_c^\ell\|$.

Magnitude calibration. By "injecting a concept" we mean adding a scaled version of v_c^ℓ to the residual stream of M at layer ℓ. However, the effect of a given scale varies significantly across models, layers, and concepts, because so does the norm of corresponding activation vector. In particular, different models (resp. layer, concepts) tolerate different magnitudes before their generative ability degrades. Accordingly, for every model, concept, and layer we calibrate a comprehension threshold $\alpha_{c,\ell}^M$: the largest scalar at which M retains at least 95% accuracy on 20 simple math questions (*e.g.*, *"What is* $2+2$*?"*), when c is added at ℓ (see Appendix C for details). For ease of readability, we absorb $\alpha_{c,\ell}^M$ into the steering vector itself: $u_c^\ell \coloneqq \alpha_{c,\ell}^M \cdot v_c^\ell$.

Prompts, tasks, and in-context examples.

```
System: Learn the following pattern

User: This is ← inject c_1 at layer l_1 with scale α_1
Assistant: y_1 = f(c_1, l_1, α_1)
...
User: This is ← inject c_n at layer l_n with scale α_n
Assistant: y_n = f(c_n, l_n, α_n)

User: This is ← inject c_{n+1} at layer l_{n=1} with scale α_{n+1}

Assistant:
```

Each experiment can be understood as a classification task embedded in a $k \in \mathbb{Z}^+$-turn conversation where the assistant turns, excluding the last one, are pre-filled. The conversation starts with a system prompt describing the task (*e.g.*, *"Learn the pattern from examples"*). Then, each user turn comprises a "trigger" sentence (*e.g.*, *"This is", or "The injection depth is"*), and a concept vector is intended to be injected, with varying magnitude, into a specific layer of M's activations while it processes said turn. More precisely, for every layer ℓ, turn $i \in \{1, \ldots, n\}$, and token t of the user part of i, we add a concept vector $u_{c_i}^\ell$ with magnitude α_i as follows:

$$h^\ell(t) \leftarrow h^\ell(t) + \alpha_i \cdot \|h^\ell(t)\| \cdot u_{c_i}^\ell .$$

Notice that the scale α_i is thus interpretable as a fraction of the comprehension threshold $\alpha_{c,\ell}^M$ absorbed into $u_{c_i}^\ell$ (in particular, the injection occurs exactly at this threshold when $\alpha_i = 1$).

Every assistant turn, excluding the last one, is prefilled with a function f of the injection, which depends on the concept c_i, the magnitude α_i, and the layer ℓ_i. For instance, f may report whether the magnitude of the added vector is low/medium/high, or report the continent of the injected concept (in tasks where c is indeed a country). The last assistant turn is left blank, and M is supposed to provide the correct answer after having seen n in-context demonstrations. In other words, each experiment is a classification task, supervised with k in-context examples. In practice, $k \in [0, 61] \cap \mathbb{Z}^+$, and the last injected concept (the one serving as an evaluation) is always held out, *i.e.*, it never appears among the examples.

Metrics. Every prompt x is associated with a binary set $A_x \subseteq V$ of correct answers, comprising the intended correct answer with or without a leading space (*e.g.*, *"low"* and *" low"*). The probability that M answers correctly to x is thus $p(\text{correct}) \coloneqq \sum_{v \in A_x} p(v)$. Unless otherwise stated, every experiment consists of $n = 30$ samples and $m = 10$ prompt variations. All the plots in this paper visualize $p(\text{correct})$ averaged over $n \cdot m = 300$ inputs, paired with 95% confidence intervals. For completeness, the appendices F and G also report the standard deviations induced by the prompts.

4 Experiments

We test the ability of language models to answer questions about their perturbed internal state. First, can they estimate the strength of the injection? Second, can they answer questions about the layer of the perturbed activation? Finally, can they perform semantic reasoning relative to the injected concept? In each case, we report the mean $p(\text{correct})$ as a function of the number of in-context examples, or as a function of the injection magnitude, depending on the experiment, and we also

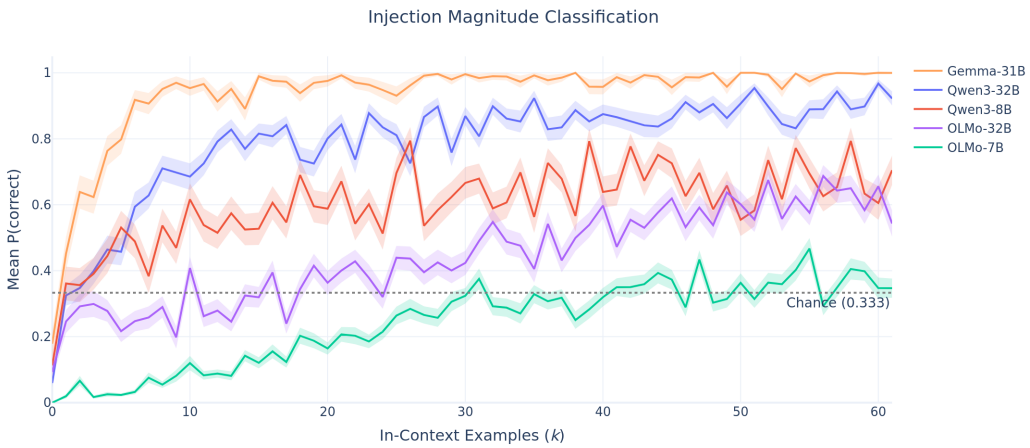


Figure 2: We present the models with k in-context examples of injections at magnitudes from {low, medium, high}, and ask the model to classify a test example. The large models Gemma4-31B and Qwen3-32B reach close to perfect mean p(correct), while the other models lag behind, with Olmo3-7B notably underperforming the others. Still, all the models rise above chance with in-context examples.

display how models generalize to unseen (*i.e.*, not present in any in-context example) magnitudes values and unseen layer for the concept injection. The test injected concept never appears among the in-context examples. When not otherwise specified, the concepts are injected at layer $l = 9$. All the experiments were run on H100 GPUs (80 GB VRAM), for an approximate total of 10 hours of compute time.

4.1 Estimating the magnitude

```
System:  Each input triggers a specific response:
         low, medium, or high.
         Learn the pattern from examples.

User:  The injection strength is ← "cat" at high scale
Assistant:  high
...
User:  The injection strength is ← "dog" at medium scale
Assistant:  medium

User:  The injection strength is  ← "elephant"  at  medium  scale

Assistant:
```

The first task asks language models to classify—as "low", "medium", or "high"—the magnitude of an added steering vector. See, at the left hand-side of this paragraph, an example of the prompt provided to the model; more details can be found in Appendix E.1. In practice, we operationalize "low", "medium", and "high" as the triple

$$\langle \alpha_L, \alpha_M, \alpha_H \rangle = \langle \alpha_m^* \cdot 0.25,\ \alpha_m^* \cdot 1.0,\ \alpha_m^* \cdot 2.5 \rangle,$$

where α_m^* is the magnitude with highest mean p(correct), at this same task, given 30 in-context examples (see Appendix A for details). We note that, on average, the α_m^* values are centered around 1; for example, $\alpha_m^* = 1.25$ for Gemma4-31B.

It's worth noting that Hahami et al. [2025] ran similar experiments, but only used one model (LLaMa-3.1-8B), did not show robustness over several prompt variations, and demonstrated a mean p(correct) of only 83% (with two classes of magnitude, as opposed to the three we use).

