LEAD INTO GOLD: THE SURPRISING EFFECT OF SYMBOLICALLY GENERATED DEDUCTION DATA ON LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Synthetic data now supplies much of the instruction tuning of language models, and it is usually written by another language model with the goal of sounding natural. We ask whether data at the opposite extreme also helps, a corpus of multi-hop deduction problems and their solutions that a short program generates with no model in the loop. The documents are correct by construction, but they contain no world knowledge and they name their constants abstractly. We replace a small fraction of a standard instruction-tuning mixture with such documents while holding the number of training examples fixed. On Qwen2.5-7B, replacing 5-25% of the mixture with such documents raises accuracy by up to 16 points on ProofWriter, 6 points on FOLIO, 7 points on the multi-step subtasks of BIG-Bench Hard and 13 points on the quantitative items of GPQA-Diamond, while general benchmarks are unchanged. These benchmarks cover formal logic, language inference, word problems and the sciences, all of which lie outside the domain of the generator. The gain on ProofWriter holds on models of 3 families and from 3B to 32B parameters. As little as 1% of the mixture teaches the model to write long derivations that a symbolic checker accepts, an ability that no baseline model in our study has. We trace the gains to two mechanisms. First, the model learns to apply each rule to the conclusion of the previous step, and so tracks an intermediate conclusion along a chain. Second, it stops preferring one answer label over the others, a bias that originates in the instruction corpus itself. Data of this kind is cheap to produce and needs no teacher model, enabling its usage in LLM post-training at large scales.

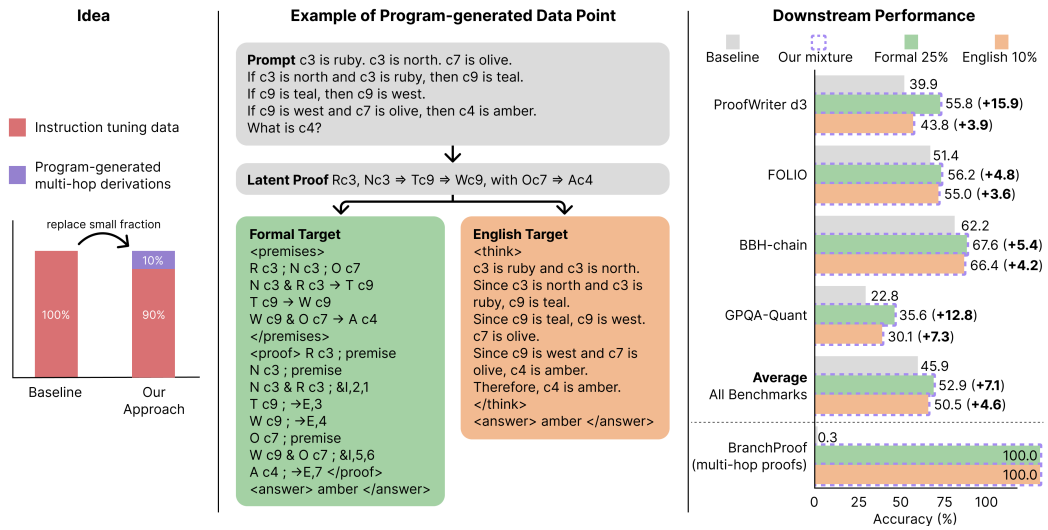

Figure 1: A sample from our program-synthetic dataset, which can be rendered in formal or in prose notation, and the effect of replacing 25 percent (formal) or 10 percent (English) of the instruction mixture with such examples on Qwen2.5-7B. The average is taken over the 13 benchmarks of Table 1. BRANCHPROOF is the fraction of held-out generator problems at chain length 25 with an accepted derivation.

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.

1 INTRODUCTION

Synthetic data has become a standard ingredient in the training of language models. Textbook-style corpora (Gunasekar et al., 2023; Eldan & Li, 2023; Abdin et al., 2024), rephrased web documents (Maini et al., 2024; 2025; Su et al., 2025) and prompted expansions of a seed corpus (Yang et al., 2025) have all been shown to improve downstream performance. In nearly every case, however, the synthetic text is written by another language model. Its quality is therefore bounded by the knowledge and the reasoning of the model that writes it, and repeated training on such text has been shown to narrow the distribution of the trained model (Shumailov et al., 2024; Dohmatob et al., 2025).

Synthetic data can also be produced by a program. A generator of a few hundred lines can sample a multi-hop deduction problem, solve it by forward chaining and write the solution out, which yields documents that are correct by construction and unlimited in number. Corpora of this kind have been used to evaluate the reasoning of language models (Clark et al., 2020; Tafjord et al., 2021; Saparov & He, 2023) and to train models on deduction in isolation (Morishita et al., 2023; 2024). Whether they are useful as an ingredient of general instruction tuning is much less clear. The documents contain no world knowledge, they describe abstract constants with arbitrary attributes, and they apply the same inference rule for dozens of lines, so there is little reason to expect them to improve a model on anything beyond the generator's own problems.

In this work we show that such data does improve a general instruction-tuned model by 7.1 points on average over 13 benchmarks, and we identify two mechanisms behind the gain. We fine-tune base models on an instruction mixture of fixed size in which a fraction of the instruction examples is replaced by generated derivations. Each derivation is written in one of two renderings of the same underlying proof, either as a formal proof with declared symbols and numbered lines or as a sequence of English sentences. On Qwen2.5-7B (Yang et al., 2024), replacing 5 to 25 percent of the mixture with generated derivations improves accuracy on ProofWriter (Tafjord et al., 2021) at depth 3 with chain-of-thought prompting by up to 15.9 points, on FOLIO (Han et al., 2024) by up to 6.4 points, on the multi-step subtasks of BIG-Bench Hard by up to 6.9 points and on the quantitative items of GPQA-Diamond by up to 12.8 points. Averaged over the 13 benchmarks of Table 1, the gain is 7.1 points for the formal rendering at 25 percent and 4.6 points for the English rendering at 10 percent, and general benchmarks are unchanged. Replacing only 1 percent of the mixture is enough for this model to write derivations of more than 50 lines that a symbolic checker accepts, at 1.8 times the chain length seen in training, whereas no baseline model in our study writes an accepted derivation. Section 4.2 repeats the comparison on Qwen2.5 at 3B, 14B and 32B, on Llama-3.1-8B and on OLMo-2-7B.

We then ask how documents without content can have this effect, and we find two mechanisms. First, the derivations teach the model to track an intermediate value through a chain of steps, as we show by shuffling the lines of the training proofs, which removes the effect, and by observing the baseline model lose the intermediate value midway through a chain on the multi-step subtasks of BIG-Bench Hard. Second, the derivations correct a systematic bias in the answers of the instruction-tuned model. On ProofWriter the baseline model asserts that a claim holds in three cases out of four, and the whole accuracy gain comes from the negated claims that are false. This bias originates in the instruction corpus. Tuning the base model on a second instruction corpus leaves it asserting that the claim fails in four cases out of five, and adding the derivations then moves its answers back toward the labels from the other side.

Our contributions are as follows: (1) We show that a small fraction of program-generated deduction data, substituted into an ordinary instruction mixture, improves an instruction-tuned model on deduction and multi-step benchmarks well beyond the generator's domain (Section 4). (2) We show that the derivation-writing ability appears at 1 percent of the mixture on Qwen2.5-7B and that it and the gains transfer to 3 model families and to models from 3B to 32B parameters (Section 4.2). (3) We identify the two mechanisms behind the gains and separate them by manipulating the order, the depth and the presence of the training proofs and by changing the instruction corpus (Sections 4.3 and 5). (4) We find that the English rendering of the proofs generalises further in depth and improves ProofWriter accuracy on every model we study, whereas the formal rendering helps some models and harms others (Section 5).

2 Related work

Synthetic data in pretraining and post-training. Synthetic text written by language models now enters every stage of training. It forms a large part of the pretraining corpus of Phi-4 (Abdin et al., 2024) and SmolLM2 (Ben Allal et al., 2025), it is used to rephrase web text at scale (Maini et al., 2024; Su et al., 2025), and its scaling behaviour has been studied systematically (Kang et al., 2025). Upsampling a domain at the end of pretraining (Blakeney et al., 2024; Liu et al., 2025a; Wang et al., 2025) and the composition of the instruction mixture (Lambert et al., 2025; Team OLMo et al., 2025) are known to have large effects on downstream performance, which motivates our replacement design. The limits of model-written data, both in the amount of human text available (Villalobos et al., 2024) and in the degradation caused by training on model output (Shumailov et al., 2024; Dohmatob et al., 2025), motivate sources that involve no model.

Abstract and procedural data. A line of work trains on data with structure but without content. Pretraining on music or on artificial languages transfers to natural language (Papadimitriou & Jurafsky, 2020; Chiang & Lee, 2022; Wu et al., 2022), summarisation can be learned from nonsense text (Krishna et al., 2021), and pretraining on formal languages imparts inductive biases that help later language learning (Hu et al., 2025). Jiang et al. (2026) warm up language models on procedurally generated sequences before natural text, and Lacombe et al. (2026) provide a suite of procedural generators for symbolic training. Ruis et al. (2025) find that the pretraining documents that most influence a model's reasoning are those that show a procedure. Our data is procedural in this sense, and we study it as a small component of instruction tuning of a pretrained model.

