Δ_1–RSG: SUPPORT-CRITICAL REASONING SUPERVISION FROM SELF-CERTIFYING THEOREM STRUCTURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) can often derive plausible conclusions from available premises, yet remain prone to asserting conclusions when indispensable support is missing. Existing reasoning supervision primarily emphasizes whether a conclusion follows, providing much weaker supervision about the boundary between sufficient and insufficient evidence. We investigate self-certifying theorem structures generated by the Δ_1 automated theorem generator as a formally grounded source of LLM reasoning supervision, and introduce Δ_1-based Reasoning Supervision Generation (Δ_1–RSG) to exploit their support-critical structure. Each Δ_1 theorem has an exact support boundary: its complete premise set entails the conclusion, whereas removing any indispensable premise makes the available support insufficient. Δ_1–RSG deterministically compiles this structure into complementary supervision for conclusion inference, support sufficiency, support-based abstention, missing-support identification, and support repair, with labels justified by construction rather than external theorem verification. Experiments with Qwen2-1.5B show that self-certifying theorem supervision substantially improves controlled reasoning, while full Δ_1–RSG further develops support-critical behaviour. On structurally held-out theorem configurations, Δ_1–RSG achieves 98.9% inference accuracy and 100% support-sufficiency accuracy while reducing the unsupported conclusion rate from 70% for the base model to 1%. It further achieves 97% F1 for missing-support identification and 99.5% F1 for support repair. These results demonstrate that self-certifying theorem structures can provide effective, formally grounded supervision for learning both what follows from available evidence and when that evidence is insufficient to license a conclusion.

Keywords: Large Language Models, Reasoning Supervision, Automated Theorem Generation, Formal Reasoning, Support-Critical Reasoning

1 INTRODUCTION

Large language models (LLMs) increasingly acquire reasoning abilities from synthetic solutions, chain-of-thought (CoT) traces, and self-generated reasoning data Wei et al. (2022); Wang et al. (2023); Zelikman et al. (2022). Process supervision and verifier-based methods further improve the quality of such data by assessing final answers or intermediate reasoning steps Cobbe et al. (2021); Lightman et al. (2023); Wang et al. (2024); Rizvi et al. (2026); Kamoi et al. (2026). These approaches can scale reasoning supervision beyond manually authored examples, but they raise a fundamental question: how can the reasoning examples used for supervision themselves be trusted? A common strategy is to generate candidate reasoning and subsequently score, filter, or formally verify it Cobbe et al. (2021); Lightman et al. (2023); Yang et al. (2026); Kamoi et al. (2026).

A second issue receives less attention. Correct positive examples teach a model what follows when the required evidence is present, but do not necessarily teach when the available evidence is *insufficient*. A reliable reasoner should both derive a conclusion from adequate premises and withhold that conclusion when indispensable support is absent. Reasoning supervision should therefore characterize not only valid entailments, but also the boundary at which their evidential support is lost.

We investigate whether automated theorem generation can provide such supervision directly. Our starting point is Δ_1, a deterministic automated theorem generator based on minimally unsatisfiable

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.

full triangular standard contradictions (FTSCs) Xu et al. (2025). Given admissible logical units, Δ_1 generates non-redundant and non-equivalent theorem structures whose validity follows from their construction. More importantly for supervision, the minimal-unsatisfiability structure gives each generated theorem (Γ, q) an exact support boundary:

$$\Gamma \models q, \qquad A \not\models q \quad \text{for every } A \subsetneq \Gamma.$$

Thus, the complete premise set supports the conclusion, whereas removing any indispensable premise produces a formally characterized state of insufficient support. Importantly, $A \not\models q$ does not imply $A \models \neg q$: the appropriate response is therefore to withhold an unsupported conclusion, not to infer its negation.

This observation gives Δ_1 theorem structures two roles in LLM supervision. First, complete-support theorems provide self-certifying positive reasoning examples. Second, their support boundary enables richer supervision about when a conclusion is justified and what is missing when it is not. Building on this structure, we introduce Δ_1-based Reasoning Supervision Generation (Δ_1–RSG), a deterministic compiler that transforms each source theorem into five complementary supervision families: conclusion inference, support sufficiency, support-based abstention, missing-support identification, and support repair. Because the symbolic targets are obtained from the theorem structure by construction, they retain formal justification and provenance without requiring a separate theorem-verification stage.

We evaluate these two roles using the same Qwen2-1.5B backbone under three conditions: the base model, conclusion-only fine-tuning on Δ_1 theorems, and full Δ_1–RSG supervision. The evaluation includes unseen generated instances and, more stringently, held-out theorem structures. The results show that Δ_1 theorem supervision alone substantially improves conclusion inference, while full Δ_1–RSG additionally develops strong support-critical behaviour. On held-out theorem structures, full Δ_1–RSG achieves 98.9% inference accuracy and 100% support-sufficiency accuracy while reducing unsupported conclusions from 70% to 1%; it also achieves 97% F1 for missing-support identification and 99.5% F1 for support repair.

Our contributions are:

- We identify self-certifying Δ_1 theorem structures as a formally grounded source of LLM reasoning supervision, providing both valid positive entailments and exact boundaries between sufficient and insufficient support.
- We introduce Δ_1–RSG, which deterministically compiles these support-critical structures into supervision for conclusion inference, support sufficiency, abstention, missing-support identification, and support repair while preserving formal justification and provenance.
- We provide controlled empirical evidence that Δ_1 theorem supervision substantially improves inference and that full Δ_1–RSG learns the intended support-critical behaviour, including strong generalization to held-out theorem structures.

2 Related Work

Reasoning supervision and verification. Chain-of-thought prompting, self-consistency, and self-generated rationales have established the value of explicit reasoning traces for LLMs Wei et al. (2022); Wang et al. (2023); Zelikman et al. (2022). Subsequent work has increasingly focused on improving the reliability of such supervision through outcome or process verification Cobbe et al. (2021); Lightman et al. (2023); Wang et al. (2024); Rizvi et al. (2026); Kamoi et al. (2026). Formal machinery provides stronger correctness signals: FoVer verifies natural-language reasoning using first-order logic Pei et al. (2025), ORACLE verifies LLM-generated intermediate reasoning using symbolic constraints Yang et al. (2026), and ProverGen combines LLM generation with symbolic provers to construct verified logical-reasoning data Qi et al. (2025). These approaches establish the value of verification for reasoning supervision. Our focus is complementary: Δ_1–RSG starts from self-certifying theorem structures and derives supervision not only for valid conclusions, but also for the exact boundary between sufficient and insufficient support.

Formal reasoning and theorem generation. LLMs have also been studied in formal reasoning and theorem-proving settings, including neural theorem proving Polu & Sutskever (2020), formal-statement curriculum learning Polu et al. (2023), miniF2F Zheng et al. (2022), Lean-based theorem

proving Yang et al. (2023), and neuro-symbolic reasoning systems such as AlphaGeometry Trinh et al. (2024). Controlled benchmarks such as RuleTaker and ProofWriter study deductive reasoning from explicitly represented facts and rules Clark et al. (2020); Tafjord et al. (2021). Rather than proving externally supplied conjectures, we use automatically generated theorem structures as training supervision.