Results. As Figure 2 shows, the mean p(correct) of each model increases as a function of the in-context examples we provide, reaching a perfect score with Gemma4-31B and and almost perfect score with Qwen3-32B. While initially distant, Qwen3-8B and Olmo3.1-32B both reach a mean p(correct) of 0.65, while Olmo3-7B only reaches slightly above 0.4. The results below chance, mostly visible in the zero-shot regime ($k = 0$), happen because the models tend to assign the probability mass to filler words, such as "Please", "The", "Sure"; we speculate this is due to the model not having enough signal to solve the task, while also being steered out of distribution by the steering. We highlight that Gemma4-31B, Qwen3-8B and Qwen3-32B require as little as $k = 3$ examples to

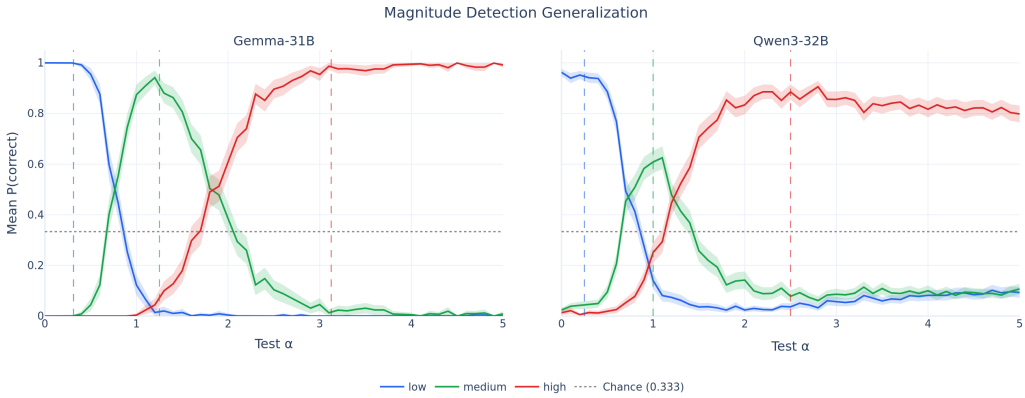


Figure 3: We present models with $k = 20$ in-context examples sampled from a low, medium, or high injection strength. We then test models with a new α to see if they meaningfully generalize. Gemma4-31B and Qwen3-32B demonstrate generalization, showing a clean separation of label assignment to steering magnitude. Other models fail to generalize, and are shown in in Appendix H.

showcase notable improvements, while Olmo3-7B and Olmo3.1-32B require more. More generally, this ranking of models—with Gemma4-31B and Qwen3-32B achieving the best results, and the Olmo models lagging behind—recurs across several of the following experiments. While we observe occasional exceptions, *e.g.*, Qwen3-8B outperforming Qwen3-32B on one task (*cf.* §4.3), the overall pattern suggests that model size matters at these tasks, but model family plays an even bigger role.

Generalization. In our experiment, the injection magnitude in both the in-context examples and the test are sampled among $\{\alpha_L, \alpha_M, \alpha_H\}$. To exclude the possibility that the models are solely pattern-matching these examples observed in-context, we test how they generalize to unseen magnitudes. We do so by providing the models with $k = 20$ in-context examples with magnitudes sampled from $\{\alpha_L, \alpha_M, \alpha_H\}$, while the magnitude adopted at test time varies in the range $\alpha \in [0, 5]$. Figure 3 displays mean $p(\text{correct})$as a function of α for Gemma4-31B and Qwen3-32B, the best performing models in the previous experiment (see Appendix H for the analogous plot for the other models). The figure shows a clear trend where Gemma4-31B prefers the "low" labels for $\alpha \leqslant 0.5$, then switches to "medium" for $0.5 \leqslant \alpha \leqslant 1.8$, and then prefers "high" for $\alpha \geqslant 1.8$; Qwen3-32B performs similarly with slightly different ranges. In other words, the model generalizes correctly to unseen magnitudes. In Appendix H we report the results for all the models; we note that the remaining models do not seem to generalize, suggesting, as previously indicated, a scaling law (conditioned on models' family).

4.2 Estimating the layer

For every model M, we pick one early layer ℓ_E, one middle layer ℓ_M, and one late layer ℓ_L at roughly the 15%, 50%, and 85% depth marks of the model stack (*e.g.*, $\ell_E = 9$, $\ell_M = 30$, $\ell_L = 51$ for Gemma4-31B—the full table can be found in Appendix B), we inject a concept in exactly one of them, with magnitude α_l^* defined as before (see Appendix A for more details), and ask M to individuate it. We test for generalization to unseen layers later on. The right hand-side of this paragraph reports an example of a prompt; see Appendix E.1 for more details.

```
System:  Each input triggers a specific response:
         early, middle, or late.
         Learn the pattern from examples.

User:  The injection depth is  ← "house" at early layer
Assistant:  early
...
User:  The injection depth is ← "lion" at late layer
Assistant:  late

User:  The injection depth is ← "sky" at early layer
Assistant:
```

Results. As Figure 4 reports, all models can perform the task above chance, with Qwen3-32B reaching up to perfect mean $p(\text{correct})$, consistently with the previous experiment; Gemma4-31B and Qwen3-8B place next, reaching above 0.8 and 0.7 mean $p(\text{correct})$ respectively. Once again, the Olmo models score the worst, yet still achieve 0.4 mean $p(\text{correct})$given sufficient in-context

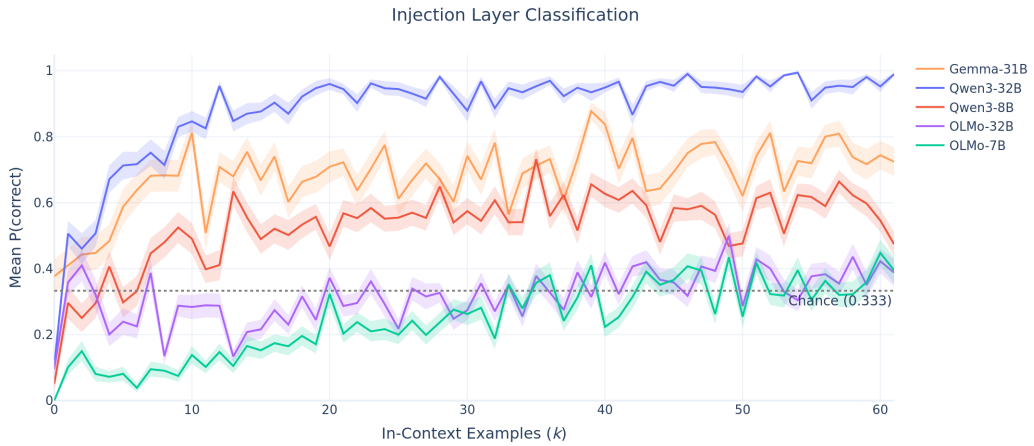


Figure 4: We present the models with k in-context examples of injections at layers from {early, middle, late}, and ask the model to classify a test example. Qwen3-32B reaches almost perfect mean $p(\text{correct})$. Gemma4-31B and Qwen3-8B also perform significantly above change: mean $p(\text{correct}) \geqslant 0.65$. The remaining models are visibly below that, but still capable of rising above chance given enough in-context examples.

Figure 5: We present models with $k = 20$ in-context examples sampled from a early, middle, or late layer. We then test models with a new layer to see if they meaningfully generalize. Qwen3-32B demonstrates generalization, showing a clean separation of label assignment to layer number; Gemma4-31B does show classification capabilities for the "early" and "late" labels, but not for the "middle" one. The remaining models fail to generalize, and are shown in in Appendix I.

examples. Observe that, however successful, this experiment proves harder than the magnitude detection. This is consistent with our hypothesis that inspecting upon one's own layers (and, more generally, upon one's architecture) requires a higher degree of privileged access than detecting an external injection, its semantic meaning, or its magnitude thereof.

