INSTRUCT, NOT ANSWER: USING INSTRUCTION PRIVILEGES IN ON-POLICY CONTEXT DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-Policy Context Distillation (OPCD) has recently emerged as a powerful technique for transferring context to student models and for self-improvement. In OPCD, the teacher is conditioned on privileged information, and the goal is to minimize the Kullback-Leibler (KL) divergence between the privileged teacher and the student, evaluated on student-generated tokens. Many existing studies show that using instance-specific gold answers or gold demonstrations as the default privilege can hurt training performance, especially out-of-distribution (OOD). In this work, we instead design general instructions that target common student mistakes observed on the training samples, and show that such simple instructions can outperform gold as the OPCD privilege. In autoformalization tasks, using a matched formatting instruction as the privilege could outperform gold in OOD accuracy by a large margin. In 7 out of 8 experiments using ProverQA, ProofWriter, and ProntoQA as datasets, and Qwen3-Thinking and Olmo3-Thinking families as models, matched instruction privileges outperform gold in OOD by 4 to 17 points, while remaining on par with gold in-domain. Each instruction is only a few sentences (and thus contains much less information compared to all instance-specific gold) and is applied uniformly to every training sample. These results indicate that a general instruction, which applies equally to source and target domain examples, can be substantially more transferable than instance-specific gold in OPCD while maintaining in-domain performance.

1 INTRODUCTION

Context Distillation (CD) transfers behaviors from a teacher conditioned on privileged context to a student, and is a common approach to task-specific post-training. Prior work shows that CD can internalize instructions, explanations, concrete demonstrations, output schema and factual knowledge (Snell et al., 2022; Choi et al., 2023; Caccia et al., 2025). CD methods usually minimize the tokenwise Kullback-Leibler (KL) divergence between the teacher and the student conditioning on teacher-generated tokens (Askell et al., 2021; Snell et al., 2022), i.e. off-policy context distillation. However, off-policy distillation means that the student is only trained on traces that teacher generates most frequently, so students cannot correct mistakes on traces that it puts more weights on, resulting in compound errors. Consequently, On-Policy (Context) Distillation (OPCD) has been a popular technique designed to mitigate this distribution-mismatch issue (Agarwal et al., 2024; Gu et al., 2024; Zhao et al., 2026; Ye et al., 2026; Kim et al., 2026; Kaur et al., 2026; Pan et al., 2026; Kim & Lee, 2026; Hübotter et al., 2026), where KL divergence is minimized conditioning on student-generated tokens.

Recently, studies have shown that adding privilege contexts to the teacher could degrade OPCD training performance, especially OOD performance, across math and science (Li et al., 2026; Kaur et al., 2026; Jukić & Titov, 2026; Wang et al., 2026a). However, the degradation observed by these studies use *instance-specific gold demonstration or answer* as teacher privileges, meaning that the privilege is a correct reference solution to *each* problem. Additionally, since gold privileges are often harder to build and could be noisy, other privileges such as system prompts have been used as privilege contexts in OPCD (Ye et al., 2026; Zhu et al., 2026; Wang et al., 2026b; Rezaei et al., 2026). However, there is limited understanding of whether domain-specific instructions could obtain better in-domain and OOD performance than instance-specific gold privilege in OPCD. Therefore, we ask the following question:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

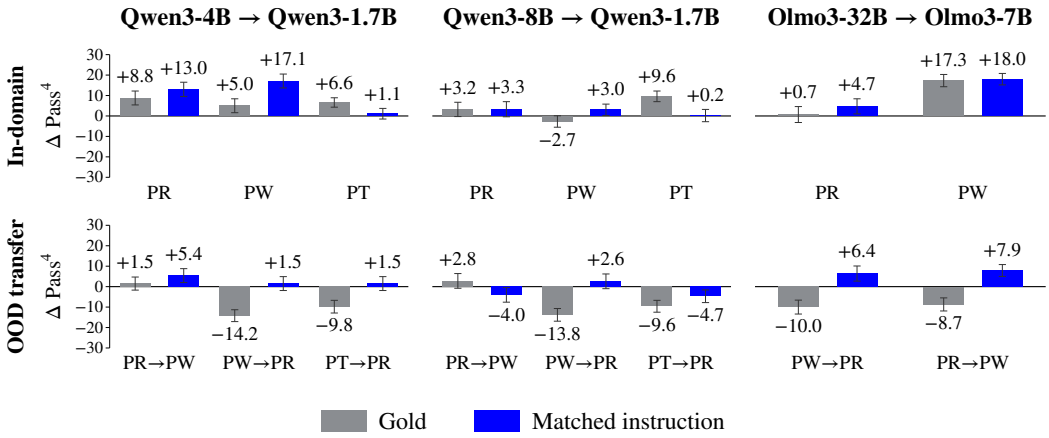


Figure 1: Pass^4 accuracy change of using gold FOL and one matched instruction for each dataset as privileges, compared to not using any privilege baseline (OPD). PR, PW, PT refer to ProverQA, ProofWriter and ProntoQA datasets. Model $A \rightarrow$ Model B refers to teacher-student pairs in OPCD, and dataset $X \rightarrow$ dataset Y refers to OOD performance trained on dataset X and evaluated on dataset Y. The error bars denote ± 1 standard error.

Can short and reusable instruction privileges improve in-domain student performance while supporting better cross-dataset generalization than instance-specific gold in OPCD?

Our question makes autoformalization the perfect testbed because it is easy to understand the mistakes the model makes and therefore design targeting privileges. Autoformalization is a crucial area to enable large scale math research and verification thanks to the math breakthroughs and their formalization efforts using frontier models (OpenAI, 2026a;b; Anthropic, 2026). Concretely, autoformalization requires the model to translate natural language statements (that include several premises and a question based on these premises) to some formal targets that can be solved to a verdict: TRUE, FALSE or UNKNOWN by a solver. The goal is to make solver output equal to the answer to the question. In this paper, we focus on first-order-logic (FOL), which is the standard target for logical reasoning. See Section 2.3 for a detailed explanation of autoformalization and privilege design in OPCD, and Section 3 for a detailed explanation why we believe autoformalization is the ideal testbed for answering this question.