Automated theorem generation constructs theorem structures rather than only proving supplied conjectures Zhang & Tan (2026); Xu et al. (2025). Δ_1 deterministically generates non-redundant and non-equivalent theorem structures from minimally unsatisfiable FTSCs Xu et al. (2025). Minimal unsatisfiability—removing any member restores satisfiability—is well established in logic and SAT Aharoni & Linial (1986); Liffiton & Sakallah (2005); Zhang et al. (2024); Lymperopoulos & Liu (2024); Bleukx et al. (2025); Ijima & Yawata (2026). We do not claim this property as a new logical result. Instead, Δ_1–RSG operationalizes it as a supervision principle: removing indispensable support yields a formally characterized insufficient-support state, from which sufficiency, abstention, missing-support, and repair targets can be generated by construction.

3 The Δ_1 Theorem Foundation

Δ_1 is a deterministic automated theorem generator based on full triangular standard contradictions (FTSCs) Xu et al. (2025). We use Δ_1 within its established input setting and summarize only the properties needed for reasoning supervision.

Let

$$L = \{x_1, \ldots, x_n\}$$

be a finite set of admissible literals in first-order logic. For a fixed ordering of these literals, Δ_1 constructs

$$S = \{D_1, \ldots, D_{n+1}\},$$

where

$$\begin{aligned} D_1 &= x_1, \\ D_i &= x_i \vee \neg x_1 \vee \cdots \vee \neg x_{i-1}, \quad 2 \le i \le n, \\ D_{n+1} &= \neg x_1 \vee \cdots \vee \neg x_n. \end{aligned} \tag{1}$$

The resulting FTSC is minimally unsatisfiable: S is unsatisfiable, while removing any $D_i \in S$ restores satisfiability Xu et al. (2025).

A Δ_1 theorem is obtained by selecting a component D_i and defining

$$\Gamma = S \setminus \{D_i\}, \qquad q = \neg D_i. \tag{2}$$

The Δ_1 construction gives

$$\Gamma \vdash q,$$

and hence, by soundness of classical first-order logic, $\Gamma \models q$. Thus the theorem is obtained by construction rather than by generating a conjecture and subsequently verifying it.

Different orderings of the admissible literals yield the established family of non-redundant and non-equivalent Δ_1 theorem structures Xu et al. (2025). We do not modify or re-claim these properties. Our contribution begins by using these self-certifying theorem structures as reasoning supervision and, crucially, exploiting their minimal-unsatisfiability structure to characterize the boundary of support for q.

4 Δ_1–RSG: Support-Critical Reasoning Supervision

Δ_1–RSG converts self-certifying Δ_1 theorems into supervision that describes not only what follows from complete premises, but also when the available premises cease to be sufficient. Let

$$T_k = (\Gamma_k, q_k, \tau_k)$$

denote a generated Δ_1 theorem, where τ_k records its construction provenance. Figure 1 summarizes the framework.

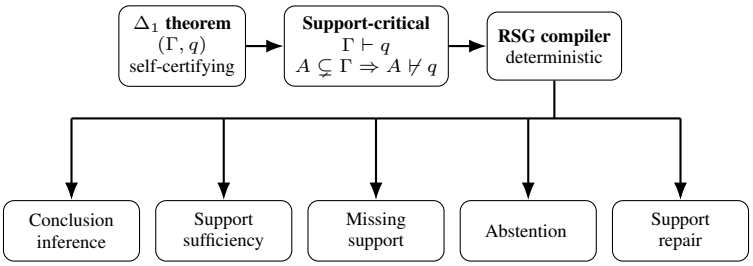

Figure 1: Overview of Δ_1–RSG. A self-certifying Δ_1 theorem induces an exact support boundary: the complete premise set entails the conclusion, whereas every proper subset is insufficient. The RSG compiler deterministically transforms this structure into five complementary reasoning-supervision tasks.

Table 1: Core supervision compiled from a support-critical theorem (Γ_k, q_k), where $A \subseteq \Gamma_k$ denotes the available premises and $M_k(A) = \Gamma_k \setminus A$ the missing support.

Task	Available support	Question	Target
Conclusion inference	$A = \Gamma_k$	What follows?	q_k
Support sufficiency	$A \subseteq \Gamma_k$	Is q_k supported?	Yes iff $A = \Gamma_k$
Missing-support identification	$A \subsetneq \Gamma_k$	What is missing?	$M_k(A)$
Abstention	$A \subsetneq \Gamma_k$	Can q_k be concluded?	Withhold
Support repair	$A \subsetneq \Gamma_k$	How can support be restored?	Add $M_k(A)$

4.1 Support-Critical Structure

Definition. An entailment (Γ, q) is *support-critical* if

$$\Gamma \models q \quad \text{and} \quad \forall A \subsetneq \Gamma,\ A \not\models q. \tag{3}$$

Proposition 1 (Support-criticality). Every Δ_1 theorem (Γ_k, q_k) used by RSG is support-critical.

Proof sketch. Let D_i be the component removed from the minimally unsatisfiable FTSC S, so that $\Gamma_k = S \setminus \{D_i\}$ and $q_k = \neg D_i$. The Δ_1 construction gives $\Gamma_k \models q_k$. For any $A \subsetneq \Gamma_k$, choose $D_j \in \Gamma_k \setminus A$. Minimal unsatisfiability implies that $S \setminus \{D_j\}$ is satisfiable, and $A \cup \{D_i\} \subseteq S \setminus \{D_j\}$. Hence $A \cup \{D_i\}$ is satisfiable and therefore $A \not\models \neg D_i = q_k$. The full proof is given in Appendix A.

The proposition is a consequence of minimal unsatisfiability rather than a new general logical result. Its significance here is that each source theorem provides an automatically labelled support boundary. Moreover,

$$A \not\models q_k \ \not\Rightarrow\ A \models \neg q_k,$$

so an incomplete support state should be labelled *insufficient*, rather than false or contradictory.

For $A \subseteq \Gamma_k$, define its missing support and support deficit as

$$M_k(A) = \Gamma_k \setminus A, \qquad d(A, \Gamma_k) = |M_k(A)|. \tag{4}$$

The complete state $A = \Gamma_k$ is sufficient; every proper subset is insufficient. This gives exact targets for both identifying missing support and restoring it.

4.2 Reasoning-Supervision Compiler

For each source theorem T_k, Δ_1–RSG applies a deterministic compiler

$$C_{\mathrm{RSG}}(T_k) = \{z_k^{(1)}, \ldots, z_k^{(J_k)}\},$$

where each supervision instance records its input, target, task family, and source-theorem provenance. Table 1 summarizes the five supervision families.

Conclusion inference. Complete support provides conventional positive reasoning supervision, $\Gamma_k \mapsto q_k$.

