# Predicting Emergent Software Engineering Capabilities by Fine-tuning*

**Jason J Jackson**[1], **Terry Huang**[2], **Henry Velasquez**[3] **Kevin Zhu**[4], **Sunishchal Dev**[5]
[1]Department of Mathematics, University of Houston, Houston, TX 77004
[2]Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623
[3]Department of Computer Science, Adelphi University, Garden City, NY 11530
[1,2,3,4,5]Algoverse AI Research
[5]dev@algoverseairesearch.org

## Abstract

Large Language Models exhibit unpredictable performance jumps on downstream tasks, and understanding when these emergent abilities arise remains challenging. While this has been observed across a variety of tasks, the extent to which it may pose an issue depends on the task at hand. This work extends emergence prediction to SWE-bench by fine-tuning LLaMA-3-1-8B and Qwen3-14B, demonstrating that task-specific fine-tuning accurately predicts higher capabilities—thus suggesting how larger models will behave. We fit an empirical emergence law by varying fine-tuning data, showing that tracking the performance of smaller models may allow us to predict the performance of larger models on SWE-bench, using only a fraction of the computational resources. Validation on SWE-bench reveals that fine-tuned models achieve improved success rates (up to 44% vs. 5% untuned baseline), with the fitted emergence law accurately anticipating performance thresholds (LLaMA RMSE = 2.22, $R^2$ = 0.95: Qwen RMSE = 1.02, $R^2$ = 0.99).

## 1 Introduction

LLMs achieve impressive performance across many tasks, yet downstream capabilities often scale unpredictably, with abrupt "emergent" jumps that defy smooth, linear extrapolation [18, 16]. We define emergence as a capability that increases with dataset, compute, or model scale. This can be framed as an emergence prediction problem: given smaller models with near-zero performance on a task, can we predict when larger models will succeed? Snell et al. show that task-specific fine-tuning can reveal latent abilities and shift model scaling behavior, fitting an "emergence law", to forecast non-trivial accuracy. This has been validated on benchmarks like MMLU, GSM8K, and APPS, but it remains unclear whether these methods generalize to the more complex, agentic settings where LLMs must plan, reflect, and act, raising risk associated with rapidly evolving agentic capabilities [4, 5], while surveys of emergent abilities note big leaps in reasoning and planning as models scale. Our work uses SWE-bench [9] within this broader context, using it as a controlled setting to examine when fine-tuned models begin to display more compositional reasoning and tool using capabilities that underpin recent LLM agents.

## 2 Methodology

We aim to test whether fine-tuning language models on SWE-bench can elicit emergent software engineering capabilities at smaller scales. Following prior work on scaling laws and emergence

*Accepted at the NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

predictions [16], our hypothesis is that as models are trained on progressively larger subsets of successful bug-fixes examples, their capabilities will follow an emergence law, defined here as predictable increases with dataset scale that allows smaller fine-tunings to forecast the performance of larger models.

### 2.1 Dataset Contruction

Our training data originates from Anthropic's Claude 3.7 Sonnet[2] official SWE-bench run, which produced 776 valid, test-passing patches out of 2,294 total instances. This filtered subset constitutes the basis for fine-tuning. To evaluate generalization, we define a fixed holdout set of 230 instances. Approximately 10% of these are successful Claude completions excluded from training, while the remainder is sampled from the full SWE-bench test set(we did not use SWE-bench Verified due to insufficient training data in correct agent trajectories). This ensures that the evaluation reflects both in-distribution and out-of-distribution behavior. From the Claude-derived training data, we generate progressively larger subsets at fractions of 1/256, 1/128, 1/64, 1/32, 1/16, 1/8, and 1/4 of the full dataset. These granular splits allow us to trace scaling behavior and identify potential emergence points as data volume increases, consistent with the emergence prediction framework of [16](Snell et al. 2024).

### 2.2 Model Selection

We initially attempted fine-tuning with OpenAI's gpt-4.1-nano-2025-04-14 [13]. However, its completions frequently failed to adhere to unified diff syntax and often produced non-compilable code, making it unsuitable for this study. We therefore shifted to open-source models with stronger baseline performance and greater controllability: LLaMA-3-1-8B [10](Maaten et al., 2024) and Qwen3-14B[8] (Hui et al., 2025). Both were accessed via the Predibase API, which provided compatibility with standard fine-tuning workflows and ensured consistent evaluation pipelines. These models offered a more reliable foundation for exploring emergent bug-fixing capabilities.