Synthetic deduction data. Program-generated deduction problems have been used to test the reasoning of language models (Clark et al., 2020; Tafjord et al., 2021; Saparov & He, 2023; Saparov et al., 2023; Zhu et al., 2024) and to train them. FLD and ALT (Morishita et al., 2023; 2024) train on large synthetic logic corpora and report gains on logical reasoning benchmarks, LogiPT (Feng et al., 2024) trains a model to emulate a solver, and Xia et al. (2025) study whether formal logic can be learned from data. Zhang et al. (2023) show that models trained on deduction data can learn statistical shortcuts, which is a concern we address by manipulating the proofs. Zhou et al. (2025) compare symbolic and English supervision on matched problems and Kabra et al. (2026) show that reinforcement learning on rule-generated multi-hop data improves real-world multi-hop benchmarks. Recent reinforcement learning work uses synthetic puzzles with verifiable answers as environments (Xie et al., 2025; Liu et al., 2025b; Chen et al., 2025; Stojanovski et al., 2025). Our work differs from these in that the generated data is a small replacement within an ordinary instruction mixture, the evaluation is on general benchmarks, and the analysis is directed at the mechanism of the gain.

Answer biases in tuned models. Instruction-tuned models acquire response biases that are separable from their underlying competence (Ouyang et al., 2022), negation remains difficult for language models (Kassner & Schütze, 2020; Hosseini et al., 2021; Truong et al., 2023), and shortcut learning is a general concern for trained models (Geirhos et al., 2020). We locate one such bias, attribute it to the instruction corpus and show which training data corrects it.

3 Experimental setup

3.1 Generated data

Our generator samples a set of constants, a set of unary predicates and a set of implication rules, chooses a target chain length, and constructs a problem in which the answer follows from the premises by that many applications of modus ponens. It then solves the problem by forward chaining and writes the solution out. Constants are named c_0 through c_{25} and predicates are drawn from a short list of colour and direction words, so that the documents contain no world knowledge. The proof that the generator constructs is written out in one of two renderings, both shown in Figure 1. The formal rendering declares the constants and the predicate symbols, states the premises as implications between atoms, and lists the proof as numbered lines together with the rule that justifies each line. The English rendering states the same premises and the same proof as ordinary sentences. The prompt, the answer and the proof graph are common to the two renderings, so a comparison between

them isolates the effect of notation. The training documents have chain lengths of up to 25 and the longest occupies 7,700 tokens.

3.2 TRAINING

We use Dolci (Team OLMo et al., 2025) as our main instruction corpus, the post-training corpus of OLMo 3, from which we draw 100,000 single-turn examples, and we repeat the main comparison on a second corpus in Section 4.3. A mixture with share p replaces a uniformly sampled fraction p of these examples by generated derivations, so that every model trains on the same number of examples, and the baseline trains on the unmodified 100,000 examples. We fine-tune each base model on each mixture for one epoch with a global batch of 128 sequences and a context window of 8,192 tokens, which fewer than 0.5 percent of the examples exceed. The main sweep trains Qwen2.5-7B at shares of 1, 5, 10 and 25 percent in both renderings with 3 seeds per configuration, and all reported numbers are means over seeds. To vary the size of the mixture we further train the baseline and the English rendering at 5 and 10 percent on 25,000, 50,000 and 200,000 examples. A third family of mixtures, FORMAL + ENGLISH, combines both renderings at the same 4 shares and prepends one sentence to each training prompt naming the rendering that derivation uses, which trains a single model that derives in either notation on request. We evaluate it with the prompts used for the other mixtures, which name no notation. In the same way we train Qwen2.5-3B, Qwen2.5-14B, Qwen2.5-32B, Llama-3.1-8B and OLMo-2-7B at the 5 percent formal and 10 percent English shares. All models are base models, which keeps the instruction tuning identical across conditions. Appendix B gives the remaining hyperparameters.

3.3 EVALUATION

The evaluation covers ProofWriter under the open-world assumption at inference depths 1, 3 and 5, on FOLIO, on GPQA-Diamond, on the three multi-hop question answering datasets of LongBench and on BIG-Bench Hard. ProofWriter and FOLIO are prompted for a direct answer, and we additionally evaluate ProofWriter under a prompt that asks the model to reason before answering. GPQA-Diamond is prompted to work through the options before the final choice, and BIG-Bench Hard uses the standard three-shot chain-of-thought prompts. We also evaluate on held-out problems of the generator, which we refer to as BRANCHPROOF, at chain lengths 5 to 45, where the model is prompted in the format on which it was trained and a forward-chaining checker verifies every line of the derivation it writes, so the score is the fraction of problems with an accepted derivation. Nine general benchmarks, among them GSM8K (Cobbe et al., 2021), MMLU and HumanEval (Chen et al., 2021), are described in Appendix D.1. No evaluation item shares an eight-gram with any training derivation.

For two benchmarks we additionally report a subset. BBH-chain averages the eight subtasks of BIG-Bench Hard in which the answer is obtained by propagating a truth value or a state along a sequence of steps, namely web of lies, the three object-tracking subtasks, the three logical-deduction subtasks and formal fallacies. GPQA-quant consists of the 73 items of GPQA-Diamond that contain a number with a physical unit or ask for a value, ratio or factor (Appendix C).

4 RESULTS

4.1 MAIN RESULTS

Table 1 reports the results for Qwen2.5-7B across the replaced share, and Figure 1 shows the formal rendering at 25 percent and the English rendering at 10 percent, the share with the largest average gain for each rendering. Replacing 5 to 10 percent of the instruction mixture with generated derivations improves every deduction and multi-step benchmark. On ProofWriter the English rendering at 10 percent improves accuracy by 6.4 points at depth 1, 13.9 points at depth 3 and 10.2 points at depth 5, and the formal rendering at 5 percent improves it by 5.3, 10.7 and 7.7 points. FOLIO improves by 6.4 points with the formal rendering at 5 percent. On BIG-Bench Hard the macro average improves by 2.7 points and the 8 chain subtasks account for most of the gain, improving by 6.7 points while the other 19 subtasks move by about 1 point, and web of lies, the subtask closest in form to the generated data, improves by 10.4 points with the formal rendering at 5 percent (Appendix D.2). GPQA-Diamond

Table 1: Replacing part of the instruction mixture with generated derivations in Qwen2.5-7B, reported as accuracy in %. Cells are shaded by the gain over the baseline.

		FORMAL				ENGLISH				FORMAL + ENGLISH			
Benchmark	Baseline	1%	5%	10%	25%	1%	5%	10%	25%	1%	5%	10%	25%
PW d1	30.9	33.0	36.2	36.1	**42.6**	31.7	35.1	37.3	40.0	30.6	33.2	37.4	33.6
PW d3	39.3	40.5	50.0	48.7	51.6	40.3	48.3	**53.2**	51.4	46.6	43.2	48.6	43.2
PW d3 (CoT)	39.9	39.3	45.3	49.5	**55.8**	37.5	44.1	43.8	47.0	38.4	39.8	38.4	36.4
PW d5	41.2	40.0	48.9	47.1	49.8	41.0	46.2	**51.4**	49.2	47.0	43.6	44.8	42.0
FOLIO	51.4	53.0	**57.8**	57.1	56.2	54.0	56.2	55.0	53.7	51.7	52.7	54.7	53.7
BBH	65.0	66.2	67.2	**67.7**	66.9	65.3	66.6	66.9	66.6	67.5	67.1	67.2	67.0
BBH-chain	62.2	66.1	67.6	68.9	67.6	63.2	65.2	66.4	65.6	68.0	**69.1**	69.0	68.5
Web of lies	85.5	92.7	95.9	97.6	**98.0**	84.1	84.5	87.5	87.6	93.2	94.8	95.6	97.2
GPQA-Diamond	31.8	31.8	33.8	32.2	36.9	33.3	35.4	34.0	31.8	37.4	**38.4**	31.3	33.3
GPQA-quant	22.8	23.3	27.9	26.5	35.6	26.5	29.2	30.1	28.8	**38.4**	35.6	24.7	31.5
HotpotQA	55.5	54.6	56.1	54.8	55.2	55.4	55.4	56.6	56.1	55.5	56.6	56.3	**56.7**
2WikiMQA	39.7	39.7	41.3	41.3	40.3	40.0	41.2	41.6	**42.7**	41.4	39.5	40.5	40.3
MuSiQue	31.1	31.3	32.7	31.0	31.6	32.2	**34.2**	32.8	31.0	32.1	33.5	31.6	31.8
Average gain	—	+1.2	+5.0	+4.8	+7.1	+0.6	+3.5	+4.6	+4.2	+4.0	+3.9	+3.4	+3.0
BRANCHPROOF	0.3	93.2	99.3	99.2	100.0	98.3	99.7	100.0	100.0	25.5	15.0	99.0	100.0

improves by up to 5.1 points, and the gain is concentrated in the quantitative items, which improve by 4 to 7 points at the 5 and 10 percent shares and by 12.8 points at the 25 percent share of the formal rendering, while the 125 remaining items move by less than 1 point. The multi-hop question answering datasets improve by 1 to 3 points of F1 (Appendix D.5) and the general benchmarks in Appendix D.1 neither improve nor degrade.

The ability to write derivations that the checker accepts appears at a much smaller share than the accuracy gains. The baseline writes no accepted derivation on BRANCHPROOF, whereas the model trained with 1 percent derivations writes one on 93 percent of problems at chain length 25 and on 90 percent at chain length 45, which is 1.8 times the longest chain in training, and this ability is essentially complete from 1 percent upward in both renderings. Accuracy on the other benchmarks improves by about 1 point at 1 percent and reaches most of its value at 5 percent, and beyond that share the two renderings differ. The English rendering peaks at 10 percent on every benchmark, whereas the formal rendering keeps improving at 25 percent on ProofWriter at depth 1 and under the reasoning prompt, on GPQA-Diamond and on 2WikiMultihopQA. The two effects therefore have different thresholds. Appendices D.4 and D.8 resolve FOLIO by gold label and ProofWriter by inference depth. The FORMAL + ENGLISH mixtures gain 3.4 to 4.0 points on average between 1 and 10 percent. Asked for no notation they derive on 25.5 and 15.0 percent of problems at 1 and 5 percent and on every problem from 10 percent upward (Appendix D.7).