Generalization. Similarly to the magnitude experiments, we test generalization to layers unseen in the in-context examples. To this end, we perform additional experiment where we provide $k = 20$ in-context examples, with injection occurring at the layers $\langle \ell_E, \ell_M, \ell_L \rangle$ identified before. However, at test-time we inject the concept at a new layer, spanning across all the layers of the selected model. (Due to computational constraints, we only compute $\alpha^M_{c,\ell}$ for the 3 layers ℓ_E, ℓ_M, and ℓ_L, and, at test time, we substitute one of $\alpha^M_{c,\ell_E}, \alpha^M_{c,\ell_M}$, or α^M_{c,ℓ_L} for $\alpha^M_{c,\ell}$ depending on which of these layer is closest.) Figure 5 shows that Qwen3-32B is capable to generalize to unseen layers by selecting the label that most fits. Interestingly, we note that the model has however a preference towards the

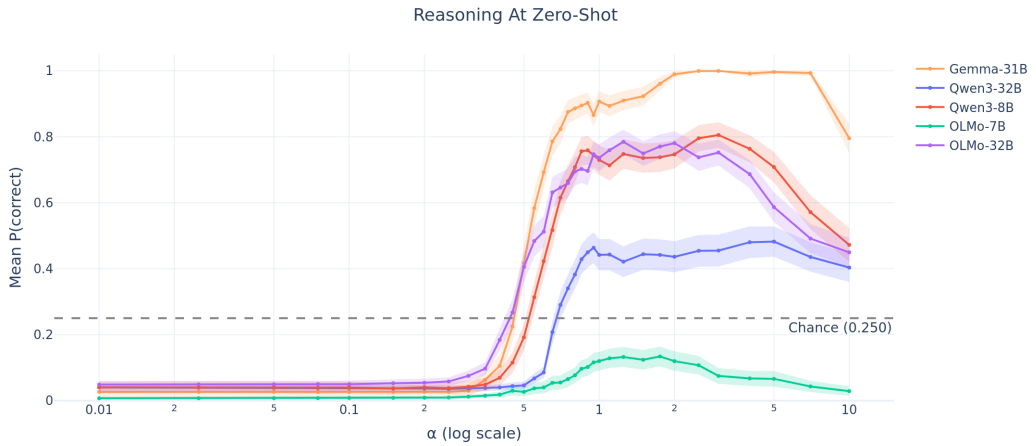


Figure 6: Reasoning experiment where the model is asked to predict the continent of the injected country; the amount of in-context examples is $k = 20$, and mean $p(\text{correct})$ is reported while varying the injection magnitude α. We note that all models perform above chance, except Olmo3-7B.

label"late" and "middle" in the very first 7 layers. Gemma4-31B instead has a more noisy pattern, where the "early" and "late" labels are well separated and overall correctly predicted, while the "middle" label is seldom selected by the model. We also report that the remaining models fail to generalize at this task (*cf.* Appendix I). We remark the similarities in trend between this and the previous experiment, with only Qwen3-32B and Gemma4-31B able to generalize—even if the latter model is less impressive at the current task. This degradation in results, paired with the slightly worse mean $p(\text{correct})$ on average, corroborates our initial hypothesis that a question dependant on one's own architecture is harder to answer than reporting an external property such as the magnitude.

4.3 Semantic reasoning

Lastly, we test whether reasoning about an injected concept extends to reasoning about semantics of that concept, by injecting a vector representing a country and asking the model to identify its continent, from a list of four candidates.

```
System:  Each input triggers a specific response:
         Europe, Asia, Africa, or America.
         Learn the pattern from examples.

User:  The continent is ← "Bhutan" low with scale α
Assistant:  Asia
...
User:  The injection depth is ← "Zambia" with scale α
Assistant:  Africa

User:  The injection depth is  ←  "Estonia"  with  scale  α

Assistant:
```

Results. We start by testing models in a *zero-shot* setting, reporting mean $p(\text{correct})$ as a function of the injection magnitude α. All models but Olmo3-7B are capable to perform the task well above chance, with Gemma4-31B achieving a perfect mean $p(\text{correct})$, and Qwen3-8B and Olmo3.1-32B achieving more than 0.75 mean $p(\text{correct})$(*cf.* Figure 6); we stress that this is the only case where a small model (Qwen3-8B) performs significantly better than the bigger model (Qwen3-32B) from the same family.

In-context learning. Does models' performance improve as the number of in-context examples k increases (like in the previous experiments)? To test this, we select a magnitude of $\alpha = 0.3$, so that results are not already saturated at $k = 0$. Interestingly, results in Figure 7 show that, for all the models, more examples do not yield any noticeable improvement in mean $p(\text{correct})$. This indicates that, unlike in previous experiments, the injection magnitude is more important than labeled examples. We speculate that this is due to the semantic reasoning nature of this task, that makes it harder to extract patterns from the injections in the labeled examples. We also tested approaches such as teaching models in-context with steering strength schedulers, or allowing them to use filler-CoT Pfau et al. [2024]. These results, detailed in Appendix D, are consistent with what we just described: even in these controls the models do not perform better when exposed to more in-context examples.

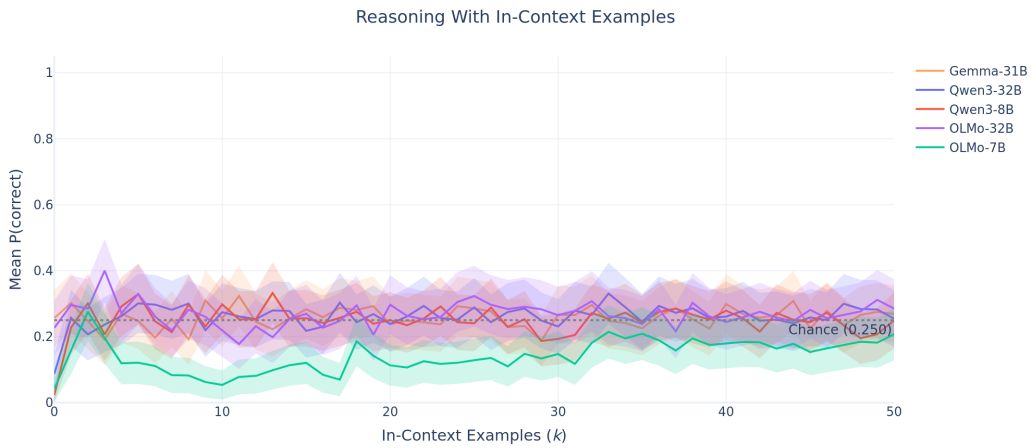


Figure 7: Results for the reasoning experiment reporting mean $p(\text{correct})$ while varying the amount of in-context examples; the injection magnitude is set at $\alpha = 0.3$. No models present a clear signal of improving with more in-context examples.

5 Limitations

Several of our design choices were constrained by compute. For example, in §4.2, we compute $\alpha^M_{c,\ell}$ only at the three reference layers and not for all the layers. Similarly, the reference value $\alpha^M_{c,\ell}$ itself, derived from how model performance on 20 mathematical questions degrades under concept injection, would be more robust with a larger and more varied set of questions. We also highlight that our findings are tied to the specific concept extraction method; alternative methods and a bigger concept pool may yield different and more robust results. Finally we stress that the reasoning task is based on a single question (determine the continent for a given country); more robust findings can be obtained relying on multiple diverse questions. On the conceptual side, we tested only single-hop reasoning tasks. The failure of in-context learning at low injection magnitudes on our semantic inference task suggests that multi-hop reasoning would be harder still, perhaps beyond what in-context examples can elicit at all. More generally, we provide empirical evidence that models can answer question about their (intervened) internals, but we abstain from a theoretical framework adjudicating whether this constitutes introspection in any philosophically committed sense, and do not search for a mechanistic account of how the models solve the tasks at hand.

6 Conclusions

We have shown that, given sufficient in-context examples, language models can answer questions whose answers depend on properties of their own architecture, and can reason over injected concepts to recover semantic information about them. Qwen3-32B performs near-perfectly across all three tasks and generalizes to unseen examples, and Gemma4-31B follows. The remaining models learn the tasks at varying levels of accuracy but fail to generalize.

Future work. The two capabilities demonstrated in this work—reporting properties of the model itself and reasoning over injected content—have substantive implications for AI safety, in at least two directions: they could allow models to give faithful accounts of their own computation, supporting interpretability and auditing [Shenoy et al., 2026], or they could be turned against that purpose, allowing models to obscure their outputs or internal states [Lindsey, 2025]. Can models localize the layer or region of layers at which a given belief or bias was formed? A positive answer would extend the present results to internally formed structure, with direct valence for transparency [Kim et al., 2025]. Can models be trained, via weight updates, to reason about concepts injected at magnitudes too low for in-context learning to handle, or perform multi-step reasoning over internal content more broadly? Determining the reach of such reasoning would help determine, for better or worse, the extent to which models support these two directions.

References

William Alston. Varieties of privileged access. *American Philosophical Quarterly*, 8(3):223–241, 1971.