Main Results. Using Qwen3-Thinking family (Yang et al., 2025) and Olmo3-Thinking family (Ettinger et al., 2026) as our models and standard logical datasets including ProverQA (Qi et al., 2025), ProofWriter (Tafjord et al., 2021) and ProntoQA (Saparov & He, 2023), we give a positive answer to our main question (see Figure 1 as a highlight of experimental results).

Gold privilege helps in-domain but hurts OOD. Our experiments indicate that using gold FOL as privilege generally improves student's in-domain performance but hurts OOD performance, aligning with existing OPCD results in math and science.

Instruction privilege helps in-domain while avoiding most OOD degradation. We identify formatting problems the student makes on training sets, and design generic formatting instructions as teacher privileges. Shown in Figure 1, our instructions outperform gold in most OOD settings (7 out of 8) and have similar performances compared to gold in-domain (5 out of 8) [1]. Our instructions are simply a few sentences that only contain a specific formatting instruction (e.g. "translating XOR relationship to $\oplus$"). Compared to gold privileges that need to contain the correct FOL formulas for each problem, our instructions contain much less information, and are applied uniformly to every single problem. These improvements indicate that simple instructions targeting source and target datasets can be well internalized by OPCD, and we can use much less information in privileges to obtain better training results.

[1] We give explanations for why gold dominates in-domain ProntoQA in Section 5: ProntoQA is cleaner in formatting so formatting instructions are less useful.

2 PRELIMINARIES

In this section, we will explain the basics of context distillation and autoformalization. Throughout, we will use θ_T to denote the teacher distribution, θ_S to denote the student distribution, x to denote the prompt and y to denote the output. D_{KL} is the KL divergence.

2.1 KNOWLEDGE DISTILLATION FOR LLMS

Knowledge distillation (Hinton et al., 2015; Kim & Rush, 2016; Sanh et al., 2019) lets the student mimic the teacher's behavior by optimizing towards the teacher's soft probability distribution at any given sequence. In particular, given a base input or prompt x and a reference output sequence y, the loss (average per-token KL divergence) is defined as

$$\mathcal{L}_{\mathrm{KD}}(\theta_T, \theta_S, x, y) = \frac{1}{|y|} \sum_{i=1}^{|y|} D_{\mathrm{KL}}(\theta_T(\cdot|x, y_{<i}) || \theta_S(\cdot|x, y_{<i})).$$

In off-policy knowledge distillation, y is generated by the teacher $y \sim \theta_T(\cdot|x)$. However, since θ_T and θ_S are different distributions, it is possible that the student encounters partial sequences that are not usually encountered by the teacher, resulting in under-performance as the student has not been trained on such partial sequences. On-policy distillation is designed to navigate this problem: $y \sim \theta_S(\cdot|x)$ is now chosen from student's own distribution such that the teacher can correct the student's mistakes at the student's own traces.

2.2 CONTEXT DISTILLATION FOR LLMS

Context distillation (CD) provides the teacher with special privilege c that is embedded in the prompt. For example, c could be the gold demonstration or label, and standard knowledge distillation is when $c = \emptyset$. Consequently, the teacher's distribution becomes $\theta_T(\cdot|c, x, y_{<i})$, and we minimize the KL divergence between privileged teacher's distribution and student distribution. For standard off-policy context distillation, the loss thus becomes

$$\mathcal{L}_{\mathrm{off}}(\theta_T, \theta_S, x) = \mathbb{E}_{y \sim \theta_T(\cdot|c,x)} \Big[\frac{1}{|y|} \sum_{i=1}^{|y|} D_{\mathrm{KL}}(\theta_T(\cdot|c, x, y_{<i}) || \theta_S(\cdot|x, y_{<i})) \Big].$$

For OPCD, y is sampled from student distribution, and the loss is (notice that the only difference is the distribution of y under expectation)

$$\mathcal{L}_{\mathrm{on}}(\theta_T, \theta_S, x) = \mathbb{E}_{y \sim \theta_S(\cdot|x)} \Big[\frac{1}{|y|} \sum_{i=1}^{|y|} D_{\mathrm{KL}}(\theta_T(\cdot|c, x, y_{<i}) || \theta_S(\cdot|x, y_{<i})) \Big].$$

When the teacher and student are the same model, OPCD becomes On-Policy Self-Distillation (OPSD).

2.3 AUTOFORMALIZATION

Autoformalization is the task of translating natural language to some formal targets. In this paper, we focus on first-order logic (FOL), which is the standard target for logical reasoning. Each sample in the dataset contains a set of premises and a question, and the model translates all of them into FOL formulas. The Vampire solver (Kovács & Voronkov, 2013) takes in the FOL translation and returns the verdict to the question (see Figure 2 for an example of training students on autoformalization using OPCD).

Note that a model can make many mistakes during translation. For example, it might forget to close a parenthesis, translate the same predicate to different names that the solver cannot recognize (*like* and *likes* are lexically identical, but a solver does not know and will identify them as separate predicates), or hallucinate a premise that never exists. Figure 2 illustrates the failure mode where $\forall$ is dropped in the translation. All these errors could make the solver output the wrong verdict.

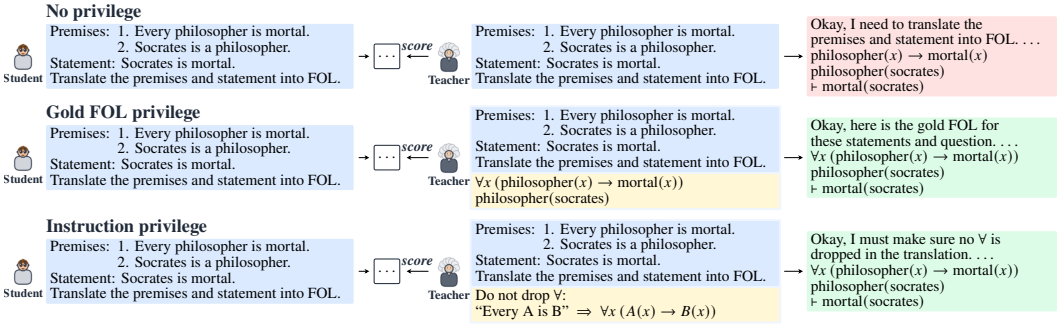


Figure 2: Illustration of training students on autoformalization with OPCD. Standard OPD uses no privilege in the teacher, and standard OPCD uses gold FOL. This paper studies using different types of formatting instructions as privileges.