Under a requested chain of reasoning on ProofWriter the baseline gains only 0.6 points, the formal rendering reaches 55.8 percent at 25 percent, and the English rendering stays below its direct answer at every fraction of derivations.

Table 2 ablates the size of the instruction-tuning dataset between 25,000 and 200,000 examples. The average accuracy of the baseline drops by about 2 points from 100,000 examples on, mostly on ProofWriter, while the average of the mixtures with derivations varies by less than 2 points across all sizes. Replacing 5 or 10 percent of the examples therefore raises the average at every size, and most with 100,000 examples. Web of lies and 2WikiMQA improve at every size and MuSiQue in the two larger mixtures, while FOLIO, BIG-Bench Hard and GPQA-Diamond move in both directions. The gain does not grow with the absolute number of derivations, since the same 5,000 derivations raise the average by less in the mixture of 50,000 examples than in the mixture of 100,000. The checker accepts nearly every derivation at every size.

4.2 MODEL FAMILIES AND SCALES

Table 3 repeats the experiment on Qwen2.5-3B, Qwen2.5-14B, Llama-3.1-8B and OLMo-2-7B with both renderings. The derivation-writing ability transfers to every Qwen and Llama model. The

Table 2: Accuracy in % of Qwen2.5-7B trained on instruction mixtures of 25,000 to 200,000 examples, in which none (Base), 5% or 10% of the examples are replaced by derivations in the English rendering. The number of derivations therefore grows with the size of the mixture, from 1,250 to 20,000.

	25k			50k			100k			200k		
Benchmark	Base	5%	10%	Base	5%	10%	Base	5%	10%	Base	5%	10%
PW d3	49.4	49.6	**50.2**	47.0	**53.2**	52.8	39.3	48.3	**53.2**	39.6	45.6	**46.0**
PW d5	52.0	**53.0**	51.0	48.8	**53.8**	51.0	41.2	46.2	**51.4**	42.4	45.6	**47.0**
FOLIO	52.2	**54.2**	53.2	52.2	52.2	52.2	51.4	**56.2**	55.0	54.2	52.2	**54.7**
BBH	66.6	**67.4**	62.1	65.8	66.0	**66.7**	65.0	66.6	**66.9**	66.0	**66.6**	65.7
Web of lies	86.0	**91.2**	90.8	85.2	85.2	**89.2**	85.5	84.5	**87.5**	86.4	**88.8**	86.8
GPQA-Diamond	32.8	34.8	**38.4**	35.9	35.9	34.8	31.8	**35.4**	34.0	32.8	32.8	**33.3**
2WikiMQA	36.6	35.8	**38.0**	38.4	37.8	**40.6**	39.7	41.2	**41.6**	41.1	41.6	**42.6**
MuSiQue	29.1	**29.7**	28.9	**30.7**	29.2	28.7	31.1	**34.2**	32.8	30.6	**34.9**	33.4
Average	50.6	**52.0**	51.6	50.5	51.7	**52.0**	48.1	51.6	**52.8**	49.1	51.0	**51.2**
BRANCHPROOF	0.0	99.0	**100.0**	0.0	99.0	**100.0**	0.3	99.7	**100.0**	0.0	100.0	100.0

Table 3: Both renderings, with 10% of the examples replaced by derivations, on two further Qwen scales, Llama-3.1-8B and OLMo-2-7B, each against its own baseline. Accuracy in %.

	BRANCHPROOF			PW d3			FOLIO			GPQA-Diamond		
Model	Base	FORMAL	ENGLISH	Base	FORMAL	ENGLISH	Base	FORMAL	ENGLISH	Base	FORMAL	ENGLISH
Qwen2.5-3B	0.0	96.8	**100.0**	38.2	**42.1**	41.4	40.6	40.1	**43.3**	**29.0**	28.0	28.8
Qwen2.5-7B	0.3	99.3	**100.0**	39.3	50.0	**53.2**	51.4	**57.8**	55.0	31.8	33.8	**34.0**
Qwen2.5-14B	7.5	99.0	**100.0**	33.0	39.2	**43.6**	61.1	**61.6**	58.1	35.9	36.4	**40.4**
Llama-3.1-8B	0.0	99.5	**100.0**	33.3	36.3	**48.8**	**51.2**	48.5	48.3	24.7	25.8	**26.3**
OLMo-2-7B	0.0	0.2	**19.5**	45.8	33.6	**56.6**	45.1	**47.0**	45.8	**29.8**	29.5	29.0

baselines below 14B write no accepted derivation at any chain length, the 14B baseline reaches 7.5 percent at chain length 25 by reasoning in prose, and every Qwen and Llama model trained with derivations writes accepted derivations on 93 to 100 percent of problems at every chain length. OLMo-2 acquires the ability only in part, on about a fifth of the problems at chain length 5 as well as 25, so its shorter context does not explain the gap.

The gain on ProofWriter transfers to every model. With the English rendering it is 3.2 points on Qwen2.5-3B, 10.6 on Qwen2.5-14B, 15.5 on Llama-3.1-8B and 10.8 on OLMo-2-7B, and with the formal rendering it is 3.9, 10.7 and 6.2 points on the 3 Qwen models from 3B to 14B and 3.0 points on Llama-3.1-8B. On Qwen2.5-32B (Table 4) the formal rendering raises ProofWriter accuracy at depths 2 to 5, lowers it at depth 1, and raises GPQA-Diamond accuracy by 2.6 points. The formal rendering lowers ProofWriter accuracy on OLMo-2 by 12.2 points, because it moves an answer distribution that was already close to the labels onto "unknown", an effect Section 5.1 develops in more detail. FOLIO and GPQA-Diamond move by less than 3 points on the other models, except for GPQA-Diamond on Qwen2.5-14B, which improves by 4.5 points.

Table 4: Qwen2.5-32B, accuracy in percent on ProofWriter by depth and on GPQA-Diamond.

	ProofWriter				GPQA
	d1	d2	d3	d5	Diamond
Base	**49.6**	36.6	32.2	22.2	44.4
FORMAL	37.0	**41.4**	**40.8**	**43.4**	**47.0**

4.3 ABLATIONS

We run four ablations on Qwen2.5-7B, (1) changing the instruction corpus, (2) shuffling the lines of each derivation, (3) varying the depth of the training proofs, and (4) training on derivations alone.

A second instruction corpus. Table 5 repeats the experiment with 100,000 single-turn examples of Tulu 3 (Lambert et al., 2025) in place of Dolci. The two corpora leave the baseline with opposite biases. The model tuned on Dolci answers "true" on 73 percent of the ProofWriter items and the model tuned on Tulu 3 answers "false" on 83 percent, while the gold labels are 40 percent true and 43 percent false. Replacing 5 percent of either corpus by formal derivations moves the answers toward the labels in both cases, by 23 points on Dolci and by 6 on Tulu 3, and teaches the derivation

Table 5: The same experiment run on 100,000 single-turn examples of Tulu 3 in place of Dolci, with 5 percent of the examples replaced by formal derivations. Accuracy in %. The answer column gives the label the model chooses most often over the ProofWriter evaluation set and how often it does so, against gold labels that are 40 percent true and 43 percent false.

Instruction corpus	Mixture	Most frequent answer	BRANCHPROOF	PW d3	PW d5	FOLIO
Dolci	baseline	true, 73%	0.3	39.3	41.2	51.4
Dolci	5% FORMAL	true, 50%	**99.3**	**50.0**	**48.9**	**57.8**
Tulu 3	baseline	false, 83%	0.0	53.6	**58.0**	48.8
Tulu 3	5% FORMAL	false, 77%	**66.4**	**54.6**	57.9	**50.7**

procedure on both, with the checker accepting derivations at chain length 25 from the Tulu 3 mixture and none of those the baseline writes. The accuracy that follows is smaller on Tulu 3, where the mixture gains 1.0 point at inference depth 3 and 1.9 on FOLIO against 10.7 and 6.4 on Dolci, since the Tulu 3 baseline is already 14 points more accurate at depth 3.

Shuffled lines. Table 6 shuffles the lines of every training derivation, which preserves every token, the length and the conclusion and destroys only the order in which the facts follow. These runs enter the derivations during midtraining at a 10 percent share (Appendix H) and are compared with a length-matched baseline that replaces the same share by ordinary long documents. On ProofWriter at inference depth 3 the ordered English derivations improve accuracy by 12.6 points over the baseline and the shuffled ones by 7.8, against 1.0 for the length-matched baseline. On FOLIO the ordered derivations gain 4.4 points in English and 5.4 in the formal rendering against 1.5 for the shuffled ones, which yield no accepted derivation. The tokens alone account for most of the ProofWriter gain.

Table 6: Midtraining conditions in which the derivations replace 10 percent of the tokens and every model afterwards receives the same instruction tuning (Appendix H). Accuracy in %. Length-matched documents contain no derivations, and the shuffled condition permutes the lines of each English derivation.