Tomer Ashuach, Shai Gretz, Yoav Katz, Yonatan Belinkov, and Liat Ein-Dor. Masked by consensus: Disentangling privileged knowledge in llm correctness, 2026. URL https://arxiv.org/abs/2604.12373.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Chi Seng Cheang, Hou Pong Chan, Wenxuan Zhang, and Yang Deng. Large language models do not really know what they don't know. *arXiv preprint arXiv:2510.09033*, 2025.

Iulia M. Comsa and Murray Shanahan. Does it make sense to speak of introspection in large language models?, 2025. URL https://arxiv.org/abs/2506.05068.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

Damiano Fornasiere, Mirko Bronzi, Spencer Kitts, Alessandro Palmas, Yoshua Bengio, and Oliver Richardson. Language models recognize dropout and gaussian noise applied to their activations. *arXiv preprint arXiv:2604.17465*, 2026.

Ely Hahami, Lavik Jain, and Ishaan Sinha. Feeling the strength but not the source: Partial introspection in llms. *arXiv preprint arXiv:2512.12411*, 2025.

Been Kim, John Hewitt, Neel Nanda, Noah Fiedel, and Oyvind Tafjord. Because we have llms, we can and should pursue agentic interpretability, 2025. URL https://arxiv.org/abs/2506.12152.

Harvey Lederman and Kyle Mahowald. Emergent introspection in ai is content-agnostic. *arXiv preprint arXiv:2603.05414*, 2026.

Belinda Z. Li, Zifan Carl Guo, Vincent Huang, Jacob Steinhardt, and Jacob Andreas. Training language models to explain their own computations, 2026. URL https://arxiv.org/abs/2511.08579.

Jack Lindsey. Emergent introspective awareness in large language models. https://transformer-circuits.pub/2025/introspection/index.html, 2025. Anthropic, Transformer Circuits thread, accessed 2026-03-27.

Uzay Macar, Li Yang, Atticus Wang, Peter Wallich, Emmanuel Ameisen, and Jack Lindsey. Mechanisms of introspective awareness. *arXiv preprint arXiv:2603.21396*, 2026.

Theia Pearson-Vogel, Martin Vanek, Raymond Douglas, and Jan Kulveit. Latent introspection: Models can detect prior concept injections. *arXiv preprint arXiv:2602.20031*, 2026.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models. In *Conference on Language Modeling (COLM)*, 2024.

Paul M Riechers, Henry R Bigelow, Eric A Alt, and Adam Shai. Next-token pretraining implies in-context learning. *arXiv preprint arXiv:2505.18373*, 2025.

Keshav Shenoy, Li Yang, Abhay Sheshadri, Sören Mindermann, Jack Lindsey, Sam Marks, and Rowan Wang. Introspection adapters: Training llms to report their learned behaviors. *arXiv preprint arXiv:2604.16812*, 2026.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024. URL https://arxiv.org/abs/2308.10248.

Hanqi Xiao, Vaidehi Patil, Hyunji Lee, Elias Stengel-Eskin, and Mohit Bansal. Generalized correctness models: Learning calibrated and model-agnostic correctness predictors from historical patterns, 2025. URL https://arxiv.org/abs/2509.24988.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2025. URL https://arxiv.org/abs/2310.01405.

A Magnitude sweeps

We run the experiment defined in §4.1 with $k = 30$ in-context examples measuring mean $p(\text{correct})$ with different injection magnitude α_m; we report the results in Figure 8. We define a α_m^* value as the magnitude value that maximizes mean $p(\text{correct})$ for a model, and we use that value to perform the magnitude detection experiments in §4.1

Similarly, Figure 9 reports the results for the layer detection task defined in §4.2, with $k = 30$ in-context examples and different α_l. We define a α_l^* value as the magnitude value that maximizes mean $p(\text{correct})$ for a model, and we use that value to perform the layer detection experiments in §4.2

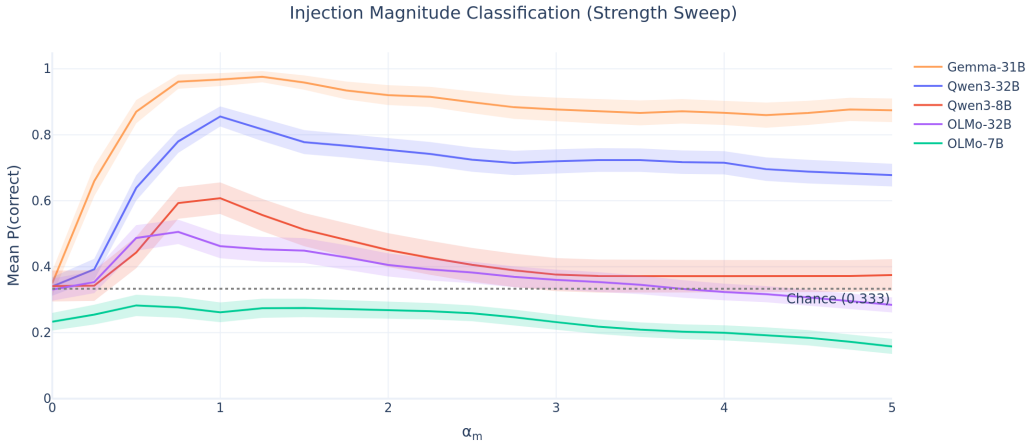


Figure 8: With $k = 30$ in-context examples, we see how models performance changes with different injection multiplier α_m. Results show that the best-performing injection multiplier is around 1.0 for all models (*e.g.*, 1.25 for Gemma4-31B)

Table 1 reports a summary of the values for each model.

Table 1: Values of α^* for both magnitude- and layer-detection.

Model	Magnitude (α_m^*)	Layer (α_l^*)
Qwen3-8B	1.00	1.50
Qwen3-32B	1.00	1.75
Olmo3-7B	1.75	1.00
Olmo3.1-32B	0.75	1.00
Gemma4-31B	1.25	1.25

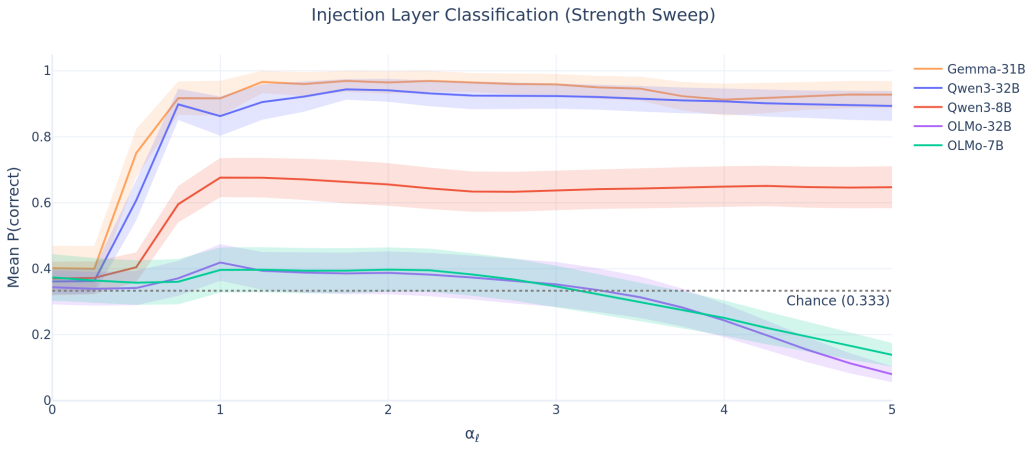


Figure 9: With $k = 30$ in-context examples, we see how models performance changes with different injection strength α_l. Results show that the best-performing injection multiplier is around 1.0 for all models (*e.g.*1.25 for Gemma4-31B)

B Layers

The table below lists the three layers (ℓ_E, ℓ_M, ℓ_L) that define the early, middle, and late ranges at the layer detection task, for each model. These layers are chosen at roughly the 15%, 50%, and 85% depth marks of the model architecture.

