

Chain-of-Thought Reasoning in Zen Models: Verified CoT Training with Step-Level Supervision

Technical Report v2025.05

Zen LM Research Team
research@zenlm.org

May 2025

Abstract

Chain-of-thought (CoT) reasoning enables language models to solve complex multi-step problems by generating explicit intermediate reasoning steps before producing a final answer. We present a comprehensive training methodology for CoT in the Zen MoDE model family, combining scratchpad pretraining, process reward models (PRMs) for step-level supervision, and a novel verified CoT training objective that assigns credit to individual reasoning steps based on their logical necessity for the correct answer. Our approach achieves 94.1% on GSM8K and 72.4% on the MATH benchmark with Zen MoDE-72B, with a step-level analysis showing that verified CoT substantially reduces *reasoning shortcuts* — chains that arrive at the correct answer through flawed intermediate steps. We further provide a stratified analysis by reasoning length (number of steps) and problem difficulty, characterizing where CoT provides the largest gains.

Contents

1	Introduction	3
1.1	Summary of Contributions	3
2	Background	3
2.1	Chain-of-Thought Prompting	3
2.2	Process vs. Outcome Reward Models	3
2.3	Reasoning Shortcuts	4
3	CoT Training Methodology	4
3.1	Scratchpad Pretraining	4
3.2	Process Reward Model Architecture	4
3.3	Verified CoT Training Objective	4
3.4	Step-Level Credit Assignment	5
4	Experiments	5
4.1	Benchmark Descriptions	5
4.2	Main Results	5
4.3	CoT vs. No CoT	5
4.4	Stratified Analysis by Reasoning Length	6
4.5	MATH Subject Breakdown	6
4.6	Reasoning Shortcut Analysis	6
4.7	Effect of γ (Step vs. Outcome Balance)	6

5	Discussion	7
5.1	Why Step-Level Supervision Matters	7
5.2	Scalability of PRM Training	7
5.3	Compute Budget for CoT	7
6	Conclusion	7

1 Introduction

Mathematical and logical reasoning requires multi-step inference that exceeds the capacity of a single forward pass for current transformer architectures. Chain-of-thought (CoT) prompting [1] elicits intermediate steps by few-shot demonstration, while zero-shot CoT [2] uses the prompt “Let’s think step by step”. Both approaches improve performance substantially on arithmetic, symbolic, and commonsense reasoning tasks, but rely on model capabilities baked in during pretraining or fine-tuning.

The key insight of our work is that not all chains of thought are equal: a model may arrive at the correct final answer via a logically flawed intermediate path (a *reasoning shortcut*), which does not generalize to unseen problems and does not benefit from increased chain length. Standard CoT training using outcome supervision (reward only for correct final answers) cannot distinguish correct reasoning from reasoning shortcuts.

We address this through *verified CoT training*, which applies step-level supervision using a process reward model (PRM) trained to identify logically necessary and sufficient reasoning steps. The PRM rewards each step based on its logical contribution to the final answer, as verified by a symbolic verifier where available (mathematics) or a judge model where not (general reasoning).

1.1 Summary of Contributions

1. A CoT scratchpad pretraining protocol that teaches models the format and conventions of structured intermediate reasoning at scale.
2. A process reward model (PRM) trained on step-annotated mathematics problems, extended to general reasoning via automated step verification.
3. A verified CoT training objective combining outcome rewards and step-level PRM scores in a policy gradient framework.
4. State-of-the-art results on GSM8K (94.1%) and MATH (72.4%), with a stratified analysis showing CoT gains as a function of reasoning length and problem difficulty.
5. A reasoning shortcut analysis showing that verified CoT reduces shortcut incidence by 71% compared to outcome-supervised CoT.

2 Background

2.1 Chain-of-Thought Prompting

Wei et al. [1] showed that few-shot examples with intermediate reasoning steps substantially improve multi-step reasoning performance. Let x denote a problem, $r = (r_1, r_2, \dots, r_L)$ a reasoning chain of L steps, and a the final answer. CoT models $p(r, a \mid x)$ jointly, with the final answer conditioned on the generated reasoning.

2.2 Process vs. Outcome Reward Models

Outcome reward models (ORMs) assign a scalar reward to the final answer: $R_{\text{ORM}}(a)$. Process reward models (PRMs) assign step-level rewards: $R_{\text{PRM}} = \{r_\ell\}_{\ell=1}^L$, where $r_\ell \in [-1, 1]$ scores the correctness and relevance of step ℓ . Lightman et al. [3] demonstrated that PRMs outperform ORMs on mathematical reasoning. Our work extends PRMs to general reasoning domains.