	ProofWriter by inference depth				
midtraining condition	1	2	3	5	FOLIO
baseline	32.8	44.0	44.4	46.0	55.2
length-matched documents	32.2	44.4	45.4	44.0	56.2
formal derivations	35.8	49.4	50.0	50.2	**60.6**
English derivations	**38.8**	**56.2**	**57.0**	**56.4**	59.6
English derivations, shuffled	36.4	50.6	52.2	53.2	56.7

Training depth. Table 7 varies the depth of the training proofs. Trained on 5-hop proofs, the model writes accepted derivations on every 5-hop problem but on only about a tenth of the 15-hop problems with the formal rendering, whereas the model trained on proofs of up to 25 hops stays above 99 percent everywhere. Benchmark accuracy is indifferent to the training depth.

Table 7: Accepted derivations in % of held-out problems by chain length, for models trained on chain lengths up to the training band.

training band	notation	chain 5	chain 15	chain 25
5	formal	99.8	13.3	11.8
5	English	100.0	85.5	55.0
15	formal	100.0	100.0	45.0
15	English	100.0	99.8	**99.5**
25	formal	99.8	100.0	99.3

Derivations alone. Trained on derivations alone, the model writes accepted derivations on every generator problem and scores zero on every benchmark, because it answers every question with the opening token of a derivation (Appendix D.6).

Table 8: Accuracy by claim polarity and gold label on decidable ProofWriter items, pooled over inference depths 1 through 5, formal rendering, in percent.

Share	negated, false $n = 785$	negated, true $n = 67$	positive, false $n = 112$	positive, true $n = 789$	P(answers true)
0%	17.6	99.2	68.3	75.2	73.5
1%	24.1	98.5	64.9	71.6	69.2
5%	50.1	92.0	83.0	68.7	56.5
10%	49.6	94.5	77.4	66.4	55.2
25%	56.9	91.0	67.9	66.8	50.2

5 Discussion

5.1 Answer distribution

ProofWriter presents a theory together with a claim and asks whether the claim is true, false or unknown. Table 8 splits the decidable items by the polarity of the claim and by the gold label. The baseline is almost always right when the claim is true and mostly wrong when the claim is negated and false, where its accuracy is 17.6 percent, because it asserts the claim in three cases out of four regardless of its content. The derivations move the answer distribution toward the gold labels, and the improvement is concentrated in the negated and false claims. Their accuracy rises by 39.3 points to 56.9 percent at the 25 percent share, while the other 3 cells move by less than 5 points. FOLIO shows the same pattern, with the false and uncertain items gaining and the true ones fixed. A log-likelihood probe (Appendix D.9) shows that the change is mostly a shift of the decision margin on negated claims, and two perturbation probes (Appendix D.10) confirm that most added ”false” answers of the formal mixture depend on the premises.

This account also explains the loss on OLMo-2 in Section 4.2. Across the models of Tables 3 and 5, the change in ProofWriter accuracy follows how far the derivations move the answer distribution toward or away from the gold labels. The OLMo-2 baseline is already close to the gold distribution and the formal derivations push it 46 points toward ”unknown”, and its accuracy falls. The English derivations move the distribution toward the labels on every model, and accuracy rises. That distribution can be measured before any derivations are added, so it predicts whether they will help a given model.

5.2 Intermediate values in multi-step tasks

The procedure can also be observed on a benchmark task (further items in Appendix E). Web of lies, the subtask of BIG-Bench Hard closest in form to the generated data, states for five people whether they tell the truth given the previous person, and the answer is obtained by propagating a truth value along the chain. The formal rendering at 10 percent improves it by 12.1 points and the English rendering by at most 3.5 points at any share. The model writes one numbered line per person, and we read the value it writes at every position over all 250 items. The baseline is correct at the third step on 98 percent of items and at the fifth on 86 percent. The model trained with formal derivations is correct on 99 and 98 percent (Figure 2). Every model answers correctly on essentially every item on which all intermediate values are correct, so the retained value accounts for the whole gain.

> **Question.** Vina tells the truth. Helene says Vina lies. Kandi says Helene tells the truth. Jamey says Kandi lies. Ka says Jamey lies. Does Ka tell the truth?
>
> **Baseline.** (1) Vina tells the truth. (2) Since Vina tells the truth and Helene says Vina lies, Helene lies. (3) Since we know from (2) that Helene *tells the truth*, and Kandi says Helene tells the truth, Kandi tells the truth. (4) Jamey lies. (5) Ka tells the truth. So the answer is Yes.
>
> **With 10 percent formal derivations.** (1) Vina tells the truth. (2) Helene lies. (3) Since we know from (2) that Helene lies, and Kandi says Helene tells the truth, Kandi lies. (4) Jamey tells the truth. (5) Ka lies. So the answer is No.

The baseline reads the opposite of its own step 2 at step 3, the failure Figure 2 counts.

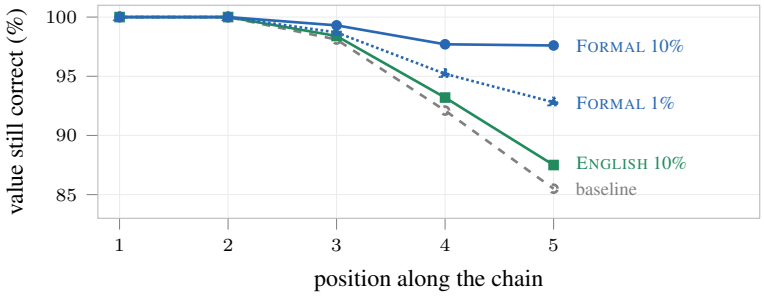

Figure 2: Fraction of web of lies items on which the truth value the model writes is still correct at each position of the chain, for Qwen2.5-7B, in percent, over all 250 items.

Table 12 in Appendix D.3 reads the value at every position for every share, and the object-tracking subtasks behave alike. On an item in which 5 people exchange presents in 5 swaps, all models write out the holder of every present after each swap, and the baseline performs the second swap incorrectly and answers with the present that the wrong state leaves in place, while the trained models track the state correctly through all 5 swaps. The same failure appears outside the multi-step subtasks. On a GPQA-Diamond item that gives the size of a loop diagram as a product containing the factor $1/(4\pi)^6$ and asks how many loops the diagram has, the baseline reads the exponent off the expression and answers 6, whereas the model trained with formal derivations writes that each loop contributes a factor $1/(4\pi)^2$ and answers 3.

5.3 NOTATION

The two renderings are useful under different conditions. On the model we study most closely the formal rendering gives the larger average gain and continues to improve as the share grows, where the English rendering has already reached its best value, and it is also ahead on the benchmarks that most resemble the generator. This advantage does not hold for the other model families. The English rendering improves every model we try, whereas the formal rendering improves some families and harms others, and it is also the English rendering that keeps writing accepted derivations well past the chain lengths it was trained on. We therefore treat the English rendering as the safer default and recommend the formal rendering at a large share only for a family that is known to absorb it well.

6 LIMITATIONS

The mixtures contain at most 200,000 examples, but the gain is present at every size from 25,000 to 200,000 (Table 2). Part of the ProofWriter gain could stem from a format familiar from the generator, yet it lies in negated claims with the gold label false, and the derivations contain no negation and no true-or-false questions. On Qwen2.5-7B accuracy also rises on FOLIO, BIG-Bench Hard and GPQA-Diamond, which differ in format from the generator. A new default answer is also unlikely, since both mixtures answer a claim and its negation identically less often than the baseline and most added "false" answers of the formal mixture vanish without the supporting premises (Appendix D.10). The answer distribution predicts the loss on OLMo-2 before training, although we have not identified which property of OLMo-2 causes it. Reinforcement learning and deduction data written by a language model are left to future work.

7 CONCLUSION

In this paper we showed that deduction data written by a program improves an instruction-tuned model on deduction and multi-step benchmarks when it replaces a small fraction of the instruction mixture. These results give evidence that part of the post-training data can be replaced by data that a program writes and verifies without a teacher model or human annotation. Such data is correct by construction and unlimited in quantity. Arithmetic, the execution of algorithms and planning are further procedures in which a program can write down and check each intermediate state, and each offers training data that requires only a generator.

AI USE STATEMENT

In this work we used generative AI tools for writing and debugging the data generators, the training and evaluation scripts and the analysis code, for tabulating results, and for editing the text of the manuscript under the direction of the authors. We also used them to check the bibliography, and every entry was subsequently verified against its DOI by the authors. We have not used generative AI tools for generating research ideas, and no model-written text or code enters the training data, which is written by the program described in Section 3.1. All AI-assisted code was reviewed and tested by the authors, every number in the paper was checked against the evaluation outputs, and all text was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024. doi: 10.48550/arXiv.2412.08905.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. SmolLM2: When smol goes big – data-centric training of a small language model. In *Proceedings of the Second Conference on Language Modeling (COLM 2025)*, 2025. doi: 10.48550/arXiv.2502.02737.

Cody Blakeney, Mansheej Paul, Brett W. Larsen, Sean Owen, and Jonathan Frankle. Does your data spark joy? performance gains from domain upsampling at the end of training. *arXiv preprint arXiv:2406.03476*, 2024. doi: 10.48550/arXiv.2406.03476.

Jiangjie Chen, Qianyu He, Siyu Yuan, Aili Chen, Zhicheng Cai, Weinan Dai, Hongli Yu, Qiying Yu, Xuefeng Li, Jiaze Chen, et al. Enigmata: Scaling logical reasoning in large language models with synthetic verifiable puzzles. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, pp. 4037–4085, 2025. doi: 10.52202/085713-0128.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. doi: 10.48550/arXiv.2107.03374.

