Understanding the Role of Training Data in Test-Time Scaling

Adel Javanmard[1,3], Baharan Mirzasoleiman[2,3], Vahab Mirrokni[3]

1: University of Southern California (USC), 2: University of California Los Angeles (UCLA), 3: Google Research

Introduction

- Test-time scaling improves large language models (LLMs) by allocating extra compute to generate longer Chains-of-Thoughts (CoTs).
- Longer CoTs allow models to break down complex problems, evaluate multiple options, and backtrack to correct mistakes.
- Despite strong reasoning performance in models like OpenAI's o1 and DeepSeek R1, the exact conditions in the training data that enable long CoTs remain unclear.
- This paper theoretically investigates whether increasing test-time compute always improves reasoning and how it affects training-time requirements.

Theoretical Framework

- The study analyzes in-context learning (ICL) for linear regression using a transformer with a single linear self-attention (LSA) layer trained via gradient descent.

Data: $y_{\tau,i} = \langle w_\tau, x_{\tau,i} \rangle$ (each prompt has its own w_τ)

- **During training**, the transformer engages in direct in-context learning to predict the linear weight vector from a sequence of input prompts.

(No chain-of-thought)

$$E_\tau = \begin{bmatrix} x_{\tau,1} & \cdots & x_{\tau,n} & 0 \\ y_{\tau,1} & \cdots & y_{\tau,n} & 0 \\ 0 & \cdots & 0 & \hat{w}_0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} := \begin{bmatrix} X_\tau & 0 \\ y_\tau & 0 \\ 0_{d\times n} & \hat{w}_0 \\ 0_{1\times n} & 1 \end{bmatrix}$$

(embedding of prompt τ)

Linear-Self Attention (LSA)

$$f_{\mathrm{LSA}}(E;\theta) = E + VE \cdot \frac{E^\top W E}{\rho} \qquad \hat{w}_\tau = f_{\mathrm{LSA}}(E_\tau;\theta)_{[d+2:2d+1,-1]}$$

- **During test time,** CoT prompting is used to generate intermediate reasoning steps before arriving at a final prediction.

$$E_i = \begin{bmatrix} x_1 & \cdots & x_m & 0 & 0 & \dots & 0 \\ y_1 & \cdots & y_m & 0 & 0 & \dots & 0 \\ 0 & \cdots & 0 & w_0 & w_1 & \dots & w_i \\ 0 & \cdots & 0 & 1 & 1 & \dots & 1 \end{bmatrix}$$

(Test-time chain of thought)

$$w_i := f_{\mathrm{LSA}}(E_{i-1})_{[d+2:2d+1,-1]}$$

- With CoT prompting at test time, the transformer effectively implements a multi-step (pseudo-) Newton's method for loss optimization. We show that assuming $x_{\tau,i} \sim \mathrm{N}(0,\Lambda)$ the updates are given by:

$$w_{i+1} = w_i - \frac{1}{m}\Gamma^{-1} X_{\mathrm{test}} X_{\mathrm{test}}^\top (w_i - w_{\mathrm{test}}) \qquad \Gamma := \left(1+\frac{1}{n}\right)\Lambda + \frac{1}{n}\mathrm{tr}(\Lambda) I_d \in \mathbb{R}^{d\times d}$$

Task hardness

- A task is defined by the covariance matrix of its features, denoted as Λ
- Each eigenvector of Λ represents a specific skill needed to solve the task, and the corresponding eigenvalues indicate the strength of those skills.
- Task hardness is mathematically formulated as the ratio of the trace to the smallest eigenvalue:

$$\mathrm{Hard}(\Lambda) := \frac{\mathrm{tr}(\Lambda)}{\lambda_{\min}(\Lambda)}$$

Key Findings on Test-Time Scaling

- **The Compute Trade-off:** For any fixed test error, increasing test-time compute allows for a reduction in the required context length (number of in-context examples) in training prompts.
- **The "Overthinking" Hazard:** If the skills required for a downstream task are not sufficiently represented in the training data, increasing test-time compute can actually harm performance.
- **Optimal Training Strategy:** Formulating optimal task selection as a quadratic optimization problem reveals that training on a diverse, relevant, and sufficiently hard set of tasks yields the best test-time scaling performance