2.3 Reasoning Shortcuts

A *reasoning shortcut* is a chain $(r_1, \dots, r_L, a)$ where a is correct but the chain contains at least one step r_ℓ that is logically incorrect or irrelevant, meaning the correct answer could not be reliably derived from the steps alone. Shortcuts undermine generalization: models exploiting shortcuts fail on out-of-distribution problems that require the intermediate steps to be logically sound.

3 CoT Training Methodology

3.1 Scratchpad Pretraining

We pretrain on a scratchpad corpus of 50B tokens extracted from mathematical textbooks, competition problem sets, and scientific papers. Each document is processed to extract explicit step-by-step derivations in a standardized scratchpad format:

```
<think>
Step 1: [First reasoning step]
Step 2: [Second reasoning step]
...
Step L: [Final derivation]
</think>
<answer>[Final answer]</answer>
```

Where explicit step structure is not present, we use a LM-based step extractor (fine-tuned on annotated examples) to segment existing derivations into steps and format them. Scratchpad pretraining is performed for 5,000 gradient steps on top of the base model checkpoint.

3.2 Process Reward Model Architecture

Our PRM is a separate transformer model (identical architecture to the policy but smaller: 1/8th the parameter count) that takes as input a problem x and a partial chain $(r_1, \dots, r_\ell)$ and outputs a step quality score $q_\ell \in [-1, 1]$:

$$q_\ell = \text{PRM}(x, r_1, \dots, r_\ell) \quad (1)$$

The PRM is trained on a step-annotated dataset of 500K mathematical problems with human-verified step labels. Label $y_\ell \in \{-1, 0, +1\}$ indicates: incorrect/ irrelevant (-1), neutral (0), or correct and necessary ($+1$). Training loss:

$$\mathcal{L}_{\text{PRM}} = -\frac{1}{L} \sum_{\ell=1}^L y_\ell \log \sigma(q_\ell) + (1 - y_\ell^2) \log(1 - \sigma^2(q_\ell)) \quad (2)$$

For domains without human-annotated step labels (general reasoning), we use an automated verifier: a symbolic solver for mathematics and a judge LM for natural language reasoning. The judge LM is prompted to assess whether each step is logically entailed by prior steps and the problem statement.

3.3 Verified CoT Training Objective

The verified CoT training objective combines outcome reward R_{out} with step-level PRM scores:

$$R_{\text{VCoT}}(r, a \mid x) = R_{\text{out}}(a) + \gamma \sum_{\ell=1}^L q_\ell(r_1, \dots, r_\ell \mid x) \quad (3)$$

where $\gamma \in [0, 1]$ balances step-level and outcome credit. The policy is optimized via REINFORCE with baseline:

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{(r,a) \sim p_{\theta}(\cdot|x)} [(R_{\text{VCoT}}(r, a | x) - b(x)) \nabla_{\theta} \log p_{\theta}(r, a | x)] \quad (4)$$

where $b(x) = \mathbb{E}_{(r,a)}[R_{\text{VCoT}}]$ is a state-dependent baseline estimated by a value network.

3.4 Step-Level Credit Assignment

A key challenge in CoT training is credit assignment: which steps caused the final answer to be correct or incorrect? We use a step-dropout technique to estimate step importance:

$$I_{\ell} = \mathbb{E}_{a \sim p_{\theta}(\cdot|x, r_{-\ell})}[R_{\text{out}}(a)] - \mathbb{E}_{a \sim p_{\theta}(\cdot|x, r)}[R_{\text{out}}(a)] \quad (5)$$

where $r_{-\ell}$ denotes the chain with step ℓ replaced by a mask token. A high importance I_{ℓ} indicates step ℓ is causally necessary; low importance suggests a shortcut step that can be removed without affecting the final answer.

4 Experiments

4.1 Benchmark Descriptions

GSM8K [4]: 8,500 grade school math problems requiring 2–8 arithmetic steps. We use the standard test split of 1,319 problems.

MATH [5]: 12,500 competition mathematics problems across 5 difficulty levels and 7 subject areas (Algebra, Number Theory, Geometry, etc.). We use the full test split of 5,000 problems.

4.2 Main Results

Table 1: GSM8K and MATH accuracy (pass@1). VCoT denotes our verified CoT training; CoT (ORM) denotes standard outcome-supervised CoT fine-tuning.

Model	GSM8K (base)	GSM8K (VCoT)	MATH (base)	MATH (VCoT)
Zen MoDE-7B	82.4	88.6	52.1	61.3
Zen MoDE-32B	88.7	92.1	63.4	69.8
Zen MoDE-72B	90.8	94.1	67.2	72.4

4.3 CoT vs. No CoT

Table 2: Impact of chain-of-thought reasoning. CoT provides the largest gains on MATH (where multi-step reasoning is necessary) and smaller gains on GSM8K.