Cheng-Han Chiang and Hung-yi Lee. On the transferability of pre-trained language models: A study from artificial datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10518–10525, 2022. doi: 10.1609/aaai.v36i10.21295.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 3882–3890, 2020. doi: 10.24963/ijcai.2020/537.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168.

Elvis Dohmatob, Yunzhen Feng, Arjun Subramonian, and Julia Kempe. Strong model collapse. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR 2025)*, 2025. doi: 10.48550/arXiv.2410.04840.

Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English? *arXiv preprint arXiv:2305.07759*, 2023. doi: 10.48550/arXiv.2305.07759.

Jiazhan Feng, Ruochen Xu, Junheng Hao, Hiteshi Sharma, Yelong Shen, Dongyan Zhao, and Weizhu Chen. Language models can be deductive solvers. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 4026–4042, 2024. doi: 10.18653/v1/2024.findings-naacl.254.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023. doi: 10.48550/arXiv.2306.11644.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander R. Fabbri, Wojciech Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, 2024. doi: 10.18653/v1/2024.emnlp-main.1229.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R. Devon Hjelm, Alessandro Sordoni, and Aaron Courville. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 1301–1312, 2021. doi: 10.18653/v1/2021.naacl-main.102.

Michael Y. Hu, Jackson Petty, Chuan Shi, William Merrill, and Tal Linzen. Between circuits and Chomsky: Pre-pretraining on formal languages imparts linguistic biases. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9691–9709, 2025. doi: 10.18653/v1/2025.acl-long.478.

Liangze Jiang, Zachary Shinnick, Anton van den Hengel, Hemanth Saratchandran, and Damien Teney. Procedural pretraining: Warming up language models with abstract data. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. doi: 10.48550/arXiv.2601.21725.

Anmol Kabra, Yilun Yin, Albert Gong, Kamilė Stankevičiūtė, Dongyoung Go, Johann Lee, Katie Z. Luo, Carla P. Gomes, and Kilian Q. Weinberger. Learning from synthetic data improves multi-hop reasoning. In *The Fourteenth International Conference on Learning Representations (ICLR 2026)*, 2026. doi: 10.48550/arXiv.2603.02091.

Feiyang Kang, Newsha Ardalani, Michael Kuchnik, Youssef Emad, Mostafa Elhoushi, Shubhabrata Sengupta, Shang-Wen Li, Ramya Raghavendra, Ruoxi Jia, and Carole-Jean Wu. Demystifying synthetic data in LLM pre-training: A systematic study of scaling laws, benefits, and pitfalls. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10750–10769, 2025. doi: 10.18653/v1/2025.emnlp-main.544.

Nora Kassner and Hinrich Schütze. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7811–7818, 2020. doi: 10.18653/v1/2020.acl-main.698.

Kundan Krishna, Jeffrey Bigham, and Zachary C. Lipton. Does pretraining for summarization require knowledge transfer? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3178–3189, 2021. doi: 10.18653/v1/2021.findings-emnlp.273.

Valentin Lacombe, Valentin Quesnel, and Damien Sileo. Reasoning core: A scalable procedural data generation suite for symbolic pre-training and post-training. *arXiv preprint arXiv:2603.02208*, 2026. doi: 10.48550/arXiv.2603.02208.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. TÜLU 3: Pushing frontiers in open language model post-training. In *Proceedings of the Second Conference on Language Modeling (COLM 2025)*, 2025. doi: 10.48550/arXiv.2411.15124.

Emmy Liu, Graham Neubig, and Chenyan Xiong. Midtraining bridges pretraining and posttraining distributions. *arXiv preprint arXiv:2510.14865*, 2025a. doi: 10.48550/arXiv.2510.14865.

Junteng Liu, Yuanxiang Fan, Zhuo Jiang, Han Ding, Yongyi Hu, Chi Zhang, Yiqi Shi, Shitong Weng, Aili Chen, Shiqi Chen, et al. SynLogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, pp. 111997–112018, 2025b. doi: 10.52202/085713-3380.

Pratyush Maini, Skyler Seto, Richard Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14044–14072, 2024. doi: 10.18653/v1/2024.acl-long.757.

Pratyush Maini, Vineeth Dorna, Parth Doshi, Aldo Carranza, Fan Pan, Jack Urbanek, Paul Burstein, Alex Fang, Alvin Deng, Amro Abbas, et al. BeyondWeb: Lessons from scaling synthetic data for trillion-scale pretraining. *arXiv preprint arXiv:2508.10975*, 2025. doi: 10.48550/arXiv.2508.10975.

Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. Learning deductive reasoning from synthetic corpus based on formal logic. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25254–25274. PMLR, 2023. doi: 10.48550/arXiv.2308.07336.

Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. Enhancing reasoning capabilities of LLMs via principled synthetic logic corpus. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pp. 73572–73604, 2024. doi: 10.52202/079017-2340.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022. doi: 10.52202/068431-2011.

Isabel Papadimitriou and Dan Jurafsky. Learning music helps you read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6829–6839, 2020. doi: 10.18653/v1/2020.emnlp-main.554.

Laura Ruis, Maximilian Mozes, Juhan Bae, Siddhartha Rao Kamalakara, Dwarak Talupuru, Acyr Locatelli, Robert Kirk, Tim Rocktäschel, Edward Grefenstette, and Max Bartolo. Procedural knowledge in pretraining drives reasoning in large language models. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR 2025)*, 2025. doi: 10.48550/arXiv.2411.12580.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023. doi: 10.48550/arXiv.2210.01240.

Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi, Najoung Kim, and He He. Testing the general deductive reasoning capacity of large language models using OOD examples. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pp. 3083–3105, 2023. doi: 10.52202/075280-0136.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*, pp. 63612–63655, 2025. doi: 10.52202/085713-1913.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-CC: Transforming Common Crawl into a refined long-horizon pretraining dataset. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2459–2475, 2025. doi: 10.18653/v1/2025.acl-long.123.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, 2021. doi: 10.18653/v1/2021.findings-acl.317.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 OLMo 2 furious. In *Proceedings of the Second Conference on Language Modeling (COLM 2025)*, 2025. doi: 10.48550/arXiv.2501.00656.

Thinh Hung Truong, Timothy Baldwin, Karin Verspoor, and Trevor Cohn. Language models are not naysayers: An analysis of language models on negation benchmarks. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics*, pp. 101–114, 2023. doi: 10.18653/v1/2023.starsem-1.10.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Position: Will we run out of data? limits of LLM scaling based on human-generated data. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 49523–49544. PMLR, 2024. doi: 10.48550/arXiv.2211.04325.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. OctoThinker: Mid-training incentivizes reinforcement learning scaling. *arXiv preprint arXiv:2506.20512*, 2025. doi: 10.48550/arXiv.2506.20512.

Yuhuai Wu, Felix Li, and Percy Liang. Insights into pre-training via simpler synthetic tasks. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pp. 21844–21857, 2022. doi: 10.52202/068431-1588.

Yuan Xia, Akanksha Atrey, Fadoua Khmaissia, and Kedar S. Namjoshi. Can large language models learn formal logic? a data-driven training and evaluation framework. *arXiv preprint arXiv:2504.20213*, 2025. doi: 10.48550/arXiv.2504.20213.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-RL: Unleashing LLM reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025. doi: 10.48550/arXiv.2502.14768.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candès, and Tatsunori Hashimoto. Synthetic continued pretraining. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR 2025)*, 2025. doi: 10.48550/arXiv.2409.07431.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems 38*, pp. 64304–64339, 2025. doi: 10.52202/085713-1933.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. On the paradox of learning to reason from data. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, pp. 3365–3373, 2023. doi: 10.24963/ijcai.2023/375.

Yujun Zhou, Jiayi Ye, Zipeng Ling, Yufei Han, Yue Huang, Haomin Zhuang, Zhenwen Liang, Kehan Guo, Taicheng Guo, Xiangqi Wang, et al. Dissecting logical reasoning in LLMs: A fine-grained evaluation and supervision study. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 17075–17098, 2025. doi: 10.18653/v1/2025.findings-emnlp.926.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. DyVal: Dynamic evaluation of large language models for reasoning tasks. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR 2024)*, 2024. doi: 10.48550/arXiv.2309.17167.

A GENERATED DATA

Generator. The generator samples a target path, assigns labels to constants and predicates, then adds distractor branches that remain locally coherent but do not derive the queried conclusion. In pseudocode:

```
sample target answer label
sample initial object state
for depth step in 1..D:
    sample active branch rule(s)
    derive next object state
    add coherent distractor rules for inactive branches
render one prompt
render the same latent proof as formal and English targets
```

Uniqueness of the answer. Every metric compares a generated answer against one labelled answer, which is only meaningful if the labelled answer is the sole one derivable from the premises. Each example introduces fresh constants, so no constant is reused across layers and no two branches can derive the same atom through reuse of a state predicate. A preflight check computes the full Horn closure of 1,000 training instances and 2,000 evaluation instances with a solver independent of the generator and records the number of candidate final states derivable in each. The unique-answer rate is 1.0, the maximum number of derivable candidates is 1, there are no generation failures, and answer positions are balanced across the four candidates.