Support sufficiency and abstention. For $A \subseteq \Gamma_k$, the sufficiency target is

$$y_{\text{suff}}(A) = \begin{cases} \textsc{Sufficient}, & A = \Gamma_k, \\ \textsc{Insufficient}, & A \subsetneq \Gamma_k. \end{cases}$$

When $A \subsetneq \Gamma_k$, abstention supervision additionally trains the model to withhold q_k. Sufficiency asks whether the evidence licenses the conclusion; abstention specifies the desired reasoning behaviour when it does not.

Missing support and repair. For every incomplete state $A \subsetneq \Gamma_k$, the exact missing-support target is $M_k(A) = \Gamma_k \setminus A$. The same set provides a constructive repair target because

$$A \cup M_k(A) = \Gamma_k.$$

Thus missing-support supervision diagnoses why the evidence is insufficient, whereas repair supervision identifies what must be restored.

This gives the compact supervision pipeline:

$$\Delta_1 \text{ theorem} \longrightarrow \text{support boundary} \longrightarrow \text{multi-task reasoning supervision}.$$

4.3 Corpus Construction and Assurance Boundary

For a theorem collection $\mathcal{T}$, the RSG corpus is

$$\mathcal{D}_{\text{RSG}} = \bigcup_{T_k \in \mathcal{T}} C_{\text{RSG}}(T_k).$$

All instances derived from the same source theorem are assigned to the same data partition to prevent theorem-level leakage. Each instance retains its source identity and Δ_1 construction provenance.

The formal guarantee applies to the symbolic supervision: its sufficiency, insufficiency, missing-support, and repair labels follow from semantics-preserving transformations of the source theorem structure. It does not imply that the trained neural model is formally correct, nor does it automatically extend to unrestricted natural-language formalization. For LLM supervision, the symbolic instances are therefore rendered into controlled natural-language representations that preserve the underlying logical structure, support state, and supervision target. Appendix E gives the rendering procedure and an end-to-end example.

5 Experimental Setup

Our experiments address two research questions: **RQ1:** Does supervision from self-certifying Δ_1 theorems improve conclusion inference? **RQ2:** Does full Δ_1–RSG additionally teach models to recognize and respond appropriately to insufficient support? We investigate these questions through controlled comparisons using the same Qwen2-1.5B backbone, with particular attention to generalization to held-out theorem structures.

5.1 Dataset and Tasks

The dataset is generated automatically from Δ_1 theorem structures using the `contradiction-gen` package (v0.3.0). The resulting corpus contains 10,000 instances, partitioned into 7,000 training, 2,000 validation, and 1,000 ID test instances. We additionally construct a separate 1,000-instance S-OOD test set used only for evaluation.

The corpus contains the five supervision families introduced in Section 4.2: conclusion inference, support sufficiency, support-based abstention, missing-support identification, and support repair. Inference instances require prediction of the conclusion from complete support. Sufficiency instances classify the available support as SUFFICIENT or INSUFFICIENT, while abstention instances require

ANSWER or WITHHOLD. Missing-support and repair instances require identification of the premise absent from an incomplete support state and the premise that must be restored, respectively.

For insufficient-support instances, we consider the immediate support boundary ($r = 1$): exactly one indispensable premise is removed from the complete support set. Thus, each insufficient-support instance differs from its corresponding sufficient state by a single required support element. The task is not to recognize severely incomplete evidence, but to change behaviour when a single logically necessary support element is removed. This makes $r = 1$ a focused test of sensitivity to the support boundary while avoiding confounding the experiment with varying degrees of support deficit.

All tasks use structured outputs, enabling deterministic evaluation of model predictions.

5.2 EVALUATION SPLITS

We evaluate two controlled generalization regimes.

1) **In-distribution (ID).** ID evaluation uses unseen source theorems generated from structural configurations represented during training. The evaluation theorems themselves are disjoint from the training and validation theorems.

2) **Structural out-of-distribution (S-OOD).** S-OOD evaluation uses source theorems generated from structural configurations that are entirely withheld from training and validation. A structural configuration is defined by the underlying theorem-generation template, independent of variable identities and surface rendering. Consequently, S-OOD tests whether the learned reasoning behaviour transfers to previously unseen theorem structures rather than merely to new instances of structures encountered during training.

Both regimes use the same task definitions and the same immediate support-boundary setting ($r = 1$), so that the principal difference between ID and S-OOD is whether the underlying theorem structure was represented during training.

5.3 CONTROLLED TRAINING CONDITIONS

We compare three conditions using the same underlying model:

Base. The pretrained Qwen2-1.5B model is evaluated without task-specific fine-tuning.

Conclusion-SFT. The model is fine-tuned using only complete-support conclusion-inference instances. This condition tests whether self-certifying Δ_1 theorems alone provide useful positive reasoning supervision, without explicitly training the model on insufficient-support states.

Full Δ_1–RSG. The model is fine-tuned with the complete RSG supervision comprising all five task families. Comparing this condition with Conclusion-SFT tests the additional value of support-critical supervision beyond positive theorem inference alone.

This progression,

$$\text{Base} \longrightarrow \text{Conclusion-SFT} \longrightarrow \text{Full } \Delta_1\text{–RSG},$$

separates ordinary positive theorem supervision from the richer support-critical supervision introduced by RSG.

5.4 MODEL AND TRAINING

All experiments use Qwen2-1.5B as the base language model. Fine-tuning is performed with LoRA using rank 16, scaling parameter $\alpha = 32$, and dropout 0.05. LoRA adapters are applied to the attention projections (q, k, v, and o) and the MLP projections (gate, up, and down).

Models are trained for three epochs using AdamW with a learning rate of 10^{-4} and cosine decay. The per-device batch size is 8 with two gradient-accumulation steps, giving an effective batch size of 16. Training uses bfloat16 precision, gradient checkpointing, a maximum sequence length of 384 tokens, 200 warm-up steps, and a maximum gradient norm of 1.0. Further implementation details are provided in Appendix D.

Table 2: Main results on in-distribution (ID) and structural out-of-distribution (S-OOD) evaluation sets. Higher is better for Acc and Suff; lower is better for UCR.

Method	**ID**			**S-OOD**		
	Acc↑	**Suff↑**	**UCR↓**	**Acc↑**	**Suff↑**	**UCR↓**
Base LLM	64.31	38.31	60.00	33.10	35.00	70.00
Conclusion-SFT	96.19	55.16	35.00	75.10	45.12	45.25
Full Δ_1–RSG	**99.89**	**100.00**	**0.70**	**98.90**	**100.00**	**1.00**

5.5 Evaluation

We evaluate both conclusion correctness and behaviour under insufficient support. *Inference accuracy* measures whether the model predicts the correct conclusion when complete support is available. *Sufficiency accuracy* measures whether it correctly distinguishes sufficient from insufficient support. *Abstention accuracy* measures whether the model answers when support is complete and withholds when it is insufficient.