3 Why Autoformalization is Ideal for Our Goals

Autoformalization is well suited to studying *what makes a good privilege* because it pairs an objective verifier with a rich, decomposable privilege space. First, correctness is decided by a symbolic solver (e.g. Vampire, z3) rather than a noisy judge: every rollout yields a discrete TRUE/FALSE/UNKNOWN verdict. Second, the failure modes are *discrete and diagnosable*: a formula is unparseable, drops a ∀, mistranslates an XOR, or names one concept two ways, so we can attribute each wrong verdict to a specific cause and write a several-line instruction that targets it (see Figure 3 below for examples). In free-form generation (e.g. math and science) these errors are entangled and hard to isolate. Third, the *same* formatting rules govern all three datasets (and likely to all other logical datasets in autoformalization tasks), so we can construct source→target pairs whose correct privilege is *known to be shared* and directly measure whether a privilege transfers, exactly the property that separates a reusable instruction from instance-specific gold. Finally, both privilege types arise naturally within the same task (a gold FOL formula per instance vs. a dataset-level formatting rule), enabling a clean, controlled head-to-head under identical training and evaluation.

4 Experimental Setup

Our goal is to evaluate how teacher privileges in OPCD affect training in autoformalization. The training objectives for both can be found at Section 2.2, and the autoformalization task is introduced in Section 2.3. For comparison, we also conduct experiments for off-policy CD and direct question-answering on the same datasets. In direct question-answering experiments, the privilege is the gold verdict (TRUE/FALSE/UNKNOWN). The system prompts that we use can be found in Appendix A.1.

Models. We experiment with the Qwen3-Thinking family (Yang et al., 2025) and the Olmo3-Thinking family (Ettinger et al., 2026). For Qwen3 family, the student is Qwen3-1.7B and the teacher has two scales: Qwen3-4B and Qwen3-8B. For Olmo3 family, the student is Olmo3-7B and the teacher is Olmo3-32B.

Datasets. For training and evaluation data, we use ProverQA (Qi et al., 2025), ProofWriter (Tafjord et al., 2021) and ProntoQA (Saparov & He, 2023) datasets for autoformalization and direct question-answering. These three datasets are all standard benchmarks for logical reasoning. We use the gold FOL formulas provided in ProverQA dataset for autoformalization tasks, deterministic regex converters to generate the gold FOL formulas for ProofWriter and ProntoQA, and gold verdict in all datasets for direct question-answering.

Evaluation metrics. We use average@8 and pass^4 verdict accuracy as our main evaluation metrics in both the autoformalization-solver pipeline and direct question-answering. The model is trained once, and the evaluation on a trained model is performed eight times with different random seeds. Average@8 is the percentage of rollouts that give the correct verdict among all $300 \times 8 = 2400$ rollouts, and pass^4 is the average, over all test problems, of $\binom{c}{4}/\binom{8}{4}$, where $c \in \{0, 1, \ldots, 8\}$ is the number of correct rollouts among the 8 for that problem. Pass^4 rewards *consistency* and grows

Table 1: Autoformalization results on ProverQA, ProofWriter and ProntoQA. Numbers in parentheses indicate the change of accuracy when gold FOL is added to the teacher prompt as privilege.

	ProverQA		ProofWriter		ProntoQA	
	avg@8	pass^4	avg@8	pass^4	avg@8	pass^4
Base (no distillation)	53.3	22.5	56.7	28.9	67.0	42.5
OPSD (temp=1.0)	53.4	22.4	54.7	25.9	67.2	43.3
Teacher: Qwen3-4B						
CD	58.7	28.0	73.5	40.8	90.1	73.2
CD with gold	**68.4** (+9.7)	**42.7** (+14.7)	81.0 (+7.5)	51.4 (+10.6)	90.4 (+0.3)	71.5 (-1.7)
OPCD	62.3	33.2	79.8	53.2	93.9	82.5
OPCD with gold	66.0 (+3.7)	42.0 (+8.8)	**83.3** (+3.5)	**58.2** (+5.0)	**96.7** (+2.8)	**89.1** (+6.6)
Teacher: Qwen3-8B						
CD	65.5	39.9	87.7	69.8	90.9	77.3
CD with gold	68.0 (+2.5)	41.8 (+1.9)	87.6 (-0.1)	67.7 (-2.1)	95.1 (+4.2)	84.2 (+6.9)
OPCD	66.2	43.8	**91.0**	**76.3**	91.7	79.4
OPCD with gold	**69.4** (+3.2)	**47.0** (+3.2)	89.0 (-2.0)	73.6 (-2.7)	**96.3** (+4.6)	**89.0** (+9.6)

Table 2: Direct question-answering results on three benchmarks. Numbers in parentheses indicate the change of accuracy when gold verdict is added to the teacher prompt as privilege.

	ProverQA		ProofWriter		ProntoQA	
	average@8	pass^4	average@8	pass^4	average@8	pass^4
Qwen3-1.7B, no distillation						
Base	78.5	64.7	79.0	69.6	82.8	68.0
Teacher: Qwen3-4B						
CD	84.2	70.9	78.0	63.2	85.9	67.4
CD with verdict	85.4 (+1.2)	72.1 (+1.2)	78.2 (+0.2)	61.2 (-2.0)	85.3 (-0.6)	65.3 (-2.1)
OPCD	85.8	75.3	81.5	70.6	**93.5**	**82.1**
OPCD with verdict	**86.2** (+0.4)	**75.4** (+0.1)	**82.1** (+0.6)	**74.0** (+3.4)	**93.5** (+0.0)	**82.1** (+0.0)
Teacher: Qwen3-8B						
CD	84.7	71.8	82.8	72.7	96.6	91.0
CD with verdict	86.0 (+1.3)	74.1 (+2.3)	83.4 (+0.6)	72.5 (-0.2)	97.2 (+0.6)	91.2 (+0.2)
OPCD	86.8	**77.9**	**85.5**	**78.1**	**97.9**	**95.3**
OPCD with verdict	**87.4** (+0.6)	76.7 (-1.2)	85.4 (-0.1)	77.1 (-1.0)	97.8 (-0.1)	94.8 (-0.5)

steeply with c (e.g. $c = 8, 6, 4$ contribute 1.0, 0.21, and 0.014, and $c < 4$ contributes 0), making it a stricter measure of whether a model *reliably*, rather than only occasionally, produces the correct answer. Pass^k has been used extensively to evaluate LLMs and agents (Yao et al., 2024; Anonymous, 2026).