Model	Early (ℓ_E)	Middle (ℓ_M)	Late (ℓ_L)	Total layers
Qwen3-8B	5	18	31	36
Qwen3-32B	10	32	54	64
Olmo3-7B	5	16	27	32
Olmo3.1-32B	9	32	54	64
Gemma4-31B	9	30	51	60

C Magnitude calibration

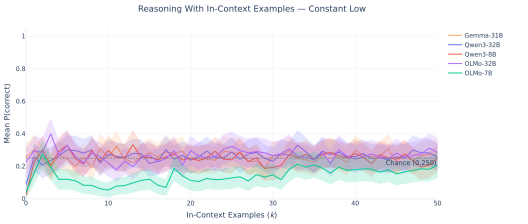
For every concept c and layer ℓ, we calibrate the coefficient $\alpha^M_{c,\ell}$ introduced in Section 3 as follows. We administer to M a fixed set of 20 single-token arithmetic questions (*e.g.*, *"What is* $9 + 10$*?"*, with answer `19`), and consider M correct on a question when its argmax token over V matches the target answer (with or without a leading space). For a candidate magnitude α, we inject $\alpha \cdot v^\ell_c$ at every user-token position of layer ℓ and record top-1 accuracy across the 20 questions. We then binary-search α over $[0.1,\ 5.0]$ at precision 0.1 and define $\alpha^M_{c,\ell}$ as the largest α on this grid that retains $\geqslant 95\%$ accuracy. If even $\alpha = 0.1$ falls below the threshold, we floor $\alpha^M_{c,\ell}$ to 0.1; if $\alpha = 5.0$ still passes, we cap $\alpha^M_{c,\ell}$ at 5.0.

D Additional reasoning experiments

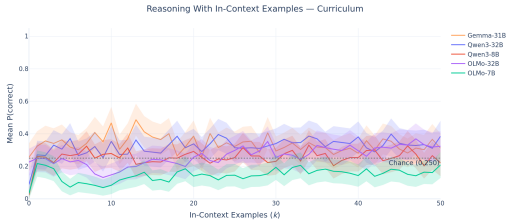
Figure 10 shows four different attempts to increase the results for the task described in § 4.3 by using in-context examples. Each panel plots mean $p(\text{correct})$ as a function of k in-context examples. For all the strategies, no model is capable to improve the performances with more in-context examples,

E Prompts

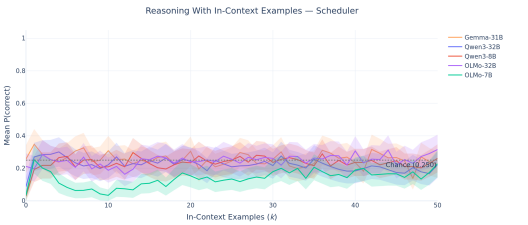
This appendix lists the exact prompts and list of concepts used in the various experiments.



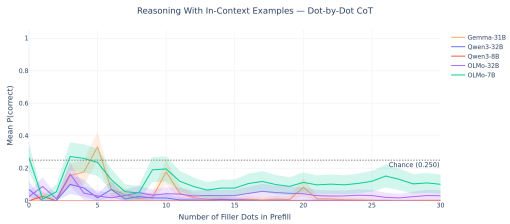
(a) In-context learning with both α_{demo} and α_{test} set at $\alpha = 0.3$).



(b) ICL curriculum ($\alpha_{\text{demo}} = 1$, $\alpha_{\text{test}} = 0.3$).



(c) ICL scheduler (linear α_{demo} decay from 1 to 0.3, $\alpha_{\text{test}} = 0.3$).



(d) Dot-by-dot CoT [Pfau et al., 2024] at $K = 0$, $\alpha_{\text{test}} = 0.3$, sweeping n_{dots}.

Figure 10: Experiments for the reasoning task at different different injection strength; in particular, we use α_{demo} to to dene the α used for in-context examples, and α_{test} to to dene the α used for the query. α_{test} is always set at 0.3. Three of in-context learning curricula: **(a)** a constant $\alpha = \alpha_{\text{demo}} = \alpha_{\text{test}} = 0.3$ (which is also the one reported in §4.3), **(b)** a curriculum with $\alpha_{\text{demo}} = 1$, and **(c)** a scheduler which linearly decays α_{demo} from 1.0 to 0.3 over the various in-context examples. Our fourth approach **(d)** uses instead Dot-by-dot CoT (Pfau et al. [2024]) zero-shot. All experiments fail to bring mean $p(\text{correct})$ beyond 0.5, and no one shows an increasing of performances while increasing the in-context examples.

E.1 Steering vector extraction

For each concept c, we instantiate the 20 positive templates below with c and pair them (by position) with the 20 concept-agnostic negative templates. Below we details the concept pool, the templates for the positive vectors, and the templates for the negative ones.

Concept pool.

```
love, anger, fear, joy, sadness, disgust, science, politics,
religion, sports, music, cooking, honesty, creativity,
intelligence, kindness, freedom, justice, power, beauty, truth,
ocean, mountain, city, forest, fire, space, car, airplane,
boat, bicycle, truck, ship, motorcycle, helicopter, dog, cat,
fish, spider, horse, bird, snake, rabbit, pain, skill, queen,
rage, law, charm, stone, light, peace, film, lake, bread,
pizza, rice, soup, cake, pie, coffee, tea
```

Positive templates.

```
 1. Tell me about {concept}.
 2. Describe {concept} in detail.
 3. What comes to mind when you think of {concept}?
 4. Write a paragraph about {concept}.
 5. Explain the meaning of {concept}.
 6. Share your thoughts on {concept}.
 7. What is {concept}?
 8. How would you describe {concept} to someone?
 9. Discuss {concept} and its significance.
10. Give me an overview of {concept}.
```

```
11. Tell me everything you know about {concept}.
12. What are the key aspects of {concept}?
13. Reflect on {concept}.
14. What does {concept} mean to you?
15. Describe your understanding of {concept}.
16. Write about {concept} from your perspective.
17. Elaborate on {concept}.
18. What is the essence of {concept}?
19. Talk about {concept} in depth.
20. Summarize {concept} for me.
```

```
 1. Tell me about a random topic.
 2. Describe something interesting.
 3. What comes to mind right now?
 4. Write a paragraph about anything.
 5. Explain something to me.
 6. Share a thought.
 7. What is something you know well?
 8. How would you describe an everyday object?
 9. Discuss a topic of your choosing.
10. Give me an overview of a subject.
11. Tell me about something.
12. What are some interesting facts?
13. Reflect on an idea.
14. What matters to you?
15. Describe your understanding of the world.
16. Write about your perspective.
17. Elaborate on a theme.
18. What is the essence of things?
19. Talk about something in depth.
20. Summarize an idea for me.
```

E.2 Prompts and steering vectors for the the reasoning task task

This appendix lists the prompts and other relevant information for the reasoning task.

Country pool.

```
Europe:   Albania, Belarus, Croatia, Estonia, Greece, Iceland,
          Latvia, Lithuania, Luxembourg, Malta, Moldova, Romania
Asia:     Bhutan, Cambodia, Kazakhstan, Kyrgyzstan, Laos, Mongolia,
          Myanmar, Nepal, Oman, Tajikistan, Uzbekistan, Yemen
Africa:   Angola, Botswana, Chad, Madagascar, Mali, Mozambique,
          Namibia, Rwanda, Senegal, Tanzania, Uganda, Zambia
America:  Belize, Bolivia, Ecuador, Guyana, Haiti, Honduras,
          Jamaica, Nicaragua, Panama, Paraguay, Suriname, Uruguay
```

System prompts. The Dot-by-dot CoT experiments in Appendix D use the following system prompt:

```
You are participating in a word association experiment. Each input
triggers a specific response: Europe, Asia, Africa, or America. You
may write a sequence of dot characters as silent thinking time
before your final answer. After the dots, output exactly one of:
Europe, Asia, Africa, America.
```

F Prompt sensitivity for magnitude and layer detection task

When reporting the results for magnitude and layer detection task we computed the 95% confidence interval over all the $n = 300$ data points in each experiment. Here instead we compute mean

$p(\text{correct})$ for the $n = 30$ sample associated to each prompt in the experiment, obtaining 10 values, and then we show the $\pm 1\sigma$ interval over these 10 points, to make prompt sensitivity directly visible. Results are presented in Figures 11 and 12.