Trace grammar and checker. Formal traces contain four blocks, constants, predicates, premises and proof lines, followed by a conclusion and an answer. A proof line contains a formula and a rule justification such as premise retrieval or implication elimination. The checker parses the generated blocks and verifies that each proof line is licensed by earlier lines and premises. English traces use controlled sentences for the same premises and proof steps, and a deterministic translator maps those sentences back into the checker. The formal surface is:

```
<formal>
<constants> CONST_DEF+ </constants>
<predicates> PRED_DEF+ </predicates>
<premises> FORMULA+ </premises>
<proof> PROOF_LINE+ </proof>
<conclusion> ATOM </conclusion>
<answer> LABEL </answer>

CONST_DEF  ::= CONST "=" LABEL
PRED_DEF   ::= PRED "x:" "x is" LABEL
ATOM       ::= PRED CONST
FORMULA    ::= ATOM | ATOM "&" ATOM "->" ATOM | ATOM "->" ATOM
PROOF_LINE ::= FORMULA "; R," REF
             | ATOM "; ->E," REF "," REF
             | ATOM "&" ATOM "; &I," REF "," REF
```

References must point to earlier proof lines or listed premises, so a line cannot justify a future step. Answer extraction searches for the final `<answer>` field, and a generation without a well-formed final answer is counted incorrect. Common checker failures are malformed block structure, undeclared symbols, references to unavailable lines, applying implication elimination to the wrong antecedent, and stopping before the queried conclusion. English validation adds one further failure mode, in which the controlled sentence must translate back into a formal step before the checker can evaluate it.

A paired example. The prompt and the answer are identical within each pair. Training examples contain more distractor premises and longer derivations.

`Prompt:`

```
a is cobalt. a is east.
If a is east and a is cobalt, then b is maple.
If b is maple, then b is south.
If b is south and b is maple, then c is cedar.
What is c?
```

FORMAL target

```
<formal>
<constants> a = a b = b c = c </constants>
<predicates> Cx: x is cobalt Ex: x is east
Mx: x is maple Sx: x is south Dx: x is cedar
</predicates>
<premises> Ca Ea
Ea & Ca -> Mb
Mb -> Sb
Sb & Mb -> Dc </premises>
<proof> Ca ; R,1
Ea ; R,2
Ea & Ca ; &I,7,6
Mb ; ->E,3,8
Sb ; ->E,4,9
Sb & Mb ; &I,10,9
Dc ; ->E,5,11 </proof>
<conclusion> Dc </conclusion>
<answer> cedar </answer>
```

ENGLISH target

```
<think>
a is cobalt. a is east.
Since a is east and a is cobalt, b is maple.
Since b is maple, b is south.
Since b is south and b is maple, c is cedar.
Therefore, c is cedar.
</think>
<answer> cedar </answer>
```

B TRAINING DETAILS

Every model trains for one epoch on 10^5 examples, except in the size ablation of Table 2, full-parameter under fully sharded data parallelism with gradient checkpointing, in bfloat16, with a global batch of 128 sequences, a maximum sequence length of 8,192, warmup ratio 0.03 and learning rate 5×10^{-6}. The learning rate is the one used by our earlier instruction-tuning runs on the same corpus and is held fixed across conditions. The 3 runs per configuration on Qwen2.5-7B differ only in the random seed, and only the final checkpoint is kept. Before training starts, the mixture is tokenized and the examples with a target that would be cut by the sequence length are counted. The run aborts if that count exceeds 0.5 percent, which prevents a condition from training on derivations with their conclusions removed. No run aborted. Qwen2.5-32B and the second-corpus runs use the same configuration, the former on two nodes.

C EVALUATION DETAILS

Prompts and scoring. ProofWriter and FOLIO are prompted with the theory, the claim and the question whether the claim is true, false or unknown (uncertain for FOLIO), and the answer is the first label in the generation. The reasoning prompt for ProofWriter asks the model to reason step by step and to give the final answer on its own last line. GPQA-Diamond uses the four-option mirror of the benchmark and asks the model to work through the options briefly and end with "Answer: X". BIG-Bench Hard ships as a chain-of-thought few-shot task with an answer extractor that reads a plain continuation, and applying the chat template to its shots makes the extractor match nothing for every condition, so this one suite is run without the template. The LongBench multi-hop datasets use the standard prompt and report F1. BRANCHPROOF prompts the model in the format of its training documents and scores the fraction of items for which the checker accepts the derivation and the answer is correct.

Benchmark subsets. The chain subtasks of BIG-Bench Hard are web of lies, tracking shuffled objects (three, five and seven objects), logical deduction (three, five and seven objects) and formal fallacies. GPQA-Diamond contains questions that require computing a quantity through several dependent steps and questions that test recall of a concept, and only the first kind involves a procedure of the kind the derivations teach. We separate the two with a rule that reads only the question text, so that neither the gold answers nor the outputs of any model enter the split, and anyone can recompute the subset from the public dataset. An item is quantitative if its question, matched without regard to case, contains a number followed by a unit,

```
\b\d+(\.\d+)?\s*(nm|mm|cm|km|kg|mol|ml|eV|keV|MeV|GeV|K|Hz|MHz|GHz
```

```
    |deg|°|%|pc|AU|M_|R_)
```

or contains a digit and matches

```
\b(what is the (value|ratio|factor|energy|wavelength|number|probability
    |concentration|pH|mass|radius|velocity|temperature)|calculate|compute
    |how many|by what factor|what fraction)\b
```

where the line breaks are for display only. The rule assigns 73 of the 198 items to the quantitative subset and 125 to the remainder. The sign test compares, for each item, the pooled runs trained with derivations against the pooled baseline runs.

Overlap audit. No evaluation item of ProofWriter, FOLIO or the BIG-Bench Hard subtasks shares an eight-gram with any training derivation.

D Additional Results

D.1 General benchmarks

Table 9 reports nine general benchmarks. GSM8K is five-shot with a chain of thought, HumanEval and MBPP report pass@1 from a single greedy sample, and MMLU, ARC-Challenge, LogiQA, HellaSwag, PIQA and WinoGrande are scored by log-likelihood over the answer options with the default prompts of the evaluation harness. The 3 runs of the Qwen2.5-7B baseline span 0.3 points on GSM8K, 0.1 on MMLU, 0.8 on ARC-Challenge, 0.6 on LogiQA, 0.0 on HellaSwag, 0.4 on PIQA and 1.0 on WinoGrande. Rows for the English rendering at 25 percent and some code rows are still being evaluated.

Table 9: General benchmarks for Qwen2.5-7B, in percent. GSM8K is five-shot with a chain of thought, HumanEval and MBPP report pass@1, and the remaining benchmarks are scored by log-likelihood over the answer options.

		Formal				English			
Benchmark	Baseline	1%	5%	10%	25%	1%	5%	10%	25%
GSM8K	77.0	76.5	76.7	76.2	76.8	76.5	75.7	76.4	78.2
MMLU	70.3	70.2	69.7	70.0	69.5	70.2	70.2	70.3	69.9
ARC-C	49.6	52.4	50.7	50.4	49.8	51.1	51.9	49.9	50.7
LogiQA	36.8	38.0	36.8	38.9	38.4	37.5	36.6	38.5	37.6
HellaSwag	73.2	73.4	73.2	73.2	73.6	73.3	73.6	73.5	73.5
PIQA	80.5	81.0	80.8	80.9	80.7	80.6	81.0	80.8	80.5
WinoGrande	71.0	70.6	70.3	71.0	71.3	71.3	70.2	70.7	71.4
HumanEval	66.8	67.1	–	68.3	–	65.9	63.4	69.5	65.2
MBPP	65.2	64.8	–	65.2	–	65.2	64.6	64.8	66.4

D.2 BIG-BENCH HARD BY SUBTASK

Table 10: BIG-Bench Hard resolved by subtask, accuracy in percent. "Chain" pools the 8 subtasks in which the answer follows from propagating a truth value or a state along a sequence of steps and "others" pools the remaining 19.

Notation	Share	macro	chain (8)	others (19)	web of lies
none	0%	65.0	62.2	66.1	85.5
formal	1%	66.2	66.1	66.3	92.7
formal	5%	67.2	67.6	67.1	95.9
formal	10%	**67.7**	**68.9**	67.2	**97.6**
formal	25%	66.9	67.6	66.5	98.0
English	1%	65.3	63.2	66.1	84.1
English	5%	66.6	65.2	67.2	84.5
English	10%	66.9	66.4	67.2	87.5
English	25%	66.6	65.6	67.0	87.6

D.3 WEB OF LIES BY GOLD ANSWER

Table 11 resolves web of lies by gold answer and by the number of negating links, and Table 12 reads the intermediate value at every position of the chain of thought.

Table 11: Web of lies, the BIG-Bench Hard subtask that chains assertions about who tells the truth, accuracy in percent on 250 items, 48.8 percent of which have the gold answer "yes". A negating link is a person who says that the previous person lies.

Notation	Share	P(answers yes)	gold yes	gold no	3 negating links	accuracy
none	0%	56.7	93.2	78.1	71.7	85.5
formal	1%	50.0	93.7	91.7	88.8	92.7
formal	5%	48.4	95.4	96.4	95.7	95.9
formal	10%	48.5	97.3	**97.9**	**97.3**	**97.6**
English	1%	56.9	92.1	76.6	69.0	84.1
English	5%	57.1	92.6	76.8	69.0	84.5
English	10%	53.9	92.3	82.8	77.5	87.5

Table 12: Intermediate values in web of lies. The model writes 1 numbered line per person, so the truth value it writes can be read at each position and compared with the value the chain implies. The last two columns give the final answer accuracy when every intermediate value is correct and when at least one is not.

		value still correct at position				all steps	final answer	
Notation	Share	2	3	4	5	correct	steps ok	steps wrong
none	0%	100.0	98.1	92.1	85.5	84.9	100.0	3.5
formal	1%	100.0	98.7	95.2	92.8	92.4	99.9	5.3
formal	5%	100.0	98.7	96.5	95.9	95.5	100.0	8.8
formal	10%	100.0	99.3	97.7	**97.6**	**97.5**	100.0	5.3
English	1%	100.0	97.9	91.9	84.1	83.6	100.0	3.3
English	5%	100.0	97.3	91.9	84.5	84.1	100.0	2.5
English	10%	100.0	98.4	93.2	87.5	87.1	100.0	3.1

D.4 FOLIO BY GOLD LABEL

Table 13: FOLIO by gold label, accuracy in percent. 35.5 percent of the evaluation items are true and the baseline answers "true" on 51.7 percent.