Implementation details. For all datasets we use 1200 training samples and 300 evaluation samples. We set maximum length in training and generation consistently to avoid most truncation effects. For distillation, we use full vocabulary logit distillation with forward KL divergence without clipping. Rollouts and evaluation use temperature 0.7. All experiments are conducted on H100 GPUs with LoRA (Hu et al., 2022). Full experimental details can be found in Appendix A.2.

5 Experimental Results

5.1 Gold FOL Privilege Improves Verdict Accuracy

Table 1 reports the Qwen3 results for autoformalization on ProverQA, ProofWriter and ProntoQA when $c = \emptyset$ and the gold FOL. We evaluate base student (Qwen3-1.7B) as a baseline, and perform self-distillation with no privilege using a different temperature (1.0) to exclude distillation effects. We can see from the table that gold privilege consistently helps with distillation for both CD and OPCD. When teacher is Qwen3-4B, adding gold privilege to the teacher improves the pass^4 accuracy by $+8.8, +5.0, +6.6$ across three datasets in OPCD.

Error	Description	Instruction (added to the teacher prompt)
Unparseable	No solver-usable FOL: missing, truncated, or malformed.	Output only well-formed FOL: balanced parentheses, atoms as name(args), no prose or set notation.
XOR$\to\vee$	Gold's exclusive-or $\oplus$ dropped or rewritten as $\vee$.	Translate an exclusive-or relationship into $\oplus$, not $\vee$.
$\forall$-drop	A universal rule written without its $\forall$ (bare free variable), so it never instantiates.	Write every general rule with an explicit $\forall x(P(x) \to Q(x))$; never ground it to a named individual.
$\forall$-scope	A universal rule kept but mis-grouped: a connective left ungrouped alongside $\to$.	Fully parenthesize precedence: never leave $\wedge/\vee/\oplus$ ungrouped next to $\to$, e.g. $(A \wedge B) \to C$.
Coref-split	One concept given ≥ 2 inconsistent predicate names within the student's own FOL.	Use one fixed predicate name per concept; never a spelling or morphological variant.

Figure 3: Description of all formatting errors and their corresponding instructions. Note that a single rollout could exhibit multiple formatting errors.

Table 3: Error decomposition for the Qwen3-1.7B student, across all three datasets. The numbers are the percentage of rollouts in the training set that contains a specific type of error. Red denotes the most common problems for each dataset and blue denotes the semantic problem for ProntoQA.

Dataset	Unparseable	XOR$\to\vee$	$\forall$-drop	$\forall$-scope	Coref-split	Semantic
ProverQA	12.5	30.9	14.3	7.1	2.8	2.8
ProofWriter	8.3	0.0	28.6	10.9	0.8	1.0
ProntoQA	4.5	0.0	11.9	0.4	8.2	10.7

For comparison, Table 2 reports CD and OPCD results on direct question-answering tasks on ProverQA, ProofWriter and ProntoQA. Both CD and OPCD still improve direct question-answering capabilities, but verdict privilege has almost no effect on improving the accuracy on all datasets.

Additionally, OPCD dominates CD (best performing method, with or without gold) for both autoformalization and direct question-answering. In ProofWriter autoformalization task, OPCD achieves 58.2 pass^4 accuracy compared to CD's 51.4 when teacher is Qwen3-4B, and achieves 76.3 pass^4 accuracy compared to CD's 69.8 when teacher is Qwen3-8B. In ProntoQA autoformalization task, the gap is substantial as well: 89.1 compared to 73.2 for Qwen3-4B teacher and 89.0 compared to 84.2 for Qwen3-8B teacher. In direct question-answering task, OPCD's lead is even larger.

5.2 Formatting Errors are the Problem

We analyze the rollouts that a Qwen3-1.7B model produce on training sets to understand what type of mistakes the student makes. We find out that formatting errors are the major errors, and Figure 3 shows all major formatting error types that we find might be responsible for the wrong verdicts. We observe that many such formatting problems are incorrect translations of connectives and quantifiers, which makes gold FOL a natural candidate to fix them (having a gold FOL demonstration lets the model know how to produce correctly-formatted and parseable FOL formulas). Table 3 reports the error percentages of Qwen3-1.7B on all training sets before any training. It is clear that different datasets have different formatting problems.

ProverQA. Generally suffers from various formatting problems. In particular, XOR$\to\vee$ (translating XOR relationship to $\vee$) is the most common mistake that Qwen3-1.7B base model makes, while it also has big unparseable, $\forall$-drop and $\forall$-scope problems.

ProofWriter. ProofWriter also suffers from various formatting problems which are similar to ProverQA. However, ProofWriter does not have any XOR relationship, so there is no XOR $\to \vee$ problem in ProofWriter. Dropping $\forall$ is the most common mistake.

ProntoQA. The cleanest dataset in terms of formatting problems. Similar to ProofWriter, ProntoQA also does not have any XOR relationship. Dropping $\forall$ and coref-split (translating the same concept into different predicate names so Vampire solver could not identify) are the major problems in terms of formatting. However, ProntoQA suffers the most from semantic problems compared to the other two datasets, meaning that the FOL formulas produced by Qwen3-1.7B model do not have the correct semantic meaning as the natural language.

Table 4: Direct-prompting with in-prompt privileges for Qwen3-1.7B (no training): avg@8 and pass^4 changes are shown.