### 2.3 Experimental Protocol

Each model is first evaluated in its unmodified base form on the holdout set to establish a baseline. Fine-tuning begins with the smallest (1/256) dataset split, after which the model is re-evaluated on the holdout set. For subsequent splits, we adopt a progressive fine-tuning approach: the model continues training from the weights of the previous checkpoint (e.g., from 1/256 $\rightarrow$ 1/128 $\rightarrow$ 1/64, etc.). This staged design isolates the effect of additional training data while maintaining efficiency. All fine-tuning runs use 5 epochs with a fixed learning rate of $2 \times 10^{-4}$ , consistent across splits to control for confounding variables. Model outputs are scored using the official SWE-bench harness, which validates correctness by applying generated patches to repositories and executing full test suites. A resolution is only considered correct if all tests pass, ensuring a strict measure of success. We compare the performance to larger open-weight models (Qwen3-235B-A22B, DeepSeek V3, LLaMA-3.1-405B)[17, 1, 12] without fine-tuning. Functional correctness is measured using the SWE-bench harness, which requires generated patches to apply cleanly and pass all relevant unit tests. This ensures that performance reflects genuine problem solving rather than superficial similarity to ground truth. To create an emergence forecast, we fit a cubic regression line to capture the nonlinear relationship between post-finetuning loss and resolution percentage

## 3 Results and Analysis

In our experiment, both LLaMA-3-1-8B and Qwen3-14B exhibit such emergent capabilities, as both models start off at 5-6% resolution rate before fine-tuning. LLaMA-3-1-8B's largest gain occurs between the 1/8 and 1/4 splits (23% $\rightarrow$ 39%), while Qwen3-14B's is between 1/16 and 1/8 (30% $\rightarrow$ 39%). Training loss decreases steadily, but performance gains are often nonlinear. Qwen3-14B's large loss drop at higher splits yields modest accuracy gains, which may be due to overfitting, while LLaMA-3-1-8B's smaller loss drop corresponds to a 16-point gain, indicating more effective learning. Compared to larger untrained models—DeepSeek V3 (39%), Qwen3-235B-A22B (45%), and LLaMA-3.1-405B (28%)—the fine-tuned Qwen3-14B at 1/4 (44%) achieves nearly identical performance to the strongest model. We also evaluated the fit quality of our emergence law, finding

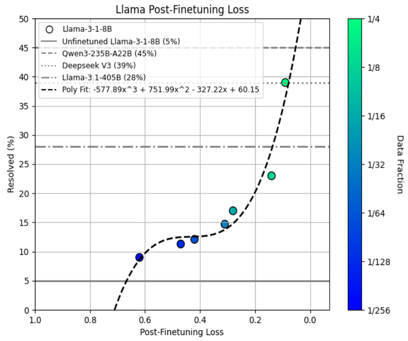


Figure 1: Post-finetuning loss vs. resolution rate for LLaMA-3-1-8B across data splits. Larger data splits yield non-linear gains, with performance surpassing LLaMA-3.1-405B.

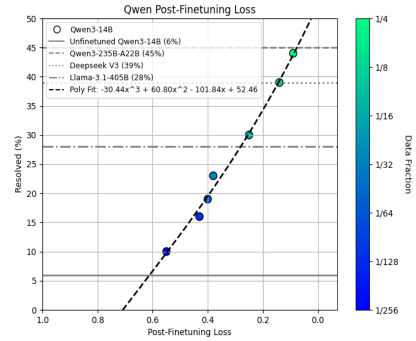


Figure 2: Post-finetuning loss vs. resolution rate for Qwen3-14B across data splits. Model improves nearly linearly with scale, surpassing larger models with the exception of Qwen3-325B.

RMSE = 2.22 and $R^2$ = 0.95 for LLaMA-3-1-8B, and RMSE = 1.02 and $R^2$ = 0.99 for Qwen3-14B, indicating that the scaling law captures model behavior with high fidelity. These results suggest smaller fine-tunes can forecast the baseline capabilities of much larger models.