Model	GSM8K (no CoT)	GSM8K (+CoT)	MATH (no CoT)	MATH (+CoT)
Zen MoDE-7B	64.2	88.6 (+24.4)	28.3	61.3 (+33.0)
Zen MoDE-32B	74.8	92.1 (+17.3)	39.7	69.8 (+30.1)
Zen MoDE-72B	79.3	94.1 (+14.8)	46.8	72.4 (+25.6)

Smaller models benefit more from CoT (larger absolute gains), consistent with the hypothesis that CoT compensates for limited parametric reasoning capacity.

4.4 Stratified Analysis by Reasoning Length

Table 3: GSM8K accuracy stratified by number of required reasoning steps (Zen MoDE-72B). VCoT provides the largest gains for long chains (≥ 6 steps).

Steps required	Base	CoT (ORM)	VCoT
2–3 steps	96.8	97.4	97.6
4–5 steps	91.2	93.7	94.9
6–7 steps	85.4	89.1	92.4
≥ 8 steps	76.3	83.6	89.7

4.5 MATH Subject Breakdown

Table 4: MATH accuracy by subject area, Zen MoDE-72B with VCoT.

Subject	Base	VCoT	Δ
Algebra	78.4	84.2	+5.8
Number Theory	62.1	71.3	+9.2
Geometry	58.9	67.8	+8.9
Counting & Prob.	71.3	78.9	+7.6
Precalculus	55.4	65.1	+9.7
Intermediate Alg.	61.8	70.4	+8.6
Pre-algebra	88.1	91.6	+3.5

4.6 Reasoning Shortcut Analysis

Table 5: Reasoning shortcut incidence rate: fraction of correctly-answered problems where at least one step has PRM score $q_\ell < 0$.

Training method	GSM8K shortcut rate (%)	MATH shortcut rate (%)
CoT (ORM, no step supervision)	18.4	31.7
CoT (PRM labels only)	12.1	22.4
VCoT (ours)	5.3	9.1

VCoT reduces shortcut incidence by 71% on GSM8K and 71% on MATH compared to standard ORM-based CoT training.

4.7 Effect of γ (Step vs. Outcome Balance)

Table 6: GSM8K accuracy as a function of the step reward weight γ in Equation 3. $\gamma = 0.3$ gives the best trade-off.

γ	GSM8K (%)	MATH (%)	Shortcut rate (%)
0.0 (ORM only)	92.3	69.1	18.4
0.1	93.1	70.4	11.2
0.3	94.1	72.4	5.3
0.5	93.7	71.8	4.1
1.0	91.4	68.9	3.8

5 Discussion

5.1 Why Step-Level Supervision Matters

Outcome-only supervision creates an exploitation pressure: the model searches for any path to the correct final answer, including logically invalid shortcuts. Step-level supervision provides a curriculum of intermediate correctness that forces the model to internalize valid reasoning patterns. This is especially beneficial for out-of-distribution generalization: models trained with VCoT show 14% higher accuracy on held-out competition problems not in the training distribution.

5.2 Scalability of PRM Training

Training the PRM requires step-annotated data. We reduce the human annotation burden by using automated verifiers for mathematics (symbolic solvers) and a judge model for natural language. The judge model’s agreement with human annotators is 87%, sufficient for reliable step-level training signal.

5.3 Compute Budget for CoT

Longer chains increase inference cost proportionally. Our analysis shows the optimal chain length (maximizing accuracy per FLOP) is 6–8 steps for GSM8K and 10–14 steps for MATH. Beyond this, marginal accuracy gains do not justify the additional compute.

6 Conclusion

We have presented verified CoT training — combining scratchpad pretraining, process reward models, and step-level policy gradient updates — as a method for developing robust multi-step reasoning in Zen MoDE models. Key results: 94.1% on GSM8K, 72.4% on MATH, and a 71% reduction in reasoning shortcuts. Step-level supervision is the critical ingredient: it forces the model to internalize logically valid reasoning paths rather than exploiting answer-level shortcuts.

References

- [1] J. Wei, X. Wang, D. Schuurmans, et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. NeurIPS, 2022.
- [2] T. Kojima, S.S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa. *Large Language Models are Zero-Shot Reasoners*. NeurIPS, 2022.
- [3] H. Lightman, V. Kosaraju, Y. Burda, et al. *Let’s Verify Step by Step*. ICLR, 2024.
- [4] K. Cobbe, V. Kosaraju, M. Bavarian, et al. *Training Verifiers to Solve Math Word Problems*. arXiv:2110.14168, 2021.
- [5] D. Hendrycks, C. Burns, S. Kadavath, et al. *Measuring Mathematical Problem Solving With the MATH Dataset*. NeurIPS, 2021.