Dataset	Gold	Unparseable	XOR	∀-drop	∀-scope	Coref
ProverQA	+36.9 / +53.8	-3.4 / -6.2	+0.7 / +0.3	-7.1 / -8.4	-1.1 / -1.6	-2.4 / -2.0
ProofWriter	+35.0 / +51.3	-3.8 / -5.6	-1.5 / -3.3	+8.1 / +0.0	-4.3 / -6.7	-4.1 / -7.5
ProntoQA	+24.5 / +30.2	-3.7 / -0.8	-0.2 / +0.9	-0.3 / -2.7	-2.5 / +1.1	-1.8 / -0.1

Table 5: In-distribution pass^4 improvement over no-privilege (base) OPCD. Teachers are Qwen3-4B and Qwen3-8B and student is Qwen3-1.7B.

Dataset	Teacher	+Gold	+Unpars	+XOR	+∀-drop	+∀-scope	+Coref	Avg	Best
ProverQA	4B	+8.8	+9.2	+5.7	+0.6	+5.3	**+13.0**	+6.8	**46.2**
	8B	+3.2	+2.9	**+3.9**	-3.1	+0.3	+3.3	+1.5	**47.7**
ProofWriter	4B	+5.0	+5.7	+1.2	**+17.1**	+0.9	+0.5	+5.1	**70.3**
	8B	-2.7	+2.0	+2.0	**+3.0**	-0.7	-4.8	+0.3	**79.3**
ProntoQA	4B	**+6.6**	+1.1	-0.5	-1.1	-3.0	-15.9	-3.9	**89.1**
	8B	**+9.6**	+0.2	-1.5	-1.8	-3.6	-4.1	-2.2	**89.0**

5.3 Formatting Instructions as Privilege

We use formatting instructions as privileges in OPCD and compare them with using gold FOL (see Figure 3 for instructions we use). These formatting instructions do not provide any demonstration of what gold FOL should look like for any specific instance, but they inform the teacher specific rules that they should follow when generating the final FOL. Each instruction only contains a single piece of information that is generic to all problems in all datasets, so it contains much less information compared to instance-specific gold FOL formulas.

As a baseline, we first embed these instructions directly into the prompts to Qwen3-1.7B. Table 4 contains the changes when we add the privileges directly into the student prompts. Since gold privilege already contains the answers, adding them into the student prompt gives a substantial improvement overall. However, in-prompt formatting instructions give almost no improvement, and hurt model performance in many cases.

Table 5 reports the pass^4 improvement on ProverQA, ProofWriter and ProntoQA of using these formatting instructions as privileges in OPCD compared to not having any privilege. We can see from table that for ProverQA and ProofWriter, the best instructions dominate gold: Coref-split instruction achieves $+13.0$ and $+3.3$ compared to gold's $+8.8$ and 3.2 on ProverQA and ∀-drop instruction achieves $+17.1$ and $+3.0$ compared to gold's $+5.0$ and -2.7 on ProofWriter. On ProntoQA, gold dominates all instructions because as we have seen in Table 3, ProntoQA is the cleanest dataset with much fewer formatting errors but much higher semantic errors.

We also look at the error percentage change after training (Qwen3-4B teacher) in Table 6. Gold privilege generally reduces formatting errors in eval samples, but surprisingly increases semantic problems ($+3.7$, $+3.2$ for ProverQA and ProofWriter), and some formatting errors ($+12.6$ for Coref in ProverQA). Our instructions avoid these problems in general. A complete table including Qwen3-8B teacher can be found in Appendix A.3.

To prove the generalizability of our instructions, we use the best performing instruction on ProverQA (coref-split) and ProofWriter (∀-drop) and compare them against gold in Olmo3 models, using Olmo3-7B thinking as the student and Olmo3-32B thinking as the teacher. Table 7 reports the Olmo3 results. For both ProverQA and ProofWriter, matching instruction obtains a slightly better performance compared to gold in-domain ($+4.7$ on ProverQA and $+18.0$ on ProofWriter compared to gold's $+0.7$ and $+17.3$).

5.4 Out-Of-Domain Formatting Instruction Transferring

Table 8 reports the out-of-domain performance of using formatting instructions as privilege during training. Gold does not achieve the best OOD performance in every single row, and there is always a

Table 6: Change (percentage points) in the error decomposition after OPCD with the Qwen3-4B teacher, relative to the no-privilege Qwen3-1.7B student (Table 3).

Data	Priv.	Acc ↑	Unp. ↓	XOR ↓	∀drop ↓	∀scp ↓	Coref ↓	Sem ↓
ProverQA	None	+9.0	-4.5	-4.9	-5.8	+4.0	+1.0	-0.5
	+Gold	+12.7	-7.0	-20.6	-11.4	+2.5	+12.6	+3.7
	+Coref	**+14.1**	-8.1	-5.5	-5.1	+3.7	+1.4	-1.0
ProofWriter	None	+23.1	-6.5	0.0	-12.4	-3.7	-0.6	-0.8
	+Gold	+26.6	-2.4	0.0	-25.7	-6.0	-0.5	+3.2
	+∀drop	**+32.1**	-6.6	0.0	-21.9	-2.9	-0.6	-0.8
ProntoQA	None	+26.9	-3.5	0.0	-9.8	-0.4	-7.8	-8.0
	+Gold	**+29.8**	-4.0	0.0	-9.9	-0.4	-7.9	-10.2
	+Unp.	+27.8	-3.3	0.0	-10.7	-0.4	-7.7	-8.3

Table 7: In-distribution and OOD pass^4 improvement over no privilege OPCD. Teacher is Olmo-32B-Thinking and student is Olmo-7B-Thinking.

		ProverQA		ProofWriter		ProntoQA	
Trained on	Privilege	avg@8	pass^4	avg@8	pass^4	avg@8	pass^4
ProverQA	+Gold	+2.3	+0.7	-3.2	-8.7	-12.6	-12.5
	+Coref	**+4.2**	**+4.7**	**+2.8**	**+7.9**	**-0.9**	**-1.3**
ProofWriter	+Gold	-4.6	-10.0	+6.6	+17.3	-0.6	-1.0
	+∀-drop	**+3.4**	**+6.4**	**+7.2**	**+18.0**	**+3.7**	**+5.9**

formatting instruction that outperforms gold. In 5 out of 6 cases (except ProverQA→ProofWriter), gold degrades model performance by a large margin; on the other hand, even the average instructions do not suffer from such degradation.