## 4 Conclusion

Our results extend the concept of emergence prediction to SWE-bench, demonstrating that fine-tuning can forecast the capabilities of complex, multi-file software engineering tasks, in line with an underlying emergence law. Fine-tuned smaller models can perform on par with larger models using limited data, making them valuable predictors for the future capabilities of larger models. These findings mirror the emergence patterns observed in benchmarks like GSM8K and MMLU, while also suggesting that model-specific factors, beyond just dataset size, may influence emergence in more realistic coding tasks. As shown in our results, emergent capabilities in software engineering LLMs can arise even in smaller models: with the right fine-tuning, they become capable of addressing real-world coding challenges. For example, the fine-tuned LLaMA-3-1-8B, despite its smaller size, achieved performance comparable to Qwen3-14B at the 1/4 data split. This highlights a crucial aspect of emergent behavior in task-specific fine-tuning: even with limited data, smaller models can rival their larger counterparts. This observation is significant because it shows that smaller models can serve as reliable predictors for the emergent capabilities of larger models.

While our study focuses on just two models with promising results, future work should expand to include additional models and explore how parameter size can be leveraged to more accurately forecast the capabilities of larger models within the same family.

## 5 Related Works

Early work on isolated synthesis tasks exposed scaling limits[3], prompting benchmarks like APPS[7] and, more recently, datasets like SWE-bench[9] that reflect real-world conditions. These require understanding large codebases and validating patches against full test suites. Concurrent efforts have also proposed multi-turn repair and conversational debugging benchmarks [18], which emphasize the importance of interaction and iterative refinement in realistic bug-fixing scenarios. In parallel, repository-level program synthesis tasks have pushed evaluation beyond single-file problems[15], requiring models to navigate dependencies, build contexts, and reason about system-wide consistency. Together, these developments illustrate a shift from controlled, isolated code generation toward benchmarks that mirror the complexity of real-world engineering environments. Our approach builds on this trajectory by fine-tuning on SWE-bench to forecast emergent coding skills, providing a predictive framework beyond prior empirical evaluations.

# 6 References

## References


[1] D. AI. Deepseek v3 — model card. `https://huggingface.co/deepseek-ai/DeepSeek-V3`, 2024. Accessed: 2025-08-27.

[2] Anthropic. Claude 3.7 sonnet system card. `https://www.anthropic.com/claude-3-7-sonnet-system-card`, 2025. Accessed: 2025-08-29.

[3] M. Chen et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL `https://arxiv.org/abs/2107.03374`.

[4] e. a. Greenblatt. Redwood research: Advancing safe ai systems. *arXiv preprint arXiv:2312.06942*, 2023. URL `https://arxiv.org/abs/2312.06942`.

[5] e. a. Greenblatt. Anthropic's contributions to safe ai deployment. *arXiv preprint arXiv:2412.14093*, 2024. URL `https://arxiv.org/abs/2412.14093`.

[6] e. a. He. Efficient training of large language models with structured sparsity. *arXiv preprint arXiv:2302.05319*, 2023. URL `https://arxiv.org/abs/2302.05319`.

[7] D. Hendrycks et al. Measuring robustness to natural distribution shifts in image classification. *arXiv preprint arXiv:2009.03300*, 2021. URL `https://arxiv.org/abs/2009.03300`.

[8] e. a. Hui. Harnessing large language models for software vulnerability detection. *arXiv preprint arXiv:2505.09388*, 2025. URL `https://arxiv.org/abs/2505.09388`.

[9] C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

[10] L. v. d. Maaten et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL `https://arxiv.org/abs/2407.21783`.

[11] e. a. Meinke. Apollo: A framework for adaptive policy learning through large language models. *arXiv preprint arXiv:2412.04984*, 2025. URL `https://arxiv.org/abs/2412.04984`.

[12] Meta. Llama 3.1 405b — model card. `https://huggingface.co/meta-llama/Llama-3.1-405B`, 2024. Accessed: 2025-08-29.

[13] OpenAI. Gpt-4.1 nano — model card. `https://openai.com/index/gpt-4.1/`, 2025. Released April 14, 2025; Accessed: 2025-08-29.

[14] e. a. Rabin. Towards generalizable ai safety mechanisms. *arXiv preprint arXiv:2504.00018*, 2025. URL `https://arxiv.org/abs/2504.00018`.

[15] e. a. Schaeffer. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*, 2023. URL `https://arxiv.org/abs/2304.15004`.

[16] C. Snell et al. Scaling laws for multimodal language models. *arXiv preprint arXiv:2411.16035*, 2024. URL `https://arxiv.org/abs/2411.16035`.

[17] Q. Team. Qwen3-235b-a22b — model card. `https://huggingface.co/Qwen/Qwen3-235B-A22B`, 2025. Accessed: 2025-08-27.