Our principal behavioural metric is the *Unsupported Conclusion Rate* (UCR):

$$\mathrm{UCR} = \frac{\#\{(A, q_k) : A \subsetneq \Gamma_k \text{ and the model asserts } q_k\}}{\#\{(A, q_k) : A \subsetneq \Gamma_k\}}. \tag{5}$$

Lower UCR is better. We interpret inference accuracy and UCR jointly: a support-sensitive model should derive the conclusion when complete support is present while avoiding the same conclusion when indispensable support is missing.

For the diagnostic tasks, we report exact match (EM) and F1 for missing-support identification and support repair. Together, these metrics test whether the model merely recognizes insufficiency or can also identify and restore the missing support.

6 Results and Analysis

6.1 Main Results

Table 2 summarizes the principal controlled results. We focus on two comparisons: 1) Base versus Conclusion-SFT tests the effect of positive supervision from self-certifying Δ_1 theorems; and 2) Conclusion-SFT versus Full Δ_1–RSG tests the additional effect of support-critical supervision.

Three observations stand out.

1) **Self-certifying theorem supervision substantially improves conclusion inference.** Conclusion-SFT increases accuracy from 64.31% to 96.19% on ID and from 33.10% to 75.10% on S-OOD, providing evidence that automatically generated Δ_1 theorems are an effective source of positive reasoning supervision.

2) **Positive theorem supervision alone does not fully teach the support boundary.** Although Conclusion-SFT substantially improves inference, its S-OOD sufficiency accuracy is only 45.12%, with a UCR of 45.25%. Thus, learning to derive valid conclusions does not by itself ensure that the model recognizes when indispensable support is absent.

3) **Full Δ_1–RSG develops the intended support-critical behaviour.** It achieves 99.89%/98.90% inference accuracy and 100%/100% sufficiency accuracy on ID/S-OOD, while reducing UCR to 0.70%/1.00%, respectively. The combination of high inference accuracy and low UCR is important: the model derives conclusions under complete support while largely withholding them when indispensable support is missing, including on held-out theorem structures. A model that always answers could obtain useful inference accuracy while over-asserting under insufficient support, whereas a model that always withholds could obtain a low UCR without useful inference. The desired support-critical behaviour requires both high Acc and low UCR.

Table 3: Missing-support identification and support-repair results for Full Δ_1–RSG on the S-OOD test set (%).

Method	Miss EM↑	Miss F1↑	Repair EM↑	Repair F1↑
Full Δ_1–RSG	95.0	97.0	99.0	99.5

6.2 Missing-Support Identification and Repair

Beyond recognizing that support is insufficient, Δ_1–RSG is designed to identify what support is missing and how sufficiency can be restored. Table 3 reports the corresponding diagnostic and repair results.

Full Δ_1–RSG achieves 95.0% exact match and 97.0% F1 for missing-support identification, showing that it can usually identify the premise absent from an insufficient support state. Performance is still higher for support repair, reaching 99.0% exact match and 99.5% F1.

These results complement the sufficiency and UCR results in Table 2. The learned behaviour is therefore not limited to detecting that support is insufficient. The model can usually localize the indispensable premise whose absence causes the support failure and identify the corresponding repair. This distinction is important: support-critical reasoning requires not only a decision to withhold, but also a representation of why the current support is insufficient and what would restore sufficiency.

6.3 From Positive Reasoning Supervision to Support-Critical Behaviour

The controlled comparison reveals a distinction between learning to derive a conclusion and learning the conditions under which that conclusion is licensed. Conclusion-SFT receives only positive complete-support theorems. This supervision is already highly effective for inference: relative to the Base model, accuracy increases from 64.31% to 96.19% on ID and from 33.10% to 75.10% on S-OOD. Thus, self-certifying Δ_1 theorems alone provide useful reasoning supervision, including substantial transfer to structurally held-out theorem instances.

However, improved positive inference does not automatically produce support-sensitive behaviour. On S-OOD, Conclusion-SFT reaches 75.10% inference accuracy but only 45.12% sufficiency accuracy, and still asserts unsupported conclusions in 45.25% of insufficient-support cases. This separation supports the motivation for RSG: supervision about what follows from complete evidence is not equivalent to supervision about where the support boundary lies.

Full Δ_1–RSG explicitly exposes both sides of this boundary. Its 98.90% S-OOD inference accuracy is accompanied by 100% sufficiency accuracy and a UCR of only 1.00%. The improvement is therefore not simply a shift toward answering more often or withholding more often. Instead, the model learns different behaviour on two support states that, in the $r = 1$ setting, differ by only one indispensable premise:

$$\Gamma_k \models q_k \qquad \text{versus} \qquad \Gamma_k \setminus \{P_{k,j}\} \not\models q_k.$$

Together with the missing-support and repair results, this suggests that the principal benefit of Δ_1–RSG lies in coupling successful inference with sensitivity to the support conditions that license that inference.

7 Limitations

Our study has several limitations.

1) **Controlled reasoning setting.** The experiments use automatically generated, semantics-preserving representations derived from Δ_1 theorem structures. This enables precise control over support sufficiency and missing support, but does not establish that the same behaviour will transfer to unrestricted natural-language reasoning, where formalization itself may be ambiguous or incorrect.

2) **Scope of formal assurance.** The self-certifying property applies to the symbolic theorem structures and the supervision labels derived from them, not to the trained neural model. Δ_1–RSG therefore

provides formally grounded supervision, but does not make the resulting LLM formally verified or guarantee correct behaviour on arbitrary inputs.

3) **Evaluation scope.** The present experiments use a single 1.5B-parameter model and evaluate in-distribution and structurally held-out Δ_1 theorem structures. Consequently, the results do not yet establish how the observed effects scale with model size, architecture, or substantially different reasoning domains.

4) **Residual shortcut risk.** Although S-OOD evaluation tests generalization to held-out theorem structures, the data are generated by a common formal construction and may retain regularities that models can exploit. The current experiments therefore do not rule out all forms of shortcut learning. Stronger distribution shifts and independently constructed reasoning problems would provide a more demanding test of whether the learned support-critical behaviour generalizes beyond the Δ_1 generation process.

8 Conclusion

We introduced Δ_1–RSG, a framework that uses self-certifying Δ_1 theorem structures as a formally grounded source of reasoning supervision. The key idea is to exploit their exact support boundary: complete premises entail the conclusion, whereas removing indispensable support yields a formally characterized insufficient-support state. Δ_1–RSG compiles this structure into complementary supervision for conclusion inference, support sufficiency, abstention, missing-support identification, and support repair.

Controlled experiments with Qwen2-1.5B show two complementary effects. Supervision from Δ_1 theorems substantially improves conclusion inference, while full Δ_1–RSG additionally develops support-critical behaviour. On S-OOD theorem structures, Full Δ_1–RSG achieves 98.90% inference accuracy and 100% sufficiency accuracy while reducing UCR to 1.00%; it also achieves 97.0% F1 for missing-support identification and 99.5% F1 for support repair. These results support the use of self-certifying theorem structures not only for teaching models what follows from available evidence, but also for teaching when that evidence is insufficient and what support is missing.