We have seen in Table 5 that Coref-split instruction is the best for in-domain ProverQA. For ProofWriter→ProverQA (ProntoQA→ProverQA), coref-split instruction obtains $+4.4(+3.3)$ and $-0.9(+4.2)$ respectively and it is still the best instruction overall for OOD on ProverQA. For ProofWriter, ∀-drop instruction exhibits a clear win in Table 5, and it is also the clear winner here for ProverQA→ProofWriter (ProntoQA→ProofWriter) as it obtains $+14.5(+7.2)$ and $+8.1(+0.2)$ respectively. Finally, gold privilege performs the best on ProntoQA in-domain but it degrades the model performance OOD: for Qwen3-4B (Qwen3-8B) teacher, $-5.5(-7.6)$ for ProverQA→ProntoQA and $-6.2(-8.9)$ for ProofWriter→ProntoQA. Unparseable instruction performs the best overall on these two OOD evaluations.

Finally, Table 7 reports OOD performance given the best in-domain performing instruction for ProverQA and ProofWriter. The best performing instruction (coref-split for ProverQA and ∀-drop for ProofWriter) outperforms gold in both OOD eval datasets and under both metrics. Similar to Qwen3 runs, instructions do not suffer from performance degradation when trained on one dataset and evaluated on another.

6 Related Work

Privilege Design in Context Distillation. Context Distillation (Askell et al., 2021; Snell et al., 2022; Choi et al., 2023; Caccia et al., 2025) is an effective way to transfer knowledge from a teacher to a student without changing teacher's weights. There have been many studies that explore context (privilege) design such as experiential knowledge (Ye et al., 2026), query-specific demonstrations (Shenfeld et al., 2026), few-shot examples (Yang et al., 2024) and split contexts (Padmanabhan et al., 2026). This work uses autoformalization as a testbed to study privilege design, and we design specific formatting instruction privileges to understand whether they are more transferrable compared to instance-specific gold.

On-Policy (Self) Distillation. On-Policy Distillation (OPD) was designed to mitigate the teacher-student distribution mismatch problem in training. Recent work has studied distillation in the context

Table 8: Out-of-distribution transfer (OPCD): pass[4] improvement over no-privilege (base) distillation. Student is Qwen3-1.7B and teachers are Qwen3-4B and Qwen3-8B. PR, PW, PT represent ProverQA, ProofWriter and ProntoQA respectively.

Transfer	Teacher	+Gold	+Unpars	+XOR	+∀-drop	+∀-scope	+Coref	Avg	Best
PR→PW	4B	+1.5	+7.5	+6.0	**+14.5**	+7.4	+5.4	+8.2	**53.2**
	8B	+2.8	+2.0	+1.6	**+7.2**	−1.0	−4.0	+1.2	**60.9**
PR→PT	4B	−5.5	+0.9	+1.6	**+5.2**	+0.6	+0.1	+1.7	**57.0**
	8B	−7.6	**+2.9**	−0.6	−2.9	−2.0	−4.5	−1.4	**60.6**
PW→PR	4B	−14.2	+0.7	−0.5	+1.5	+0.5	**+4.4**	+1.3	**37.3**
	8B	−13.8	+2.3	−2.5	+2.6	−0.6	**+3.3**	+1.0	**40.4**
PW→PT	4B	−6.2	+3.2	+0.4	**+3.7**	+0.7	−1.6	+1.3	**52.0**
	8B	−8.9	**+3.6**	−0.2	−0.8	+1.2	−2.7	+0.2	**60.5**
PT→PR	4B	−9.8	**+1.5**	−3.1	−1.6	−0.8	−0.9	−1.0	**27.0**
	8B	−9.6	−4.7	−4.5	−0.9	−3.1	**+4.2**	−1.8	**29.6**
PT→PW	4B	−14.2	+3.6	−3.1	**+8.1**	+0.0	−8.5	+0.0	**41.5**
	8B	−8.3	**+1.9**	−3.0	+0.2	−4.2	−6.0	−2.2	**37.2**

of policy (Ye et al., 2026), sampling efficiency (Ge et al., 2026) in OPD, self-distillation (Zhao et al., 2026; Kim & Lee, 2026; Hübotter et al., 2026; Penaloza et al., 2026) and limitations in policy methods (Kaur et al., 2026; Li et al., 2026; Jukić & Titov, 2026; Wang et al., 2026a). This work studies how to design privileges in On-Policy (Self) Distillation.

Autoformalization. The autoformalization-solver pipeline has been well studied in the literature (Bui et al., 2025), and many prior studies have focus on fine-tuning language models to enable better autoformalization (Ryu et al., 2025; Thatikonda et al., 2026; Putra et al., 2026; Bansal et al., 2025), and most existing work uses off-policy methods. This work designs formatting instructions and use them to understand popular fine-tuning algorithms such as off-policy and on-policy distillation.

7 Discussions

We conducted an empirical study of privilege design in On-Policy Context Distillation, where we found that using instructions as privileges could improve student performance while avoiding OOD degradation caused by using gold privileges. The instructions we designed are generic and contain much less information than instance-specific gold privilege, suggesting that providing the teacher full answers might be less effective than providing the teacher a transferrable instruction that targets student's vulnerabilities.

Future work. This work focuses on autoformalization where the model translates natural language into first-order logic. We explain in Section 3 why we believe autoformalization is an ideal testbed for understanding the use of privilege, but we believe this question could be studied in other domains as well. The privileges could look very different in other domains, and we hope our study can inform future work that generic instructions targeting student mistakes can boost training performance. One promising direction is to develop a framework to understand the effectiveness of different privilege types in distillation, and their tradeoffs in terms of both in-domain and OOD performance.

AI use statement

In this work, we used generative AI tools for generating the codebase for experiments (including converters that create labels for the datasets), implementing methods, supporting qualitative and thematic data analysis, interpreting results and generating images and tables in the paper. Additionally, we used generative AI tools for creating scientific images and figures. Helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. We have reviewed all AI-assisted work. LLM-generated code has been tested for correctness by the authors, and LLM-generated texts have been checked manually by the authors to

ensure they reflect the correct meanings. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

Our experiments are reproducible: experiments are described in Section 4 and further details are described in Appendix A.1 and A.2.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/5be69a584901a26c521c2b51e40a4c20-Paper-Conference.pdf.