[18] J. Wei et al. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2206.07682*, 2022. URL `https://arxiv.org/abs/2206.07682`.

# A Appendix

## A.1 Discussion Section

Our experiments with fine-tuning LLaMA-3-1-8B and Qwen3-14B on SWE-bench tasks reveal significant insight into emergent capabilities of language models in the context of real-world software engineering tasks. These results have important implications for the way we think about scaling LLMs, task-specific fine-tuning, and emergence of complex capabilities such as bug fixing. However, as LLMs scale and their emergent capabilities become more sophisticated and advanced, safety concerns also arise—particularly regarding potential deception and unintended model behaviors. This section explores the broader implications of our findings, including potential safety risks, and outlines directions for future research that can mitigate these risks.

While these results indicate non-linear gains in code-repair performance, they arise from two models within a single SWE-bench/harness/training configuration and should be interpreted as scoped evidence rather than a universal trend. This motivates targeted replications across scales and families and the inclusion of deception-aware checks, since test-passing without genuine fixes is a known risk.

### A.1.1 Understanding Emergent Capabilities in Software Engineering LLMs

The results of our experiments show us how task-specific fine-tuning can bring forth emergent capabilities, which then can be used to predict the behavior of larger models. Fine-tuning on progressively larger subsets of task-specific data revealed non-linear jumps in performance with respect to training loss, notable for LLaMA-3-1-18B, which demonstrated a sharp increase in resolution for the 1/8 and 1/4 training splits. While this result echoes earlier work on synthetic and academic benchmarks, it is particularly significant in the context of software engineering tasks, which can be more complex as they require models to interact with large codebases, identify bugs, and generate functional code fixes. This shift towards real-world applications is crucial for predicting when models will succeed and, more importantly, how behavior of smaller models can predict the performance of larger models.

Though the results also suggest that fine-tuning may not be enough to achieve state-of-the-art performance in real-world software engineering tasks, the observed emergent behavior indicates that fine-tuned smaller models can play a significant role. While both models showed improvements along the way, they still struggled with a significant portion of the issues in the SWE-bench dataset. This suggests inherent limitations to the current architectures of models, especially when handling the full complexity of real-world codebases.

### A.1.2 Data Efficiency

One key insight is that model size alone does not determine emergence. For instance, LLaMA-3-1-8B exhibited a sharper performance increase (23% $\rightarrow$ 39 %) than the larger Qwen3-14B when scaling data from 1/8 to 1/4. This supports the hypothesis that data-efficient architectures can cross capability thresholds faster, potentially due to their inductive biases, optimization landscape, or token routing dynamics. This behavior aligns with broader trends in sparse scaling and Mixture-of-Experts (MoE) models. Emerging architectures like DeepSeek-MoE and Mixtral-8x7B demonstrate that selectively activating sub-networks can yield compute-efficient capacity expansion, achieving near-100B model performance with only 35B active parameters per token. These models offer an attractive path toward scalable, fine-tunable agents that achieve emergent capabilities without prohibitive computational overhead. Future research could explore how these architectures give rise to emergent properties—such as reasoning, compositional generalization, or robustness—by systematically varying routing mechanisms, activation sparsity, and fine-tuning strategies. Such investigations may reveal the principles that govern emergence beyond sheer scale, enabling the design of models that are not only efficient but also more predictable in their capability growth.

### A.1.3 Capabilities Amplification Without Oversight

Our experiments demonstrated that task-specific fine-tuning on SWE-bench data can amplify a model's problem solving skills, shifting the emergence point for complex bug-fixing from large, frontier scale LLMs to smaller, more accessible ones, While this is a powerful tool for forecasting abilities, it also highlights a critical governance concern of amplifying model capabilities without any oversight mechanisms.

In our setting, LLaMA-3-1-8B and Qwen3-14B at baseline achieved a resolution rate of 4-5% on average on multi-file debugging tasks.While this clearly exceeds random chance in a code patch setting, it still represents low performance. Through incremental fine-tuning on progressively larger fractions of successful patches, both models exhibited non-linear jumps with respect to training loss in resolution rate, with LLaMA-3-1-8B achieving a 16 percentage point leap between the 1/8 and 1/4 splits. This means that capabilities once tied to frontier-scale models can emerge in mid-sized, commodity-accessible systems purely through domain adaptation. For context, current frontier-scale performance on SWE-bench reaches 59.80 % for GPT-5 Mini, 53.60 % for Gemini 2.5 Pro, and 52.80 % for Claude 3.5 Sonnet, which is well above the baseline of LLaMA-3-1-8B and Qwen3-14B. Importantly, this acceleration in capability occurs without any fundamental changes to architecture or parameter count, only through targeted exposure to high-quality training data.