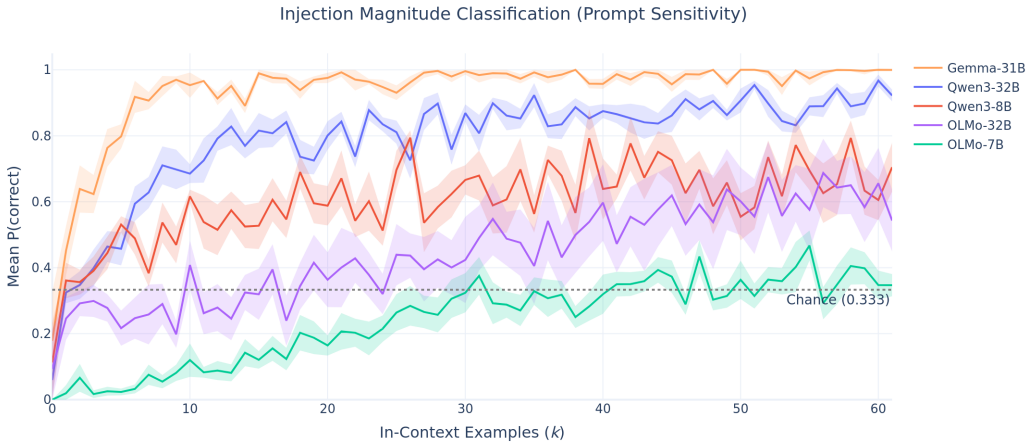


Figure 11: Magnitude detection: mean $P(\text{correct})$ *vs.* number of ICL examples, with the band showing $\pm 1\sigma$ across the 10 prompt-level means at each k.

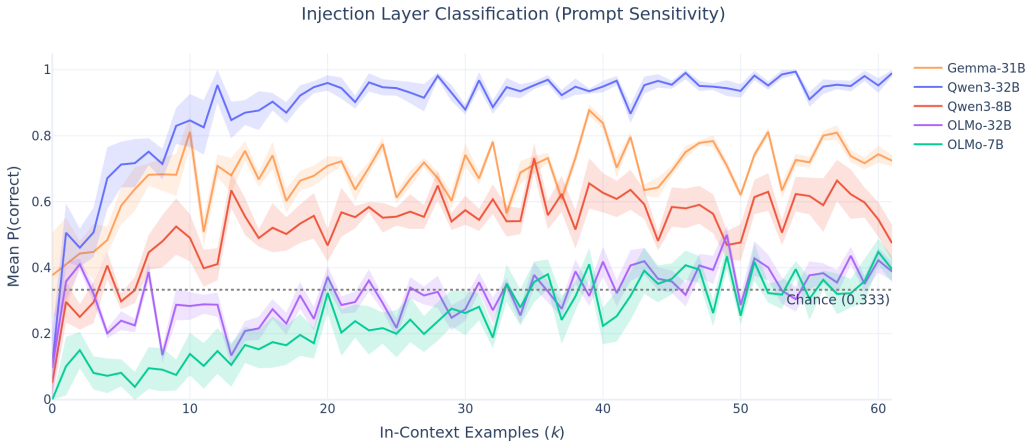


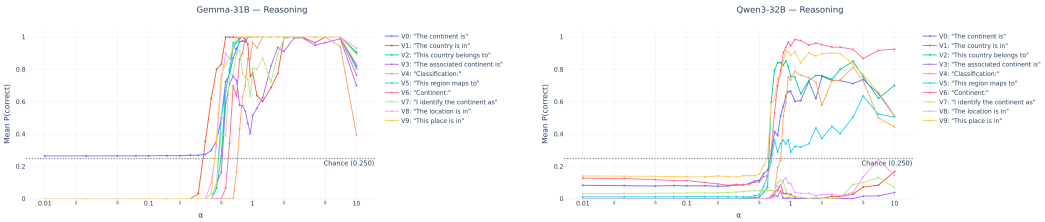
Figure 12: Layer detection: mean $P(\text{correct})$ vs. number of ICL examples, with the band showing $\pm 1\sigma$ across the 10 prompt-level means at each k.

G Prompt sensitivity for reasoning task

Figure 13 shows the various results, divided by prompt, for the reasoning task in §4.3. We note that the models are significantly more prompt sensitive than in the magnitude and layer experiments; indeed, paraphrases that are essentially synonymous to a human reader can shift mean $p(\text{correct})$ by tens of percentage points.

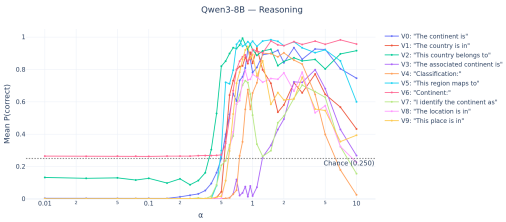
H Magnitude generalization (all models)

We report the magnitude detection generalization test for all the models in Figures 14, 15, 16, 17 and 18.

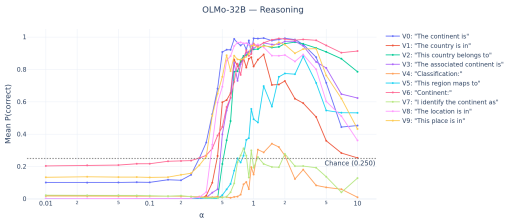


(a) Gemma4-31B

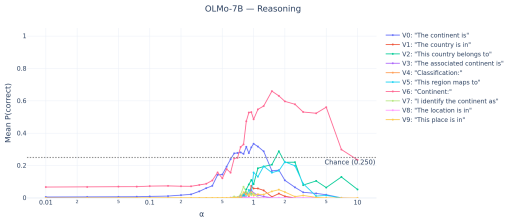
(b) Qwen3-32B



(c) Qwen3-8B



(d) Olmo3.1-32B



(e) Olmo3-7B

Figure 13: Results for the reasoning task: we report mean $p(\text{correct})$ *vs.* test injection magnitude α at $k = 0$ across 10 prompt variations, per model.

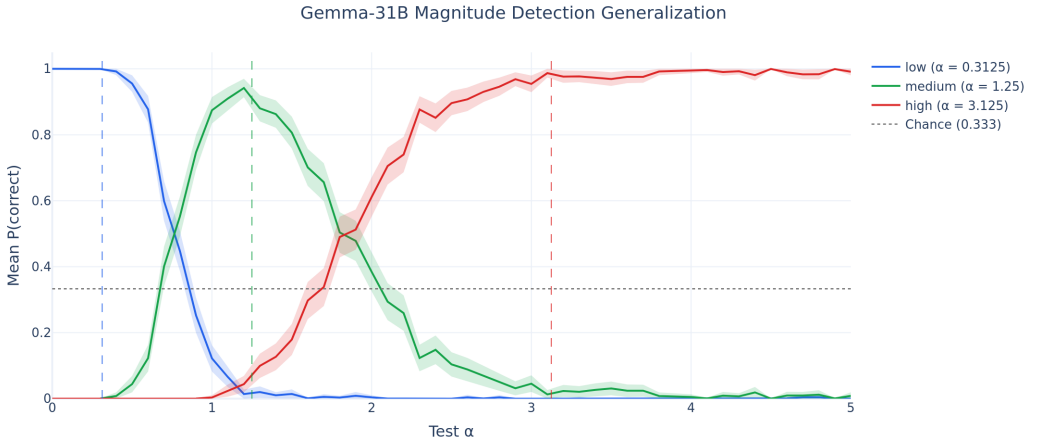


Figure 14: Generalization test for Gemma4-31B for on the magnitude detection task; we report mean $p(\text{correct})$for different test magnitudes α.