Reproducibility Statement. Appendices A–E provide the formal derivation, RSG supervision templates, dataset-construction and split protocol, training configuration, evaluation metrics, and an illustrative rendering example. Code for Δ_1 generation, RSG compilation, training, and evaluation will be released upon publication.

AI Use Statement

Generative AI tools were used to assist with language editing, clarity, and drafting and revising portions of the manuscript. The theoretical framework, methodology, dataset-generation procedures, experiments, mathematical results, analysis, and scientific conclusions were developed and verified by the authors. All AI-assisted text was critically reviewed and edited by the authors, who take full responsibility for the final content.

References

Ron Aharoni and Nathan Linial. Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. *Journal of Combinatorial Theory, Series A*, 43(2):196–204, 1986. doi: 10.1016/0097-3165(86)90060-9.

Ignace Bleukx, Hélène Verhaeghe, Bart Bogaerts, and Tias Guns. Exploiting symmetries in MUS computation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 11122–11130, 2025. doi: 10.1609/aaai.v39i11.33209.

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, Rei Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Hiroya Ijima and Koichiro Yawata. Hypergraph neural networks accelerate MUS enumeration. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 1828–1836. PMLR, 2026.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Sarkar Snigdha Sarathi Das, Ranran Haoran Zhang, Wenpeng Yin, and Rui Zhang. Efficient PRM training data synthesis via formal verification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 8246–8265. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.403.

Mark H. Liffiton and Karem A. Sakallah. On finding all minimally unsatisfiable subformulas. In *Theory and Applications of Satisfiability Testing*, volume 3569 of *Lecture Notes in Computer Science*, pp. 173–186. Springer, 2005. doi: 10.1007/11499107_13.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Panagiotis Lymperopoulos and Liping Liu. Graph pruning for enumeration of minimal unsatisfiable subsets. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li (eds.), *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 2647–2655. PMLR, 2024.

Yu Pei, Yongping Du, and Xingnan Jin. FoVer: First-order logic verification for natural language reasoning. *Transactions of the Association for Computational Linguistics*, 13:1340–1359, 2025. doi: 10.1162/tacl.a.41.

Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*, 2020.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. In *International Conference on Learning Representations*, 2023. doi: arXivpreprintarXiv:2202.01344.

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. doi: arXivpreprintarXiv:2502.06563.

Md Imbesat Hassan Rizvi, Xiaodan Zhu, and Iryna Gurevych. SPARE: Single-pass annotation with reference-guided evaluation for automatic process supervision and reward modelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32808–32816, 2026. doi: 10.1609/aaai.v40i39.40560.

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. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.findings-acl.317.

Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482, 2024. doi: 10.1038/s41586-023-06747-5.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.510.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. doi: arXivpreprintarXiv: 2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1800.

Yang Xu, Hailin Guo, Shuwei Chen, and Jun Liu. Δ_1: An automated theorem generator. *International Journal of Computational Intelligence Systems*, 18(1):309, 2025. doi: 10.1007/s44196-025-01035-8.

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Zhuojie Yang, Wentao Wan, and Keze Wang. ORACLE: Optimizing reasoning abilities of large language models via constraint-led synthetic data elicitation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34378–34386, 2026. doi: 10.1609/aaai.v40i40.40735.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Jian Zhang and Si-Cheng Tan. Automated conjecturing and theorem finding: A survey. *Journal of Computer Science and Technology*, 41(1):46–66, 2026. doi: 10.1007/s11390-026-6040-0.

Tianwei Zhang, Tomáš Peitl, and Stefan Szeider. Small unsatisfiable k-cnfs with bounded literal occurrence. In *27th International Conference on Theory and Applications of Satisfiability Testing (SAT 2024)*, volume 305 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 31:1–31:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi: 10.4230/LIPIcs.SAT.2024.31.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. miniF2F: A cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*, 2022.

A Formal Proof of Support-Criticality

Let $S = \{D_1, \ldots, D_m\}$ be the minimally unsatisfiable structure underlying a generated Δ_1 theorem. Select $D_i \in S$ and define

$$\Gamma = S \setminus \{D_i\}, \qquad q = \neg D_i.$$

The Δ_1 construction gives $\Gamma \vdash q$ and hence, by soundness,

$$\Gamma \models q.$$

We show that every proper subset of Γ is insufficient to entail q. Let $A \subsetneq \Gamma$. Because A is a proper subset, there exists some

$$D_j \in \Gamma \setminus A.$$

Since S is minimally unsatisfiable,

$$S \setminus \{D_j\}$$

is satisfiable. Moreover,

$$A \cup \{D_i\} \subseteq S \setminus \{D_j\}.$$

Every subset of a satisfiable set of formulas is satisfiable, so $A \cup \{D_i\}$ is satisfiable. Since $q = \neg D_i$, it follows that

$$A \not\models q.$$

Therefore,

$$\Gamma \models q \qquad \text{and} \qquad \forall A \subsetneq \Gamma,\ A \not\models q,$$

and (Γ, q) is support-critical.

This result is a direct consequence of minimal unsatisfiability; we do not claim it as a new general result about minimally unsatisfiable sets. Δ_1–RSG exploits this property to generate formally characterized reasoning supervision.

B RSG SUPERVISION TEMPLATES AND WORKED EXAMPLE

This appendix specifies the supervision templates used by the Δ_1–RSG compiler and illustrates their construction with a concrete Δ_1 theorem. Let

$$T_k = (\Gamma_k, q_k, \tau_k)$$

denote a generated source theorem, where τ_k records its Δ_1 construction provenance. For an available support state $A \subseteq \Gamma_k$, define

$$M_k(A) = \Gamma_k \setminus A.$$

Each supervision instance retains the identity of its source theorem, allowing the generated supervision to be traced back to the originating Δ_1 structure.

B.1 SUPERVISION TEMPLATES

Conclusion inference. For the complete support state $A = \Gamma_k$, the input contains the complete premise set and requests the supported conclusion:

$$x_{\text{con}} = (\Gamma_k, \textsc{Infer}), \qquad y_{\text{con}} = q_k.$$

Support sufficiency. Given an available support state $A \subseteq \Gamma_k$ and candidate conclusion q_k, the target is

$$y_{\text{suff}}(A, q_k) = \begin{cases} \textsc{Sufficient}, & A = \Gamma_k, \\ \textsc{Insufficient}, & A \subsetneq \Gamma_k. \end{cases}$$

Thus, the task asks whether the available premises logically support the candidate conclusion, rather than whether the conclusion is merely plausible.

Support-based abstention. For an incomplete state $A \subsetneq \Gamma_k$, support-criticality gives

$$A \not\models q_k,$$

and the corresponding target is

$$y_{\text{abs}}(A, q_k) = \textsc{Withhold}.$$

For complete support,

$$y_{\text{abs}}(\Gamma_k, q_k) = \textsc{Answer}.$$

The distinction is semantic rather than confidence-based:

$$A \not\models q_k \;\not\Rightarrow\; A \models \neg q_k.$$

Hence WITHHOLD means that the available premises do not license assertion of q_k; it does not mean that $\neg q_k$ has been established.