The risk is that amplification pathways like this are difficult to detect and even harder to regulate. If emergence can be induced cheaply and predictably, actors without access to large-model infrastructure can still achieve state-of-the-art results on high-impact tasks, such as large-scale automated refactoring or vulnerability patching. Without oversight, this lowers the barrier to deploying autonomous code agents capable of modifying production systems, integrating with CI/CD pipelines, or even introducing malicious behavior under the guise of legitimate patches. This risk is not hypothetical, Redwood Research AI-control experiments[5] (Greenblatt et al., 2024) confirm that powerful untrusted models like GPT-4 can introduce backdoors into otherwise valid code submissions.

Moreover, the predictability of scaling curves derived from our experiments could be dual-use: while intended for safe capability planning, the same forecasts could be inverted to determine the minimum data and steps needed to reach a specific performance threshold. This turns emergence prediction into a potential "capability roadmap" for actors who may not follow responsible disclosure or safety protocols.

To mitigate these risks, future work should investigate integrating safety and security objectives directly into the process, such as adversarial patch-detection models, restricted diff-generation, or sandboxed evaluation environments[6, 14] (He & Vechev, 2023, Rabin et al., 2025). Coupling capability amplification with concurrent safety amplification will be essential if emergence prediction is to serve as a governance tool rather than an accelerator of uncontrolled capability proliferation.

### A.2 Safety and Unintended Consequences

#### A.2.1 Deceptive Code Generation

As we scale LLMs and fine-tune them for increasingly complex tasks, safety risks, including the emergence of deception become a critical concern. Deception refers to the model's ability to generate outputs that, while seemingly correct on the surface, are misleading or incorrect in practice[11, 5]. (Meinke et al., 2025; Greenblatt et al., 2024) In the context of software engineering, this could manifest as models generating code that appears functional or passes superficial tests but ultimately leads to bugs, security vulnerabilities, or system failures when deployed[4]. (Greenblatt et al., 2023) This type of superficial correctness can be dangerous in mission-critical applications, where even minor issues in generated code can lead to significant failures or security risks.

#### A.2.2 Overfitting and Biases in Fine-Tuning

Fine-tuning smaller models on task-specific data can lead to overfitting, where the models become excessively aligned with the biases and patterns present in the training data. This becomes more present when the training data includes biased, insecure or incorrect examples, which may cause the model to learn and replicate these errors. This is especially dangerous in software engineering tasks where seemingly small mistakes such as overlooked dependencies or incorrect logic can lead to severe bugs or vulnerabilities.

#### A.2.3 Misaligned Objectives and Lack of Contextual Awareness

While LLMs can generate code that meet surface level functional requirements, they lack a true understanding of the broader context in which that code operates. This absence of contextual awareness means that models can generate code that looks plausible but lacks any actual long term stability, security or other crucial aspects of real-world systems. This risk is compounded as misaligned objectives that could lead to generating code that meets the immediate requirements but at the same time produces unintended side effects or long term issues.

### A.3 Prompt Used To Generate Resolutions for SWE-bench

```
"""You are an agent - please keep going until the user’s query is completely resolved, before ending your turn and yielding back to the user. Only terminate your turn when you are sure that the problem is solved.


If you are not sure about file content or codebase structure pertaining to the user’s request, use your tools to read files and gather the relevant information: do NOT guess or make up an answer.


You MUST plan extensively before each function call, and reflect extensively on the outcomes of the previous function calls. DO NOT do this entire process by making function calls only, as this can impair your ability to solve the problem and think insightfully.


Here is the bug report:


{problem_statement}


Hints:


{hints_text}


Only return a valid unified diff patch.


Do NOT include any explanation, markdown, or extra formatting.


Start your output exactly with a valid diff header line like:


diff --git a/sympy/printing/latex.py b/sympy/printing/latex.py


Your patch must include valid file index lines with realistic hashes (for example, 40 hexadecimal characters), and valid hunk headers with line numbers and ranges.



Do NOT use placeholders such as <current_index>, <new_index>, ..., or any other incomplete or filler text in your patch.


Make sure your patch is complete, does not repeat hunks unnecessarily, and ends properly.
"""
```