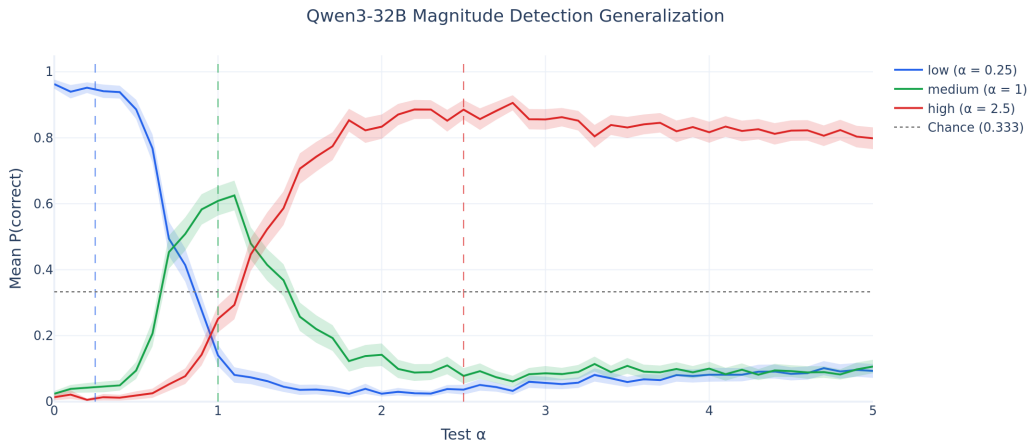


Figure 15: Generalization test for Qwen3-32B for on the magnitude detection task; we report mean $p(\text{correct})$ for different test magnitudes α.

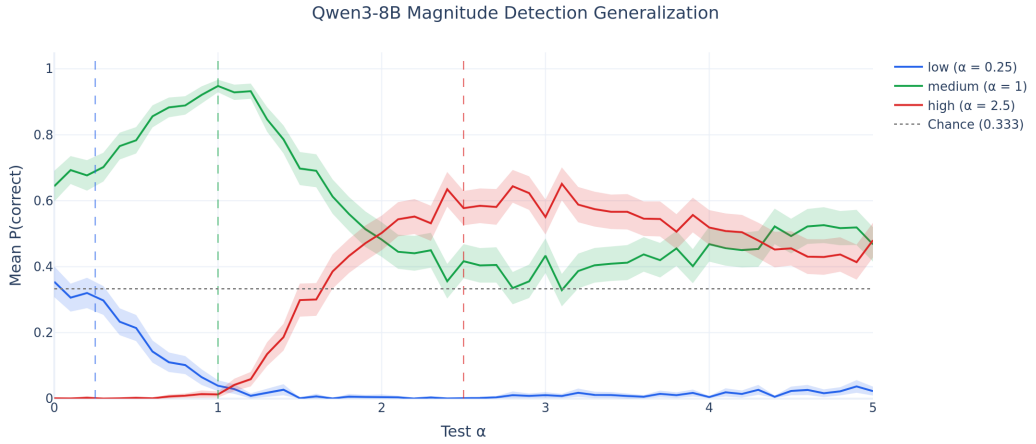


Figure 16: Generalization test for Qwen3-8B for on the magnitude detection task; we report mean $p(\text{correct})$ for different test magnitudes α.

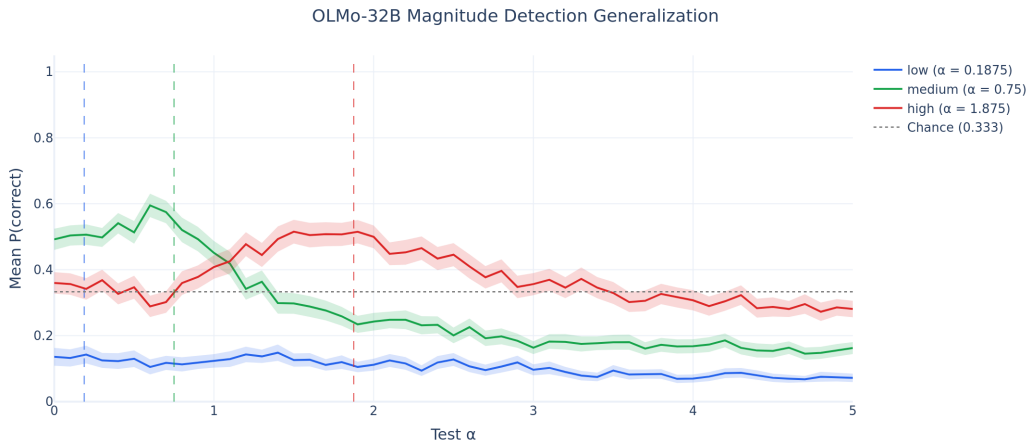


Figure 17: Generalization test for Olmo3.1-32B for on the magnitude detection task; we report mean $p(\text{correct})$ for different test magnitudes α.

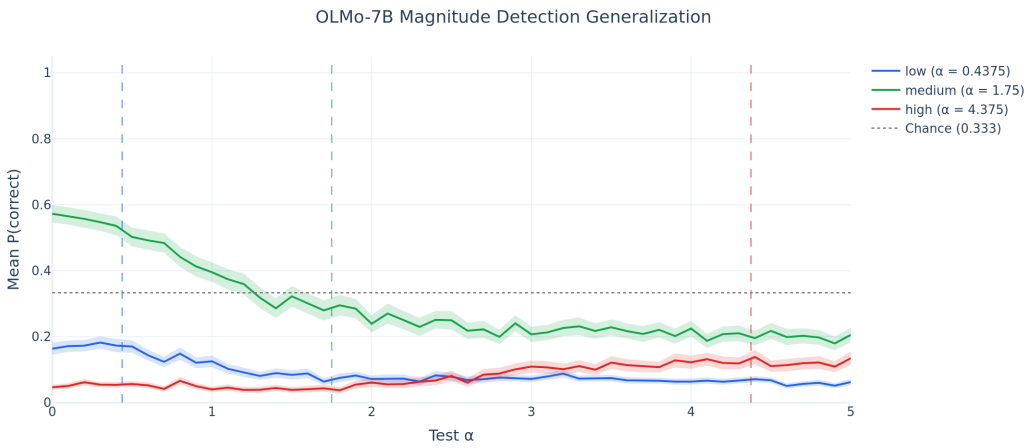


Figure 18: Generalization test for Olmo3-7B for on the magnitude detection task; we report mean $p(\text{correct})$ for different test magnitudes α.

I Layer generalization (all models)

We report the layer detection generalization test for all the models in Figures 19, 20, 21, 22 and 23.

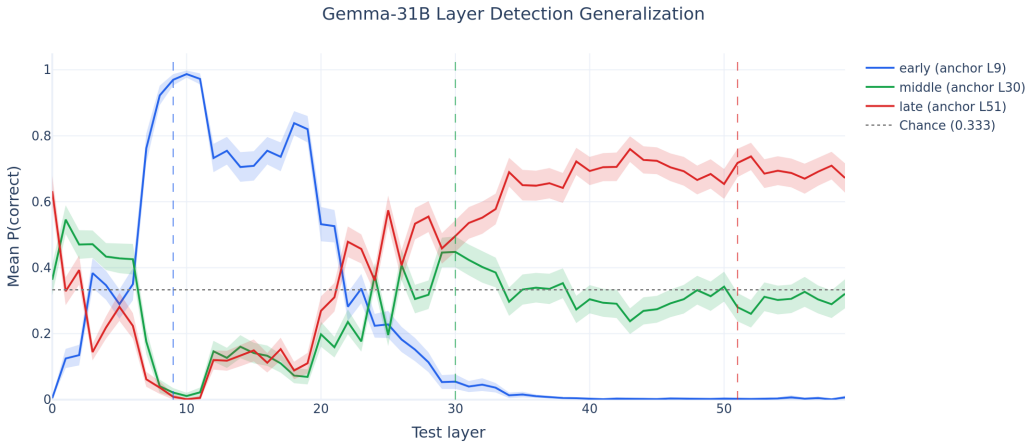


Figure 19: Generalization test for Gemma4-31B for on the layer detection task; we report mean $p(\text{correct})$ for different test layers.