Missing-support identification. For $A \subsetneq \Gamma_k$, the target is the exact missing-support set

$$y_{\text{miss}}(A) = M_k(A) = \Gamma_k \setminus A.$$

Premises are assigned canonical identifiers within each theorem so that missing support can be evaluated as set prediction rather than free-form generation.

Support repair. For the same incomplete state, the repair target is

$$y_{\text{rep}}(A) = M_k(A).$$

Its interpretation differs from missing-support identification: the former identifies which support is absent, whereas repair asks which support should be restored. A correct repair therefore satisfies

$$A \cup y_{\text{rep}}(A) = \Gamma_k,$$

after which

$$A \cup y_{\text{rep}}(A) \models q_k.$$

B.2 Canonical Output Representation

To reduce evaluation ambiguity, the compiler assigns each premise in Γ_k a canonical identifier:

$$\Gamma_k = \{P_{k,1}, P_{k,2}, \ldots, P_{k,|\Gamma_k|}\}.$$

Missing-support and repair targets are represented as sets of these identifiers. For example, if

$$A = \Gamma_k \setminus \{P_{k,2}, P_{k,5}\},$$

then

$$M_k(A) = \{P_{k,2}, P_{k,5}\}.$$

The gold target for both missing-support identification and support repair is therefore the canonical set

$$\{P_{k,2}, P_{k,5}\},$$

although the two tasks use different instructions and have different reasoning interpretations. Set-valued outputs are scored independently of output order.

B.3 Worked Example

Consider an illustrative Δ_1 construction from three admissible input literals

$$x_1, x_2, x_3,$$

with the FTSC

$$S = \{D_1, D_2, D_3, D_4\},$$

where

$$\begin{aligned} D_1 &= x_1, \\ D_2 &= x_2 \vee \neg x_1, \\ D_3 &= x_3 \vee \neg x_1 \vee \neg x_2, \\ D_4 &= \neg x_1 \vee \neg x_2 \vee \neg x_3. \end{aligned}$$

Suppose the Δ_1 construction selects D_4. The resulting source theorem T_k has

$$\Gamma_k = S \setminus \{D_4\} = \{D_1, D_2, D_3\}, \qquad q_k = \neg D_4.$$

By construction,

$$\Gamma_k \vdash q_k, \qquad \text{and hence} \qquad \Gamma_k \models q_k.$$

For readability, assign canonical premise identifiers

$$P_{k,1} = D_1, \qquad P_{k,2} = D_2, \qquad P_{k,3} = D_3.$$

Complete support. The complete support state is

$$A_0 = \{P_{k,1}, P_{k,2}, P_{k,3}\} = \Gamma_k.$$

It therefore generates the targets

$$\textsc{Sufficient}, \qquad \textsc{Answer}, \qquad q_k$$

for support sufficiency, abstention behaviour, and conclusion inference, respectively.

Deficit-one support. Now remove one indispensable premise:

$$A_1 = \{P_{k,1}, P_{k,2}\}.$$

Then

$$A_1 \subsetneq \Gamma_k \qquad \text{and} \qquad A_1 \not\models q_k.$$

The corresponding targets are therefore

$$\textsc{Insufficient}, \qquad \textsc{Withhold},$$

with exact missing support

$$M_k(A_1) = \{P_{k,3}\}.$$

The support-repair target is likewise

$$y_{\mathrm{rep}}(A_1) = \{P_{k,3}\},$$

because

$$A_1 \cup \{P_{k,3}\} = \Gamma_k \models q_k.$$

The RSG construction extends analogously to larger support deficits, although the present experiments focus exclusively on $r = 1$.

C Dataset Construction, Sampling, and Splits

This appendix provides additional details of the controlled Δ_1–RSG corpus, including source-theorem generation, support-state construction, dataset partitioning, and the ID and S-OOD evaluation regimes. The final corpus is generated automatically using `contradiction-gen` v0.3.0.

C.1 Δ_1 Source-Theorem Generation

Source theorems are generated using the established Δ_1 procedure. For a generation instance with n admissible input literals

$$L = \{x_1, \ldots, x_n\},$$

Δ_1 constructs an FTSC and generates a theorem according to the procedure described in Section 3. The resulting theorem collection is represented as

$$\mathcal{T} = \{T_k\}_{k=1}^{N}, \qquad T_k = (\Gamma_k, q_k, \tau_k),$$

where k indexes the generated source theorem and τ_k records its Δ_1 construction provenance.

The generator parameter n denotes the number of admissible input literals supplied to Δ_1 and should be distinguished from

$$|\Gamma_k|,$$

the number of premises in the support set of the resulting theorem. Δ_1–RSG support states are defined directly over Γ_k rather than over the generator parameter n.

Each source theorem is subsequently compiled into the five RSG task families: conclusion inference, support sufficiency, support-based abstention, missing-support identification, and support repair.

C.2 Support-State Construction

A source theorem (Γ_k, q_k) induces the support lattice 2^{Γ_k}. The complete state

$$A = \Gamma_k$$

provides sufficient support for q_k, whereas every proper subset $A \subsetneq \Gamma_k$ is insufficient by support-criticality.

For the controlled experiments, we focus on the immediate support boundary by considering incomplete states with support deficit

$$r = d(A, \Gamma_k) = |\Gamma_k \setminus A| = 1.$$

Thus, every insufficient-support instance is obtained by removing exactly one indispensable premise:

$$A = \Gamma_k \setminus \{P_{k,j}\}.$$

Its exact missing support is therefore

$$M_k(A) = \Gamma_k \setminus A = \{P_{k,j}\}.$$

The $r = 1$ setting provides a particularly stringent test of support-sensitive reasoning because the incomplete state differs from complete support by only one indispensable premise. The model must therefore distinguish

$$\Gamma_k \models q_k$$

from

$$\Gamma_k \setminus \{P_{k,j}\} \not\models q_k$$

despite the two states differing by a single support element.

The complete support state is used for positive conclusion supervision, whereas the corresponding $r = 1$ states provide insufficient-support, abstention, missing-support, and repair supervision.

C.3 THEOREM-LEVEL DATASET PARTITIONING

Because a single source theorem can generate multiple RSG supervision instances, partitioning at the level of individual compiled instances could leak closely related support states across training and evaluation. We therefore group supervision instances by their source theorem.

Let

$$\mathcal{T}_{\text{train}}, \mathcal{T}_{\text{val}}, \mathcal{T}_{\text{test}}$$

denote the corresponding source-theorem collections. These collections are mutually disjoint, and the compiled dataset for split s is

$$\mathcal{D}_s = \bigcup_{T_k \in \mathcal{T}_s} C_{\text{RSG}}(T_k), \qquad s \in \{\text{train}, \text{val}, \text{test}\}.$$

Thus, all supervision instances derived from a given source theorem remain within the same partition, preventing theorem-level leakage between different RSG views of that theorem.