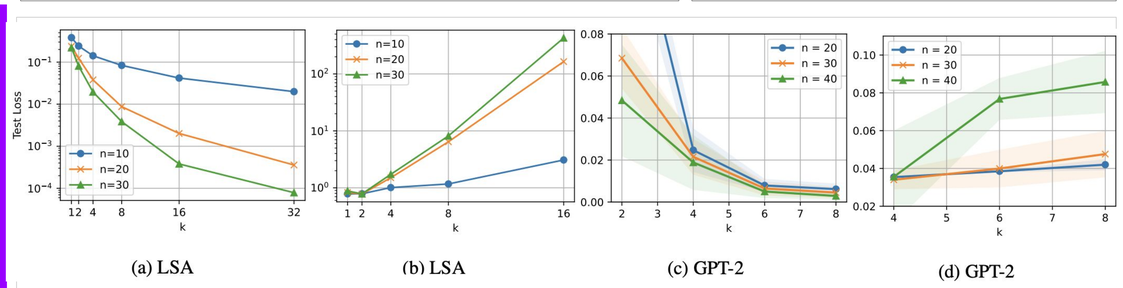
Task selection: Given T tasks, we show that the optimal way to set the task selection probabilities is by solving the following quadratic optimization problem:

$$\min_{\{\pi_\ell\}_{\ell\in[T]}} \left\| I - \Sigma^{-1} \sum_{\ell\in[T]} \Lambda_\ell \pi_\ell \right\|_F^2$$

$$\text{subject to} \sum_{\ell\in[T]} \pi_\ell = 1, \quad \pi_\ell \geq 0, \ \forall \ell \in [T]$$

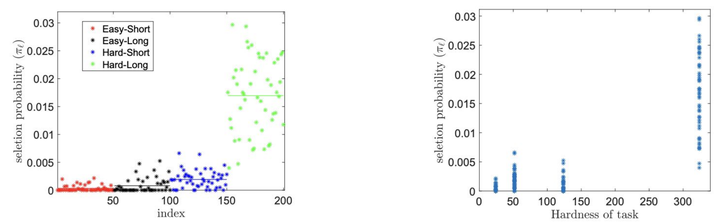
This emphasizes the importance of balancing training corpora with sufficient coverage of topics matched to downstream tasks, as imbalanced data composition can impair generalization

Experiments



(a) LSA (b) LSA (c) GPT-2 (d) GPT-2

More test-time compute reduces training-time requirements for (a) one-layer transformer and (c) GPT-2. But, insufficient task coverage in training data makes longer CoTs harmful for (b) one-layer transformer and (d) GPT-2. For GPT-2, the errorbars are std of 10 runs. For LSA, std is negligible as we start from the fixed initialization in Eq. (3.6). Same value for n is used in training and test.



(a) Selection probabilities for different task types (b) Selection probabilities vs. task hardness

Task selection in a multi-task setup (a) Each color corresponds to a task type with solid lines indicating the average selection probability per type. As we observe harder and more diverse tasks receive higher selection probabilities, while easier, more concentrated tasks are weighted lower (b) Task selection probabilities versus task hardness. As we see harder task are favored in the selection.

Fine-tuning Qwen2.5-7B Instruct (Base) on GCD (Qwen-GCD) and Poly (Qwen-Poly) subsets of OMEGA dataset (Sun et al. 2025)

CoT length	[0, 1k)	[1k, 2k]
Qwen-Base	30.39% (30% data)	27.2% (70% data)
Qwen-GCD	75% (15% data)	38.4% (85% data)
Qwen-Poly	29% (32% data)	20.83% (68% data)

% in () shows fraction of test data with the corresponding test-time CoT length. Shorter CoTs (0-1k) considerably improves the performance of Qwen-GCD (75% versus 30.39%) and slightly harms the performance of Qwen-Poly (29% versus 30.39%). Longer CoTs improve the performance of Qwen-GCD (38.4% vs 27.2%) & significantly harm the performance of Qwen-Poly (20.83% vs 27.2%).