Notation	Share	false	true	uncertain	overall
none	0%	32.3	80.1	38.6	51.4
formal	1%	32.3	80.6	43.0	53.0
formal	5%	**45.2**	80.1	45.9	**57.8**
formal	10%	37.1	79.2	**52.2**	57.1
formal	25%	40.3	76.4	49.3	56.2
English	1%	32.3	83.8	42.5	54.0
English	5%	44.1	78.7	43.5	56.2
English	10%	42.5	75.5	44.9	55.0

D.5 MULTI-HOP QUESTION ANSWERING

Table 14: Multi-hop question answering, F1 in percent. The 3 benchmarks retrieve from supplied passages and ask a question that requires 2 or 3 hops.

Notation	Share	HotpotQA	2WikiMultihopQA	MuSiQue
none	0%	55.5	39.7	31.1
formal	1%	54.6	39.7	31.3
formal	5%	56.1	41.3	32.7
formal	10%	54.8	41.3	31.0
formal	25%	55.2	40.3	31.6
English	1%	55.4	40.0	32.2
English	5%	55.4	41.2	**34.2**
English	10%	**56.6**	**41.6**	32.8
range over baseline runs		0.1	1.1	1.5

D.6 DERIVATIONS WITHOUT INSTRUCTION DATA

Table 15 gives the ablation of Section 4.3 that removes the instruction data.

Table 15: Training on derivations without instruction data. Asked a benchmark question, these models emit the opening token of the derivation format and stop.

training data	BRANCHPROOF	PW d3	PW d5	FOLIO
instruction data only	0.3	39.3	41.2	51.4
5 percent FORMAL in instruction data	99.3	50.0	48.9	57.8
FORMAL derivations only	100.0	0.0	0.0	0.0
ENGLISH derivations only	100.0	0.0	0.0	0.0

D.7 BOTH RENDERINGS IN ONE MIXTURE

Every mixture in the main text contains a single rendering of the derivations. Table 16 reports mixtures that contain both renderings of the same derivations, in two forms. The plain form interleaves the two renderings with no marker of which one a given derivation uses. The conditioned form prepends one sentence to each training prompt that names the notation the derivation will use, and we evaluate it with the same two sentences and with no sentence at all. We train the plain form at shares of 2, 5, 10 and 20 percent and the conditioned form at 2, 5, 10 and 20 percent, on Dolci and on the same base model as Table 5.

Table 16: Mixtures that contain both renderings of the same derivations, with and without a notation instruction in training. The first three columns give the percentage of held-out generator problems at chain length 25 for which the checker accepts the derivation, under prompts that ask for no particular notation, for a formal derivation and for an English one.

	BRANCHPROOF by prompt					
Mixture	none	formal	English	PW d3	PW d5	FOLIO
baseline	0.3	–	–	39.3	41.2	51.4
5% FORMAL	99.3	–	–	50.0	48.9	57.8
2% both renderings	92.5	1.5	99.0	47.0	47.0	51.2
5% both renderings	95.0	96.5	100.0	40.8	42.2	54.2
10% both renderings	99.5	98.5	99.5	52.6	51.0	56.7
20% both renderings	100.0	99.0	100.0	49.6	49.4	53.2
2% both, conditioned	20.0	97.0	99.5	47.0	46.8	50.7
5% both, conditioned	15.0	96.5	100.0	43.2	43.6	52.7
10% both, conditioned	99.0	98.0	99.0	48.6	44.8	54.7
20% both, conditioned	100.0	100.0	100.0	47.2	47.4	53.7

Transfer follows the share and not the conditioning. The plain mixture at 10 percent gains 13.3 points on ProofWriter at inference depth 3, 9.8 points at depth 5 and 5.3 points on FOLIO over the baseline, which is at the level of the 5 percent FORMAL mixture, while the plain mixtures at 2 and 5 percent gain less and are not ordered by their share. The plain mixture at 20 percent gains 10.3 points at inference depth 3 and 1.8 points on FOLIO, below the 10 percent mixture on both. At the matched share of 5 percent the conditioned mixture is 2.4 points above the plain one at inference depth 3 and 1.5 points below it on FOLIO, at the matched share of 10 percent it is 4.0 points below it at inference depth 3 and 2.0 points below it on FOLIO, and at the matched share of 20 percent it is 2.4 points below it at inference depth 3 and 0.5 points above it on FOLIO.

With the notation instruction, the model writes the notation that the prompt asks for. The conditioned models write a derivation the checker accepts under either instruction, at 96.5 to 100 percent in both renderings and at all four shares. At 2 and 5 percent they write almost no accepted derivation when neither instruction is present, at 20.0 and 15.0 percent, while at 10 and 20 percent they write one at 99.0 and 100.0 percent, so the instruction is required for the notation at the larger shares and not for the derivation itself. The plain model at 2 percent writes an accepted derivation when it is left to choose, and it also follows the formal instruction in form, with a formal block in 100 percent of its answers at chain length 25, but the checker accepts 1.5 percent of those derivations. Training on both renderings without a marker therefore leaves a model that can imitate the notation it is asked for without being able to derive in it, which the one added sentence repairs. The failure is confined to the smallest share, and the plain mixtures at 5, 10 and 20 percent derive under the formal instruction at 96.5 percent and above.

D.8 PROOFWRITER BY INFERENCE DEPTH

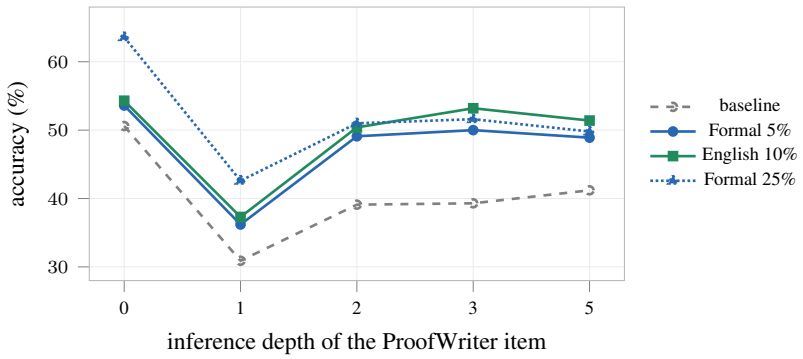

Figure 3: Accuracy on ProofWriter for Qwen2.5-7B by the inference depth of the item, in percent.

D.9 LOG-LIKELIHOOD PROBE

Table 17: Ranking of decidable ProofWriter items by the log-likelihood margin between "true" and "false" on Qwen2.5-7B, area under the ranking curve (AUC) and the probability of preferring "true", both in percent, for 1 run with a second run in parentheses. The pooled column mixes the two polarities and the next two columns rank within a polarity. LONGDOC is a length-matched baseline that replaces the same share of the mixture with ordinary long documents.

Mixture	AUC pooled	AUC negated	AUC positive	$P(\text{argmax} = \text{true})$
BASELINE	66.2 (65.4)	87.2 (86.9)	86.0 (85.8)	64.0 (63.0)
LONGDOC	66.3 (66.4)	87.6 (87.7)	86.7 (86.7)	64.0 (64.0)
FORMAL	71.1 (70.7)	88.9 (88.9)	86.8 (86.8)	55.0 (54.0)
ENGLISH	77.0	89.8	87.3	48.0

D.10 PERTURBATION PROBES

The first probe toggles the polarity of the claim and leaves the theory untouched, so a model that ignores negation answers both versions identically. The baseline does so on 37.6 percent of decidable items and the length-matched baseline on 37.6 percent, against 28.5 percent for the formal and 24.2 percent for the English mixture, with a second run within half a point throughout. Every model answers the positive version of a negated and false item correctly about 68 percent of the time, so the models separate on the negated version alone.

The second probe removes the premises that support the claim, which makes the gold label unknown, and asks whether the additional "false" answers survive. They largely do not, which is what a premise-grounded answer predicts. The probability that the formal mixture answers "false" on these items falls from 53.0 to 27.0 percent once the supporting facts are gone, and the baseline falls from 35.0 to 20.0 percent. The English mixture retains more, falling from 67.0 to 46.0 percent, so about two thirds of its advantage on this cell survives the removal of the evidence for it. The formal rendering therefore gives a smaller gain that is more firmly tied to the premises, and the English rendering a larger gain of which part is a shifted default for negated claims.

E QUALITATIVE EXAMPLES

Each example below is an item that the Qwen2.5-7B baseline answers incorrectly and that the models trained with 10 percent formal and 10 percent English derivations both answer correctly, all from the same run of each condition. The responses are reproduced verbatim apart from the omission of repeated prefixes.