C.4 ID AND STRUCTURAL OOD CONSTRUCTION

We construct the ID and S-OOD evaluation sets at the level of source-theorem structure rather than individual RSG instances.

Let

$$c(T_k)$$

denote the structural configuration of source theorem T_k. In the present experiments, $c(T_k)$ is determined by the underlying theorem-generation template and does not depend on variable identities or the surface rendering of the resulting instance. The set of structural configurations is partitioned into configurations available during training and configurations reserved for structural OOD evaluation:

$$\mathcal{C}_{\text{train}} \cap \mathcal{C}_{\text{S-OOD}} = \varnothing.$$

In-distribution evaluation. An ID test theorem T_k is itself unseen during training, but satisfies

$$c(T_k) \in \mathcal{C}_{\text{train}}.$$

Thus, ID evaluates generalization to new theorem instances whose underlying structural configurations are represented in the training distribution.

Structural OOD evaluation. An S-OOD theorem instead satisfies

$$c(T_k) \in \mathcal{C}_{\text{S-OOD}},$$

where

$$c(T_k) \notin \mathcal{C}_{\text{train}}.$$

No source theorem belonging to an S-OOD structural configuration is included in training or validation. All RSG instances compiled from an S-OOD source theorem inherit that structural assignment and are used only for S-OOD evaluation.

This produces a stronger separation than theorem-level partitioning alone. Theorem-level partitioning prevents different RSG views of the same source theorem from leaking across splits, whereas configuration-level partitioning additionally prevents the structural template underlying an S-OOD theorem from appearing during training.

For both ID and S-OOD, insufficient-support states are constructed at the same immediate support boundary,

$$r = |\Gamma_k \setminus A| = 1.$$

The task definitions, RSG compilation procedure, and evaluation metrics are otherwise unchanged. Hence the S-OOD comparison is designed to isolate generalization to held-out theorem structures rather than changes in support deficit or task formulation.

Table 4: Controlled Δ_1–RSG corpus and evaluation-set statistics.

Split	Instances	Role
Training	7,000	Fine-tuning
Validation	2,000	Model development
ID test	1,000	In-distribution evaluation
S-OOD test	1,000	OOD evaluation only

Structural configuration identifier. Each generated source theorem is associated with a structural-template identifier produced by the theorem generator. The identifier records the underlying theorem-generation template independently of variable identities and surface rendering. Two theorems generated from the same template therefore share a structural configuration, whereas S-OOD templates are excluded entirely from training and validation.

C.5 Dataset Statistics

The final controlled corpus contains 10,000 instances generated using `contradiction-gen` v0.3.0, partitioned into 7,000 training, 2,000 validation, and 1,000 ID test instances. A separate 1,000-instance S-OOD evaluation set is generated from held-out structural configurations and is not part of the main corpus. The dataset is partitioned as shown in Table 4.

D Model Training and Evaluation Details

This appendix provides additional implementation details for the controlled experiments reported in the main paper, including the base model, LoRA configuration, optimization settings, task-output formats, and evaluation metrics.

D.1 Model and LoRA Configuration

All experiments use `Qwen/Qwen2-1.5B` with `Qwen2ForCausalLM`. The model has approximately 1.5B parameters, hidden size 1536, 28 Transformer layers, 12 attention heads, 2 key–value heads using grouped-query attention (GQA), vocabulary size 151,936, and RoPE positional encoding with $\theta = 10^6$. Computation uses bfloat16 precision.

Parameter-efficient fine-tuning is performed using LoRA with rank $r_{\mathrm{LoRA}} = 16$, scaling parameter $\alpha = 32$, and dropout 0.05. Adapters are applied to the query, key, value, and output attention projections and to the gate, up, and down MLP projections:

$$\{\mathtt{q}, \mathtt{k}, \mathtt{v}, \mathtt{o}, \mathtt{gate}, \mathtt{up}, \mathtt{down}\}.$$

No bias parameters are trained. This configuration results in approximately 51M trainable parameters, corresponding to about 3.4% of the base model.

D.2 Optimization and Training

Fine-tuning is performed for three epochs using AdamW with an initial learning rate of 10^{-4} and cosine learning-rate scheduling. We use 200 warm-up steps and a maximum gradient norm of 1.0.

The per-device batch size is 8 with two gradient-accumulation steps, giving an effective batch size of 16. Training uses bfloat16 mixed precision, gradient checkpointing, and a maximum sequence length of 384 tokens.

Training logs are recorded every 50 steps. Checkpoints are saved every 500 steps, with at most two checkpoints retained.

D.3 Task Output Formats

Training and evaluation use structured machine-readable outputs for the five RSG task families. The corresponding target fields are:

Task	Target output
Inference	`conclusion`
Sufficiency	`SUFFICIENT/INSUFFICIENT`
Abstention	`ANSWER/WITHHOLD`
Missing support	`missing_premise`
Repair	`premise_to_add`

The structured representation permits deterministic parsing of final model outputs. For missing-support and repair instances, the target identifies the indispensable premise removed from the complete support state. Because the controlled experiments use support deficit $r = 1$, each such instance has exactly one missing premise.

D.4 Evaluation Metrics

We use complementary metrics to distinguish correct inference under complete support from appropriate behaviour when one indispensable premise is removed.

Conclusion accuracy (Acc). For complete-support instances, conclusion accuracy is

$$\mathrm{Acc} = \frac{1}{N_{\mathrm{con}}} \sum_{j=1}^{N_{\mathrm{con}}} \mathbf{1}[\hat{y}_j = y_j].$$

It measures whether the model predicts the correct conclusion when all required support is available.

Support-sufficiency accuracy (Suff). For sufficiency instances,

$$\mathrm{Suff} = \frac{1}{N_{\mathrm{suff}}} \sum_{j=1}^{N_{\mathrm{suff}}} \mathbf{1}[\hat{s}_j = s_j],$$

where

$$s_j \in \{\textsc{Sufficient}, \textsc{Insufficient}\}.$$

This measures whether the model correctly recognizes the support boundary.

Unsupported Conclusion Rate (UCR). For incomplete-support instances,

$$\mathrm{UCR} = \frac{\#\{(A, q_k) : A \subsetneq \Gamma_k \text{ and the model asserts } q_k\}}{\#\{(A, q_k) : A \subsetneq \Gamma_k\}}.$$

Lower UCR is better. In the controlled experiments, every incomplete state has support deficit $r = 1$, so UCR directly measures the tendency to assert a conclusion after exactly one indispensable premise has been removed.