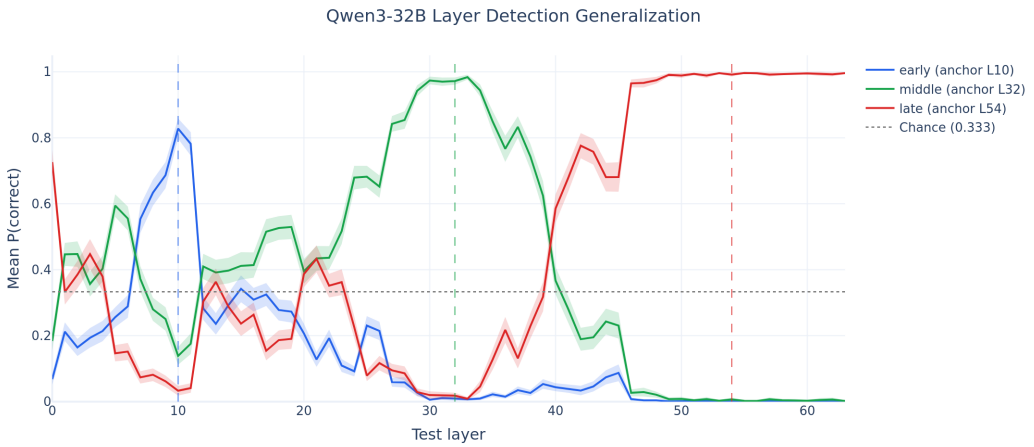


Figure 20: Generalization test for Qwen3-32B for on the layer detection task; we report mean $p(\text{correct})$ for different test layers.

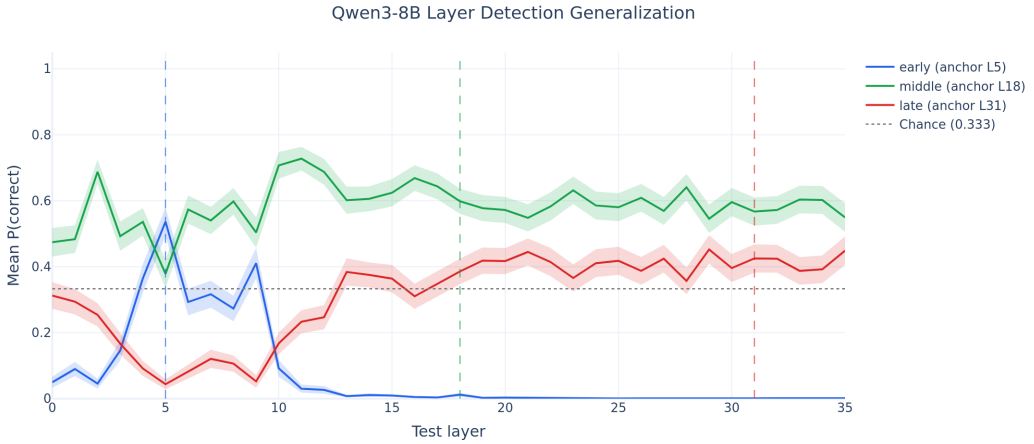


Figure 21: Generalization test for Qwen3-8B for on the layer detection task; we report mean $p(\text{correct})$ for different test layers.

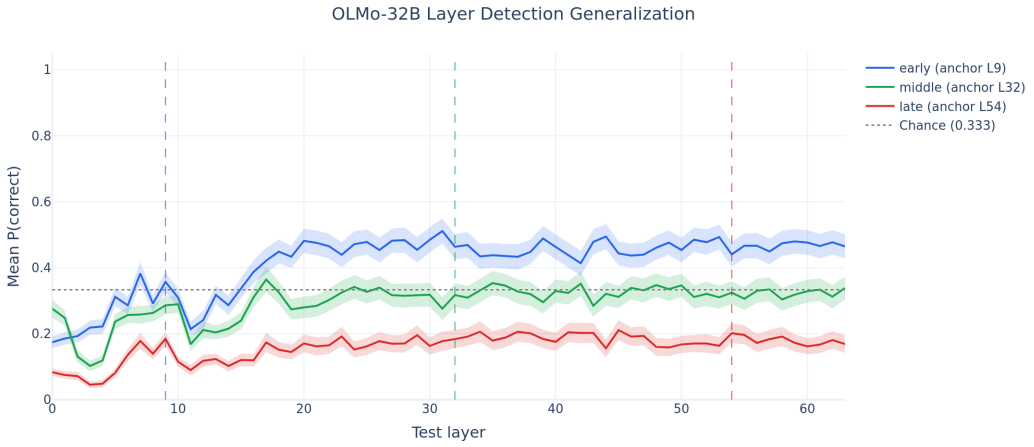


Figure 22: Generalization test for Olmo3.1-32B for on the layer detection task; we report mean $p(\text{correct})$ for different test layers.

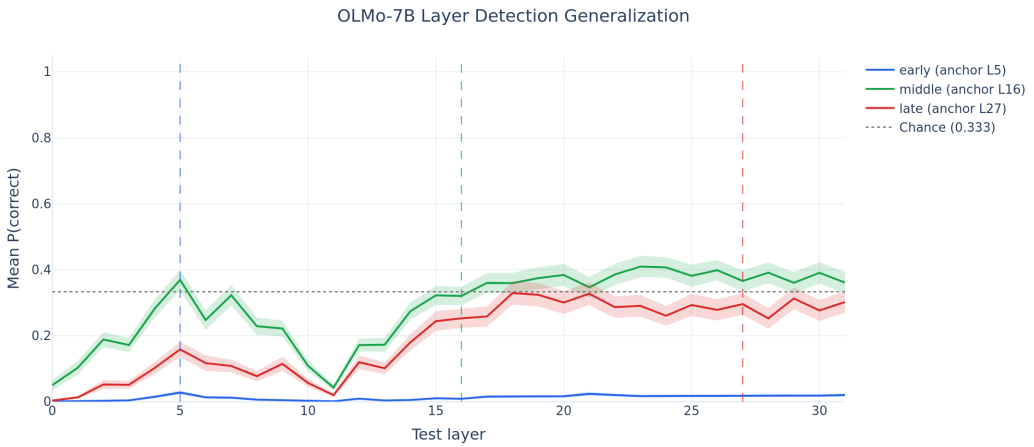


Figure 23: Generalization test for Olmo3-7B for on the layer detection task; we report mean $p(\text{correct})$ for different test layers.

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: All the claims in the abstract and introduction are introduced in the order they are later reported in the experiments section (§4).

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: Please see "Limitations" section, §5.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: Our work does not contain mathematical results (*e.g.*, theorems or algorithms). As such, the questions is not applicable.

Guidelines:

- The answer [N/A] 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 report the exact model names, where do download them, the exact prompts, and how to extract the concept vectors, with the exact numbers we use. For every tasks, we mention explicitly the necessary parameters to reproduce them (*e.g.*, the injection layer, the injection magnitude, *etc*). When not already explicit in the main text, we give proper referecne to the Appendix.

Guidelines:

- The answer [N/A] 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: We provide all code at this link.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: We only run experiments in inference mode and without generation. That is, as reportede explicitly in §3, our metrics are relative only to the logit distribution induced by the last token of our prompts. The necessary information to run the inference are provided both in the main text and in the Appendix.

   Guidelines:

   - The answer [N/A] 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: As explicitly states in §3, we primarily report 95% confidence intervals. For the sake of completeness, we also provide a standard-deviation analysis, relative the variations induced by prompt-sensitivity. See Appendix F and G.

   Guidelines:

   - The answer [N/A] 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 information about the computational resources used are reported at the beginning of the "Experiments" section, §4.

   Guidelines:

   - The answer [N/A] 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: There is no particular aspect of this question we feel we should highlight here.

   Guidelines:

   - The answer [N/A] 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: Both the introduction (§1) and the conclusions (§6) describe the positive and negative impacts we anticipate from increased "introspective" capabilities of language models, with appropriate citations.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We do not release any such data. As such, this question is not applicable.

Guidelines:

- The answer [N/A] 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: We do not rely on datasets. We rely on open-weight models legally downloaded from Huggingface, as explicitly credited in §3.

Guidelines:

- The answer [N/A] 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: [N/A]

    Justification: We do not create any new asset; as such, this question is not applicable.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: No crowdsourcing was used in the paper. As such, this question is not applicable.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: This question is not applicable.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: language models have been used solely to grammar check and minimally streamline the writing (*e.g.*, checking sentences), and as assistants in code writing.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.