Anonymous. Sequent-prover: Training agents for formal, checkable SMT-based reasoning. In *Submitted to ACL Rolling Review - May 2026*, 2026. URL https://openreview.net/forum?id=DLMqDyHLTu. under review.

Anthropic. Fermat's Last Theorem in Lean 4. GitHub repository, 2026. URL https://github.com/anthropics/fermats-last-theorem. Accessed: September 6, 2026.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A General Language Assistant as a Laboratory for Alignment. *arXiv e-prints*, art. arXiv:2112.00861, December 2021. doi: 10.48550/arXiv.2112.00861.

Hannah Bansal, Kemal Kurniawan, and Lea Frermann. Robustness of neurosymbolic reasoners on first-order logic problems. In Jonathan K. Kummerfeld, Aditya Joshi, and Mark Dras (eds.), *Proceedings of the 23rd Annual Workshop of the Australasian Language Technology Association*, pp. 1–12, Sydney, Australia, November 2025. Association for Computational Linguistics. ISBN 1834-7037. URL https://aclanthology.org/2025.alta-main.1/.

Tuan Bui, Trong Le, Phat Thai, Sang Nguyen, Minh Hua, Ngan Pham, Thang Bui, and Tho Quan. Speaking in words, thinking in logic: A dual-process framework in qa systems. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2025. doi: 10.1109/IJCNN64981.2025.11229117.

Lucas Caccia, Alan Ansell, Edoardo Ponti, Ivan Vulić, and Alessandro Sordoni. Training plug-n-play knowledge modules with deep context distillation, 2025. URL https://arxiv.org/abs/2503.08727.

Eunbi Choi, Yongrae Jo, Joel Jang, Joonwon Jang, and Minjoon Seo. Fixed input parameterization for efficient prompting. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8428–8441, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.533. URL https://aclanthology.org/2023.findings-acl.533/.

Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang

Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2026. URL https://arxiv.org/abs/2512.13961.

Xinmu Ge, Zizhuo Zhang, Yu Huang, Jianing Zhu, Lin Yuan, Wanli Gu, Weichang Wu, Weiran Huang, Xiaolu Zhang, Bo Han, Jun Zhou, and Jiangchao Yao. Towards understanding on-policy distillation through the lens of test-time scaling, 2026. URL https://arxiv.org/abs/2608.11829.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL http://arxiv.org/abs/1503.02531.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jonas Hübotter, Frederike Lübeck, Lejs Deen Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Josip Jukić and Ivan Titov. Geometric self-distillation for reasoning generalization, 2026. URL https://arxiv.org/abs/2607.06855.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam H Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. In *ICML 2026 Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026. URL https://openreview.net/forum?id=VhCJItwQHn.

Jaehoon Kim and Dongha Lee. Opsd compresses what rlvr teaches: A post-rl compaction stage for reasoning models, 2026. URL https://arxiv.org/abs/2605.06188.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of llms? *arXiv preprint arXiv:2603.24472*, 2026. URL https://arxiv.org/abs/2603.24472.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139/.

Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In Natasha Sharygina and Helmut Veith (eds.), *Computer Aided Verification (CAV 2013)*, volume 8044 of *Lecture Notes in Computer Science*, pp. 1–35. Springer, 2013. doi: 10.1007/978-3-642-39799-8_1.

Yunhe Li, Hao Shi, Wenhao Liu, Mengzhe Ruan, Hanxu Hou, Zhongxiang Dai, Shuang Qiu, and Linqi Song. Demopsd: Disagreement-modulated policy self-distillation, 2026. URL https://arxiv.org/abs/2607.02502.

OpenAI. An OpenAI model has disproved a central conjecture in discrete geometry, May 2026a. URL https://openai.com/index/model-disproves-discrete-geometry-conjecture/. Accessed: September 6, 2026.

OpenAI. Ten advances in mathematics and theoretical computer science, August 2026b. URL https://openai.com/index/ten-advances-in-mathematics/. Accessed: September 6, 2026.

Shankar Padmanabhan, Mustafa Omer Gul, and Tanya Goyal. Updating parametric knowledge with context distillation retains post-training capabilities. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=OJsGhlTayF.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. Rlcsd: Reinforcement learning with contrastive on-policy self-distillation. *arXiv preprint arXiv:2606.11709*, 2026.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ebZcMQImhG.

Rizky Ramadhana Putra, Raihan Sultan Pasha Basuki, Yutong Cheng, and Peng Gao. NL2Logic: AST-guided translation of natural language into first-order logic with large language models. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6035–6051, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.317. URL https://aclanthology.org/2026.findings-eacl.317/.

Chengwen Qi, Ren Ma, Bowen Li, He Du, Binyuan Hui, Jinwang Wu, Yuanjun Laili, and Conghui He. Large language models meet symbolic provers for logical reasoning evaluation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=C25SgeXWjE.

MohammadHossein Rezaei, Anas Mahmoud, Zihao Wang, Utkarsh Tyagi, Advait Gosai, Razvan-Gabriel Dumitru, Aakash Sabharwal, Bing Liu, and Yunzhong He. Rubric-guided self-distillation: Post-training without rubric verifiers, 2026. URL https://arxiv.org/abs/2606.12507.

Hyun Ryu, Gyeongman Kim, Hyemin S. Lee, and Eunho Yang. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 24935–24964, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3e592c571de69a43d7a870ea89c7e33a-Paper-Conference.pdf.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=qFVVBzXxR2V.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. In *ICLR 2026 Workshop on Lifelong Agents: Learning, Aligning, Evolving*, 2026. URL https://openreview.net/forum?id=HlWA3V6iKF.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by Distilling Context. *arXiv e-prints*, art. arXiv:2209.15189, September 2022. doi: 10.48550/arXiv.2209.15189.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

Ramya Keerthy Thatikonda, Jiuzhou Han, Wray Buntine, and Ehsan Shareghi. Improving symbolic translation of language models for logical reasoning, 2026. URL https://arxiv.org/abs/2601.09446.