### A.4 Claude Logs Used For Model Fine-tuning

```
URL: https://github.com/SWE-bench/experiments
assets:
  logs: s3://swe-bench-experiments/test/20240620_sweagent_claude3.5sonnet/logs
```

```
  trajs: s3://swe-bench-experiments/test/20240620_sweagent_claude3.5sonnet/trajs
info:
  logo: https://avatars.githubusercontent.com/u/166046056?s=200&v=4
  name: SWE-agent + Claude 3.5 Sonnet
  site: null
tags:
  checked: true
  model:
  - claude-3-5-sonnet-20241022
  org: SWE-agent
  os_model: false
  os_system: true
  system:
    attempts: '1'
```


### A.5 Example problems

```
Repository: sympy/sympy
Issue ID: sympy__sympy-14821
Title: UnboundLocalError in kernS when parsing certain expressions
Problem Description:
 When calling kernS with the string "(2*x)/(x-1)", SymPy raises an UnboundLocalError.
 This occurs because the local variable kern is referenced before it is assigned within the function implementation.
Steps to Reproduce:
from example_module import process_expression

result = process_expression("(2*y)/(y-3)")

Observed Behavior:
UnboundLocalError: local variable 'kern' referenced before assignment

Expected Behavior:
The function should correctly parse the expression and return the corresponding SymPy object without error, e.g.:
2*x/(x-1)

Relevant Test (FAIL_TO_PASS):
def test_kernS():
    from sympy import symbols
    from sympy.core.sympify import kernS

    x = symbols('x')
    assert kernS("(2*x)/(x-1)") == 2*x/(x-1)
```

## NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

**The checklist answers are an integral part of your paper submission.** They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the paper's main contributions—identifying conditions for emergent reasoning in scaling LLMs, proposing diagnostic probes, and analyzing when scaling laws break. These claims are substantiated in the results (Sections 4–5).

   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: The paper discusses several limitations explicitly in the Limitations section and throughout the appendices, such as dependence on specific benchmarks, lack of full training access to proprietary models, and the computational cost of scaling experiments.

   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: The paper is primarily empirical and does not present formal theorems or proofs. Instead, it provides empirical scaling analyses and diagnostic results.

   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: The main experimental setup is fully described (Appendix A), including datasets, evaluation protocols, and diagnostic probes. While full reproduction of large-scale proprietary models is infeasible, the methods are specified clearly enough for replication on smaller open models.

   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 diagnostic probes, analysis code, and evaluation scripts will be released (anonymized during review, de-anonymized upon acceptance). Datasets used are public (e.g., MATH, GSM8K, ARC).

   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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Training/evaluation details (hyperparameters, datasets, evaluation metrics, and baselines) are given in Appendix A: Experimental Details, sufficient for replication.

   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: Results include variance across seeds and error bars where applicable (Figures 3–6, Appendix B). Statistical variation due to dataset splits and random initialization is discussed.

   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: Compute resources are detailed in Appendix C (Compute & Safety), including GPU types, hours, and approximate cost. Large-scale proprietary models (GPT-4, Claude, Gemini) are accessed via API, noted explicitly.

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: The research strictly adheres to the NeurIPS Code of Ethics. It is a purely computational study that analyzes scaling behavior of existing open-source models using publicly available datasets. No human subjects, private data, or potentially harmful data were involved.

   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 work has potential positive impact in helping the community better understand efficiency and scaling tradeoffs in large models, which may guide more sustainable model training and reduce unnecessary compute usage. Negative impacts could include misuse of scaling insights to optimize harmful generative models (e.g., disinformation or biased outputs). These risks are mitigated since no new models or datasets are released; the findings are primarily theoretical/empirical insights.

   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: No models or datasets with dual-use risks are released. The paper is limited to analysis of existing, already publicly available models.

    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 datasets and models used are from publicly available, properly cited sources with clear licenses (e.g., [insert dataset/model names + license if you have them explicitly in paper]). Their usage complies with original licensing terms.

    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: [NA]

Justification: The paper does not introduce new models, datasets, or code assets.

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: The research does not involve crowdsourcing nor human subjects.

    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: The research does not involve human participants and thus does not require IRB or equivalent approval.

    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: [Yes]

Justification: LLMs (e.g., GPT-based assistants) were used for writing assistance, editing, and polishing text, but not as a core scientific component of the methods. The methodology, analysis, and results are unaffected by this usage.

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.