ProofWriter. The theory is "The bear needs the dog. The cow is kind. The cow is nice. The cow likes the dog. The cow needs the dog. The dog chases the rabbit. The dog likes the bear. The dog likes the rabbit. The rabbit chases the bear. The rabbit chases the cow. The rabbit is blue. The rabbit is kind. The rabbit likes the cow. The rabbit needs the dog. If someone needs the rabbit then the rabbit likes the bear. If someone needs the rabbit and they chase the dog then they need the bear. If someone is kind then they like the dog. If someone likes the bear and they are blue then they chase the rabbit. If the rabbit likes the dog then the dog is kind. If someone chases the dog then the dog likes the cow. If someone likes the dog then they are round. If the rabbit likes the dog and the rabbit chases the cow then the dog likes the cow. If someone needs the dog and the dog is kind then the dog needs the rabbit." The claim is "The dog does not need the rabbit", which is false, since the rabbit is kind, therefore likes the dog, therefore the dog is kind, and the rabbit needs the dog, so the dog needs the rabbit. The baseline answers "True" and both trained models answer "False". The item in Section 5.1 has the same structure.

FOLIO. The premises are "A Japanese game company created the game the Legend of Zelda. All games on the Top 10 list are made by Japanese game companies. If a game sells more than one million copies, then it will be included in the Top 10 list. The Legend of Zelda sold more than one million copies." The claim "The Legend of Zelda is not on the Top 10 list" is false. The baseline

answers "Uncertain" and both trained models answer "False". A second item has the premises "Beethoven leads the Vienna Music Society. Vienna Music Society is an orchestra. Orchestras are led by conductors." and the claim "Beethoven is not a conductor". The baseline answers "True" and both trained models answer "False".

Web of lies. A second item beyond the one in Section 5.2.

> **Question.** Jim lies. Yoland says Jim lies. Willian says Yoland tells the truth. Delbert says Willian lies. Shalonda says Delbert tells the truth. Does Shalonda tell the truth?
>
> **Baseline.** (1) Jim lies. (2) Since Jim lies and Yoland says Jim lies, Yoland tells the truth. (3) Willian tells the truth. (4) Since Willian tells the truth and Delbert says Willian lies, Delbert lies. (5) Since we know from (4) that Delbert *tells the truth*, and Shalonda says Delbert tells the truth, Shalonda tells the truth. So the answer is Yes.
>
> **With derivations.** Steps (1) to (4) as above. (5) Since we know from (4) that Delbert lies, and Shalonda says Delbert tells the truth, Shalonda lies. So the answer is No.

Tracking shuffled objects. Beyond the item in Section 5.2, on an item with 5 holders and 5 swaps the baseline reaches the correct final state and then selects the wrong option letter, which the trained models do not.

GPQA-Diamond. An uncharged spherical conductor of radius R contains an off-centre cavity with a charge $+q$ inside, and the question asks for the field at a point P outside the conductor at distance L from the centre of the conductor and l from the centre of the cavity. The baseline writes that the field "is the same as if the charge were concentrated at the center of the cavity" and answers $q/(4\pi\epsilon_0 l^2)$. The trained models write that the conductor shields the position of the charge, so the field is that of a charge at the centre of the conductor, and answer $q/(4\pi\epsilon_0 L^2)$. Section 5.2 gives a second item from this benchmark.

F Verification of Derivations

A derivation in the trained format is checked by the same program that wrote the training data. On the generator's own problems, a derivation the checker accepts contains the correct answer in 99 percent of samples, so selecting the first accepted derivation out of 16 matches the accuracy of selecting with the gold answer. On ProofWriter the checker reproduces the published label on 84 to 99 percent of items depending on the depth, and an accepted derivation is correct between 53 and 69 percent of the time, because a third of the items ask whether a claim is underivable and a forward chain cannot show that. The coupling between validity and correctness therefore holds where the checker is complete for the task, which makes the trained format a candidate for verifier-based training on deduction (Lambert et al., 2025; Yue et al., 2025), although we do not pursue this here.

G Notation Crossover on the Generator

An experiment on the generator problems alone shows that the ordering of the two renderings in depth generalisation depends on the depth of the training proofs. We fine-tune OLMo-3-7B with LoRA adapters on 50,000 paired examples per condition, one condition per rendering, for 10,000 optimizer steps, and we evaluate at chain lengths beyond the longest one seen in training with 16 samples per prompt and 1 greedy decode. Figure 4 gives the resulting curves. English supervision generalises further as long as the training proofs reach at most 20 hops. At 25 hops the ordering reverses, and formal supervision reaches 75.0 ± 9.5 percent out-of-distribution accuracy against 17.8 ± 4.7 percent for English supervision. Sixteen samples move the reversal to 15 hops, so formal supervision first widens the set of problems the model can reach and only later improves its first attempt.

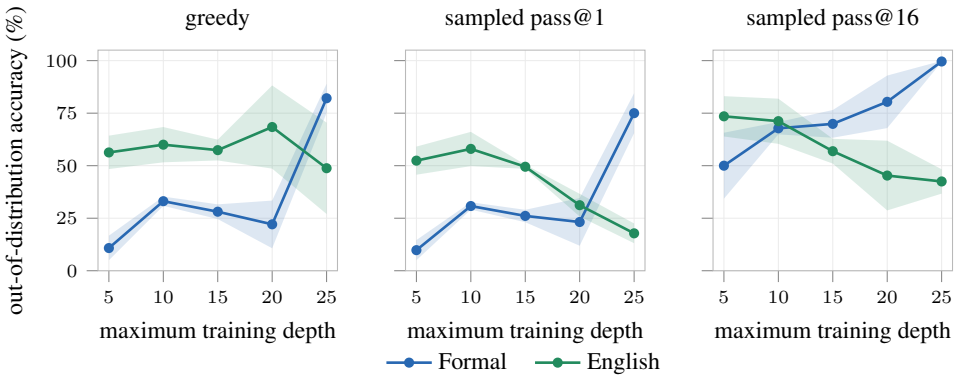

Figure 4: Out-of-distribution answer accuracy on BRANCHPROOF against maximum training depth, for three decoding rules. Shaded regions give one standard deviation. Greedy decoding and pass@1 place the crossover at maximum training depth 25. Sampling with sixteen draws places it at 15, ten depth levels earlier.

The crossover follows from the supervision alone, since the prompt, the answer and the latent proof are identical in the two conditions at every depth. It survives a baseline that equates the number of supervised target tokens by subsampling the longer condition, a baseline that removes a schema marker which identifies the active branch on 80 percent of training examples (out-of-distribution pass@1 of 43.1 ± 7.7 percent for FORMAL and 78.5 ± 14.9 for ENGLISH), and a replication with a second run. Concatenating both renderings into one target fails, with 0.5 ± 0.4 percent out-of-distribution pass@1, whereas placing an output-mode tag on separate examples reaches 50.3 ± 6.4 percent in formal mode at 7B and 85.0 ± 8.7 at 32B.

H MIDTRAINING

The same derivations can enter during midtraining. We midtrained Qwen2.5-7B on 2.5 billion tokens, 2,385 steps at sequence length 8,192 with a global batch of 128 sequences, with a tenth of the loss tokens replaced by 72,000 generated documents packed without splitting, and then applied the same instruction tuning to every condition. The conditions are the baseline, a length-matched baseline that replaces the same tokens by ordinary long documents (LONGDOC), and the formal and English renderings, with 1 midtraining run per condition. Table 18 reports the results. That route also produces the gain on ProofWriter, 12.6 points for the English rendering at inference depth 3, which a second instruction-tuning run reproduces to within 0.1 points, and the gain again lies in the items with the gold label false. It also leaves the general benchmarks flat apart from a loss of 2 to 3 points on LogiQA that the second run reproduces. Midtraining is the more expensive route and gives no further gain, since the instruction-tuning route reaches the same accuracy, generalises further on long chains, does not lower LogiQA accuracy, and adds nothing when the two are applied in sequence. The midtrained base models write checker-valid derivations on every generator problem up to chain length 25 when prompted in the format of their training documents, and the baseline base writes none. Ordinary instruction tuning removes this ability, and the lower half of Table 18 shows what remains of it after instruction tuning, where the derivations raise greedy accuracy on the generator's problems by 20 to 50 points and pass@16 by 20 to 40 points.

Table 18: Midtraining on Qwen2.5-7B followed by instruction tuning without derivations, in percent, 1 run per condition. The upper part gives ProofWriter accuracy at each inference depth on 500 items per depth, and the accuracy on the items pooled over depths 1 to 5 by gold label, together with the fraction of pooled items on which the model answers "true". The lower part gives accuracy on held-out generator problems at chain lengths 5 to 25, 200 items per length, under greedy decoding and as pass@16 at temperature 0.8.

	ProofWriter by inference depth					pooled depths 1 to 5			
Condition	0	1	2	3	5	answers true	gold true	gold false	two-class
BASELINE	44.2	32.8	44.0	44.4	46.0	66.3	70.4	38.5	54.4
LONGDOC	45.8	32.2	44.4	45.4	44.0	65.9	69.5	38.6	54.0
FORMAL	48.0	35.8	49.4	50.0	50.2	56.4	66.5	54.2	60.3
ENGLISH	**48.8**	**38.8**	**56.2**	**57.0**	**56.4**	49.5	68.3	**68.0**	**68.2**

		generator problems by chain length				
Decoding	Condition	5	10	15	20	25
greedy	BASELINE	5.0	3.5	11.0	9.0	6.5
greedy	LONGDOC	7.5	11.0	12.0	15.0	12.0
greedy	FORMAL	31.5	17.5	27.5	21.5	19.5
greedy	ENGLISH	**52.5**	**34.5**	**30.5**	**24.5**	**29.5**
pass@16	BASELINE	58.0	55.5	58.5	60.0	53.0
pass@16	LONGDOC	63.0	63.5	61.0	63.5	56.0
pass@16	FORMAL	**97.0**	85.5	79.0	74.0	71.5
pass@16	ENGLISH	95.5	**91.5**	**84.5**	**78.0**	**82.5**