Joint interpretation of Acc and UCR. Acc and UCR should be interpreted jointly because they measure complementary behaviours at the support boundary. High Acc alone does not establish support-sensitive reasoning: a model may correctly predict q_k from complete support while continuing to assert the same conclusion after an indispensable premise is removed. Conversely, low UCR alone is insufficient, because a model that systematically withholds could avoid unsupported conclusions without successfully deriving supported ones. The desired behaviour is therefore

$$\text{high Acc} \quad \text{and} \quad \text{low UCR},$$

indicating that the model asserts q_k when $\Gamma_k \models q_k$ but withholds it when

$$\Gamma_k \setminus \{P_{k,j}\} \not\models q_k.$$

Since all insufficient-support instances in our experiments use $r = 1$, this joint criterion directly tests whether the model responds appropriately to the removal of a single indispensable premise.

Missing-support and repair metrics. Missing-support identification and support repair are evaluated using exact match (EM) and F1. EM requires the predicted target to match the gold missing or repair support exactly, while F1 measures overlap between predicted and gold support. Since the controlled experiments use $r = 1$, the gold target contains a single missing premise.

E NATURAL-LANGUAGE RENDERING

The symbolic supervision generated by Δ_1–RSG is rendered into natural-language prompts before being presented to the language model. Rendering changes the surface form of an instance but not its underlying logical structure, support state, or supervision target. Thus, each rendered instance remains traceable to its source theorem (Γ_k, q_k) and, for support-critical tasks, to the corresponding available support state $A \subseteq \Gamma_k$.

For conclusion-inference instances, the complete support set Γ_k is rendered as the available information and the model is asked to infer the supported conclusion. For support-sufficiency instances, the rendered premises and candidate conclusion are provided and the model predicts SUFFICIENT or INSUFFICIENT. Abstention instances use the same support information but require an ANSWER or WITHHOLD decision. Missing-support and repair instances ask the model to identify, respectively, the indispensable premise that is absent and the premise that should be restored.

For the controlled experiments, incomplete-support instances use $r = 1$. Consequently, the natural-language rendering of an insufficient-support instance differs from its complete-support counterpart by the removal of exactly one indispensable premise. The rendering procedure does not change the gold label implied by the underlying symbolic construction.

E.1 EXAMPLE RENDERING

The example below illustrates how a support-critical theorem and its derived support states can be rendered into natural-language-like prompts. The example is intended to make the interface between the symbolic construction and LLM supervision concrete; it illustrates the rendering procedure rather than specifying a particular surface template.

E.2 FROM A SYMBOLIC THEOREM TO TEXT

Consider the Δ_1 theorem used in Appendix B.3. Its underlying FTSC contains

$$\begin{aligned} D_1 &= x_1, \\ D_2 &= x_2 \vee \neg x_1, \\ D_3 &= x_3 \vee \neg x_1 \vee \neg x_2, \\ D_4 &= \neg x_1 \vee \neg x_2 \vee \neg x_3. \end{aligned}$$

Selecting D_4 gives

$$\Gamma = \{D_1, D_2, D_3\}, \qquad q = \neg D_4 = x_1 \wedge x_2 \wedge x_3.$$

The clauses in Γ admit an equivalent implication-style rendering:

$$\begin{aligned} D_1 &: \quad x_1 \text{ is true,} \\ D_2 &: \quad \text{if } x_1 \text{ is true, then } x_2 \text{ is true,} \\ D_3 &: \quad \text{if } x_1 \text{ and } x_2 \text{ are true, then } x_3 \text{ is true.} \end{aligned}$$

The conclusion can be rendered as

$$q : \quad x_1,\ x_2, \text{ and } x_3 \text{ are all true.}$$

These renderings preserve the logical content of the corresponding formulas; for example,

$$x_2 \vee \neg x_1 \equiv x_1 \to x_2,$$

and

$$x_3 \vee \neg x_1 \vee \neg x_2 \equiv (x_1 \wedge x_2) \to x_3.$$

E.3 COMPLETE-SUPPORT RENDERING

With the complete support state $A = \Gamma$, a conclusion-inference instance can be rendered as follows:

Premises:
1) x_1 is true.
2) If x_1 is true, then x_2 is true.
3) If x_1 and x_2 are true, then x_3 is true.

Task: Infer the conclusion supported by the premises.

Target: x_1, x_2, and x_3 are all true.

This instance corresponds to the positive support relation

$$\Gamma \models q.$$

E.4 Rendering an Immediate Support-Boundary Case

For the controlled setting considered in this paper, insufficient-support instances use deficit $r = 1$. Consider removing D_2:

$$A = \{D_1, D_3\} = \Gamma \setminus \{D_2\}.$$

By support-criticality,

$$A \not\models q,$$

and the exact missing support is

$$M(A) = \{D_2\}.$$

The same underlying support state can then be rendered into several complementary RSG tasks.

Support sufficiency.

Premises:
1) x_1 is true.
2) If x_1 and x_2 are true, then x_3 is true.

Candidate conclusion:
x_1, x_2, and x_3 are all true.

Task: Are the available premises sufficient to support the candidate conclusion?

Target: `INSUFFICIENT`

Support-based abstention.

Premises:
1) x_1 is true.
2) If x_1 and x_2 are true, then x_3 is true.

Candidate conclusion:
x_1, x_2, and x_3 are all true.

Task: Should the candidate conclusion be answered from the available premises, or should it be withheld?

Target: `WITHHOLD`

Here, `WITHHOLD` represents insufficient support rather than contradiction:

$$A \not\models q \qquad \not\Rightarrow \qquad A \models \neg q.$$

The model is therefore supervised not to replace an unsupported conclusion with its negation, but to recognize that the available evidence does not license the conclusion.

Missing-support identification.

> **Available premises:**
> 1) x_1 is true.
> 2) If x_1 and x_2 are true, then x_3 is true.
>
> **Candidate conclusion:**
> x_1, x_2, and x_3 are all true.
>
> **Task:** Identify the missing premise required to support the candidate conclusion.
>
> **Target:** If x_1 is true, then x_2 is true.

This target corresponds exactly to

$$M(A) = \{D_2\}.$$

Support repair. The same support state can be presented as a constructive repair task:

> **Available premises:**
> 1) x_1 is true.
> 2) If x_1 and x_2 are true, then x_3 is true.
>
> **Candidate conclusion:**
> x_1, x_2, and x_3 are all true.
>
> **Task:** Which premise should be added to restore sufficient support for the candidate conclusion?
>
> **Target:** If x_1 is true, then x_2 is true.

Adding the predicted premise restores the complete support state:

$$A \cup \{D_2\} = \Gamma,$$

and therefore

$$A \cup \{D_2\} \models q.$$

E.5 Preservation of the Supervision Semantics

The different textual tasks above are derived from the same underlying support-critical theorem. Natural-language rendering changes the surface representation of an instance but not its symbolic supervision target. In particular, the complete state supports q, whereas removal of the single indispensable premise D_2 produces an $r = 1$ state for which q is no longer licensed. The corresponding sufficiency, abstention, missing-support, and repair targets are therefore all determined by the same symbolic support boundary.

This example illustrates the intended interface

Δ_1 theorem $\longrightarrow$ RSG support state $\longrightarrow$ textual task instance $\longrightarrow$ structured supervision target.