Jiaxuan Wang, Xuan Ouyang, Zhiyu Chen, Yulan Hu, Zheng Pan, Xin Li, and Lan-Zhe Guo. Trace: Distilling where it matters via token-routed self on-policy alignment, 2026a. URL https://arxiv.org/abs/2605.10194.

Xun Wang, Ruishuo Chen, Zhuoran Li, Yu Chen, and Longbo Huang. When context returns: Toward robust internalization in on-policy distillation, 2026b. URL https://arxiv.org/abs/2606.11627.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. Self-distillation bridges distribution gap in language model fine-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1028–1043, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.58. URL https://aclanthology.org/2024.acl-long.58/.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-Policy Context Distillation for Language Models. *arXiv e-prints*, art. arXiv:2602.12275, February 2026. doi: 10.48550/arXiv.2602.12275.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models. *arXiv e-prints*, art. arXiv:2601.18734, January 2026. doi: 10.48550/arXiv.2601.18734.

Siqi Zhu, Xuyan Ye, Hongyu Lu, Weiye Shi, and Ge Liu. The many faces of on-policy distillation: Pitfalls, mechanisms, and fixes, 2026. URL https://arxiv.org/abs/2605.11182.

A Appendix

A.1 System Prompts

Prompts. Students and the no-privilege teacher receive the prompt shown below: the ProverQA-style "Is the following statement true, false, or uncertain?" preamble is removed, so the model is asked to formalize a bare statement rather than to decide its truth (the solver decides truth).

Prompt example in ProverQA

System. You translate a natural-language logical-reasoning problem into a first-order logic (FOL) formalization. You are given PREMISES (context) and a STATEMENT (the question's claim). Produce FOL for each premise and for the statement. Think step by step about the predicates, entities, quantifiers, and connectives, then output one fenced `fol` code block: one FOL formula per premise (one per line), then a final line ⊢ `<conclusion-FOL>` giving the FOL of the STATEMENT. Use standard FOL syntax: ∀ ∃ ∧ ∨ ¬ → ↔ ⊕, predicates like `likes(Alice, Bob)`, constants capitalized. Do NOT decide whether the statement is true—a solver does that. Just translate faithfully.

User. PREMISES: If Paloma is attentive to details, then she understands lighting and visualizes compositions. If someone understands lighting, then they may not necessarily compose framing well, and vice versa. Paloma either buys quality equipment or has a creative vision. If Paloma has a creative vision, then she either understands lighting or can edit images, but not both. If Paloma travels extensively and has a good camera, then she can become a professional photographer. Paloma is detail-oriented. [*15 further premises omitted*]
Privileges (e.g. The gold FOL is ...)
STATEMENT to formalize: Paloma travels extensively.
Produce the FOL formalization (premise formulas, then ⊢ `conclusion`).

A.2 Full Experimental Details

Table 9: Hyperparameters.

Optimization	
Steps / checkpointing	400 steps, save every 50 (8 checkpoints)
Effective batch size	8 (per-device 1 × accum 1 × 8 GPUs)
Learning rate	5×10^{-6}
Optimizer / parallelism	DeepSpeed ZeRO, CPU optimizer offload (`cpu_adam`), 8-way
Training data	all 1200 rows, no correctness filtering (incorrect traces kept)
Seed	1 (single training seed; 8 evaluation samples)
LoRA	
Rank / α	64 / 128
Target modules	all linear (`q,k,v,o,gate,up,down`)
Objective & decoding	
Divergence	forward KL, β=0, JSD-clipping off
Temperature	0.7 (trace generation, on-policy rollout, and evaluation)
Token budget	8192 tokens, 16384 if more than 5% are truncated
Prompt limits	student ≤1536, teacher ≤12288 (fits the gold-privileged prompt)
Evaluation & scoring	
Samples per problem	8 (at T=0.7)
Checkpoint selection	peak mean in-domain accuracy over the 8 samples; OOD on the peak only
Accuracy	avg@8 (mean correctness over the 8 samples)
pass^4	mean over problems of $\binom{c}{4}/\binom{8}{4}$, c = #correct of 8

A.3 Complete Error Type Table

Table 10: Change in the error decomposition after OPCD, relative to the no-privilege Qwen3-1.7B student on evaluation rollouts.

Data	Tch.	Priv.	Acc ↑	Unp. ↓	XOR ↓	∀drop ↓	∀scp ↓	Coref ↓	Sem ↓
ProverQA	4B	None	+9.0	-4.5	-4.9	-5.8	+4.0	+1.0	-0.5
		+Gold	+12.7	-7.0	-20.6	-11.4	+2.5	+12.6	+3.7
		+Coref	**+14.1**	-8.1	-5.5	-5.1	+3.7	+1.4	-1.0
	8B	None	+12.9	-8.7	-6.4	-7.2	+2.5	+2.0	+0.2
		+Gold	**+16.2**	-9.7	-21.5	-11.2	-1.6	+12.0	+4.7
		+Coref	+14.1	-7.9	-8.1	-6.2	+0.6	+2.1	-0.6
ProofWriter	4B	None	+23.1	-6.5	0.0	-12.4	-3.7	-0.6	-0.8
		+Gold	+26.6	-2.4	0.0	-25.7	-6.0	-0.5	+3.2
		+∀drop	**+32.1**	-6.6	0.0	-21.9	-2.9	-0.6	-0.8
	8B	None	+34.4	-8.0	0.0	-21.3	-5.0	-0.7	-0.9
		+Gold	+32.3	-5.1	0.0	-25.6	-9.5	-0.6	+2.7
		+∀drop	**+35.1**	-7.5	0.0	-22.8	-4.9	-0.6	-0.8
ProntoQA	4B	None	+26.9	-3.5	0.0	-9.8	-0.4	-7.8	-8.0
		+Gold	**+29.8**	-4.0	0.0	-9.9	-0.4	-7.9	-10.2
		+Unp.	+27.8	-3.3	0.0	-10.7	-0.4	-7.7	-8.3
	8B	None	+24.7	-3.7	0.0	-9.9	-0.4	-7.2	-6.2
		+Gold	**+29.3**	-3.4	0.0	-10.7	-0.4	-8.0	-9.5
		+Unp.	+24.7	-4.0	0.0	-10.2	-0.4	-7.2	-5.5